PAC55XX Family User Guide

Power Application Controller®

Multi-Mode Power Manager™

Configurable Analog Front End™

Application Specific Power Drivers™

Arm® Cortex®-M4F Controller Core

Rev 2.1 – 2025-04-09

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF REGISTERS

1 OVERVIEW

This document is the PAC55XX Family User Guide. It details the operation of all of the peripherals in the PAC55XX family of controllers.

Each feature is described in detail with block diagrams and all programming registers needed for its operation. Not every peripheral is available in each PAC device, so the user should consult the device data sheet to determine which peripherals and IOs are available in your selected devices.

This document supports multiple versions of the PAC55XX MCU. Certain features and functions are not available in the first version of the MCU (MCU v1).

2 STYLE AND FORMATTING CONVENTIONS

This chapter describes the formatting and styles used throughout this document.

2.1 Number Representation

Numbers other than decimal will have a postfix indicator. All numbers use little endian formatting, with the most significant bit/digit to the left. Digits for binary and hexadecimal representation are grouped with a single space every four digits to improve readability. Binary numbers use “b” as a postfix and hexadecimal numbers use “h” as a postfix.

For example, 1011b binary = Bh hexadecimal = 11 decimal.

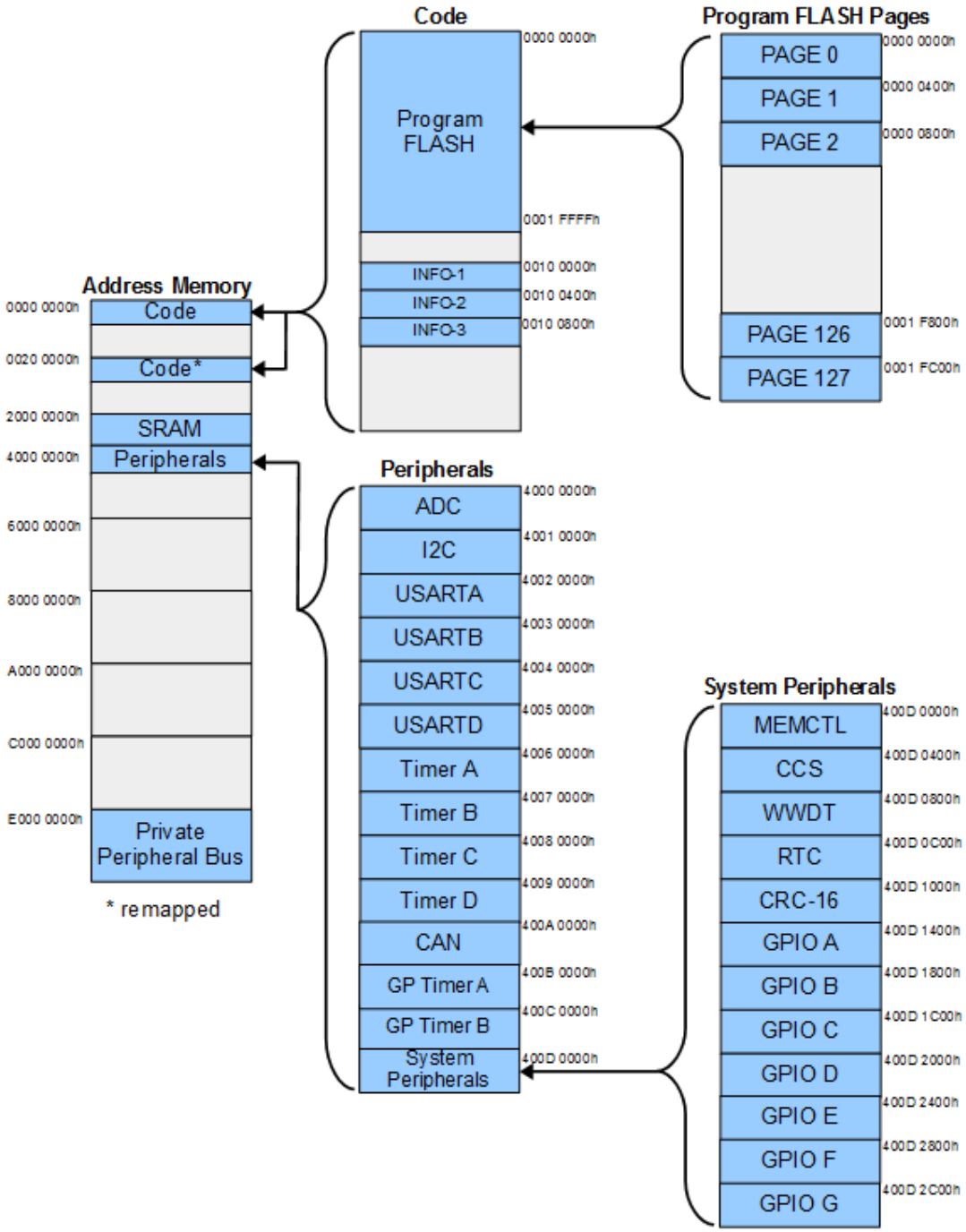
2.2 Formatting Styles

TYPE	EXAMPLE	DESCRIPTION
Register Name	**RTCCTL**	Register names use a capital letter and **boldface** type.
Register Bit(s)	**RTCCTL.RTCCLKDIV**	Register bits are always represented with the register name separated with a period.
Function selected by register bit(s)	**[RTCCTL.RTCCLKDIV]**	Within text blocks, functions selected with a register bit setting are set in brackets. For example **[RTCCTL.RTCCLKDIV]** means divider settings /2 to /65536.
Pin Function	PA5	Pin functions use capital letters
Internal signals	*PWMA3*	Internal signals use *italicized* font.
Formulas	`CLK = FCLK / DIV`	Formulas use `monospaced` text.
Links	Link	Hyperlinks are underlined and blue.
CPU Mnemonic	`MRS`	CPU Mnemonic uses `monospaced` text.
Operands	*`{Rd, }, Rn, Rm`*	Operands use *`monospaced italic`* text.
Code examples	`b loopA`	Code examples use monospaced text.

3 MEMORY AND REGISTER MAP

3.1 Memory Map

Figure 3-1 Memory Map



3.2 FLASH Memory Map

Table 3-1 Embedded FLASH Register Map

ADDRESS	NAME	DESCRIPTION
0000 0000h	EFLASHP0	EFLASH page 0
0000 0400h	EFLASHP1	EFLASH page 1
0000 0800h	EFLASHP2	EFLASH page 2
0000 0C00h	EFLASHP3	EFLASH page 3
0000 1000h	EFLASHP4	EFLASH page 4
0000 1400h	EFLASHP5	EFLASH page 5
0000 1800h	EFLASHP6	EFLASH page 6
0000 1C00h	EFLASHP7	EFLASH page 7
0000 2000h	EFLASHP8	EFLASH page 8
0000 2400h	EFLASHP9	EFLASH page 9
0000 2800h	EFLASHP10	EFLASH page 10
0000 2C00h	EFLASHP11	EFLASH page 11
0000 3000h	EFLASHP12	EFLASH page 12
0000 3400h	EFLASHP13	EFLASH page 13
0000 3800h	EFLASHP14	EFLASH page 14
0000 3C00h	EFLASHP15	EFLASH page 15
0000 4000h	EFLASHP16	EFLASH page 16
0000 4400h	EFLASHP17	EFLASH page 17
0000 4800h	EFLASHP18	EFLASH page 18
0000 4C00h	EFLASHP19	EFLASH page 19
0000 5000h	EFLASHP20	EFLASH page 20
0000 5400h	EFLASHP21	EFLASH page 21
0000 5800h	EFLASHP22	EFLASH page 22
0000 5C00h	EFLASHP23	EFLASH page 23
0000 6000h	EFLASHP24	EFLASH page 24
0000 6400h	EFLASHP25	EFLASH page 25
0000 6800h	EFLASHP26	EFLASH page 26
0000 6C00h	EFLASHP27	EFLASH page 27
0000 7000h	EFLASHP28	EFLASH page 28
0000 7400h	EFLASHP29	EFLASH page 29
0000 7800h	EFLASHP30	EFLASH page 30

0000 7C00h	EFLASHP31	EFLASH page 31
0000 8000h	EFLASHP32	EFLASH page 32
0000 8400h	EFLASHP33	EFLASH page 33
0000 8800h	EFLASHP34	EFLASH page 34
0000 8C00h	EFLASHP35	EFLASH page 35
0000 9000h	EFLASHP36	EFLASH page 36
0000 9400h	EFLASHP37	EFLASH page 37
0000 9800h	EFLASHP38	EFLASH page 38
0000 9C00h	EFLASHP39	EFLASH page 39
0000 A000h	EFLASHP40	EFLASH page 40
0000 A400h	EFLASHP41	EFLASH page 41
0000 A800h	EFLASHP42	EFLASH page 42
0000 AC00h	EFLASHP43	EFLASH page 43
0000 B000h	EFLASHP44	EFLASH page 44
0000 B400h	EFLASHP45	EFLASH page 45
0000 B800h	EFLASHP46	EFLASH page 46
0000 BC00h	EFLASHP47	EFLASH page 47
0000 C000h	EFLASHP48	EFLASH page 48
0000 C400h	EFLASHP49	EFLASH page 49
0000 C800h	EFLASHP50	EFLASH page 50
0000 CC00h	EFLASHP51	EFLASH page 51
0000 D000h	EFLASHP52	EFLASH page 52
0000 D400h	EFLASHP53	EFLASH page 53
0000 D800h	EFLASHP54	EFLASH page 54
0000 DC00h	EFLASHP55	EFLASH page 55
0000 E000h	EFLASHP56	EFLASH page 56
0000 E400h	EFLASHP57	EFLASH page 57
0000 E800h	EFLASHP58	EFLASH page 58
0000 EC00h	EFLASHP59	EFLASH page 59
0000 F000h	EFLASHP60	EFLASH page 60
0000 F400h	EFLASHP61	EFLASH page 61
0000 F800h	EFLASHP62	EFLASH page 62
0000 FC00h	EFLASHP63	EFLASH page 63
0001 0000h	EFLASHP64	EFLASH page 64

0001 0400h	EFLASHP65	EFLASH page 65
0001 0800h	EFLASHP66	EFLASH page 66
0001 0C00h	EFLASHP67	EFLASH page 67
0001 1000h	EFLASHP68	EFLASH page 68
0001 1400h	EFLASHP69	EFLASH page 69
0001 1800h	EFLASHP70	EFLASH page 70
0001 1C00h	EFLASHP71	EFLASH page 71
0001 2000h	EFLASHP72	EFLASH page 72
0001 2400h	EFLASHP73	EFLASH page 73
0001 2800h	EFLASHP74	EFLASH page 74
0001 2C00h	EFLASHP75	EFLASH page 75
0001 3000h	EFLASHP76	EFLASH page 76
0001 3400h	EFLASHP77	EFLASH page 77
0001 3800h	EFLASHP78	EFLASH page 78
0001 3C00h	EFLASHP79	EFLASH page 79
0001 4000h	EFLASHP80	EFLASH page 80
0001 4400h	EFLASHP81	EFLASH page 81
0001 4800h	EFLASHP82	EFLASH page 82
0001 4C00h	EFLASHP83	EFLASH page 83
0001 5000h	EFLASHP84	EFLASH page 84
0001 5400h	EFLASHP85	EFLASH page 85
0001 5800h	EFLASHP86	EFLASH page 86
0001 5C00h	EFLASHP87	EFLASH page 87
0001 6000h	EFLASHP88	EFLASH page 88
0001 6400h	EFLASHP89	EFLASH page 89
0001 6800h	EFLASHP90	EFLASH page 90
0001 6C00h	EFLASHP91	EFLASH page 91
0001 7000h	EFLASHP92	EFLASH page 92
0001 7400h	EFLASHP93	EFLASH page 93
0001 7800h	EFLASHP94	EFLASH page 94
0001 7C00h	EFLASHP95	EFLASH page 95
0001 8000h	EFLASHP96	EFLASH page 96
0001 8400h	EFLASHP97	EFLASH page 97
0001 8800h	EFLASHP98	EFLASH page 98

0001 8C00h	EFLASHP99	EFLASH page 99
0001 9000h	EFLASHP100	EFLASH page 100
0001 9400h	EFLASHP101	EFLASH page 101
0001 9800h	EFLASHP102	EFLASH page 102
0001 9C00h	EFLASHP103	EFLASH page 103
0001 A000h	EFLASHP104	EFLASH page 104
0001 A400h	EFLASHP105	EFLASH page 105
0001 A800h	EFLASHP106	EFLASH page 106
0001 AC00h	EFLASHP107	EFLASH page 107
0001 B000h	EFLASHP108	EFLASH page 108
0001 B400h	EFLASHP109	EFLASH page 109
0001 B800h	EFLASHP110	EFLASH page 110
0001 BC00h	EFLASHP111	EFLASH page 111
0001 C000h	EFLASHP112	EFLASH page 112
0001 C400h	EFLASHP113	EFLASH page 113
0001 C800h	EFLASHP114	EFLASH page 114
0001 CC00h	EFLASHP115	EFLASH page 115
0001 D000h	EFLASHP116	EFLASH page 116
0001 D400h	EFLASHP117	EFLASH page 117
0001 D800h	EFLASHP118	EFLASH page 118
0001 DC00h	EFLASHP129	EFLASH page 119
0001 E000h	EFLASHP120	EFLASH page 120
0001 E400h	EFLASHP121	EFLASH page 121
0001 E800h	EFLASHP122	EFLASH page 122
0001 EC00h	EFLASHP123	EFLASH page 123
0001 F000h	EFLASHP124	EFLASH page 124
0001 F400h	EFLASHP125	EFLASH page 125
0001 F800h	EFLASHP126	EFLASH page 126
0001 FC00h	EFLASHP127	EFLASH page 127

3.3 INFO-1 Register Map

Table 3-2 INFO-1 Register Map

ADDRESS	OFFSET			
	3	2	1	0
0010 0000h	UNIQUEID			
0010 0004h				
0010 0008h				
0010 000Ch – 0010 03FFh	RESERVED			

3.4 INFO-2 Register Map

Table 3-3 INFO-2 Register Map

ADDRESS	OFFSET			
	3	2	1	0
0010 0400h	Reserved			
0010 0404h	Reserved			
0010 0408h	PACIDR			
0010 040Ch	Reserved			
0010 0410h	VMS200		VMS100	
0010 0414h	ADCOFFSET			
0010 0418h	ADCGAIN			
0010 041Ch	FTTEMP		TEMPS	
0010 0420h	Reserved			SECEN
0010 0424h	MAXADDR			
0010 0428h	ROSC			
0010 042Ch	CLKREF			
0010 0430h	Reserved			
0010 0434h	VREF		Reserved	
0010 0438h	Reserved		VPTAT	
0010 0430h-0010 043Bh	Reserved			
0010 043Ch	SWDFUSE			
0010 0440h – 0010 045Fh	Reserved			
0010 0460h	CRC1_CLONE		CRC1	
0010 0464h	CRC2_CLONE		CRC2	
0010 0468h – 0010 07FFh				

Table 3-4 INFO-1, INFO-2 Field Name and Description

NAME	DESCRIPTION
UNIQUEID	96-bit unique device ID.
PACIDR	Device part number and revision.
VMS100	VMS voltage when VM = 100V. See Device User Guide for information on how to convert to °C.
VMS200	VMS voltage when VM = 200V. See Device User Guide for information on how to convert to °C.
ADCOFFSET	ADC offset in mV.
ADCGAIN	ADC gain in mV.
TEMPS	Temperature sensor ADC counts test temperature in FTTEMP.
FTTEMP	Test temperature for internal temperature sensor in °C.
SECEN	Code Security Enabled
MAXADDR	Maximum FLASH address
ROSC	ROSC frequency measured in Hz / 10 (nominal: 20MHz / 10).
CLKREF	CLKREF frequency measured in Hz (nominal: 4MHz).
VREF	Voltage reference measured in mV.
VPTAT	Internal temperature sensor measured in mV.
SWDFUSE	SWD Fuse Key Value.
CRC1	16-bit CRC calculated over all INFO-1.
CRC1_CLONE	Clone of **INFO2.CRC1** field.
CRC2	16-bit CRC calculated over range 0010 0400h to 0010 045Fh.
CRC2_CLONE	Clone of **INFO2.CRC2** field.

3.5 INFO-3 Register Map

Table 3-5 INFO-3 Register Map

ADDRESS	CONTENTS
0010 0800h – 0010 0BFFh	User Defined[1]

3.6 Peripheral Register Maps

Table 3-6 ADC Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
EMUXCTL	4000 0000h	EMUX Control	RW	0000 0000h
EMUXDATA	4000 0004h	EMUX Data	RW	0000 0000h
ADCCTL	4000 0008h	ADC Control	RW	0000 0000h
ADCRES	4000 000Ch	ADC Result	RW	0000 0000h
ADCINT	4000 0010h	ADC Interrupt Control	RW	0000 0000h
DTSETRIGENT0TO3	4000 0040h	DTSE Trigger Entry 0 to 3	RW	0000 0000h
DTSETRIGENT4TO7	4000 0044h	DTSE Trigger Entry 4 to 7	RW	0000 0000h
DTSETRIGENT8TO11	4000 0048h	DTSE Trigger Entry 8 to 11	RW	0000 0000h
DTSETRIGENT12TO15	4000 004Ch	DTSE Trigger Entry 12 to 15	RW	0000 0000h
DTSETRIGENT16TO19	4000 0050h	DTSE Trigger Entry 16 to 19	RW	0000 0000h
DTSETRIGENT20TO23	4000 0054h	DTSE Trigger Entry 20 to 23	RW	0000 0000h
DTSETRIGENT24TO27	4000 0058h	DTSE Trigger Entry 24 to 27	RW	0000 0000h
DTSETRIGENT28TO31	4000 005Ch	DTSE Trigger Entry 28 to 31	RW	0000 0000h
DTSESEQCFG0	4000 0080h	DTSE Sequence Config 0	RW	0000 0000h
DTSESEQCFG1	4000 0084h	DTSE Sequence Config 1	RW	0000 0000h
DTSESEQCFG2	4000 0088h	DTSE Sequence Config 2	RW	0000 0000h
DTSESEQCFG3	4000 008Ch	DTSE Sequence Config 3	RW	0000 0000h
DTSESEQCFG4	4000 0090h	DTSE Sequence Config 4	RW	0000 0000h
DTSESEQCFG5	4000 0094h	DTSE Sequence Config 5	RW	0000 0000h
DTSESEQCFG6	4000 0098h	DTSE Sequence Config 6	RW	0000 0000h
DTSESEQCFG7	4000 009Ch	DTSE Sequence Config 7	RW	0000 0000h

[1] If code security is enabled, then INFO-3 is reserved for code security functions. It may be erased by the memory controller during operation. See section below on Code Protection for more details.

DTSESEQCFG8	4000 00A0h	DTSE Sequence Config 8	RW	0000 0000h
DTSESEQCFG9	4000 00A4h	DTSE Sequence Config 9	RW	0000 0000h
DTSESEQCFG10	4000 00A8h	DTSE Sequence Config 10	RW	0000 0000h
DTSESEQCFG11	4000 00ACh	DTSE Sequence Config 11	RW	0000 0000h
DTSESEQCFG12	4000 00B0h	DTSE Sequence Config 12	RW	0000 0000h
DTSESEQCFG13	4000 00B4h	DTSE Sequence Config 13	RW	0000 0000h
DTSESEQCFG14	4000 00B8h	DTSE Sequence Config 14	RW	0000 0000h
DTSESEQCFG15	4000 00BCh	DTSE Sequence Config 15	RW	0000 0000h
DTSESEQCFG16	4000 00C0h	DTSE Sequence Config 16	RW	0000 0000h
DTSESEQCFG17	4000 00C4h	DTSE Sequence Config 17	RW	0000 0000h
DTSESEQCFG18	4000 00C8h	DTSE Sequence Config 18	RW	0000 0000h
DTSESEQCFG19	4000 00CCh	DTSE Sequence Config 19	RW	0000 0000h
DTSESEQCFG20	4000 00D0h	DTSE Sequence Config 20	RW	0000 0000h
DTSESEQCFG21	4000 00D4h	DTSE Sequence Config 21	RW	0000 0000h
DTSESEQCFG22	4000 00D8h	DTSE Sequence Config 22	RW	0000 0000h
DTSESEQCFG23	4000 00DCh	DTSE Sequence Config 23	RW	0000 0000h
DTSERES0	4000 0100h	DTSE Result 0	RO	0000 0000h
DTSERES1	4000 0104h	DTSE Result 1	RO	0000 0000h
DTSERES2	4000 0108h	DTSE Result 2	RO	0000 0000h
DTSERES3	4000 010Ch	DTSE Result 3	RO	0000 0000h
DTSERES4	4000 0110h	DTSE Result 4	RO	0000 0000h
DTSERES5	4000 0114h	DTSE Result 5	RO	0000 0000h
DTSERES6	4000 0118h	DTSE Result 6	RO	0000 0000h
DTSERES7	4000 011Ch	DTSE Result 7	RO	0000 0000h
DTSERES8	4000 0120h	DTSE Result 8	RO	0000 0000h
DTSERES9	4000 0124h	DTSE Result 9	RO	0000 0000h
DTSERES10	4000 0128h	DTSE Result 10	RO	0000 0000h
DTSERES11	4000 012Ch	DTSE Result 11	RO	0000 0000h
DTSERES12	4000 0130h	DTSE Result 12	RO	0000 0000h
DTSERES13	4000 0134h	DTSE Result 13	RO	0000 0000h
DTSERES14	4000 0138h	DTSE Result 14	RO	0000 0000h
DTSERES15	4000 013Ch	DTSE Result 15	RO	0000 0000h
DTSERES16	4000 0140h	DTSE Result 16	RO	0000 0000h
DTSERES17	4000 0144h	DTSE Result 17	RO	0000 0000h

DTSERES18	4000 0148h	DTSE Result 18	RO	0000 0000h
DTSERES19	4000 014Ch	DTSE Result 19	RO	0000 0000h
DTSERES20	4000 0150h	DTSE Result 20	RO	0000 0000h
DTSERES21	4000 0154h	DTSE Result 21	RO	0000 0000h
DTSERES22	4000 0158h	DTSE Result 22	RO	0000 0000h
DTSERES23	4000 015Ch	DTSE Result 23	RO	0000 0000h

Table 3-7 I2C Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
I2CCONSET	4001 0000h	I2C Control Set Register	RW	0000 0000h
I2CCONCLR	4001 0004h	I2C Control Clear Register	WO	--
I2CSTAT	4001 0008h	I2C Status Register	RO	0000 00F8h
I2CDAT	4001 000Ch	I2C Data Register	RW	0000 0000h
I2CCLK	4001 0010h	I2C Clock Control Register	RW	0000 0000h
I2CADR0	4001 0014h	I2C Slave Address Register 0	RW	0000 0000h
I2CADRM0	4001 0018h	I2C Slave Address Mask Register 0	RW	0000 00FEh
I2CXADR0	4001 001C	I2C Extended Slave Address Register 0	RW	0000 0000h
I2CXADRM0	4001 0020h	I2C Extended Slave Address Mask Register 0	RW	0000 07FEh
I2CRST	4001 0024h	I2C Software Reset Register	RW	0000 0000h
I2CADR1	4001 0028h	I2C Slave Address Register 1	RW	0000 0000h
I2CADRM1	4001 002Ch	I2C Slave Address Mask Register 1	RW	0000 00FEh
I2CADR2	4001 0030h	I2C Slave Address Register 2	RW	0000 0000h
I2CADRM2	4001 0034h	I2C Slave Address Mask Register 2	RW	0000 00FEh
I2CADR3	4001 0038h	I2C Slave Address Register 3	RW	0000 0000h
I2CADRM3	4001 003Ch	I2C Slave Address Mask Register 3	RW	0000 00FEh

Table 3-8 USARTA Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
UARTARBR	4002 0000h	UART A Receive Buffer Register	RO	--
UARTATHR	4002 0004h	UART A Transmit Holding Register	WO	--
UARTADLR	4002 0008h	UART A Divisor Latch Register	RW	0000 0001h
UARTAIER	4002 000Ch	UART A Interrupt Enable Register	RW	0000 0000h
UARTAIIR	4002 0010h	UART A Interrupt Identification Register	RO	0000 0001h
UARTAFCR	4002 0014h	UART A FIFO Control Register	RW	0000 0000h
UARTALCR	4002 0018h	UART A Line control Register	RW	0000 0000h
UARTALSR	4002 0020h	UART A Line Status Register	RO	0000 0060h
UARTASCR	4002 0028h	UART A Scratch Pad Register	RW	--
UARTAEFR	4002 002Ch	UART A Enhanced Features Register	RW	0000 0000h
SSPACON	4002 0000h	SSP A Control Register	RW	0000 0000h
SSPASTAT	4002 0004h	SSP A Status Register	RO	0000 0003h
SSPADAT	4002 0008h	SSP A Data Register	RW	0000 0000h
SSPACLK	4002 000Ch	SSP A Clock Control Register	RW	0000 0000h
SSPAIMSC	4002 0010h	SSP A Interrupt Mask Set and Clear Register	RW	0000 0000h
SSPARIS	4002 0014h	SSP A Raw Interrupt Status Register	RO	0000 0000h
SSPAMIS	4002 0018h	SSP A Masked Interrupt Status Register	RO	0000 0000h
SSPAICLR	4002 001Ch	SSP A Interrupt Clear Register	RW	0000 0000h
SSPASSCR	4002 0028h	SSP A Slave Select Configuration Register	RW	0000 0000h

Table 3-9 USARTB Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
UARTBRBR	4003 0000h	UART B Receive Buffer Register	RO	--
UARTBTHR	4003 0004h	UART B Transmit Holding Register	WO	--
UARTBDLR	4003 0008h	UART B Divisor Latch Register	RW	0000 0001h
UARTBIER	4003 000Ch	UART B Interrupt Enable Register	RW	0000 0000h
UARTBIIR	4003 0010h	UART B Interrupt Identification Register	RO	0000 0001h
UARTBFCR	4003 0014h	UART B FIFO Control Register	RW	0000 0000h
UARTBLCR	4003 0018h	UART B Line control Register	RW	0000 0000h
UARTBLSR	4003 0020h	UART B Line Status Register	RO	0000 0060h
UARTBSCR	4003 0028h	UART B Scratch Pad Register	RW	--
UARTBEFR	4003 002Ch	UART B Enhanced Features Register	RW	0000 0000h
SSPBCON	4003 0000h	SSP B Control Register	RW	0000 0000h
SSPBSTAT	4003 0004h	SSP B Status Register	RO	0000 0003h
SSPBDAT	4003 0008h	SSP B Data Register	RW	0000 0000h
SSPBCLK	4003 000Ch	SSP B Clock Control Register	RW	0000 0000h
SSPBIMSC	4003 0010h	SSP B Interrupt Mask Set and Clear Register	RW	0000 0000h
SSPBRIS	4003 0014h	SSP B Raw Interrupt Status Register	RO	0000 0000h
SSPBMIS	4003 0018h	SSP B Masked Interrupt Status Register	RO	0000 0000h
SSPBICLR	4003 001Ch	SSP B Interrupt Clear Register	RW	0000 0000h
SSPBSSCR	4003 0028h	SSP B Slave Select Configuration Register	RW	0000 0000h

Table 3-10 USARTC Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
UARTCRBR	4004 0000h	UART C Receive Buffer Register	RO	--
UARTCTHR	4004 0004h	UART C Transmit Holding Register	WO	--
UARTCDLR	4004 0008h	UART C Divisor Latch Register	RW	0000 0001h
UARTCIER	4004 000Ch	UART C Interrupt Enable Register	RW	0000 0000h
UARTCIIR	4004 0010h	UART C Interrupt Identification Register	RO	0000 0001h
UARTCFCR	4004 0014h	UART C FIFO Control Register	RW	0000 0000h
UARTCLCR	4004 0018h	UART C Line control Register	RW	0000 0000h
UARTCLSR	4004 0020h	UART C Line Status Register	RO	0000 0060h
UARTCSCR	4004 0028h	UART C Scratch Pad Register	RW	--
UARTCEFR	4004 002Ch	UART C Enhanced Features Register	RW	0000 0000h
SSPCCON	4004 0000h	SSP C Control Register	RW	0000 0000h
SSPCSTAT	4004 0004h	SSP C Status Register	RO	0000 0003h
SSPCDAT	4004 0008h	SSP C Data Register	RW	0000 0000h
SSPCCLK	4004 000Ch	SSP C Clock Control Register	RW	0000 0000h
SSPCIMSC	4004 0010h	SSP C Interrupt Mask Set and Clear Register	RW	0000 0000h
SSPCRIS	4004 0014h	SSP C Raw Interrupt Status Register	RO	0000 0000h
SSPCMIS	4004 0018h	SSP C Masked Interrupt Status Register	RO	0000 0000h
SSPCICLR	4004 001Ch	SSP C Interrupt Clear Register	RW	0000 0000h
SSPCSSCR	4004 0028h	SSP C Slave Select Configuration Register	RW	0000 0000h

Table 3-11 USARTD Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
UARTDRBR	4005 0000h	UART D Receive Buffer Register	RO	--
UARTDTHR	4005 0004h	UART D Transmit Holding Register	WO	--
UARTDDLR	4005 0008h	UART D Divisor Latch Register	RW	0000 0001h
UARTDIER	4005 000Ch	UART D Interrupt Enable Register	RW	0000 0000h
UARTDIIR	4005 0010h	UART D Interrupt Identification Register	RO	0000 0001h
UARTDFCR	4005 0014h	UART D FIFO Control Register	RW	0000 0000h
UARTDLCR	4005 0018h	UART D Line control Register	RW	0000 0000h
UARTDLSR	4005 0020h	UART D Line Status Register	RO	0000 0060h
UARTDSCR	4005 0028h	UART D Scratch Pad Register	RW	--
UARTDEFR	4005 002Ch	UART D Enhanced Features Register	RW	0000 0000h
SSPDCON	4005 0000h	SSP D Control Register	RW	0000 0000h
SSPDSTAT	4005 0004h	SSP D Status Register	RO	0000 0003h
SSPDDAT	4005 0008h	SSP D Data Register	RW	0000 0000h
SSPDCLK	4005 000Ch	SSP D Clock Control Register	RW	0000 0000h
SSPDIMSC	4005 0010h	SSP D Interrupt Mask Set and Clear Register	RW	0000 0000h
SSPDRIS	4005 0014h	SSP D Raw Interrupt Status Register	RO	0000 0000h
SSPDMIS	4005 0018h	SSP D Masked Interrupt Status Register	RO	0000 0000h
SSPDICLR	4005 001Ch	SSP D Interrupt Clear Register	RW	0000 0000h
SSPDSSCR	4005 0028h	SSP D Slave Select Configuration Register	RW	0000 0000h

Table 3-12 Timer A Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
TACTL	4006 0000h	Timer A Control	RW	0000 0000h
TAINT	4006 0004h	Timer A Interrupt Control	RW	0000 0000h
TAPRD	4006 0008h	Timer A Period	RW	0000 0000h
TACTR	4006 000Ch	Timer A Counter	RW	0000 0000h
TAQEPCTL	4006 0010h	Timer A QEP Control	RW	0000 0000h
TACCTL0	4006 0100h	Timer A CC Control 0	RW	0000 0000h
TACCTR0	4006 0104h	Timer A CC Counter 0	RW	0000 0000h
TACCTL1	4006 0108h	Timer A CC Control 1	RW	0000 0000h
TACCTR1	4006 010Ch	Timer A CC Counter 1	RW	0000 0000h
TACCTL2	4006 0110h	Timer A CC Control 2	RW	0000 0000h
TACCTR2	4006 0114h	Timer A CC Counter 2	RW	0000 0000h
TACCTL3	4006 0118h	Timer A CC Control 3	RW	0000 0000h
TACCTR3	4006 011Ch	Timer A CC Counter 3	RW	0000 0000h
TACCTL4	4006 0120h	Timer A CC Control 4	RW	0000 0000h
TACCTR4	4006 0124h	Timer A CC Counter 4	RW	0000 0000h
TACCTL5	4006 0128h	Timer A CC Control 5	RW	0000 0000h
TACCTR5	4006 012Ch	Timer A CC Counter 5	RW	0000 0000h
TACCTL6	4006 0130h	Timer A CC Control 6	RW	0000 0000h
TACCTR6	4006 0134h	Timer A CC Counter 6	RW	0000 0000h
TACCTL7	4006 0138h	Timer A CC Control 7	RW	0000 0000h
TACCTR7	4006 013Ch	Timer A CC Counter 7	RW	0000 0000h
TADTGCTL0	4006 0200h	Timer A DTG Control 0	RW	0000 0000h
TADTGCTL1	4006 0204h	Timer A DTG Control 1	RW	0000 0000h
TADTGCTL2	4006 0208h	Timer A DTG Control 2	RW	0000 0000h
TADTGCTL3	4006 020Ch	Timer A DTG Control 3	RW	0000 0000h

Table 3-13 Timer B Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
TBCTL	4007 0000h	Timer B Control	RW	0000 0000h
TBINT	4007 0004h	Timer B Interrupt Control	RW	0000 0000h
TBPRD	4007 0008h	Timer B Period	RW	0000 0000h
TBCTR	4007 000Ch	Timer B Counter	RW	0000 0000h
TBQEPCTL	4007 0010h	Timer B QEP Control	RW	0000 0000h
TBCCTL0	4007 0100h	Timer B CC Control 0	RW	0000 0000h
TBCCTR0	4007 0104h	Timer B CC Counter 0	RW	0000 0000h
TBCCTL1	4007 0108h	Timer B CC Control 1	RW	0000 0000h
TBCCTR1	4007 010Ch	Timer B CC Counter 1	RW	0000 0000h
TBCCTL2	4007 0110h	Timer B CC Control 2	RW	0000 0000h
TBCCTR2	4007 0114h	Timer B CC Counter 2	RW	0000 0000h
TBCCTL3	4007 0118h	Timer B CC Control 3	RW	0000 0000h
TBCCTR3	4007 011Ch	Timer B CC Counter 3	RW	0000 0000h
TBCCTL4	4007 0120h	Timer B CC Control 4	RW	0000 0000h
TBCCTR4	4007 0124h	Timer B CC Counter 4	RW	0000 0000h
TBCCTL5	4007 0128h	Timer B CC Control 5	RW	0000 0000h
TBCCTR5	4007 012Ch	Timer B CC Counter 5	RW	0000 0000h
TBCCTL6	4007 0130h	Timer B CC Control 6	RW	0000 0000h
TBCCTR6	4007 0134h	Timer B CC Counter 6	RW	0000 0000h
TBCCTL7	4007 0138h	Timer B CC Control 7	RW	0000 0000h
TBCCTR7	4007 013Ch	Timer B CC Counter 7	RW	0000 0000h
TBDTGCTL0	4007 0200h	Timer B DTG Control 0	RW	0000 0000h
TBDTGCTL1	4007 0204h	Timer B DTG Control 1	RW	0000 0000h
TBDTGCTL2	4007 0208h	Timer B DTG Control 2	RW	0000 0000h
TBDTGCTL3	4007 020Ch	Timer B DTG Control 3	RW	0000 0000h

Table 3-14 Timer C Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
TCCTL	4008 0000h	Timer C Control	RW	0000 0000h
TCINT	4008 0004h	Timer C Interrupt Control	RW	0000 0000h
TCPRD	4008 0008h	Timer C Period	RW	0000 0000h
TCCTR	4008 000Ch	Timer C Counter	RW	0000 0000h
TCQEPCTL	4008 0010h	Timer C QEP Control	RW	0000 0000h
TCCCTL0	4008 0100h	Timer C CC Control 0	RW	0000 0000h
TCCCTR0	4008 0104h	Timer C CC Counter 0	RW	0000 0000h
TCCCTL1	4008 0108h	Timer C CC Control 1	RW	0000 0000h
TCCCTR1	4008 010Ch	Timer C CC Counter 1	RW	0000 0000h
TCCCTL2	4008 0110h	Timer C CC Control 2	RW	0000 0000h
TCCCTR2	4008 0114h	Timer C CC Counter 2	RW	0000 0000h
TCCCTL3	4008 0118h	Timer C CC Control 3	RW	0000 0000h
TCCCTR3	4008 011Ch	Timer C CC Counter 3	RW	0000 0000h
TCCCTL4	4008 0120h	Timer C CC Control 4	RW	0000 0000h
TCCCTR4	4008 0124h	Timer C CC Counter 4	RW	0000 0000h
TCCCTL5	4008 0128h	Timer C CC Control 5	RW	0000 0000h
TCCCTR5	4008 012Ch	Timer C CC Counter 5	RW	0000 0000h
TCCCTL6	4008 0130h	Timer C CC Control 6	RW	0000 0000h
TCCCTR6	4008 0134h	Timer C CC Counter 6	RW	0000 0000h
TCCCTL7	4008 0138h	Timer C CC Control 7	RW	0000 0000h
TCCCTR7	4008 013Ch	Timer C CC Counter 7	RW	0000 0000h
TCDTGCTL0	4008 0200h	Timer C DTG Control 0	RW	0000 0000h
TCDTGCTL1	4008 0204h	Timer C DTG Control 1	RW	0000 0000h
TCDTGCTL2	4008 0208h	Timer C DTG Control 2	RW	0000 0000h
TCDTGCTL3	4008 020Ch	Timer C DTG Control 3	RW	0000 0000h

Table 3-15 Timer D Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
TDCTL	4009 0000h	Timer D Control	RW	0000 0000h
TDINT	4009 0004h	Timer D Interrupt Control	RW	0000 0000h
TDPRD	4009 0008h	Timer D Period	RW	0000 0000h
TDCTR	4009 000Ch	Timer D Counter	RW	0000 0000h
TDQEPCTL	4009 0010h	Timer D QEP Control	RW	0000 0000h
TDCCTL0	4009 0100h	Timer D CC Control 0	RW	0000 0000h
TDCCTR0	4009 0104h	Timer D CC Counter 0	RW	0000 0000h
TDCCTL1	4009 0108h	Timer D CC Control 1	RW	0000 0000h
TDCCTR1	4009 010Ch	Timer D CC Counter 1	RW	0000 0000h
TDCCTL2	4009 0110h	Timer D CC Control 2	RW	0000 0000h
TDCCTR2	4009 0114h	Timer D CC Counter 2	RW	0000 0000h
TDCCTL3	4009 0118h	Timer D CC Control 3	RW	0000 0000h
TDCCTR3	4009 011Ch	Timer D CC Counter 3	RW	0000 0000h
TDCCTL4	4009 0120h	Timer D CC Control 4	RW	0000 0000h
TDCCTR4	4009 0124h	Timer D CC Counter 4	RW	0000 0000h
TDCCTL5	4009 0128h	Timer D CC Control 5	RW	0000 0000h
TDCCTR5	4009 012Ch	Timer D CC Counter 5	RW	0000 0000h
TDCCTL6	4009 0130h	Timer D CC Control 6	RW	0000 0000h
TDCCTR6	4009 0134h	Timer D CC Counter 6	RW	0000 0000h
TDCCTL7	4009 0138h	Timer D CC Control 7	RW	0000 0000h
TDCCTR7	4009 013Ch	Timer D CC Counter 7	RW	0000 0000h
TDDTGCTL0	4009 0200h	Timer D DTG Control 0	RW	0000 0000h
TDDTGCTL1	4009 0204h	Timer D DTG Control 1	RW	0000 0000h
TDDTGCTL2	4009 0208h	Timer D DTG Control 2	RW	0000 0000h
TDDTGCTL3	4009 020Ch	Timer D DTG Control 3	RW	0000 0000h

Table 3-16 CAN Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
CANMR	400A 0000h	CAN Mode	RW	04h
CANCMR	400A 0001h	CAN Command	RW	00h
CANSR	400A 0002h	CAN Status Register	RO	00h
CANISR	400A 0003h	CAN Interrupt Status/ACK Register	RW	00h
CANIMR	400A 0004h	CAN Interrupt Mask Register	RW	00h
CANRMC	400A 0005h	CAN Receive Message Counter	RO	00h
CANBTR0	400A 0006h	CAN Bus Timing 0 Register	RW	00h
CANBTR1	400A 0007h	CAN Bus Timing 1 Register	RW	00h
CANTXBUF	400A 0008h	CAN Transmit Buffer Register	RW	00000000h
CANRXBUF	400A 000Ch	CAN Receive Buffer Register	RO	00000000h
CANACR	400A 0010h	CAN Acceptance Code Register	RW	00000000h
CANAMR	400A 0014h	CAN Acceptance Mask Register	RW	00000000h
CANECC	400A 0018h	CAN Error Code Capture Register	RO	00h
CANRXERR	400A 0019h	CAN RX Error Counter Register	RO	00h
CANTXERR	400A 001Ah	CAN TX Error Counter Register	RO	00h
CANALC	400A 001Bh	CAN Arbitration Lost Code Capture Register	RO	00h

Table 3-17 GP Timer A Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPTACTL	400B 0000h	GPTimer A Control	RW	000F FF00h
GPTACTR	400B 0004h	GPTimer A Counter	RO	00FF FFFFh

Table 3-18 GP Timer B Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPTBCTL	400C 0000h	GPTimer B Control	RW	000F FF00h
GPTBCTR	400C 0004h	GPTimer B Counter	RO	00FF FFFFh

Table 3-19 Memory Controller Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
MEMCTL	400D 0000h	Memory Controller Configuration	RW	0032 0000h
MEMSTATUS	400D 0004h	Memory Controller Status	RW	0000 0000h
FLASHLOCK	400D 0008h	FLASH Lock Access	RW	0000 0000h
FLASHPAGE	400D 000Ch	FLASH Page	RW	0000 0000h
SWDUNLOCK	400D 0010h	SWD Unlock	RW	0000 0000h
FLASHERASE	400D 0020h	FLASH Erase	RW	0000 0000h

Table 3-20 System and Clock Control (SCC) Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
CCSCTL	400D 0400h	CCS Control	RW	0000 F004h
CCSPLLCTL	400D 0404h	PLL Control	RW	0000 0000h
CCSROSCTRIM	400D 0408h	ROSC Trim Control	RW	0000 007Fh
PAMUXSEL	400D 040Ch	PA Peripheral MUX Select	RW	0011 1110h
PBMUXSEL	400D 0410h	PB Peripheral MUX Select	RW	0000 0000h
PCMUXSEL	400D 0414h	PC Peripheral MUX Select	RW	0000 0000h
PDMUXSEL	400D 0418h	PD Peripheral MUX Select	RW	0000 0000h
PEMUXSEL	400D 041Ch	PE Peripheral MUX Select	RW	0000 0000h
PFMUXSEL	400D 0420h	PF Peripheral MUX Select	RW	0000 3333h
PGMUXSEL	400D 0424h	PG Peripheral MUX Select	RW	0000 0000h
PAPUEN	400D 0428h	PA Weak Pull-up Enable	RW	0000 0000h
PBPUEN	400D 042Ch	PB Weak Pull-up Enable	RW	0000 0000h
PCPUEN	400D 0430h	PC Weak Pull-up Enable	RW	0000 0000h
PDPUEN	400D 0434h	PD Weak Pull-up Enable	RW	0000 0000h
PEPUEN	400D 0438h	PE Weak Pull-up Enable	RW	0000 0000h
PFPUEN	400D 043Ch	PF Weak Pull-up Enable	RW	0000 0000h
PGPUEN	400D 0440h	PG Weak Pull-up Enable	RW	0000 0000h
PAPDEN	400D 0444h	PA Weak Pull-down Enable	RW	0000 0000h
PBPDEN	400D 0448h	PB Weak Pull-down Enable	RW	0000 0000h
PCPDEN	400D 044Ch	PC Weak Pull-down Enable	RW	0000 0000h
PDPDEN	400D 0450h	PD Weak Pull-down Enable	RW	0000 0000h
PEPDEN	400D 0454h	PE Weak Pull-down Enable	RW	0000 0000h
PFPDEN	400D 0458h	PF Weak Pull-down Enable	RW	0000 0000h
PGPDEN	400D 045Ch	PG Weak Pull-down Enable	RW	0000 0000h
PADS	400D 0460h	PA Drive Strength/Schmitt Trigger	RW	0000 0000h
PBDS	400D 0464h	PB Drive Strength/Schmitt Trigger	RW	0000 0000h
PCDS	400D 0468h	PC Drive Strength/Schmitt Trigger	RW	0000 0000h
PDDS	400D 046Ch	PD Drive Strength/Schmitt Trigger	RW	0000 0000h
PEDS	400D 0470h	PE Drive Strength/Schmitt Trigger	RW	0000 0000h
PFDS	400D 0474h	PF Drive Strength/Schmitt Trigger	RW	0000 0000h
PGDS	400D 0478h	PG Drive Strength/Schmitt Trigger	RW	0000 0000h

Table 3-21 Windowed Watchdog Timer (WWDT) Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
WWDTCTL	400D 0800h	WWDT Control	RW	0000 0000h
WWDTLOAD	400D 0804h	WWDT Load Counter Value	RW	0010 FFFFh
WWDTCTR	400D 0808h	WWDT Counter	RO	0000 FFFFh
WWDTINTF	400D 080Ch	WWDT Interrupt Flag	RO	0000 0000h
WWDTCLEAR	400D 0810h	WWDT Clear	WO	--
WWDTLOCK	400D 0814h	WWDT Lock	RW	0000 0000h

Table 3-22 Real-Time Clock (RTC) Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
RTCCTL	400D 0C00h	RTC Control	RW	0000 0008h
RTCTIME	400D 0C04h	RTC Time	RO	0000 0000h
RTCDATE	400D 0C08h	RTC Date	RO	0001 0100h
RTCTIMESET	400D 0C0Ch	RTC Time Setting	RW	0000 0000h
RTCDATESET	400D 0C10h	RTC Date Setting	RW	0001 0100h
RTCALARMSET	400D 0C14h	RTC Alarm Setting	RW	0000 0000h

Table 3-23 CRC Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
CRCCTL	400D 1000h	CRC Control	RW	0000 0000h
CRCDATAIN	400D 1004h	CRC Data Input	RW	0000 0000h
CRCSEED	400D 1008h	CRC Seed Value	RW	0000 0000h
CRCDATAOUT	400D 100Ch	CRC Data Output	RO	0000 0000h

Table 3-24 GPIOA Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOAMODE	400D 1400h	GPIOA Pin Mode Select	RW	0000 FFFFh
GPIOAOUTMASK	400D 1404h	GPIOA Data Output Write Mask	RW	0000 0000h
GPIOAOUT	400D 1408h	GPIOA Data Output Value	RW	FFFF FFFFh
GPIOAIN	400D 140Ch	GPIOA Data Input Value	RO	--
GPIOAINTEN	400D 1410h	GPIOA Interrupt Enable	RW	0000 0000h
GPIOAINTFLAG	400D 1414h	GPIOA Interrupt Flag	RO	0000 0000h
GPIOAINTCLEAR	400D 141Ch	GPIOA Interrupt Clear	WO	--
GPIOAINTTYPE	400D 1420h	GPIOA Interrupt Type	RW	0000 0000h
GPIOAINTVALUE	400D 1424h	GPIOA Interrupt Value	RW	0000 0000h
GPIOAINTEDGEBOTH	400D 1428h	GPIOA Interrupt Edge Both	RW	0000 0000h
GPIOACLKSYNC	400D 142Ch	GPIOA Clock Synchronization	RW	0000 0000h
GPIOADOSET	400D 1430h	GPIOA Data Output Set	WO	--
GPIOADOCLEAR	400D 1434h	GPIOA Data Output Clear	WO	--

Table 3-25 GPIOB Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOBMODE	400D 1800h	GPIOB Pin Mode Select	RW	0000 FFFFh
GPIOBOUTMASK	400D 1804h	GPIOB Data Output Write Mask	RW	0000 0000h
GPIOBOUT	400D 1808h	GPIOB Data Output Value	RW	FFFF FFFFh
GPIOBIN	400D 180Ch	GPIOB Data Input Value	RO	--
GPIOBINTEN	400D 1810h	GPIOB Interrupt Enable	RW	0000 0000h
GPIOBINTFLAG	400D 1814h	GPIOB Interrupt Flag	RO	0000 0000h
GPIOBINTCLEAR	400D 181Ch	GPIOB Interrupt Clear	WO	--
GPIOBINTTYPE	400D 1820h	GPIOB Interrupt Type	RW	0000 0000h
GPIOBINTVALUE	400D 1824h	GPIOB Interrupt Value	RW	0000 0000h
GPIOBINTEDGEBOTH	400D 1828h	GPIOB Interrupt Edge Both	RW	0000 0000h
GPIOBCLKSYNC	400D 182Ch	GPIOB Clock Synchronization	RW	0000 0000h
GPIOBDOSET	400D 1830h	GPIOB Data Output Set	WO	--
GPIOBDOCLEAR	400D 1834h	GPIOB Data Output Clear	WO	--

Table 3-26 GPIOC Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOCMODE	400D 1400h	GPIOC Pin Mode Select	RW	0000 FFFFh
GPIOCOUTMASK	400D 1404h	GPIOC Data Output Write Mask	RW	0000 0000h
GPIOCOUT	400D 1408h	GPIOC Data Output Value	RW	FFFF FFFFh
GPIOCIN	400D 140Ch	GPIOC Data Input Value	RO	--
GPIOCINTEN	400D 1410h	GPIOC Interrupt Enable	RW	0000 0000h
GPIOCINTFLAG	400D 1414h	GPIOC Interrupt Flag	RO	0000 0000h
GPIOCINTCLEAR	400D 141Ch	GPIOC Interrupt Clear	WO	--
GPIOCINTTYPE	400D 1420h	GPIOC Interrupt Type	RW	0000 0000h
GPIOCINTVALUE	400D 1424h	GPIOC Interrupt Value	RW	0000 0000h
GPIOCINTEDGEBOTH	400D 1428h	GPIOC Interrupt Edge Both	RW	0000 0000h
GPIOCCLKSYNC	400D 142Ch	GPIOC Clock Synchronization	RW	0000 0000h
GPIOCDOSET	400D 1430h	GPIOC Data Output Set	WO	--
GPIOCDOCLEAR	400D 1434h	GPIOC Data Output Clear	WO	--

Table 3-27 GPIOD Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIODMODE	400D 2000h	GPIOD Pin Mode Select	RW	0000 FFFFh
GPIODOUTMASK	400D 2004h	GPIOD Data Output Write Mask	RW	0000 0000h
GPIODOUT	400D 2008h	GPIOD Data Output Value	RW	FFFF FFFFh
GPIODIN	400D 200Ch	GPIOD Data Input Value	RO	--
GPIODINTEN	400D 2010h	GPIOD Interrupt Enable	RW	0000 0000h
GPIODINTFLAG	400D 2014h	GPIOD Interrupt Flag	RO	0000 0000h
GPIODINTCLEAR	400D 201Ch	GPIOD Interrupt Clear	WO	--
GPIODINTTYPE	400D 2020h	GPIOD Interrupt Type	RW	0000 0000h
GPIODINTVALUE	400D 2024h	GPIOD Interrupt Value	RW	0000 0000h
GPIODINTEDGEBOTH	400D 2028h	GPIOD Interrupt Edge Both	RW	0000 0000h
GPIODCLKSYNC	400D 202Ch	GPIOD Clock Synchronization	RW	0000 0000h
GPIODDOSET	400D 2030h	GPIOD Data Output Set	WO	--
GPIODDOCLEAR	400D 2034h	GPIOD Data Output Clear	WO	--

Table 3-28 GPIOE Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOEMODE	400D 2400h	GPIOE Pin Mode Select	RW	0000 FFFFh
GPIOEOUTMASK	400D 2404h	GPIOE Data Output Write Mask	RW	0000 0000h
GPIOEOUT	400D 2408h	GPIOE Data Output Value	RW	FFFF FFFFh
GPIOEIN	400D 240Ch	GPIOE Data Input Value	RO	--
GPIOEINTEN	400D 2410h	GPIOE Interrupt Enable	RW	0000 0000h
GPIOEINTFLAG	400D 2414h	GPIOE Interrupt Flag	RO	0000 0000h
GPIOEINTCLEAR	400D 241Ch	GPIOE Interrupt Clear	WO	--
GPIOEINTTYPE	400D 2420h	GPIOE Interrupt Type	RW	0000 0000h
GPIOEINTVALUE	400D 2424h	GPIOE Interrupt Value	RW	0000 0000h
GPIOEINTEDGEBOTH	400D 2428h	GPIOE Interrupt Edge Both	RW	0000 0000h
GPIOECLKSYNC	400D 242Ch	GPIOE Clock Synchronization	RW	0000 0000h
GPIOEDOSET	400D 2430h	GPIOE Data Output Set	WO	--
GPIOEDOCLEAR	400D 2434h	GPIOE Data Output Clear	WO	--

Table 3-29 GPIOF Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOFMODE	400D 2800h	GPIOF Pin Mode Select	RW	0000 FFFFh
GPIOFOUTMASK	400D 2804h	GPIOF Data Output Write Mask	RW	0000 0000h
GPIOFOUT	400D 2808h	GPIOF Data Output Value	RW	FFFF FFFFh
GPIOFIN	400D 280Ch	GPIOF Data Input Value	RO	--
GPIOFINTEN	400D 2810h	GPIOF Interrupt Enable	RW	0000 0000h
GPIOFINTFLAG	400D 2814h	GPIOF Interrupt Flag	RO	0000 0000h
GPIOFINTCLEAR	400D 281Ch	GPIOF Interrupt Clear	WO	--
GPIOFINTTYPE	400D 2820h	GPIOF Interrupt Type	RW	0000 0000h
GPIOFINTVALUE	400D 2824h	GPIOF Interrupt Value	RW	0000 0000h
GPIOFINTEDGEBOTH	400D 2828h	GPIOF Interrupt Edge Both	RW	0000 0000h
GPIOFCLKSYNC	400D 282Ch	GPIOF Clock Synchronization	RW	0000 0000h
GPIOFDOSET	400D 2830h	GPIOF Data Output Set	WO	--
GPIOFDOCLEAR	400D 2834h	GPIOF Data Output Clear	WO	--

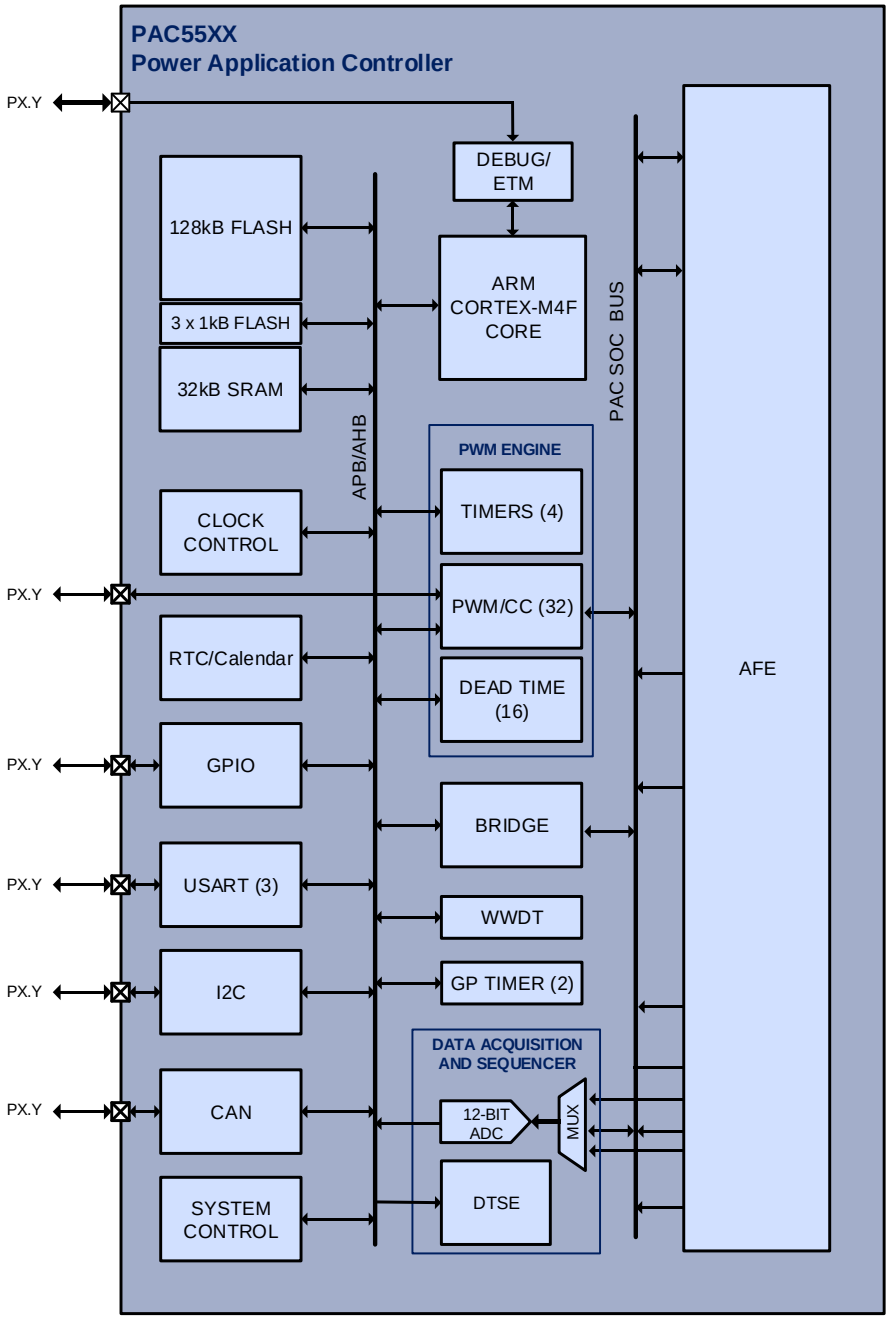
Table 3-30 GPIOG Register Map

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOGMODE	400D 2C00h	GPIOG Pin Mode Select	RW	0000 FFFFh
GPIOGOUTMASK	400D 2C04h	GPIOG Data Output Write Mask	RW	0000 0000h
GPIOGOUT	400D 2C08h	GPIOG Data Output Value	RW	FFFF FFFFh
GPIOGIN	400D 2C0Ch	GPIOG Data Input Value	RO	--
GPIOGINTEN	400D 2C10h	GPIOG Interrupt Enable	RW	0000 0000h
GPIOGINTFLAG	400D 2C14h	GPIOG Interrupt Flag	RO	0000 0000h
GPIOGINTCLEAR	400D 2C1Ch	GPIOG Interrupt Clear	WO	--
GPIOGINTTYPE	400D 2C20h	GPIOG Interrupt Type	RW	0000 0000h
GPIOGINTVALUE	400D 2C24h	GPIOG Interrupt Value	RW	0000 0000h
GPIOGINTEDGEBOTH	400D 2C28h	GPIOG Interrupt Edge Both	RW	0000 0000h
GPIOGCLKSYNC	400D 2C2Ch	GPIOG Clock Synchronization	RW	0000 0000h
GPIOGDOSET	400D 2C30h	GPIOG Data Output Set	WO	--
GPIOGDOCLEAR	400D 2C34h	GPIOG Data Output Clear	WO	--

4 ARCHITECTURAL BLOCK DIAGRAM

The following is an architectural block diagram of the MCU and associated peripherals for the PAC55XX family.

Figure 4-1 PAC55XX Architectural Block Diagram



5 SYSTEM AND CLOCK CONTROL (SCC)

5.1 Overview

The System and Clock Control (SCC) module controls the clock and other system components for the PAC55XX family of devices.

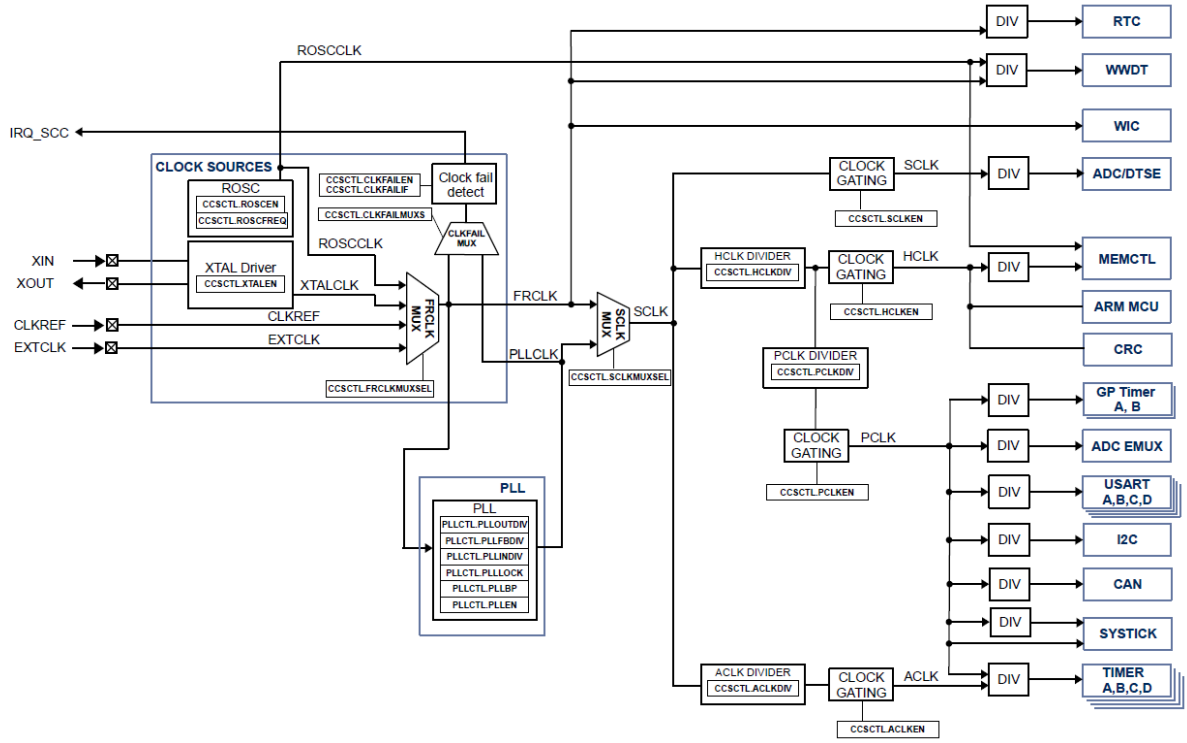
5.2 Features

- Clock Control System (CCS):
  - 4 clock sources:
    - 4MHz internally generated 2% RC oscillator
    - 16MHz Ring Oscillator
    - External clock input for up to 20MHz external clock sources
    - Crystal driver for up to 10MHz crystals or ceramic resonators[2]
  - 300MHz PLL
  - 5 system clocks for peripherals with programmable clock dividers
  - Clock gating for low-power mode support
- USART Mode Control
  - Selects between SSP or UART modes
- Interrupt Vector Table
  - 32 interrupts
  - 8 levels of priority

[2] Note that only some devices contain the pins for the crystal input and output.

5.3 Clock Control System Block Diagram

Figure 5-1 CCS Block Diagram



5.4 Clock Sources

The PAC55XX can support up to 4 clock sources:

- Ring Oscillator Clock (***ROSCCLK***)
- Clock Reference (***CLKREF***)
- External Clock (***EXTCLK***)

The PAC55XX may select any of these clock sources using the FRCLK MUX. The output of the FRCLK MUX becomes the ***FRCLK*** system clock.

Table 5-1 CCS Clock Source Selections

CCSCTL.FRCLKMUXSEL	FRCLK
00b (default)	ROSCCLK
01b	CLKREF
10b	Reserved
11b	EXTCLK

5.4.1 ROSCCLK

The ***ROSCCLK*** is the output of the 16MHz Ring Oscillator. ***ROSCCLK*** is the default system clock when the PAC55XX comes out of reset. ***ROSCCLK*** is the default input for ***FRCLK*** and can optionally be used as a 2nd time-base for the WWDT peripheral.

The ROSC may be enabled or disabled using the **CCSCTL.ROSCEN** bit.

5.4.2 CLKREF

The ***CLKREF*** is an internally generated and trimmed 4MHz time base. The ***CLKREF*** is always running, even when not used.

5.4.3 EXTCLK

The ***EXTCLK*** is an external clock input that allows up to 20MHz, 50% duty cycle clock to be used as a system clock.

EXTCLK is available through a device pin using the Digital Peripheral MUX.

5.5 System Clocks

5.5.1 FRCLK

FRCLK (Free-Running Clock) is the output of the FRCLK MUX and is the default input to the SCLK MUX used for generating **SCLK**. **FRCLK** is the input to the PLL.

5.5.2 PLLCLK

PLLCLK (PLL Clock) is the clock output of the PLL. **PLLCLK** is an optional input to the **SCLK** MUX used for generating **SCLK**. The input to the PLL is **FRCLK**.

5.5.3 SCLK

SCLK (System Clock) is the clock output of the SCLK MUX. SCLK is gated when the PAC55XX is put into ARM deep sleep mode.

The input to the SCLK MUX may be selected as shown below.

Table 5-2 SCLK Clock Input Selections

CCSCTL.SCLKMUXSEL	SCLK
0b (default)	FRCLK
1b	PLLCLK

5.5.4 PCLK

PCLK (APB Clock) is the main peripheral system clock which is generated from ***HCLK***. There is an optional input clock divider for ***PCLK*** that may be used to reduce the frequency to less than 150MHz for the digital peripherals.

PCLK may be gated when the PAC55XX is put into ARM deep sleep mode.

The ***PCLK*** divider may be configured as follows.

Table 5-3 PCLK Clock Divider Settings

CCSCTL.PCLKDIV	PCLK
000b (default)	SCLK /1
001b	SCLK /2
010b	SCLK /3
011b	SCLK /4
100b	SCLK /5
101b	SCLK /6
110b	SCLK /7
111b	SCLK /8

5.5.5 ACLK

ACLK (Auxiliary Clock) is clock generated from ***SCLK***. ***ACLK*** is used as an optional clock source for the PWM Timers, to provide very fine timer resolution. The maximum speed of ***ACLK*** is 300MHz and it has an input clock divider.

ACLK may be gated when the PAC55XX is put into ARM deep sleep mode.

The ***ACLK*** divider may be configured as follows.

Table 5-4 ACLK Clock Divider Settings

CCSCTL.ACLKDIV	ACLK
000b (default)	SCLK /1
001b	SCLK /2
010b	SCLK /3
011b	SCLK /4
100b	SCLK /5
101b	SCLK /6
110b	SCLK /7
111b	SCLK /8

5.5.6 HCLK

HCLK (AHB Clock) is clock generated from ***SCLK***. ***HCLK*** is used as the clock source for the ***PCLK*** system clock and the AHB peripherals such as the Arm® Cortex®-M4F MCU, memory controller and CRC engine. The maximum speed of ***HCLK*** is 150MHz and it has an input clock divider.

HCLK may be gated when the PAC55XX is put into ARM deep sleep mode.

The ***HCLK*** divider may be configured as follows.

Table 5-5 HCLK Clock Divider Settings

CCSCTL.HCLKDIV	HCLK
000b (default)	SCLK /1
001b	SCLK /2
010b	SCLK /3
011b	SCLK /4
100b	SCLK /5
101b	SCLK /6

110b	SCLK /7
111b	SCLK /8

5.6 PLL Configuration

The clock input to the PLL may be configured to be ***CLKREF*** or ***EXTCLK*** by using **CCSCTL.PLLMUXSEL**. The PLL may be enabled or disabled using **CCSCTL.PLLEN**.

To configure the PLL output frequency, the user must know the specific input frequency and the user must set the PLL dividers. The variables used for calculating the PLL output frequency are shown below.

Table 5-6 PLL Configuration Variables

VARIABLE	SOURCE	DESCRIPTION	CONSTRAINTS
PLLOUT	-	PLL Output Frequency in MHz	62.5MHz – 300MHz
PLLIN	**CCSCTL.PLLMUXS**	PLL Input Frequency in MHz	1MHz <= PLLIN / PLLINDIV <= 50MHz
PLLINDIV	**PLLCTL.PLLINDIV**	PLL Input Divider	1 <= PLLINDIV <= 15
PLLOUTDIV	**PLLCTL.PLLODIV**	PLL Output Divider	0 <= PLLOUTDIV <= 3
PLLFBDIV	**PLLCTL.PLLFBDIV**	PLL Feedback Divider	4 <= PLLFBDIV <= 16383
PLLVCO		PLL VCO Frequency	200MHz <= PLLVCO <= 400MHz

The following formulas can be used to calculate the PLL VCO frequency and PLL output frequency:

- NO = $2^{PLLOUTDIV}$
- PLLOUT = PLLIN * (PLLFBDIV / PLLINDIV) / ($2^{PLLOUTDIV}$)
- PLLVCO = PLLIN * (PLLFBDIV / PLLINDIV)

The following table shows some example configurations that may be used for PLL configuration.

Table 5-7 PLL Configuration Examples

PLLIN	PLLINDIV	PLLFBDIV	PLLOUTDIV	PLLVCO	PLLOUT
4MHz	1	15	1	60MHz	30 MHz
4 MHz	1	25	0	100 MHz	50 MHz
4MHz	1	50	1	200 MHz	100 MHz
4 MHz	1	50	0	200 MHz	200 MHz
4 MHz	1	75	0	300 MHz	300 MHz

After changing any of the PLL dividers, there is a 500µs time period before the PLL is locked and stable. The user should take care to a make sure that ***PLLOUT*** is not used as SCLK, until the PLL has locked.

Additionally, the user by read the **PLLCTL.LOCK** bit to see if the PLL is locked.

5.7 Peripheral Clock Selection

Each peripheral in the PAC55XX has at least one clock input that is available.

The following table shows the set of system peripherals, and what system clocks are available for their use.

Table 5-8 Peripheral Clock Input

PERIPHERAL	FRCLK	SCLK	PCLK	PCLK /3	ACLK	HCLK	ROSCCLK	REFCLK
RTC	X							
WWDT	X						X	
WIC (Wakeup Interrupt Controller)						X		
ADC and DTSE		X						
GP Timer A, B			X					
ADC EMUX			X					
USART A, B, C, D			X					
I2C			X					
CAN			X					
CRC						X		
Timer A, B, C, D			X		X			
SysTick Timer						X[3]		
ARM MCU						X		
SRAM						X		
Memory Controller						X	X	

[3] May be HCLK, or HCLK/8

5.8 Low-Power Clock Gating

The PAC55XX supports low-power operation. There are three different modes of low-power operation:

- Arm® Cortex®-M4F Sleep Mode
- Arm® Cortex®-M4F Deep Sleep Mode
- System Hibernate Mode

When Arm® Cortex®-M4F Sleep Mode is entered, just the MCU is put to sleep. All other clocks and peripherals remain active. Any interrupt may wake up the system from this mode.

In System Hibernate Mode, the entire MCU is not powered at all. To wake up, the user must set a wakeup timer in the Analog sub-system, or wake the system with a push-button event on the Analog sub-system. For more information on this mode, see the device-specific User Guide.

In Arm® Cortex®-M4F Deep Sleep Mode, the user may gate several clocks in the PAC55XX to the peripherals to reduce energy consumption as much as possible, without removing power to the device. Only certain peripherals are able to 'wake up' the device from this state.

Before entering this mode, the firmware should take the following steps:

- Set **CCSCTL.SCLKMUXSEL** to 0b to set ***FRCLK*** as the ***SCLK*** input clock (not ***PLLCLK***)
- Disable the PLL by setting **CCSCTL.PLLEN** = 0b
- Set **CCSCTL.FRCLKMUXSEL** to 01b (***CLKREF***) to generate ***FRCLK*** from ***CLKREF***, which will always be present
- Disable ROSC by setting **CCSCTL.ROSCEN** = 0b
- Disable any unwanted analog peripherals such as the ADC (**ADCCTL.ENABLE** = 0b)
- Disable PCLK by setting **CCSCTL.PCLKEN** = 0b
- Disable ACLK by setting **CCSCTL.ACLKEN** = 0b
- Disable ADCCLK by setting **CCSCTL.ADCCLKEN** = 0b

When in this mode, the only remaining clocks are ***FRCLK***, and possibly ***ROSCLK*** if the user did not disable it. In order to wake up from this mode, a peripheral that uses ***FRCLK*** or ***ROSCCLK*** must be used, such as the RTC, WWDT or GPIO.

The STCLK (SysTick timer alternative clock) is fixed to HCLK / 8. This clock may be gated automatically during deep sleep mode by setting the **CCSCTL.STCLKSLPEN** to 1b. When set to a 0b, this clock remains active during deep sleep mode.

Interrupts from any of these peripherals while in Arm® Cortex®-M4F deep sleep mode will wake up the device.

For details on which instructions to execute to enter Arm® Cortex®-M4F deep sleep mode, refer to the Arm® Cortex®-M4F Technical Reference Manual at http://www.arm.com.

5.9 Clock Failure Detection

The PAC55XX contains a configurable clock failure detection circuit. The clock failure detection monitors the system clock and if it stops or slows significantly, the device will enable the ROSCCLK and switch system functions to this clock so it may continue operation.

To enable the Clock Failure Detection feature, set **CCSCTL.CLKFAILEN** to 1b.

The user may select the clock to sample FRCLK or PLLCLK by setting **CCSCTL.CLKFAILMUXSEL** to the desired value. When the configured clock stops, then the **CCSCTL.CLKFAILIF** flag set and the IRQ to the MCU is asserted.

5.10 USART Mode

The **CCSCTL** register is used to configure the USART A to USART D mode to be SSP or UART. The description of how to use these mode settings can be found in:

- USART A
- USART B
- USART C
- USART D

5.11 Interrupt Vector Table

The PAC55XX supports 31 user interrupts for the various peripherals. The interrupt vector table is located at address 0 in FLASH memory by default.

Each interrupt has 8 levels of priority

The interrupt vector table is shown below.

Table 5-9 PAC55XX Interrupt Vector Table

Exception	IRQ	Addr Offset	Interrupt	Description
System Exceptions				
		0000h	Initial SP Value	Initial value of the stack pointer
1		0004h	Reset	Reset Vector Address
2	-14	0008h	NMI	Non-Maskable Interrupt
3	-13	000Ch	HardFault	
4	-12	0010h	MemManage	
5	-11	0014h	BusFault	
6	-10	0018h	UsageFault	
7		001Ch	Reserved	
8		0020h	Reserved	
9		0024h	Reserved	
10		0028h	Reserved	
11	-5	002Ch	SVCall	
12		0030h	Reserved for Debug	
13		0034h	Reserved	
14	-2	0038h	PendSV	
15	-1	003Ch	SysTick	
User Interrupts				
16	0	0040h	Memory Controller	Memory Controller IRQ
17	1	0044h	WWDT	WWDT IRQ
18	2	0048h	RTC	RTC with Calendar IRQ
19	3	004Ch	ADC0	ADC IRQ 0
20	4	0050h	ADC1	ADC IRQ 1
21	5	0054h	ADC2	ADC IRQ 2
22	6	0058h	ADC3	ADC IRQ 3

23	7	005Ch	Timer A	Timer A IRQ
24	8	0060h	Timer B	Timer B IRQ
25	9	0064h	Timer C	Timer C IRQ
26	10	0068h	Timer D	Timer D IRQ
27	11	006Ch	Timer A QEP	Timer A QEP IRQ
28	12	0070h	Timer B QEP	Timer B QEP IRQ
29	13	0074h	Timer C QEP	Timer C QEP IRQ
30	14	0078h	Timer D QEP	Timer D QEP IRQ
31	15	007Ch	GPIOA	GPIO A IRQ
32	16	0080h	GPIOB	GPIO B IRQ
33	17	0084h	GPIOC	GPIO C IRQ
34	18	0088h	GPIOD	GPIO D IRQ
35	19	008Ch	GPIOE	GPIO E IRQ
36	20	0090h	GPIOF	GPIO F IRQ
37	21	0094h	GPIOG	GPIO G IRQ
38	22	0098h	I2C	I2C IRQ
39	23	009Ch	USARTA	USART A IRQ
40	24	0100h	USARTB	USART B IRQ
41	25	0104h	USARTC	USART C IRQ
42	26	0108h	USARTD	USART D IRQ
43	27	010Ch	CAN	CAN IRQ
44	28	0110h	GPTIMERA	GP Timer A IRQ
45	29	0114h	GPTIMERB	GP Timer B IRQ
46	30	0118h	SCC	System Clock Control IRQ
47	31		Reserved	

5.12 Register Summary

Table 5-10 SCC Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
CCSCTL	400D 0400h	CCS Control	RW	0000 F104h
CCSPLLCTL	400D 0404h	PLL Control	RW	0000 0000h
CCSROSCTRIM	400D 0408h	ROSC Trim Control	RW	0000 007Fh

5.13 Register Detail

5.13.1 CCSCTL

Register 5-1 CCSCTL (CCS Configuration, 400D 0400h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**USDMODE**	RW	0b	USART D Mode: 0b: SSP 1b: UART
30	**USCMODE**	RW	0b	USART C Mode: 0b: SSP 1b: UART
29	**USBMODE**	RW	0b	USART B Mode: 0b: SSP 1b: UART
28	**USAMODE**	RW	0b	USART A Mode: 0b: SSP 1b: UART
27	**Reserved**	RO	0b	Reserved
26:24	**HCLKDIV**	RW	000b	HCLK Divider: 000b: SCLK /1 001b: SCLK /2 010b: SCLK /3 011b: SCLK /4 100b: SCLK /5 101b: SCLK /6 110b: SCLK /7 111b: SCLK /8
23	**Reserved**	RO	0b	Reserved
22:20	**ACLKDIV**	RW	000b	ACLK Divider: 000b: SCLK /1 001b: SCLK /2 010b: SCLK /3 011b: SCLK /4 100b: SCLK /5 101b: SCLK /6 110b: SCLK /7 111b: SCLK /8
19	**Reserved**	RO	0b	Reserved
18:16	**PCLKDIV**	RW	000b	PCLK Divider: 000b: HCLK /1 001b: HCLK /2 010b: HCLK /3 011b: HCLK /4 100b: HCLK /5 101b: HCLK /6 110b: HCLK /7 111b: HCLK /8
15	**STCLKSLPEN**	RW	1b	STCLK Sleep enable: 0b: disabled: STCLK active in deep sleep mode 1b: enabled: STCLK gated in deep sleep mode

14	**ADCCLKEN**	RW	1b	ADCCLK enable: 0b: disabled 1b: enabled
13	**ACLKEN**	RW	1b	ACLK enable: 0b: disabled 1b: enabled
12	**PCLKEN**	RW	1b	PCLK enable: 0b: disabled 1b: enabled
11	**SWRESET**	RW	0b	Software POR
10:9	**Reserved**	RO	0000b	Reserved
8	**LDOEN**	RW	1b	1.8V LDO Enable: 0b: disabled 1b: enabled
7	**CLKFAILIF**	W1C	0	Clock Fail Interrupt Flag: 0b: no flag 1b: flag
6	**CLKFAILMUXSEL**	RW	0	Clock Fail MUX Select: 0b: FRCLK 1b: PLLCLK
5	**CLKFAILEN**[4]	RW	0	Clock Fail Detection Enabled: 0b: disabled 1b: enabled
4	**SCLKMUXSEL**	RW	0	SCLK MUX Select: 0b: FRCLK 1b: PLLCLK
3	**Reserved**	RO	0	Reserved
2	**ROSCEN**	RW	1	ROSC Enabled: 0b: disabled 1b: enabled
1:0	**FRCLKMUXSEL**	RW	00b	FRCLK MUX Select: 00b: ROSC 01b: CLKREF 10b: Reserved 11b: EXTCLK

[4] Note that clock failure detection does not function in VER1 of the PAC55XX MCU

5.13.2 CCSPLLCTL

Register 5-2 CCSPLLCTL (CCS PLL Configuration, 400D 0404h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:25	**Reserved**	RO	0b	Reserved
24	**PLLLOCK**	RO	0b	PLL Lock Status: 0b: PLL not locked 1b: PLL locked
23:22	**Reserved**	RO	0b	
21:8	**PLLFBDIV**	RW	0	PLL Feedback Divider. Must be between 4 – 16383.
7:4	**PLLINDIV**	RW	0	PLL Input Divider. Must be between 1 – 15.
3:2	**PLLOUTDIV**	RW	0	PLL Output Divider: 00b: /1 01b: /2 10b: /4 11b: /8
1	**PLLBP**	RW	0	PLL Bypass: 0b: Bypass inactive 1b: Bypass active
0	**PLLEN**	RW	0	PLL Enabled: 0b: PLL Disabled 1b: PLL Enabled

5.13.3 CCSROSCTRIM

Register 5-3 CCSROSCTRIM (CCS ROSC Trim Configuration, 400D 0408h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:7	**Reserved**	RO	0b	Reserved
6:0	**TRIM**	RW	111 1111b	ROSC Trim Value.

6 MEMCTL

6.1 Overview

The Memory Controller (MEMCTL) allows access to the internal program FLASH, as well as INFO-1, INFO-2, INFO-3 and SRAM. The access to the memory controller is through the AHB bus for reading and writing FLASH and RAM, as well as register access for control and configuration.

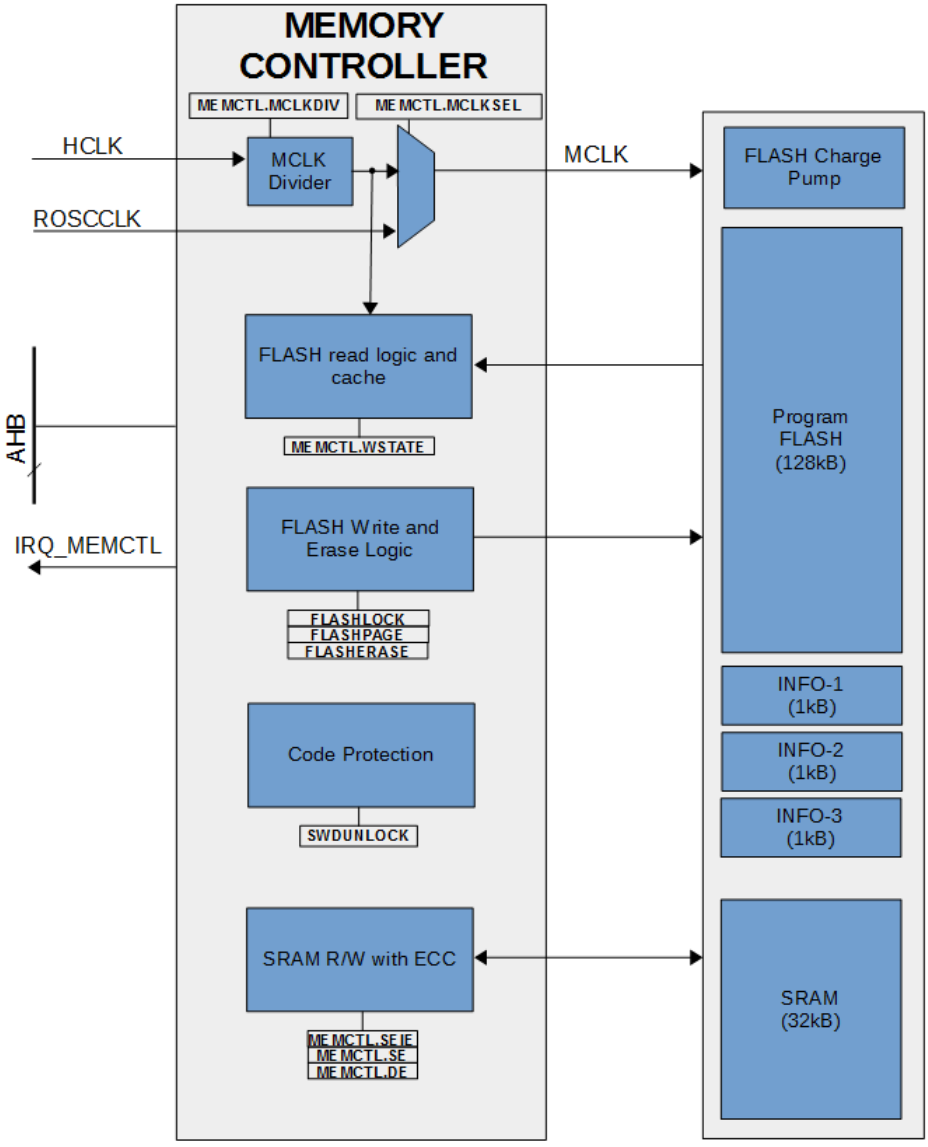
The memory controller is clocked from a divided HCLK and ROSCCLK and has a single interrupt signal to the NVIC (IRQ_MEMCTL).

6.2 Features

- FLASH control for read, erase, write operations
- FLASH read cache
- Invalid address interrupts
- SRAM control for read/write
- SECDED ECC
  - Single Error Correction, Dual Error Detection
- Code Protection

6.3 System Block Diagram

Figure 6-1 Memory Controller System Block Diagram



6.4 Functional Description

6.4.1 Clocking

The memory controller is an AHB bus client and it may be clocked with either HCLK or ROSCCLK. By default, the PAC55XX will supply a 16MHz ROSCCLK to the memory controller via HCLK.

The FLASH memory on the PAC55XX has two clocking requirements:

- FLASH read frequency must be a maximum of 25MHz
- The FLASH charge pump has an input clock (MCLK) that must obey the following clock frequencies:
  - 16MHz – 30MHz nominal MCLK for FLASH read
  - 30MHz nominal MCLK for FLASH read, write or erase

In order to support these frequencies, the memory controller can be configured as described below.

The memory controller has a clock divider for HCLK to generate MCLK (16MHz – 30MHz). In order to do this, there is a MCLK divider that the user may use to generate the proper MCLK. The MCLK divider may be configured using the **MEMCTL.MCLKDIV** field as shown below.

The maximum FLASH read frequency cannot be above the configured MCLK frequency.

Note that any time the user wants to write the **MEMCTL** register, they must first write the **FLASHLOCK** with the value D513 B490h.

Table 6-1 Memory Controller MCLK Divider

MEMCTL.MCLKDIV	MCLK Output
0000b	HCLK /1
0001b	HCLK /2
0010b	HCLK /3
0011b	HCLK /4
0100b	HCLK /5
0101b	HCLK /6
0110b	HCLK /7
0111b	HCLK /8
1000b	HCLK /9
1001b	HCLK /10
1010b	HCLK /11
1011b	HCLK /12
1100b	HCLK /13
1101b	HCLK /14
1110b	HCLK /15
1111b	HCLK /16

The maximum FLASH read frequency is 25MHz. In order to control this frequency to FLASH, the memory controller has wait states that it can insert when performing transactions to FLASH. This will insert a programmable number of wait states to HCLK for FLASH access as shown in the table below.

There are always two wait states, so the **MEMCTL.WSTATE** value will add to the two wait states for accessing FLASH.

Table 6-2 Memory Controller FLASH Wait States

MEMCTL.MCLKDIV	Wait States
0000b	0 wait states
0001b	1 wait states
0010b	2 wait states
0011b	3 wait states
0100b	4 wait states
0101b	5 wait states
0110b	6 wait states
0111b	7 wait states
1000b	8 wait states
1001b	9 wait states
1010b	10 wait states
1011b	11 wait states
1100b	12 wait states
1101b	13 wait states
1110b	14 wait states
1111b	15 wait states

The table below shows some examples of how to configure the memory controller to support the FLASH read frequency and MCLK frequency for various HCLK input clock frequencies.

Table 6-3 FLASH Read Clock Configuration Examples

HCLK	MEMCTL.MCLKDIV	MCLK	MEMCTL.WSTATE	FLASH READ FREQ	FLASH READ OK	FLASH WRITE OK
150MHz	0100b (HCLK /5)	30MHz	0110b (6 wait states)	150MHz / 6 = 25MHz	X	X

125MHz	0011b (HCLK /4)	31.25MHz[5]	0101b (5 wait states)	125MHz / 5 = 25MHz	X	X
90MHz	0010b (HCLK /3)	30MHz	0100b (4 wait states)	90MHz / 4 = 22.5MHz	X	X
60MHz	0001b (HCLK /2)	30MHz	0011b (3 wait states)	60MHz / 3 = 20MHz	X	X
16MHz	0000b (HCLK /1)	16MHz	0010b (2 wait states)	30MHz / 2 = 15MHz	X	

Note that for the last entry in the table above, 16MHz is only valid for FLASH read, not for FLASH program or erase.

6.4.2 Low-Power Operation

If FLASH is going to be unused by the application and the user wishes to conserve power, it may put the FLASH into a standby mode. If using this mode, be sure that the program does not need to execute any code from FLASH, only from RAM otherwise the system will stall.

To enable standby mode set the **MEMCTL.STDBY** to 1b. To exit standby mode set **MEMCTL.STDBY** to 0b. After exiting standby mode or between writing this field from a 1b to a 0b, there is a 20µs delay before accessing FLASH memory (see the device data sheet).

Note that any time the user wants to write the **MEMCTL** register, they must first write the **FLASHLOCK** with the value D513 B490h.

To disable the 1.8V FLASH supply, the user may write the **CCSCTL.LDOEN** bit to 0b. When set to 0b, the FLASH memory will not be usable.

6.4.3 Invalid Memory Accesses

If the PAC55XX attempts to make an invalid memory access, the memory controller will set the **MEMSTATUS.INVADDR** interrupt flag to a 1b. If the **MEMCTL.INVADDRIE** is set to 1b when this happens, then the memory controller will assert the IRQ_MEMCTL interrupt signal to the NVIC.

The **MEMSTATUS.INVADDR** interrupt flag may be cleared by writing it to 1b.

Note that any time the user wants to write the **MEMCTL** register, they must first write the **FLASHLOCK** with the value D513 B490h.

6.4.4 SRAM

The PAC55XX contains 32kB of SRAM. SRAM may be accessed through AHB transactions at a rate of up to 150MHz. The memory controller supports 32-bit word aligned 32-bit, 16-bit and 8-bit read write transactions to SRAM.

[5] The MCLK frequency range may be from 14.4MHz – 33MHz for FLASH read and 27MHz – 33MHz for FLASH write, so this frequency is in the acceptable range.

Because the data bus to SRAM is 32-bits, any 8-bit or 16-bit transactions to SRAM require a read-modify-write operation, and will have reduced performance, compared to 32-bit transactions.

The MCU may read and write data or execute code from SRAM.

The PAC55XX SRAM contains support for Single Error Correction and Dual Error Detection (SECDED) for enhanced code and data integrity checking. Each word of SRAM contains additional 7-bits for SECDED error checking.

For word that is read from SRAM, the memory controller calculates the SECDED value and compares it to the 7-bits stored with that word in SRAM to verify that it is correct. For any word that is written to SRAM, the memory controller calculates the SECDED value and stores it with that word in SRAM.

The SECDED checking may be enabled by setting **MEMCTL.ECCDIS** to 0b and disabled by setting **MEMCTL.ECCDIS** to 1b. By default, SECDED is disabled after reset.

Single-bit errors may be detected and corrected by the memory controller. If a single-bit error is detected, then the memory controller will set the **MEMSTATUS.SE** bit to 1b. If the **MEMCTL.SEIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC will be asserted. The user may clear **MEMSTATUS.SE** by writing a 1b to it.

Dual-bit errors are able to be detected, but not able to be corrected by the memory controller. If a dual-bit error is detected, the memory controller will set the **MEMSTATUS.DE** bit to 1b and assert the IRQ_MEMCTL signal to the NVIC. This interrupt signal is a NMI.

Note that any time the user wants to write the **MEMCTL** register, they must first write the **FLASHLOCK** with the value D513 B490h.

6.4.5 FLASH Read Cache

The maximum HCLK frequency in the PAC55XX is 150MHz and the maximum FLASH frequency is 25MHz. To help provide faster memory access, the memory controller provides a small cache for reading instructions to provide a faster instruction read frequency.

The cache's performance is application-dependent. In general, the cache performs better if less jump or branch instructions are performed.

If the FLASH read execution speed is not fast enough, then the user has the option of running parts of their program from SRAM, where it can execute up to 150MHz.

The FLASH read cache may be enabled by setting **MEMCTL.CACHEDIS** to 0b and disabled by setting **MEMCTL.CACHEDIS** to 1b. By default, the cache is disabled after a reset.

6.4.6 Erasing a FLASH Page

To erase one page of FLASH, the user must follow these steps:

- Write **FLASHLOCK** to the value 43DF 140Ah to allow a FLASH page erase operation
- Write the FLASH page number to the **FLASHPAGE** register
- Write **FLASHERASE** to the value 8C79 9CA7h

The memory controller will set **MEMSTATUS.EBUSY** to a 1b while the erase operation is in progress. This bit will be cleared as soon as the FLASH erase operation is complete.

6.4.7 Erase all FLASH Pages

To erase all pages of FLASH, the user must follow these steps:

- Write **FLASHLOCK** to the value 43DF 140Ah to allow FLASH all page erase operation
- Write **FLASHERASE** to the value 09EE 76C9h

The memory controller will set **MEMSTATUS.EBUSY** to a 1b while the erase operation is in progress. This bit will be cleared as soon as the FLASH erase operation is complete.

6.4.8 Erase INFO-3

To erase the INFO-3 FLASH, the user must follow the following steps:

- Write **FLASHLOCK** to the value 43DF 140Ah to allow FLASH all page erase operation
- Write **FLASEIERASE** to the value 1266 FF45h

The memory controller will set **MEMSTATUS.EBUSY** to a 1b while the erase operation is in progress. This bit will be cleared as soon as the FLASH erase operation is complete.

6.4.9 Writing FLASH

Once FLASH memory is erased, FLASH may be written using the AHB bus. Before writing FLASH to an erased memory, the user should set the **FLASHLOCK** register to a value of 43DF 140Ah.

The memory controller updates FLASH on aligned 128-bit boundaries. The user may write contiguous 32-bit words of FLASH through the AHB bus. When the memory controller receives the fourth 32-bit write to FLASH that completes an aligned 128-bit segment, the memory controller will write all 128-bits at one time to FLASH memory.

During this process, the memory controller will set **MEMSTATUS.WBUSY** to a 1b. When the FLASH has been updated, this will bit will be cleared. The user should wait for 10µs after **MEMSTATUS.WBUSY** is cleared to perform the next transaction to FLASH (read or write).

6.5 Code Protection

The PAC55XX Memory Controller provides a programmable Code Protection function that allows the user to configure several of levels of security depending on the application needs.

6.5.1 Code Protection Variables

Code Protection is configured by variables in INFO-1 and INFO-2 that are used to control the behavior and security of the various Code Protection levels.

INFO2.SECEN – Enables Code Protection

INFO2.SWDFUSE – Fuse to permanently disable SWD/JTAG access to the device

INFO-3 contains additional variables as shown below.

Table 6-4 INFO-3 Format for Code Protection

ADDRESS	BYTE OFFSET			
	3	2	1	0
0010 0800h	RMASK			
0010 0804h				
0010 0808h				
0010 080Ch				
0010 0810h – 0010 081F	Reserved			
0010 0820h	WMASK			
0010 0824h				
0010 0828h				
0010 082Ch				
0010 0830h – 0010 083Ch	Reserved			
0010 0840h	Reserved		SECLEVEL	IMASK

INFO3.RMASK is a 128-bit, bit-mask for each of the 128 segments of FLASH memory. The LSB of 0010 0800h represents segment 0 and the MSB of 0010 080Fh represents segment 127. This mask specifies which segments the application allows for read-access when in levels 1, 2 and 3 of code protection as described below.

INFO3.WMASK is a 128-bit, bit-mask for each of the 128 segments of FLASH memory. The LSB of 0010 0820h represents segment 0 and the MSB of 0010 082Fh represents segment 127. This mask specifies which segments the application allows for write-access when in levels 1, 2 and 3 of code protection as described below.

INFO3.IMASK[1] specifies the read access for INFO-3.

INFO3.IMASK[3] specifies the write access for INFO-3.

INFO2.SECLEVEL specifies the code protection security level as described below.

Table 6-5 Code Protection Levels

LEVEL	INFO3.SECLEVEL	INFO2.SWDFUSE	NAME	DESCRIPTION
0	FFh	Any value except DEAF DEAFh	UNLOCKED	• No restrictions
1	FEh	Any value except DEAF DEAFh	RW PROTECTION	• SWD/JTAG Enabled • No FLASH erase operations allowed • If in ARM debug mode, restrict access to FLASH, according to **INFO3.RMASK** and **INFO3.WMASK** • If in ARM debug mode, restrict access to INFO-2, according to **INFO3.IMASK**
2	FCh	Any value except DEAF DEAFh	SWD DISABLED	• SWD/JTAG Disabled • No FLASH erase operations allowed • If in ARM debug mode, restrict access to FLASH, according to **INFO3.RMASK** and **INFO3.WMASK** • If in ARM debug mode, restrict access to INFO-2, according to **INFO3.IMASK** • May re-enable SWD/JTAG temporarily
3	n/a	DEAF DEAFh	PERMANENT SWD DISABLE	• SWD/JTAG Permanently Disabled • No FLASH erase operations allowed • If in ARM debug mode, restrict access to FLASH, according to **INFO3.RMASK** and **INFO3.WMASK** • If in ARM debug mode, restrict access to INFO-2, according to **INFO3.IMASK**

When using code protection, the INFO-3 memory is used by the Code Protection feature and is not usable by the application. The Memory Controller may erase the contents of INFO-3 when the security levels are changed. The table below shows the format of INFO-3 when used with code protection enabled.

When code protection is enabled, segments that are marked as read-only will also be remapped to 0020 0000h from 0000 0000h.

6.5.2 Code Security Levels

To enable code security, first set the **INFO2.SECEN** to a value other than 0xFF.

*NOTE: When writing the **INFO2.SECEN** it may not be undone and code security will be enabled from that point on.*

After enabling code security, write **INFO3.SECLEVEL** or **INFO2.SWDFUSE** to the correct value to enable the desired security level.

6.5.3 Level 0 – UNLOCKED

When in level 0, Code Protection is in the *UNLOCKED* state.

In this state, there are no restrictions on read or writing transactions to FLASH memory.

6.5.4 Level 1 – RW PROTECTION

When in level 1, Code Protection is in the *RW PROTECTION* state.

In this state, SWD/JTAG remains enabled. All FLASH erase operations by the user are disabled.

After SWD/JTAG has been initialized, the Memory Controller restricts access to FLASH memory according to the **INFO3.RMASK** and **INFO3.WMASK** fields.

When the Memory Controller receives a read request for FLASH memory, if the corresponding bit for the segment of FLASH is cleared in **INFO3.RMASK**, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for FLASH memory, if the corresponding bit for the segment of FLASH is cleared in **INFO3.WMASK**, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a read request for INFO-2, if the **INFO3.IMASK[0]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a read request for INFO-3, if the **INFO3.IMASK[1]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for INFO-2, if the **INFO3.IMASK[2]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for INFO-3, if the **INFO3.IMASK[3]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

To revert to level 0, the user must perform a mass program and INFO-3 erase. This will erase program FLASH followed by INFO-3 FLASH.

6.5.5 Level 2 – SWD DISABLED

When in level 2, Code Protection is in the *SWD DISABLED* state, and SWD/JTAG access is disabled. All FLASH erase operations are disabled. After a reset, the device will be returned to Security Level 0.

When the device is in level 2, SWD access may be temporarily enabled using the **SWDUNLOCK** register. The procedure to enable SWD is to write the **SWDUNLOCK** register a first time with a 32-bit value during device initialization. After this first write, SWD remains disabled. Then, later, when it's desired to unlock SWD, the **SWDUNLOCK** register should be written a second time with the same 32-bit value previously written. The **SWDUNLOCK** register must be written from code running from SRAM (using the *__ramfunc* intrinsic). After writing **SWDUNLOCK**, the program must wait for the **MEMSTATUS.WBUSY** bit to be 0. Then, an additional 20µs delay must be added for the operation to be completed correctly.

When SWD is unlocked, **INFO3.RMASK**, **INFO3.WMASK** and **INFO3.IMASK** values will be in effect and protecting FLASH.

When the Arm® Cortex® MCU is in debug mode, the Memory Controller restricts access to FLASH memory according to the **INFO3.RMASK** and **INFO3.WMASK** fields.

When the Memory Controller receives a read request for FLASH memory, if the corresponding bit for the segment of FLASH is not cleared in **INFO3.RMASK**, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for FLASH memory, if the corresponding bit for the segment of FLASH is not cleared in **INFO3.WMASK**, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a read request for INFO-2, if the **INFO3.IMASK[0]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a read request for INFO-3, if the **INFO3.IMASK[1]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for INFO-2, if the **INFO3.IMASK[2]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for INFO-3, if the **INFO3.IMASK[3]** is not cleared, then the transaction is denied. The **MEMSTATUSL.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

To revert to level 0, the user must perform a mass program and INFO-3 erase. This will erase program FLASH followed by INFO-3 FLASH.

6.5.6 Level 3 – PERMANENT SWD DISABLED

When in level 3, Code Protection is in the *PERMANENT SWD DISABLED* state. SWD and JTAG may not be enabled again.

In this state, SWD/JTAG access is disabled. All FLASH erase operations by the user are disabled.

When the Arm® Cortex® MCU is in debug mode, the Memory Controller restricts access to FLASH memory according to the **INFO3.RMASK** and **INFO3.WMASK** fields.

When the Memory Controller receives a read request for FLASH memory, if the corresponding bit for the segment of FLASH is not cleared in **INFO3.RMASK**, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for FLASH memory, if the corresponding bit for the segment of FLASH is not cleared in **INFO3.WMASK**, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a read request for INFO-2, if the **INFO3.IMASK[0]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a read request for INFO-3, if the **INFO3.IMASK[1]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for INFO-2, if the **INFO3.IMASK[2]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

When the Memory Controller receives a write request for INFO-3, if the **INFO3.IMASK[3]** is not cleared, then the transaction is denied. The **MEMSTATUS.INVADDR** bit is set to 1b. If the **MEMCTL.INVADDRIE** bit is set to 1b, then the IRQ_MEMCTL signal to the NVIC is asserted.

6.6 Register Summary

Table 6-6 FLASH Controller Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
MEMCTL	400D 0000h	Memory Controller Configuration	RW	0032 0045h
MEMSTATUS	400D 0004h	Memory Controller Status	RW	0000 0000h
FLASHLOCK	400D 0008h	FLASH Lock Access	RW	0000 0000h
FLASHPAGE	400D 000Ch	FLASH Page	RW	0000 0000h
SWDUNLOCK	400D 0010h	SWD Unlock	RW	0000 0000h
FLASHERASE	400D 0020h	FLASH Erase	RW	0000 0000h

6.7 Register Detail

6.7.1 MEMCTL

Register 6-1 MEMCTL (Memory Controller Configuration, 400D 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:23	**Reserved**	RO	0	Reserved
22	**MCLKSEL**	RW	0	MCLK MUX Select: 0b: ROSCCLK 1b: MCLK
21	**CACHEDIS**	RW	1	FLASH Read Cache Disable: 0b: Read Cache Enabled 1b: Read Cache Disabled
20	**ECCDIS**	RW	1	SECDED (ECC) Disable: 0b: SECDED Enabled 1b: SECDED Disabled
19	**STDBY**	RW	0	FLASH Standby Mode: 0b: normal mode 1b: standby mode
18	**INVADDRIE**	RW	0	Invalid Memory Access Interrupt Enable: 0b: disabled 1b: enabled
17	**DEIE**	RW	1	Dual Bit Detection Interrupt Enable (ECC): 0b: disabled 1b: enabled
16	**SEIE**	RW	0	Single Bit Detection Interrupt Enable (ECC): 0b: disabled 1b: enabled
15:10	**Reserved**	RO	0	Reserved
9:8	**WRITEWORDCNT**	RW	0	Write Data Counter: Write to 0 to reset the write data buffer for FLASH.
7:4	**MCLKDIV**	RW	0100b	MCLK Divider for FLASH controller: 0000b: /1 0001b: /2 0010b: /3 0011b: /4 0100b: /5 0101b: /6 0110b: /7 0111b: /8 1000b: /9 1001b: /10 1010b: /11 1011b: /12 1100b: /13 1101b: /14 1110b: /15 1111b: /16
3:0	**WSTATE**	RW	0101b	FLASH Read Wait States: 0000b: 1 wait states 0001b: 2 wait state 0010b: 3 wait states

				0011b: 4 wait states 0100b: 5 wait states 0101b: 6 wait states 0110b: 7 wait states 0111b: 8 wait states 1000b: 9 wait states 1001b: 10 wait states 1010b: 11 wait states 1011b: 12 wait states 1100b: 13 wait states 1101b: 14 wait states 1110b: 15 wait states 1111b: 16 wait states

6.7.2 MEMSTATUS

Register 6-2 MEMSTATUS (Memory Controller Status, 400D 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:19	**Reserved**	RO	0	Reserved
18	**INVADDR**	W1C	0	Invalid Address Fetch Flag: 0b: no flag 1b: flag
17	**DE**	W1C	0	Double-bit Detection Flag (ECC): 0b: no flag 1b: flag
16	**SE**	W1C	0	Single-bit Detection Flag (ECC): 0b: no flag 1b: flag
15:10	**Reserved**	RO	0	Reserved
9:8	**WRITEWORDCNT**	RO	0	Number of bytes written to FLASH for the write data buffer: 00b: 4 bytes 01b: 8 bytes 10b: 12 bytes 11b: 16 bytes
7:2	**Reserved**	RO	0	Reserved
1	**EBUSY**	RO	0	Erase Busy: 0b: Erase operation not in progress 1b: Erase operation in progress
0	**WBUSY**	RO	0	Write Busy: 0b: Write operation not in progress 1b: Write operation in progress

6.7.3 FLASHLOCK

Register 6-3 FLASHLOCK (Memory Controller Status, 400D 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**FLASHLOCK**	RW	0	FLASHLOCK value for Memory Controller Operations: `43DF 140Ah`: Allow write an erase operations to FLASH `D513 B490h`: Allow write to **MEMCTL** register `79B4 F762h`: Allow write access to INFO2.SWDFUSE to permanently disable SWD

6.7.4 FLASHPAGE

Register 6-4 FLASHPAGE (FLASH Page, 400D 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:7	**Reserved**	RO	0	Reserved
6:0	**PAGE**	RW	0	FLASH Page Selection: 000 0000b: Page 0, 0000 0000h to 0000 03FFh 000 0001b: Page 1, 0000 0400h to 0000 0800h 000 0010b: Page 2, 0000 0800h to 0000 01BFFh … 111 1101b: Page 125, 0001 F400h to 0001 F7FFh 111 1110b: Page 126, 0001 F800h to 0001 FBFFh 111 1111b: Page 127, 0001 FC00h to 0001 FFFFh

6.7.5 SWDUNLOCK

Register 6-5 SWDUNLOCK (SWD Unlock, 400D 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**KEY_SWDUNLOCK**	WO	0	Writing the last written value to this register will unlock the SWD port.

6.7.6 FLASHERASE

Register 6-6 FLASHERASE (FLASH Erase, 400D 0020h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**KEY**	RW	0	Writing the value to this register will start the given erase operation: Page Erase: 8C79 9CA7h Mass Page Erase: 09EE 76C9h INFO-3 Erase: 1266 FF45h Reading this register will always return 0.

7 GPIO and DPM

7.1 Overview

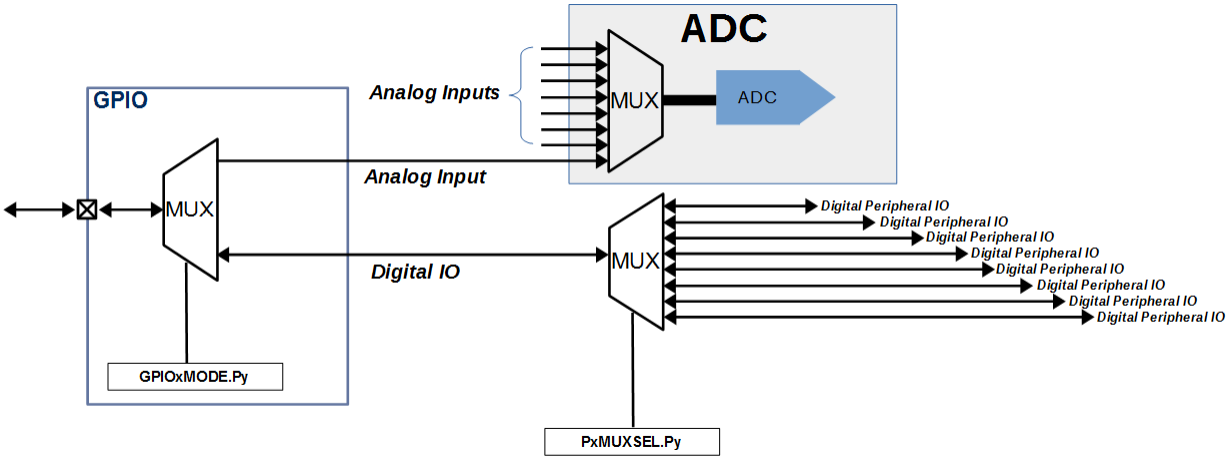
The GPIO and Digital Peripheral MUX (DPM) on the PAC55XX allows configuration of the GPIO drive strength, pull-up/pull-down and peripheral connection to the IO pins.

Each digital peripheral has several digital signals that are used in its operation. For example, a USART in SPI mode will have a SPI clock, slave select, input and output data.

In order to support the most flexible assignment of digital peripheral IOs to device pins, the PAC55XX has a highly flexible Digital Peripheral MUX. The Digital Peripheral MUX allows the assignment of peripheral functions to device pins.

Each GPIO in the PAC55XX has the capability to perform digital IO and some GPIO can act as an Analog Input for the ADC. See the diagram below for an example.

Figure 7-1 GPIO and Peripheral MUX Block Diagram



The GPIO can select if it acts as a Digital IO or Analog Input by the **GPIOxMODE.Py** register.

If **GPIOxMODE.Py** = 00b (Analog Input), then the GPIO acts as an analog input and the signal is connected to the analog MUX in the ADC.

If the GPIOxMODE.Py is not 00b (High-Impedance Digital Input, Push-pull Output, Open-drain Output) then the GPIO acts as a digital input and connects the signal to the Digital Peripheral MUX. This MUX can select which peripheral function is connected to the GPIO by the **PxMUXS.Py** register.

The following section shows the set of peripheral functions that are available in the Digital Peripheral MUX.

7.2 GPIO Drive Strength

The registers to configure GPIO Drive Strength are in the SCC module as shown below. The description of how to use these registers is given in the following GPIO sections:

- GPIOA Drive Strength

Each GPIOB pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOBMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOB pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOBCLKSYNC.Px** = 0b). To enable GPIOB Clock Synchronization set **GPIOBCLKSYNC.Px** = 1b.

- GPIOB Drive Strength

Each GPIOC pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOCMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOC pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOCCLKSYNC.Px** = 0b). To enable GPIOC Clock Synchronization set **GPIOCCLKSYNC.Px** = 1b.

- GPIOC Drive Strength

Each GPIOD pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIODMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOD pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIODCLKSYNC.Px** = 0b). To enable GPIOD Clock Synchronization set **GPIODCLKSYNC.Px** = 1b.

- GPIOD Drive Strength

Each GPIOE pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOEMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOE pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOECLKSYNC.Px** = 0b). To enable GPIOE Clock Synchronization set **GPIOECLKSYNC.Px** = 1b.

- GPIOE Drive Strength

Each GPIOF pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOFMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOF pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOFCLKSYNC.Px** = 0b). To enable GPIOF Clock Synchronization set **GPIOFCLKSYNC.Px** = 1b.

- GPIOF Drive Strength

Each GPIOG pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOGMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOG pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOGCLKSYNC.Px** = 0b). To enable GPIOG Clock Synchronization set **GPIOGCLKSYNC.Px** = 1b.

- GPIOG Drive Strength

7.3 GPIO Pull-up and Pull-down Configuration

The registers to configure GPIO weak pull-up and pull-down as well as Schmitt Trigger Input are in the SCC module as shown below.

The description of how to use these registers is given in the following GPIO sections:

- GPIOA Weak Pull-up and Pull-down
- GPIOB Weak Pull-up and Pull-down
- GPIOC Weak Pull-up and Pull-down
- GPIOD Weak Pull-up and Pull-down
- GPIOE Weak Pull-up and Pull-down
- GPIOF Weak Pull-up and Pull-down
- GPIOG Weak Pull-up and Pull-down

7.4 Digital Peripheral Functions

There are many digital peripherals in the PAC55XX. The sections below discuss each peripheral, and what digital IO signals are used in that peripheral.

For certain peripherals, the peripheral signals can be discretely assigned to the IO (like PWM timers). For other peripheral functions, the entire set needs to be assigned for the peripheral function to work (like a communication module).

Table 7-1 Peripheral Functions

PERIPHERAL GROUP	SIGNAL NAME	DESCRIPTION
SYSTEM	FRCLK	Free-running clock (output from SCC)
	EXTCLK	External Clock Input (input to SCC)
DEBUG	TMS/SWDIO	JTAG TMS or SWD IO (Debug Port)
	TCK/SWDCLK	JTAG clock or SWD clock (Debug Port)
	TDI	JTAG data input (Debug Port)
	TDO	JTAG data output (Debug Port)
ETM	TRACECLK	ETM Instruction Trace Clock (ETM)
	TRACED0	ETM Instruction Trace Data 0 (ETM)
	TRACED1	ETM Instruction Trace Data 1 (ETM)
	TRACED2	ETM Instruction Trace Data 2 (ETM)
	TRACED3	ETM Instruction Trace Data 3 (ETM)
EMUX	EMUXC	EMUX Clock (ADC DTSE)
	EMUXD	EMUX Data (ADC DTSE)
I2C	I2CSCL	I2C Serial Clock (I2C)
	I2CSDA	I2C Serial Data (I2C)
CAN	CANTXD	CAN controller TX data (CAN)
	CANRXD	CAN controller RX data (CAN)
USARTA	USACLK	USARTA Clock
	USAMOSI/UARTTX	USARTA SPI MOSI, UART TX
	USAMISO/UARTRX	USARTA SPI MISO, UART RX
	USASCLK	USARTA SPI SCLK
	USASS	USARTA SPI Slave Select
USARTB	USBCLK	USARTB Clock
	USBMOSI/UARTTX	USARTB SPI MOSI, UART TX
	USBMISO/UARTRX	USARTB SPI MISO, UART RX
	USBSCLK	USARTB SPI SCLK

	USBSS	USARTB SPI Slave Select
USARTC	USCCLK	USARTC Clock
	USCMOSI/UARTTX	USARTC SPI MOSI, UART TX
	USCMISO/UARTRX	USARTC SPI MISO, UART RX
	USCSCLK	USARTC SPI SCLK
	USCSS	USARTC SPI Slave Select
USARTD	USDCLK	USARTD Clock
	USDMOSI/UARTTX	USARTD SPI MOSI, UART TX
	USDMISO/UARTRX	USARTD SPI MISO, UART RX
	USDSCLK	USARTD SPI SCLK
	USDSS	USARTD SPI Slave Select
TIMERA	TAPWM0	Timer A PWM output/CCR0
	TAPWM1	Timer A PWM output/CCR1
	TAPWM2	Timer A PWM output/CCR2
	TAPWM3	Timer A PWM output/CCR3
	TAPWM4	Timer A PWM output/CCR4
	TAPWM5	Timer A PWM output/CCR5
	TAPWM6	Timer A PWM output/CCR6
	TAPWM7	Timer A PWM output/CCR7
	TAQEPPHA	Timer A QEP Phase A Input
	TAQEPPHB	Timer A QEP Phase B Input
	TAQEPIDX	Timer A QEP Index Input
TIMERB	TBPWM0	Timer B PWM output/CCR0
	TBPWM1	Timer B PWM output/CCR1
	TBPWM2	Timer B PWM output/CCR2
	TBPWM3	Timer B PWM output/CCR3
	TBPWM4	Timer B PWM output/CCR4
	TBPWM5	Timer B PWM output/CCR5
	TBPWM6	Timer B PWM output/CCR6
	TBPWM7	Timer B PWM output/CCR7
	TBQEPPHA	Timer B QEP Phase A Input
	TBQEPPHB	Timer B QEP Phase B Input
	TBQEPIDX	Timer B QEP Index Input
TIMERC	TCPWM0	Timer C PWM output/CCR0

	TCPWM1	Timer C PWM output/CCR1
	TCPWM2	Timer C PWM output/CCR2
	TCPWM3	Timer C PWM output/CCR3
	TCPWM4	Timer C PWM output/CCR4
	TCPWM5	Timer C PWM output/CCR5
	TCPWM6	Timer C PWM output/CCR6
	TCPWM7	Timer C PWM output/CCR7
	TCQEPPHA	Timer C QEP Phase A Input
	TCQEPPHB	Timer C QEP Phase B Input
	TCQEPIDX	Timer C QEP Index Input
TIMERD	TDPWM0	Timer D PWM output/CCR0
	TDPWM1	Timer D PWM output/CCR1
	TDPWM2	Timer D PWM output/CCR2
	TDPWM3	Timer D PWM output/CCR3
	TDPWM4	Timer D PWM output/CCR4
	TDPWM5	Timer D PWM output/CCR5
	TDPWM6	Timer D PWM output/CCR6
	TDPWM7	Timer D PWM output/CCR7
	TDQEPPHA	Timer D QEP Phase A Input
	TDQEPPHB	Timer D QEP Phase B Input
	TDQEPIDX	Timer D QEP Index Input

7.5 Peripheral MUX Settings

The table below shows the Digital Peripheral MUX Settings for each GPIO in the PAC55XX according to the setting of the **PxMUXSEL** register for the given GPIO.

Items in the table that are in boldface type are the default values.

Table 7-2 Peripheral MUX Settings

PORT	Pin	PxMUXSEL.Py							
		000b	001b	010b	011b	100b	101b	110b	111b
PA	**P0**	**GPIOA0**							
	P1	GPIOA1	**EMUXD**						
	P2	GPIOA2	**EMUXC**						
	P3	GPIOA3	**USASCLK**	USBSCLK					
	P4	GPIOA4	**USAMOSI**	USBMOSI					
	P5	GPIOA5	**USAMISO**	USBMISO					
	P6	GPIOA6	**USASS**	USBSS					
	P7	**GPIOA7**							
PB	**P0**	**GPIOB0**	TAPWM0	TBPWM0		TCPWM0	TDPWM0		
	P1	**GPIOB1**	TAPWM1	TBPWM1		TCPWM1	TDPWM1		
	P2	**GPIOB2**	TAPWM2	TBPWM2		TCPWM2	TDPWM2		
	P3	**GPIOB3**	TAPWM3	TBPWM3		TCPWM3	TDPWM3		
	P4	**GPIOB4**	TAPWM4	TBPWM4	TCPWM0	TCPWM4	TDPWM4		
	P5	**GPIOB5**	TAPWM5	TBPWM5	TCPWM1	TCPWM5	TDPWM5		
	P6	**GPIOB6**	TAPWM6	TBPWM6	TCPWM2	TCPWM6	TDPWM6		
	P7	**GPIOB7**	TAPWM7	TBPWM7	TCPWM3	TCPWM7	TDPWM7		
PC	**P0**	**GPIOC0**	TBPWM0	TCPWM0	TBQEPIDX	USBMOSI	USCSCLK	CANRXD	I2CSCL
	P1	**GPIOC1**	TBPWM1	TCPWM1	TBQEPPHA	USBMISO	USCSS	CANTXD	I2CSDA
	P2	**GPIOC2**	TBPWM2	TCPWM2	TBQEPPHB	USBSCLK	USCMOSI		EMUXD
	P3	**GPIOC3**	TBPWM3	TCPWM3		USBSS	USCMISO		EMUXC
	P4	**GPIOC4**	TBPWM4	TCPWM4	TCQEPIDX	USBMOSI	USCSCLK	CANRXD	I2CSCL
	P5	**GPIOC5**	TBPWM5	TCPWM5	TCQEPPHA	USBMISO	USCSS	CANTXD	I2CSDA
	P6	**GPIOC6**	TBPWM6	TCPWM6	TCQEPPHB	USBSCLK	USCMOSI		EMUXD
	P7	**GPIOC7**	TBPWM7	TCPWM7		USBSS	USCMISO	FRCLK	EMUXC
PD	**P0**	**GPIOD0**	TBPWM0	TCPWM0	TDQEPIDX		USCSCLK	CANTXD	EMUXD
	P1	**GPIOD1**	TBPWM1	TCPWM1	TDQEPPHA		USCSS	CANRXD	EMUXC

	P2	**GPIOD2**	TBPWM2	TCPWM2	TDQEPPHB		USCMOSI		
	P3	**GPIOD3**	TBPWM3	TCPWM3			USCMISO	FRCLK	TRACED3
	P4	**GPIOD4**	TBPWM4	TCPWM4	TDQEPIDX	TBQEPIDX	USDSCLK	TRACED3	USDMOSI
	P5	**GPIOD5**	TBPWM5	TCPWM5	TDQEPPHA	TBQEPPHA	USDSS	CANRXD	USDMISO
	P6	**GPIOD6**	TBPWM6	TCPWM6	TDQEPPHB	TBQEPPHB	USDMOSI	CANTXD	I2CSDA
	P7	**GPIOD7**	TBPWM7	TCPWM7			USDMISO	CANRXD	I2CSCL
PE	**P0**	**GPIOE0**	TCPWM4	TDPWM0	TAQEPIDX	TBQEPIDX	USCSCLK	I2CSCL	EMUXC
	P1	**GPIOE1**	TCPWM5	TDPWM1	TAQEPPHA	TBQEPPHA	USCSS	I2CSDA	EMUXD
	P2	**GPIOE2**	TCPWM6	TDPWM2	TAQEPPHB	TBQEPPHB	USCMOSI	CANRXD	EXTCLK
	P3	**GPIOE3**	TCPWM7	TDPWM3	FRCLK		USCMISO	CANTXD	
	P4	**GPIOE4**	TCPWM4	TDPWM4	TDQEPIDX	USBSCLK	USDMOSI	I2CSCL	
	P5	**GPIOE5**	TCPWM5	TDPWM5	TDQEPPHA	USBSS	USDMISO	I2CSDA	
	P6	**GPIOE6**	TCPWM6	TDPWM6	TDQEPPHB	USBMOSI	USDSCLK	CANRXD	
	P7	**GPIOE7**	TCPWM7	TDPWM7		USBMISO	USDSS	CANTXD	
PF	**P0**	GPIOF0	TCPWM0	TDPWM0	**TCK/SWDCLK**	TBQEPIDX	USBSCLK	TRACED2	TRACECLK
	P1	GPIOF1	TCPWM1	TDPWM1	**TMS/SWDIO**	TBQEPPHA	USBSS	TRACED1	TRACED0
	P2	GPIOF2	TCPWM2	TDPWM2	**TDI**	TBQEPPHB	USBMOSI	TRACED0	TRACED1
	P3	GPIOF3	TCPWM3	TDPWM3	**TDO**	FRCLK	USBMISO	TRACECLK	TRACED2
	P4	**GPIOF4**	TCPWM4	TDPWM4		TCQEPIDX	USDSCLK	TRACED3	EMUXC
	P5	**GPIOF5**	TCPWM5	TDPWM5		TCQEPPHA	USDSS		EMUXD
	P6	**GPIOF6**	TCPWM6	TDPWM6		TCQEPPHB	USDMOSI	CANRXD	I2CSCL
	P7	**GPIOF7**	TCPWM7	TDPWM7			USDMISO	CANTXD	I2CSDA
PG	**P0**	**GPIOG0**	TCPWM0	TDPWM0	EMUXC		USDSCLK	TRACECLK	TCQEPIDX
	P1	**GPIOG1**	TCPWM1	TDPWM1	EMUXD		USDSS	TRACED0	TCQEPPHA
	P2	**GPIOG2**	TCPWM2	TDPWM2	FRCLK		USDMOSI	TRACED1	TCQEPPHB
	P3	**GPIOG3**	TCPWM3	TDPWM3			USDMISO	TRACED2	
	P4	**GPIOG4**	TCPWM4	TDPWM4	EMUXD	I2CSCL	USDSS	TRACED3	TDQEPIDX
	P5	**GPIOG5**	TCPWM5	TDPWM5	EMUXC		USDMOSI	CANRXD	TDQEPPHA
	P6	**GPIOG6**	TCPWM6	TDPWM6	I2CSDA		USDMISO	CANTXD	TDQEPPHB
	P7	**GPIOG7**		TDQEPIDX			USDSCLK		

7.6 Register Summary

Table 7-3 GPIO and DPM Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
PAMUXSEL	400D 040Ch	PA Peripheral MUX Select	RW	0111 1110h
PBMUXSEL	400D 0410h	PB Peripheral MUX Select	RW	0000 0000h
PCMUXSEL	400D 0414h	PC Peripheral MUX Select	RW	0000 0000h
PDMUXSEL	400D 0418h	PD Peripheral MUX Select	RW	0000 0000h
PEMUXSEL	400D 041Ch	PE Peripheral MUX Select	RW	0000 0000h
PFMUXSEL	400D 0420h	PF Peripheral MUX Select	RW	0000 3333h
PGMUXSEL	400D 0424h	PG Peripheral MUX Select	RW	0000 0000h
PAPUEN	400D 0428h	PA Weak Pull-up Enable	RW	0000 0000h
PBPUEN	400D 042Ch	PB Weak Pull-up Enable	RW	0000 0000h
PCPUEN	400D 0430h	PC Weak Pull-up Enable	RW	0000 0000h
PDPUEN	400D 0434h	PD Weak Pull-up Enable	RW	0000 0000h
PEPUEN	400D 0438h	PE Weak Pull-up Enable	RW	0000 0000h
PFPUEN	400D 043Ch	PF Weak Pull-up Enable	RW	0000 0000h
PGPUEN	400D 0440h	PG Weak Pull-up Enable	RW	0000 0000h
PAPDEN	400D 0444h	PA Weak Pull-down Enable	RW	0000 0000h
PBPDEN	400D 0448h	PB Weak Pull-down Enable	RW	0000 0000h
PCPDEN	400D 044Ch	PC Weak Pull-down Enable	RW	0000 0000h
PDPDEN	400D 0450h	PD Weak Pull-down Enable	RW	0000 0000h
PEPDEN	400D 0454h	PE Weak Pull-down Enable	RW	0000 0000h
PFPDEN	400D 0458h	PF Weak Pull-down Enable	RW	0000 0000h
PGPDEN	400D 045Ch	PG Weak Pull-down Enable	RW	0000 0000h
PADS	400D 0460h	PA Drive Strength/Schmitt Trigger	RW	0000 0000h
PBDS	400D 0464h	PB Drive Strength/Schmitt Trigger	RW	0000 0000h
PCDS	400D 0468h	PC Drive Strength/Schmitt Trigger	RW	0000 0000h
PDDS	400D 046Ch	PD Drive Strength/Schmitt Trigger	RW	0000 0000h
PEDS	400D 0470h	PE Drive Strength/Schmitt Trigger	RW	0000 0000h
PFDS	400D 0474h	PF Drive Strength/Schmitt Trigger	RW	0000 0000h
PGDS	400D 0478h	PG Drive Strength/Schmitt Trigger	RW	0000 0000h

7.6.1 PAMUXSEL

Register 7-1 PAMUXSEL (PA Peripheral MUX Select, 400D 040Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**Reserved**	RO	0b	Reserved
30:28	**P7**	RW	000b	Pin 7 MUX Select: 000b: GPIOA7 [default] 001b-111b: Reserved
27	**Reserved**	RO	0b	Reserved
26:24	**P6**	RW	001b	Pin 6 MUX Select: 000b: GPIOA6 001b: USASS [default] 010b: USBSS 011b-111b: Reserved
23	**Reserved**	RW	0b	Reserved
22:20	**P5**	RW	001b	Pin 5 MUX Select: 000b: GPIOA5 001b: USAMISO [default] 010b: USBMISO 011b-111b: Reserved
19	**Reserved**	RO	0b	Reserved
18:16	**P4**	RW	001b	Pin 4 MUX Select: 000b: GPIOA4 001b: USAMOSI [default] 010b: USBMOSI 011b-111b: Reserved
15	**Reserved**	RO	0b	Reserved
14:12	**P3**	RW	001b	Pin 3 MUX Select: 000b: GPIOA3 001b: USASCLK [default] 010b: USBSCLK 011b-111b: Reserved
11	**Reserved**	RO	0b	Reserved
10:8	**P2**	RW	001b	Pin 2 MUX Select: 000b: GPIOA2 001b: EMUXC [default] 010b-111b: Reserved
7	**Reserved**	RO	0b	Reserved
6:4	**P1**	RW	001b	Pin 1 MUX Select: 000b: GPIOA1 001b: EMUXD [default] 010b-111b: Reserved
3	**Reserved**	RO	0b	Reserved
2:0	**P0**	RW	000b	Pin 0 MUX Select: 000b: GPIOA0 [default] 001b-111b: Reserved

7.6.2 PBMUXSEL

Register 7-2 PBMUXSEL (PB Peripheral MUX Select, 400D 0410h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**Reserved**	RO	0b	Reserved
30:28	**P7**	RW	000b	Pin 7 MUX Select: 000b: GPIOB7 [default] 001b: TAPWM7 010b: TBPWM7 011b-111b: Reserved
27	**Reserved**	RO	0b	Reserved
26:24	**P6**	RW	000b	Pin 6 MUX Select: 000b: GPIOB6 [default] 001b: TAPWM6 010b: TBPWM6 011b-111b: Reserved
23	**Reserved**	RW	0b	Reserved
22:20	**P5**	RW	000b	Pin 5 MUX Select: 000b: GPIOB5 [default] 001b: TAPWM5 010b: TBPWM5 011b-111b: Reserved
19	**Reserved**	RO	0b	Reserved
18:16	**P4**	RW	000b	Pin 4 MUX Select: 000b: GPIOB4 [default] 001b: TAPWM4 010b: TBPWM4 011b-111b: Reserved
15	**Reserved**	RO	0b	Reserved
14:12	**P3**	RW	000b	Pin 3 MUX Select: 000b: GPIOB3 [default] 001b: TAPWM3 010b: TBPWM3 011b-111b: Reserved
11	**Reserved**	RO	0b	Reserved
10:8	**P2**	RW	000b	Pin 2 MUX Select: 000b: GPIOB2 [default] 001b: TAPWM2 010b: TBPWM2 011b-111b: Reserved
7	**Reserved**	RO	0b	Reserved
6:4	**P1**	RW	000b	Pin 1 MUX Select: 000b: GPIOB1 [default] 001b: TAPWM1 010b: TBPWM1 011b-111b: Reserved

3	**Reserved**	RO	0b	Reserved
2:0	**P0**	RW	000b	Pin 0 MUX Select: 000b: GPIOB0 [default] 001b: TAPWM0 010b: TBPWM0

7.6.3 PCMUXSEL

Register 7-3 PCMUXSEL (PC Peripheral MUX Select, 400D 0414h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**Reserved**	RO	0b	Reserved
30:28	**P7**	RW	000b	Pin 7 MUX Select: 000b: GPIOC7 [default] 001b: TBPWM7 010b: TCPWM7 011b: Reserved 100b: USBSS 101b: USCMISO 110b: FRCLK 111b: EMUXC
27	**Reserved**	RO	0b	Reserved
26:24	**P6**	RW	000b	Pin 6 MUX Select: 000b: GPIOC6 [default] 001b: TBPWM6 010b: TCPWM6 011b: TCQEPPHB 100b: USBSCLK 101b: USCMOSI 110b: Reserved 111b: EMUXD
23	**Reserved**	RW	0b	Reserved
22:20	**P5**	RW	000b	Pin 5 MUX Select: 000b: GPIOC5 [default] 001b: TBPWM5 010b: TCPWM5 011b: TCQEPPHA 100b: USBMISO 101b: USCSS 110b: CANTXD 111b: I2CSDA
19	**Reserved**	RO	0b	Reserved
18:16	**P4**	RW	001b	Pin 4 MUX Select: 000b: GPIOC4 [default] 001b: TBPWM4 010b: TCPWM4 011b: TCQEPIDX 100b: USBMOSI 101b: USCSCLK 110b: CANRXD 111b: I2CSCL
15	**Reserved**	RO	0b	Reserved
14:12	**P3**	RW	000b	Pin 3 MUX Select: 000b: GPIOC3 [default] 001b: TBPWM3 010b: TCPWM3 011b: Reserved 100b: USBSS 101b: USCMISO 110b: Reserved

				111b: EMUXC
11	**Reserved**	RO	0b	Reserved
10:8	**P2**	RW	000b	Pin 2 MUX Select: 000b: GPIOC2 [default] 001b: TBPWM2 010b: TCPWM2 011b: TBQEPPHB 100b: USBSCLK 101b: USCMOSI 110b: Reserved 111b: EMUXD
7	**Reserved**	RO	0b	Reserved
6:4	**P1**	RW	000b	Pin 1 MUX Select: 000b: GPIOC1 [default] 001b: TBPWM1 010b: TCPWM1 011b: TBQEPPHA 100b: USBMISO 101b: USCSS 110b: CANTXD 111b: I2CSDA
3	**Reserved**	RO	0b	Reserved
2:0	**P0**	RW	000b	Pin 0 MUX Select: 000b: GPIOC1 [default] 001b: TBPWM0 010b: TCPWM0 011b: TBQEPIDX 100b: USBMOSI 101b: USCSCLK 110b: CANRXD 111b: I2CSCL

7.6.4 PDMUXSEL

Register 7-4 PDMUXSEL (PD Peripheral MUX Select, 400D 0418h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**Reserved**	RO	0b	Reserved
30:28	**P7**	RW	000b	Pin 7 MUX Select: 000b: GPIOD7 [default] 001b: TBPWM7 010b: TCPWM7 011b: Reserved 100b: Reserved 101b: USDMISO 110b: CANRXD 111b: I2CSCL
27	**Reserved**	RO	0b	Reserved
26:24	**P6**	RW	000b	Pin 6 MUX Select: 000b: GPIOC6 [default] 001b: TBPWM6 010b: TCPWM6 011b: TDQEPPHB 100b: TBQEPPHB 101b: USDMOSI 110b: CANTXD 111b: I2CSDA
23	**Reserved**	RW	0b	Reserved
22:20	**P5**	RW	000b	Pin 5 MUX Select: 000b: GPIOD5 [default] 001b: TBPWM5 010b: TCPWM5 011b: TDQEPPHA 100b: TBQEPPHA 101b: USDSS 110b: CANRXD 111b: USDMISO
19	**Reserved**	RO	0b	Reserved
18:16	**P4**	RW	000b	Pin 4 MUX Select: 000b: GPIOD4 [default] 001b: TBPWM4 010b: TCPWM4 011b: TDQEPIDX 100b: TBQEPIDX 101b: USDSCLK 110b: TRACED3 111b: USDMOSI
15	**Reserved**	RO	0b	Reserved
14:12	**P3**	RW	000b	Pin 3 MUX Select: 000b: GPIOD3 [default] 001b: TBPWM3 010b: TCPWM3 011b: Reserved 100b: Reserved 101b: USCMISO 110b: FRCLK

				111b: TRACED3
11	**Reserved**	RO	0b	Reserved
10:8	**P2**	RW	000b	Pin 2 MUX Select: 000b: GPIOD2 [default] 001b: TBPWM2 010b: TCPWM2 011b: TDQEPPHB 100b: Reserved 101b: USCMOSI 110b: Reserved 111b: Reserved
7	**Reserved**	RO	0b	Reserved
6:4	**P1**	RW	000b	Pin 1 MUX Select: 000b: GPIOD1 [default] 001b: TBPWM1 010b: TCPWM1 011b: TDQEPPHA 100b: Reserved 101b: USCSS 110b: CANRXD 111b: EMUXC
3	**Reserved**	RO	0b	Reserved
2:0	**P0**	RW	000b	Pin 0 MUX Select: 000b: GPIOD0 [default] 001b: TBPWM0 010b: TCPWM0 011b: TDQEPIDX 100b: Reserved 101b: USCSCLK 110b: CANTXD 111b: EMUXD

7.6.5 PEMUXSEL

Register 7-5 PEMUXSEL (PE Peripheral MUX Select, 400D 041Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**Reserved**	RO	0b	Reserved
30:28	**P7**	RW	000b	Pin 7 MUX Select: 000b: GPIOE7 [default] 001b: TCPWM7 010b: TDPWM7 011b: Reserved 100b: USBMISO 101b: USDSS 110b: CANTXD 111b: Reserved
27	**Reserved**	RO	0b	Reserved
26:24	**P6**	RW	000b	Pin 6 MUX Select: 000b: GPIOE6 [default] 001b: TCPWM6 010b: TDPWM6 011b: TDQEPPHB 100b: USBMOSI 101b: USDSCLK 110b: CANRXD 111b: Reserved
23	**Reserved**	RW	0b	Reserved
22:20	**P5**	RW	000b	Pin 5 MUX Select: 000b: GPIOE5 [default] 001b: TCPWM5 010b: TDPWM5 011b: TDQEPPHA 100b: USBSS 101b: USDMISO 110b: I2CSDA 111b: Reserved
19	**Reserved**	RO	0b	Reserved
18:16	**P4**	RW	000b	Pin 4 MUX Select: 000b: GPIOE4 [default] 001b: TCPWM4 010b: TDPWM4 011b: TDQEPIDX 100b: USBSCLK 101b: USDMOSI 110b: I2CSCL 111b: Reserved
15	**Reserved**	RO	0b	Reserved
14:12	**P3**	RW	000b	Pin 3 MUX Select: 000b: GPIOE3 [default] 001b: TCPWM7 010b: TDPWM3 011b: FRCLK 100b: Reserved 101b: USCMISO 110b: CANTXD

				111b: Reserved
11	**Reserved**	RO	0b	Reserved
10:8	**P2**	RW	000b	Pin 2 MUX Select: 000b: GPIOE2 [default] 001b: TCPWM6 010b: TDPWM2 011b: TAQEPPHA 100b: TAQEPPHB 101b: USCMOSI 110b: CANRXD 111b: EXTCLK
7	**Reserved**	RO	0b	Reserved
6:4	**P1**	RW	000b	Pin 1 MUX Select: 000b: GPIOE1 [default] 001b: TCPWM5 010b: TDPWM1 011b: TAQEPPHA 100b: TBQEPPHA 101b: USCSS 110b: I2CSDA 111b: EMUXD
3	**Reserved**	RO	0b	Reserved
2:0	**P0**	RW	000b	Pin 0 MUX Select: 000b: GPIOE0 [default] 001b: TCPWM4 010b: TDPWM0 011b: TAQEPIDX 100b: TBQEPIDX 101b: USCSCLK 110b: I2CSCL 111b: EMUXC

7.6.6 PFMUXSEL

Register 7-6 PFMUXSEL (PF Peripheral MUX Select, 400D 0420h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**Reserved**	RO	0b	Reserved
30:28	**P7**	RW	000b	Pin 7 MUX Select: 000b: GPIOF7 [default] 001b: TCPWM7 010b: TDPWM7 011b: Reserved 100b: Reserved 101b: USDMISO 110b: CANTXD 111b: I2CSDA
27	**Reserved**	RO	0b	Reserved
26:24	**P6**	RW	000b	Pin 6 MUX Select: 000b: GPIOF6 [default] 001b: TCPWM6 010b: TDPWM6 011b: Reserved 100b: TCQEPPHB 101b: USDMOSI 110b: CANRXD 111b: I2CSCL
23	**Reserved**	RW	0b	Reserved
22:20	**P5**	RW	000b	Pin 5 MUX Select: 000b: GPIOF5 [default] 001b: TCPWM5 010b: TDPWM5 011b: Reserved 100b: TCQEPPHA 101b: USDSS 110b: Reserved 111b: EMUXD
19	**Reserved**	RO	0b	Reserved
18:16	**P4**	RW	000b	Pin 4 MUX Select: 000b: GPIOF4 [default] 001b: TCPWM4 010b: TDPWM4 011b: Reserved 100b: TCQEPIDX 101b: USDSCLK 110b: TRACED3 111b: EMUXC
15	**Reserved**	RO	0b	Reserved
14:12	**P3**	RW	011b	Pin 3 MUX Select: 000b: GPIOF3 001b: TCPWM3 010b: TDPWM3 011b: TDO [default] 100b: FRCLK 101b: USBMISO 110b: TRACED2

				111b: Reserved
11	**Reserved**	RO	0b	Reserved
10:8	**P2**	RW	011b	Pin 2 MUX Select: 000b: GPIOF2 001b: TCPWM2 010b: TDPWM2 011b: TDI [default] 100b: TBQEPPHB 101b: USBMOSI 110b: TRACED1 111b: Reserved
7	**Reserved**	RO	0b	Reserved
6:4	**P1**	RW	011b	Pin 1 MUX Select: 000b: GPIOF1 001b: TCPWM1 010b: TDPWM1 011b: TMS/SWDIO [default] 100b: TBQEPPHA 101b: USBSS 110b: TRACED0 111b: Reserved
3	**Reserved**	RO	0b	Reserved
2:0	**P0**	RW	011b	Pin 0 MUX Select: 000b: GPIOF0 001b: TCPWM0 010b: TDPWM0 011b: TCK/SWDCLK [default] 100b: TBQEPIDX 101b: USBSCLK 110b: TRACECLK 111b: Reserved

7.6.7 PGMUXSEL

Register 7-7 PGMUXSEL (PG Peripheral MUX Select, 400D 0424h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**Reserved**	RO	0b	Reserved
30:28	**P7**	RW	000b	Pin 7 MUX Select: 000b: GPIOG7 [default] 001b: Reserved 010b: TDQEPIDX 011b: Reserved 100b: Reserved 101b: USDCLK 110b: Reserved 111b: Reserved
27	**Reserved**	RO	0b	Reserved
26:24	**P6**	RW	000b	Pin 6 MUX Select: 000b: GPIOG6 [default] 001b: TCPWM6 010b: TDPWM6 011b: I2CSDA 100b: Reserved 101b: USDMISO 110b: CANTXD 111b: TDQEPPHB
23	**Reserved**	RW	0b	Reserved
22:20	**P5**	RW	000b	Pin 5 MUX Select: 000b: GPIOG5 [default] 001b: TCPWM5 010b: TDPWM5 011b: EMUXC 100b: Reserved 101b: USDMOSI 110b: CANRXD 111b: TDQEPPHA
19	**Reserved**	RO	0b	Reserved
18:16	**P4**	RW	000b	Pin 4 MUX Select: 000b: GPIOG4 [default] 001b: TCPWM4 010b: TDPWM4 011b: EMUXD 100b: I2CSCL 101b: USDSS 110b: TRACED3 111b: TDQEPIDX
15	**Reserved**	RO	0b	Reserved
14:12	**P3**	RW	000b	Pin 3 MUX Select: 000b: GPIOG3 [default] 001b: TCPWM3 010b: TDPWM3 011b: Reserved 100b: Reserved 101b: USDMISO 110b: TRACED2

				111b: Reserved
11	**Reserved**	RO	0b	Reserved
10:8	**P2**	RW	000b	Pin 2 MUX Select: 000b: GPIOG2 [default] 001b: TCPWM2 010b: TDPWM2 011b: FRCLK 100b: Reserved 101b: USDMOSI 110b: TRACED1 111b: TCQEPPHB
7	**Reserved**	RO	0b	Reserved
6:4	**P1**	RW	000b	Pin 1 MUX Select: 000b: GPIOG1 [default] 001b: TCPWM1 010b: TDPWM1 011b: EMUXD 100b: Reserved 101b: USDSS 110b: TRACED0 111b: TCQEPPHA
3	**Reserved**	RO	0b	Reserved
2:0	**P0**	RW	000b	Pin 0 MUX Select: 000b: GPIOG0 [default] 001b: TCPWM0 010b: TDPWM0 011b: EMUXC 100b: Reserved 101b: USDCLK 110b: TRACECLK 111b: TCQEPIDX

7.6.8 PAPUEN

Register 7-8 PAPUEN (PA Pull-up Enable, 400D 0428h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-up enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-up enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-up enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-up enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-up enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-up enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-up enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-up enabled: 0b: disabled 1b: enabled

7.6.9 PBPUEN

Register 7-9 PBPUEN (PB Pull-up Enable, 400D 042Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-up enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-up enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-up enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-up enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-up enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-up enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-up enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-up enabled: 0b: disabled 1b: enabled

7.6.10 PCPUEN

Register 7-10 PCPUEN (PC Pull-up Enable, 400D 0430h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-up enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-up enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-up enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-up enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-up enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-up enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-up enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-up enabled: 0b: disabled 1b: enabled

7.6.11 PDPUEN

Register 7-11 PDPUEN (PD Pull-up Enable, 400D 0034h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-up enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-up enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-up enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-up enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-up enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-up enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-up enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-up enabled: 0b: disabled 1b: enabled

7.6.12 PEPUEN

Register 7-12 PEPUEN (PE Pull-up Enable, 400D 0438h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-up enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-up enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-up enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-up enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-up enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-up enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-up enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-up enabled: 0b: disabled 1b: enabled

7.6.13 PFPUEN

Register 7-13 PFPUEN (PF Pull-up Enable, 400D 043Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-up enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-up enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-up enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-up enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-up enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-up enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-up enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-up enabled: 0b: disabled 1b: enabled

7.6.14 PGPUEN

Register 7-14 PGPUEN (PG Pull-up Enable, 400D 0440h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-up enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-up enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-up enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-up enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-up enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-up enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-up enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-up enabled: 0b: disabled 1b: enabled

7.6.15 PAPDEN

Register 7-15 PAPDEN (PA Pull-down Enable, 400D 0444h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-down enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-down enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-down enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-down enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-down enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-down enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-down enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-down enabled: 0b: disabled 1b: enabled

7.6.16 PBPDEN

Register 7-16 PBPDEN (PB Pull-down Enable, 400D 0448h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-down enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-down enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-down enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-down enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-down enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-down enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-down enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-down enabled: 0b: disabled 1b: enabled

7.6.17 PCPDEN

Register 7-17 PCPDEN (PC Pull-down Enable, 400D 044Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-down enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-down enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-down enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-down enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-down enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-down enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-down enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-down enabled: 0b: disabled 1b: enabled

7.6.18 PDPDEN

Register 7-18 PDPDEN (PD Pull-down Enable, 400D 0450h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-down enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-down enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-down enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-down enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-down enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-down enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-down enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-down enabled: 0b: disabled 1b: enabled

7.6.19 PEPDEN

Register 7-19 PEPDEN (PE Pull-down Enable, 400D 0454h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-down enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-down enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-down enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-down enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-down enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-down enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-down enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-down enabled: 0b: disabled 1b: enabled

7.6.20 PFPDEN

Register 7-20 PFPDEN (PF Pull-down Enable, 400D 0458h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-down enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-down enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-down enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-down enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-down enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-down enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-down enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-down enabled: 0b: disabled 1b: enabled

7.6.21 PGPDEN

Register 7-21 PGPDEN (PG Pull-down Enable, 400D 045Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0b	Reserved
7	**P7**	RW	0b	Pin 7 Weak pull-down enabled: 0b: disabled 1b: enabled
6	**P6**	RW	0b	Pin 6 Weak pull-down enabled: 0b: disabled 1b: enabled
5	**P5**	RW	0b	Pin 5 Weak pull-down enabled: 0b: disabled 1b: enabled
4	**P4**	RW	0b	Pin 4 Weak pull-down enabled: 0b: disabled 1b: enabled
3	**P3**	RW	0b	Pin 3 Weak pull-down enabled: 0b: disabled 1b: enabled
2	**P2**	RW	0b	Pin 2 Weak pull-down enabled: 0b: disabled 1b: enabled
1	**P1**	RW	0b	Pin 1 Weak pull-down enabled: 0b: disabled 1b: enabled
0	**P0**	RW	0b	Pin 0 Weak pull-down enabled: 0b: disabled 1b: enabled

7.6.22 PADS

Register 7-22 PADS (PA Drive Strength/Schmitt Trigger, 400D 0460h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**P7ST**	RW	0b	Pin 7 Schmitt Trigger Enable: 0b: disabled 1b: enabled
30:28	**P7DS**	RW	000b	Pin 7 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
37	**P6ST**	RW	0b	Pin 6 Schmitt Trigger Enable: 0b: disabled 1b: enabled
36:24	**P6DS**	RW	000b	Pin 6 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
23	**P5ST**	RW	0b	Pin 5 Schmitt Trigger Enable: 0b: disabled 1b: enabled
22:20	**P5DS**	RW	000b	Pin 5 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
19	**P4ST**	RW	0b	Pin 4 Schmitt Trigger Enable: 0b: disabled 1b: enabled
18:16	**P4DS**	RW	000b	Pin 4 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
15	**P3ST**	RW	0b	Pin 3 Schmitt 3rigger Enable: 0b: disabled

				1b: enabled
14:12	**P3DS**	RW	000b	Pin 3 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
11	**P2ST**	RW	0b	Pin 2 Schmitt Trigger Enable: 0b: disabled 1b: enabled
10:8	**P2DS**	RW	000b	Pin 2 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
7	**P1ST**	RW	0b	Pin 1 Schmitt Trigger Enable: 0b: disabled 1b: enabled
6:4	**P1DS**	RW	000b	Pin 1 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
3	**P0ST**	RW	0b	Pin 0 Schmitt Trigger Enable: 0b: disabled 1b: enabled
2:0	**P0DS**	RW	000b	Pin 0 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA

7.6.23 PBDS

Register 7-23 PBDS (PB Drive Strength/Schmitt Trigger, 400D 0464h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**P7ST**	RW	0b	Pin 7 Schmitt Trigger Enable: 0b: disabled 1b: enabled
30:28	**P7DS**	RW	000b	Pin 7 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
37	**P6ST**	RW	0b	Pin 6 Schmitt Trigger Enable: 0b: disabled 1b: enabled
36:24	**P6DS**	RW	000b	Pin 6 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
23	**P5ST**	RW	0b	Pin 5 Schmitt Trigger Enable: 0b: disabled 1b: enabled
22:20	**P5DS**	RW	000b	Pin 5 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
19	**P4ST**	RW	0b	Pin 4 Schmitt Trigger Enable: 0b: disabled 1b: enabled
18:16	**P4DS**	RW	000b	Pin 4 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
15	**P3ST**	RW	0b	Pin 3 Schmitt 3rigger Enable: 0b: disabled

				1b: enabled
14:12	**P3DS**	RW	000b	Pin 3 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
11	**P2ST**	RW	0b	Pin 2 Schmitt Trigger Enable: 0b: disabled 1b: enabled
10:8	**P2DS**	RW	000b	Pin 2 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
7	**P1ST**	RW	0b	Pin 1 Schmitt Trigger Enable: 0b: disabled 1b: enabled
6:4	**P1DS**	RW	000b	Pin 1 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
3	**P0ST**	RW	0b	Pin 0 Schmitt Trigger Enable: 0b: disabled 1b: enabled
2:0	**P0DS**	RW	000b	Pin 0 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA

7.6.24 PCDS

Register 7-24 PCDS (PC Drive Strength/Schmitt Trigger, 400D 0468h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**P7ST**	RW	0b	Pin 7 Schmitt Trigger Enable: 0b: disabled 1b: enabled
30:28	**P7DS**	RW	000b	Pin 7 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
37	**P6ST**	RW	0b	Pin 6 Schmitt Trigger Enable: 0b: disabled 1b: enabled
36:24	**P6DS**	RW	000b	Pin 6 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
23	**P5ST**	RW	0b	Pin 5 Schmitt Trigger Enable: 0b: disabled 1b: enabled
22:20	**P5DS**	RW	000b	Pin 5 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
19	**P4ST**	RW	0b	Pin 4 Schmitt Trigger Enable: 0b: disabled 1b: enabled
18:16	**P4DS**	RW	000b	Pin 4 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
15	**P3ST**	RW	0b	Pin 3 Schmitt 3rigger Enable: 0b: disabled

				1b: enabled
14:12	**P3DS**	RW	000b	Pin 3 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
11	**P2ST**	RW	0b	Pin 2 Schmitt Trigger Enable: 0b: disabled 1b: enabled
10:8	**P2DS**	RW	000b	Pin 2 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
7	**P1ST**	RW	0b	Pin 1 Schmitt Trigger Enable: 0b: disabled 1b: enabled
6:4	**P1DS**	RW	000b	Pin 1 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
3	**P0ST**	RW	0b	Pin 0 Schmitt Trigger Enable: 0b: disabled 1b: enabled
2:0	**P0DS**	RW	000b	Pin 0 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA

7.6.25 PDDS

Register 7-25 PDDS (PD Drive Strength/Schmitt Trigger, 400D 046Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**P7ST**	RW	0b	Pin 7 Schmitt Trigger Enable: 0b: disabled 1b: enabled
30:28	**P7DS**	RW	000b	Pin 7 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
37	**P6ST**	RW	0b	Pin 6 Schmitt Trigger Enable: 0b: disabled 1b: enabled
36:24	**P6DS**	RW	000b	Pin 6 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
23	**P5ST**	RW	0b	Pin 5 Schmitt Trigger Enable: 0b: disabled 1b: enabled
22:20	**P5DS**	RW	000b	Pin 5 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
19	**P4ST**	RW	0b	Pin 4 Schmitt Trigger Enable: 0b: disabled 1b: enabled
18:16	**P4DS**	RW	000b	Pin 4 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
15	**P3ST**	RW	0b	Pin 3 Schmitt 3rigger Enable: 0b: disabled

				1b: enabled
14:12	**P3DS**	RW	000b	Pin 3 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
11	**P2ST**	RW	0b	Pin 2 Schmitt Trigger Enable: 0b: disabled 1b: enabled
10:8	**P2DS**	RW	000b	Pin 2 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
7	**P1ST**	RW	0b	Pin 1 Schmitt Trigger Enable: 0b: disabled 1b: enabled
6:4	**P1DS**	RW	000b	Pin 1 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
3	**P0ST**	RW	0b	Pin 0 Schmitt Trigger Enable: 0b: disabled 1b: enabled
2:0	**P0DS**	RW	000b	Pin 0 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA

7.6.26 PEDS

Register 7-26 PEDS (PE Drive Strength/Schmitt Trigger, 400D 0470h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**P7ST**	RW	0b	Pin 7 Schmitt Trigger Enable: 0b: disabled 1b: enabled

30:28	**P7DS**	RW	000b	Pin 7 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
37	**P6ST**	RW	0b	Pin 6 Schmitt Trigger Enable: 0b: disabled 1b: enabled
36:24	**P6DS**	RW	000b	Pin 6 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
23	**P5ST**	RW	0b	Pin 5 Schmitt Trigger Enable: 0b: disabled 1b: enabled
22:20	**P5DS**	RW	000b	Pin 5 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
19	**P4ST**	RW	0b	Pin 4 Schmitt Trigger Enable: 0b: disabled 1b: enabled
18:16	**P4DS**	RW	000b	Pin 4 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
15	**P3ST**	RW	0b	Pin 3 Schmitt 3rigger Enable: 0b: disabled 1b: enabled
14:12	**P3DS**	RW	000b	Pin 3 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA

				111b: 25mA
11	**P2ST**	RW	0b	Pin 2 Schmitt Trigger Enable: 0b: disabled 1b: enabled
10:8	**P2DS**	RW	000b	Pin 2 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
7	**P1ST**	RW	0b	Pin 1 Schmitt Trigger Enable: 0b: disabled 1b: enabled
6:4	**P1DS**	RW	000b	Pin 1 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
3	**P0ST**	RW	0b	Pin 0 Schmitt Trigger Enable: 0b: disabled 1b: enabled
2:0	**P0DS**	RW	000b	Pin 0 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA

7.6.27 PFDS

Register 7-27 PFDS (PF Drive Strength/Schmitt Trigger, 400D 0474h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**P7ST**	RW	0b	Pin 7 Schmitt Trigger Enable: 0b: disabled 1b: enabled
30:28	**P7DS**	RW	000b	Pin 7 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA

37	**P6ST**	RW	0b	Pin 6 Schmitt Trigger Enable: 0b: disabled 1b: enabled
36:24	**P6DS**	RW	000b	Pin 6 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
23	**P5ST**	RW	0b	Pin 5 Schmitt Trigger Enable: 0b: disabled 1b: enabled
22:20	**P5DS**	RW	000b	Pin 5 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
19	**P4ST**	RW	0b	Pin 4 Schmitt Trigger Enable: 0b: disabled 1b: enabled
18:16	**P4DS**	RW	000b	Pin 4 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
15	**P3ST**	RW	0b	Pin 3 Schmitt 3rigger Enable: 0b: disabled 1b: enabled
14:12	**P3DS**	RW	000b	Pin 3 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
11	**P2ST**	RW	0b	Pin 2 Schmitt Trigger Enable: 0b: disabled 1b: enabled
10:8	**P2DS**	RW	000b	Pin 2 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA

				011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
7	**P1ST**	RW	0b	Pin 1 Schmitt Trigger Enable: 0b: disabled 1b: enabled
6:4	**P1DS**	RW	000b	Pin 1 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
3	**P0ST**	RW	0b	Pin 0 Schmitt Trigger Enable: 0b: disabled 1b: enabled
2:0	**P0DS**	RW	000b	Pin 0 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA

7.6.28 PGDS

Register 7-28 PGDS (PG Drive Strength/Schmitt Trigger, 400D 0478h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**P7ST**	RW	0b	Pin 7 Schmitt Trigger Enable: 0b: disabled 1b: enabled
30:28	**P7DS**	RW	000b	Pin 7 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
37	**P6ST**	RW	0b	Pin 6 Schmitt Trigger Enable: 0b: disabled 1b: enabled
36:24	**P6DS**	RW	000b	Pin 6 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA

				100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
23	**P5ST**	RW	0b	Pin 5 Schmitt Trigger Enable: 0b: disabled 1b: enabled
22:20	**P5DS**	RW	000b	Pin 5 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
19	**P4ST**	RW	0b	Pin 4 Schmitt Trigger Enable: 0b: disabled 1b: enabled
18:16	**P4DS**	RW	000b	Pin 4 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
15	**P3ST**	RW	0b	Pin 3 Schmitt 3rigger Enable: 0b: disabled 1b: enabled
14:12	**P3DS**	RW	000b	Pin 3 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
11	**P2ST**	RW	0b	Pin 2 Schmitt Trigger Enable: 0b: disabled 1b: enabled
10:8	**P2DS**	RW	000b	Pin 2 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
7	**P1ST**	RW	0b	Pin 1 Schmitt Trigger Enable: 0b: disabled 1b: enabled

6:4	**P1DS**	RW	000b	Pin 1 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA
3	**P0ST**	RW	0b	Pin 0 Schmitt Trigger Enable: 0b: disabled 1b: enabled
2:0	**P0DS**	RW	000b	Pin 0 Drive Strength: 000b: 6mA 001b: 8mA 010b: 11mA 011b: 14mA 100b: 17mA 101b: 20mA 110b: 22mA 111b: 25mA

8 WWDT

8.1 Overview

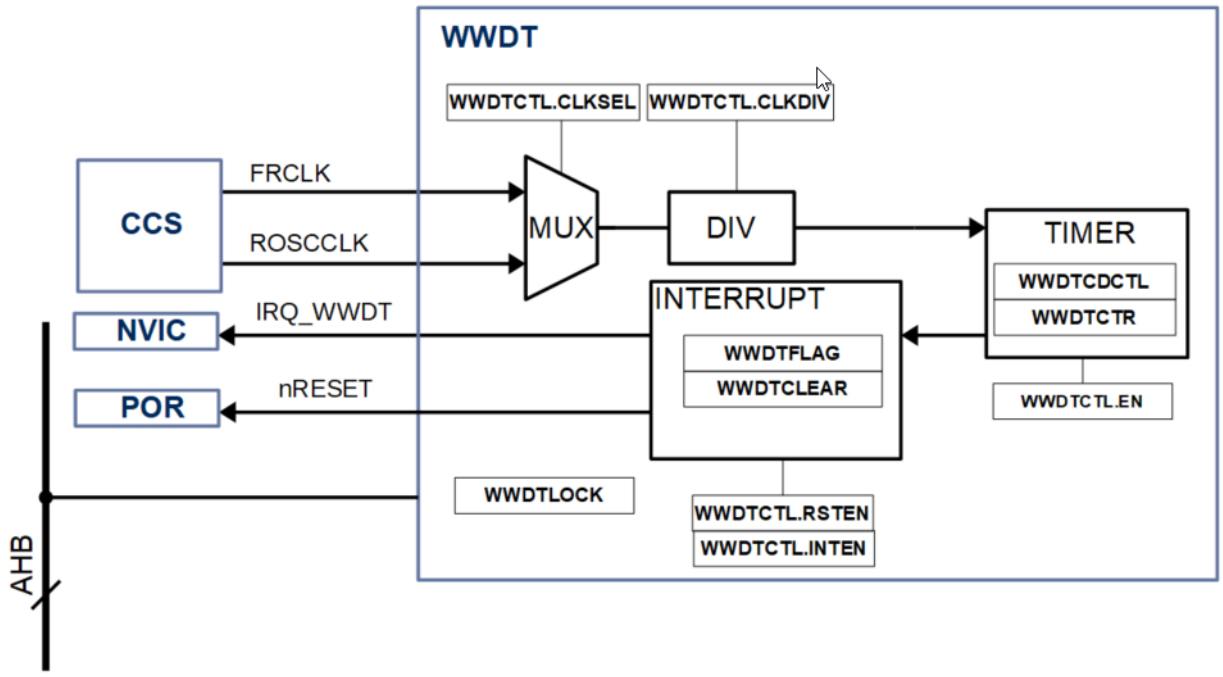
The PAC55XX contains a Windowed Watchdog Timer (WWDT) peripheral. The WWDT is an AHB client that allows the user to specify safety timeout by which the timer must be cleared, to avoid a reset.

8.2 Features

- 16-bit Count-down timer
- Input clock selection
- Input clock divider
- Watchdog mode
- Interval timer mode
- Register write protection

8.3 System Block Diagram

Figure 8-1 WWDT System Block Diagram



8.4 Functional Description

The WWDT is a 16-bit count-down timer. The WWDT is an AHB client and has a clock input of FRCLK or ROSCCLK. The input clock may be selected by the **WWDTCTL.CLKSEL** field.

The WWDT has an input clock divider that may be selected as shown below.

Table 8-1 WWDT Input Clock Divider

WWDTCTL.CLKDIV	WWDT Clock
0000b	WWDTCTL.CLKSEL /1
0001b	WWDTCTL.CLKSEL /2
0010b	WWDTCTL.CLKSEL /4
0011b	WWDTCTL.CLKSEL /8
…	…
1110b	WWDTCTL.CLKSEL /16384
1111b	WWDTCTL.CLKSEL /32768

If the user wants to write any register in the WWDT, **WWDTLOCK** must be set to 55AA 6699h.

If the user wants to read any register in the WWDT, **WWDTLOCK** must be set to 55AA 6698h.

8.4.1 Watchdog Mode

If **WWDTCTL.RSTEN** bit is set to 1b and **WWDTCTL.INTEN** bit is set to 0b, then the WWDT operates in Watchdog mode.

When the timer is enabled by setting **WWDTCTL.EN** to 1b, the timer loads the value of **WWDTCDCTL.CDV** into **WWDTCTR**. The timer will count down **WWDTCTR.VALUE**.

The user may clear or reset the timer at any time by writing **WWDTCLEAR** to any value. This reloads the **WWDTCDCTL.CDV** value into **WWDTCTR.VALUE**. If **WWDTCTR.VALUE** counts down to 0 the **WWDTFLAG.RSTF** is set to a 1b, the nRESET signal is asserted, and the PAC55XX will perform a warm reset. The **WWDTFLAG.RSTF** flag is only set to 1b in the event of a WWDT reset. Other types of resets will not set this bit. The **WWDTFLAG.RSTF** flag bit is W1C (write 1 to clear).

The windowing function of the WWDT makes sure the user only resets the timer by writing the **WWDTCLEAR** register when the **WWDTCTR.VALUE** is < **WWDTCDCTL.WINDOW**. If this is not true, then the **WWDTFLAG.RSTF** is set to a 1b, the nRESET signal is asserted, and the PAC55XX will perform a warm reset.

8.4.2 Interval Timer Mode

If **WWDTCTL.INTEN** bit is set to 1b, and **WWDTCTL.RSTEN** bit is set to 0b, then the WWDT interrupt will be enabled and operates as an interval timer. In this mode, **WWDTCDCTL.WINDOW** should be set to 0.

When the timer is enabled by setting **WWDTCTL.EN** to 1b, the timer loads the value of **WWDTCDCTL.CDV** into **WWDTCTR**. The timer will count down **WWDTCTR.VALUE**. When the **WWDTCTR.VALUE** reaches 0, the **WWDTCDCTL.CDV** is reloaded into **WWDTCTR**, and the **WWDTFLAG.IF** is set to a 1b. Since **WWDTCTL.INTEN** = 1, an interrupt signal is sent to the NVIC. The **WWDTFLAG.IF** flag bit is W1C (write 1 to clear).

8.5 Register Summary

Table 8-2 WWDT Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
WWDTCTL	400D 0800h	WWDT Control	RW	0000 0000h
WWDTCDCTL	400D 0804h	WWDT Count-down Control	RW	0010 FFFFh
WWDTCTR	400D 0808h	WWDT Counter	RO	0000 FFFFh
WWDTFLAG	400D 080Ch	WWDT Interrupt Flag	RO	0000 0000h
WWDTCLEAR	400D 0810h	WWDT Clear	WO	--
WWDTLOCK	400D 0814h	WWDT Lock	RW	0000 0000h

8.6 Register Detail

8.6.1 WWDTCTL

Register 8-1 WWDTCTL (WWDT Control, 400D 0800h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:4	**CLKDIV**	RW	0	WWDT Input Clock Divider: 0000b: **WWDTCTL.CLKSEL** /1 0001b: **WWDTCTL.CLKSEL** /2 0010b: **WWDTCTL.CLKSEL** /4 0011b: **WWDTCTL.CLKSEL** /8 0100b: **WWDTCTL.CLKSEL** /16 0101b: **WWDTCTL.CLKSEL** /32 0110b: **WWDTCTL.CLKSEL** /64 0110b: **WWDTCTL.CLKSEL** /128 0111b: **WWDTCTL.CLKSEL** /256 1000b: **WWDTCTL.CLKSEL** /512 1001b: **WWDTCTL.CLKSEL** /1024 1010b: **WWDTCTL.CLKSEL** /2048 1011b: **WWDTCTL.CLKSEL** /4096 1100b: **WWDTCTL.CLKSEL** /8192 1101b: **WWDTCTL.CLKSEL** /16384 1111b: **WWDTCTL.CLKSEL** /32768
3	**CLKSEL**	RW	0	WWDT Input Clock Selection: 0b: FRCLK 1b: ROSCCLK
2	**RSTEN**	RW	0	WWDT POR Enable: 0b: Do not assert nRESET 1b: Assert nRESET on **WWDTCTR** = 0
1	**INTEN**	RW	0	WWDT Interrupt Enable: 0b: no interrupt 1b: Assert IRQ_WWDT when **WWDTFLAG.IF** = 1b
0	**EN**	RW	0	WWDT Enable: 0b: WWDT Disabled 1b: WWDT Enabled

8.6.2 WWDTCDCTL

Register 8-2 WWDTCDCTL (WWDT Count-down Control, 400D 0804h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**WINDOW**	RW	0010h	Can clear WWDT when **WWDTCTR** is below this value.
15:0	**CDV**	RW	FFFFh	WWDT Count-down Value. This value is loaded into **WWDTCTR** when the counter reaches a value of 0. When this register is written, the count is immediately restarted from the new value. The minimum valid value is 1.

8.6.3 WWDTCTR

Register 8-3 WWDTCTR (WWDT Counter Value, 400D 0808h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**VALUE**	RO	FFFFh	WWDT Counter Value. This is the current WWDT counter value.

8.6.4 WWDTFLAG

Register 8-4 WWDTFLAG (WWDT Flag, 400D 080Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:2	**Reserved**	RO	0	Reserved
1	**RSTF**	RW	0	WWDT Reset Flag: 0b: no flag 1b: flag (write 1 to clear)
0	**IF**	RW	0	WWDT Interrupt Flag: 0b: no flag 1b: flag (write 1 to clear)

8.6.5 WWDTCLEAR

Register 8-5 WWDTCLEAR (WWDT Clear, 400D 0810h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**VALUE**	WO	-	WWDT Interrupt Clear. The user may clear or reset the timer at any time by writing **WWDTCLEAR** with any value. This reloads the value from **WWDTCDCTL.CDV** into **WWDTCTR.VALUE**.

8.6.6 WWDTLOCK

Register 8-6 WWDTLOCK (WWDT Lock, 400D 0814h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**VALUE**	RW	0	WWDT Lock. This register will allow all of the WWDT registers to be locked for writing. The possible values for this register are (all 32-bits): 55AA 6699h: All registers in the WWDT available for writing 55AA 6698h: All registers in the WWDT are read-only Any other values for this register will be ignored.

9 RTC

9.1 Overview

The PAC55XX contains a Real-Time Clock (RTC) with Calendar and Alarm functions.

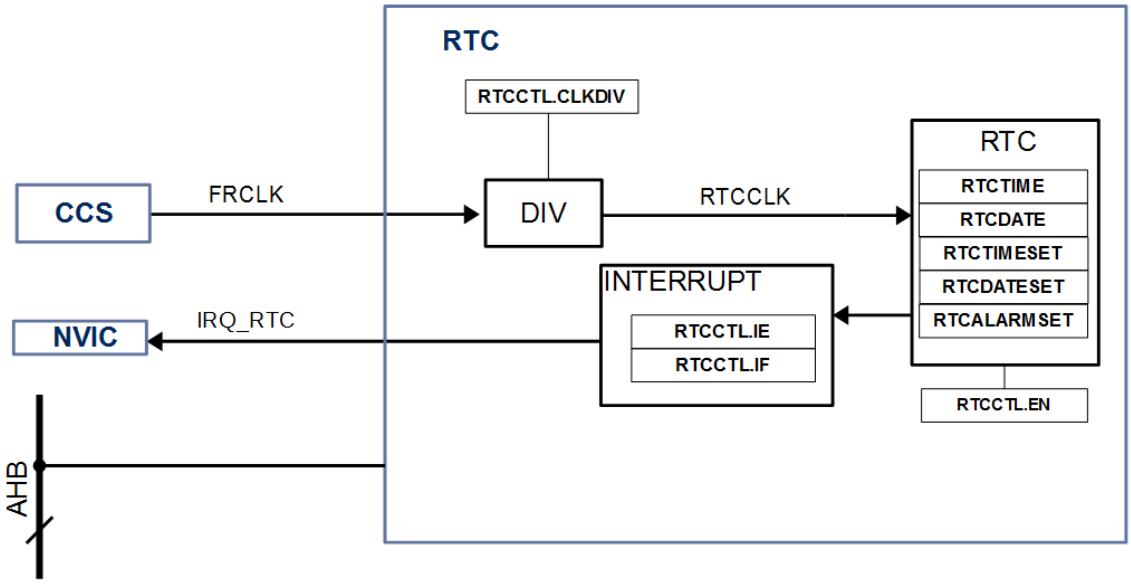
The RTC is an AHB client and is clocked using the FRCLK clock.

9.2 Features

- RTC with Calendar and Alarm Functions
- Interrupt on matching years, months, weeks, days, hours, minutes, seconds
- Enable/Disable to save power
- Input clock divider

9.3 System Block Diagram

Figure 9-1 RTC System Block Diagram



9.4 Functional Description

9.4.1 Overview

The Real-Time Clock (RTC) is a timer peripheral that allows the user to program an alarm time in the future using years, months, weeks, days, hours, minutes and seconds. The user can program the RTC to interrupt the MCU when an alarm time is reached.

The RTC is clocked by the FRCLK system clock. The RTC is intended to work with a 1MHz clock source, so the user must set the **RTCCTL.CLKDIV** field with the proper divider to generate a 1MHz RTCCLK from FRCLK. This will insure that this will count the real-time properly.

The RTC can be enabled writing setting **RTCCTL.EN** to 1b. It can be disabled by setting this bit to 0b.

The user may read the current time and date from the **RTCTIME** and **RTCDATE** read-only registers at any time.

To set the current time and date, the user may write the **RTCTIMESET** and **RTCDATESET** registers with the current date and time. To copy the time and date to the internal registers used for keeping the time, set the **RTCCTL.SETCAL** to 1b. In order for this to work, the **RTCCTL.EN** bit must already be set to 1b. The **RTCCTL.SETCAL** is always read as a 0b.

The user may set an alarm based on days, week day, hours or minutes by using the **RTCALARMSET** register.

To match the day alarm setting, set the **RTCALARMSET.DAYALARM** to the day number and **RTCALARMSET.DAYALARMEN** to 1b.

To match the week day alarm setting, set the **RTCALARMSET.DAYOFWEEKALARM** to the week day number and **RTCALARMSET. DAYOFWEEKALARMEN** to 1b.

To match the hour alarm setting, set the **RTCALARMSET.HOURALARM** to the hour number and **RTCALARMSET.HOURALARMEN** to 1b.

To match the minute setting, set the **RTCALARMSET.MINUTEALARM** to the minute number and **RTCALARMSET.MINUTEALARMEN** to 1b.

If the alarm matches the time, **RTCCTL.IF** will be set to a 1b. If the **RTCCTL.IE** is set to 1b, then the IRQ_RTC signal will be asserted to the NVIC.

9.5 Register Summary

Table 9-1 RTC Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
RTCCTL	400D 0C00h	RTC Control	RW	0000 0010h
RTCTIME	400D 0C04h	RTC Time	RO	0000 0000h
RTCDATE	400D 0C08h	RTC Date	RO	0001 0100h
RTCTIMESET	400D 0C0Ch	RTC Time Setting	RW	0000 0000h
RTCDATESET	400D 0C10h	RTC Date Setting	RW	0001 0100h
RTCALARMSET	400D 0C14h	RTC Alarm Setting	RW	0000 0000h

9.6 Register Detail

9.6.1 RTCCTL

Register 9-1 RTCCTL (RTC Control, 400D 0C00h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**SETCAL**	RW	0	When **RTCCTL.EN** is a 1b, writing 1b to this bit sets the internal TIME/DATA registers to the value of the **TIMESET** and **RTCDATESET** registers. **RTCCTL.EN** must be a 1b to write this bit. When read, this bit will always return 0b.
30:9	**Reserved**	RO	0	Reserved
8:4	**CLKDIV**	RW	0 0001b	Clock divider for the RTC input clock (FRCLK). The user should set this divider so that the output of the divider generates a 1MHz clock, in order for the RTC to correctly count the real-time. 0 0000b: FRCLK /1 0 0001b: FRCLK /2 0 0010b: FRCLK /3 … 1 1110b: FRCLK /31 1 1111b: FRCLK /32
3	**Reserved**	RO	0	Reserved
2	**IF**	W1C	0b	Alarm match interrupt flag: 0b: no alarm 1b: alarm
1	**IE**	RW	0b	Alarm match interrupt enable: 0b: not enabled 1b: enabled
0	**EN**	RW	0b	RTC Enabled: 0b: not enabled 1b: enabled

9.6.2 RTCTIME

Register 9-2 RTCTIME (RTC Time, 400D 0C04h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:22	**Reserved**	RO	0	Reserved
21:16	**HOURS**	RO	0	RTC Hours
15	**Reserved**	RO	0	Reserved
14:8	**MINUTES**	RO	0	RTC Minutes
7	**Reserved**	RO	0	Reserved
6:0	**SECONDS**	RO	0	RTC Seconds

9.6.3 RTCDATE

Register 9-3 RTCDATE (RTC Date, 400D 0C08h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:24	**YEAR**	RO	0	RTC Year
23:21	**Reserved**	RO	0	Reserved
20:16	**MONTH**	RO	1	RTC Month
15:14	**Reserved**	RO	0	Reserved
13:8	**DAY**	RO	1	RTC Day
7:3	**Reserved**	RO	0	Reserved
2:0	**DAYOFWEEK**	RO	0	RTC Day of Week

9.6.4 RTCTIMESET

Register 9-4 RTCTIMESET (RTC Time Setting, 400D 0C0Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:22	**Reserved**	RO	0	Reserved
21:16	**HOUR**	RW	0	RTC Hour Setting
15	**Reserved**	RO	0	Reserved
14:8	**MINUTE**	RW	0	RTC Minute Setting
7	**Reserved**	RO	0	Reserved
6:0	**SECOND**	RW	0	RTC Second Setting

9.6.5 RTCDATESET

Register 9-5 RTCDATESET (RTC Date Setting, 400D 0C10h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:24	**YEAR**	RW	0	RTC Year Setting
23:21	**Reserved**	RO	0	Reserved
20:16	**MONTH**	RW	1	RTC Month Setting
15:14	**Reserved**	RO	0	Reserved
13:8	**DAY**	RW	1	RTC Day Setting
7:3	**Reserved**	RO	0	Reserved
2:0	**DAYOFWEEK**	RW	0	RTC Day of Week Setting

9.6.6 RTCALARMSET

Register 9-6 RTCALARMSET (RTC Alarm Setting, 400D 0C14h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**Reserved**	RO	0	Reserved
30	**DAYALARMEN**	RW	0	Day Alarm Enabled: 0b: disabled 1b: enabled
29:24	**DAYALARM**	RW	0	Day Alarm Setting
23:20	**Reserved**	RO	0	Reserved
19	**DAYOFWEEKALARMEN**	RW	0	Day of Week Alarm Enabled: 0b: disabled 1b: enabled
18:16	**DAYOFWEEKALARM**	RW	0	Day of Week Alarm Setting
15	**Reserved**	RO	0	Reserved
14	**HOURALARMEN**	RW	0	Hour Alarm Enabled: 0b: disabled 1b: enabled
13:8	**HOURALARM**	RW	0	Hour Alarm Setting
7	**MINUTEALARMEN**	RW	0	Minute Alarm Enabled: 0b: disabled 1b: enabled
6:0	**MINUTEALARM**	RW	0	Minute Alarm Setting

10 GPIOA

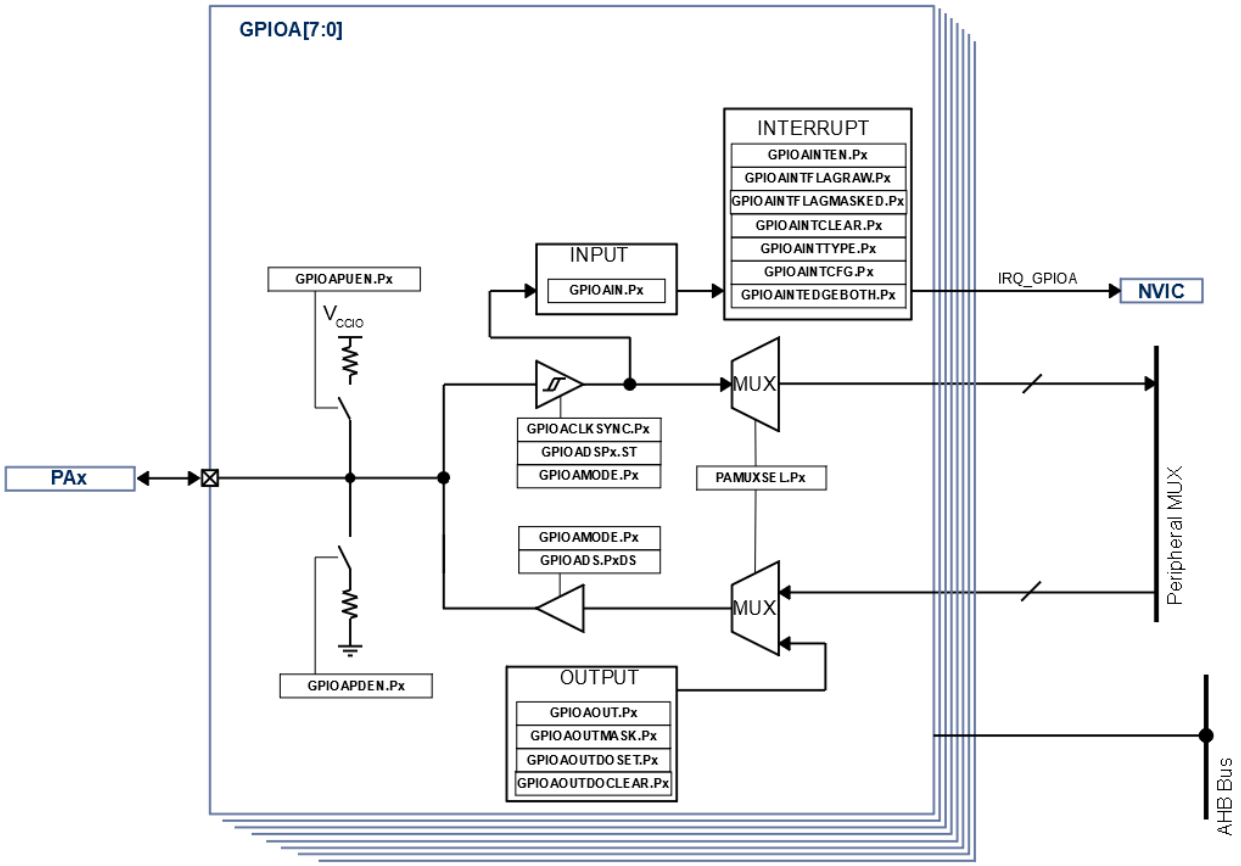
10.1 Overview

The GPIOA port contains 8 GPIO pin, GPIOA0 to GPIOA7.

Each GPIO pin may be used for Digital Input or Digital Output. Each GPIO is capable of generating 3.3V output and is 3.3V input tolerant and has a set of features described below.

10.2 GPIOA Block Diagram

Figure 10-1 GPIOA Block Diagram



10.3 Features

- 3.3V output, 3.3V input tolerant
- Configurable Pin Modes:
  - High-Impedance Digital Input
  - Push-Pull Digital Output
  - Open-Drain Digital Output
- Configurable Drive Strength up to 25mA
- Configurable Input Schmitt Trigger
- Configurable Weak Pull-up or Pull-down
- Edge or Level Sensitive Interrupts
  - Rising, Falling or both edges
- Clock Synchronization

10.4 Functional Description

10.4.1 IO Mode

Each GPIO pin supports up to 4 different modes and can selected from the **GPIOAMODE** register as shown below.

Table 10-1 GPIOA Mode Configuration

GPIOAMODE VALUE	DESCRIPTION
00b	Reserved
01b	Push-Pull Digital Output
10b	Open-Drain Digital Output
11b	High-Impedance Digital Input

The reset value for the GPIO pin mode is High-Impedance Digital Input.

10.4.2 Digital Output

Each GPIO pin may be configured for digital output by setting the **GPIOAMODE.Px** to either 01b (Push-pull output) or to 10b (Open-drain output).

When configured for Push-pull output, the GPIO is driven to the high state using the VCCIO 3.3V power supply. When configured for Open-drain output, the GPIO is pulled up externally.

When **GPIOAOUTMASK.Px** is set to 0b, then the state of the GPIO pin will be set from **GPIOAOUT.Px** (0b: logic low, 1b: logic high). If the **GPIOAOUTMASK.Px** is set to 1b, then any changes to **GPIOAOUT.Px** have no effect.

In either of these two output modes, the user may read the digital state of the input by reading the **GPIOAIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports both digital output modes.

10.4.3 Digital Input

If **GPIOAMODE.Px** is set to 11b, then the GPIO pin is configured for high-impedance input. For application safety, this is the reset value for the GPIO pin mode.

In this mode, the user may read the state of the digital input by reading the **GPIOAIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports this mode.

10.4.4 GPIOA Interrupts

Each GPIO pin supports configurable interrupts. In order for interrupts to work, the GPIO must be set for high-impedance input mode (**GPIOAMODE.Px** = 11b).

The user may configure either edge or level sensitive interrupts for each GPIO pin. If the user configures edge sensitive interrupts, they may select rising, falling or both edges for edge interrupts.

See the table below for how to configure the GPIO for the various types of interrupts.

Table 10-2 GPIOA Interrupt Configuration

GPIOAINTTYPE	GPIOAINTCFG	GPIOAEDGEBOTH	INTERRUPT MODE
0b	0b	0b	Falling Edge
	1b	0b	Rising Edge
	X	1b	Both Rising and Falling Edge
1b	0b	X	Level Logic Low
	1b	X	Level Logic High

Interrupts may be enabled or disabled using the **GPIOAINTEN** register. To enable interrupts on a pin, set the **GPIOAINTEN.Px** to 1b. To disable interrupts on a pin, set the **GPIOAINTEN.Px** to 0b.

If an edge or level interrupt is detected, the **GPIOAINTFLAG.Px** will be set to 1b for that GPIO pin. If **GPIOAINTFLAG.Px** is set to 1b and **GPIOAINTEN.Px** is set to 1b, then the IRQ_GPIOA signal to the NVIC is asserted.

To clear the **GPIOAINTFLAG.Px** interrupt flag, write **GPIOAINTCLEAR.Px** to a 1b.

10.4.5 GPIOA Clock Synchronization

Each GPIOA pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOAMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOA pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOACLKSYNC.Px** = 0b). To enable GPIOA Clock Synchronization set **GPIOACLKSYNC.Px** = 1b.

10.4.6 GPIOA Drive Strength

Each GPIO pin has configurable drive strength when the ports are configured as a digital output.

See the table below for the settings for GPIOA Drive strength.

Table 10-3 GPIOA Drive Strength

PADS.PxDS	DRIVE STRENGTH
000b	6mA
001b	8mA
010b	11mA
011b	14mA
100b	17mA
101b	20mA
110b	22mA
111b	25mA

The drive strength shown in the table above represents the minimum drive strength for the GPIO pin. For the details on the electrical characteristics of the drive strength, set the device data sheet.

Note that the register configuration for the GPIOA Drive Strength is located in the Clock Control System (SCC).

10.4.7 GPIOA Schmitt Trigger

Each GPIO has a Schmitt Trigger that can be enabled when the GPIO is configured as a high-impedance digital input (**GPIOAMODE.Px** = 11b).

The Schmitt Trigger is disabled by default (**PADS.PxST** = 0b). To enable the input Schmitt Triger, set **PADS.PxST** = 1b.

Note that the register configuration for the GPIO Schmitt Trigger is located in the GPIO and Digital Peripheral MUX (DPM).

10.4.8 GPIOA Weak Pull-up and Pull-down

Each GPIO pin has a configurable weak 60k pull-up to VCCIO or pull-down to ground.

To enable the weak pull-up to VCCIO, set **GPIOAPU.Px** to 1b. To disable the pull-up, set **GPIOAPU.Px** to 0b.

To enable the weak pull-down to ground, set the **GPIOAPD.Px** to 1b. To disable the pull-down, set **GPIOAPD.Px** to 0b.

WARNING: Do not configure the weak pull-up and pull-down to be active at the same time. Doing so may cause device damage.

10.5 Peripheral IO Mapping

The GPIOA peripheral is connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 10-4 GPIOA Peripheral IO Mapping

GPIOA SIGNAL	IO PIN
GPIOA0	PA0
GPIOA1	PA1
GPIOA2	PA2
GPIOA3	PA3
GPIOA4	PA4
GPIOA5	PA5
GPIOA6	PA6
GPIOA7	PA7

10.6 Register Summary

Table 10-5 GPIOA Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOAMODE	400D 1400h	GPIOA Pin Mode Select	RW	0000 FFFFh
GPIOAOUTMASK	400D 1404h	GPIOA Data Output Write Mask	RW	0000 0000h
GPIOAOUT	400D 1408h	GPIOA Data Output Value	RW	0000 00FFh
GPIOAIN	400D 140Ch	GPIOA Data Input Value	RO	--
GPIOAINTEN	400D 1410h	GPIOA Interrupt Enable	RW	0000 0000h
GPIOAINTFLAG	400D 1414h	GPIOA Interrupt Flag	RO	0000 0000h
GPIOAINTCLEAR	400D 141Ch	GPIOA Interrupt Clear	WO	--
GPIOAINTTYPE	400D 1420h	GPIOA Interrupt Type	RW	0000 0000h
GPIOAINTCFG	400D 1424h	GPIOA Interrupt Configuration	RW	0000 0000h
GPIOAINTEDGEBOTH	400D 1428h	GPIOA Interrupt Edge Both	RW	0000 0000h
GPIOACLKSYNC	400D 142Ch	GPIOA Clock Synchronization	RW	0000 0000h
GPIOADOSET	400D 1430h	GPIOA Data Output Set	WO	--
GPIOADOCLEAR	400D 1434h	GPIOA Data Output Clear	WO	--

10.7 Register Detail

10.7.1 GPIOAMODE

Register 10-1 GPIOAMODE (GPIOA Mode Configuration, 400D 1400h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:14	**P7**	RW	11b	Pin 7 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
13:12	**P6**	RW	11b	Pin 6 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
11:10	**P5**	RW	11b	Pin 5 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
9:8	**P4**	RW	11b	Pin 4 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
7:6	**P3**	RW	11b	Pin 3 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
5:4	**P2**	RW	11b	Pin 2 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
3:2	**P1**	RW	11b	Pin 1 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
1:0	**P0**	RW	11b	Pin 0 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input

10.7.2 GPIOAOUTMASK

Register 10-2 GPIOAOUTMASK (GPIOA Output Mask, 400D 1404h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Output Data Mask: 0b: Not masked 1b: Masked
6	**P6**	RW	0	Pin 6 Output Data Mask: 0b: Not masked 1b: Masked
5	**P5**	RW	0	Pin 5 Output Data Mask: 0b: Not masked 1b: Masked
4	**P4**	RW	0	Pin 4 Output Data Mask: 0b: Not masked 1b: Masked
3	**P3**	RW	0	Pin 3 Output Data Mask: 0b: Not masked 1b: Masked
2	**P2**	RW	0	Pin 2 Output Data Mask: 0b: Not masked 1b: Masked
1	**P1**	RW	0	Pin 1 Output Data Mask: 0b: Not masked 1b: Masked
0	**P0**	RW	0	Pin 0 Output Data Mask: 0b: Not masked 1b: Masked

10.7.3 GPIOAOUT

Register 10-3 GPIOAOUT (GPIOA Output Data Value, 400D 1408h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Output Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RW	1	Pin 6 Output Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RW	1	Pin 5 Output Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RW	1	Pin 4 Output Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RW	1	Pin 3 Output Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RW	1	Pin 2 Output Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RW	1	Pin 1 Output Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RW	1	Pin 0 Output Data Value: 0b: Logic low 1b: Logic high

10.7.4 GPIOAIN

Register 10-4 GPIOAIN (GPIOA Input Data Value, 400D 140Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	-	Pin 7 Input Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RO	-	Pin 6 Input Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RO	-	Pin 5 Input Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RO	-	Pin 4 Input Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RO	-	Pin 3 Input Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RO	-	Pin 2 Input Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RO	-	Pin 1 Input Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RO	-	Pin 0 Input Data Value: 0b: Logic low 1b: Logic high

10.7.5 GPIOAINTEN

Register 10-5 GPIOAINTEN (GPIOA Interrupt Enable, 400D 1410h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Enable: 0b: Disabled 1b: Enabled
6	**P6**	RW	0	Pin 6 Interrupt Enable: 0b: Disabled 1b: Enabled
5	**P5**	RW	0	Pin 5 Interrupt Enable: 0b: Disabled 1b: Enabled
4	**P4**	RW	0	Pin 4 Interrupt Enable: 0b: Disabled 1b: Enabled
3	**P3**	RW	0	Pin 3 Interrupt Enable: 0b: Disabled 1b: Enabled
2	**P2**	RW	0	Pin 2 Interrupt Enable: 0b: Disabled 1b: Enabled
1	**P1**	RW	0	Pin 1 Interrupt Enable: 0b: Disabled 1b: Enabled
0	**P0**	RW	0	Pin 0 Interrupt Enable: 0b: Disabled 1b: Enabled

10.7.6 GPIOAINTFLAG

Register 10-6 GPIOAINTFLAG (GPIOA Interrupt Flag, 400D 1414h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	0	Pin 7 Interrupt Flag: 0b: No flag 1b: Flag
6	**P6**	RO	0	Pin 6 Interrupt Flag: 0b: No flag 1b: Flag
5	**P5**	RO	0	Pin 5 Interrupt Flag: 0b: No flag 1b: Flag
4	**P4**	RO	0	Pin 4 Interrupt Flag: 0b: No flag 1b: Flag
3	**P3**	RO	0	Pin 3 Interrupt Flag: 0b: No flag 1b: Flag
2	**P2**	RO	0	Pin 2 Interrupt Flag: 0b: No flag 1b: Flag
1	**P1**	RO	0	Pin 1 Interrupt Flag: 0b: No flag 1b: Flag
0	**P0**	RO	0	Pin 0 Interrupt Flag: 0b: No flag 1b: Flag

10.7.7 GPIOAINTCLEAR

Register 10-7 GPIOAINTCLEAR (GPIOA Interrupt Clear, 400D 141Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Interrupt Clear: 0b: No effect 1b: Clear **GPIOAINTFLAG.P7**
6	**P6**	WO	-	Pin 6 Interrupt Clear: 0b: No effect 1b: Clear **GPIOAINTFLAG.P6**
5	**P5**	WO	-	Pin 5 Interrupt Clear: 0b: No effect 1b: Clear **GPIOAINTFLAG.P5**
4	**P4**	WO	-	Pin 4 Interrupt Clear: 0b: No effect 1b: Clear **GPIOAINTFLAG.P4**
3	**P3**	WO	-	Pin 3 Interrupt Clear: 0b: No effect 1b: Clear **GPIOAINTFLAG.P3**
2	**P2**	WO	-	Pin 2 Interrupt Clear: 0b: No effect 1b: Clear **GPIOAINTFLAG.P2**
1	**P1**	WO	-	Pin 1 Interrupt Clear: 0b: No effect 1b: Clear **GPIOAINTFLAG.P1**
0	**P0**	WO	-	Pin 0 Interrupt Clear: 0b: No effect 1b: Clear **GPIOAINTFLAG.P0**

10.7.8 GPIOAINTTYPE

Register 10-8 GPIOAINTTYPE (GPIOA Interrupt Type Configuration, 400D 1420h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Type: 0b: Edge triggered 1b: Level triggered
6	**P6**	RW	0	Pin 6 Interrupt Type: 0b: Edge triggered 1b: Level triggered
5	**P5**	RW	0	Pin 5 Interrupt Type: 0b: Edge triggered 1b: Level triggered
4	**P4**	RW	0	Pin 4 Interrupt Type: 0b: Edge triggered 1b: Level triggered
3	**P3**	RW	0	Pin 3 Interrupt Type: 0b: Edge triggered 1b: Level triggered
2	**P2**	RW	0	Pin 2 Interrupt Type: 0b: Edge triggered 1b: Level triggered
1	**P1**	RW	0	Pin 1 Interrupt Type: 0b: Edge triggered 1b: Level triggered
0	**P0**	RW	0	Pin 0 Interrupt Type: 0b: Edge triggered 1b: Level triggered

10.7.9 GPIOAINTCFG

Register 10-9 GPIOAINTCFG (GPIOA Interrupt Value Configuration, 400D 1424h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	If **GPIOAINTTYPE.Px** = 0 (edge-triggered), Pin 7 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOAINTTYPE.Px** = 1 (level-triggered), Pin 7 Interrupt Level: 0b: Logic low 1b: Logic high
6	**P6**	RW	0	If **GPIOAINTTYPE.Px** = 0 (edge-triggered), Pin 6 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOAINTTYPE.Px** = 1 (level-triggered), Pin 6 Interrupt Level: 0b: Logic low 1b: Logic high
5	**P5**	RW	0	If **GPIOAINTTYPE.Px** = 0 (edge-triggered), Pin 5 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOAINTTYPE.Px** = 1 (level-triggered), Pin 5 Interrupt Level: 0b: Logic low 1b: Logic high
4	**P4**	RW	0	If **GPIOAINTTYPE.Px** = 0 (edge-triggered), Pin 4 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOAINTTYPE.Px** = 1 (level-triggered), Pin 4 Interrupt Level: 0b: Logic low 1b: Logic high
3	**P3**	RW	0	If **GPIOAINTTYPE.Px** = 0 (edge-triggered), Pin 3 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOAINTTYPE.Px** = 1 (level-triggered), Pin 3 Interrupt Level: 0b: Logic low 1b: Logic high
2	**P2**	RW	0	If **GPIOAINTTYPE.Px** = 0 (edge-triggered), Pin 2 Interrupt Edge:

				0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOAINTTYPE.Px** = 1 (level-triggered), Pin 2 Interrupt Level: 0b: Logic low 1b: Logic high
1	**P1**	RW	0	If **GPIOAINTTYPE.Px** = 0 (edge-triggered), Pin 1 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOAINTTYPE.Px** = 1 (level-triggered), Pin 1 Interrupt Level: 0b: Logic low 1b: Logic high
0	**P0**	RW	0	If **GPIOAINTTYPE.Px** = 0 (edge-triggered), Pin 0 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOAINTTYPE.Px** = 1 (level-triggered), Pin 0 Interrupt Level: 0b: Logic low 1b: Logic high

10.7.10 GPIOAINTEDGEBOTH

Register 10-10 GPIOAINTEDGEBOTH (GPIOA Interrupt Both Edge Configuration, 400D 1428h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOAINTCFG.P7** 1b: Trigger on both rising and falling edge
6	**P6**	RW	0	Pin 6 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOAINTCFG.P6** 1b: Trigger on both rising and falling edge
5	**P5**	RW	0	Pin 5 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOAINTCFG.P5** 1b: Trigger on both rising and falling edge
4	**P4**	RW	0	Pin 4 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOAINTCFG.P4** 1b: Trigger on both rising and falling edge
3	**P3**	RW	0	Pin 3 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOAINTCFG.P3** 1b: Trigger on both rising and falling edge
2	**P2**	RW	0	Pin 2 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOAINTCFG.P2** 1b: Trigger on both rising and falling edge
1	**P1**	RW	0	Pin 1 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOAINTCFG.P1** 1b: Trigger on both rising and falling edge
0	**P0**	RW	0	Pin 0 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOAINTCFG.P0** 1b: Trigger on both rising and falling edge

10.7.11 GPIOACLKSYNC[6]

Register 10-11 GPIOACLKSYNC (GPIOA Clock Synchronization Enable, 400D 142Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
6	**P6**	RW	1	Pin 6 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
5	**P5**	RW	1	Pin 5 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
4	**P4**	RW	1	Pin 4 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
3	**P3**	RW	1	Pin 3 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
2	**P2**	RW	1	Pin 2 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
1	**P1**	RW	1	Pin 1 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
0	**P0**	RW	1	Pin 0 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)

[6] In VER1 of the PAC55XX MCU, the default value for the clock synchronization is disabled (0b).

10.7.12 GPIOADOSET

Register 10-12 GPIOADOSET (GPIOA Data Output Set, 400D 1430h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P7** to 1b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P6** to 1b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P5** to 1b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P4** to 1b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P3** to 1b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P2** to 1b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P1** to 1b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P0** to 1b

10.7.13 GPIOADOCLEAR

Register 10-13 GPIOADOCLEAR (GPIOA Data Output Clear, 400D 1434h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Clear: 0b: No effect 1b: Set **GPIOAOUT.P7** to 0b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P6** to 0b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P5** to 0b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P4** to 0b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P3** to 0b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P2** to 0b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P1** to 0b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOAOUT.P0** to 0b

11 GPIOB

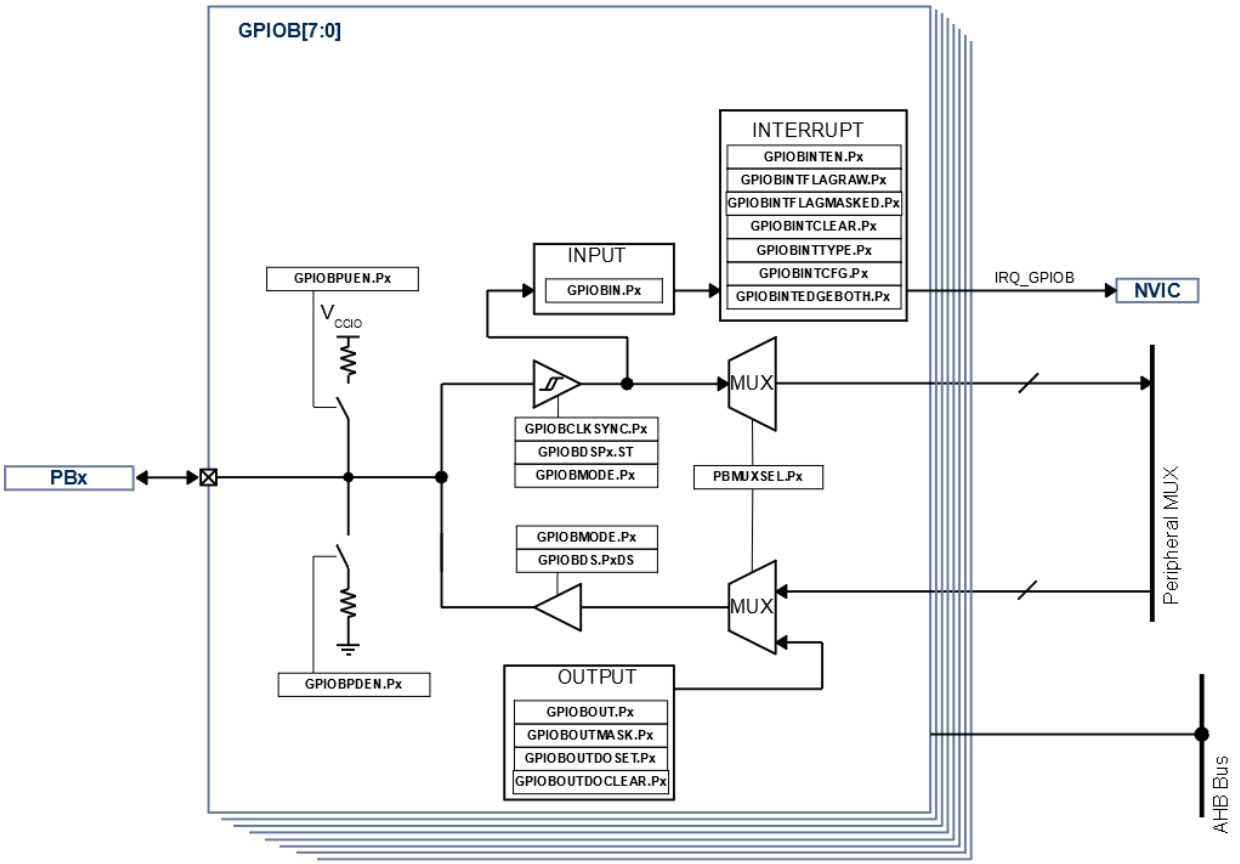
11.1 Overview

The GPIOB port contains 8 GPIO pin, GPIOB0 to GPIOB7.

Each GPIO pin may be used for Digital Input or Digital Output. Each GPIO is capable of generating 3.3V output and is 3.3V input tolerant and has a set of features described below.

11.2 GPIOB Block Diagram

Figure 11-1 GPIOB Block Diagram



11.3 Features

- 3.3V output, 3.3V input tolerant
- Configurable Pin Modes:
  - High-Impedance Digital Input
  - Push-Pull Digital Output
  - Open-Drain Digital Output
- Configurable Drive Strength up to 25mA
- Configurable Input Schmitt Trigger
- Configurable Weak Pull-up or Pull-down
- Edge or Level Sensitive Interrupts
  - Rising, Falling or both edges
- Clock Synchronization

11.4 Functional Description

11.4.1 IO Mode

Each GPIO pin supports up to 4 different modes and can selected from the **GPIOBMODE** register as shown below.

Table 11-1 GPIOB Mode Configuration

GPIOBMODE VALUE	DESCRIPTION
00b	Reserved
01b	Push-Pull Digital Output
10b	Open-Drain Digital Output
11b	High-Impedance Digital Input

The reset value for the GPIO pin mode is High-Impedance Digital Input.

11.4.2 Digital Output

Each GPIO pin may be configured for digital output by setting the **GPIOBMODE.Px** to either 01b (Push-pull output) or to 10b (Open-drain output).

When configured for Push-pull output, the GPIO is driven to the high state using the VCCIO 3.3V power supply. When configured for Open-drain output, the GPIO is pulled up externally.

When **GPIOBOUTMASK.Px** is set to 0b, then the state of the GPIO pin will be set from **GPIOBOUT.Px** (0b: logic low, 1b: logic high). If the **GPIOBOUTMASK.Px** is set to 1b, then any changes to **GPIOBOUT.Px** have no effect.

In either of these two output modes, the user may read the digital state of the input by reading the **GPIOBIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports both digital output modes.

11.4.3 Digital Input

If **GPIOBMODE.Px** is set to 11b, then the GPIO pin is configured for high-impedance input. For application safety, this is the reset value for the GPIO pin mode.

In this mode, the user may read the state of the digital input by reading the **GPIOBIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports this mode.

11.4.4 GPIOB Interrupts

Each GPIO pin supports configurable interrupts. In order for interrupts to work, the GPIO must be set for high-impedance input mode (**GPIOBMODE.Px** = 11b).

The user may configure either edge or level sensitive interrupts for each GPIO pin. If the user configures edge sensitive interrupts, they may select rising, falling or both edges for edge interrupts.

See the table below for how to configure the GPIO for the various types of interrupts.

Table 11-2 GPIOB Interrupt Configuration

GPIOBINTTYPE	GPIOBINTCFG	GPIOBEDGEBOTH	INTERRUPT MODE
0b	0b	0b	Falling Edge
	1b	0b	Rising Edge
	X	1b	Both Rising and Falling Edge
1b	0b	X	Logic Low
	1b	X	Logic High

Interrupts may be enabled or disabled using the **GPIOBINTEN** register. To enable interrupts on a pin, set the **GPIOBINTEN.Px** to 1b. To disable interrupts on a pin, set the **GPIOBINTEN.Px** to 0b.

If an edge or level interrupt is detected, the **GPIOBINTFLAG.Px** will be set to 1b for that GPIO pin. If **GPIOBINTFLAG.Px** is set to 1b and **GPIOBINTEN.Px** is set to 1b, then the IRQ_GPIOB signal to the NVIC is asserted.

To clear the **GPIOBINTFLAG.Px** interrupt flag, write **GPIOBINTCLEAR.Px** to a 1b.

11.4.5 GPIOB Clock Synchronization

Each GPIOB pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOBMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOB pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOBCLKSYNC.Px** = 0b). To enable GPIOB Clock Synchronization set **GPIOBCLKSYNC.Px** = 1b.

11.4.6 GPIOB Drive Strength

Each GPIO pin has configurable drive strength when the ports are configured as a digital output.

See the table below for the settings for GPIOB Drive strength.

Table 11-3 GPIOB Drive Strength

PBDS.PxDS	DRIVE STRENGTH
000b	6mA
001b	8mA
010b	11mA
011b	14mA
100b	17mA
101b	20mA
110b	22mA
111b	25mA

The drive strength shown in the table above represents the minimum drive strength for the GPIO pin. For the details on the electrical characteristics of the drive strength, set the device data sheet.

Note that the register configuration for the GPIOB Drive Strength is located in the System and Clock Control (SCC).

11.4.7 GPIOB Schmitt Trigger

Each GPIO has a Schmitt Trigger that can be enabled when the GPIO is configured as a high-impedance digital input (**GPIOBMODE.Px** = 11b).

The Schmitt Trigger is disabled by default (**PB.PxST** = 0b). To enable the input Schmitt Triger, set **PB.PxST** = 1b.

Note that the register configuration for the GPIO Schmitt Trigger is located in the GPIO and Digital Peripheral MUX (DPM).

11.4.8 GPIOB Weak Pull-up and Pull-down

Each GPIO pin has a configurable weak 60k pull-up to VCCIO or pull-down to ground.

To enable the weak pull-up to VCCIO, set **GPIOBPU.Px** to 1b. To disable the pull-up, set **GPIOBPU.Px** to 0b.

To enable the weak pull-down to ground, set the **GPIOBPD.Px** to 1b. To disable the pull-down, set **GPIOBPD.Px** to 0b.

WARNING: Do not configure the weak pull-up and pull-down to be active at the same time. Doing so may cause device damage.

11.5 Peripheral IO Mapping

The GPIOB peripheral is connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 11-4 GPIOB Peripheral IO Mapping

GPIOB SIGNAL	IO PIN
GPIOB0	PB0
GPIOB1	PB1
GPIOB2	PB2
GPIOB3	PB3
GPIOB4	PB4
GPIOB5	PB5
GPIOB6	PB6
GPIOB7	PB7

11.6 Register Summary

Table 11-5 GPIOB Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOBMODE	400D 1800h	GPIOB Pin Mode Select	RW	0000 FFFFh
GPIOBOUTMASK	400D 1804h	GPIOB Data Output Write Mask	RW	0000 0000h
GPIOBOUT	400D 1808h	GPIOB Data Output Value	RW	0000 00FFh
GPIOBIN	400D 180Ch	GPIOB Data Input Value	RO	--
GPIOBINTEN	400D 1810h	GPIOB Interrupt Enable	RW	0000 0000h
GPIOBINTFLAG	400D 1814h	GPIOB Interrupt Flag	RO	0000 0000h
GPIOBINTCLEAR	400D 181Ch	GPIOB Interrupt Clear	WO	--
GPIOBINTTYPE	400D 1820h	GPIOB Interrupt Type	RW	0000 0000h
GPIOBINTCFG	400D 1824h	GPIOB Interrupt Configuration	RW	0000 0000h
GPIOBINTEDGEBOTH	400D 1828h	GPIOB Interrupt Edge Both	RW	0000 0000h
GPIOBCLKSYNC	400D 182Ch	GPIOB Clock Synchronization	RW	0000 0000h
GPIOBDOSET	400D 1830h	GPIOB Data Output Set	WO	--
GPIOBDOCLEAR	400D 1834h	GPIOB Data Output Clear	WO	--

11.7 Register Detail

11.7.1 GPIOBMODE

Register 11-1 GPIOBMODE (GPIOB Mode Configuration, 400D 1800h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:14	**P7**	RW	11b	Pin 7 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
13:12	**P6**	RW	11b	Pin 6 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
11:10	**P5**	RW	11b	Pin 5 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
9:8	**P4**	RW	11b	Pin 4 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
7:6	**P3**	RW	11b	Pin 3 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
5:4	**P2**	RW	11b	Pin 2 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
3:2	**P1**	RW	11b	Pin 1 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
1:0	**P0**	RW	11b	Pin 0 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input

11.7.2 GPIOBOUTMASK

Register 11-2 GPIOBOUTMASK (GPIOB Output Mask, 400D 1804h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Output Data Mask: 0b: Not masked 1b: Masked
6	**P6**	RW	0	Pin 6 Output Data Mask: 0b: Not masked 1b: Masked
5	**P5**	RW	0	Pin 5 Output Data Mask: 0b: Not masked 1b: Masked
4	**P4**	RW	0	Pin 4 Output Data Mask: 0b: Not masked 1b: Masked
3	**P3**	RW	0	Pin 3 Output Data Mask: 0b: Not masked 1b: Masked
2	**P2**	RW	0	Pin 2 Output Data Mask: 0b: Not masked 1b: Masked
1	**P1**	RW	0	Pin 1 Output Data Mask: 0b: Not masked 1b: Masked
0	**P0**	RW	0	Pin 0 Output Data Mask: 0b: Not masked 1b: Masked

11.7.3 GPIOBOUT

Register 11-3 GPIOBOUT (GPIOB Output Data Value, 400D 1808h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Output Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RW	1	Pin 6 Output Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RW	1	Pin 5 Output Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RW	1	Pin 4 Output Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RW	1	Pin 3 Output Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RW	1	Pin 2 Output Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RW	1	Pin 1 Output Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RW	1	Pin 0 Output Data Value: 0b: Logic low 1b: Logic high

11.7.4 GPIOBIN

Register 11-4 GPIOBIN (GPIOB Input Data Value, 400D 180Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	-	Pin 7 Input Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RO	-	Pin 6 Input Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RO	-	Pin 5 Input Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RO	-	Pin 4 Input Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RO	-	Pin 3 Input Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RO	-	Pin 2 Input Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RO	-	Pin 1 Input Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RO	-	Pin 0 Input Data Value: 0b: Logic low 1b: Logic high

11.7.5 GPIOBINTEN

Register 11-5 GPIOBINTEN (GPIOB Interrupt Enable, 400D 1810h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Enable: 0b: Disabled 1b: Enabled
6	**P6**	RW	0	Pin 6 Interrupt Enable: 0b: Disabled 1b: Enabled
5	**P5**	RW	0	Pin 5 Interrupt Enable: 0b: Disabled 1b: Enabled
4	**P4**	RW	0	Pin 4 Interrupt Enable: 0b: Disabled 1b: Enabled
3	**P3**	RW	0	Pin 3 Interrupt Enable: 0b: Disabled 1b: Enabled
2	**P2**	RW	0	Pin 2 Interrupt Enable: 0b: Disabled 1b: Enabled
1	**P1**	RW	0	Pin 1 Interrupt Enable: 0b: Disabled 1b: Enabled
0	**P0**	RW	0	Pin 0 Interrupt Enable: 0b: Disabled 1b: Enabled

11.7.6 GPIOBINTFLAG

Register 11-6 GPIOBINTFLAG (GPIOB Interrupt Flag, 400D 1814h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	0	Pin 7 Interrupt Flag: 0b: No flag 1b: Flag
6	**P6**	RO	0	Pin 6 Interrupt Flag: 0b: No flag 1b: Flag
5	**P5**	RO	0	Pin 5 Interrupt Flag: 0b: No flag 1b: Flag
4	**P4**	RO	0	Pin 4 Interrupt Flag: 0b: No flag 1b: Flag
3	**P3**	RO	0	Pin 3 Interrupt Flag: 0b: No flag 1b: Flag
2	**P2**	RO	0	Pin 2 Interrupt Flag: 0b: No flag 1b: Flag
1	**P1**	RO	0	Pin 1 Interrupt Flag: 0b: No flag 1b: Flag
0	**P0**	RO	0	Pin 0 Interrupt Flag: 0b: No flag 1b: Flag

11.7.7 GPIOBINTCLEAR

Register 11-7 GPIOBINTCLEAR (GPIOB Interrupt Clear, 400D 181Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Interrupt Clear: 0b: No effect 1b: Clear **GPIOBINTFLAG.P7**
6	**P6**	WO	-	Pin 6 Interrupt Clear: 0b: No effect 1b: Clear **GPIOBINTFLAG.P6**
5	**P5**	WO	-	Pin 5 Interrupt Clear: 0b: No effect 1b: Clear **GPIOBINTFLAG.P5**
4	**P4**	WO	-	Pin 4 Interrupt Clear: 0b: No effect 1b: Clear **GPIOBINTFLAG.P4**
3	**P3**	WO	-	Pin 3 Interrupt Clear: 0b: No effect 1b: Clear **GPIOBINTFLAG.P3**
2	**P2**	WO	-	Pin 2 Interrupt Clear: 0b: No effect 1b: Clear **GPIOBINTFLAG.P2**
1	**P1**	WO	-	Pin 1 Interrupt Clear: 0b: No effect 1b: Clear **GPIOBINTFLAG.P1**
0	**P0**	WO	-	Pin 0 Interrupt Clear: 0b: No effect 1b: Clear **GPIOBINTFLAG.P0**

11.7.8 GPIOBINTTYPE

Register 11-8 GPIOBINTTYPE (GPIOB Interrupt Type Configuration, 400D 1820h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Type: 0b: Edge triggered 1b: Level triggered
6	**P6**	RW	0	Pin 6 Interrupt Type: 0b: Edge triggered 1b: Level triggered
5	**P5**	RW	0	Pin 5 Interrupt Type: 0b: Edge triggered 1b: Level triggered
4	**P4**	RW	0	Pin 4 Interrupt Type: 0b: Edge triggered 1b: Level triggered
3	**P3**	RW	0	Pin 3 Interrupt Type: 0b: Edge triggered 1b: Level triggered
2	**P2**	RW	0	Pin 2 Interrupt Type: 0b: Edge triggered 1b: Level triggered
1	**P1**	RW	0	Pin 1 Interrupt Type: 0b: Edge triggered 1b: Level triggered
0	**P0**	RW	0	Pin 0 Interrupt Type: 0b: Edge triggered 1b: Level triggered

11.7.9 GPIOBINTCFG

Register 11-9 GPIOBINTCFG (GPIOB Interrupt Value Configuration, 400D 1824h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	If **GPIOBINTTYPE.Px** = 0b (edge), Pin 7 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOINTTYPE.Px** = 1b (level), Pin 7 Interrupt Level: 0b: Logic low 1b: Logic high
6	**P6**	RW	0	If **GPIOBINTTYPE.Px** = 0b (edge), Pin 6 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOINTTYPE.Px** = 1b (level), Pin 6 Interrupt Level: 0b: Logic low 1b: Logic high
5	**P5**	RW	0	If **GPIOBINTTYPE.Px** = 0b (edge), Pin 5 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOINTTYPE.Px** = 1b (level), Pin 5 Interrupt Level: 0b: Logic low 1b: Logic high
4	**P4**	RW	0	If **GPIOBINTTYPE.Px** = 0b (edge), Pin 4 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOINTTYPE.Px** = 1b (level), Pin 4 Interrupt Level: 0b: Logic low 1b: Logic high
3	**P3**	RW	0	If **GPIOBINTTYPE.Px** = 0b (edge), Pin 3 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOINTTYPE.Px** = 1b (level), Pin 3 Interrupt Level: 0b: Logic low 1b: Logic high
2	**P2**	RW	0	If **GPIOBINTTYPE.Px** = 0b (edge), Pin 2 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOINTTYPE.Px** = 1b (level), Pin 2 Interrupt Level: 0b: Logic low 1b: Logic high
1	**P1**	RW	0	If **GPIOBINTTYPE.Px** = 0b (edge), Pin 1 Interrupt Edge: 0b: Trigger on falling edge

				1b: Trigger on rising edge If **GPIOINTTYPE.Px** = 1b (level), Pin 1 Interrupt Level: 0b: Logic low 1b: Logic high
0	**P0**	RW	0	If **GPIOBINTTYPE.Px** = 0b (edge), Pin 0 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOINTTYPE.Px** = 1b (level), Pin 0 Interrupt Level: 0b: Logic low 1b: Logic high

11.7.10 GPIOBINTEDGEBOTH

Register 11-10 GPIOBINTEDGEBOTH (GPIOB Interrupt Both Edge Configuration, 400D 1828h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOBINTCFG.P7** 1b: Trigger on both rising and falling edge
6	**P6**	RW	0	Pin 6 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOBINTCFG.P6** 1b: Trigger on both rising and falling edge
5	**P5**	RW	0	Pin 5 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOBINTCFG.P5** 1b: Trigger on both rising and falling edge
4	**P4**	RW	0	Pin 4 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOBINTCFG.P4** 1b: Trigger on both rising and falling edge
3	**P3**	RW	0	Pin 3 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOBINTCFG.P3** 1b: Trigger on both rising and falling edge
2	**P2**	RW	0	Pin 2 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOBINTCFG.P2** 1b: Trigger on both rising and falling edge
1	**P1**	RW	0	Pin 1 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOBINTCFG.P1** 1b: Trigger on both rising and falling edge
0	**P0**	RW	0	Pin 0 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOBINTCFG.P0** 1b: Trigger on both rising and falling edge

11.7.11 GPIOBCLKSYNC[7]

Register 11-11 GPIOBCLKSYNC (GPIOB Clock Synchronization Enable, 400D 182Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
6	**P6**	RW	1	Pin 6 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
5	**P5**	RW	1	Pin 5 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
4	**P4**	RW	1	Pin 4 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
3	**P3**	RW	1	Pin 3 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
2	**P2**	RW	1	Pin 2 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
1	**P1**	RW	1	Pin 1 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
0	**P0**	RW	1	Pin 0 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)

[7] In VER1 of the PAC55XX MCU, the default value for the clock synchronization is disabled (0b).

11.7.12 GPIOBDOSET

Register 11-12 GPIOBDOSET (GPIOB Data Output Set, 400D 1830h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P7** to 1b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P6** to 1b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P5** to 1b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P4** to 1b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P3** to 1b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P2** to 1b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P1** to 1b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P0** to 1b

11.7.13 GPIOBDOCLEAR

Register 11-13 GPIOBDOCLEAR (GPIOB Data Output Clear, 400D 1834h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Clear: 0b: No effect 1b: Set **GPIOBOUT.P7** to 0b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P6** to 0b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P5** to 0b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P4** to 0b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P3** to 0b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P2** to 0b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P1** to 0b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOBOUT.P0** to 0b

12 GPIOC

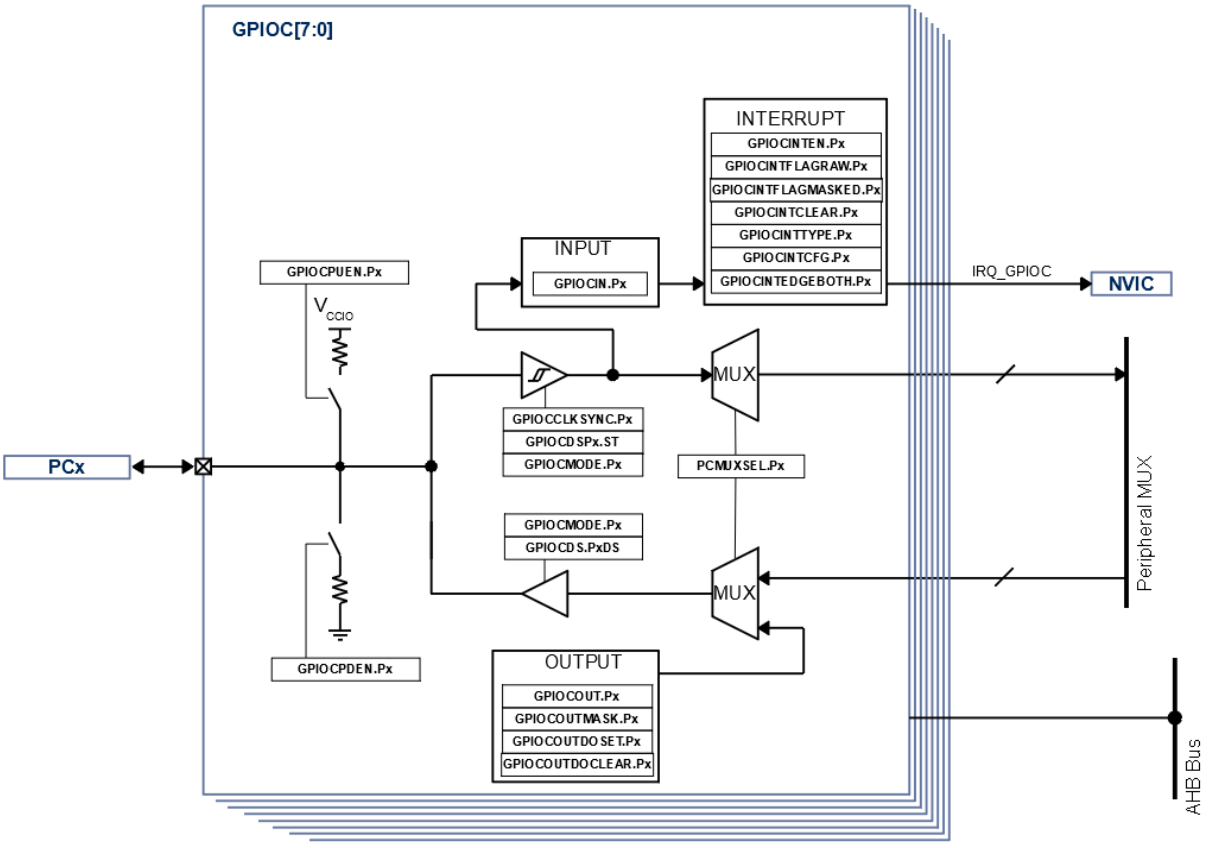
12.1 Overview

The GPIOC port contains 8 GPIO pin, GPIOC0 to GPIOC7.

Each GPIO pin may be used for Digital Input or Digital Output. Each GPIO is capable of generating 3.3V output and is 3.3V input tolerant and has a set of features described below.

12.2 GPIOC Block Diagram

Figure 12-1 GPIOC Block Diagram



12.3 Features

- 3.3V output, 3.3V input tolerant
- Configurable Pin Modes:
  - High-Impedance Digital Input
  - Push-Pull Digital Output
  - Open-Drain Digital Output
- Configurable Drive Strength up to 25mA
- Configurable Input Schmitt Trigger
- Configurable Weak Pull-up or Pull-down
- Edge or Level Sensitive Interrupts
  - Rising, Falling or both edges
- Clock Synchronization

12.4 Functional Description

12.4.1 IO Mode

Each GPIO pin supports up to 4 different modes and can selected from the **GPIOCMODE** register as shown below.

Table 12-1 GPIOC Mode Configuration

GPIOCMODE VALUE	DESCRIPTION
00b	Reserved
01b	Push-Pull Digital Output
10b	Open-Drain Digital Output
11b	High-Impedance Digital Input

The reset value for the GPIO pin mode is High-Impedance Digital Input.

12.4.2 Digital Output

Each GPIO pin may be configured for digital output by setting the **GPIOCMODE.Px** to either 01b (Push-pull output) or to 10b (Open-drain output).

When configured for Push-pull output, the GPIO is driven to the high state using the VCCIO 3.3V power supply. When configured for Open-drain output, the GPIO is pulled up externally.

When **GPIOCOUTMASK.Px** is set to 0b, then the state of the GPIO pin will be set from **GPIOCOUT.Px** (0b: logic low, 1b: logic high). If the **GPIOCOUTMASK.Px** is set to 1b, then any changes to **GPIOCOUT.Px** have no effect.

In either of these two output modes, the user may read the digital state of the input by reading the **GPIOCIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports both digital output modes.

12.4.3 Digital Input

If **GPIOCMODE.Px** is set to 11b, then the GPIO pin is configured for high-impedance input. For application safety, this is the reset value for the GPIO pin mode.

In this mode, the user may read the state of the digital input by reading the **GPIOCIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports this mode.

12.4.4 GPIOC Interrupts

Each GPIO pin supports configurable interrupts. In order for interrupts to work, the GPIO must be set for high-impedance input mode (**GPIOCMODE.Px** = 11b).

The user may configure either edge or level sensitive interrupts for each GPIO pin. If the user configures edge sensitive interrupts, they may select rising, falling or both edges for edge interrupts.

See the table below for how to configure the GPIO for the various types of interrupts.

Table 12-2 GPIOC Interrupt Configuration

GPIOCINTTYPE	GPIOCINTCFG	GPIOCEDGEBOTH	INTERRUPT MODE
0b	0b	0b	Falling Edge
	1b	0b	Rising Edge
	X	1b	Both Rising and Falling Edge
1b	0b	X	Logic Low
	1b	X	Logic High

Interrupts may be enabled or disabled using the **GPIOCINTEN** register. To enable interrupts on a pin, set the **GPIOCINTEN.Px** to 1b. To disable interrupts on a pin, set the **GPIOCINTEN.Px** to 0b.

If an edge or level interrupt is detected, the **GPIOCINTFLAG.Px** will be set to 1b for that GPIO pin. If **GPIOCINTFLAG.Px** is set to 1b and **GPIOCINTEN.Px** is set to 1b, then the IRQ_GPIOC signal to the NVIC is asserted.

To clear the **GPIOCINTFLAG.Px** interrupt flag, write **GPIOCINTCLEAR.Px** to a 1b.

12.4.5 GPIOC Clock Synchronization

Each GPIOC pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOCMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOC pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOCCLKSYNC.Px** = 0b). To enable GPIOC Clock Synchronization set **GPIOCCLKSYNC.Px** = 1b.

12.4.6 GPIOC Drive Strength

Each GPIO pin has configurable drive strength when the ports are configured as a digital output.

See the table below for the settings for GPIOC Drive strength.

Table 12-3 GPIOC Drive Strength

PCDS.PxDS	DRIVE STRENGTH
000b	6mA
001b	8mA
010b	11mA
011b	14mA
100b	17mA
101b	20mA
110b	22mA
111b	25mA

The drive strength shown in the table above represents the minimum drive strength for the GPIO pin. For the details on the electrical characteristics of the drive strength, set the device data sheet.

Note that the register configuration for the GPIOC Drive Strength is in the System and Clock Control (SCC).

12.4.7 GPIOC Schmitt Trigger

Each GPIO has a Schmitt Trigger that can be enabled when the GPIO is configured as a high-impedance digital input (**GPIOCMODE.Px** = 11b).

The Schmitt Trigger is disabled by default (**PCDS.PxST** = 0b). To enable the input Schmitt Triger, set **PCDS.PxST** = 1b.

Note that the register configuration for the GPIO Schmitt Trigger is in the GPIO and Digital Peripheral MUX (DPM).

12.4.8 GPIOC Weak Pull-up and Pull-down

Each GPIO pin has a configurable weak 60k pull-up to VCCIO or pull-down to ground.

To enable the weak pull-up to VCCIO, set **GPIOCPU.Px** to 1b. To disable the pull-up, set **GPIOCPU.Px** to 0b.

To enable the weak pull-down to ground, set the **GPIOCPD.Px** to 1b. To disable the pull-down, set **GPIOCPD.Px** to 0b.

WARNING: Do not configure the weak pull-up and pull-down to be active at the same time. Doing so may cause device damage.

12.5 Peripheral IO Mapping

The GPIOC peripheral is connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 12-4 GPIOC Peripheral IO Mapping

GPIOC SIGNAL	IO PIN
GPIOC0	PC0
GPIOC1	PC1
GPIOC2	PC2
GPIOC3	PC3
GPIOC4	PC4
GPIOC5	PC5
GPIOC6	PC6
GPIOC7	PC7

12.6 Register Summary

Table 12-5 GPIOC Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOCMODE	400D 1C00h	GPIOC Pin Mode Select	RW	0000 FFFFh
GPIOCOUTMASK	400D 1C04h	GPIOC Data Output Write Mask	RW	0000 0000h
GPIOCOUT	400D 1C08h	GPIOC Data Output Value	RW	0000 00FFh
GPIOCIN	400D 1C0Ch	GPIOC Data Input Value	RO	--
GPIOCINTEN	400D 1C10h	GPIOC Interrupt Enable	RW	0000 0000h
GPIOCINTFLAG	400D 1C14h	GPIOC Interrupt Flag	RO	0000 0000h
GPIOCINTCLEAR	400D 1C1Ch	GPIOC Interrupt Clear	WO	--
GPIOCINTTYPE	400D 1C20h	GPIOC Interrupt Type	RW	0000 0000h
GPIOCINTCFG	400D 1C24h	GPIOC Interrupt Configuration	RW	0000 0000h
GPIOCINTEDGEBOTH	400D 1C28h	GPIOC Interrupt Edge Both	RW	0000 0000h
GPIOCCLKSYNC	400D 1C2Ch	GPIOC Clock Synchronization	RW	0000 0000h
GPIOCDOSET	400D 1C30h	GPIOC Data Output Set	WO	--
GPIOCDOCLEAR	400D 1C34h	GPIOC Data Output Clear	WO	--

12.7 Register Detail

12.7.1 GPIOCMODE

Register 12-1 GPIOCMODE (GPIOC Mode Configuration, 400D 1C00h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:14	**P7**	RW	11b	Pin 7 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
13:12	**P6**	RW	11b	Pin 6 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
11:10	**P5**	RW	11b	Pin 5 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
9:8	**P4**	RW	11b	Pin 4 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
7:6	**P3**	RW	11b	Pin 3 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
5:4	**P2**	RW	11b	Pin 2 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
3:2	**P1**	RW	11b	Pin 1 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
1:0	**P0**	RW	11b	Pin 0 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input

12.7.2 GPIOCOUTMASK

Register 12-2 GPIOCOUTMASK (GPIOC Output Mask, 400D 1C04h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Output Data Mask: 0b: Not masked 1b: Masked
6	**P6**	RW	0	Pin 6 Output Data Mask: 0b: Not masked 1b: Masked
5	**P5**	RW	0	Pin 5 Output Data Mask: 0b: Not masked 1b: Masked
4	**P4**	RW	0	Pin 4 Output Data Mask: 0b: Not masked 1b: Masked
3	**P3**	RW	0	Pin 3 Output Data Mask: 0b: Not masked 1b: Masked
2	**P2**	RW	0	Pin 2 Output Data Mask: 0b: Not masked 1b: Masked
1	**P1**	RW	0	Pin 1 Output Data Mask: 0b: Not masked 1b: Masked
0	**P0**	RW	0	Pin 0 Output Data Mask: 0b: Not masked 1b: Masked

12.7.3 GPIOCOUT

Register 12-3 GPIOCOUT (GPIOC Output Data Value, 400D 1C08h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Output Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RW	1	Pin 6 Output Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RW	1	Pin 5 Output Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RW	1	Pin 4 Output Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RW	1	Pin 3 Output Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RW	1	Pin 2 Output Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RW	1	Pin 1 Output Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RW	1	Pin 0 Output Data Value: 0b: Logic low 1b: Logic high

12.7.4 GPIOCIN

Register 12-4 GPIOCIN (GPIOC Input Data Value, 400D 1C0Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	-	Pin 7 Input Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RO	-	Pin 6 Input Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RO	-	Pin 5 Input Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RO	-	Pin 4 Input Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RO	-	Pin 3 Input Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RO	-	Pin 2 Input Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RO	-	Pin 1 Input Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RO	-	Pin 0 Input Data Value: 0b: Logic low 1b: Logic high

12.7.5 GPIOCINTEN

Register 12-5 GPIOCINTEN (GPIOC Interrupt Enable, 400D 1C10h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Enable: 0b: Disabled 1b: Enabled
6	**P6**	RW	0	Pin 6 Interrupt Enable : 0b: Disabled 1b: Enabled
5	**P5**	RW	0	Pin 5 Interrupt Enable: 0b: Disabled 1b: Enabled
4	**P4**	RW	0	Pin 4 Interrupt Enable: 0b: Disabled 1b: Enabled
3	**P3**	RW	0	Pin 3 Interrupt Enable: 0b: Disabled 1b: Enabled
2	**P2**	RW	0	Pin 2 Interrupt Enable: 0b: Disabled 1b: Enabled
1	**P1**	RW	0	Pin 1 Interrupt Enable: 0b: Disabled 1b: Enabled
0	**P0**	RW	0	Pin 0 Interrupt Enable: 0b: Disabled 1b: Enabled

12.7.6 GPIOCINTFLAG

Register 12-6 GPIOCINTFLAG (GPIOC Interrupt Flag, 400D 1C14h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	0	Pin 7 Interrupt Flag: 0b: No flag 1b: Flag
6	**P6**	RO	0	Pin 6 Interrupt Flag: 0b: No flag 1b: Flag
5	**P5**	RO	0	Pin 5 Interrupt Flag: 0b: No flag 1b: Flag
4	**P4**	RO	0	Pin 4 Interrupt Flag: 0b: No flag 1b: Flag
3	**P3**	RO	0	Pin 3 Interrupt Flag: 0b: No flag 1b: Flag
2	**P2**	RO	0	Pin 2 Interrupt Flag: 0b: No flag 1b: Flag
1	**P1**	RO	0	Pin 1 Interrupt Flag: 0b: No flag 1b: Flag
0	**P0**	RO	0	Pin 0 Interrupt Flag: 0b: No flag 1b: Flag

12.7.7 GPIOCINTCLEAR

Register 12-7 GPIOCINTCLEAR (GPIOC Interrupt Clear, 400D 1C1Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Interrupt Clear: 0b: No effect 1b: Clear **GPIOCINTFLAG.P7**
6	**P6**	WO	-	Pin 6 Interrupt Clear: 0b: No effect 1b: Clear **GPIOCINTFLAG.P6**
5	**P5**	WO	-	Pin 5 Interrupt Clear: 0b: No effect 1b: Clear **GPIOCINTFLAG.P5**
4	**P4**	WO	-	Pin 4 Interrupt Clear: 0b: No effect 1b: Clear **GPIOCINTFLAG.P4**
3	**P3**	WO	-	Pin 3 Interrupt Clear: 0b: No effect 1b: Clear **GPIOCINTFLAG.P3**
2	**P2**	WO	-	Pin 2 Interrupt Clear: 0b: No effect 1b: Clear **GPIOCINTFLAG.P2**
1	**P1**	WO	-	Pin 1 Interrupt Clear: 0b: No effect 1b: Clear **GPIOCINTFLAG.P1**
0	**P0**	WO	-	Pin 0 Interrupt Clear: 0b: No effect 1b: Clear **GPIOCINTFLAG.P0**

12.7.8 GPIOCINTTYPE

Register 12-8 GPIOCINTTYPE (GPIOC Interrupt Type Configuration, 400D 1C20h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Type: 0b: Edge triggered 1b: Level triggered
6	**P6**	RW	0	Pin 6 Interrupt Type: 0b: Edge triggered 1b: Level triggered
5	**P5**	RW	0	Pin 5 Interrupt Type: 0b: Edge triggered 1b: Level triggered
4	**P4**	RW	0	Pin 4 Interrupt Type: 0b: Edge triggered 1b: Level triggered
3	**P3**	RW	0	Pin 3 Interrupt Type: 0b: Edge triggered 1b: Level triggered
2	**P2**	RW	0	Pin 2 Interrupt Type: 0b: Edge triggered 1b: Level triggered
1	**P1**	RW	0	Pin 1 Interrupt Type: 0b: Edge triggered 1b: Level triggered
0	**P0**	RW	0	Pin 0 Interrupt Type: 0b: Edge triggered 1b: Level triggered

12.7.9 GPIOCINTCFG

Register 12-9 GPIOCINTCFG (GPIOC Interrupt Configuration, 400D 1C24h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	If **GPIOCINTTYPE.Px** = 0b (edge), Pin 7 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOCINTTYPE.Px** = 1b (level), Pin 7 Interrupt Level: 0b: Logic low 1b: Logic high
6	**P6**	RW	0	If **GPIOCINTTYPE.Px** = 0b (edge), Pin 6 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOCINTTYPE.Px** = 1b (level), Pin 6 Interrupt Level: 0b: Logic low 1b: Logic high
5	**P5**	RW	0	If **GPIOCINTTYPE.Px** = 0b (edge), Pin 5 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOCINTTYPE.Px** = 1b (level), Pin 5 Interrupt Level: 0b: Logic low 1b: Logic high
4	**P4**	RW	0	If **GPIOCINTTYPE.Px** = 0b (edge), Pin 4 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOCINTTYPE.Px** = 1b (level), Pin 4 Interrupt Level: 0b: Logic low 1b: Logic high
3	**P3**	RW	0	If **GPIOCINTTYPE.Px** = 0b (edge), Pin 3 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOCINTTYPE.Px** = 1b (level), Pin 3 Interrupt Level: 0b: Logic low 1b: Logic high
2	**P2**	RW	0	If **GPIOCINTTYPE.Px** = 0b (edge), Pin 2 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOCINTTYPE.Px** = 1b (level), Pin 2 Interrupt Level: 0b: Logic low 1b: Logic high
1	**P1**	RW	0	If **GPIOCINTTYPE.Px** = 0b (edge), Pin 1 Interrupt Edge: 0b: Trigger on falling edge

				1b: Trigger on rising edge If **GPIOCINTTYPE.Px** = 1b (level), Pin 1 Interrupt Level: 0b: Logic low 1b: Logic high
0	**P0**	RW	0	If **GPIOCINTTYPE.Px** = 0b (edge), Pin 0 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOCINTTYPE.Px** = 1b (level), Pin 0 Interrupt Level: 0b: Logic low 1b: Logic high

12.7.10 GPIOCINTEDGEBOTH

Register 12-10 GPIOCINTEDGEBOTH (GPIOC Interrupt Both Edge Configuration, 400D 1C28h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOCINTCFG.P7** 1b: Trigger on both rising and falling edge
6	**P6**	RW	0	Pin 6 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOCINTCFG.P6** 1b: Trigger on both rising and falling edge
5	**P5**	RW	0	Pin 5 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOCINTCFG.P5** 1b: Trigger on both rising and falling edge
4	**P4**	RW	0	Pin 4 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOCINTCFG.P4** 1b: Trigger on both rising and falling edge
3	**P3**	RW	0	Pin 3 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOCINTCFG.P3** 1b: Trigger on both rising and falling edge
2	**P2**	RW	0	Pin 2 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOCINTCFG.P2** 1b: Trigger on both rising and falling edge
1	**P1**	RW	0	Pin 1 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOCINTCFG.P1** 1b: Trigger on both rising and falling edge
0	**P0**	RW	0	Pin 0 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOCINTCFG.P0** 1b: Trigger on both rising and falling edge

12.7.11 GPIOCCLKSYNC[8]

Register 12-11 GPIOCCLKSYNC (GPIOC Clock Synchronization Enable, 400D 1C2Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
6	**P6**	RW	1	Pin 6 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
5	**P5**	RW	1	Pin 5 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
4	**P4**	RW	1	Pin 4 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
3	**P3**	RW	1	Pin 3 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
2	**P2**	RW	1	Pin 2 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
1	**P1**	RW	1	Pin 1 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
0	**P0**	RW	1	Pin 0 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)

[8] In VER1 of the PAC55XX MCU, the default value for the clock synchronization is disabled (0b).

12.7.12 GPIOCDOSET

Register 12-12 GPIOCDOSET (GPIOC Data Output Set, 400D 1C30h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P7** to 1b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P6** to 1b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P5** to 1b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P4** to 1b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P3** to 1b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P2** to 1b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P1** to 1b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P0** to 1b

12.7.13 GPIOCDOCLEAR

Register 12-13 GPIOCDOCLEAR (GPIOC Data Output Clear, 400D 1C34h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Clear: 0b: No effect 1b: Set **GPIOCOUT.P7** to 0b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P6** to 0b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P5** to 0b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P4** to 0b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P3** to 0b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P2** to 0b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P1** to 0b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOCOUT.P0** to 0b

13 GPIOD

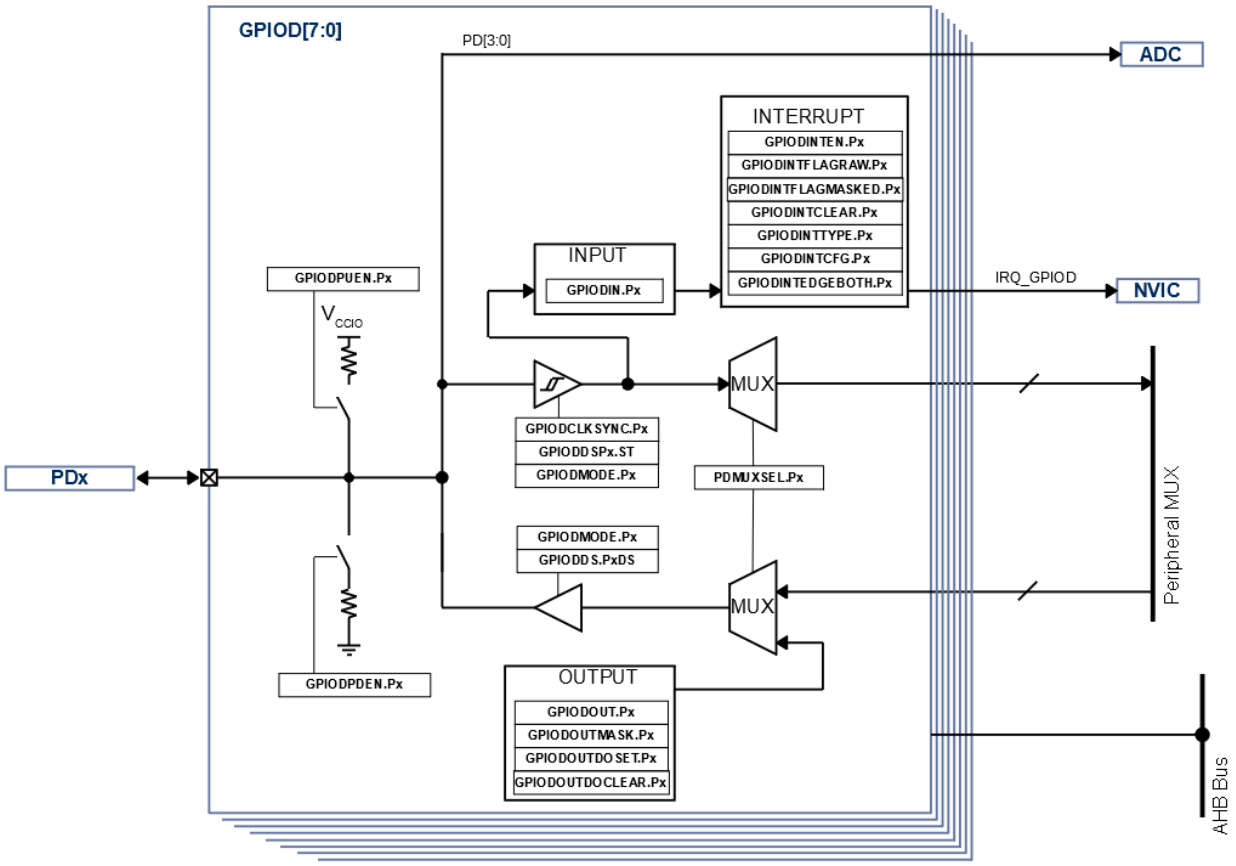
13.1 Overview

The GPIOD port contains 8 GPIO pin, GPIOD0 to GPIOD7.

Each GPIO pin may be used for Digital Input, Digital Output and some may be used for Analog Input.[9] Each GPIO is capable of generating 3.3V output and is 3.3V input tolerant and has a set of features described below.

13.2 GPIOD Block Diagram

Figure 13-1 GPIOD Block Diagram



[9] Consult the device data sheet for information on which GPIO pins are present on any given PAC55XX device.

13.3 Features

- 3.3V output, 3.3V input tolerant
- Configurable Pin Modes:
  - High-Impedance Digital Input
  - Push-Pull Digital Output
  - Open-Drain Digital Output
  - Analog Input
- Configurable Drive Strength up to 25mA
- Configurable Input Schmitt Trigger
- Configurable Weak Pull-up or Pull-down
- Edge or Level Sensitive Interrupts
  - Rising, Falling or both edges
- Clock Synchronization

13.4 Functional Description

13.4.1 IO Mode

Each GPIO pin supports up to 4 different modes and can selected from the **GPIODMODE** register as shown below.

Table 13-1 GPIOD Mode Configuration

GPIODMODE VALUE	DESCRIPTION
00b	Analog Input
01b	Push-Pull Digital Output
10b	Open-Drain Digital Output
11b	High-Impedance Digital Input

The reset value for the GPIO pin mode is High-Impedance Digital Input.

13.4.2 Digital Output

Each GPIO pin may be configured for digital output by setting the **GPIODMODE.Px** to either 01b (Push-pull output) or to 10b (Open-drain output).

When configured for Push-pull output, the GPIO is driven to the high state using the VCCIO 3.3V power supply. When configured for Open-drain output, the GPIO is pulled up externally.

When **GPIOCOUTMASK.Px** is set to 0b, then the state of the GPIO pin will be set from **GPIOCOUT.Px** (0b: logic low, 1b: logic high). If the **GPIOCOUTMASK.Px** is set to 1b, then any changes to **GPIOCOUT.Px** have no effect.

In either of these two output modes, the user may read the digital state of the input by reading the **GPIODIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports both digital output modes.

13.4.3 Digital Input

If **GPIODMODE.Px** is set to 11b, then the GPIO pin is configured for high-impedance input. For application safety, this is the reset value for the GPIO pin mode.

In this mode, the user may read the state of the digital input by reading the **GPIODIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports this mode.

13.4.4 Analog Input

Some GPIO pins may be configured for Analog Input.

To configure a GPIO pin for Analog Input, the user may set **GPIODMODE.Px** to 00b. In this mode, the user MUST be sure to set the output mode to high-impedance input by setting **GPIODMODE.Px** to 11b or the Analog Input will not work properly.

To determine if your device supports Analog Input for each GPIO pin, consult the device data sheet.

13.4.5 GPIOD Interrupts

Each GPIO pin supports configurable interrupts. In order for interrupts to work, the GPIO must be set for high-impedance input mode (**GPIODMODE.Px** = 11b).

The user may configure either edge or level sensitive interrupts for each GPIO pin. If the user configures edge sensitive interrupts, they may select rising, falling or both edges for edge interrupts.

See the table below for how to configure the GPIO for the various types of interrupts.

Table 13-2 GPIOD Interrupt Configuration

GPIODINTTYPE	GPIODINTCFG	GPIODEDGEBOTH	INTERRUPT MODE
0b	0b	0b	Falling Edge
	1b	0b	Rising Edge
	X	1b	Both Rising and Falling Edge
1b	0b	X	Logic Low
	1b	X	Logic High

Interrupts may be enabled or disabled using the **GPIODINTEN** register. To enable interrupts on a pin, set the **GPIODINTEN.Px** to 1b. To disable interrupts on a pin, set the **GPIODINTEN.Px** to 0b.

If an edge or level interrupt is detected, the **GPIODINTFLAG.Px** will be set to 1b for that GPIO pin. If **GPIODINTFLAG.Px** is set to 1b and **GPIODINTEN.Px** is set to 1b, then the IRQ_GPIOD signal to the NVIC is asserted.

To clear the **GPIODINTFLAG.Px** interrupt flag, write **GPIODINTCLEAR.Px** to a 1b.

13.4.6 GPIOD Clock Synchronization

Each GPIOD pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIODMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOD pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIODCLKSYNC.Px** = 0b). To enable GPIOD Clock Synchronization set **GPIODCLKSYNC.Px** = 1b.

13.4.7 GPIOD Drive Strength

Each GPIO pin has configurable drive strength when the ports are configured as a digital output.

See the table below for the settings for GPIOD Drive strength.

Table 13-3 GPIOD Drive Strength

PDDS.PxDS	DRIVE STRENGTH
000b	6mA
001b	8mA
010b	11mA
011b	14mA
100b	17mA
101b	20mA
110b	22mA
111b	25mA

The drive strength shown in the table above represents the minimum drive strength for the GPIO pin. For the details on the electrical characteristics of the drive strength, set the device data sheet.

Note that the register configuration for the GPIOD Drive Strength is in the System and Clock Control (SCC).

13.4.8 GPIOD Schmitt Trigger

Each GPIO has a Schmitt Trigger that can be enabled when the GPIO is configured as a high-impedance digital input (**GPIODMODE.Px** = 11b).

The Schmitt Trigger is disabled by default (**PDDS.PxST** = 0b). To enable the input Schmitt Triger, set **PDDS.PxST** = 1b.

Note that the register configuration for the GPIO Schmitt Trigger is in the GPIO and Digital Peripheral MUX (DPM).

13.4.9 GPIOD Weak Pull-up and Pull-down

Each GPIO pin has a configurable weak 60k pull-up to VCCIO or pull-down to ground.

To enable the weak pull-up to VCCIO, set **GPIODPU.Px** to 1b. To disable the pull-up, set **GPIODPU.Px** to 0b.

To enable the weak pull-down to ground, set the **GPIODPD.Px** to 1b. To disable the pull-down, set **GPIODPD.Px** to 0b.

WARNING: Do not configure the weak pull-up and pull-down to be active at the same time. Doing so may cause device damage.

13.5 Peripheral IO Mapping

The GPIOD peripheral is connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 13-4 GPIOD Peripheral IO Mapping

GPIOD SIGNAL	IO PIN
GPIOD0	PD0
GPIOD1	PD1
GPIOD2	PD2
GPIOD3	PD3
GPIOD4	PD4
GPIOD5	PD5
GPIOD6	PD6
GPIOD7	PD7

13.6 Register Summary

Table 13-5 GPIOD Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIODMODE	400D 2000h	GPIOD Pin Mode Select	RW	0000 FFFFh
GPIODOUTMASK	400D 2004h	GPIOD Data Output Write Mask	RW	0000 0000h
GPIODOUT	400D 2008h	GPIOD Data Output Value	RW	0000 00FFh
GPIODIN	400D 200Ch	GPIOD Data Input Value	RO	--
GPIODINTEN	400D 2010h	GPIOD Interrupt Enable	RW	0000 0000h
GPIODINTFLAG	400D 2014h	GPIOD Interrupt Flag	RO	0000 0000h
GPIODINTCLEAR	400D 201Ch	GPIOD Interrupt Clear	WO	0000 0000h
GPIODINTTYPE	400D 2020h	GPIOD Interrupt Type	RW	0000 0000h
GPIODINTCFG	400D 2024h	GPIOD Interrupt Configuration	RW	0000 0000h
GPIODINTEDGEBOTH	400D 2028h	GPIOD Interrupt Edge Both	RW	0000 0000h
GPIODCLKSYNC	400D 202Ch	GPIOD Clock Synchronization	RW	--
GPIODDOSET	400D 2030h	GPIOD Data Output Set	WO	--
GPIODDOCLEAR	400D 2034h	GPIOD Data Output Clear	WO	0000 FFFFh

13.7 Register Detail

13.7.1 GPIODMODE

Register 13-1 GPIODMODE (GPIOD Mode Configuration, 400D 2000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:14	**P7**	RW	11b	Pin 7 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
13:12	**P6**	RW	11b	Pin 6 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
11:10	**P5**	RW	11b	Pin 5 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
9:8	**P4**	RW	11b	Pin 4 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
7:6	**P3**	RW	11b	Pin 3 Mode: 00b: Analog Input (ADC1) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
5:4	**P2**	RW	11b	Pin 2 Mode: 00b: Analog Input (ADC2) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
3:2	**P1**	RW	11b	Pin 1 Mode: 00b: Analog Input (ADC3) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
1:0	**P0**	RW	11b	Pin 0 Mode: 00b: Analog Input (ADC4) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input

13.7.2 GPIODOUTMASK

Register 13-2 GPIODOUTMASK (GPIOD Output Mask, 400D 2004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Output Data Mask: 0b: Not masked 1b: Masked
6	**P6**	RW	0	Pin 6 Output Data Mask: 0b: Not masked 1b: Masked
5	**P5**	RW	0	Pin 5 Output Data Mask: 0b: Not masked 1b: Masked
4	**P4**	RW	0	Pin 4 Output Data Mask: 0b: Not masked 1b: Masked
3	**P3**	RW	0	Pin 3 Output Data Mask: 0b: Not masked 1b: Masked
2	**P2**	RW	0	Pin 2 Output Data Mask: 0b: Not masked 1b: Masked
1	**P1**	RW	0	Pin 1 Output Data Mask: 0b: Not masked 1b: Masked
0	**P0**	RW	0	Pin 0 Output Data Mask: 0b: Not masked 1b: Masked

13.7.3 GPIODOUT

Register 13-3 GPIODOUT (GPIOD Output Data Value, 400D 2008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Output Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RW	1	Pin 6 Output Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RW	1	Pin 5 Output Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RW	1	Pin 4 Output Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RW	1	Pin 3 Output Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RW	1	Pin 2 Output Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RW	1	Pin 1 Output Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RW	1	Pin 0 Output Data Value: 0b: Logic low 1b: Logic high

13.7.4 GPIODIN

Register 13-4 GPIODIN (GPIOD Input Data Value, 400D 200Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	-	Pin 7 Input Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RO	-	Pin 6 Input Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RO	-	Pin 5 Input Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RO	-	Pin 4 Input Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RO	-	Pin 3 Input Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RO	-	Pin 2 Input Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RO	-	Pin 1 Input Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RO	-	Pin 0 Input Data Value: 0b: Logic low 1b: Logic high

13.7.5 GPIODINTEN

Register 13-5 GPIODINTEN (GPIOD Interrupt Enable, 400D 2010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Enable: 0b: Disabled 1b: Enabled
6	**P6**	RW	0	Pin 6 Interrupt Enable: 0b: Disabled 1b: Enabled
5	**P5**	RW	0	Pin 5 Interrupt Enable: 0b: Disabled 1b: Enabled
4	**P4**	RW	0	Pin 4 Interrupt Enable: 0b: Disabled 1b: Enabled
3	**P3**	RW	0	Pin 3 Interrupt Enable: 0b: Disabled 1b: Enabled
2	**P2**	RW	0	Pin 2 Interrupt Enable: 0b: Disabled 1b: Enabled
1	**P1**	RW	0	Pin 1 Interrupt Enable: 0b: Disabled 1b: Enabled
0	**P0**	RW	0	Pin 0 Interrupt Enable: 0b: Disabled 1b: Enabled

13.7.6 GPIODINTFLAG

Register 13-6 GPIODINTFLAG (GPIOD Interrupt Flag, 400D 2014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	0	Pin 7 Interrupt Flag: 0b: No flag 1b: Flag
6	**P6**	RO	0	Pin 6 Interrupt Flag: 0b: No flag 1b: Flag
5	**P5**	RO	0	Pin 5 Interrupt Flag: 0b: No flag 1b: Flag
4	**P4**	RO	0	Pin 4 Interrupt Flag: 0b: No flag 1b: Flag
3	**P3**	RO	0	Pin 3 Interrupt Flag: 0b: No flag 1b: Flag
2	**P2**	RO	0	Pin 2 Interrupt Flag: 0b: No flag 1b: Flag
1	**P1**	RO	0	Pin 1 Interrupt Flag: 0b: No flag 1b: Flag
0	**P0**	RO	0	Pin 0 Interrupt Flag: 0b: No flag 1b: Flag

13.7.7 GPIODINTCLEAR

Register 13-7 GPIODINTCLEAR (GPIOD Interrupt Clear, 400D 201Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Interrupt Clear: 0b: No effect 1b: Clear **GPIODINTFLAG.P7**
6	**P6**	WO	-	Pin 6 Interrupt Clear: 0b: No effect 1b: Clear **GPIODINTFLAG.P6**
5	**P5**	WO	-	Pin 5 Interrupt Clear: 0b: No effect 1b: Clear **GPIODINTFLAG.P5**
4	**P4**	WO	-	Pin 4 Interrupt Clear: 0b: No effect 1b: Clear **GPIODINTFLAG.P4**
3	**P3**	WO	-	Pin 3 Interrupt Clear: 0b: No effect 1b: Clear **GPIODINTFLAG.P3**
2	**P2**	WO	-	Pin 2 Interrupt Clear: 0b: No effect 1b: Clear **GPIODINTFLAG.P2**
1	**P1**	WO	-	Pin 1 Interrupt Clear: 0b: No effect 1b: Clear **GPIODINTFLAG.P1**
0	**P0**	WO	-	Pin 0 Interrupt Clear: 0b: No effect 1b: Clear **GPIODINTFLAG.P0**

13.7.8 GPIODINTTYPE

Register 13-8 GPIODINTTYPE (GPIOD Interrupt Type Configuration, 400D 2020h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Type: 0b: Edge triggered 1b: Level triggered
6	**P6**	RW	0	Pin 6 Interrupt Type: 0b: Edge triggered 1b: Level triggered
5	**P5**	RW	0	Pin 5 Interrupt Type: 0b: Edge triggered 1b: Level triggered
4	**P4**	RW	0	Pin 4 Interrupt Type: 0b: Edge triggered 1b: Level triggered
3	**P3**	RW	0	Pin 3 Interrupt Type: 0b: Edge triggered 1b: Level triggered
2	**P2**	RW	0	Pin 2 Interrupt Type: 0b: Edge triggered 1b: Level triggered
1	**P1**	RW	0	Pin 1 Interrupt Type: 0b: Edge triggered 1b: Level triggered
0	**P0**	RW	0	Pin 0 Interrupt Type: 0b: Edge triggered 1b: Level triggered

13.7.9 GPIODINTCFG

Register 13-9 GPIODINTCFG (GPIOD Interrupt Configuration, 400D 2024h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	If **GPIODINTTYPE.Px** = 0b (edge), Pin 7 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIODINTTYPE.Px** = 1b (level), Pin 7 Interrupt Level: 0b: Logic low 1b: Logic high
6	**P6**	RW	0	If **GPIODINTTYPE.Px** = 0b (edge), Pin 6 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIODINTTYPE.Px** = 1b (level), Pin 6 Interrupt Level: 0b: Logic low 1b: Logic high
5	**P5**	RW	0	If **GPIODINTTYPE.Px** = 0b (edge), Pin 5 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIODINTTYPE.Px** = 1b (level), Pin 5 Interrupt Level: 0b: Logic low 1b: Logic high
4	**P4**	RW	0	If **GPIODINTTYPE.Px** = 0b (edge), Pin 4 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIODINTTYPE.Px** = 1b (level), Pin 4 Interrupt Level: 0b: Logic low 1b: Logic high
3	**P3**	RW	0	If **GPIODINTTYPE.Px** = 0b (edge), Pin 3 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIODINTTYPE.Px** = 1b (level), Pin 3 Interrupt Level: 0b: Logic low 1b: Logic high
2	**P2**	RW	0	If **GPIODINTTYPE.Px** = 0b (edge), Pin 2 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIODINTTYPE.Px** = 1b (level), Pin 2 Interrupt Level: 0b: Logic low 1b: Logic high
1	**P1**	RW	0	If **GPIODINTTYPE.Px** = 0b (edge), Pin 1 Interrupt Edge: 0b: Trigger on falling edge

				1b: Trigger on rising edge If **GPIODINTTYPE.Px** = 1b (level), Pin 1 Interrupt Level: 0b: Logic low 1b: Logic high
0	**P0**	RW	0	If **GPIODINTTYPE.Px** = 0b (edge), Pin 0 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIODINTTYPE.Px** = 1b (level), Pin 0 Interrupt Level: 0b: Logic low 1b: Logic high

13.7.10 GPIODINTEDGEBOTH

Register 13-10 GPIODINTEDGEBOTH (GPIOD Interrupt Both Edge Configuration, 400D 2028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIODINTCFG.P7** 1b: Trigger on both rising and falling edge
6	**P6**	RW	0	Pin 6 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIODINTCFG.P6** 1b: Trigger on both rising and falling edge
5	**P5**	RW	0	Pin 5 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIODINTCFG.P5** 1b: Trigger on both rising and falling edge
4	**P4**	RW	0	Pin 4 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIODINTCFG.P4** 1b: Trigger on both rising and falling edge
3	**P3**	RW	0	Pin 3 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIODINTCFG.P3** 1b: Trigger on both rising and falling edge
2	**P2**	RW	0	Pin 2 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIODINTCFG.P2** 1b: Trigger on both rising and falling edge
1	**P1**	RW	0	Pin 1 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIODINTCFG.P1** 1b: Trigger on both rising and falling edge
0	**P0**	RW	0	Pin 0 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIODINTCFG.P0** 1b: Trigger on both rising and falling edge

13.7.11 GPIODCLKSYNC[10]

Register 13-11 GPIODCLKSYNC (GPIOD Clock Synchronization Enable, 400D 202Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
6	**P6**	RW	1	Pin 6 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
5	**P5**	RW	1	Pin 5 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
4	**P4**	RW	1	Pin 4 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
3	**P3**	RW	1	Pin 3 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
2	**P2**	RW	1	Pin 2 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
1	**P1**	RW	1	Pin 1 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
0	**P0**	RW	1	Pin 0 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)

[10] In VER1 of the PAC55XX MCU, the default value for the clock synchronization is disabled (0b).

13.7.12 GPIODDOSET

Register 13-12 GPIODDOSET (GPIOD Data Output Set, 400D 2030h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P7** to 1b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P6** to 1b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P5** to 1b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P4** to 1b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P3** to 1b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P2** to 1b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P1** to 1b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P0** to 1b

13.7.13 GPIODDOCLEAR

Register 13-13 GPIODDOCLEAR (GPIOD Data Output Clear, 400D 2034h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Clear: 0b: No effect 1b: Set **GPIODOUT.P7** to 0b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P6** to 0b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P5** to 0b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P4** to 0b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P3** to 0b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P2** to 0b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P1** to 0b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIODOUT.P0** to 0b

14 GPIOE

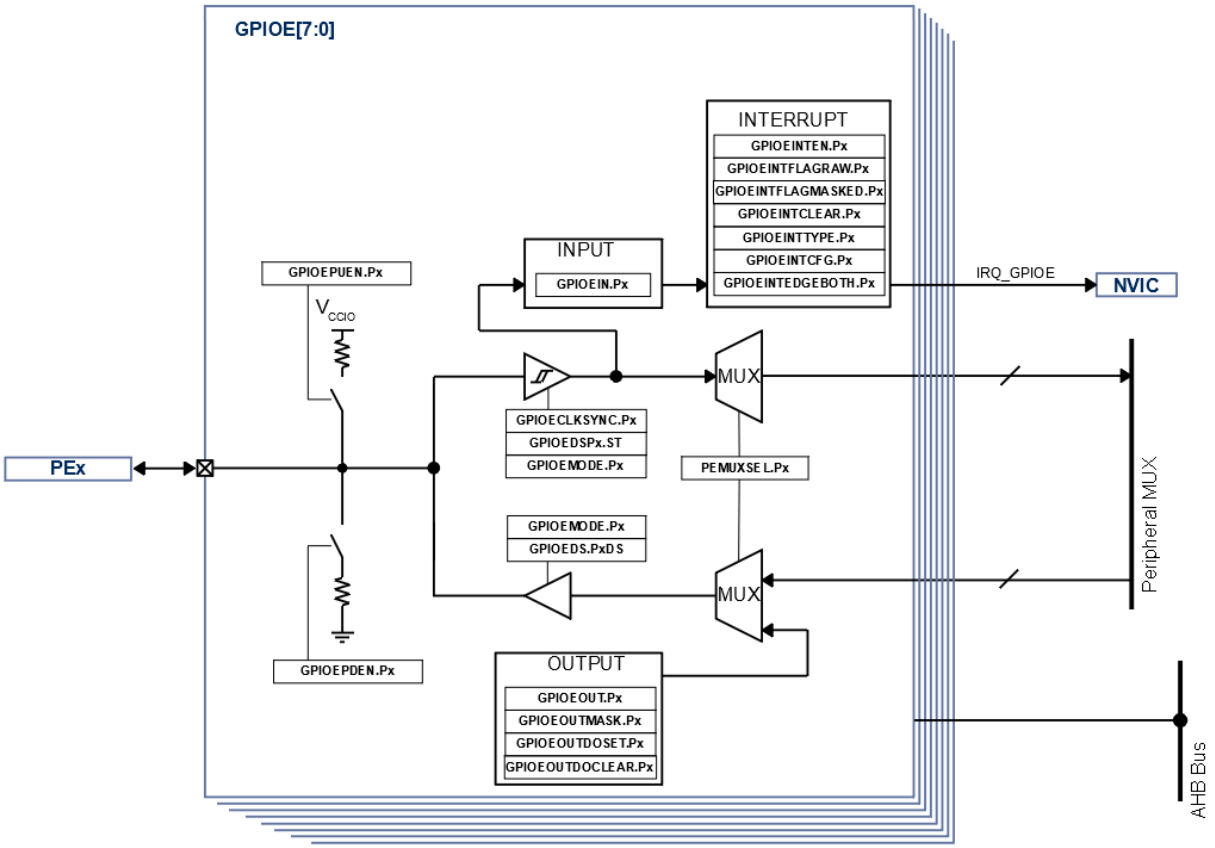
14.1 Overview

The GPIOE port contains 8 GPIO pin, GPIOE0 to GPIOE7.

Each GPIO pin may be used for Digital Input or Digital Output. Each GPIO is capable of generating 3.3V output and is 3.3V input tolerant and has a set of features described below.

14.2 GPIOE Block Diagram

Figure 14-1 GPIOE Block Diagram



14.3 Features

- 3.3V output, 3.3V input tolerant
- Configurable Pin Modes:
  - High-Impedance Digital Input
  - Push-Pull Digital Output
  - Open-Drain Digital Output
- Configurable Drive Strength up to 25mA
- Configurable Input Schmitt Trigger
- Configurable Weak Pull-up or Pull-down
- Edge or Level Sensitive Interrupts
  - Rising, Falling or both edges
- Clock Synchronization

14.4 Functional Description

14.4.1 IO Mode

Each GPIO pin supports up to 4 different modes and can selected from the **GPIOEMODE** register as shown below.

Table 14-1 GPIOE Mode Configuration

GPIOEMODE VALUE	DESCRIPTION
00b	Reserved
01b	Push-Pull Digital Output
10b	Open-Drain Digital Output
11b	High-Impedance Digital Input

The reset value for the GPIO pin mode is High-Impedance Digital Input.

14.4.2 Digital Output

Each GPIO pin may be configured for digital output by setting the **GPIOEMODE.Px** to either 01b (Push-pull output) or to 10b (Open-drain output).

When configured for Push-pull output, the GPIO is driven to the high state using the VCCIO 3.3V power supply. When configured for Open-drain output, the GPIO is pulled up externally.

When **GPIOEOUTMASK.Px** is set to 0b, then the state of the GPIO pin will be set from **GPIOEOUT.Px** (0b: logic low, 1b: logic high). If the **GPIOEOUTMASK.Px** is set to 1b, then any changes to **GPIOEOUT.Px** have no effect.

In either of these two output modes, the user may read the digital state of the input by reading the **GPIOEIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports both digital output modes.

14.4.3 Digital Input

If **GPIOEMODE.Px** is set to 11b, then the GPIO pin is configured for high-impedance input. For application safety, this is the reset value for the GPIO pin mode.

In this mode, the user may read the state of the digital input by reading the **GPIOEIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports this mode.

14.4.4 GPIOE Interrupts

Each GPIO pin supports configurable interrupts. In order for interrupts to work, the GPIO must be set for high-impedance input mode (**GPIOEMODE.Px** = 11b).

The user may configure either edge or level sensitive interrupts for each GPIO pin. If the user configures edge sensitive interrupts, they may select rising, falling or both edges for edge interrupts.

See the table below for how to configure the GPIO for the various types of interrupts.

Table 14-2 GPIOE Interrupt Configuration

GPIOEINTTYPE	GPIOEINTCFG	GPIOEEDGEBOTH	INTERRUPT MODE
0b	0b	0b	Falling Edge
	1b	0b	Rising Edge
	X	1b	Both Rising and Falling Edge
1b	0b	X	Logic Low
	1b	X	Logic High

Interrupts may be enabled or disabled using the **GPIOEINTEN** register. To enable interrupts on a pin, set the **GPIOEINTEN.Px** to 1b. To disable interrupts on a pin, set the **GPIOEINTEN.Px** to 0b.

If an edge or level interrupt is detected, the **GPIOEINTFLAG.Px** will be set to 1b for that GPIO pin. If **GPIOEINTFLAG.Px** is set to 1b and **GPIOEINTEN.Px** is set to 1b, then the IRQ_GPIOE signal to the NVIC is asserted.

To clear the **GPIOEINTFLAG.Px** interrupt flag, write **GPIOEINTCLEAR.Px** to a 1b.

14.4.5 GPIOE Clock Synchronization

Each GPIOE pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOEMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOE pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOECLKSYNC.Px** = 0b). To enable GPIOE Clock Synchronization set **GPIOECLKSYNC.Px** = 1b.

14.4.6 GPIOE Drive Strength

Each GPIO pin has configurable drive strength when the ports are configured as a digital output.

See the table below for the settings for GPIOE Drive strength.

Table 14-3 GPIOE Drive Strength

PEDS.PxDS	DRIVE STRENGTH
000b	6mA
001b	8mA
010b	11mA
011b	14mA
100b	17mA
101b	20mA
110b	22mA
111b	25mA

The drive strength shown in the table above represents the minimum drive strength for the GPIO pin. For the details on the electrical characteristics of the drive strength, set the device data sheet.

Note that the register configuration for the GPIOE Drive Strength is in the System and Clock Control (SCC).

14.4.7 GPIOE Schmitt Trigger

Each GPIO has a Schmitt Trigger that can be enabled when the GPIO is configured as a high-impedance digital input (**GPIOEMODE.Px** = 11b).

The Schmitt Trigger is disabled by default (**PEDS.PxST** = 0b). To enable the input Schmitt Triger, set **PEDS.PxST** = 1b.

Note that the register configuration for the GPIO Schmitt Trigger is in the System and GPIO and Digital Peripheral MUX (DPM).

14.4.8 GPIOE Weak Pull-up and Pull-down

Each GPIO pin has a configurable weak 60k pull-up to VCCIO or pull-down to ground.

To enable the weak pull-up to VCCIO, set **GPIOEPU.Px** to 1b. To disable the pull-up, set **GPIOEPU.Px** to 0b.

To enable the weak pull-down to ground, set the **GPIOEPD.Px** to 1b. To disable the pull-down, set **GPIOEPD.Px** to 0b.

WARNING: Do not configure the weak pull-up and pull-down to be active at the same time. Doing so may cause device damage.

14.5 Peripheral IO Mapping

The GPIOE peripheral is connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 14-4 GPIOE Peripheral IO Mapping

GPIOE SIGNAL	IO PIN
GPIOE0	PE0
GPIOE1	PE1
GPIOE2	PE2
GPIOE3	PE3
GPIOE4	PE4
GPIOE5	PE5
GPIOE6	PE6
GPIOE7	PE7

14.6 Register Summary

Table 14-5 GPIOE Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOEMODE	400D 2400h	GPIOE Pin Mode Select	RW	0000 FFFFh
GPIOEOUTMASK	400D 2404h	GPIOE Data Output Write Mask	RW	0000 0000h
GPIOEOUT	400D 2408h	GPIOE Data Output Value	RW	0000 00FFh
GPIOEIN	400D 240Ch	GPIOE Data Input Value	RO	--
GPIOEINTEN	400D 2410h	GPIOE Interrupt Enable	RW	0000 0000h
GPIOEINTFLAG	400D 2414h	GPIOE Interrupt Flag	RO	0000 0000h
GPIOEINTCLEAR	400D 241Ch	GPIOE Interrupt Clear	WO	0000 0000h
GPIOEINTTYPE	400D 2420h	GPIOE Interrupt Type	RW	0000 0000h
GPIOEINTCFG	400D 2424h	GPIOE Interrupt Configuration	RW	0000 0000h
GPIOEINTEDGEBOTH	400D 2428h	GPIOE Interrupt Edge Both	RW	0000 0000h
GPIOECLKSYNC	400D 242Ch	GPIOE Clock Synchronization	RW	--
GPIOEDOSET	400D 2430h	GPIOE Data Output Set	WO	--
GPIOEDOCLEAR	400D 2434h	GPIOE Data Output Clear	WO	0000 FFFFh

14.7 Register Detail

14.7.1 GPIOEMODE

Register 14-1 GPIOEMODE (GPIOE Mode Configuration, 400D 2400h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:14	**P7**	RW	11b	Pin 7 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
13:12	**P6**	RW	11b	Pin 6 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
11:10	**P5**	RW	11b	Pin 5 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
9:8	**P4**	RW	11b	Pin 4 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
7:6	**P3**	RW	11b	Pin 3 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
5:4	**P2**	RW	11b	Pin 2 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
3:2	**P1**	RW	11b	Pin 1 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
1:0	**P0**	RW	11b	Pin 0 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input

14.7.2 GPIOEOUTMASK

Register 14-2 GPIOEOUTMASK (GPIOE Output Mask, 400D 2404h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Output Data Mask: 0b: Not masked 1b: Masked
6	**P6**	RW	0	Pin 6 Output Data Mask: 0b: Not masked 1b: Masked
5	**P5**	RW	0	Pin 5 Output Data Mask: 0b: Not masked 1b: Masked
4	**P4**	RW	0	Pin 4 Output Data Mask: 0b: Not masked 1b: Masked
3	**P3**	RW	0	Pin 3 Output Data Mask: 0b: Not masked 1b: Masked
2	**P2**	RW	0	Pin 2 Output Data Mask: 0b: Not masked 1b: Masked
1	**P1**	RW	0	Pin 1 Output Data Mask: 0b: Not masked 1b: Masked
0	**P0**	RW	0	Pin 0 Output Data Mask: 0b: Not masked 1b: Masked

14.7.3 GPIOEOUT

Register 14-3 GPIOEOUT (GPIOE Output Data Value, 400E 2408h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Output Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RW	1	Pin 6 Output Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RW	1	Pin 5 Output Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RW	1	Pin 4 Output Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RW	1	Pin 3 Output Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RW	1	Pin 2 Output Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RW	1	Pin 1 Output Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RW	1	Pin 0 Output Data Value: 0b: Logic low 1b: Logic high

14.7.4 GPIOEIN

Register 14-4 GPIOEIN (GPIOE Input Data Value, 400D 240Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	-	Pin 7 Input Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RO	-	Pin 6 Input Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RO	-	Pin 5 Input Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RO	-	Pin 4 Input Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RO	-	Pin 3 Input Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RO	-	Pin 2 Input Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RO	-	Pin 1 Input Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RO	-	Pin 0 Input Data Value: 0b: Logic low 1b: Logic high

14.7.5 GPIOEINTEN

Register 14-5 GPIOEINTEN (GPIOE Interrupt Enable, 400D 2410h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Enable: 0b: Disabled 1b: Enabled
6	**P6**	RW	0	Pin 6 Interrupt Enable: 0b: Disabled 1b: Enabled
5	**P5**	RW	0	Pin 5 Interrupt Enable: 0b: Disabled 1b: Enabled
4	**P4**	RW	0	Pin 4 Interrupt Enable: 0b: Disabled 1b: Enabled
3	**P3**	RW	0	Pin 3 Interrupt Enable: 0b: Disabled 1b: Enabled
2	**P2**	RW	0	Pin 2 Interrupt Enable: 0b: Disabled 1b: Enabled
1	**P1**	RW	0	Pin 1 Interrupt Enable: 0b: Disabled 1b: Enabled
0	**P0**	RW	0	Pin 0 Interrupt Enable: 0b: Disabled 1b: Enabled

14.7.6 GPIOEINTFLAG

Register 14-6 GPIOEINTFLAG (GPIOE Interrupt Flag, 400D 2414h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	0	Pin 7 Interrupt Flag: 0b: No flag 1b: Flag
6	**P6**	RO	0	Pin 6 Interrupt Flag: 0b: No flag 1b: Flag
5	**P5**	RO	0	Pin 5 Interrupt Flag: 0b: No flag 1b: Flag
4	**P4**	RO	0	Pin 4 Interrupt Flag: 0b: No flag 1b: Flag
3	**P3**	RO	0	Pin 3 Interrupt Flag: 0b: No flag 1b: Flag
2	**P2**	RO	0	Pin 2 Interrupt Flag: 0b: No flag 1b: Flag
1	**P1**	RO	0	Pin 1 Interrupt Flag: 0b: No flag 1b: Flag
0	**P0**	RO	0	Pin 0 Interrupt Flag: 0b: No flag 1b: Flag

14.7.7 GPIOEINTCLEAR

Register 14-7 GPIOEINTCLEAR (GPIOE Interrupt Clear, 400D 241Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Interrupt Clear: 0b: No effect 1b: Clear **GPIOEINTFLAG.P7**
6	**P6**	WO	-	Pin 6 Interrupt Clear: 0b: No effect 1b: Clear **GPIOEINTFLAG.P6**
5	**P5**	WO	-	Pin 5 Interrupt Clear: 0b: No effect 1b: Clear **GPIOEINTFLAG.P5**
4	**P4**	WO	-	Pin 4 Interrupt Clear: 0b: No effect 1b: Clear **GPIOEINTFLAG.P4**
3	**P3**	WO	-	Pin 3 Interrupt Clear: 0b: No effect 1b: Clear **GPIOEINTFLAG.P3**
2	**P2**	WO	-	Pin 2 Interrupt Clear: 0b: No effect 1b: Clear **GPIOEINTFLAG.P2**
1	**P1**	WO	-	Pin 1 Interrupt Clear: 0b: No effect 1b: Clear **GPIOEINTFLAG.P1**
0	**P0**	WO	-	Pin 0 Interrupt Clear: 0b: No effect 1b: Clear **GPIOEINTFLAG.P0**

14.7.8 GPIOEINTTYPE

Register 14-8 GPIOEINTTYPE (GPIOE Interrupt Type Configuration, 400D 2420h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Type: 0b: Edge triggered 1b: Level triggered
6	**P6**	RW	0	Pin 6 Interrupt Type: 0b: Edge triggered 1b: Level triggered
5	**P5**	RW	0	Pin 5 Interrupt Type: 0b: Edge triggered 1b: Level triggered
4	**P4**	RW	0	Pin 4 Interrupt Type: 0b: Edge triggered 1b: Level triggered
3	**P3**	RW	0	Pin 3 Interrupt Type: 0b: Edge triggered 1b: Level triggered
2	**P2**	RW	0	Pin 2 Interrupt Type: 0b: Edge triggered 1b: Level triggered
1	**P1**	RW	0	Pin 1 Interrupt Type: 0b: Edge triggered 1b: Level triggered
0	**P0**	RW	0	Pin 0 Interrupt Type: 0b: Edge triggered 1b: Level triggered

14.7.9 GPIOEINTCFG

Register 14-9 GPIOEINTCFG (GPIOE Interrupt Configuration, 400D 2424h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	If **GPIOEINTTYPE.Px** = 0b (edge), Pin 7 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOEINTTYPE.Px** = 1b (level), Pin 7 Interrupt Level: 0b: Logic low 1b: Logic high
6	**P6**	RW	0	If **GPIOEINTTYPE.Px** = 0b (edge), Pin 6 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOEINTTYPE.Px** = 1b (level), Pin 6 Interrupt Level: 0b: Logic low 1b: Logic high
5	**P5**	RW	0	If **GPIOEINTTYPE.Px** = 0b (edge), Pin 5 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOEINTTYPE.Px** = 1b (level), Pin 5 Interrupt Level: 0b: Logic low 1b: Logic high
4	**P4**	RW	0	If **GPIOEINTTYPE.Px** = 0b (edge), Pin 4 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOEINTTYPE.Px** = 1b (level), Pin 4 Interrupt Level: 0b: Logic low 1b: Logic high
3	**P3**	RW	0	If **GPIOEINTTYPE.Px** = 0b (edge), Pin 3 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOEINTTYPE.Px** = 1b (level), Pin 3 Interrupt Level: 0b: Logic low 1b: Logic high
2	**P2**	RW	0	If **GPIOEINTTYPE.Px** = 0b (edge), Pin 2 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOEINTTYPE.Px** = 1b (level), Pin 2 Interrupt Level: 0b: Logic low 1b: Logic high
1	**P1**	RW	0	If **GPIOEINTTYPE.Px** = 0b (edge), Pin 1 Interrupt Edge: 0b: Trigger on falling edge

				1b: Trigger on rising edge If **GPIOEINTTYPE.Px** = 1b (level), Pin 1 Interrupt Level: 0b: Logic low 1b: Logic high
0	**P0**	RW	0	If **GPIOEINTTYPE.Px** = 0b (edge), Pin 0 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOEINTTYPE.Px** = 1b (level), Pin 0 Interrupt Level: 0b: Logic low 1b: Logic high

14.7.10 GPIOEINTEDGEBOTH

Register 14-10 GPIOEINTEDGEBOTH (GPIOE Interrupt Both Edge Configuration, 400D 2428h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOEINTCFG.P7** 1b: Trigger on both rising and falling edge
6	**P6**	RW	0	Pin 6 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOEINTCFG.P6** 1b: Trigger on both rising and falling edge
5	**P5**	RW	0	Pin 5 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOEINTCFG.P5** 1b: Trigger on both rising and falling edge
4	**P4**	RW	0	Pin 4 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOEINTCFG.P4** 1b: Trigger on both rising and falling edge
3	**P3**	RW	0	Pin 3 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOEINTCFG.P3** 1b: Trigger on both rising and falling edge
2	**P2**	RW	0	Pin 2 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOEINTCFG.P2** 1b: Trigger on both rising and falling edge
1	**P1**	RW	0	Pin 1 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOEINTCFG.P1** 1b: Trigger on both rising and falling edge
0	**P0**	RW	0	Pin 0 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOEINTCFGE.P0** 1b: Trigger on both rising and falling edge

14.7.11 GPIOECLKSYNC[11]

Register 14-11 GPIOECLKSYNC (GPIOE Clock Synchronization Enable, 400D 242Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
6	**P6**	RW	1	Pin 6 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
5	**P5**	RW	1	Pin 5 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
4	**P4**	RW	1	Pin 4 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
3	**P3**	RW	1	Pin 3 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
2	**P2**	RW	1	Pin 2 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
1	**P1**	RW	1	Pin 1 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
0	**P0**	RW	1	Pin 0 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)

[11] In VER1 of the PAC55XX MCU, the default value for the clock synchronization is disabled (0b).

14.7.12 GPIOEDOSET

Register 14-12 GPIOEDOSET (GPIOE Data Output Set, 400D 2430h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P7** to 1b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P6** to 1b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P5** to 1b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P4** to 1b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P3** to 1b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P2** to 1b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P1** to 1b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P0** to 1b

14.7.13 GPIOEDOCLEAR

Register 14-13 GPIOEDOCLEAR (GPIOE Data Output Clear, 400D 2434h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Clear: 0b: No effect 1b: Set **GPIOEOUT.P7** to 0b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P6** to 0b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P5** to 0b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P4** to 0b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P3** to 0b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P2** to 0b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P1** to 0b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P0** to 0b

15 GPIOF

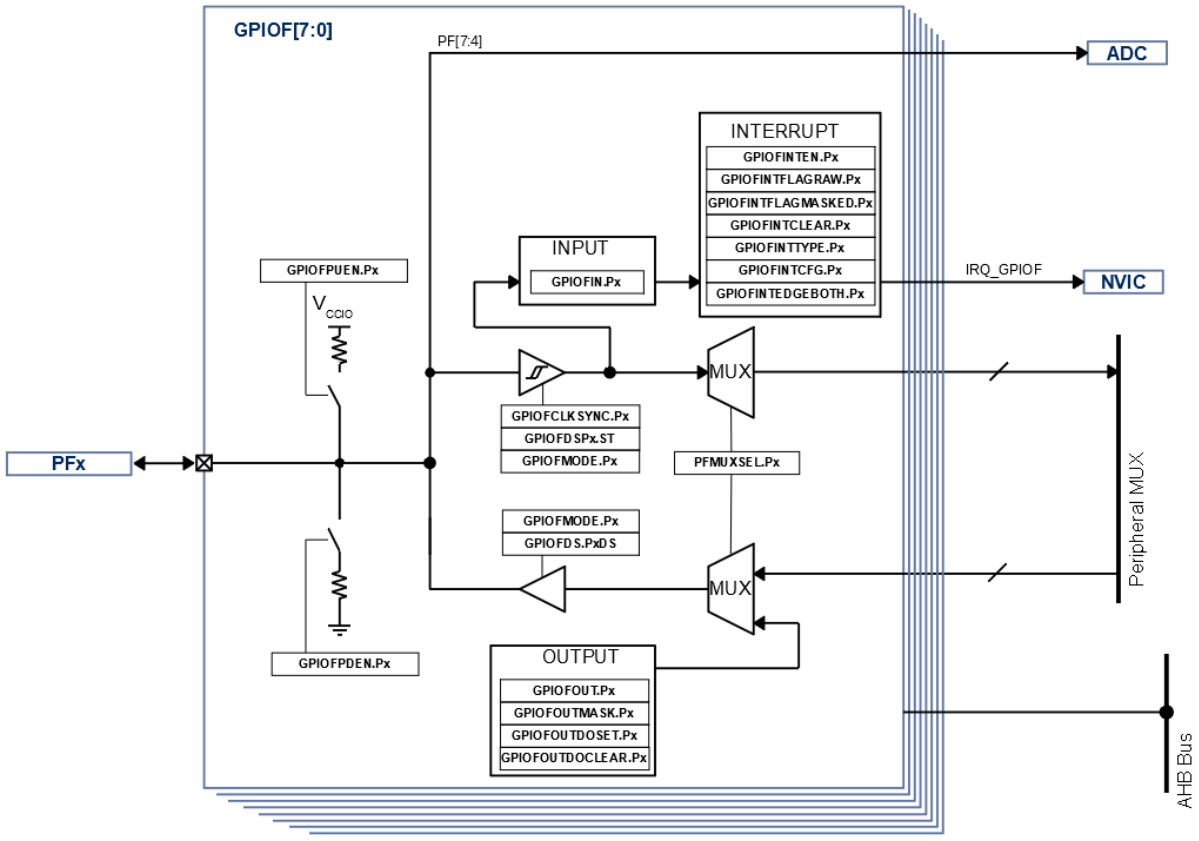
15.1 Overview

The GPIOF port contains 8 GPIO pin, GPIOF0 to GPIOF7.

Each GPIO pin may be used for Digital Input, Digital Output and some may be used for Analog Input.[12] Each GPIO is capable of generating 3.3V output and is 3.3V input tolerant and has a set of features described below.

15.2 GPIOF Block Diagram

Figure 15-1 GPIOF Block Diagram



[12] Consult the device data sheet for information on which GPIO pins are present on any given PAC55XX device.

15.3 Features

- 3.3V output, 3.3V input tolerant
- Configurable Pin Modes:
  - High-Impedance Digital Input
  - Push-Pull Digital Output
  - Open-Drain Digital Output
  - Analog Input
- Configurable Drive Strength up to 25mA
- Configurable Input Schmitt Trigger
- Configurable Weak Pull-up or Pull-down
- Edge or Level Sensitive Interrupts
  - Rising, Falling or both edges
- Clock Synchronization

15.4 Functional Description

15.4.1 IO Mode

Each GPIO pin supports up to 4 different modes and can selected from the **GPIOFMODE** register as shown below.

Table 15-1 GPIOF Mode Configuration

GPIOFMODE VALUE	DESCRIPTION
00b	Analog Input
01b	Push-Pull Digital Output
10b	Open-Drain Digital Output
11b	High-Impedance Digital Input

The reset value for the GPIO pin mode is High-Impedance Digital Input.

15.4.2 Digital Output

Each GPIO pin may be configured for digital output by setting the **GPIOFMODE.Px** to either 01b (Push-pull output) or to 10b (Open-drain output).

When configured for Push-pull output, the GPIO is driven to the high state using the VCCIO 3.3V power supply. When configured for Open-drain output, the GPIO is pulled up externally.

When **GPIOFOUTMASK.Px** is set to 0b, then the state of the GPIO pin will be set from **GPIOFOUT.Px** (0b: logic low, 1b: logic high). If the **GPIOFOUTMASK.Px** is set to 1b, then any changes to **GPIOFOUT.Px** have no effect.

In either of these two output modes, the user may read the digital state of the input by reading the **GPIOFIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports both digital output modes.

15.4.3 Digital Input

If **GPIOFMODE.Px** is set to 11b, then the GPIO pin is configured for high-impedance input. For application safety, this is the reset value for the GPIO pin mode.

In this mode, the user may read the state of the digital input by reading the **GPIOFIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports this mode.

15.4.4 Analog Input

Some GPIO pins may be configured for Analog Input.

To configure a GPIO pin for Analog Input, the user may set **GPIOFMODE.Px** to 00b. In this mode, the user MUST be sure to set the output mode to high-impedance input by setting **GPIOFMODE.Px** to 11b or the Analog Input will not work properly.

To determine if your device supports Analog Input for each GPIO pin, consult the device data sheet.

15.4.5 GPIOF Interrupts

Each GPIO pin supports configurable interrupts. In order for interrupts to work, the GPIO must be set for high-impedance input mode (**GPIOFMODE.Px** = 11b).

The user may configure either edge or level sensitive interrupts for each GPIO pin. If the user configures edge sensitive interrupts, they may select rising, falling or both edges for edge interrupts.

See the table below for how to configure the GPIO for the various types of interrupts.

Table 15-2 GPIOF Interrupt Configuration

GPIOFINTTYPE	GPIOFINTCFG	GPIOFEDGEBOTH	INTERRUPT MODE
0b	0b	0b	Falling Edge
	1b	0b	Rising Edge
	X	1b	Both Rising and Falling Edge
1b	0b	X	Logic Low
	1b	X	Logic High

Interrupts may be enabled or disabled using the **GPIOFINTEN** register. To enable interrupts on a pin, set the **GPIOFINTEN.Px** to 1b. To disable interrupts on a pin, set the **GPIOFINTEN.Px** to 0b.

If an edge or level interrupt is detected, the **GPIOFINTFLAG.Px** will be set to 1b for that GPIO pin. If **GPIOFINTFLAG.Px** is set to 1b and **GPIOFINTEN.Px** is set to 1b, then the IRQ_GPIOF signal to the NVIC is asserted.

To clear the **GPIOFINTFLAG.Px** interrupt flag, write **GPIOFINTCLEAR.Px** to a 1b.

15.4.6 GPIOF Clock Synchronization

Each GPIOF pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOFMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOF pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOFCLKSYNC.Px** = 0b). To enable GPIOF Clock Synchronization set **GPIOFCLKSYNC.Px** = 1b.

15.4.7 GPIOF Drive Strength

Each GPIO pin has configurable drive strength when the ports are configured as a digital output.

See the table below for the settings for GPIOF Drive strength.

Table 15-3 GPIOF Drive Strength

PFDS.PxDS	DRIVE STRENGTH
000b	6mA
001b	8mA
010b	11mA
011b	14mA
100b	17mA
101b	20mA
110b	22mA
111b	25mA

The drive strength shown in the table above represents the minimum drive strength for the GPIO pin. For the details on the electrical characteristics of the drive strength, set the device data sheet.

Note that the register configuration for the GPIOF Drive Strength is in the System and Clock Control (SCC).

15.4.8 GPIOF Schmitt Trigger

Each GPIO has a Schmitt Trigger that can be enabled when the GPIO is configured as a high-impedance digital input (**GPIOFMODE.Px** = 11b).

The Schmitt Trigger is disabled by default (**PFDS.PxST** = 0b). To enable the input Schmitt Triger, set **PFDS.PxST** = 1b.

Note that the register configuration for the GPIO Schmitt Trigger is in the GPIO and Digital Peripheral MUX (DPM).

15.4.9 GPIOF Weak Pull-up and Pull-down

Each GPIO pin has a configurable weak 60k pull-up to VCCIO or pull-down to ground.

To enable the weak pull-up to VCCIO, set **GPIOFPU.Px** to 1b. To disable the pull-up, set **GPIOFPU.Px** to 0b.

To enable the weak pull-down to ground, set the **GPIOFPD.Px** to 1b. To disable the pull-down, set **GPIOFPD.Px** to 0b.

WARNING: Do not configure the weak pull-up and pull-down to be active at the same time. Doing so may cause device damage.

15.5 Peripheral IO Mapping

The GPIOF peripheral is connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 15-4 GPIOF Peripheral IO Mapping

GPIOF SIGNAL	IO PIN
GPIOF0	PF0
GPIOF1	PF1
GPIOF2	PF2
GPIOF3	PF3
GPIOF4	PF4
GPIOF5	PF5
GPIOF6	PF6
GPIOF7	PF7

15.6 Register Summary

Table 15-5 GPIOF Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOFMODE	400D 2800h	GPIOF Pin Mode Select	RW	0000 FFFFh
GPIOFOUTMASK	400D 2804h	GPIOF Data Output Write Mask	RW	0000 0000h
GPIOFOUT	400D 2808h	GPIOF Data Output Value	RW	0000 00FFh
GPIOFIN	400D 280Ch	GPIOF Data Input Value	RO	--
GPIOFINTEN	400D 2810h	GPIOF Interrupt Enable	RW	0000 0000h
GPIOFINTFLAG	400D 2814h	GPIOF Interrupt Flag	RO	0000 0000h
GPIOFINTCLEAR	400D 281Ch	GPIOF Interrupt Clear	WO	0000 0000h
GPIOFINTTYPE	400D 2820h	GPIOF Interrupt Type	RW	0000 0000h
GPIOFINTCFG	400D 2824h	GPIOF Interrupt Configuration	RW	0000 0000h
GPIOFINTEDGEBOTH	400D 2828h	GPIOF Interrupt Edge Both	RW	0000 0000h
GPIOFCLKSYNC	400D 282Ch	GPIOF Clock Synchronization	RW	--
GPIOFDOSET	400D 2830h	GPIOF Data Output Set	WO	--
GPIOFDOCLEAR	400D 2834h	GPIOF Data Output Clear	WO	0000 FFFFh

15.7 Register Detail

15.7.1 GPIOFMODE

Register 15-1 GPIOFMODE (GPIOF Mode Configuration, 400D 2800h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:14	**P7**	RW	11b	Pin 7 Mode: 00b: Analog Input (ADC7) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
13:12	**P6**	RW	11b	Pin 6 Mode: 00b: Analog Input (ADC6) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
11:10	**P5**	RW	11b	Pin 5 Mode: 00b: Analog Input (ADC5) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
9:8	**P4**	RW	11b	Pin 4 Mode: 00b: Analog Input (ADC4) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
7:6	**P3**	RW	11b	Pin 3 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
5:4	**P2**	RW	11b	Pin 2 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
3:2	**P1**	RW	11b	Pin 1 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
1:0	**P0**	RW	11b	Pin 0 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input

15.7.2 GPIOFOUTMASK

Register 15-2 GPIOFOUTMASK (GPIOF Output Mask, 400D 2804h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Output Data Mask: 0b: Not masked 1b: Masked
6	**P6**	RW	0	Pin 6 Output Data Mask: 0b: Not masked 1b: Masked
5	**P5**	RW	0	Pin 5 Output Data Mask: 0b: Not masked 1b: Masked
4	**P4**	RW	0	Pin 4 Output Data Mask: 0b: Not masked 1b: Masked
3	**P3**	RW	0	Pin 3 Output Data Mask: 0b: Not masked 1b: Masked
2	**P2**	RW	0	Pin 2 Output Data Mask: 0b: Not masked 1b: Masked
1	**P1**	RW	0	Pin 1 Output Data Mask: 0b: Not masked 1b: Masked
0	**P0**	RW	0	Pin 0 Output Data Mask: 0b: Not masked 1b: Masked

15.7.3 GPIOFOUT

Register 15-3 GPIOFOUT (GPIOF Output Data Value, 400E 2808h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Output Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RW	1	Pin 6 Output Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RW	1	Pin 5 Output Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RW	1	Pin 4 Output Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RW	1	Pin 3 Output Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RW	1	Pin 2 Output Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RW	1	Pin 1 Output Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RW	1	Pin 0 Output Data Value: 0b: Logic low 1b: Logic high

15.7.4 GPIOFIN

Register 15-4 GPIOFIN (GPIOF Input Data Value, 400D 280Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	-	Pin 7 Input Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RO	-	Pin 6 Input Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RO	-	Pin 5 Input Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RO	-	Pin 4 Input Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RO	-	Pin 3 Input Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RO	-	Pin 2 Input Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RO	-	Pin 1 Input Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RO	-	Pin 0 Input Data Value: 0b: Logic low 1b: Logic high

15.7.5 GPIOFINTEN

Register 15-5 GPIOFINTEN (GPIOF Interrupt Enable, 400D 2810h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Enable: 0b: Disabled 1b: Enabled
6	**P6**	RW	0	Pin 6 Interrupt Enable: 0b: Disabled 1b: Enabled
5	**P5**	RW	0	Pin 5 Interrupt Enable: 0b: Disabled 1b: Enabled
4	**P4**	RW	0	Pin 4 Interrupt Enable: 0b: Disabled 1b: Enabled
3	**P3**	RW	0	Pin 3 Interrupt Enable: 0b: Disabled 1b: Enabled
2	**P2**	RW	0	Pin 2 Interrupt Enable: 0b: Disabled 1b: Enabled
1	**P1**	RW	0	Pin 1 Interrupt Enable: 0b: Disabled 1b: Enabled
0	**P0**	RW	0	Pin 0 Interrupt Enable: 0b: Disabled 1b: Enabled

15.7.6 GPIOFINTFLAG

Register 15-6 GPIOFINTFLAG (GPIOF Interrupt Flag, 400D 2814h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	0	Pin 7 Interrupt Flag: 0b: No flag 1b: Flag
6	**P6**	RO	0	Pin 6 Interrupt Flag: 0b: No flag 1b: Flag
5	**P5**	RO	0	Pin 5 Interrupt Flag: 0b: No flag 1b: Flag
4	**P4**	RO	0	Pin 4 Interrupt Flag: 0b: No flag 1b: Flag
3	**P3**	RO	0	Pin 3 Interrupt Flag: 0b: No flag 1b: Flag
2	**P2**	RO	0	Pin 2 Interrupt Flag: 0b: No flag 1b: Flag
1	**P1**	RO	0	Pin 1 Interrupt Flag: 0b: No flag 1b: Flag
0	**P0**	RO	0	Pin 0 Interrupt Flag: 0b: No flag 1b: Flag

15.7.7 GPIOFINTCLEAR

Register 15-7 GPIOFINTCLEAR (GPIOF Interrupt Clear, 400D 281Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Interrupt Clear: 0b: No effect 1b: Clear **GPIOFINTFLAG.P7**
6	**P6**	WO	-	Pin 6 Interrupt Clear: 0b: No effect 1b: Clear **GPIOFINTFLAG.P6**
5	**P5**	WO	-	Pin 5 Interrupt Clear: 0b: No effect 1b: Clear **GPIOFINTFLAG.P5**
4	**P4**	WO	-	Pin 4 Interrupt Clear: 0b: No effect 1b: Clear **GPIOFINTFLAG.P4**
3	**P3**	WO	-	Pin 3 Interrupt Clear: 0b: No effect 1b: Clear **GPIOFINTFLAG.P3**
2	**P2**	WO	-	Pin 2 Interrupt Clear: 0b: No effect 1b: Clear **GPIOFINTFLAG.P2**
1	**P1**	WO	-	Pin 1 Interrupt Clear: 0b: No effect 1b: Clear **GPIOFINTFLAG.P1**
0	**P0**	WO	-	Pin 0 Interrupt Clear: 0b: No effect 1b: Clear **GPIOFINTFLAG.P0**

15.7.8 GPIOFINTTYPE

Register 15-8 GPIOFINTTYPE (GPIOF Interrupt Type Configuration, 400D 2820h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Type: 0b: Edge triggered 1b: Level triggered
6	**P6**	RW	0	Pin 6 Interrupt Type: 0b: Edge triggered 1b: Level triggered
5	**P5**	RW	0	Pin 5 Interrupt Type: 0b: Edge triggered 1b: Level triggered
4	**P4**	RW	0	Pin 4 Interrupt Type: 0b: Edge triggered 1b: Level triggered
3	**P3**	RW	0	Pin 3 Interrupt Type: 0b: Edge triggered 1b: Level triggered
2	**P2**	RW	0	Pin 2 Interrupt Type: 0b: Edge triggered 1b: Level triggered
1	**P1**	RW	0	Pin 1 Interrupt Type: 0b: Edge triggered 1b: Level triggered
0	**P0**	RW	0	Pin 0 Interrupt Type: 0b: Edge triggered 1b: Level triggered

15.7.9 GPIOFINTCFG

Register 15-9 GPIOFINTCFG (GPIOF Interrupt Configuration, 400D 2824h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	If **GPIOFINTTYPE.Px** = 0b (edge), Pin 7 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOFINTTYPE.Px** = 1b (level), Pin 7 Interrupt Level: 0b: Logic low 1b: Logic high
6	**P6**	RW	0	If **GPIOFINTTYPE.Px** = 0b (edge), Pin 6 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOFINTTYPE.Px** = 1b (level), Pin 6 Interrupt Level: 0b: Logic low 1b: Logic high
5	**P5**	RW	0	If **GPIOFINTTYPE.Px** = 0b (edge), Pin 5 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOFINTTYPE.Px** = 1b (level), Pin 5 Interrupt Level: 0b: Logic low 1b: Logic high
4	**P4**	RW	0	If **GPIOFINTTYPE.Px** = 0b (edge), Pin 4 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOFINTTYPE.Px** = 1b (level), Pin 4 Interrupt Level: 0b: Logic low 1b: Logic high
3	**P3**	RW	0	If **GPIOFINTTYPE.Px** = 0b (edge), Pin 3 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOFINTTYPE.Px** = 1b (level), Pin 3 Interrupt Level: 0b: Logic low 1b: Logic high
2	**P2**	RW	0	If **GPIOFINTTYPE.Px** = 0b (edge), Pin 2 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOFINTTYPE.Px** = 1b (level), Pin 2 Interrupt Level: 0b: Logic low 1b: Logic high
1	**P1**	RW	0	If **GPIOFINTTYPE.Px** = 0b (edge), Pin 1 Interrupt Edge: 0b: Trigger on falling edge

				1b: Trigger on rising edge If **GPIOFINTTYPE.Px** = 1b (level), Pin 1 Interrupt Level: 0b: Logic low 1b: Logic high
0	**P0**	RW	0	If **GPIOFINTTYPE.Px** = 0b (edge), Pin 0 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOFINTTYPE.Px** = 1b (level), Pin 0 Interrupt Level: 0b: Logic low 1b: Logic high

15.7.10 GPIOFINTEDGEBOTH

Register 15-10 GPIOFINTEDGEBOTH (GPIOF Interrupt Both Edge Configuration, 400D 2828h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOFINTCFG.P7** 1b: Trigger on both rising and falling edge
6	**P6**	RW	0	Pin 6 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOFINTCFG.P6** 1b: Trigger on both rising and falling edge
5	**P5**	RW	0	Pin 5 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOFINTCFG.P5** 1b: Trigger on both rising and falling edge
4	**P4**	RW	0	Pin 4 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOFINTCFG.P4** 1b: Trigger on both rising and falling edge
3	**P3**	RW	0	Pin 3 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOFINTCFG.P3** 1b: Trigger on both rising and falling edge
2	**P2**	RW	0	Pin 2 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOFINTCFG.P2** 1b: Trigger on both rising and falling edge
1	**P1**	RW	0	Pin 1 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOFINTCFG.P1** 1b: Trigger on both rising and falling edge
0	**P0**	RW	0	Pin 0 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOFINTCFG.P0** 1b: Trigger on both rising and falling edge

15.7.11 GPIOFCLKSYNC[13]

Register 15-11 GPIOFCLKSYNC (GPIOF Clock Synchronization Enable, 400D 282Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
6	**P6**	RW	1	Pin 6 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
5	**P5**	RW	1	Pin 5 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
4	**P4**	RW	1	Pin 4 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
3	**P3**	RW	1	Pin 3 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
2	**P2**	RW	1	Pin 2 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
1	**P1**	RW	1	Pin 1 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
0	**P0**	RW	1	Pin 0 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)

[13] In VER1 of the PAC55XX MCU, the default value for the clock synchronization is disabled (0b).

15.7.12 GPIOFDOSET

Register 15-12 GPIOFDOSET (GPIOF Data Output Set, 400D 2830h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P7** to 1b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P6** to 1b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P5** to 1b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P4** to 1b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P3** to 1b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P2** to 1b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P1** to 1b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P0** to 1b

15.7.13 GPIOFDOCLEAR

Register 15-13 GPIOFDOCLEAR (GPIOF Data Output Clear, 400D 2834h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Clear: 0b: No effect 1b: Set **GPIOFOUT.P7** to 0b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P6** to 0b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P5** to 0b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P4** to 0b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P3** to 0b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOEOUT.P2** to 0b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P1** to 0b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOFOUT.P0** to 0b

16 GPIOG

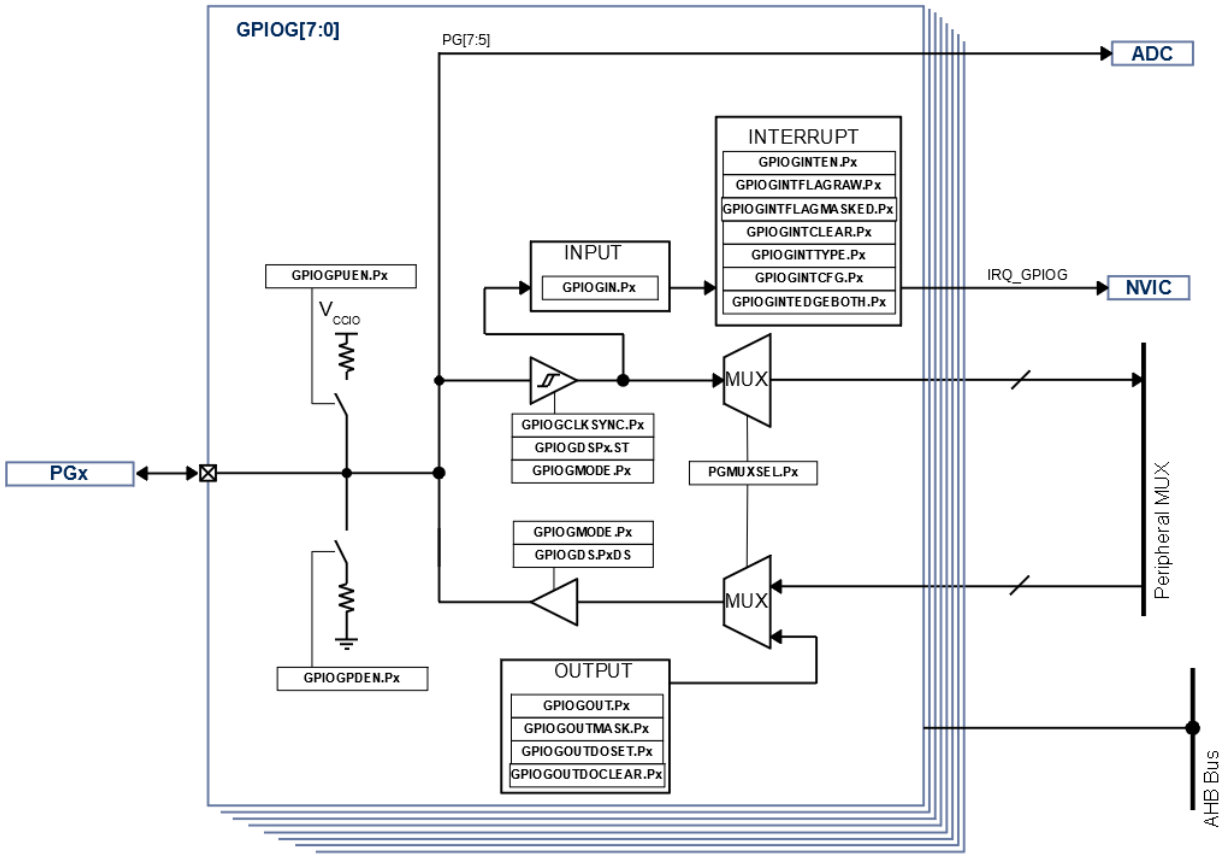
16.1 Overview

The GPIOG port contains 8 GPIO pin, GPIOG0 to GPIOG7.

Each GPIO pin may be used for Digital Input, Digital Output and some may be used for Analog Input.[14] Each GPIO is capable of generating 3.3V output and is 3.3V input tolerant and has a set of features described below.

16.2 GPIOG Block Diagram

Figure 16-1 GPIOG Block Diagram



[14] Consult the device data sheet for information on which GPIO pins are present on any given PAC55XX device.

16.3 Features

- 3.3V output, 3.3V input tolerant
- Configurable Pin Modes:
  - High-Impedance Digital Input
  - Push-Pull Digital Output
  - Open-Drain Digital Output
  - Analog Input
- Configurable Drive Strength up to 25mA
- Configurable Input Schmitt Trigger
- Configurable Weak Pull-up or Pull-down
- Edge or Level Sensitive Interrupts
  - Rising, Falling or both edges
- Clock Synchronization

16.4 Functional Description

16.4.1 IO Mode

Each GPIO pin supports up to 4 different modes and can selected from the **GPIOGMODE** register as shown below.

Table 16-1 GPIOG Mode Configuration

GPIOGMODE VALUE	DESCRIPTION
00b	Analog Input
01b	Push-Pull Digital Output
10b	Open-Drain Digital Output
11b	High-Impedance Digital Input

The reset value for the GPIO pin mode is High-Impedance Digital Input.

16.4.2 Digital Output

Each GPIO pin may be configured for digital output by setting the **GPIOGMODE.Px** to either 01b (Push-pull output) or to 10b (Open-drain output).

When configured for Push-pull output, the GPIO is driven to the high state using the VCCIO 3.3V power supply. When configured for Open-drain output, the GPIO is pulled up externally.

When **GPIOGOUTMASK.Px** is set to 0b, then the state of the GPIO pin will be set from **GPIOGOUT.Px** (0b: logic low, 1b: logic high). If the **GPIOGOUTMASK.Px** is set to 1b, then any changes to **GPIOGOUT.Px** have no effect.

In either of these two output modes, the user may read the digital state of the input by reading the **GPIOGIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports both digital output modes.

16.4.3 Digital Input

If **GPIOGMODE.Px** is set to 11b, then the GPIO pin is configured for high-impedance input. For application safety, this is the reset value for the GPIO pin mode.

In this mode, the user may read the state of the digital input by reading the **GPIOGIN.Px** value (0b: logic low, 1b: logic high).

Every GPIO pin supports this mode.

16.4.4 Analog Input

Some GPIO pins may be configured for Analog Input.

To configure a GPIO pin for Analog Input, the user may set **GPIOGMODE.Px** to 00b. In this mode, the user MUST be sure to set the output mode to high-impedance input by setting **GPIOGMODE.Px** to 11b or the Analog Input will not work properly.

To determine if your device supports Analog Input for each GPIO pin, consult the device data sheet.

16.4.5 GPIOG Interrupts

Each GPIO pin supports configurable interrupts. In order for interrupts to work, the GPIO must be set for high-impedance input mode (**GPIOGMODE.Px** = 11b).

The user may configure either edge or level sensitive interrupts for each GPIO pin. If the user configures edge sensitive interrupts, they may select rising, falling or both edges for edge interrupts.

See the table below for how to configure the GPIO for the various types of interrupts.

Table 16-2 GPIOG Interrupt Configuration

GPIOGINTTYPE	GPIOGINTCFG	GPIOGEDGEBOTH	INTERRUPT MODE
0b	0b	0b	Falling Edge
	1b	0b	Rising Edge
	X	1b	Both Rising and Falling Edge
1b	0b	X	Logic Low
	1b	X	Logic High

Interrupts may be enabled or disabled using the **GPIOGINTEN** register. To enable interrupts on a pin, set the **GPIOGINTEN.Px** to 1b. To disable interrupts on a pin, set the **GPIOGINTEN.Px** to 0b.

If an edge or level interrupt is detected, the **GPIOGINTFLAG.Px** will be set to 1b for that GPIO pin. If **GPIOGINTFLAG.Px** is set to 1b and **GPIOGINTEN.Px** is set to 1b, then the IRQ_GPIOG signal to the NVIC is asserted.

To clear the **GPIOGINTFLAG.Px** interrupt flag, write **GPIOGINTCLEAR.Px** to a 1b.

16.4.6 GPIOG Clock Synchronization

Each GPIOG pin has a circuit that synchronizes received data with the PCLK peripheral clock when the pin is configured as a digital input (**GPIOGMODE.Px** = 11b). This circuit is a 3-stage synchronizer that will take 3 PCLK cycles to synchronize the received data to PCLK.

NOTE: *This feature must be enabled when the GPIOG pin is configured for receive interrupts.*

The default value for this feature is disabled (**GPIOGCLKSYNC.Px** = 0b). To enable GPIOG Clock Synchronization set **GPIOGCLKSYNC.Px** = 1b.

16.4.7 GPIOG Drive Strength

Each GPIO pin has configurable drive strength when the ports are configured as a digital output.

See the table below for the settings for GPIOG Drive strength.

Table 16-3 GPIOG Drive Strength

PGDS.PxDS	DRIVE STRENGTH
000b	6mA
001b	8mA
010b	11mA
011b	14mA
100b	17mA
101b	20mA
110b	22mA
111b	25mA

The drive strength shown in the table above represents the minimum drive strength for the GPIO pin. For the details on the electrical characteristics of the drive strength, set the device data sheet.

Note that the register configuration for the GPIOG Drive Strength is in the System and Clock Control (SCC).

16.4.8 GPIOG Schmitt Trigger

Each GPIO has a Schmitt Trigger that can be enabled when the GPIO is configured as a high-impedance digital input (**GPIOGMODE.Px** = 11b).

The Schmitt Trigger is disabled by default (**PGDS.PxST** = 0b). To enable the input Schmitt Triger, set **PGDS.PxST** = 1b.

Note that the register configuration for the GPIO Schmitt Trigger is in the GPIO and Digital Peripheral MUX (DPM).

16.4.9 GPIOG Weak Pull-up and Pull-down

Each GPIO pin has a configurable weak 60k pull-up to VCCIO or pull-down to ground.

To enable the weak pull-up to VCCIO, set **GPIOGPU.Px** to 1b. To disable the pull-up, set **GPIOGPU.Px** to 0b.

To enable the weak pull-down to ground, set the **GPIOGPD.Px** to 1b. To disable the pull-down, set **GPIOGPD.Px** to 0b.

WARNING: Do not configure the weak pull-up and pull-down to be active at the same time. Doing so may cause device damage.

16.5 Peripheral IO Mapping

The GPIOG peripheral is connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 16-4 GPIOG Peripheral IO Mapping

GPIOG SIGNAL	IO PIN
GPIOG0	PG0
GPIOG1	PG1
GPIOG2	PG2
GPIOG3	PG3
GPIOG4	PG4
GPIOG5	PG5
GPIOG6	PG6
GPIOG7	PG7

16.6 Register Summary

Table 16-5 GPIOG Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPIOGMODE	400D 2C00h	GPIOG Pin Mode Select	RW	0000 FFFFh
GPIOGOUTMASK	400D 2C04h	GPIOG Data Output Write Mask	RW	0000 0000h
GPIOGOUT	400D 2C08h	GPIOG Data Output Value	RW	0000 00FFh
GPIOGIN	400D 2C0Ch	GPIOG Data Input Value	RO	--
GPIOGINTEN	400D 2C10h	GPIOG Interrupt Enable	RW	0000 0000h
GPIOGINTFLAG	400D 2C14h	GPIOG Interrupt Flag	RO	0000 0000h
GPIOGINTCLEAR	400D 2C1Ch	GPIOG Interrupt Clear	WO	--
GPIOGINTTYPE	400D 2C20h	GPIOG Interrupt Type	RW	0000 0000h
GPIOGINTVALUE	400D 2C24h	GPIOG Interrupt Value	RW	0000 0000h
GPIOGINTEDGE	400D 2C28h	GPIOG Interrupt Edge	RW	0000 0000h
GPIOGCLKSYNC	400D 2C2Ch	GPIOG Clock Synchronization	RW	0000 0000h
GPIOGDOSET	400D 2C30h	GPIOG Data Output Set	WO	--
GPIOGDOCLEAR	400D 2C34h	GPIOG Data Output Clear	WO	--

16.7 Register Detail

16.7.1 GPIOGMODE

Register 16-1 GPIOGMODE (GPIOG Mode Configuration, 400D 2C00h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:14	**P7**	RW	11b	Pin 7 Mode: 00b: Analog Input (ADC0) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
13:12	**P6**	RW	11b	Pin 6 Mode: 00b: Analog Input (ADC2) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
11:10	**P5**	RW	11b	Pin 5 Mode: 00b: Analog Input (ADC1) 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
9:8	**P4**	RW	11b	Pin 4 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
7:6	**P3**	RW	11b	Pin 3 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
5:4	**P2**	RW	11b	Pin 2 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
3:2	**P1**	RW	11b	Pin 1 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input
1:0	**P0**	RW	11b	Pin 0 Mode: 00b: Reserved 01b: Push-Pull Output 10b: Open-Drain Output 11b: High-Impedance Input

16.7.2 GPIOGOUTMASK

Register 16-2 GPIOGOUTMASK (GPIOG Output Mask, 400D 2C04h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Output Data Mask: 0b: Not masked 1b: Masked
6	**P6**	RW	0	Pin 6 Output Data Mask: 0b: Not masked 1b: Masked
5	**P5**	RW	0	Pin 5 Output Data Mask: 0b: Not masked 1b: Masked
4	**P4**	RW	0	Pin 4 Output Data Mask: 0b: Not masked 1b: Masked
3	**P3**	RW	0	Pin 3 Output Data Mask: 0b: Not masked 1b: Masked
2	**P2**	RW	0	Pin 2 Output Data Mask: 0b: Not masked 1b: Masked
1	**P1**	RW	0	Pin 1 Output Data Mask: 0b: Not masked 1b: Masked
0	**P0**	RW	0	Pin 0 Output Data Mask: 0b: Not masked 1b: Masked

16.7.3 GPIOGOUT

Register 16-3 GPIOGOUT (GPIOG Output Data Value, 400E 2C08h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Output Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RW	1	Pin 6 Output Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RW	1	Pin 5 Output Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RW	1	Pin 4 Output Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RW	1	Pin 3 Output Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RW	1	Pin 2 Output Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RW	1	Pin 1 Output Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RW	1	Pin 0 Output Data Value: 0b: Logic low 1b: Logic high

16.7.4 GPIOGIN

Register 16-4 GPIOFIN (GPIOG Input Data Value, 400D 2C0Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	-	Pin 7 Input Data Value: 0b: Logic low 1b: Logic high
6	**P6**	RO	-	Pin 6 Input Data Value: 0b: Logic low 1b: Logic high
5	**P5**	RO	-	Pin 5 Input Data Value: 0b: Logic low 1b: Logic high
4	**P4**	RO	-	Pin 4 Input Data Value: 0b: Logic low 1b: Logic high
3	**P3**	RO	-	Pin 3 Input Data Value: 0b: Logic low 1b: Logic high
2	**P2**	RO	-	Pin 2 Input Data Value: 0b: Logic low 1b: Logic high
1	**P1**	RO	-	Pin 1 Input Data Value: 0b: Logic low 1b: Logic high
0	**P0**	RO	-	Pin 0 Input Data Value: 0b: Logic low 1b: Logic high

16.7.5 GPIOGINTEN

Register 16-5 GPIOGINTEN (GPIOG Interrupt Enable, 400D 2C10h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Enable: 0b: Disabled 1b: Enabled
6	**P6**	RW	0	Pin 6 Interrupt Enable: 0b: Disabled 1b: Enabled
5	**P5**	RW	0	Pin 5 Interrupt Enable: 0b: Disabled 1b: Enabled
4	**P4**	RW	0	Pin 4 Interrupt Enable: 0b: Disabled 1b: Enabled
3	**P3**	RW	0	Pin 3 Interrupt Enable: 0b: Disabled 1b: Enabled
2	**P2**	RW	0	Pin 2 Interrupt Enable: 0b: Disabled 1b: Enabled
1	**P1**	RW	0	Pin 1 Interrupt Enable: 0b: Disabled 1b: Enabled
0	**P0**	RW	0	Pin 0 Interrupt Enable: 0b: Disabled 1b: Enabled

16.7.6 GPIOGINTFLAG

Register 16-6 GPIOGINTFLAG (GPIOG Interrupt Flag, 400D 2C14h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RO	0	Pin 7 Interrupt Flag: 0b: No flag 1b: Flag
6	**P6**	RO	0	Pin 6 Interrupt Flag: 0b: No flag 1b: Flag
5	**P5**	RO	0	Pin 5 Interrupt Flag: 0b: No flag 1b: Flag
4	**P4**	RO	0	Pin 4 Interrupt Flag: 0b: No flag 1b: Flag
3	**P3**	RO	0	Pin 3 Interrupt Flag: 0b: No flag 1b: Flag
2	**P2**	RO	0	Pin 2 Interrupt Flag: 0b: No flag 1b: Flag
1	**P1**	RO	0	Pin 1 Interrupt Flag: 0b: No flag 1b: Flag
0	**P0**	RO	0	Pin 0 Interrupt Flag: 0b: No flag 1b: Flag

16.7.7 GPIOGINTCLEAR

Register 16-7 GPIOGINTCLEAR (GPIOG Interrupt Clear, 400D 2C1Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Interrupt Clear: 0b: No effect 1b: Clear **GPIOGINTFLAG.P7**
6	**P6**	WO	-	Pin 6 Interrupt Clear: 0b: No effect 1b: Clear **GPIOGINTFLAG.P6**
5	**P5**	WO	-	Pin 5 Interrupt Clear: 0b: No effect 1b: Clear **GPIOGINTFLAG.P5**
4	**P4**	WO	-	Pin 4 Interrupt Clear: 0b: No effect 1b: Clear **GPIOGINTFLAG.P4**
3	**P3**	WO	-	Pin 3 Interrupt Clear: 0b: No effect 1b: Clear **GPIOGINTFLAG.P3**
2	**P2**	WO	-	Pin 2 Interrupt Clear: 0b: No effect 1b: Clear **GPIOGINTFLAG.P2**
1	**P1**	WO	-	Pin 1 Interrupt Clear: 0b: No effect 1b: Clear **GPIOGINTFLAG.P1**
0	**P0**	WO	-	Pin 0 Interrupt Clear: 0b: No effect 1b: Clear **GPIOGINTFLAG.P0**

16.7.8 GPIOGINTTYPE

Register 16-8 GPIOGINTTYPE (GPIOG Interrupt Type Configuration, 400D 2C20h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Type: 0b: Edge triggered 1b: Level triggered
6	**P6**	RW	0	Pin 6 Interrupt Type: 0b: Edge triggered 1b: Level triggered
5	**P5**	RW	0	Pin 5 Interrupt Type: 0b: Edge triggered 1b: Level triggered
4	**P4**	RW	0	Pin 4 Interrupt Type: 0b: Edge triggered 1b: Level triggered
3	**P3**	RW	0	Pin 3 Interrupt Type: 0b: Edge triggered 1b: Level triggered
2	**P2**	RW	0	Pin 2 Interrupt Type: 0b: Edge triggered 1b: Level triggered
1	**P1**	RW	0	Pin 1 Interrupt Type: 0b: Edge triggered 1b: Level triggered
0	**P0**	RW	0	Pin 0 Interrupt Type: 0b: Edge triggered 1b: Level triggered

16.7.9 GPIOGINTCFG

Register 16-9 GPIOGINTCFG (GPIOG Interrupt Configuration, 400D 2C24h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	If **GPIOGINTTYPE.Px** = 0b (edge), Pin 7 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOGINTTYPE.Px** = 1b (level), Pin 7 Interrupt Level: 0b: Logic low 1b: Logic high
6	**P6**	RW	0	If **GPIOGINTTYPE.Px** = 0b (edge), Pin 6 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOGINTTYPE.Px** = 1b (level), Pin 6 Interrupt Level: 0b: Logic low 1b: Logic high
5	**P5**	RW	0	If **GPIOGINTTYPE.Px** = 0b (edge), Pin 5 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOGINTTYPE.Px** = 1b (level), Pin 5 Interrupt Level: 0b: Logic low 1b: Logic high
4	**P4**	RW	0	If **GPIOGINTTYPE.Px** = 0b (edge), Pin 4 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOGINTTYPE.Px** = 1b (level), Pin 4 Interrupt Level: 0b: Logic low 1b: Logic high
3	**P3**	RW	0	If **GPIOGINTTYPE.Px** = 0b (edge), Pin 3 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOGINTTYPE.Px** = 1b (level), Pin 3 Interrupt Level: 0b: Logic low 1b: Logic high
2	**P2**	RW	0	If **GPIOGINTTYPE.Px** = 0b (edge), Pin 2 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOGINTTYPE.Px** = 1b (level), Pin 2 Interrupt Level: 0b: Logic low 1b: Logic high
1	**P1**	RW	0	If **GPIOGINTTYPE.Px** = 0b (edge), Pin 1 Interrupt Edge: 0b: Trigger on falling edge

				1b: Trigger on rising edge If **GPIOGINTTYPE.Px** = 1b (level), Pin 1 Interrupt Level: 0b: Logic low 1b: Logic high
0	**P0**	RW	0	If **GPIOGINTTYPE.Px** = 0b (edge), Pin 0 Interrupt Edge: 0b: Trigger on falling edge 1b: Trigger on rising edge If **GPIOGINTTYPE.Px** = 1b (level), Pin 0 Interrupt Level: 0b: Logic low 1b: Logic high

16.7.10 GPIOGINTEDGEBOTH

Register 16-10 GPIOGINTEDGEBOTH (GPIOG Interrupt Both Edge Configuration, 400D 2C28h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	0	Pin 7 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOGINTCFG.P7** 1b: Trigger on both rising and falling edge
6	**P6**	RW	0	Pin 6 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOGINTCFG.P6** 1b: Trigger on both rising and falling edge
5	**P5**	RW	0	Pin 5 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOGINTCFG.P5** 1b: Trigger on both rising and falling edge
4	**P4**	RW	0	Pin 4 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOGINTCFG.P4** 1b: Trigger on both rising and falling edge
3	**P3**	RW	0	Pin 3 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOGINTCFG.P3** 1b: Trigger on both rising and falling edge
2	**P2**	RW	0	Pin 2 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOGINTCFG.P2** 1b: Trigger on both rising and falling edge
1	**P1**	RW	0	Pin 1 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOGINTCFG.P1** 1b: Trigger on both rising and falling edge
0	**P0**	RW	0	Pin 0 Interrupt Both Edge Configuration: 0b: Trigger on rising or falling edge according to **GPIOGINTCFG.P0** 1b: Trigger on both rising and falling edge

16.7.11 GPIOGCLKSYNC[15]

Register 16-11 GPIOFCLKSYNC (GPIOG Clock Synchronization Enable, 400D 2C2Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	RW	1	Pin 7 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
6	**P6**	RW	1	Pin 6 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
5	**P5**	RW	1	Pin 5 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
4	**P4**	RW	1	Pin 4 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
3	**P3**	RW	1	Pin 3 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
2	**P2**	RW	1	Pin 2 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
1	**P1**	RW	1	Pin 1 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)
0	**P0**	RW	1	Pin 0 Clock Synchronization Enabled: 0b: Disabled 1b: Enabled (set when receive interrupts enabled)

[15] In VER1 of the PAC55XX MCU, the default value for the clock synchronization is disabled (0b).

16.7.12 GPIOGDOSET

Register 16-12 GPIOGDOSET (GPIOG Data Output Set, 400D 2C30h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P7** to 1b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P6** to 1b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P5** to 1b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P4** to 1b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P3** to 1b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P2** to 1b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P1** to 1b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P0** to 1b

16.7.13 GPIOGDOCLEAR

Register 16-13 GPIOGDOCLEAR (GPIOG Data Output Clear, 400D 2C34h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**P7**	WO	-	Pin 7 Data Output Clear: 0b: No effect 1b: Set **GPIOGOUT.P7** to 0b
6	**P6**	WO	-	Pin 6 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P6** to 0b
5	**P5**	WO	-	Pin 5 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P5** to 0b
4	**P4**	WO	-	Pin 4 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P4** to 0b
3	**P3**	WO	-	Pin 3 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P3** to 0b
2	**P2**	WO	-	Pin 2 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P2** to 0b
1	**P1**	WO	-	Pin 1 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P1** to 0b
0	**P0**	WO	-	Pin 0 Data Output Set: 0b: No effect 1b: Set **GPIOGOUT.P0** to 0b

17 GPTIMERA

17.1 Overview

General Purpose Timer A (GPTimer A) is a general-purpose system timer.

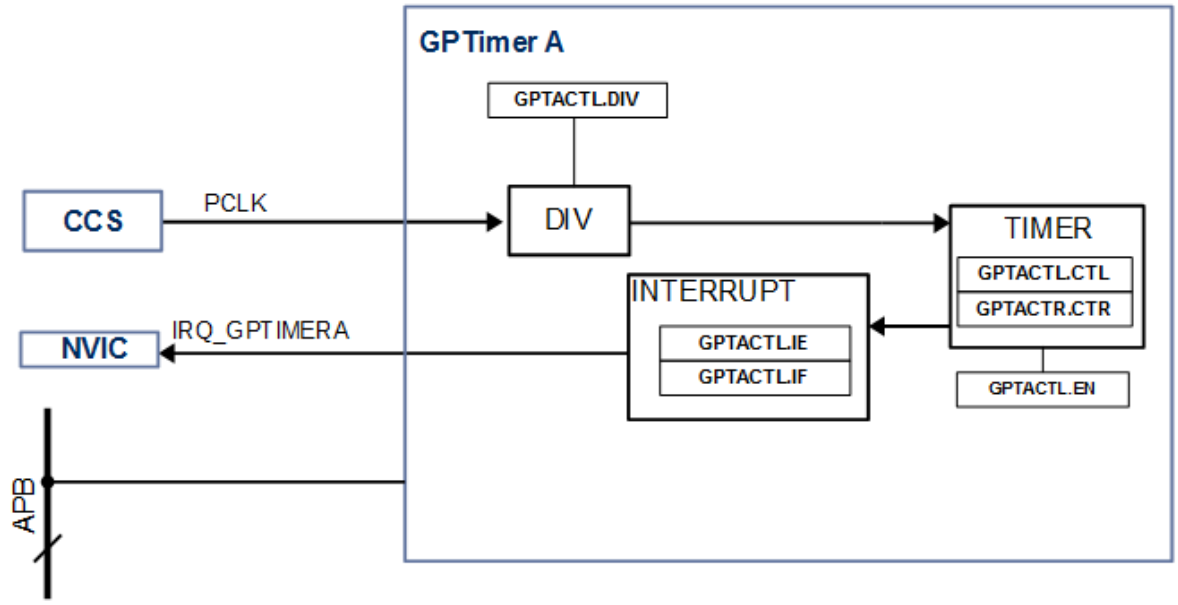
GPTimer A is an APB client.

17.2 Features

- 24-bit Count-down Timer
- Auto-reload
- Interrupt
- User-configurable count-down value
- Input clock Divider

17.3 System Block Diagram

Figure 17-1 GP Timer A System Block Diagram



17.4 Functional Description

GPTimer A is a general-purpose, 24-bit count-down system timer.

GPTimer A is an APB peripheral and the input clock is PCLK. This timer has a user-configurable clock divider that may be set as follows.

Table 17-1 GPTimer A Input Clock Divider

GPTACTL.DIV	GP Timer A Clock
0000b	PCLK /1
0001b	PCLK /2
0010b	PCLK /4
0011b	PCLK /8
…	…
1110b	PCLK /16384
1111b	PCLK /32768

GPTimer A may be enabled by setting **GPTACTL.EN** to 1b. When enabled, the timer will then copy the value of **GPTACTL.CDV** to **GPTACTR.CTR** and begin counting down.

When timer counts down and reaches 0, it will set **GPTACTL.IF** to a 1b. If the **GPTACTL.IE** is set to 1b, then the timer will assert the IRQ_GPTIMERA signal to the NVIC.

17.5 Register Summary

Table 17-2 GPTimer A Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPTACTL	400B 0000h	GPTimer A Control	RW	000F FF00h
GPTACTR	400B 0004h	GPTimer A Count-down Value	RO	00FF FFFFh

17.6 Register Detail

17.6.1 GPTACTL

Register 17-1 GPTACTL (GPTimer A Control, 400B 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**CDV**	RW	FFFh	Count-down value.
7	**Reserved**	RO	0	Reserved
6:3	**DIV**	RW	0	GPTimer A Clock Divider: 0000b: PCLK /1 0001b: PCLK /2 0010b: PCLK /4 0011b: PCLK /8 … 1110b: PCLK /16384 1111b: PCLK /32768
2	**IF**	W1C	0	Interrupt flag: 0b: no flag 1b: flag
1	**IE**	RW	0	Interrupt enable: 0b: not enabled 1b: enabled
0	**EN**	RW	0	Timer Enabled: 0b: disabled 1b: enabled

17.6.2 GPTACTR

Register 17-2 GPTACTR (GPTimer A Counter, 400B 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:24	**Reserved**	RO	0	Reserved
23:0	**CTR**	RW	FF FFFFh	Counter value.

18 GPTIMERB

18.1 Overview

General Purpose Timer B (GPTimer B) is a general-purpose system timer.

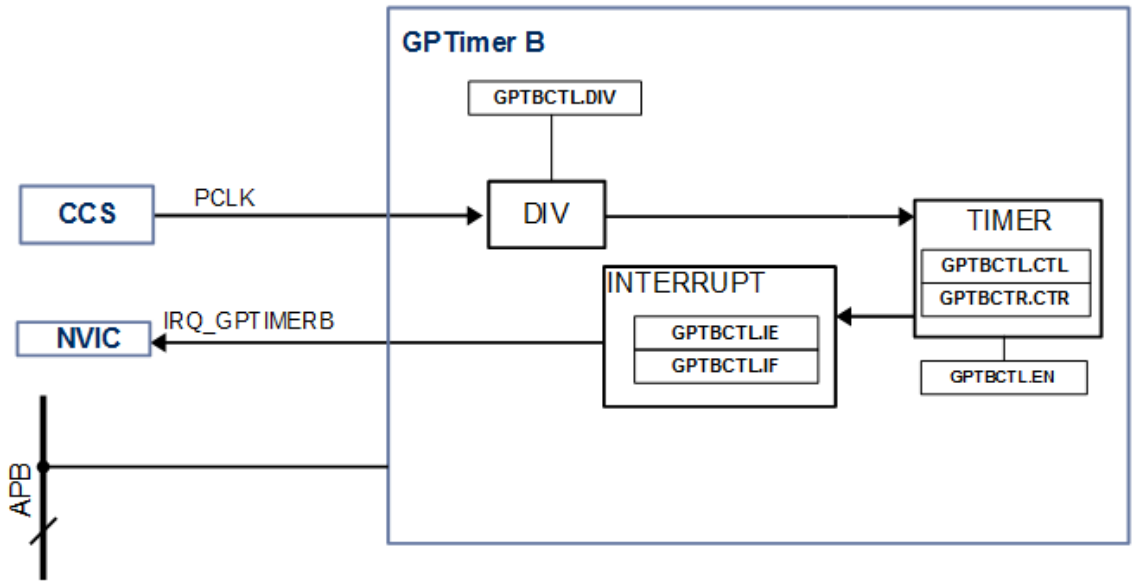
GPTimer B is an APB client.

18.2 Features

- 24-bit Count-down Timer
- Auto-reload
- Interrupt
- User-configurable count-down value
- Input clock Divider

18.3 System Block Diagram

Figure 18-1 GP Timer B System Block Diagram



18.4 Functional Description

GPTimer B is a general-purpose, 24-bit count-down system timer.

GPTimer B is an APB peripheral and the input clock is PCLK. This timer has a user-configurable clock divider that may be set as follows.

Table 18-1 GPTimer B Input Clock Divider

GPTACTL.DIV	GP Timer A Clock
0000b	PCLK /1
0001b	PCLK /2
0010b	PCLK /4
0011b	PCLK /8
…	…
1110b	PCLK /16384
1111b	PCLK /32768

GPTimer B may be enabled by setting **GPTBCTL.EN** to 1b. When enabled, the timer will then copy the value of **GPTBCTL.CDV** to **GPTBCTR.CTR** and begin counting down.

When timer counts down and reaches 0, it will set **GPTBCTL.IF** to a 1b. IF the **GPTBCTL.IE** is set to 1b, then the timer will assert the IRQ_GPTIMERB signal to the NVIC.

18.5 Register Summary

Table 18-2 GPTimer B Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
GPTBCTL	400C 0000h	GPTimer B Control	RW	000F FF00h
GPTBCTR	400C 0004h	GPTimer B Count-down Value	RO	00FF FFFFh

18.6 Register Detail

18.6.1 GPTBCTL

Register 18-1 GPTBCTL (GPTimer B Control, 400C 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**CDV**	RW	0	Count-down value.
7	**Reserved**	RO	0	Reserved
6:3	**DIV**	RW	0	GPTimer B Clock Divider: 0000b: PCLK /1 0001b: PCLK /2 0010b: PCLK /4 0011b: PCLK /8 … 1110b: PCLK /16384 1111b: PCLK /32768
2	**IF**	W1C	0	Interrupt flag: 0b: no flag 1b: flag
1	**IE**	RW	0	Interrupt enable: 0b: not enabled 1b: enabled
0	**EN**	RW	0	Timer Enabled: 0b: disabled 1b: enabled

18.6.2 GPTBCTR

Register 18-2 GPTBCTR (GPTimer B Counter, 400C 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:24	**Reserved**	RO	0	Reserved
23:0	**CTR**	RW	FF FFFFh	Counter value.

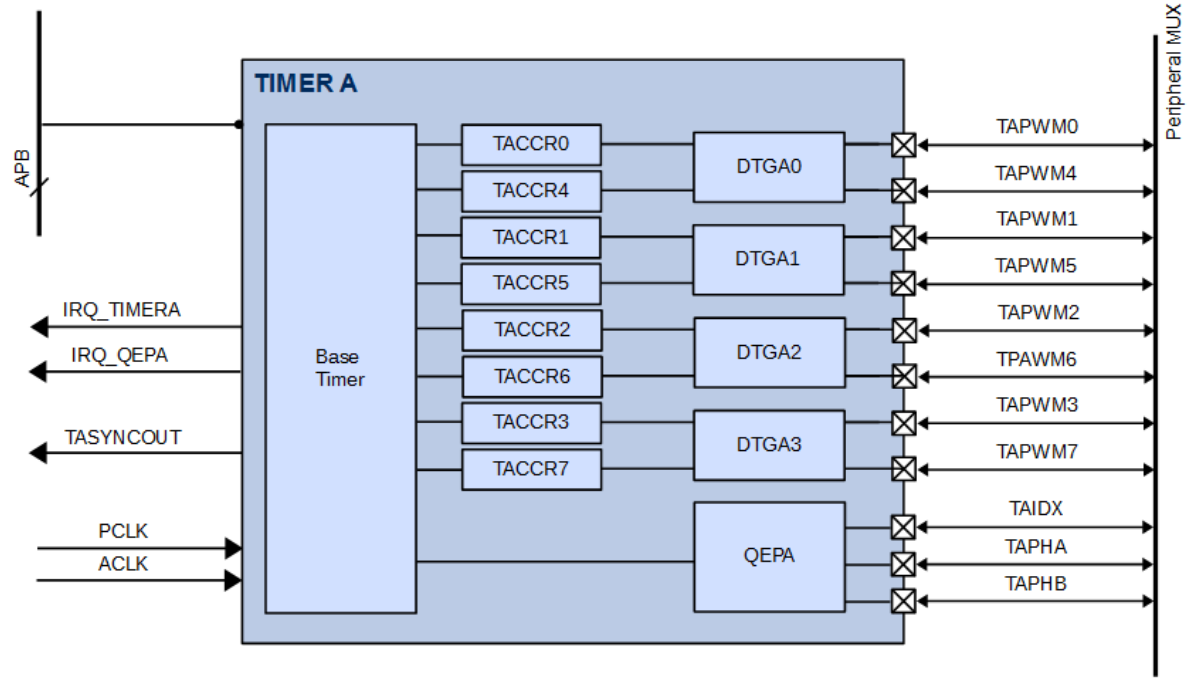
19 PWM TIMER A

19.1 Overview

All devices in the PAC55XX family of controllers have a Timer A peripheral. This peripheral is a 16-bit timer that allows support for 8 Capture and Compare Units (CCR) capable of PWM generation; capture input processing and a QEP decoder for various control applications.

Below is a simplified block diagram of the Timer peripheral.

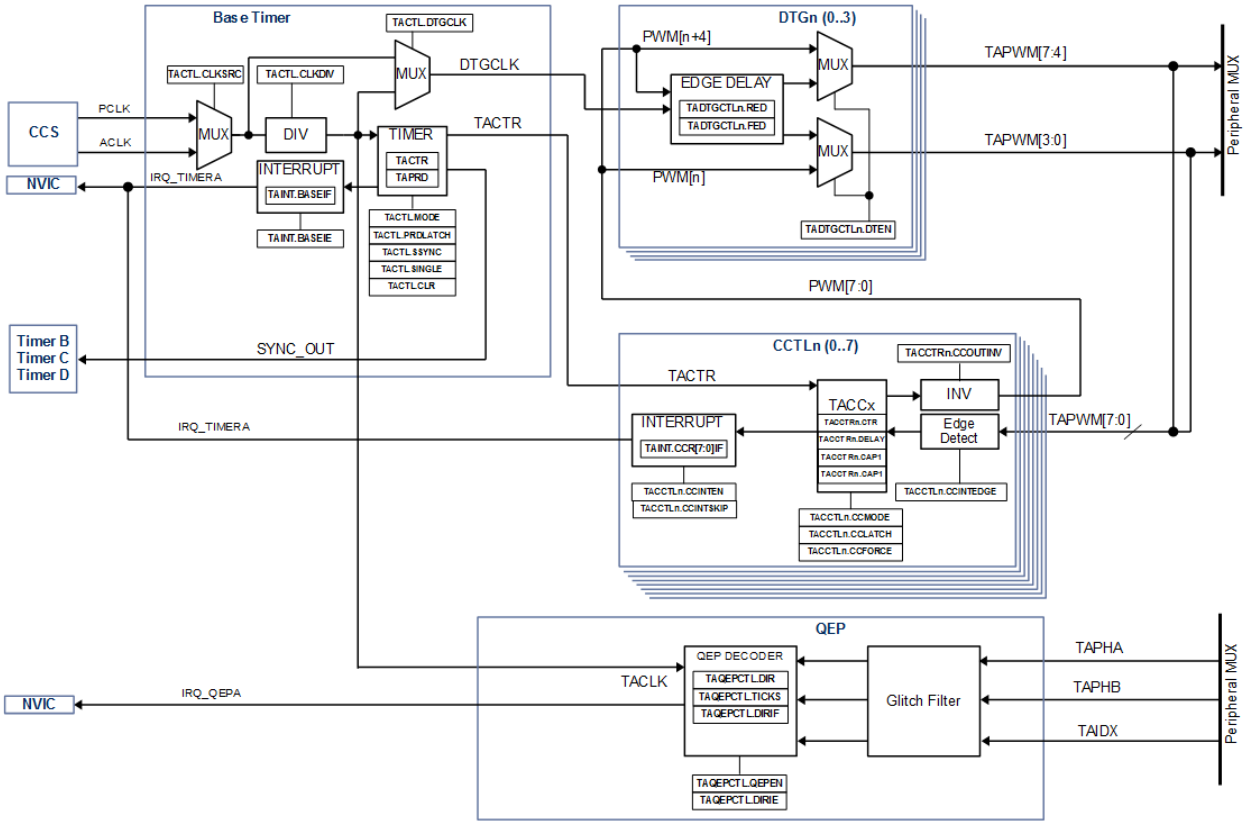
Figure 19-1 Timer A Simplified Block Diagram



The timer module is an APB bus client. There are 8 timer channel input/outputs to the digital peripheral MUX, in addition to three IOs for QEP. Each timer block has two interrupt signal outputs to the NVIC; one for timer functions and one for QEP functions. The user may select either PCLK or ACLK as the clock input for each timer module.

19.2 Timer A Block Diagram

Figure 19-2 Timer A Block Diagram



19.3 Features

The PWM Timer peripheral has the following features in the PAC55XX family of controllers.

General Features:

- Configurable input clock: PCLK or ACLK
- Up to 300MHz clock input for 3.33ns PWM edge resolution
- 3-bit input clock divider
- Latch timer period and all CCR values on command

Base Timer Features:

- Single-shot or auto-reload
- Base timer interrupts
- Timer synchronization
- Timer Modes:
  - Disabled
  - Up mode
  - Up/Down mode
  - Up/Down Asymmetric mode
- Timer Register Latching Options
  - Latch TAPRD when counter = 0
  - Latch TAPRD when counter = period
  - Latch TAPRD immediately
- Dead-time Generator Input Clock
  - DTG clock = PCLK
  - DTG clock = ACLK

CCR/PWM Features:

- PWM output or input capture
- 8 CCR units per timer
- CCR interrupts
- CCR interrupt skips
- SW force CCR interrupt
- CCR interrupt type:
  - Rising, falling, both
- CCR latch modes:
  - Compare/PWM mode: counter = 0, counter = period, immediate
  - Capture input: rising edge, falling edge, both

- Force compare event
- Invert CCR output
- CCR phase delay for phase shifted drive topologies
- ADC trigger outputs:
  - PWM rising or falling edge

Dead-Time Generator (DTG) Features:

- DTG enable or bypass
- 12-bit rising edge delay
- 12-bit falling edge delay

QEP Decoder Features:

- QEP decoder enabled
- Direction status
- Configurable interrupts:
  - Phase A rising edge
  - Phase B rising edge
  - Index event
  - Counter wrap
- 4 Different counting modes for best resolution, range and speed performance

19.4 Functional Description

19.4.1 Timer Clock Structure

The timer peripheral input clock can be selected as either PCLK or ACLK. The input clock may be selected by the **TACTL.CLKSRC** register.

Each timer peripheral has a 4-bit divider that can be used to divide the selected input clock. The user may set this divider by the **TACTL.CLKDIV** register. There are 16 settings between /1 and /128 for the input clock divider.

The base timer also supplies the clock for the Dead-Time Generators (DTGs) to allow for different range and resolution for dead-time. The timer may select the DTGCLK to be the timer clock before or after the clock divider. The DTG input clock may be selected by the **TACTL.DTGCLK** register.

To use the timer clock before the input clock divider, set **TACTL.DTGCLK** to 0b. To use the timer clock after the input clock divider, set **TACTL.DTGCLK** to 1b.

19.4.2 Timer Counter

The base timer is a 16-bit timer that can count either up, or up then down to support both edge and center aligned and asymmetric PWM output types.

The timer period is stored in the **TAPRD** register. The current value of the timer counter is updated at every timer tick and is stored in the **TACTR** register. The **TACTR** register is a RW register, so it may be updated at any time and is changed immediately.

The **TAPRD** register and all of the **TACCTRn** registers have shadow copies that are updated at a user-specified time. See the section below on Base Timer TAPRD Latching and CCR Timer Latching for more information.

When **TACTL.MODE** = 00b, the timer is disabled. Even when disabled, all timer registers are accessible via the APB bus.

To enable the timer, set the **TACTL.MODE** to 01b (up mode), 10b (up/down mode) or 11b (up/down asymmetric mode).

When **TACTL.MODE** = 01b, the timer is configured in up mode. The timer will count from 0 to **TAPRD**. The counter is updated in **TACTR** at every timer tick. If auto-reload is active (**TACTL.SINGLE** = 0b), then the timer count will automatically count from **TAPRD** to 0 and continue counting up. If auto-reload is not active (**TACTL.SINGLE** = 1b), then the timer will count from 0 to **TAPRD** and stop. When this happens, the timer will set the **TACTL.MODE** to 00b (disabled).

When the timer is configured for up/down mode (**TACTL.MODE** = 10b) or configured for up/down asymmetric mode (**TACTL.MODE** = 11b), the timer will count from 0 to **TAPRD** and then back down to 0. If auto-reload is not active (**TACTL.SINGLE** = 1b), the time timer will stop and set **TACTL.MODE** to 00b (disabled). If auto-reload is active (**TACTL.SINGLE** = 0b), then the timer count will then continue counting back up after it reaches 0.

19.4.3 Up Mode

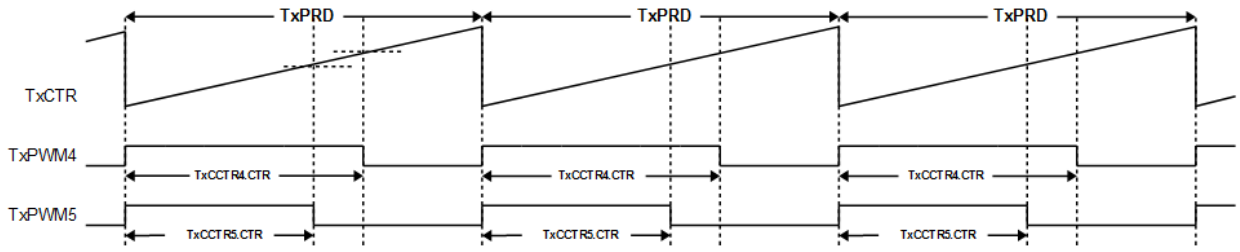
When the **TACTL.MODE** is set to 01b (up mode) the timer will begin counting up from 0. The **TACTR** will count to the value of **TAPRD** and then will reset to 0.

If the timer is configured for auto-reload mode (**TACTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it reaches 0.

The **TACTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate an edge aligned PWM as shown below.

Figure 19-3 Up Mode PWM Waveform



In this mode, when **TACTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TACTR** register reaches the **TACCTRn.CTR** threshold. When the **TACTR** counts from **TAPRD** to 0, the PWM output is transitioned to high.

19.4.4 Up/Down Mode

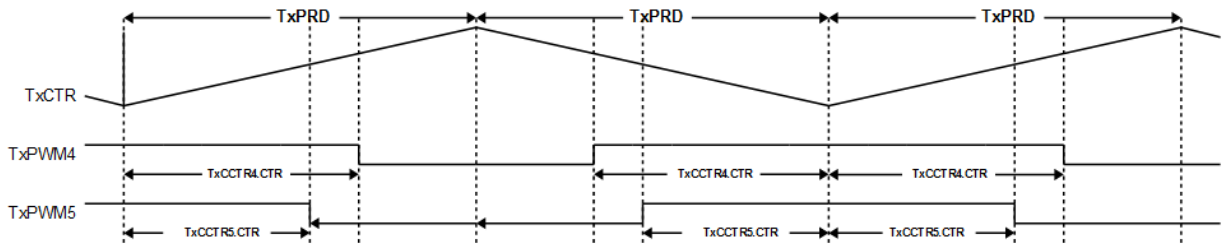
When the **TACTL.MODE** is set to 10b (up/down mode) the timer will begin counting up from 0. The **TACTR** will count to the value of **TAPRD** and then will begin counting back down to 0.

If the timer is configured for auto-reload mode (**TACTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it counts from 1 to 0.

The **TACTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate a center aligned PWM as shown below.

Figure 19-4 Up/Down Mode PWM Waveform



In this mode, when **TACTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TACTR** register reaches the **TACCTRn.CTR** threshold. When the **TACTR** counts from **TAPRD** - 1 to **TAPRD**, it will begin counting down. When the **TACTR** counts down to **TACCTRn.CTR** the PWM output is transitioned to high.

19.4.5 Up/Down Asymmetric mode

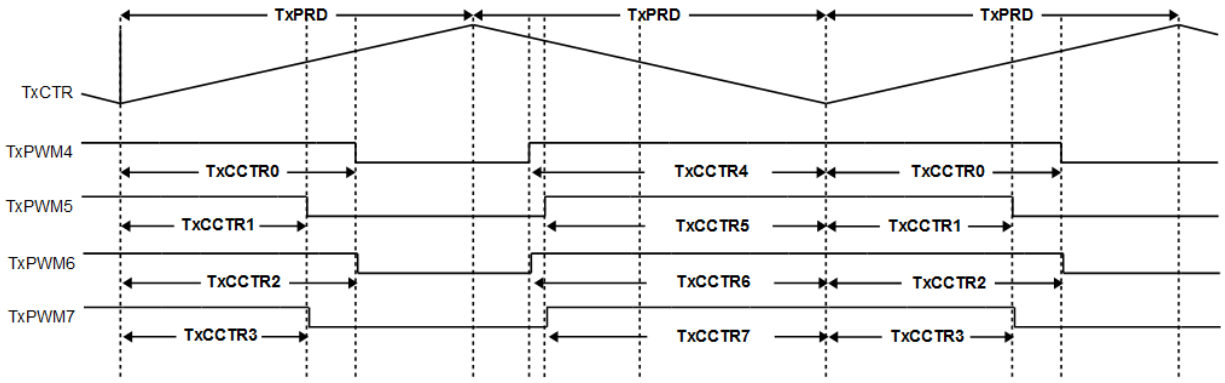
When the **TACTL.MODE** is set to 11b (up/down asymmetric mode) the timer will begin counting up from 0. The **TACTR** will count to the value of **TAPRD** and then will begin counting back down to 0.

If the timer is configured for auto-reload mode (**TACTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it counts from 1 to 0.

The **TACTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate a center aligned asymmetric PWM as shown below.

Figure 19-5 Up/Down Asymmetric Mode PWM Waveform



In this mode, when **TACTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TACTR** register reaches the **TACCTRn.CTR** threshold. When the **TACTR** counts from **TAPRD** - 1 to **TAPRD**, it will begin counting down. When the **TACTR** counts down to **TACCTRn.CTR** the PWM output is transitioned to high.

In this mode, the CCR outputs are generated such that 2 **TACCTRn.CTR** registers are used to generate a single PWM output, so that the center-aligned PWM can be generated with asymmetric on-time.

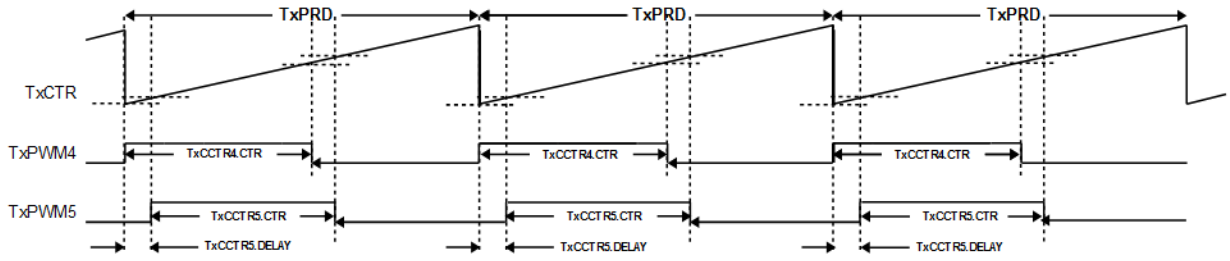
In this mode, the PWM outputs of the timer are generated as follows:

- TAPWM4 uses **TACCTR0.CTR** for the first portion of the on time and **TACCTR4.CTR** for the second portion of the on time
- TAPWM5 uses **TACCTR1.CTR** for the first portion of the on time and **TACCTR5.CTR** for the second portion of the on time
- TAPWM6 uses **TACCTR2.CTR** for the first portion of the on time and **TACCTR6.CTR** for the second portion of the on time
- TAPWM7 uses **TACCTR3.CTR** for the first portion of the on time and **TACCTR7.CTR** for the second portion of the on time

19.4.6 Up Mode with Phase Delay

When the timer is configured for up mode (**TAMODE** = 01b) the CCR allows a delay to be applied to the output PWM period, to support phase delay control topologies. If the value of **TACCTRn.DELAY** > 0, then the CCR will delay this number of ticks before starting the PWM period (transitioning the output signal from high to low).

Figure 19-6 Up Mode with Phase Delay



19.4.7 Timer Synchronization

The timer peripheral allows synchronization between the timers in the PAC55XX. The timers may be synchronized as follows:

- Timer A (master), Timer B (slave)
- Timer A (master), Timer B (slave), Timer C (slave)

- Timer A (master), Timer B (slave), Timer C (slave), Timer D (slave)

The timer master's SYNC_OUT signals are connected to the downstream timer slave's SYNC_IN signal.

When configured, the timers can synchronize their clocks so they are sharing the same time base. In order to do this, the timers must be configured with the same clock source (**TACTL.CLKSRC**) and clock divider (**TACTL.CLKDIV**).

To configure the timers for synchronization, follow these steps:

- While the timer master is disabled **TACTL.MODE** = 00b, set the timer master **TACTL.SSYNC** = 0b
- For each of the downstream timer slaves, while the timer is disabled **TACTL.MODE** = 00b, set the **TACTL.SSYNC** to 1b. Make sure that **TACTL.CLKDIV** and **TACTL.CLKSRC** for each of the timer slaves is the same as the timer master.
- Enable each of the slave timers by setting **TACTL.MODE** to 01b (up), 10b (up/down) or 11b (up/down asymmetric). The slave timers will not start counting since the **TACTL.SSYNC** is set to a 1b.
- Enable the timer master by setting **TACTL.MODE** to 01b (up), 10b (up/down) or 11b (up/down asymmetric).

At this point, the master and all configured timer slaves will begin counting.

19.4.8 Base Timer TAPRD Latching

The timer peripherals have a shadow copy of the TAPRD register that is used for the counting operations. The latching of the data from the TAPRD register to the shadow register is controlled by the timer configuration.

There are several options for latching of this register into the shadow copy. This behavior is controlled by the **TACTL.PRDLATCH** setting and depends on the setting of the **TACTL.MODE** register as well.

19.4.8.1 TAPRD Latch When TACTR = 0

- If **TACTL.MODE** = 01b (up mode) and **TACTL.PRDLATCH** = 00b (**TACTR** = 0), then the **TAPRD** register is copied into the shadow register when **TACTR** counts from **TAPRD** to 0.
- If **TACTL.MODE** = 10b (up/down mode) or if **TACTL.MODE** = 11b (up/down asymmetric mode) and **TACTL.PRDLATCH** = 00b (**TACTR** = 0), then the **TAPRD** register is copied into the shadow register when the **TACTR** counts from 1 to 0.

19.4.8.2 TAPRD Latch When TACTR = TAPRD

- If **TACTL.MODE** = 01b (up mode) or if **TACTL.MODE** = 10b (up/down mode) or if **TACTL.MODE** = 11b (up/down asymmetric mode) and **TACTL.PRDLATCH** = 01b (**TACTR** = **TAPRD**), then the **TAPRD** register is copied into the shadow register when **TACTR** counts from **TAPRD** – 1 to **TAPRD**.

19.4.8.3 TAPRD Latch Immediate

- If **TACTL.PRDLATCH** = 11b (immediate), then the **TAPRD** register is copied into the shadow register as soon as the **TAPRD** register is written.

19.4.9 CCR Timer Latching

When in compare mode, the CCR units support configurable latching of the **TACCRn.CTR** values into a shadow register that is used for timer operation. The shadow register is used for generating the PWM output when the CCR is in compare mode.

This feature can be used to control when the new duty cycle is applied, during the PWM period.

When the CCR is in compare mode (**TACCTLn.CCMODE** = 0b), the shadow register is latched as follows:

- If **TACCTLn.CCLATCH** = 00b (**TACCTLn.CTR** = 0)
  - If **TACTL.MODE** = 01b (up mode), **TACCTLn.CTR** is copied into the shadow register when **TACTR** counts from **TAPRD** to 0. If **TACTL.MODE** = 10b (up/down mode) or 11b (up/down asymmetric mode), **TACCTRn.CTR** is copied into the shadow register when **TACTR** counts from 1 to 0.
- If **TACCTLn.CCLATCH** = 01b (**TACTR = TAPRD**), then the **TACCTRn.CTR** is copied into the shadow register when **TACTR** counts from **TAPRD** – 1 to **TAPRD**.
- If **TACCTLn.CCLATCH** = 10b (latch immediate), then the TACCTRn.CTR is copied into the shadow register as soon as it is written.

When the CCR is in capture mode (**TACCTLn.CCMODE** = 1b), the configuration of **TACCTLn.CCLATCH** controls when the **TACTR** is copied into the **TACCTRn.CTR** register.

- If **TACCTLn.CCLATCH** = 00b (rising edge), then the value of **TACTR** is copied into the **TACCTLn.CTR** register upon a rising edge in the input signal.
- If **TACCTLn.CCLATCH** = 01b (falling edge), then the value of the **TACTR** is copied into the **TACCTLn.CTR** register upon a falling edge in the input signal.
- If **TACCTLn.CCLATCH** = 10b (both), then the value of the **TACTR** is copied into the TACCTLn.CTR register on both a rising and falling edge of the input signal.

19.4.10 Timer Whole Latching

It is sometimes convenient to latch both the **TAPRD** (period) and all **TACCTRn.CTR** (duty cycle) registers into the shadow registers at one time.

If the **TACTL.LATCH** bit is written to a 1b, then all the **TAPRD** and all **TACCTRn.CTR** registers will be copied into the shadow registers on the same clock cycle.

The **TACTL.LATCH** bit is self-clearing, and will always be read as a 0b.

19.4.11 Inverting CCR PWM Output

When the CCR is in compare mode (**TACCTRn.CCMODE** = 0b), the output of the CCR (the input to the DTG) may be inverted. To invert this output, set the **TACCTRn.CCOUTINV** = 1b.

This mode is useful for some control topologies, especially full-bridge.

19.4.12 Base Timer Interrupts

The base timer may be configured to generate an interrupt to the NVIC.

If **TACTL.MODE** = 01b (up mode) and **TACTR** counts from **TAPRD** to 0 then the **TAINT.BASEIF** bit is set to 1b. If the **TACTL.BASEIE** interrupt enable bit is set to a 1b, then the TIMER_IRQ signal to the NVIC is asserted.

If **TACTL.MODE** = 10b (up/down mode) or 11b (up/down asymmetric mode) and **TACTR** counts from 1 to 0, then the **TAINT.BASEIF** bit is set to 1b. If the **TACTL.BASEIE** interrupt enable bit is set to a 1b, then the TIMERA_IRQ signal to the NVIC is asserted.

The **TAINT.BASEIF** may be cleared by writing a 1b to it.

19.4.13 CCR Compare Interrupts

When **TxCCTLn.CCMODE** = 0b (compare mode), the CCR unit may be configured to generate an interrupt to the NVIC.

When the PWM edge transitions, then the CCR interrupt flag in the **TxINT** register is set as follows:

- If **TxCCTLn.CCINTEDGE** = 00b (rising edge) and a rising edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and TIMER_IRQ to the NVIC is asserted.
- If **TxCCTLn. CCINTEDGE** = 01b (falling edge) and a falling edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and TIMER_IRQ to the NVIC is asserted. **TxINT.CCRnIF** may be cleared by writing it to a 1b.
- If **TxCCTLn. CCINTEDGE** = 10b (rising or falling edge) and a rising or falling edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and

TIMER_IRQ to the NVIC is asserted. **TxINT.CCRnIF** may be cleared by writing it to a 1b.

TxINT.CCRnIF may be cleared by writing it to a 1b.

19.4.14 CCR Capture Interrupts

When **TxCCTLn.CCMODE** = 1b (capture mode), the CCR unit may be configured to generate an interrupt to the NVIC based on the input signal.

When the PWM edge transitions, then the CCR interrupt flag in the **TxINT** register is set as follows:

- If **TxCCTLn.CCINTEDGE** = 00b (rising edge) and a rising edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is set.
- If **TxCCTLn.CCINTEDGE** = 01b (falling edge) and a falling edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is set.
- If **TxCCTLn.CCINTEDGE** = 10b (rising or falling edge) and a rising or falling edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is.

If the **TxINT.CCRnIF** interrupt flag is set and the corresponding **TxCCTLn.CCINTEN** bit is set, then the TIMERA_IRQ signal to the NVIC will be asserted.

The **TxINT.CCRnIF** interrupt flag can be cleared by writing it to a 1b.

19.4.15 Timer IRQ signal

The timer unit has one IRQ output signal to the NVIC (TIMERA_IRQ).

This signal is asserted when any of the following conditions are true:

- **TAINT.BASEIF** = 1b and **TACTL.BASEIE** = 1b
- **TAINT.CCR0IF** = 1b and **TACCTL0.CCINTEN** = 1b
- **TAINT.CCR1IF** = 1b and **TACCTL1.CCINTEN** = 1b
- **TAINT.CCR2IF** = 1b and **TACCTL2.CCINTEN** = 1b
- **TAINT.CCR3IF** = 1b and **TACCTL3.CCINTEN** = 1b
- **TAINT.CCR4IF** = 1b and **TACCTL4.CCINTEN** = 1b
- **TAINT.CCR5IF** = 1b and **TACCTL5.CCINTEN** = 1b
- **TAINT.CCR6IF** = 1b and **TACCTL6.CCINTEN** = 1b
- **TAINT.CCR7IF** = 1b and **TACCTL7.CCINTEN** = 1b

The TIMERA_IRQ signal is de-asserted when all of the above conditions are false.

19.4.16 Skipping CCR Interrupts

Sometimes it is useful to not generate a CCR interrupt to the NVIC every time the **TACTR** counts to the **TACCTRn.CTR** value. For example, in a control application when the PWM frequency is fast, but you only need a MCU interrupt every 5 PWM periods.

Each CCR in the timer has the ability to skip interrupts to the NVIC to allow this capability.

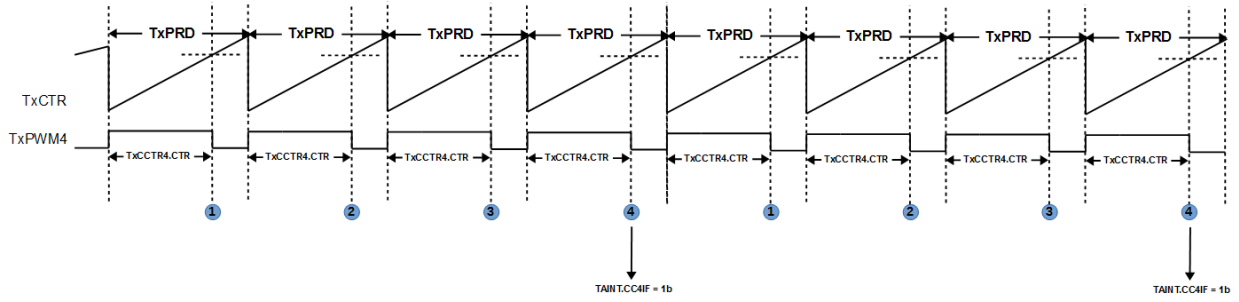
The CCR will generate interrupts to the NVIC according to the TACCTL.CCINTSKIP field as follows:

- If **TACCTLn.CCINTSKIP** = 0000b, then the CCR will generate an interrupt each time
- If **TACCTLn.CCINTSKIP** = 0001b, then the CCR will skip 1 CCR cycle before generating an interrupt to the NVIC.
- If **TACCTLn.CCINTSKIP** = 0010b, then the CCR will skip 2 CCR cycles before generating an interrupt to the NVIC.
- If **TACCTLn.CCINTSKIP** = 0011b, then the CCR will skip 3 CCR cycles before generating an interrupt to the NVIC.
- …

- If **TACCTLn.CCINTSKIP** = 1110b, then the CCR will skip 14 CCR cycles before generating an interrupt to the NVIC.
- If **TACCTLn.CCINTSKIP** = 1111b, then the CCR will skip 15 CCR cycles before generating an interrupt to the NVIC.

The diagram below shows an example of a CCR4 with interrupt skipping configured with a value of 4 (**TACCTL.CCINTSKIP** = 4).

Figure 19-7 CCR Interrupt Skipping Diagram



19.4.17 Timer ADC Triggers

Each timer CCR unit may be configured to be an ADC trigger, to allows the DTSE to automatically begin conversion sequences.

For more information on this feature, see the section on the ADC and DTSE in this user guide.

19.4.18 Dead-Time Generators (DTG)

Each timer has 4 DTG units. Each DTG unit is capable of generating a pair of complementary signals that can be configured with dead-time to drive an inverter for half-bridge topologies.

The input clock to the DTG can be configured to be the timer clock before or after the **TACTL.CLKDIV** input clock divider. If **TACTL.DTGCLK** is 0b, then the DTGCLK is the clock before the **TACTL.CLKDIV** clock divider. If **TACTL.DTGCLK** is 1b, then the DTGCLK is the clock after the **TACTL.CLKDIV** clock divider.

When **TADTGCTLn.DTEN** = 0b, the DTG is disabled (bypassed). In this mode, the timer signals have the following behavior (see the simplified block diagram above):

DTG0:

- CCR0 output is connected to TAPWM0
- CCR4 output is connected to TAPWM4

DTG1:

- CCR1 output is connected to TAPWM1
- CCR5 output is connected to TAPWM5

DTG2:

- CCR2 output is connected to TAPWM2
- CCR6 output is connected to TAPWM6

DTG3:

- CCR3 output is connected to TAPWM3
- CCR7 output is connected to TAPWM7

When **TADTGCTL0.DTEN** = 1b, the DTG is enabled. In this mode, only one CCR output is used to generate the two complementary outputs. In this mode, the timer signals have the following behavior:

DTG0:

- CCR0 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR4 output is used to generate the two complementary output signals: TAPWM0 and TAPWM4.

- TAPWM0 becomes the low-side complementary signal.
- TAPWM4 becomes the high-side complementary signal.
- The **TADTGCTL0.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, to when the low-side signal rises).
- The **TADTGCTL0.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, to when the high-side signal rises).

DTG1:

- CCR1 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR5 output is used to generate the two complementary output signals: TAPWM1 and TAPWM5.
- TAPWM1 becomes the low-side complementary signal.
- TAPWM5 becomes the high-side complementary signal.
- The **TADTGCTL1.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TADTGCTL1.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

DTG2:

- CCR2 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR6 output is used to generate the two complementary output signals: TAPWM2 and TAPWM6.
- TAPWM2 becomes the low-side complementary signal.
- TAPWM6 becomes the high-side complementary signal.
- The **TADTGCTL2.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TADTGCTL2.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

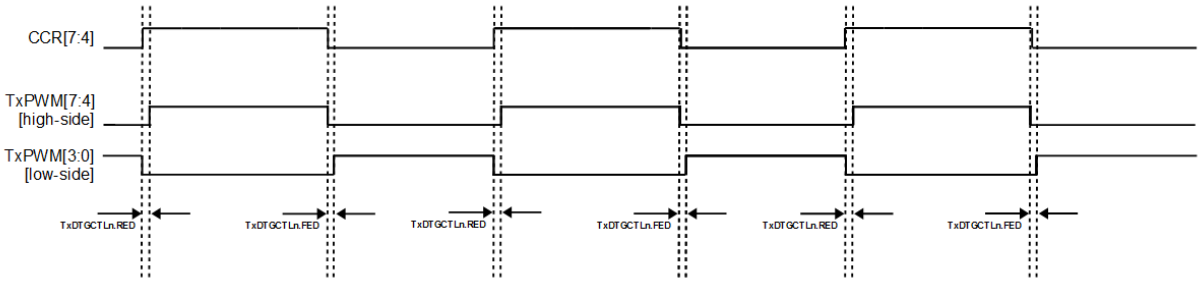
DTG3:

- CCR3 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR7 output is used to generate the two complementary output signals: TAPWM3 and TAPWM4.
- TAPWM3 becomes the low-side complementary signal.
- TAPWM7 becomes the high-side complementary signal.
- The **TADTGCTL3.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TADTGCTL0.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

The PWM waveform below shows an example of the dead-time applied to the signals through the DTG units.

19.4.19 Timer A DTG Output

Figure 19-8 Timer A DTG Output



19.5 QEP MCU v1

The PAC55XX MCU v1 contains a QEP encoder peripheral that can be used to determine position and speed of a position-encoded motor. The QEP encoder can be enabled by setting the **TAQEPCTL.QEPEN** bit to a 1b.[16]

The inputs to the QEP peripheral are TBQEPPHA, TBQEPPHB and TBQEPIDX (phase A, phase B and index). These inputs are used to determine the position and direction of the motor, when the **TAQEPCTL.QEPEN** is set to a 1b. These three inputs have a glitch filter applied to them, to filter out noise and are used for calculation of the motor position and speed.

After filtering, the TBQEPPHA, TBQEPPHB and TBQEPIDX inputs are passed to the QEP engine to determine the position and speed. If the position is determined to be clockwise, then the **TAQEPCTL.DIR** is set to 0b. If the position is determined to be counter-clockwise, then the **TAQEPCTL.DIR** is set to 1b.

During operation, if the **TAQEPCTL.CNTAB** bit is 0b, only phase A edges are counted. If this bit is set to a 1b, then both phase A and phase B edges are counted. If the **TAQEPCTL.CNTEDGE** bit is set to 0b, then only the rising edges are counted. If this bit is set to a 1b, then both rising and falling edges are counted.

If the **TAQEPCTL.IDXRST** bit is set to a 1b, then an index event clears the **TAQEPCTL.TICKS** counter to 0. If this bit is set to a 0b, then an index event does not reset the **TAQEPCTL.TICKS** counter.

If a change in the direction is made (0 to 1, or 1 to 0), then the **TAQEPCTL.DIRIF** bit is set to a 1b. If the **TAQEPCTL.DIRIE** bit is set to 1b, then the QEPA_IRQ signal is asserted to the NVIC. The **TAQEPCTL.DIRIF** bit can be cleared by writing it to 1b.

When a rising edge on phase A is detected the **TAQEPCTL.PHAIF** bit is set to 1b. If the **TAQEPCTL.PHAIE** bit is set and the **TAQEPCTL.PHAIF** bit is set, then the QEPA_IRQ signal is asserted to the NVIC. The **TAQEPCTL.PHAIF** bit can be cleared by writing a 1b to it.

When a rising edge on phase B is detected the **TAQEPCTL.PHBIF** bit is set. If the **TAQEPCTL.PHBIE** bit is set and the **TAQEPCTL.PHBIF** bit is set, then the QEPA_IRQ signal is asserted to the NVIC. The **TAQEPCTL.PHBIF** bit can be cleared by writing a 1b to it.

[16] Before enabling the QEP peripheral, be sure to set up the Digital Peripheral MUX, so that no false edges are detected by the QEP state machine.

If an overflow or underflow in the **TAQEPCTL.TICKS** is detected, the **TAQEPCTL.WRIF** bit is set (counter wrap interrupt flag). If the **TAQEPCTL.WRIE** bit is set and the **TAQEPCTL.WRIF** bits are set, then the QEPA_IRQ signal is asserted to the NVIC.

If an index event is detected, the **TAQEPCTL.IDXEVIF** bit is set. If the **TAQEPCTL.IDXEVIE** bit is set and the **TAQEPCTL.IDXEVIF** bit is set, then the QEPA_IRQ signal to the NVIC is asserted. The **TAQEPCTL.IDXEVIF** bit can be cleared by writing a 1b to it.

As the QEP decoder counts edges on PH_A and is turning clockwise (CW), it increments the counter value in **TAQEPCTL.TICKS**. If the motor is turning counter-clockwise (CCW), the **TAQEPCTL.TICKS** register is decremented. The user may write the **TAQEPCTL**.**TICKS** register at any time.

19.6 QEP MCU v2

19.6.1 Overview

Each PAC55XX MCU v2 PWM Timer contains a QEP decoder that can be used to determine the position, speed, and direction of a motor with a QEP encoder.

The inputs to the QEP peripheral are TxQEPPHA, TxQEPPHB and TxQEPIDX (phase A, phase B and index). These three inputs have a glitch filter applied to them, to filter out noise and are used for calculation of the motor position, speed, and direction.

19.6.2 Configuration

The QEP peripheral can be enabled by setting the **TxQEPCTL.QEPEN** bit to a 1b.[17] The input clock to the QEP peripheral may be configured between /1 and /128 using the **TxQEPCTL.CLKDIV** field.

If the **TxQEPCTL.CNTAB** bit is 0b, only phase A edges are processed. If the **TxQEPCTL.CNTAB** bit is set to a 1b, then both phase A and phase B edges are processed. If the **TxQEPCTL.CNTEDGE** bit is set to 0b, then only the rising edges are processed. If the **TxQEPCTL.CNTEDGE** bit is set to a 1b, then both rising and falling edges are processed.

If the direction is determined to be clockwise (CW), then the **TxQEPCTL.DIR** bit is set to 0b. If the direction is determined to be counterclockwise (CCW), then the **TxQEPCTL.DIR** bit is set to 1b. The direction will get updated only on edges that are processed.

[17] Before enabling the QEP peripheral, be sure to set up the Digital Peripheral MUX, so that no false edges are detected by the QEP state machine.

When an edge is processed, the QEP peripheral will first determine the direction and then increment or decrement the counter value in **TxQEPCTR.CTR** accordingly. **TxQEPCTR.CTR** may be written at any time.

If an edge is processed while the motor is turning clockwise, the counter value in **TxQEPCTR.CTR** is incremented. If the value of **TxQEPCTR.CTR** was **TxQEPMAX.CTRMAX** during this event, then **TxQEPCTR.CTR** is set to 0.

If an edge is processed while the motor is turning counterclockwise, the counter value in **TxQEPCTR.CTR** is decremented. If the value of **TxQEPCTR.CTR** was 0 during this event, then **TxQEPCTR.CTR** is set to **TxQEPMAX.CTRMAX**.

19.6.3 Index Reset

If the **TxQEPCTL.IDXRST** bit is set to a 1b, then an index event clears the **TxQEPCTR.CTR** counter to 0. If the motor is turning in the clockwise direction during this event, **TxQEPCTR.CTR** is set to 0. If the motor is turning in the counterclockwise direction during this event the **TxQEPCTR.CTR** is set to 0 (if not coincident with the phase edge) or **TxQEPMAX.CTRMAX** (if coincident with the phase edge).

If this bit is set to a 0b, then an index event does not reset the **TAQEPCTL.TICKS** counter.

19.6.4 Interrupts

If a change to the direction is detected (0 to 1 or 1 to 0), then the **TxQEPIF.DIRIF** bit is set to a 1b. If the **TxQEPIER.DIRIE** bit is set to 1b, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.DIRIF** bit can be cleared by writing it to 1b.

When a rising edge on phase A is detected the **TxQEPIF.PHAIF** bit is set to 1b. If the **TxQEPIER.PHAIE** bit is set and the **TxQEPIF.PHAIF** bit is set, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.PHAIF** bit can be cleared by writing a 1b to it.

When a rising edge on phase B is detected the **TxQEPIF.PHBIF** bit is set. If the **TxQEPIER.PHBIE** bit is set and the **TxQEPIF.PHBIF** bit is set, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.PHBIF** bit can be cleared by writing a 1b to it.

If an overflow or underflow in the **TxQEPCTR.CTR** is detected, the **TxQEPIF.WRIF** bit is set (counter wrap interrupt flag). If the **TxQEPIER.WRIE** bit is set and the **TxQEPIF.WRIF** bits are set, then the QEPx_IRQ signal is asserted to the NVIC.

If an index event is detected, the **TxQEPIF.IDXEVIF** bit is set. If the **TxQEPIER.IDXEVIE** bit is set and the **TxQEPIF.IDXEVIF** bit is set, then the QEPx_IRQ signal to the NVIC is asserted. The **TxQEPIF.IDXEVIF** bit can be cleared by writing a 1b to it.

19.7 Peripheral IO Mapping

The Timer A peripheral signal inputs and outputs (CCR, QEP) are connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 19-1 Timer A Peripheral IO Mapping

TIMER SIGNAL	IO PIN
TAPWM0	PB0
TAPWM1	PB1
TAPWM2	PB2
TAPWM3	PB3
TAPWM4	PB4
TAPWM5	PB5
TAPWM6	PB6
TAPWM7	PB7
TAQEPIDX	PE0
TAQEPPHA	PE1
TAQEPPHB	PE2

For more information on how to use the Digital Peripheral MUX to connect peripheral signals to IO, see GPIO and DPM.

19.8 Timer A Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
TACTL	4006 0000h	Timer A control	RW	0000 0000h
TAINT	4006 0004h	Timer A interrupt control	RW	0000 0000h
TAPRD	4006 0008h	Timer A period	RW	0000 0000h
TACTR	4006 000Ch	Timer A counter	RW	0000 0000h
TACCTL0	4006 0100h	Timer A CC control 0	RW	0000 0000h
TACCTR0	4006 0104h	Timer A CC counter 0	RW	0000 0000h
TACCTL1	4006 0108h	Timer A CC control 1	RW	0000 0000h
TACCTR1	4006 010Ch	Timer A CC counter 1	RW	0000 0000h
TACCTL2	4006 0110h	Timer A CC control 2	RW	0000 0000h
TACCTR2	4006 0114h	Timer A CC counter 2	RW	0000 0000h
TACCTL3	4006 0118h	Timer A CC control 3	RW	0000 0000h
TACCTR3	4006 011Ch	Timer A CC counter 3	RW	0000 0000h
TACCTL4	4006 0120h	Timer A CC control 4	RW	0000 0000h
TACCTR4	4006 0124h	Timer A CC counter 4	RW	0000 0000h
TACCTL5	4006 0128h	Timer A CC control 5	RW	0000 0000h
TACCTR5	4006 012Ch	Timer A CC counter 5	RW	0000 0000h
TACCTL6	4006 0130h	Timer A CC control 6	RW	0000 0000h
TACCTR6	4006 0134h	Timer A CC counter 6	RW	0000 0000h
TACCTL7	4006 0138h	Timer A CC control 7	RW	0000 0000h
TACCTR7	4006 013Ch	Timer A CC counter 7	RW	0000 0000h
TADTGCTL0	4006 0200h	Timer A DTG control 0	RW	0000 0000h
TADTGCTL1	4006 0204h	Timer A DTG control 1	RW	0000 0000h
TADTGCTL2	4006 0208h	Timer A DTG control 2	RW	0000 0000h
TADTGCTL3	4006 020Ch	Timer A DTG control 3	RW	0000 0000h
TAQEPCTL MCU v2 only	4006 0300h	Timer A QEP control	RW	0000 0000h
TAQEPCTR	4006 0304h	Timer A QEP counter	RW	0000 0000h
TAQEPMAX	4006 0308h	Timer A QEP max counter value	RW	0000 FFFFh
TAQEPIER	4006 030Ch	Timer A QEP interrupt enable	RW	0000 0000h
TAQEPIFR	4006 0310h	Timer A QEP interrupt flag	RW	0000 0000h

19.9 Register Detail

19.9.1 TACTL

Register 19-1 TACTL (Timer A Control, 4006 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:14	**Reserved**	RO	0	Reserved
13	**BASEIE**	RW	0	Base timer interrupt enable: 0b: not enabled 1b: enabled
12	**CLR**	RW	0	Timer Clear: 0b: do not clear 1b: clear timer counter
11	**LATCH**	RW	0	When written to 1b, this will latch the **TAPRD** and all **TACCTRn** registers on the same clock cycle. Writing this bit to 0b has no effect. This is a self-clearing bit.
10	**DTGCLK**	RW	0	DTG Clock Source: 0b: Before input clock divider 1b: After input clock divider
9	**CLKSRC**	RW	0	Timer Clock Source: 0b: PCLK 1b: ACLK
8:6	**CLKDIV**	RW	0	Timer Input Clock Divider: 000b: /1 001b: /2 010b: /4 011b: /8 100b: /16 101b: /32 110b: /64 111b: /128
5	**SINGLE**	RW	0	Single Shot Timer: 0b: disabled (auto-reload) 1b: enabled (single shot timer)
4	**SSYNC**	RW	0	Timer Slave Synchronization: 0b: disabled 1b: enabled
3:2	**PRDLATCH**	RW	0	Timer Period Latch Mode: 00b: Latch **TAPRD** when **TACTR** = 0 01b: Latch **TAPRD** when **TACTR** = **TAPRD** 10b: Latch **TAPRD** immediately upon register write 11b: reserved
1:0	**MODE**	RW	0	Timer Mode: 00b: disabled 01b: up mode 10b: up/down mode 11b: asymmetric mode

19.9.2 TAINT

Register 19-2 TAINT (Timer A Interrupt Control, 4006 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:9	**Reserved**	RO	0	Reserved
8	**BASEIF**	W1C	0	Base timer interrupt flag: 0b: no interrupt flag 1b: interrupt flag
7	**CCR7IF**	W1C	0	CCR7 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
6	**CCR6IF**	W1C	0	CCR6 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
5	**CCR5IF**	W1C	0	CCR5 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
4	**CCR4IF**	W1C	0	CCR4 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
3	**CCR3IF**	W1C	0	CCR3 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
2	**CCR2IF**	W1C	0	CCR2 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
1	**CCR1IF**	W1C	0	CCR1 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
0	**CCR0IF**	W1C	0	CCR0 interrupt flag: 0b: no interrupt flag 1b: interrupt flag

19.9.3 TAPRD

Register 19-3 TAPRD (Timer A Period, 4006 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**PERIOD**	RW	0	Timer period value

19.9.4 TACTR

Register 19-4 TACTR (Timer A Counter, 4006 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**COUNTER**	RW	0	Timer counter

19.9.5 TACCTL0

Register 19-5 TACCTL0 (Timer A CCR Control 0, 4006 0100h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TACCTL0.CCMODE** = 0b (compare): 00b: Latch **TACCTR0** registers when **TACTR** = 0 01b: Latch **TACCTR0** registers when **TACTR** = **TAPRD** 10b: Latch **TACCTR0** registers immediately 11b: Reserved If **TACCTL0.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR0.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR0.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR0.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TACCTL0.CCMODE** = 1b (capture), set **TAINT.CCR0IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

19.9.6 TACCTR0

Register 19-6 TACCTR0 (Timer A CCR Counter 0, 4006 0104h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TACCTL0.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TACCTRL0.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TACCTL0.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

19.9.7 TACCTL1

Register 19-7 TACCTL1 (Timer A CCR Control 1, 4006 0108h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TACCTL1.CCMODE** = 0b (compare): 00b: Latch **TACCTR1** registers when **TACTR** = 0 01b: Latch **TACCTR1** registers when **TACTR** = **TAPRD** 10b: Latch **TACCTR1** registers immediately 11b: Reserved If **TACCTL1.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR1.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR1.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR1.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TACCTL1.CCMODE** = 1b (capture), set **TAINT.CCR1IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

19.9.8 TACCTR1

Register 19-8 TACCTR1 (Timer A CCR Counter 1, 4006 010Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TACCTL1.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TACCTRL1.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TACCTL1.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

19.9.9 TACCTL2

Register 19-9 TACCTL2 (Timer A CCR Control 2, 4006 0110h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TACCTL2.CCMODE** = 0b (compare): 00b: Latch **TACCTR2** registers when **TACTR** = 0 01b: Latch **TACCTR2** registers when **TACTR** = **TAPRD** 10b: Latch **TACCTR2** registers immediately 11b: Reserved If **TACCTL2.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR2.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR2.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR2.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TACCTL2.CCMODE** = 1b (capture), set **TAINT.CCR2IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

19.9.10 TACCTR2

Register 19-10 TACCTR2 (Timer A CCR Counter 2, 4006 0114h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TACCTL2.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TACCTRL2.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TACCTL2.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

19.9.11 TACCTL3

Register 19-11 TACCTL3 (Timer A CCR Control 3, 4006 0118h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**R**	RW	0	CCR register latch mode. If **TACCTL3.CCMODE** = 0b (compare): 00b: Latch **TACCTR3** registers when **TACTR** = 0 01b: Latch **TACCTR3** registers when **TACTR** = **TAPRD** 10b: Latch **TACCTR3** registers immediately 11b: Reserved If **TACCTL3.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR3.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR3.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR3.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TACCTL3.CCMODE** = 1b (capture), set **TAINT.CCR3IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

19.9.12 TACCTR3

Register 19-12 TACCTR3 (Timer A CCR Counter 3, 4006 011Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TACCTL3.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TACCTRL3.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TACCTL3.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

19.9.13 TACCTL4

Register 19-13 TACCTL4 (Timer A CCR Control 4, 4006 0120h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TACCTL4.CCMODE** = 0b (compare): 00b: Latch **TACCTR4** registers when **TACTR** = 0 01b: Latch **TACCTR4** registers when **TACTR** = **TAPRD** 10b: Latch **TACCTR4** registers immediately 11b: Reserved If **TACCTL4.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR4.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR4.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR4.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TACCTL4.CCMODE** = 1b (capture), set **TAINT.CCR4IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

19.9.14 TACCTR4

Register 19-14 TACCTR4 (Timer A CCR Counter 4, 4006 0124h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TACCTL4.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TACCTRL4.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TACCTL4.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

19.9.15 TACCTL5

Register 19-15 TACCTL5 (Timer A CCR Control 5, 4006 0128h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TACCTL5.CCMODE** = 0b (compare): 00b: Latch **TACCTR5** registers when **TACTR** = 0 01b: Latch **TACCTR5** registers when **TACTR** = **TAPRD** 10b: Latch **TACCTR5** registers immediately 11b: Reserved If **TACCTL5.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR5.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR5.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR5.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TACCTL5.CCMODE** = 1b (capture), set **TAINT.CCR5IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

19.9.16 TACCTR5

Register 19-16 TACCTR5 (Timer A CCR Counter 5, 4006 012Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TACCTL5.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TACCTRL5.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TACCTL5.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

19.9.17 TACCTL6

Register 19-17 TACCTL6 (Timer A CCR Control 6, 4006 0130h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TACCTL6.CCMODE** = 0b (compare): 00b: Latch **TACCTR6** registers when **TACTR** = 0 01b: Latch **TACCTR6** registers when **TACTR** = **TAPRD** 10b: Latch **TACCTR6** registers immediately 11b: Reserved If **TACCTL6.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR6.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR6.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR6.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TACCTL6.CCMODE** = 1b (capture), set **TAINT.CCR6IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

19.9.18 TACCTR6

Register 19-18 TACCTR6 (Timer A CCR Counter 6, 4006 0134h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TACCTL6.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TACCTRL6.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TACCTL6.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

19.9.19 TACCTL7

Register 19-19 TACCTL7 (Timer A CCR Control 7, 4006 0138h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TACCTL7.CCMODE** = 0b (compare): 00b: Latch **TACCTR7** registers when **TACTR** = 0 01b: Latch **TACCTR7** registers when **TACTR** = **TAPRD** 10b: Latch **TACCTR7** registers immediately 11b: Reserved If **TACCTL7.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR7.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR7.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR7.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TACCTL7.CCMODE** = 1b (capture), set **TAINT.CCR7IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

19.9.20 TACCTR7

Register 19-20 TACCTR7 (Timer A CCR Counter 7, 4006 013Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TACCTL7.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TACCTRL7.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TACCTL7.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

19.9.21 TADTGCTL0

Register 19-21 TADTGCTL0 (Timer A Dead-Time Generator Control 0, 4006 0200h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG0.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG0.

19.9.22 TADTGCTL1

Register 19-22 TADTGCTL1 (Timer A Dead-Time Generator Control 1, 4006 0204h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG1.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG1.

19.9.23 TADTGCTL2

Register 19-23 TADTGCTL2 (Timer A Dead-Time Generator Control 2, 4006 0208h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG2.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG2.

19.9.24 TADTGCTL3

Register 19-24 TADTGCTL3 (Timer A Dead-Time Generator Control 3, 4006 020Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG3.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG3.

19.9.25 TAQEPCTL

Register 19-25 TAQEPCTL (Timer A QEP Control, 4006 0300h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:11	**Reserved**	RO	0	Reserved
10:8	**CLKDIV**	RW	0	QEP Input Clock Divider: 000b: /1 001b: /2 010b: /4 011b: /8 100b: /16 101b: /32 110b: /64 111b: /128
7:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXRST**	RW	0	Index reset mode. Used for resetting counter. 0b: No reset 1b: Index edge reset. When direction is CW, reset counter to 0 when index rising edge event occurs. When direction is CCW, reset when index falling edge occurs (reset to 0 if not coincident with phase edge; reset to Tx**QEPMAX** if coincident with phase edge).
3	**DIR**	RO	0	Motor direction: 0b: CW (clockwise) 1b: CCW (counter-clockwise)
2	**CNTEDGE**	RW	0	Count on edge: 0b: Rising edge only 1b: Rising and falling edge
1	**CNTAB**	RW	0	Count on A/B: 0b: Count phase A only 1b: Count phase A and phase B
0	**QEPEN**	RW	0	QEP peripheral enable: 0b: disabled 1b: enabled

19.9.26 TAQEPCTR

Register 19-26 TAQEPCTR (Timer A QEP Counter, 4006 0304h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**CTR**	RW	0	Counter Value

19.9.27 TAQEPMAX

Register 19-27 TAQEPMAX (Timer A QEP Maximum Counter, 4006 0308h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**CTRMAX**	RW	0	Maximum Counter Value.

19.9.28 TAQEPIER

Register 19-28 TAQEPIER (Timer A QEP Interrupt Enable, 4006 030Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXEVIE**	RW	0	Index event interrupt enable: 0b: disabled 1b: enabled
3	**WRIE**	RW	0	Counter wrap interrupt enable: 0b: disabled 1b: enabled
2	**PHBIE**	RW	0	Phase B rising edge interrupt enable: 0b: disabled 1b: enabled
1	**PHAIE**	RW	0	Phase A rising edge interrupt enable: 0b: disabled 1b: enabled
0	**DIRIE**	RW	0	Direction change interrupt enable: 0b: disabled 1b: enabled

19.9.29 TAQEPIF

Register 19-29 TAQEPIF (Timer A QEP Interrupt Flag, 4006 0310h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXEVIF**	W1C	0	Index event interrupt flag: 0b: disabled 1b: enabled
3	**WRIF**	W1C	0	Counter wrap interrupt flag: 0b: disabled 1b: enabled
2	**PHBIF**	W1C	0	Phase B rising edge interrupt flag: 0b: disabled 1b: enabled
1	**PHAIF**	W1C	0	Phase A rising edge interrupt flag: 0b: disabled 1b: enabled
0	**DIRIF**	W1C	0	Direction change interrupt flag: 0b: disabled 1b: enabled

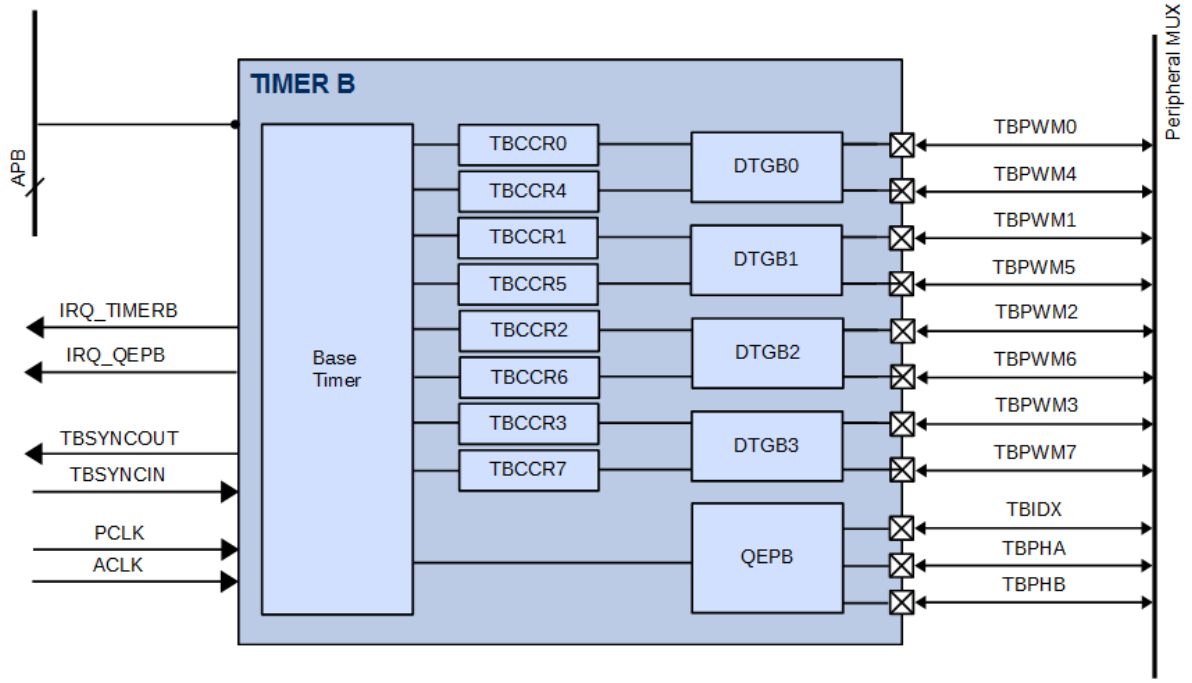
20 PWM TIMER B

20.1 Overview

All devices in the PAC55XX family of controllers have a Timer B peripheral. This peripheral is a 16-bit timer that allows support for 8 Capture and Compare Units (CCR) capable of PWM generation; capture input processing and a QEP decoder for various control applications.

Below is a simplified block diagram of the Timer peripheral.

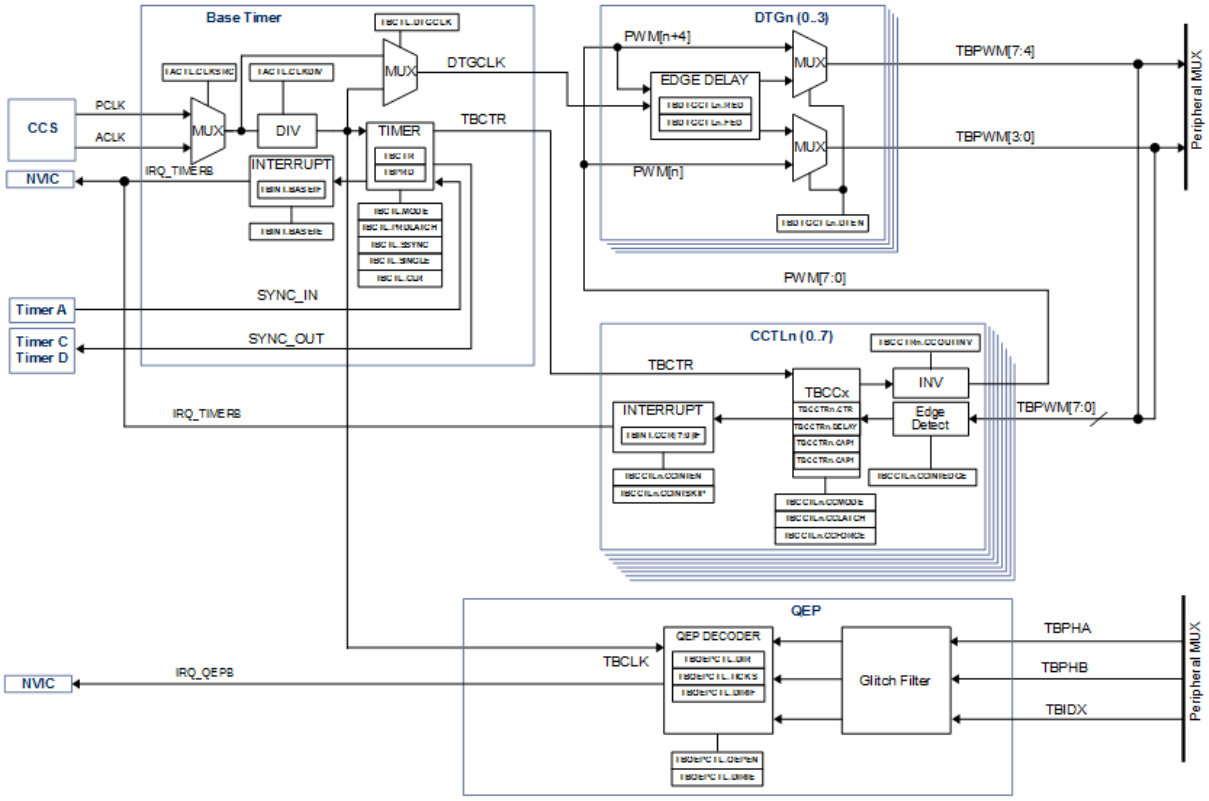
Figure 20-1 Timer B Simplified Block Diagram



The timer module is an APB bus client. There are 8 timer channel input/outputs to the digital peripheral MUX, in addition to three IOs for QEP. Each timer block has two interrupt signal outputs to the NVIC; one for timer functions and one for QEP functions. The user may select either PCLK or ACLK as the clock input for each timer module.

20.2 Timer B Block Diagram

Figure 20-2 Timer B Block Diagram



20.3 Features

The PWM Timer peripheral has the following features in the PAC55XX family of controllers.

General Features:

- Configurable input clock: PCLK or ACLK
- Up to 300MHz clock input for 3.33ns PWM edge resolution
- 3-bit input clock divider
- Latch timer period and all CCR values on command

Base Timer Features:

- Single-shot or auto-reload
- Base timer interrupts
- Timer synchronization
- Timer Modes:
  - Disabled
  - Up mode
  - Up/Down mode
  - Up/Down Asymmetric mode
- Timer Register Latching Options
  - Latch TAPRD when counter = 0
  - Latch TAPRD when counter = period
  - Latch TAPRD immediately
- Dead-time Generator Input Clock
  - DTG clock = PCLK
  - DTG clock = ACLK

CCR/PWM Features:

- PWM output or input capture
- 8 CCR units per timer
- CCR interrupts
- CCR interrupt skips
- SW force CCR interrupt
- CCR interrupt type:
  - Rising, falling, both
- CCR latch modes:
  - Compare/PWM mode: counter = 0, counter = period, immediate
  - Capture input: rising edge, falling edge, both

- Force compare event
- Invert CCR output
- CCR phase delay for phase shifted drive topologies
- ADC trigger outputs:
  - PWM rising or falling edge

Dead-Time Generator (DTG) Features:

- DTG enable or bypass
- 12-bit rising edge delay
- 12-bit falling edge delay

QEP Decoder Features:

- QEP decoder enabled
- Direction status
- Configurable interrupts:
  - Phase A rising edge
  - Phase B rising edge
  - Index event
  - Counter wrap
- 4 Different counting modes for best resolution, range and speed performance

20.4 Functional Description

20.4.1 Timer Clock Structure

The timer peripheral input clock can be selected as either PCLK or ACLK. The input clock may be selected by the **TBCTL.CLKSRC** register.

Each timer peripheral has a 4-bit divider that can be used to divide the selected input clock. The user may set this divider by the **TBCTL.CLKDIV** register. There are 16 settings between /1 and /128 for the input clock divider.

The base timer also supplies the clock for the Dead-Time Generators (DTGs) to allow for different range and resolution for dead-time. The timer may select the DTGCLK to be the timer clock before or after the clock divider. The DTG input clock may be selected by the **TBCTL.DTGCLK** register.

To use the timer clock before the input clock divider, set **TBCTL.DTGCLK** to 0b. To use the timer clock after the input clock divider, set **TBCTL.DTGCLK** to 1b.

20.4.2 Timer Counter

The base timer is a 16-bit timer that can count either up, or up then down to support both edge and center aligned and asymmetric PWM output types.

The timer period is stored in the **TBPRD** register. The current value of the timer counter is updated at every timer tick and is stored in the **TBCTR** register. The **TBCTR** register is a RW register, so it may be updated at any time and is changed immediately.

The **TBPRD** register and all of the **TBCCTRn** registers have shadow copies that are updated at a user-specified time. See the section below on Base Timer **TAPRD** Latching and CCR Timer Latching for more information.

When **TBCTL.MODE** = 00b, the timer is disabled. Even when disabled, all timer registers are accessible via the APB bus.

To enable the timer, set the **TBCTL.MODE** to 01b (up mode), 10b (up/down mode) or 11b (up/down asymmetric mode).

When **TBCTL.MODE** = 01b, the timer is configured in up mode. The timer will count from 0 to **TBPRD**. The counter is updated in **TBCTR** at every timer tick. If auto-reload is active (**TBCTL.SINGLE** = 0b), then the timer count will automatically count from **TBPRD** to 0 and continue counting up. If auto-reload is not active (**TBCTL.SINGLE** = 1b), then the timer will count from 0 to **TBPRD** and stop. When this happens, the timer will set the **TBCTL.MODE** to 00b (disabled).

When the timer is configured for up/down mode (**TBCTL.MODE** = 10b) or configured for up/down asymmetric mode (**TBCTL.MODE** = 11b), the timer will count from 0 to **TBPRD** and then back down to 0. If auto-reload is not active (**TBCTL.SINGLE** = 1b), the time timer will stop and set **TBCTL.MODE** to 00b (disabled). If auto-reload is active (**TBCTL.SINGLE** = 0b), then the timer count will then continue counting back up after it reaches 0.

20.4.3 Up Mode

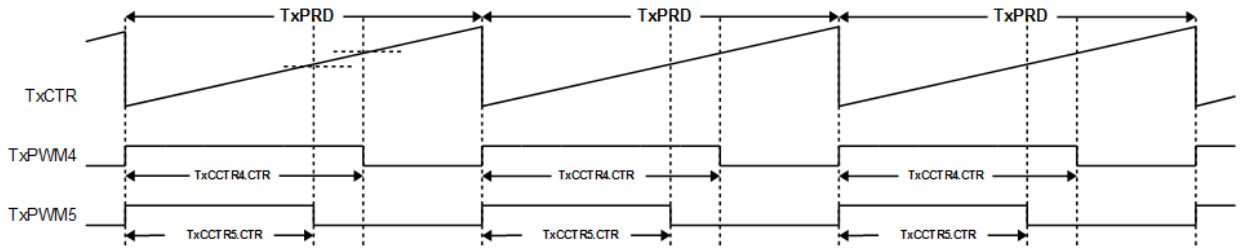
When the **TBCTL.MODE** is set to 01b (up mode) the timer will begin counting up from 0. The **TBCTR** will count to the value of **TBPRD** and then will reset to 0.

If the timer is configured for auto-reload mode (**TBCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it reaches 0.

The **TBCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate an edge aligned PWM as shown below.

Figure 20-3 Up Mode PWM Waveform



In this mode, when **TBCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TBCTR** register reaches the **TBCCTRn.CTR** threshold. When the **TBCTR** counts from **TBPRD** to 0, the PWM output is transitioned to high.

20.4.4 Up/Down Mode

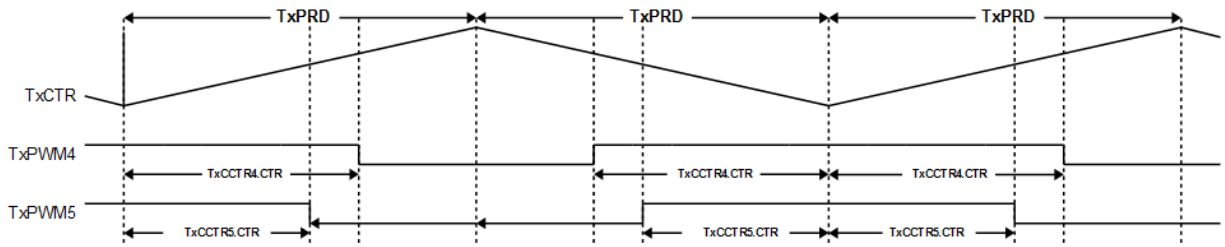
When the **TBCTL.MODE** is set to 10b (up/down mode) the timer will begin counting up from 0. The **TBCTR** will count to the value of **TBPRD** and then will begin counting back down to 0.

If the timer is configured for auto-reload mode (**TBCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it counts from 1 to 0.

The **TBCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate a center aligned PWM as shown below.

Figure 20-4 Up/Down Mode PWM Waveform



In this mode, when **TBCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TBCTR** register reaches the **TBCCTRn.CTR** threshold. When the **TBCTR** counts from **TBPRD** - 1 to **TBPRD**, it will begin counting down. When the **TBCTR** counts down to **TBCCTRn.CTR** the PWM output is transitioned to high.

20.4.5 Up/Down Asymmetric mode

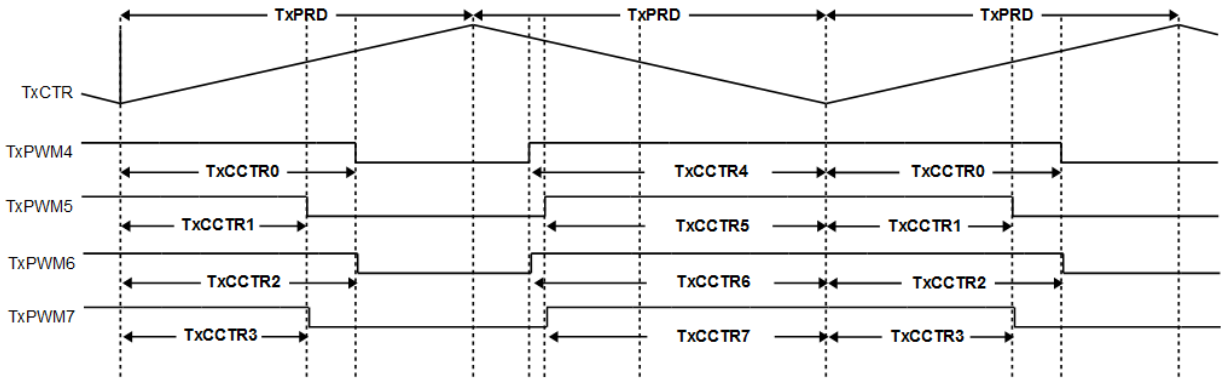
When the **TBCTL.MODE** is set to 11b (up/down asymmetric mode) the timer will begin counting up from 0. The **TBCTR** will count to the value of **TBPRD** and then will begin counting back down to 0.

If the timer is configured for auto-reload mode (**TBCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it counts from 1 to 0.

The **TBCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate a center aligned asymmetric PWM as shown below.

Figure 20-5 Up/Down Asymmetric Mode PWM Waveform



In this mode, when **TBCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TBCTR** register reaches the **TBCCTRn.CTR** threshold. When the **TBCTR** counts from **TBPRD** - 1 to **TBPRD**, it will begin counting down. When the **TBCTR** counts down to **TBCCTRn.CTR** the PWM output is transitioned to high.

In this mode, the CCR outputs are generated such that 2 **TBCCTRn.CTR** registers are used to generate a single PWM output, so that the center-aligned PWM can be generated with asymmetric on-time.

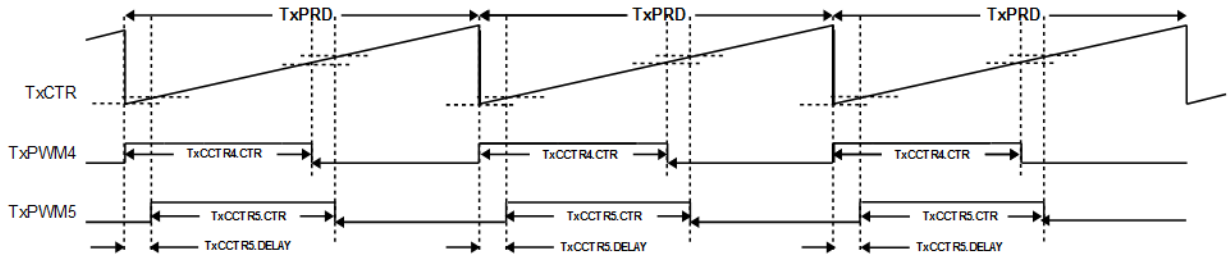
In this mode, the PWM outputs of the timer are generated as follows:

- TBPWM4 uses **TBCCTR0.CTR** for the first portion of the on time and **TBCCTR4.CTR** for the second portion of the on time
- TBPWM5 uses **TBCCTR1.CTR** for the first portion of the on time and **TBCCTR5.CTR** for the second portion of the on time
- TBPWM6 uses **TBCCTR2.CTR** for the first portion of the on time and **TBCCTR6.CTR** for the second portion of the on time
- TBPWM7 uses **TBCCTR3.CTR** for the first portion of the on time and **TBCCTR7.CTR** for the second portion of the on time

20.4.6 Up Mode with Phase Delay

When the timer is configured for up mode (**TBMODE** = 01b) the CCR allows a delay to be applied to the output PWM period, to support phase delay control topologies. If the value of **TBCCTRn.DELAY** > 0, then the CCR will delay this number of ticks before starting the PWM period (transitioning the output signal from high to low).

Figure 20-6 Up Mode with Phase Delay



20.4.7 Timer Synchronization

The timer peripheral allows synchronization between the timers in the PAC55XX. The timers may be synchronized as follows:

- Timer A (master), Timer B (slave)
- Timer A (master), Timer B (slave), Timer C (slave)

- Timer A (master), Timer B (slave), Timer C (slave), Timer D (slave)

The timer master's SYNC_OUT signals are connected to the downstream timer slave's SYNC_IN signal.

When configured, the timers can synchronize their clocks so they are sharing the same time base. In order to do this, the timers must be configured with the same clock source (**TBCTL.CLKSRC**) and clock divider (**TBCTL.CLKDIV**).

To configure the timers for synchronization, follow these steps:

- While the timer master is disabled (**TBCTL.MODE** = 00b), set the timer master **TxCTL.SSYNC** = 0b
- For each of the downstream timer slaves, while the timer is disabled **TBCTL.MODE** = 00b, set the **TBCTL.SSYNC** to 1b. Make sure that **TBCTL.CLKDIV** and **TBCTL.CLKSRC** for each of the timer slaves is the same as the timer master.
- Enable each of the slave timers by setting **TBCTL.MODE** to 01b (up), 10b (up/down) or 11b (up/down asymmetric). The slave timers will not start counting since the **TBCTL.SSYNC** is set to a 1b.
- Enable the timer master by setting **TxCTL.MODE** to 01b (up), 10b (up/down) or 11b (up/down asymmetric).

At this point, the master and all configured timer slaves will begin counting.

20.4.8 Base Timer TAPRD Latching

The timer peripherals have a shadow copy of the **TBPRD** register that is used for the counting operations. The latching of the data from the **TBPRD** register to the shadow register is controlled by the timer configuration.

There are several options for latching of this register into the shadow copy. This behavior is controlled by the **TBCTL.PRDLATCH** setting and depends on the setting of the **TBCTL.MODE** register as well.

20.4.8.1 TBPRD Latch When TACTR = 0

- If **TBCTL.MODE** = 01b (up mode) and **TBCTL.PRDLATCH** = 00b (**TBCTR** = 0), then the **TBPRD** register is copied into the shadow register when **TBCTR** counts from **TBPRD** to 0.
- If **TBCTL.MODE** = 10b (up/down mode) or if **TBCTL.MODE** = 11b (up/down asymmetric mode) and **TBCTL.PRDLATCH** = 00b (**TBCTR** = 0), then the **TBPRD** register is copied into the shadow register when the **TBCTR** counts from 1 to 0.

20.4.8.2 TBPRD Latch When TBCTR = TBPRD

- If **TBCTL.MODE** = 01b (up mode) or if **TBCTL.MODE** = 10b (up/down mode) or if **TBCTL.MODE** = 11b (up/down asymmetric mode) and **TBCTL.PRDLATCH** = 01b (**TBCTR** = **TBPRD**), then the **TBPRD** register is copied into the shadow register when **TBCTR** counts from **TBPRD** – 1 to **TBPRD**.

20.4.8.3 TBPRD Latch Immediate

- If **TBCTL.PRDLATCH** = 11b (immediate), then the **TBPRD** register is copied into the shadow register as soon as the **TBPRD** register is written.

20.4.9 CCR Timer Latching

When in compare mode, the CCR units support configurable latching of the **TBCCRn.CTR** values into a shadow register that is used for timer operation. The shadow register is used for generating the PWM output when the CCR is in compare mode.

This feature can be used to control when the new duty cycle is applied, during the PWM period.

When the CCR is in compare mode (**TBCCTLn.CCMODE** = 0b), the shadow register is latched as follows:

- If **TBCCTLn.CCLATCH** = 00b (**TBCTR** = 0)
  - If **TBCTL.MODE** = 01b (up mode), **TBCCTRn.CTR** is copied into the shadow register when **TBCTR** counts from **TBPRD** to 0. If **TBCTL.MODE** = 10b (up/down mode) or 11b (up/down asymmetric mode), **TBCCTRn.CTR** is copied into the shadow register when **TBCTR** counts from 1 to 0.
- If **TBCCTLn.CCLATCH** = 01b (**TBCTR = TBPRD**), then the **TBCCTRn.CTR** is copied into the shadow register when **TBCTR** counts from **TBPRD** – 1 to **TBPRD**.
- If **TBCCTLn.CCLATCH** = 10b (latch immediate), then the **TBCCTRn.CTR** is copied into the shadow register as soon as it is written.

When the CCR is in capture mode (**TBCCTLn.CCMODE** = 1b), the configuration of **TBCCTLn.CCLATCH** controls when the **TBCTR** is copied into the **TBCCTRn.CTR** register.

- If **TBCCTLn.CCLATCH** = 00b (rising edge), then the value of **TBCTR** is copied into the **TBCCTLn.CTR** register upon a rising edge in the input signal.
- If **TBCCTLn.CCLATCH** = 01b (falling edge), then the value of the **TBCTR** is copied into the **TBCCTLn.CTR** register upon a falling edge in the input signal.
- If **TBCCTLn.CCLATCH** = 10b (both), then the value of the **TBCTR** is copied into the **TBCCTLn.CTR** register on both a rising and falling edge of the input signal.

20.4.10 Timer Whole Latching

It is sometimes convenient to latch both the **TBPRD** (period) and all **TBCCTRn.CTR** (duty cycle) registers into the shadow registers at one time.

If the **TBCTL.LATCH** bit is written to a 1b, then all the **TBPRD** and all **TBCCTRn.CTR** registers will be copied into the shadow registers on the same clock cycle.

The **TBCTL.LATCH** bit is self-clearing, and will always be read as a 0b.

20.4.11 Inverting CCR PWM Output

When the CCR is in compare mode (**TBCCTRn.CCMODE** = 0b), the output of the CCR (the input to the DTG) may be inverted. To invert this output, set the **TBCCTRn.CCOUTINV** = 1b.

This mode is useful for some control topologies, especially full-bridge.

20.4.12 Base Timer Interrupts

The base timer may be configured to generate an interrupt to the NVIC.

If **TBCTL.MODE** = 01b (up mode) and **TBCTR** counts from **TBPRD** to 0 then the **TBINT.BASEIF** bit is set to 1b. If the **TBCTL.BASEIE** interrupt enable bit is set to a 1b, then the TIMER_IRQ signal to the NVIC is asserted.

If **TBCTL.MODE** = 10b (up/down mode) or 11b (up/down asymmetric mode) and **TBCTR** counts from 1 to 0, then the **TBINT.BASEIF** bit is set to 1b. If the **TBCTL.BASEIE** interrupt enable bit is set to a 1b, then the TIMERB_IRQ signal to the NVIC is asserted.

The **TBINT.BASEIF** may be cleared by writing a 1b to it.

20.4.13 CCR Compare Interrupts

When **TxCCTLn.CCMODE** = 0b (compare mode), the CCR unit may be configured to generate an interrupt to the NVIC.

When the PWM edge transitions, then the CCR interrupt flag in the **TxINT** register is set as follows:

- If **TxCCTLn.CCINTEDGE** = 00b (rising edge) and a rising edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and TIMER_IRQ to the NVIC is asserted.
- If **TxCCTLn. CCINTEDGE** = 01b (falling edge) and a falling edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and TIMER_IRQ to the NVIC is asserted. **TxINT.CCRnIF** may be cleared by writing it to a 1b.
- If **TxCCTLn. CCINTEDGE** = 10b (rising or falling edge) and a rising or falling edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and

TIMER_IRQ to the NVIC is asserted. **TxINT.CCRnIF** may be cleared by writing it to a 1b.

TxINT.CCRnIF may be cleared by writing it to a 1b.

20.4.14 CCR Capture Interrupts

When **TxCCTLn.CCMODE** = 1b (capture mode), the CCR unit may be configured to generate an interrupt to the NVIC based on the input signal.

When the PWM edge transitions, then the CCR interrupt flag in the **TxINT** register is set as follows:

- If **TxCCTLn.CCINTEDGE** = 00b (rising edge) and a rising edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is set.
- If **TxCCTLn.CCINTEDGE** = 01b (falling edge) and a falling edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is set.
- If **TxCCTLn.CCINTEDGE** = 10b (rising or falling edge) and a rising or falling edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is.

If the **TxINT.CCRnIF** interrupt flag is set and the corresponding **TxCCTLn.CCINTEN** bit is set, then the TIMERA_IRQ signal to the NVIC will be asserted.

The **TxINT.CCRnIF** interrupt flag can be cleared by writing it to a 1b.

20.4.15 Timer IRQ signal

The timer unit has one IRQ output signal to the NVIC (TIMERB_IRQ).

This signal is asserted when any of the following conditions are true:

- **TBINT.BASEIF** = 1b and **TBCTL.BASEIE** = 1b
- **TBINT.CCR0IF** = 1b and **TBCCTL0.CCINTEN** = 1b
- **TBINT.CCR1IF** = 1b and **TBCCTL1.CCINTEN** = 1b
- **TBINT.CCR2IF** = 1b and **TBCCTL2.CCINTEN** = 1b
- **TBINT.CCR3IF** = 1b and **TBCCTL3.CCINTEN** = 1b
- **TBINT.CCR4IF** = 1b and **TBCCTL4.CCINTEN** = 1b
- **TBINT.CCR5IF** = 1b and **TBCCTL5.CCINTEN** = 1b
- **TBINT.CCR6IF** = 1b and **TBCCTL6.CCINTEN** = 1b
- **TBINT.CCR7IF** = 1b and **TBCCTL7.CCINTEN** = 1b

The TIMERB_IRQ signal is de-asserted when all of the above conditions are false.

20.4.16 Skipping CCR Interrupts

Sometimes it is useful to not generate a CCR interrupt to the NVIC every time the **TBCTR** counts to the **TBCCTRn.CTR** value. For example, in a control application when the PWM frequency is fast, but you only need a MCU interrupt every 5 PWM periods.

Each CCR in the timer has the ability to skip interrupts to the NVIC to allow this capability.

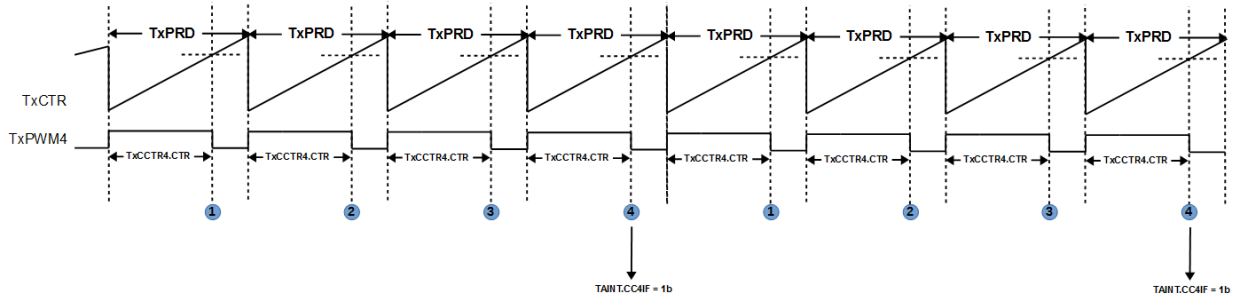
The CCR will generate interrupts to the NVIC according to the **TBCCTL.CCINTSKIP** field as follows:

- If **TBCCTLn.CCINTSKIP** = 0000b, then the CCR will generate an interrupt each time
- If **TBCCTLn.CCINTSKIP** = 0001b, then the CCR will skip 1 CCR cycle before generating an interrupt to the NVIC.
- If **TBCCTLn.CCINTSKIP** = 0010b, then the CCR will skip 2 CCR cycles before generating an interrupt to the NVIC.
- If **TBCCTLn.CCINTSKIP** = 0011b, then the CCR will skip 3 CCR cycles before generating an interrupt to the NVIC.
- …

- If **TBCCTLn.CCINTSKIP** = 1110b, then the CCR will skip 14 CCR cycles before generating an interrupt to the NVIC.
- If **TBCCTLn.CCINTSKIP** = 1111b, then the CCR will skip 15 CCR cycles before generating an interrupt to the NVIC.

The diagram below shows an example of a CCR4 with interrupt skipping configured with a value of 4 (**TBCCTL.CCINTSKIP** = 4).

Figure 20-7 CCR Interrupt Skipping Diagram



20.4.17 Timer ADC Triggers

Each timer CCR unit may be configured to be an ADC trigger, to allow the DTSE to automatically begin conversion sequences.

For more information on this feature, see the section on the ADC and DTSE in this user guide.

20.4.18 Dead-Time Generators (DTG)

Each timer has 4 DTG units. Each DTG unit is capable of generating a pair of complementary signals that can be configured with dead-time to drive an inverter for half-bridge topologies.

The input clock to the DTG can be configured to be the timer clock before or after the **TBCTL.CLKDIV** input clock divider. If **TBCTL.DTGCLK** is 0b, then the DTGCLK is the clock before the **TBCTL.CLKDIV** clock divider. If **TBCTL.DTGCLK** is 1b, then the DTGCLK is the clock after the **TBCTL.CLKDIV** clock divider.

When **TBDTGCTLn.DTEN** = 0b, the DTG is disabled (bypassed). In this mode, the timer signals have the following behavior (see the simplified block diagram above):

DTG0:

- CCR0 output is connected to TBPWM0
- CCR4 output is connected to TBPWM4

DTG1:

- CCR1 output is connected to TBPWM1
- CCR5 output is connected to TBPWM5

DTG2:

- CCR2 output is connected to TBPWM2
- CCR6 output is connected to TBPWM6

DTG3:

- CCR3 output is connected to TBPWM3
- CCR7 output is connected to TBPWM7

When **TBDTGCTL0.DTEN** = 1b, the DTG is enabled. In this mode, only one CCR output is used to generate the two complementary outputs. In this mode, the timer signals have the following behavior:

DTG0:

- CCR0 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR4 output is used to generate the two complementary output signals: TBPWM0 and TBPWM4.
- TBPWM0 becomes the low-side complementary signal.

- TBPWM4 becomes the high-side complementary signal.
- The **TBDTGCTL0.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, to when the low-side signal rises).
- The **TBDTGCTL0.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, to when the high-side signal rises).

DTG1:

- CCR1 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR5 output is used to generate the two complementary output signals: TBPWM1 and TBPWM5.
- TBPWM1 becomes the low-side complementary signal.
- TBPWM5 becomes the high-side complementary signal.
- The **TBDTGCTL1.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TBDTGCTL1.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

DTG2:

- CCR2 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR6 output is used to generate the two complementary output signals: TBPWM2 and TAPWM6.
- TAPWM2 becomes the low-side complementary signal.
- TAPWM6 becomes the high-side complementary signal.
- The **TADTGCTL2.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TADTGCTL2.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

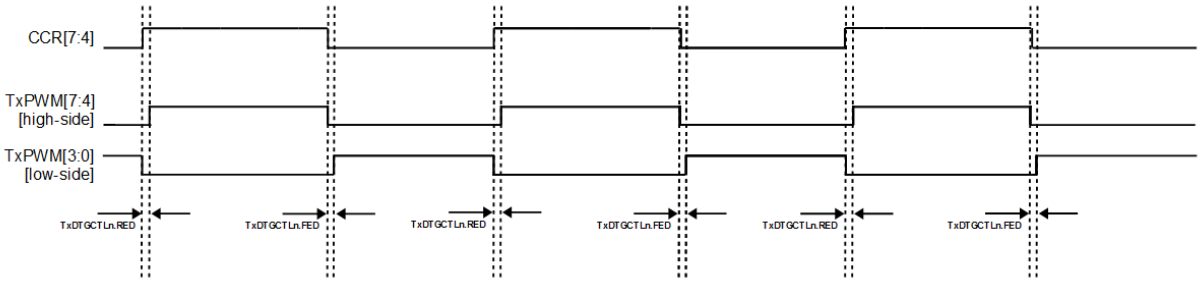
DTG3:

- CCR3 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR7 output is used to generate the two complementary output signals: TAPWM3 and TBPWM4.
- TBPWM3 becomes the low-side complementary signal.
- TBPWM7 becomes the high-side complementary signal.
- The **TBDTGCTL3.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TBDTGCTL0.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

The PWM waveform below shows an example of the dead-time applied to the signals through the DTG units.

20.4.19 Timer B DTG Output

Figure 20-8 Timer B DTG Output



20.5 QEP

20.5.1 Overview

Each PAC55XX PWM Timer contains a QEP decoder that can be used to determine the position, speed, and direction of a motor with a QEP encoder.

The inputs to the QEP peripheral are TxQEPPHA, TxQEPPHB and TxQEPIDX (phase A, phase B and index). These three inputs have a glitch filter applied to them, to filter out noise and are used for calculation of the motor position, speed, and direction.

20.5.2 Configuration

The QEP peripheral can be enabled by setting the **TxQEPCTL.QEPEN** bit to a 1b.[18] The input clock to the QEP peripheral may be configured between /1 and /128 using the **TxQEPCTL.CLKDIV** field.

If the **TxQEPCTL.CNTAB** bit is 0b, only phase A edges are processed. If the **TxQEPCTL.CNTAB** bit is set to a 1b, then both phase A and phase B edges are processed. If the **TxQEPCTL.CNTEDGE** bit is set to 0b, then only the rising edges are processed. If the **TxQEPCTL.CNTEDGE** bit is set to a 1b, then both rising and falling edges are processed.

If the direction is determined to be clockwise (CW), then the **TxQEPCTL.DIR** bit is set to 0b. If the direction is determined to be counterclockwise (CCW), then the **TxQEPCTL.DIR** bit is set to 1b. The direction will get updated only on edges that are processed.

When an edge is processed, the QEP peripheral will first determine the direction and then increment or decrement the counter value in **TxQEPCTR.CTR** accordingly. **TxQEPCTR.CTR** may be written at any time.

If an edge is processed while the motor is turning clockwise, the counter value in **TxQEPCTR.CTR** is incremented. If the value of **TxQEPCTR.CTR** was **TxQEPMAX.CTRMAX** during this event, then **TxQEPCTR.CTR** is set to 0.

If an edge is processed while the motor is turning counterclockwise, the counter value in **TxQEPCTR.CTR** is decremented. If the value of **TxQEPCTR.CTR** was 0 during this event, then **TxQEPCTR.CTR** is set to **TxQEPMAX.CTRMAX**.

[18] Before enabling the QEP peripheral, be sure to set up the Digital Peripheral MUX, so that no false edges are detected by the QEP state machine.

20.5.3 Index Reset

If the **TxQEPCTL.IDXRST** bit is set to a 1b, then an index event clears the **TxQEPCTR.CTR** counter to 0. If the motor is turning in the clockwise direction during this event, **TxQEPCTR.CTR** is set to 0. If the motor is turning in the counterclockwise direction during this event the **TxQEPCTR.CTR** is set to 0 (if not coincident with the phase edge) or **TxQEPMAX.CTRMAX** (if coincident with the phase edge).

If this bit is set to a 0b, then an index event does not reset the **TAQEPCTL.TICKS** counter.

20.5.4 Interrupts

If a change to the direction is detected (0 to 1 or 1 to 0), then the **TxQEPIF.DIRIF** bit is set to a 1b. If the **TxQEPIER.DIRIE** bit is set to 1b, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.DIRIF** bit can be cleared by writing it to 1b.

When a rising edge on phase A is detected the **TxQEPIF.PHAIF** bit is set to 1b. If the **TxQEPIER.PHAIE** bit is set and the **TxQEPIF.PHAIF** bit is set, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.PHAIF** bit can be cleared by writing a 1b to it.

When a rising edge on phase B is detected the **TxQEPIF.PHBIF** bit is set. If the **TxQEPIER.PHBIE** bit is set and the **TxQEPIF.PHBIF** bit is set, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.PHBIF** bit can be cleared by writing a 1b to it.

If an overflow or underflow in the **TxQEPCTR.CTR** is detected, the **TxQEPIF.WRIF** bit is set (counter wrap interrupt flag). If the **TxQEPIER.WRIE** bit is set and the **TxQEPIF.WRIF** bits are set, then the QEPx_IRQ signal is asserted to the NVIC.

If an index event is detected, the **TxQEPIF.IDXEVIF** bit is set. If the **TxQEPIER.IDXEVIE** bit is set and the **TxQEPIF.IDXEVIF** bit is set, then the QEPx_IRQ signal to the NVIC is asserted. The **TxQEPIF.IDXEVIF** bit can be cleared by writing a 1b to it.

20.6 Peripheral IO Mapping

The Timer B peripheral signal inputs and outputs (CCR, QEP) are connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 20-1 Timer B Peripheral IO Mapping

TIMER SIGNAL	IO PIN
TBPWM0	PB3, PB7, PC0, PD0
TBPWM1	PB2, PB6, PC1, PD1
TBPWM2	PB1, PB5, PC2, PD2
TBPWM3	PB0, PB4, PC3, PD3
TBPWM4	PC4, PD4
TBPWM5	PC5, PD5
TBPWM6	PC6, PD6
TBPWM7	PC7, PD7
TBQEPPHA	PC1, PD5, PE1, PF1
TBQEPIDX	PC0, PD4, PE0, PF0
TBQEPPHB	PC2, PD6, PE2, PF2

For more information on how to use the Digital Peripheral MUX to connect peripheral signals to IO, see GPIO and DPM.

20.7 Timer B Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
TBCTL	4007 0000h	Timer B control	RW	0000 0000h
TBINT	4007 0004h	Timer B interrupt control	RW	0000 0000h
TBPRD	4007 0008h	Timer B period	RW	0000 0000h
TBCTR	4007 000Ch	Timer B counter	RW	0000 0000h
TBCCTL0	4007 0100h	Timer B CC control 0	RW	0000 0000h
TBCCTR0	4007 0104h	Timer B CC counter 0	RW	0000 0000h
TBCCTL1	4007 0108h	Timer B CC control 1	RW	0000 0000h
TBCCTR1	4007 010Ch	Timer B CC counter 1	RW	0000 0000h
TBCCTL2	4007 0110h	Timer B CC control 2	RW	0000 0000h
TBCCTR2	4007 0114h	Timer B CC counter 2	RW	0000 0000h
TBCCTL3	4007 0118h	Timer B CC control 3	RW	0000 0000h
TBCCTR3	4007 011Ch	Timer B CC counter 3	RW	0000 0000h
TBCCTL4	4007 0120h	Timer B CC control 4	RW	0000 0000h
TBCCTR4	4007 0124h	Timer B CC counter 4	RW	0000 0000h
TBCCTL5	4007 0128h	Timer B CC control 5	RW	0000 0000h
TBCCTR5	4007 012Ch	Timer B CC counter 5	RW	0000 0000h
TBCCTL6	4007 0130h	Timer B CC control 6	RW	0000 0000h
TBCCTR6	4007 0134h	Timer B CC counter 6	RW	0000 0000h
TBCCTL7	4007 0138h	Timer B CC control 7	RW	0000 0000h
TBCCTR7	4007 013Ch	Timer B CC counter 7	RW	0000 0000h
TBDTGCTL0	4007 0200h	Timer B DTG control 0	RW	0000 0000h
TBDTGCTL1	4007 0204h	Timer B DTG control 1	RW	0000 0000h
TBDTGCTL2	4007 0208h	Timer B DTG control 2	RW	0000 0000h
TBDTGCTL3	4007 020Ch	Timer B DTG control 3	RW	0000 0000h
TBQEPCTL	4007 0300h	Timer B QEP control	RW	0000 0000h
TBQEPCTR	4007 0304h	Timer B QEP counter	RW	0000 0000h
TBQEPMAX	4007 0308h	Timer B QEP max counter value	RW	0000 FFFFh
TBQEPIER	4007 030Ch	Timer B QEP interrupt enable	RW	0000 0000h
TBQEPIFR	4007 0310h	Timer B QEP interrupt flag	RW	0000 0000h

20.8 Register Detail

20.8.1 TBCTL

Register 20-1 TBCTL (Timer B Control, 4007 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:14	**Reserved**	RO	0	Reserved
13	**BASEIE**	RW	0	Base timer interrupt enable: 0b: not enabled 1b: enabled
12	**CLR**	RW	0	Timer Clear: 0b: do not clear 1b: clear timer counter
11	**LATCH**	RW	0	When written to 1b, this will latch the **TBPRD** and all **TBCCTRn** registers on the same clock cycle. Writing this bit to 0b has no effect. This is a self-clearing bit.
10	**DTGCLK**	RW	0	DTG Clock Source: 0b: Before input clock divider 1b: After input clock divider
9	**CLKSRC**	RW	0	Timer Clock Source: 0b: PCLK 1b: ACLK
8:6	**CLKDIV**	RW	0	Timer Input Clock Divider: 000b: /1 001b: /2 010b: /4 011b: /8 100b: /16 101b: /32 110b: /64 111b: /128
5	**SINGLE**	RW	0	Single Shot Timer: 0b: disabled (auto-reload) 1b: enabled (single shot timer)
4	**SSYNC**	RW	0	Timer Slave Synchronization: 0b: disabled 1b: enabled
3:2	**PRDLATCH**	RW	0	Timer Period Latch Mode: 00b: Latch **TBPRD** when **TBCTR** = 0 01b: Latch **TBPRD** when **TBCTR** = **TBPRD** 10b: Latch **TBPRD** immediately upon register write 11b: reserved
1:0	**MODE**	RW	0	Timer Mode: 00b: disabled 01b: up mode 10b: up/down mode 11b: asymmetric mode

20.8.2 TBINT

Register 20-2 TBINT (Timer B Interrupt Control, 4007 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:9	**Reserved**	RO	0	Reserved
8	**BASEIF**	W1C	0	Base timer interrupt flag: 0b: no interrupt flag 1b: interrupt flag
7	**CCR7IF**	W1C	0	CCR7 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
6	**CCR6IF**	W1C	0	CCR6 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
5	**CCR5IF**	W1C	0	CCR5 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
4	**CCR4IF**	W1C	0	CCR4 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
3	**CCR3IF**	W1C	0	CCR3 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
2	**CCR2IF**	W1C	0	CCR2 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
1	**CCR1IF**	W1C	0	CCR1 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
0	**CCR0IF**	W1C	0	CCR0 interrupt flag: 0b: no interrupt flag 1b: interrupt flag

20.8.3 TBPRD

Register 20-3 TBPRD (Timer A Period, 4007 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**PERIOD**	RW	0	Timer period value

20.8.4 TBCTR

Register 20-4 TBCTR (Timer B Counter, 4007 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**COUNTER**	RW	0	Timer counter

20.8.5 TBCCTL0

Register 20-5 TBCCTL0 (Timer B CCR Control 0, 4007 0100h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5		RW	0	CCR register latch mode. If **TBCCTL0.CCMODE** = 0b (compare): 00b: Latch **TBCCTR0** registers when **TBCTR** = 0 01b: Latch **TBCCTR0** registers when **TBCTR** = **TBPRD** 10b: Latch **TBCCTR0** registers immediately 11b: Reserved If **TBCCTL0.CCMODE** = 1b (capture): 00b: Latch **TACTR** into **TACCTR0.CTR** on rising edge 01b: Latch **TACTR** into **TACCTR0.CTR** on falling edge 10b: Latch **TACTR** into **TACCTR0.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TBCCTL0.CCMODE** = 1b (capture), set **TBINT.CCR0IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

20.8.6 TBCCTR0

Register 20-6 TBCCTR0 (Timer B CCR Counter 0, 4007 0104h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TBCCTL0.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TBCCTRL0.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TBCCTL0.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

20.8.7 TBCCTL1

Register 20-7 TBCCTL1 (Timer A CCR Control 1, 4007 0108h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TBCCTL1.CCMODE** = 0b (compare): 00b: Latch **TBCCTR1** registers when **TBCTR** = 0 01b: Latch **TBCCTR1** registers when **TBCTR** = **TBPRD** 10b: Latch **TBCCTR1** registers immediately 11b: Reserved If **TBCCTL1.CCMODE** = 1b (capture): 00b: Latch **TBCTR** into **TBCCTR1.CTR** on rising edge 01b: Latch **TBCTR** into **TBCCTR1.CTR** on falling edge 10b: Latch **TBCTR** into **TBCCTR1.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TBCCTL1.CCMODE** = 1b (capture), set **TBINT.CCR1IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

20.8.8 TBCCTR1

Register 20-8 TBCCTR1 (Timer B CCR Counter 1, 4007 010Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TBCCTL1.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TBCCTRL1.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TBCCTL1.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

20.8.9 TBCCTL2

Register 20-9 TBCCTL2 (Timer B CCR Control 2, 4007 0110h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TBCCTL2.CCMODE** = 0b (compare): 00b: Latch **TBCCTR2** registers when **TBCTR** = 0 01b: Latch **TBCCTR2** registers when **TBCTR** = **TBPRD** 10b: Latch **TBCCTR2** registers immediately 11b: Reserved If **TBCCTL2.CCMODE** = 1b (capture): 00b: Latch **TBCTR** into **TBCCTR2.CTR** on rising edge 01b: Latch **TBCTR** into **TBCCTR2.CTR** on falling edge 10b: Latch **TBCTR** into **TBCCTR2.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TBCCTL2.CCMODE** = 1b (capture), set **TBINT.CCR2IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

20.8.10 TBCCTR2

Register 20-10 TBCCTR2 (Timer B CCR Counter 2, 4007 0114h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TBCCTL2.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TBCCTRL2.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TBCCTL2.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

20.8.11 TBCCTL3

Register 20-11 TBCCTL3 (Timer B CCR Control 3, 4007 0118h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**R**	RW	0	CCR register latch mode. If **TBCCTL3.CCMODE** = 0b (compare): 00b: Latch **TBCCTR3** registers when **TBCTR** = 0 01b: Latch **TBCCTR3** registers when **TBCTR** = **TBPRD** 10b: Latch **TBCCTR3** registers immediately 11b: Reserved If **TBCCTL3.CCMODE** = 1b (capture): 00b: Latch **TBCTR** into **TBCCTR3.CTR** on rising edge 01b: Latch **TBCTR** into **TBCCTR3.CTR** on falling edge 10b: Latch **TBCTR** into **TBCCTR3.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TBCCTL3.CCMODE** = 1b (capture), set **TBINT.CCR3IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

20.8.12 TBCCTR3

Register 20-12 TBCCTR3 (Timer B CCR Counter 3, 4007 011Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TBCCTL3.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TBCCTRL3.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TBCCTL3.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

20.8.13 TBCCTL4

Register 20-13 TBCCTL4 (Timer B CCR Control 4, 4007 0120h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TBCCTL4.CCMODE** = 0b (compare): 00b: Latch **TBCCTR4** registers when **TBCTR** = 0 01b: Latch **TBCCTR4** registers when **TBCTR** = **TBPRD** 10b: Latch **TBCCTR4** registers immediately 11b: Reserved If **TBCCTL4.CCMODE** = 1b (capture): 00b: Latch **TBCTR** into **TBCCTR4.CTR** on rising edge 01b: Latch **TBCTR** into **TBCCTR4.CTR** on falling edge 10b: Latch **TBCTR** into **TBCCTR4.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TBCCTL4.CCMODE** = 1b (capture), set **TBINT.CCR4IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

20.8.14 TBCCTR4

Register 20-14 TBCCTR4 (Timer B CCR Counter 4, 4007 0124h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TBCCTL4.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TBCCTRL4.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TBCCTL4.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

20.8.15 TBCCTL5

Register 20-15 TBCCTL5 (Timer B CCR Control 5, 4007 0128h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TBCCTL5.CCMODE** = 0b (compare): 00b: Latch **TBCCTR5** registers when **TBCTR** = 0 01b: Latch **TBCCTR5** registers when **TBCTR** = **TBPRD** 10b: Latch **TBCCTR5** registers immediately 11b: Reserved If **TBCCTL5.CCMODE** = 1b (capture): 00b: Latch **TBCTR** into **TBCCTR5.CTR** on rising edge 01b: Latch **TBCTR** into **TBCCTR5.CTR** on falling edge 10b: Latch **TBCTR** into **TBCCTR5.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TBCCTL5.CCMODE** = 1b (capture), set **TBINT.CCR5IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

20.8.16 TBCCTR5

Register 20-16 TBCCTR5 (Timer B CCR Counter 5, 4007 012Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TBCCTL5.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TBCCTRL5.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TBCCTL5.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

20.8.17 TBCCTL6

Register 20-17 TBCCTL6 (Timer B CCR Control 6, 4007 0130h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TBCCTL6.CCMODE** = 0b (compare): 00b: Latch **TBCCTR6** registers when **TBCTR** = 0 01b: Latch **TBCCTR6** registers when **TBCTR** = **TBPRD** 10b: Latch **TBCCTR6** registers immediately 11b: Reserved If **TBCCTL6.CCMODE** = 1b (capture): 00b: Latch **TBCTR** into **TBCCTR6.CTR** on rising edge 01b: Latch **TBCTR** into **TBCCTR6.CTR** on falling edge 10b: Latch **TBCTR** into **TBCCTR6.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TBCCTL6.CCMODE** = 1b (capture), set **TBINT.CCR6IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

20.8.18 TACCTR6

Register 20-18 TBCCTR6 (Timer B CCR Counter 6, 4007 0134h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TBCCTL6.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TBCCTRL6.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TBCCTL6.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

20.8.19 TBCCTL7

Register 20-19 TBCCTL7 (Timer B CCR Control 7, 4007 0138h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TBCCTL7.CCMODE** = 0b (compare): 00b: Latch **TBCCTR7** registers when **TBCTR** = 0 01b: Latch **TBCCTR7** registers when **TBCTR** = **TBPRD** 10b: Latch **TBCCTR7** registers immediately 11b: Reserved If **TBCCTL7.CCMODE** = 1b (capture): 00b: Latch **TBCTR** into **TBCCTR7.CTR** on rising edge 01b: Latch **TBCTR** into **TBCCTR7.CTR** on falling edge 10b: Latch **TBCTR** into **TBCCTR7.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TBCCTL7.CCMODE** = 1b (capture), set **TBINT.CCR7IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

20.8.20 TBCCTR7

Register 20-20 TBCCTR7 (Timer B CCR Counter 7, 4007 013Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TBCCTL7.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TBCCTRL7.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TBCCTL7.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

20.8.21 TBDTGCTL0

Register 20-21 TBDTGCTL0 (Timer B Dead-Time Generator Control 0, 4007 0200h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG0.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG0.

20.8.22 TBDTGCTL1

Register 20-22 TBDTGCTL1 (Timer B Dead-Time Generator Control 1, 4007 0204h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG1.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG1.

20.8.23 TBDTGCTL2

Register 20-23 TBDTGCTL2 (Timer B Dead-Time Generator Control 2, 4007 0208h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG2.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG2.

20.8.24 TBDTGCTL3

Register 20-24 TBDTGCTL3 (Timer B Dead-Time Generator Control 3, 4007 020Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG3.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG3.

20.8.25 TBQEPCTL

Register 20-25 TBQEPCTL (Timer B QEP Control, 4007 0300h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:11	**Reserved**	RO	0	Reserved
10:8	**CLKDIV**	RW	0	QEP Input Clock Divider: 000b: /1 001b: /2 010b: /4 011b: /8 100b: /16 101b: /32 110b: /64 111b: /128
7:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXRST**	RW	0	Index reset mode. Used for resetting counter. 0b: No reset 1b: Index edge reset. When direction is CW, reset counter to 0 when index rising edge event occurs. When direction is CCW, reset when index falling edge occurs (reset to 0 if not coincident with phase edge; reset to Tx**QEPMAX** if coincident with phase edge).
3	**DIR**	RO	0	Motor direction: 0b: CW (clockwise) 1b: CCW (counter-clockwise)
2	**CNTEDGE**	RW	0	Count on edge: 0b: Rising edge only 1b: Rising and falling edge
1	**CNTAB**	RW	0	Count on A/B: 0b: Count phase A only 1b: Count phase A and phase B
0	**QEPEN**	RW	0	QEP peripheral enable: 0b: disabled 1b: enabled

20.8.26 TBQEPCTR

Register 20-26 TBQEPCTR (Timer B QEP Counter, 4007 0304h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**CTR**	RW	0	Counter Value

20.8.27 TBQEPMAX

Register 20-27 TBQEPMAX (Timer B QEP Maximum Counter, 4007 0308h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**CTRMAX**	RW	0	Maximum Counter Value.

20.8.28 TBQEPIER

Register 20-28 TBQEPIER (Timer B QEP Interrupt Enable, 4007 030Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXEVIE**	RW	0	Index event interrupt enable: 0b: disabled 1b: enabled
3	**WRIE**	RW	0	Counter wrap interrupt enable: 0b: disabled 1b: enabled
2	**PHBIE**	RW	0	Phase B rising edge interrupt enable: 0b: disabled 1b: enabled
1	**PHAIE**	RW	0	Phase A rising edge interrupt enable: 0b: disabled 1b: enabled
0	**DIRIE**	RW	0	Direction change interrupt enable: 0b: disabled 1b: enabled

20.8.29 TBQEPIF

Register 20-29 TBQEPIF (Timer B QEP Interrupt Flag, 4007 0310h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXEVIF**	W1C	0	Index event interrupt flag: 0b: disabled 1b: enabled
3	**WRIF**	W1C	0	Counter wrap interrupt flag: 0b: disabled 1b: enabled
2	**PHBIF**	W1C	0	Phase B rising edge interrupt flag: 0b: disabled 1b: enabled
1	**PHAIF**	W1C	0	Phase A rising edge interrupt flag: 0b: disabled 1b: enabled
0	**DIRIF**	W1C	0	Direction change interrupt flag: 0b: disabled 1b: enabled

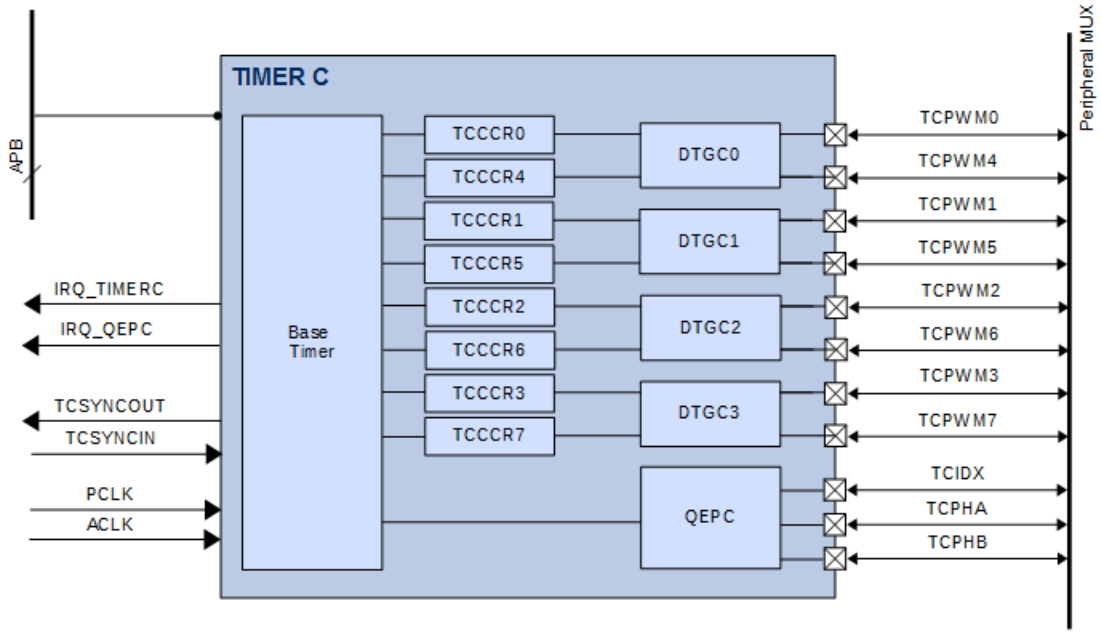
21 PWM TIMER C

21.1 Overview

All devices in the PAC55XX family of controllers have a Timer C peripheral. This peripheral is a 16-bit timer that allows support for 8 Capture and Compare Units (CCR) capable of PWM generation; capture input processing and a QEP decoder for various control applications.

Below is a simplified block diagram of the Timer peripheral.

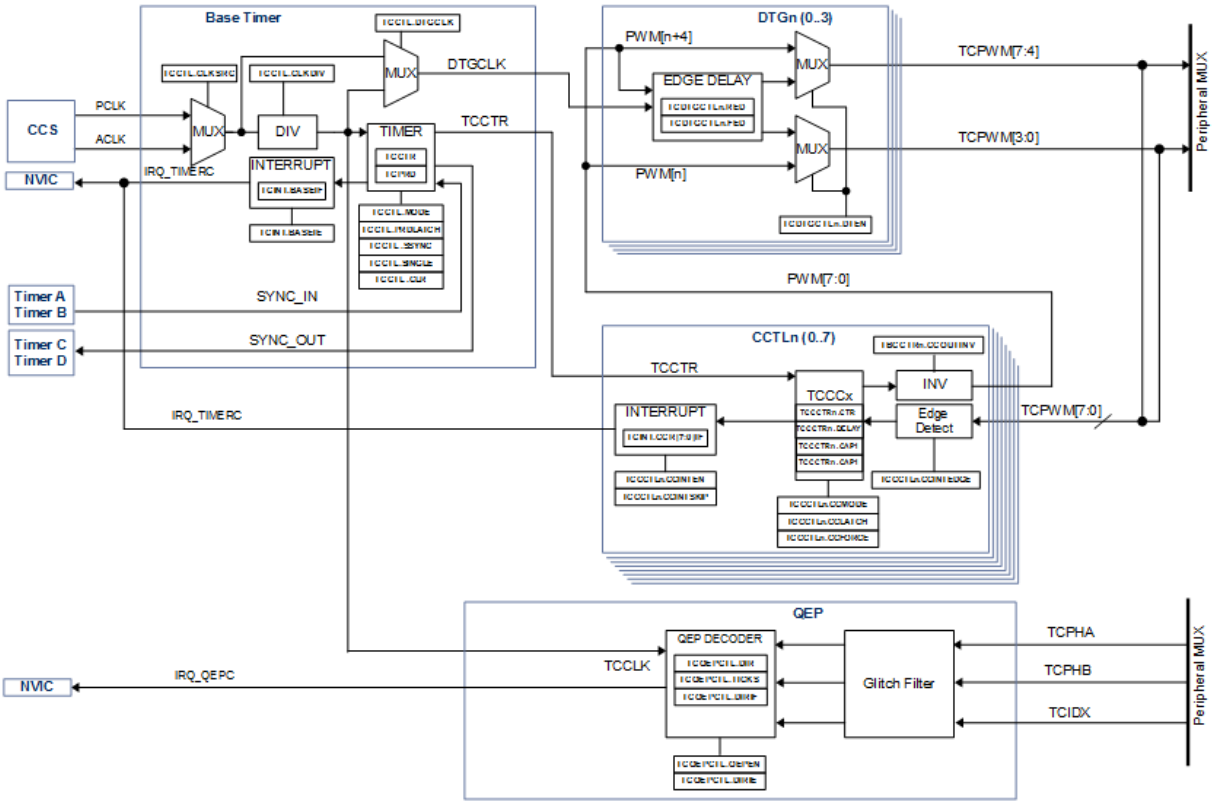
Figure 21-1 Timer C Simplified Block Diagram



The timer module is an APB bus client. There are 8 timer channel input/outputs to the digital peripheral MUX, in addition to three IOs for QEP. Each timer block has two interrupt signal outputs to the NVIC; one for timer functions and one for QEP functions. The user may select either PCLK or ACLK as the clock input for each timer module.

21.2 Timer C Block Diagram

Figure 21-2 Timer C Block Diagram



21.3 Features

The PWM Timer peripheral has the following features in the PAC55XX family of controllers.

General Features:

- Configurable input clock: PCLK or ACLK
- Up to 300MHz clock input for 3.33ns PWM edge resolution
- 3-bit input clock divider
- Latch timer period and all CCR values on command

Base Timer Features:

- Single-shot or auto-reload
- Base timer interrupts
- Timer synchronization
- Timer Modes:
  - Disabled
  - Up mode
  - Up/Down mode
  - Up/Down Asymmetric mode
- Timer Register Latching Options
  - Latch TAPRD when counter = 0
  - Latch TAPRD when counter = period
  - Latch TAPRD immediately
- Dead-time Generator Input Clock
  - DTG clock = PCLK
  - DTG clock = ACLK

CCR/PWM Features:

- PWM output or input capture
- 8 CCR units per timer
- CCR interrupts
- CCR interrupt skips
- SW force CCR interrupt
- CCR interrupt type:
  - Rising, falling, both
- CCR latch modes:
  - Compare/PWM mode: counter = 0, counter = period, immediate
  - Capture input: rising edge, falling edge, both

- Force compare event
- Invert CCR output
- CCR phase delay for phase shifted drive topologies
- ADC trigger outputs:
  - PWM rising or falling edge

Dead-Time Generator (DTG) Features:

- DTG enable or bypass
- 12-bit rising edge delay
- 12-bit falling edge delay

QEP Decoder Features:

- QEP decoder enabled
- Direction status
- Configurable interrupts:
  - Phase A rising edge
  - Phase B rising edge
  - Index event
  - Counter wrap
- 4 Different counting modes for best resolution, range and speed performance

21.4 Functional Description

21.4.1 Timer Clock Structure

The timer peripheral input clock can be selected as either PCLK or ACLK. The input clock may be selected by the **TCCTL.CLKSRC** register.

Each timer peripheral has a 4-bit divider that can be used to divide the selected input clock. The user may set this divider by the **TCCTL.CLKDIV** register. There are 16 settings between /1 and /128 for the input clock divider.

The base timer also supplies the clock for the Dead-Time Generators (DTGs) to allow for different range and resolution for dead-time. The timer may select the DTGCLK to be the timer clock before or after the clock divider. The DTG input clock may be selected by the **TCCTL.DTGCLK** register.

To use the timer clock before the input clock divider, set **TCCTL.DTGCLK** to 0b. To use the timer clock after the input clock divider, set **TCCTL.DTGCLK** to 1b.

21.4.2 Timer Counter

The base timer is a 16-bit timer that can count either up, or up then down to support both edge and center aligned and asymmetric PWM output types.

The timer period is stored in the **TCPRD** register. The current value of the timer counter is updated at every timer tick and is stored in the **TCCTR** register. The **TCCTR** register is a RW register, so it may be updated at any time and is changed immediately.

The **TCPRD** register and all of the **TCCCTRn** registers have shadow copies that are updated at a user-specified time. See the section below on Base Timer **TAPRD** Latching and CCR Timer Latching for more information.

When **TCCTL.MODE** = 00b, the timer is disabled. Even when disabled, all timer registers are accessible via the APB bus.

To enable the timer, set the **TCCTL.MODE** to 01b (up mode), 10b (up/down mode) or 11b (up/down asymmetric mode).

When **TCCTL.MODE** = 01b, the timer is configured in up mode. The timer will count from 0 to **TCPRD**. The counter is updated in **TCCTR** at every timer tick. If auto-reload is active (**TCCTL.SINGLE** = 0b), then the timer count will automatically count from **TCPRD** to 0 and continue counting up. If auto-reload is not active (**TCCTL.SINGLE** = 1b), then the timer will count from 0 to **TCPRD** and stop. When this happens, the timer will set the **TCCTL.MODE** to 00b (disabled).

When the timer is configured for up/down mode (**TCCTL.MODE** = 10b) or configured for up/down asymmetric mode (**TCCTL.MODE** = 11b), the timer will count from 0 to **TCPRD** and then back down to 0. If auto-reload is not active (**TCCTL.SINGLE** = 1b), the time timer will stop and set **TCCTL.MODE** to 00b (disabled). If auto-reload is active (**TCCTL.SINGLE** = 0b), then the timer count will then continue counting back up after it reaches 0.

21.4.3 Up Mode

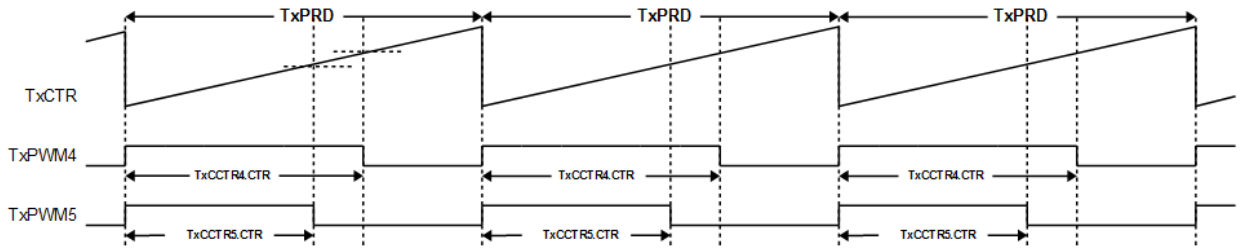
When the **TCCTL.MODE** is set to 01b (up mode) the timer will begin counting up from 0. The **TCCTR** will count to the value of **TCPRD** and then will reset to 0.

If the timer is configured for auto-reload mode (**TCCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it reaches 0.

The **TCCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate an edge aligned PWM as shown below.

Figure 21-3 Up Mode PWM Waveform



In this mode, when **TCCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TCCTR** register reaches the **TCCCTRn.CTR** threshold. When the **TCCTR** counts from **TCPRD** to 0, the PWM output is transitioned to high.

21.4.4 Up/Down Mode

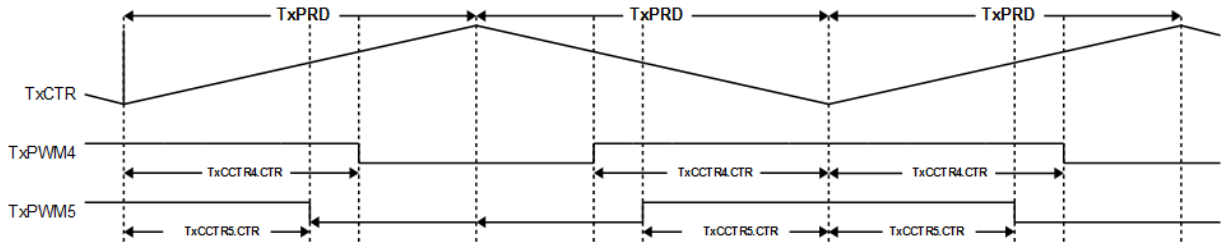
When the **TCCTL.MODE** is set to 10b (up/down mode) the timer will begin counting up from 0. The **TCCTR** will count to the value of **TCPRD** and then will begin counting back down to 0.

If the timer is configured for auto-reload mode (**TCCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it counts from 1 to 0.

The **TCCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate a center aligned PWM as shown below.

Figure 21-4 Up/Down Mode PWM Waveform



In this mode, when **TCCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TBCTR** register reaches the **TCCCTRn.CTR** threshold. When the **TCCTR** counts from **TCPRD** - 1 to **TCPRD**, it will begin counting down. When the **TCCTR** counts down to **TCCCTRn.CTR** the PWM output is transitioned to high.

21.4.5 Up/Down Asymmetric mode

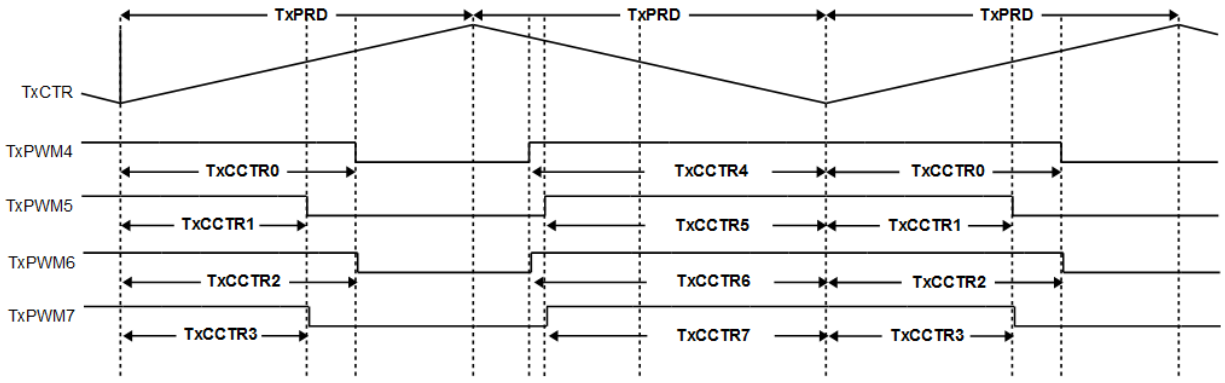
When the **TCCTL.MODE** is set to 11b (up/down asymmetric mode) the timer will begin counting up from 0. The **TCCTR** will count to the value of **TCPRD** and then will begin counting back down to 0.

If the timer is configured for auto-reload mode (**TCCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it counts from 1 to 0.

The **TCCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate a center aligned asymmetric PWM as shown below.

Figure 21-5 Up/Down Asymmetric Mode PWM Waveform



In this mode, when **TCCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TCCTR** register reaches the **TCCCTRn.CTR** threshold. When the

TCCTR counts from **TCPRD** - 1 to **TCPRD**, it will begin counting down. When the **TCCTR** counts down to **TCCCTRn.CTR** the PWM output is transitioned to high.

In this mode, the CCR outputs are generated such that 2 **TCCCTRn.CTR** registers are used to generate a single PWM output, so that the center-aligned PWM can be generated with asymmetric on-time.

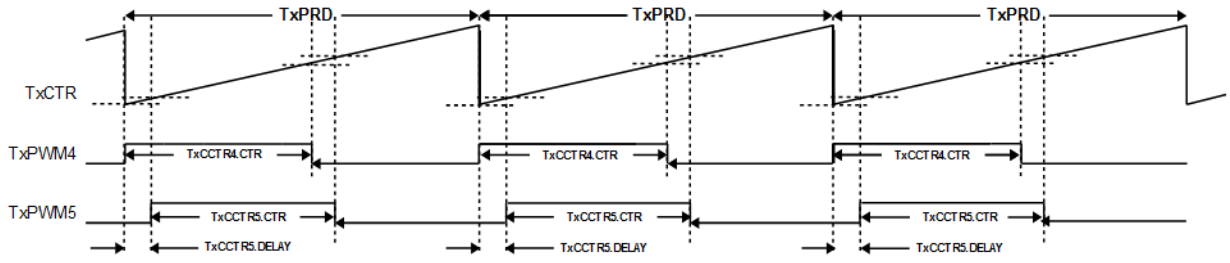
In this mode, the PWM outputs of the timer are generated as follows:

- TCPWM4 uses **TCCCTR0.CTR** for the first portion of the on time and **TCCCTR4.CTR** for the second portion of the on time
- TCPWM5 uses **TCCCTR1.CTR** for the first portion of the on time and **TCCCTR5.CTR** for the second portion of the on time
- TCPWM6 uses **TCCCTR2.CTR** for the first portion of the on time and **TCCCTR6.CTR** for the second portion of the on time
- TCPWM7 uses **TCCCTR3.CTR** for the first portion of the on time and **TCCCTR7.CTR** for the second portion of the on time

21.4.6 Up Mode with Phase Delay

When the timer is configured for up mode (**TCMODE** = 01b) the CCR allows a delay to be applied to the output PWM period, to support phase delay control topologies. If the value of **TCCCTRn.DELAY** > 0, then the CCR will delay this number of ticks before starting the PWM period (transitioning the output signal from high to low).

Figure 21-6 Up Mode with Phase Delay



21.4.7 Timer Synchronization

The timer peripheral allows synchronization between the timers in the PAC55XX. The timers may be synchronized as follows:

- Timer A (master), Timer B (slave)
- Timer A (master), Timer B (slave), Timer C (slave)
- Timer A (master), Timer B (slave), Timer C (slave), Timer D (slave)

The timer master's SYNC_OUT signals are connected to the downstream timer slave's SYNC_IN signal.

When configured, the timers can synchronize their clocks so they are sharing the same time base. In order to do this, the timers must be configured with the same clock source (**TCCTL.CLKSRC**) and clock divider (**TCCTL.CLKDIV**).

To configure the timers for synchronization, follow these steps:

- While the timer master is disabled (**TCCTL.MODE** = 00b), set the timer master **TxCTL.SSYNC** = 0b
- For each of the downstream timer slaves, while the timer is disabled **TCCTL.MODE** = 00b, set the **TCCTL.SSYNC** to 1b. Make sure that **TCCTL.CLKDIV** and **TCCTL.CLKSRC** for each of the timer slaves is the same as the timer master.
- Enable each of the slave timers by setting **TCCTL.MODE** to 01b (up), 10b (up/down) or 11b (up/down asymmetric). The slave timers will not start counting since the **TCCTL.SSYNC** is set to a 1b.
- Enable the timer master by setting **TxCTL.MODE** to 01b (up), 10b (up/down) or 11b (up/down asymmetric).

At this point, the master and all configured timer slaves will begin counting.

21.4.8 Base Timer TAPRD Latching

The timer peripherals have a shadow copy of the **TCPRD** register that is used for the counting operations. The latching of the data from the **TCPRD** register to the shadow register is controlled by the timer configuration.

There are several options for latching of this register into the shadow copy. This behavior is controlled by the **TCCTL.PRDLATCH** setting and depends on the setting of the **TCCTL.MODE** register as well.

21.4.8.1 TCPRD Latch When TCCTR = 0

- If **TCCTL.MODE** = 01b (up mode) and **TCCTL.PRDLATCH** = 00b (**TCCTR** = 0), then the **TCPRD** register is copied into the shadow register when **TCCTR** counts from **TCPRD** to 0.
- If **TCCTL.MODE** = 10b (up/down mode) or if **TCCTL.MODE** = 11b (up/down asymmetric mode) and **TCCTL.PRDLATCH** = 00b (**TCCTR** = 0), then the **TCPRD** register is copied into the shadow register when the **TCCTR** counts from 1 to 0.

21.4.8.2 TCPRD Latch When TCCTR = TCPRD

- If **TCCTL.MODE** = 01b (up mode) or if **TCCTL.MODE** = 10b (up/down mode) or if **TCCTL.MODE** = 11b (up/down asymmetric mode) and **TCCTL.PRDLATCH** = 01b

(**TCCTR** = **TBPRD**), then the **TCPRD** register is copied into the shadow register when **TCCTR** counts from **TCPRD** – 1 to **TCPRD**.

21.4.8.3 TCPRD Latch Immediate

- If **TCCTL.PRDLATCH** = 11b (immediate), then the **TCPRD** register is copied into the shadow register as soon as the **TCPRD** register is written.

21.4.9 CCR Timer Latching

When in compare mode, the CCR units support configurable latching of the **TCCCRn.CTR** values into a shadow register that is used for timer operation. The shadow register is used for generating the PWM output when the CCR is in compare mode.

This feature can be used to control when the new duty cycle is applied, during the PWM period.

When the CCR is in compare mode (**TCCCTLn.CCMODE** = 0b), the shadow register is latched as follows:

- If **TCCCTLn.CCLATCH** = 00b (**TCCTR** = 0)
  - If **TCCTL.MODE** = 01b (up mode), **TCCCTRn.CTR** is copied into the shadow register when **TCCTR** counts from **TCPRD** to 0. If **TCCTL.MODE** = 10b (up/down mode) or 11b (up/down asymmetric mode), **TCCCTRn.CTR** is copied into the shadow register when **TCCTR** counts from 1 to 0.
- If **TCCCTLn.CCLATCH** = 01b (**TCCTR = TCPRD**), then the **TCCCTRn.CTR** is copied into the shadow register when **TCCTR** counts from **TCPRD** – 1 to **TCPRD**.
- If **TCCCTLn.CCLATCH** = 10b (latch immediate), then the **TCCCTRn.CTR** is copied into the shadow register as soon as it is written.

When the CCR is in capture mode (**TCCCTLn.CCMODE** = 1b), the configuration of **TCCCTLn.CCLATCH** controls when the **TCCTR** is copied into the **TCCCTRn.CTR** register.

- If **TCCCTLn.CCLATCH** = 00b (rising edge), then the value of **TCCTR** is copied into the **TCCCTLn.CTR** register upon a rising edge in the input signal.
- If **TCCCTLn.CCLATCH** = 01b (falling edge), then the value of the **TCCTR** is copied into the **TCCCTLn.CTR** register upon a falling edge in the input signal.
- If **TCCCTLn.CCLATCH** = 10b (both), then the value of the **TCCTR** is copied into the **TCCCTLn.CTR** register on both a rising and falling edge of the input signal.

21.4.10 Timer Whole Latching

It is sometimes convenient to latch both the **TCPRD** (period) and all **TCCCTRn.CTR** (duty cycle) registers into the shadow registers at one time.

If the **TCCTL.LATCH** bit is written to a 1b, then all the **TCPRD** and all **TCCCTRn.CTR** registers will be copied into the shadow registers on the same clock cycle.

The **TCCTL.LATCH** bit is self-clearing, and will always be read as a 0b.

21.4.11 Inverting CCR PWM Output

When the CCR is in compare mode (**TCCCTRn.CCMODE** = 0b), the output of the CCR (the input to the DTG) may be inverted. To invert this output, set the **TCCCTRn.CCOUTINV** = 1b.

This mode is useful for some control topologies, especially full-bridge.

21.4.12 Base Timer Interrupts

The base timer may be configured to generate an interrupt to the NVIC.

If **TCCTL.MODE** = 01b (up mode) and **TCCTR** counts from **TCPRD** to 0 then the **TCINT.BASEIF** bit is set to 1b. If the **TCCTL.BASEIE** interrupt enable bit is set to a 1b, then the TIMERC_IRQ signal to the NVIC is asserted.

If **TCCTL.MODE** = 10b (up/down mode) or 11b (up/down asymmetric mode) and **TCCTR** counts from 1 to 0, then the **TCINT.BASEIF** bit is set to 1b. If the **TCCTL.BASEIE** interrupt enable bit is set to a 1b, then the TIMERC_IRQ signal to the NVIC is asserted.

The **TCINT.BASEIF** may be cleared by writing a 1b to it.

21.4.13 CCR Compare Interrupts

When **TxCCTLn.CCMODE** = 0b (compare mode), the CCR unit may be configured to generate an interrupt to the NVIC.

When the PWM edge transitions, then the CCR interrupt flag in the **TxINT** register is set as follows:

- If **TxCCTLn.CCINTEDGE** = 00b (rising edge) and a rising edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and TIMER_IRQ to the NVIC is asserted.
- If **TxCCTLn. CCINTEDGE** = 01b (falling edge) and a falling edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and TIMER_IRQ to the NVIC is asserted. **TxINT.CCRnIF** may be cleared by writing it to a 1b.
- If **TxCCTLn. CCINTEDGE** = 10b (rising or falling edge) and a rising or falling edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and

TIMER_IRQ to the NVIC is asserted. **TxINT.CCRnIF** may be cleared by writing it to a 1b.

TxINT.CCRnIF may be cleared by writing it to a 1b.

21.4.14 CCR Capture Interrupts

When **TxCCTLn.CCMODE** = 1b (capture mode), the CCR unit may be configured to generate an interrupt to the NVIC based on the input signal.

When the PWM edge transitions, then the CCR interrupt flag in the **TxINT** register is set as follows:

- If **TxCCTLn.CCINTEDGE** = 00b (rising edge) and a rising edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is set.
- If **TxCCTLn.CCINTEDGE** = 01b (falling edge) and a falling edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is set.
- If **TxCCTLn.CCINTEDGE** = 10b (rising or falling edge) and a rising or falling edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is.

If the **TxINT.CCRnIF** interrupt flag is set and the corresponding **TxCCTLn.CCINTEN** bit is set, then the TIMERA_IRQ signal to the NVIC will be asserted.

The **TxINT.CCRnIF** interrupt flag can be cleared by writing it to a 1b.

21.4.15 Timer IRQ signal

The timer unit has one IRQ output signal to the NVIC (TIMERC_IRQ).

This signal is asserted when any of the following conditions are true:

- **TCINT.BASEIF** = 1b and **TCCTL.BASEIE** = 1b
- **TCINT.CCR0IF** = 1b and **TCCCTL0.CCINTEN** = 1b
- **TCINT.CCR1IF** = 1b and **TCCCTL1.CCINTEN** = 1b
- **TCINT.CCR2IF** = 1b and **TCCCTL2.CCINTEN** = 1b
- **TCINT.CCR3IF** = 1b and **TCCCTL3.CCINTEN** = 1b
- **TCINT.CCR4IF** = 1b and **TCCCTL4.CCINTEN** = 1b
- **TCINT.CCR5IF** = 1b and **TCCCTL5.CCINTEN** = 1b
- **TCINT.CCR6IF** = 1b and **TCCCTL6.CCINTEN** = 1b
- **TCINT.CCR7IF** = 1b and **TCCCTL7.CCINTEN** = 1b

The TIMERC_IRQ signal is de-asserted when all of the above conditions are false.

21.4.16 Skipping CCR Interrupts

Sometimes it is useful to not generate a CCR interrupt to the NVIC every time the **TCCTR** counts to the **TCCCTRn.CTR** value. For example, in a control application when the PWM frequency is fast, but you only need a MCU interrupt every 5 PWM periods.

Each CCR in the timer has the ability to skip interrupts to the NVIC to allow this capability.

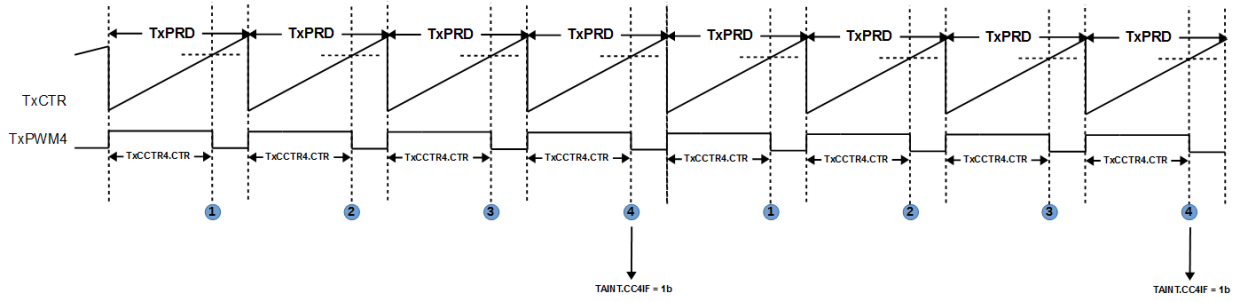
The CCR will generate interrupts to the NVIC according to the **TCCCTL.CCINTSKIP** field as follows:

- If **TCCCTLn.CCINTSKIP** = 0000b, then the CCR will generate an interrupt each time
- If **TCCCTLn.CCINTSKIP** = 0001b, then the CCR will skip 1 CCR cycle before generating an interrupt to the NVIC.
- If **TCCCTLn.CCINTSKIP** = 0010b, then the CCR will skip 2 CCR cycles before generating an interrupt to the NVIC.
- If **TCCCTLn.CCINTSKIP** = 0011b, then the CCR will skip 3 CCR cycles before generating an interrupt to the NVIC.
- …

- If **TCCCTLn.CCINTSKIP** = 1110b, then the CCR will skip 14 CCR cycles before generating an interrupt to the NVIC.
- If **TCCCTLn.CCINTSKIP** = 1111b, then the CCR will skip 15 CCR cycles before generating an interrupt to the NVIC.

The diagram below shows an example of a CCR4 with interrupt skipping configured with a value of 4 (**TCCCTL.CCINTSKIP** = 4).

Figure 21-7 CCR Interrupt Skipping Diagram



21.4.17 Timer ADC Triggers

Each timer CCR unit may be configured to be an ADC trigger, to allow the DTSE to automatically begin conversion sequences.

For more information on this feature, see the section on the ADC and DTSE in this user guide.

21.4.18 Dead-Time Generators (DTG)

Each timer has 4 DTG units. Each DTG unit is capable of generating a pair of complementary signals that can be configured with dead-time to drive an inverter for half-bridge topologies.

The input clock to the DTG can be configured to be the timer clock before or after the **TCCTL.CLKDIV** input clock divider. If **TCCTL.DTGCLK** is 0b, then the DTGCLK is the clock before the **TCCTL.CLKDIV** clock divider. If **TCCTL.DTGCLK** is 1b, then the DTGCLK is the clock after the **TCCTL.CLKDIV** clock divider.

When **TCDTGCTLn.DTEN** = 0b, the DTG is disabled (bypassed). In this mode, the timer signals have the following behavior (see the simplified block diagram above):

DTG0:

- CCR0 output is connected to TCPWM0
- CCR4 output is connected to TCPWM4

DTG1:

- CCR1 output is connected to TCPWM1
- CCR5 output is connected to TCPWM5

DTG2:

- CCR2 output is connected to TCPWM2
- CCR6 output is connected to TCPWM6

DTG3:

- CCR3 output is connected to TCPWM3
- CCR7 output is connected to TCPWM7

When **TCDTGCTL0.DTEN** = 1b, the DTG is enabled. In this mode, only one CCR output is used to generate the two complementary outputs. In this mode, the timer signals have the following behavior:

DTG0:

- CCR0 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR4 output is used to generate the two complementary output signals: TCPWM0 and TCPWM4.

- TCPWM0 becomes the low-side complementary signal.
- TCPWM4 becomes the high-side complementary signal.
- The **TCDTGCTL0.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, to when the low-side signal rises).
- The **TCDTGCTL0.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, to when the high-side signal rises).

DTG1:

- CCR1 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR5 output is used to generate the two complementary output signals: TCPWM1 and TCPWM5.
- TCPWM1 becomes the low-side complementary signal.
- TCPWM5 becomes the high-side complementary signal.
- The **TCDTGCTL1.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TCDTGCTL1.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

DTG2:

- CCR2 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR6 output is used to generate the two complementary output signals: TCPWM2 and TCPWM6.
- TCPWM2 becomes the low-side complementary signal.
- TCPWM6 becomes the high-side complementary signal.
- The **TCDTGCTL2.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TCDTGCTL2.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

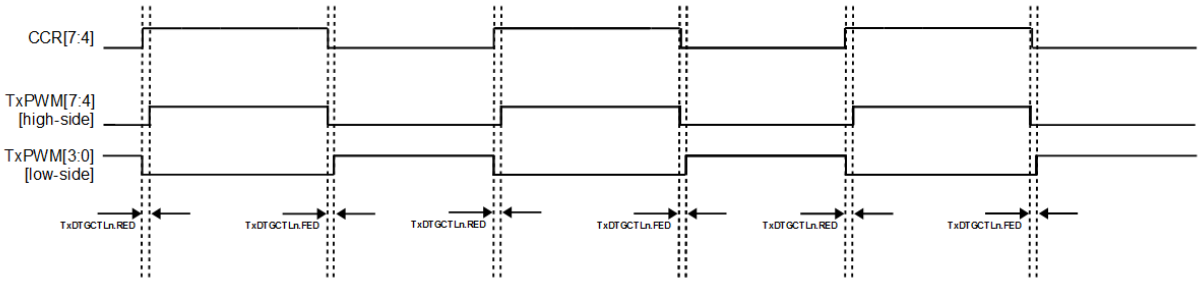
DTG3:

- CCR3 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR7 output is used to generate the two complementary output signals: TCPWM3 and TCPWM4.
- TCPWM3 becomes the low-side complementary signal.
- TCPWM7 becomes the high-side complementary signal.
- The **TCDTGCTL3.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TCDTGCTL0.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

The PWM waveform below shows an example of the dead-time applied to the signals through the DTG units.

21.4.19 Timer C DTG Output

Figure 21-8 Timer C DTG Output



21.5 QEP

21.5.1 Overview

Each PAC55XX PWM Timer contains a QEP decoder that can be used to determine the position, speed, and direction of a motor with a QEP encoder.

The inputs to the QEP peripheral are TxQEPPHA, TxQEPPHB and TxQEPIDX (phase A, phase B and index). These three inputs have a glitch filter applied to them, to filter out noise and are used for calculation of the motor position, speed, and direction.

21.5.2 Configuration

The QEP peripheral can be enabled by setting the **TxQEPCTL.QEPEN** bit to a 1b.[19] The input clock to the QEP peripheral may be configured between /1 and /128 using the **TxQEPCTL.CLKDIV** field.

If the **TxQEPCTL.CNTAB** bit is 0b, only phase A edges are processed. If the **TxQEPCTL.CNTAB** bit is set to a 1b, then both phase A and phase B edges are processed. If the **TxQEPCTL.CNTEDGE** bit is set to 0b, then only the rising edges are processed. If the **TxQEPCTL.CNTEDGE** bit is set to a 1b, then both rising and falling edges are processed.

If the direction is determined to be clockwise (CW), then the **TxQEPCTL.DIR** bit is set to 0b. If the direction is determined to be counterclockwise (CCW), then the **TxQEPCTL.DIR** bit is set to 1b. The direction will get updated only on edges that are processed.

When an edge is processed, the QEP peripheral will first determine the direction and then increment or decrement the counter value in **TxQEPCTR.CTR** accordingly. **TxQEPCTR.CTR** may be written at any time.

If an edge is processed while the motor is turning clockwise, the counter value in **TxQEPCTR.CTR** is incremented. If the value of **TxQEPCTR.CTR** was **TxQEPMAX.CTRMAX** during this event, then **TxQEPCTR.CTR** is set to 0.

If an edge is processed while the motor is turning counterclockwise, the counter value in **TxQEPCTR.CTR** is decremented. If the value of **TxQEPCTR.CTR** was 0 during this event, then **TxQEPCTR.CTR** is set to **TxQEPMAX.CTRMAX**.

[19] Before enabling the QEP peripheral, be sure to set up the Digital Peripheral MUX, so that no false edges are detected by the QEP state machine.

21.5.3 Index Reset

If the **TxQEPCTL.IDXRST** bit is set to a 1b, then an index event clears the **TxQEPCTR.CTR** counter to 0. If the motor is turning in the clockwise direction during this event, **TxQEPCTR.CTR** is set to 0. If the motor is turning in the counterclockwise direction during this event the **TxQEPCTR.CTR** is set to 0 (if not coincident with the phase edge) or **TxQEPMAX.CTRMAX** (if coincident with the phase edge).

If this bit is set to a 0b, then an index event does not reset the **TAQEPCTL.TICKS** counter.

21.5.4 Interrupts

If a change to the direction is detected (0 to 1 or 1 to 0), then the **TxQEPIF.DIRIF** bit is set to a 1b. If the **TxQEPIER.DIRIE** bit is set to 1b, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.DIRIF** bit can be cleared by writing it to 1b.

When a rising edge on phase A is detected the **TxQEPIF.PHAIF** bit is set to 1b. If the **TxQEPIER.PHAIE** bit is set and the **TxQEPIF.PHAIF** bit is set, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.PHAIF** bit can be cleared by writing a 1b to it.

When a rising edge on phase B is detected the **TxQEPIF.PHBIF** bit is set. If the **TxQEPIER.PHBIE** bit is set and the **TxQEPIF.PHBIF** bit is set, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.PHBIF** bit can be cleared by writing a 1b to it.

If an overflow or underflow in the **TxQEPCTR.CTR** is detected, the **TxQEPIF.WRIF** bit is set (counter wrap interrupt flag). If the **TxQEPIER.WRIE** bit is set and the **TxQEPIF.WRIF** bits are set, then the QEPx_IRQ signal is asserted to the NVIC.

If an index event is detected, the **TxQEPIF.IDXEVIF** bit is set. If the **TxQEPIER.IDXEVIE** bit is set and the **TxQEPIF.IDXEVIF** bit is set, then the QEPx_IRQ signal to the NVIC is asserted. The **TxQEPIF.IDXEVIF** bit can be cleared by writing a 1b to it.

21.6 Peripheral IO Mapping

The Timer C peripheral signal inputs and outputs (CCR, QEP) are connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 21-1 Timer C Peripheral IO Mapping

TIMER SIGNAL	IO PIN
TCPWM0	PC0, PD0, PE0, PF0, PG0
TCPWM1	PC1, PD1, PE1, PF1, PG1
TCPWM2	PC2, PD2, PE2, PF2, PG2
TCPWM3	PC3, PD3, PE3, PF3, PG3
TCPWM4	PC4, PD4, PE4, PF4, PG4
TCPWM5	PC5, PD5, PE5, PF5, PG5
TCPWM6	PC6, PD6, PE6, PF6, PG6
TCPWM7	PC7, PD7, PE7, PF7
TCQEPIDX	PC4, PF4, PG0
TCQEPPHA	PC5, PF5, PG1
TCQEPPHB	PC6, PF6, PG2

For more information on how to use the Digital Peripheral MUX to connect peripheral signals to IO, see GPIO and DPM.

21.7 Timer C Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
TCCTL	4008 0000h	Timer C control	RW	0000 0000h
TCINT	4008 0004h	Timer C interrupt control	RW	0000 0000h
TCPRD	4008 0008h	Timer C period	RW	0000 0000h
TCCTR	4008 000Ch	Timer C counter	RW	0000 0000h
TCCCTL0	4008 0100h	Timer C CC control 0	RW	0000 0000h
TCCCTR0	4008 0104h	Timer C CC counter 0	RW	0000 0000h
TCCCTL1	4008 0108h	Timer C CC control 1	RW	0000 0000h
TCCCTR1	4008 010Ch	Timer C CC counter 1	RW	0000 0000h
TCCCTL2	4008 0110h	Timer C CC control 2	RW	0000 0000h
TCCCTR2	4008 0114h	Timer C CC counter 2	RW	0000 0000h
TCCCTL3	4008 0118h	Timer C CC control 3	RW	0000 0000h
TCCCTR3	4008 011Ch	Timer C CC counter 3	RW	0000 0000h
TCCCTL4	4008 0120h	Timer C CC control 4	RW	0000 0000h
TCCCTR4	4008 0124h	Timer C CC counter 4	RW	0000 0000h
TCCCTL5	4008 0128h	Timer C CC control 5	RW	0000 0000h
TCCCTR5	4008 012Ch	Timer C CC counter 5	RW	0000 0000h
TCCCTL6	4008 0130h	Timer C CC control 6	RW	0000 0000h
TCCCTR6	4008 0134h	Timer C CC counter 6	RW	0000 0000h
TCCCTL7	4008 0138h	Timer C CC control 7	RW	0000 0000h
TCCCTR7	4008 013Ch	Timer C CC counter 7	RW	0000 0000h
TCDTGCTL0	4008 0200h	Timer C DTG control 0	RW	0000 0000h
TCDTGCTL1	4008 0204h	Timer C DTG control 1	RW	0000 0000h
TCDTGCTL2	4008 0208h	Timer C DTG control 2	RW	0000 0000h
TCDTGCTL3	4008 020Ch	Timer C DTG control 3	RW	0000 0000h
TCQEPCTL	4008 0300h	Timer C QEP control	RW	0000 0000h
TCQEPCTR	4008 0304h	Timer C QEP counter	RW	0000 0000h
TCQEPMAX	4008 0308h	Timer C QEP max counter value	RW	0000 FFFFh
TCQEPIER	4008 030Ch	Timer C QEP interrupt enable	RW	0000 0000h
TCQEPIFR	4008 0310h	Timer C QEP interrupt flag	RW	0000 0000h

21.8 Register Detail

21.8.1 TCCTL

Register 21-1 TCCTL (Timer C Control, 4008 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:14	**Reserved**	RO	0	Reserved
13	**BASEIE**	RW	0	Base timer interrupt enable: 0b: not enabled 1b: enabled
12	**CLR**	RW	0	Base timer clear: 0b: do not clear 1b: clear timer counter
11	**LATCH**	RW	0	When written to 1b, this will latch the **TCPRD** and all **TCCCTRn** registers on the same clock cycle. Writing this bit to 0b has no effect. This is a self-clearing bit.
10	**DTGCLK**	RW	0	DTG Clock Source: 0b: Before input clock divider 1b: After input clock divider
9	**CLKSRC**	RW	0	Timer Clock Source: 0b: PCLK 1b: ACLK
8:6	**CLKDIV**	RW	0	Timer Input Clock Divider: 000b: /1 001b: /2 010b: /4 011b: /8 100b: /16 101b: /32 110b: /64 111b: /128
5	**SINGLE**	RW	0	Single Shot Timer: 0b: disabled (auto-reload) 1b: enabled (single shot timer)
4	**SSYNC**	RW	0	Timer Slave Synchronization: 0b: disabled 1b: enabled
3:2	**PRDLATCH**	RW	0	Timer Period Latch Mode: 00b: Latch **TCPRD** when **TCCTR** = 0 01b: Latch **TCPRD** when **TCCTR** = **TCPRD** 10b: Latch **TCPRD** immediately upon register write 11b: reserved
1:0	**MODE**	RW	0	Timer Mode: 00b: disabled 01b: up mode 10b: up/down mode 11b: asymmetric mode

21.8.2 TCINT

Register 21-2 TCINT (Timer C Interrupt Control, 4008 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:9	**Reserved**	RO	0	Reserved
8	**BASEIF**	W1C	0	Base timer interrupt flag: 0b: no interrupt flag 1b: interrupt flag
7	**CCR7IF**	W1C	0	CCR7 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
6	**CCR6IF**	W1C	0	CCR6 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
5	**CCR5IF**	W1C	0	CCR5 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
4	**CCR4IF**	W1C	0	CCR4 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
3	**CCR3IF**	W1C	0	CCR3 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
2	**CCR2IF**	W1C	0	CCR2 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
1	**CCR1IF**	W1C	0	CCR1 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
0	**CCR0IF**	W1C	0	CCR0 interrupt flag: 0b: no interrupt flag 1b: interrupt flag

21.8.3 TCPRD

Register 21-3 TCPRD (Timer C Period, 4008 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**PERIOD**	RW	0	Timer period value

21.8.4 TCCTR

Register 21-4 TCCTR (Timer C Counter, 4008 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**COUNTER**	RW	0	Timer counter

21.8.5 TCCCTL0

Register 21-5 TCCCTL0 (Timer C CCR Control 0, 4008 0100h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TCCCTL0.CCMODE** = 0b (compare): 00b: Latch **TCCCTR0** registers when **TCCTR** = 0 01b: Latch **TCCCTR0** registers when **TCCTR** = **TCPRD** 10b: Latch **TCCCTR0** registers immediately 11b: Reserved If **TCCCTL0.CCMODE** = 1b (capture): 00b: Latch **TCCTR** into **TCCCTR0.CTR** on rising edge 01b: Latch **TCCTR** into **TCCCTR0.CTR** on falling edge 10b: Latch **TCCTR** into **TCCCTR0.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TCCCTL0.CCMODE** = 1b (capture), set **TCINT.CCR0IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

21.8.6 TCCCTR0

Register 21-6 TCCCTR0 (Timer C CCR Counter 0, 4008 0104h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TCCCTL0.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TCCCTRL0.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TCCCTL0.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

21.8.7 TCCCTL1

Register 21-7 TCCCTL1 (Timer C CCR Control 1, 4008 0108h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TCCCTL1.CCMODE** = 0b (compare): 00b: Latch **TCCCTR1** registers when **TCCTR** = 0 01b: Latch **TCCCTR1** registers when **TCCTR** = **TCPRD** 10b: Latch **TCCCTR1** registers immediately 11b: Reserved If **TCCCTL1.CCMODE** = 1b (capture): 00b: Latch **TCCTR** into **TCCCTR1.CTR** on rising edge 01b: Latch **TCCTR** into **TCCCTR1.CTR** on falling edge 10b: Latch **TCCTR** into **TCCCTR1.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TCCCTL1.CCMODE** = 1b (capture), set **TCINT.CCR1IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

21.8.8 TCCCTR1

Register 21-8 TCCCTR1 (Timer C CCR Counter 1, 4008 010Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TCCCTL1.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TCCCTRL1.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TCCCTL1.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

21.8.9 TCCCTL2

Register 21-9 TCCCTL2 (Timer C CCR Control 2, 4008 0110h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TCCCTL2.CCMODE** = 0b (compare): 00b: Latch **TCCCTR2** registers when **TCCTR** = 0 01b: Latch **TCCCTR2** registers when **TCCTR** = **TCPRD** 10b: Latch **TCCCTR2** registers immediately 11b: Reserved If **TCCCTL2.CCMODE** = 1b (capture): 00b: Latch **TCBCTR** into **TCCCTR2.CTR** on rising edge 01b: Latch **TCCTR** into **TCCCTR2.CTR** on falling edge 10b: Latch **TCCTR** into **TCCCTR2.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TCCCTL2.CCMODE** = 1b (capture), set **TCINT.CCR2IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

21.8.10 TCCCTR2

Register 21-10 TCCCTR2 (Timer C CCR Counter 2, 4008 0114h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TCCCTL2.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TCCCTRL2.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TCCCTL2.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

21.8.11 TCCCTL3

Register 21-11 TCCCTL3 (Timer C CCR Control 3, 4008 0118h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TCCCTL3.CCMODE** = 0b (compare): 00b: Latch **TCCCTR3** registers when **TCCTR** = 0 01b: Latch **TCCCTR3** registers when **TCCTR** = **TCPRD** 10b: Latch **TCCCTR3** registers immediately 11b: Reserved If **TCCCTL3.CCMODE** = 1b (capture): 00b: Latch **TCCTR** into **TCCCTR3.CTR** on rising edge 01b: Latch **TCCTR** into **TCCCTR3.CTR** on falling edge 10b: Latch **TCCTR** into **TCCCTR3.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TCCCTL3.CCMODE** = 1b (capture), set **TCINT.CCR3IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

21.8.12 TCCCTR3

Register 21-12 TCCCTR3 (Timer C CCR Counter 3, 4008 011Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TCCCTL3.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TCCCTRL3.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TCCCTL3.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

21.8.13 TCCCTL4

Register 21-13 TCCCTL4 (Timer C CCR Control 4, 4008 0120h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TCCCTL4.CCMODE** = 0b (compare): 00b: Latch **TCCCTR4** registers when **TCCTR** = 0 01b: Latch **TCCCTR4** registers when **TCCTR** = **TCPRD** 10b: Latch **TCCCTR4** registers immediately 11b: Reserved If **TCCCTL4.CCMODE** = 1b (capture): 00b: Latch **TCCTR** into **TCCCTR4.CTR** on rising edge 01b: Latch **TCCTR** into **TCCCTR4.CTR** on falling edge 10b: Latch **TCCTR** into **TCCCTR4.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TCCCTL4.CCMODE** = 1b (capture), set **TCINT.CCR4IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

21.8.14 TCCCTR4

Register 21-14 TCCCTR4 (Timer C CCR Counter 4, 4008 0124h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TCCCTL4.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TCCCTRL4.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TCCCTL4.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

21.8.15 TCCCTL5

Register 21-15 TCCCTL5 (Timer C CCR Control 5, 4008 0128h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TCCCTL5.CCMODE** = 0b (compare): 00b: Latch **TCCCTR5** registers when **TCCTR** = 0 01b: Latch **TCCCTR5** registers when **TCCTR** = **TCPRD** 10b: Latch **TCCCTR5** registers immediately 11b: Reserved If **TCCCTL5.CCMODE** = 1b (capture): 00b: Latch **TCCTR** into **TCCCTR5.CTR** on rising edge 01b: Latch **TCCTR** into **TCCCTR5.CTR** on falling edge 10b: Latch **TCCTR** into **TCCCTR5.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TCCCTL5.CCMODE** = 1b (capture), set **TCINT.CCR5IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

21.8.16 TCCCTR5

Register 21-16 TCCCTR5 (Timer C CCR Counter 5, 4008 012Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TCCCTL5.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TCCCTRL5.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TCCCTL5.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

21.8.17 TCCCTL6

Register 21-17 TCCCTL6 (Timer C CCR Control 6, 4008 0130h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TCCCTL6.CCMODE** = 0b (compare): 00b: Latch **TCCCTR6** registers when **TCCTR** = 0 01b: Latch **TCCCTR6** registers when **TCCTR** = **TCPRD** 10b: Latch **TCCCTR6** registers immediately 11b: Reserved If **TCCCTL6.CCMODE** = 1b (capture): 00b: Latch **TCCTR** into **TCCCTR6.CTR** on rising edge 01b: Latch **TCCTR** into **TCCCTR6.CTR** on falling edge 10b: Latch **TCCTR** into **TCCCTR6.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TCCCTL6.CCMODE** = 1b (capture), set **TCINT.CCR6IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

21.8.18 TCCCTR6

Register 21-18 TCCCTR6 (Timer C CCR Counter 6, 4008 0134h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TCCCTL6.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TCCCTRL6.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TCCCTL6.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

21.8.19 TCCCTL7

Register 21-19 TCCCTL7 (Timer C CCR Control 7, 4008 0138h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TCCCTL7.CCMODE** = 0b (compare): 00b: Latch **TCCCTR7** registers when **TCCTR** = 0 01b: Latch **TCCCTR7** registers when **TCCTR** = **TCPRD** 10b: Latch **TCCCTR7** registers immediately 11b: Reserved If **TCCCTL7.CCMODE** = 1b (capture): 00b: Latch **TCCTR** into **TCCCTR7.CTR** on rising edge 01b: Latch **TCCTR** into **TCCCTR7.CTR** on falling edge 10b: Latch **TCCTR** into **TCCCTR7.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TCCCTL7.CCMODE** = 1b (capture), set **TCINT.CCR7IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

21.8.20 TCCCTR7

Register 21-20 TCCCTR7 (Timer C CCR Counter 7, 4008 013Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TCCCTL7.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TCCCTRL7.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TCCCTL7.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

21.8.21 TCDTGCTL0

Register 21-21 TCDTGCTL0 (Timer C Dead-Time Generator Control 0, 4008 0200h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG0.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG0.

21.8.22 TCDTGCTL1

Register 21-22 TCDTGCTL1 (Timer C Dead-Time Generator Control 1, 4008 0204h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG1.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG1.

21.8.23 TCDTGCTL2

Register 21-23 TCDTGCTL2 (Timer C Dead-Time Generator Control 2, 4008 0208h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG2.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG2.

21.8.24 TCDTGCTL3

Register 21-24 TCDTGCTL3 (Timer C Dead-Time Generator Control 3, 4008 020Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG3.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG3.

21.8.25 TCQEPCTL

Register 21-25 TCQEPCTL (Timer C QEP Control, 4008 0300h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:11	**Reserved**	RO	0	Reserved
10:8	**CLKDIV**	RW	0	QEP Input Clock Divider: 000b: /1 001b: /2 010b: /4 011b: /8 100b: /16 101b: /32 110b: /64 111b: /128
7:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXRST**	RW	0	Index reset mode. Used for resetting counter. 0b: No reset 1b: Index edge reset. When direction is CW, reset counter to 0 when index rising edge event occurs. When direction is CCW, reset when index falling edge occurs (reset to 0 if not coincident with phase edge; reset to Tx**QEPMAX** if coincident with phase edge).
3	**DIR**	RO	0	Motor direction: 0b: CW (clockwise) 1b: CCW (counter-clockwise)
2	**CNTEDGE**	RW	0	Count on edge: 0b: Rising edge only 1b: Rising and falling edge
1	**CNTAB**	RW	0	Count on A/B: 0b: Count phase A only 1b: Count phase A and phase B
0	**QEPEN**	RW	0	QEP peripheral enable: 0b: disabled 1b: enabled

21.8.26 TCQEPCTR

Register 21-26 TCQEPCTR (Timer C QEP Counter, 4008 0304h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**CTR**	RW	0	Counter Value

21.8.27 TCQEPMAX

Register 21-27 TCQEPMAX (Timer C QEP Maximum Counter, 4008 0308h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**CTRMAX**	RW	0	Maximum Counter Value.

21.8.28 TCQEPIER

Register 21-28 TCQEPIER (Timer C QEP Interrupt Enable, 4008 030Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXEVIE**	RW	0	Index event interrupt enable: 0b: disabled 1b: enabled
3	**WRIE**	RW	0	Counter wrap interrupt enable: 0b: disabled 1b: enabled
2	**PHBIE**	RW	0	Phase B rising edge interrupt enable: 0b: disabled 1b: enabled
1	**PHAIE**	RW	0	Phase A rising edge interrupt enable: 0b: disabled 1b: enabled
0	**DIRIE**	RW	0	Direction change interrupt enable: 0b: disabled 1b: enabled

21.8.29 TCQEPIF

Register 21-29 TCQEPIF (Timer C QEP Interrupt Flag, 4008 0310h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXEVIF**	W1C	0	Index event interrupt flag: 0b: disabled 1b: enabled
3	**WRIF**	W1C	0	Counter wrap interrupt flag: 0b: disabled 1b: enabled
2	**PHBIF**	W1C	0	Phase B rising edge interrupt flag: 0b: disabled 1b: enabled
1	**PHAIF**	W1C	0	Phase A rising edge interrupt flag: 0b: disabled 1b: enabled
0	**DIRIF**	W1C	0	Direction change interrupt flag: 0b: disabled 1b: enabled

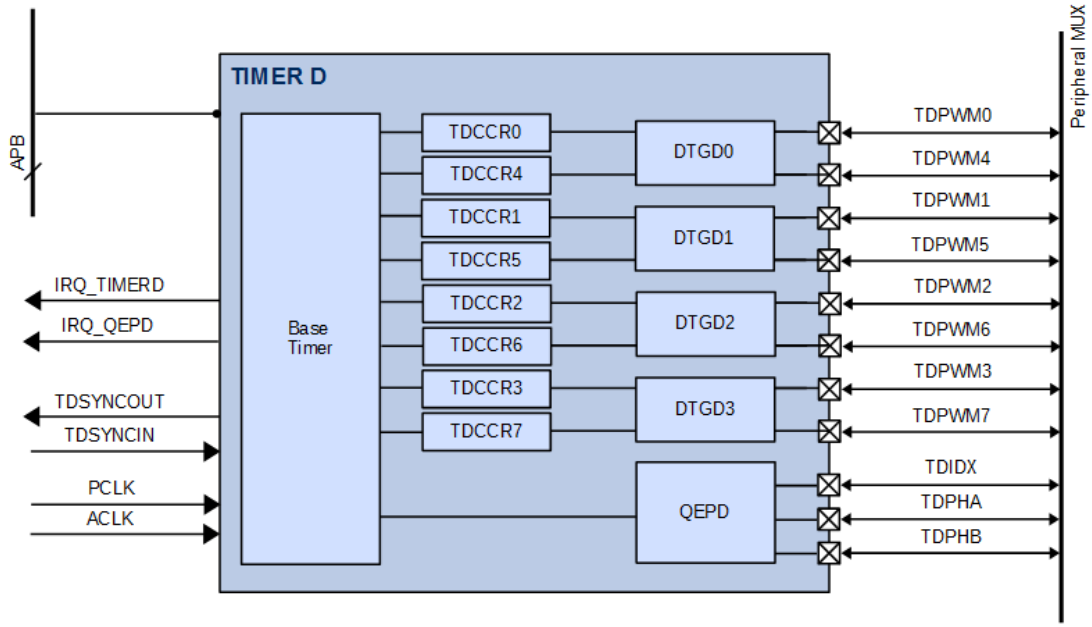
22 PWM TIMER D

22.1 Overview

All devices in the PAC55XX family of controllers have a Timer D peripheral. This peripheral is a 16-bit timer that allows support for 8 Capture and Compare Units (CCR) capable of PWM generation; capture input processing and a QEP decoder for various control applications.

Below is a simplified block diagram of the Timer peripheral.

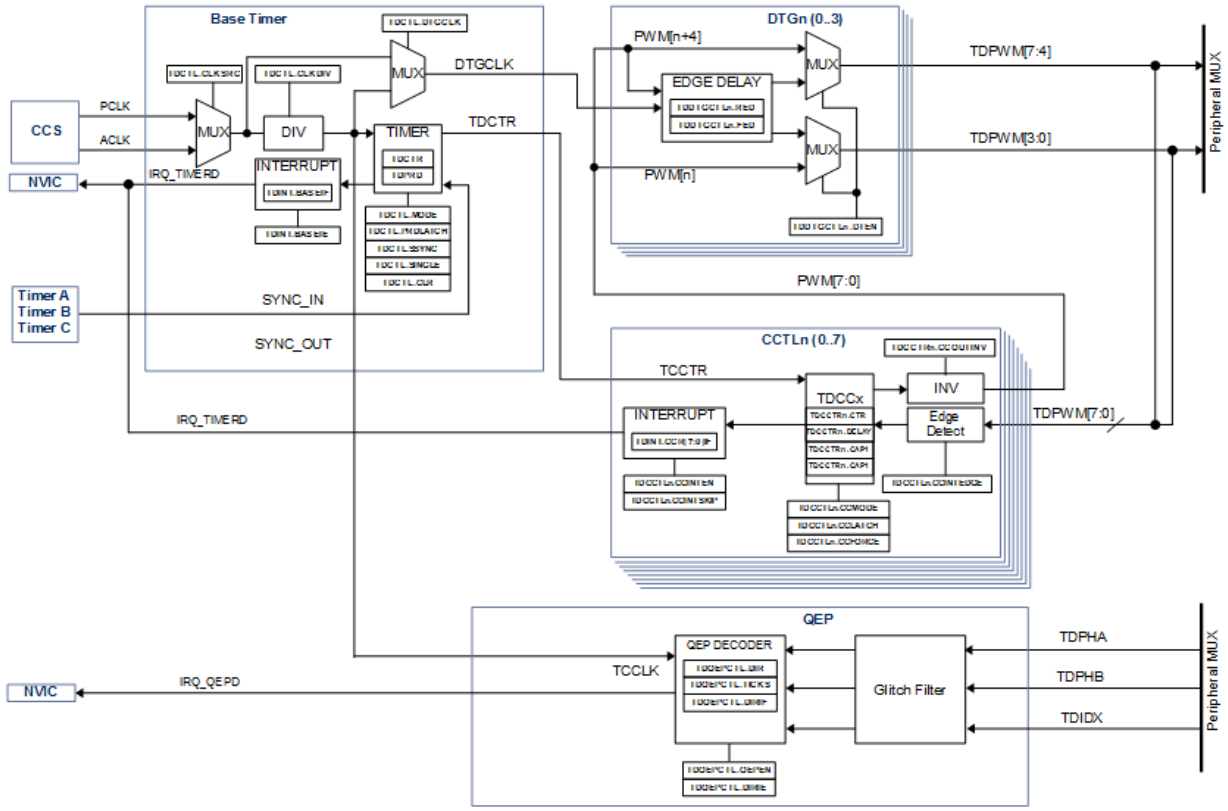
Figure 22-1 Timer D Simplified Block Diagram



The timer module is an APB bus client. There are 8 timer channel input/outputs to the digital peripheral MUX, in addition to three IOs for QEP. Each timer block has two interrupt signal outputs to the NVIC; one for timer functions and one for QEP functions. The user may select either PCLK or ACLK as the clock input for each timer module.

22.2 Timer D Block Diagram

Figure 22-2 Timer D Block Diagram



22.3 Features

The PWM Timer peripheral has the following features in the PAC55XX family of controllers.

General Features:

- Configurable input clock: PCLK or ACLK
- Up to 300MHz clock input for 3.33ns PWM edge resolution
- 3-bit input clock divider
- Latch timer period and all CCR values on command

Base Timer Features:

- Single-shot or auto-reload
- Base timer interrupts
- Timer synchronization
- Timer Modes:
  - Disabled
  - Up mode
  - Up/Down mode
  - Up/Down Asymmetric mode
- Timer Register Latching Options
  - Latch TAPRD when counter = 0
  - Latch TAPRD when counter = period
  - Latch TAPRD immediately
- Dead-time Generator Input Clock
  - DTG clock = PCLK
  - DTG clock = ACLK

CCR/PWM Features:

- PWM output or input capture
- 8 CCR units per timer
- CCR interrupts
- CCR interrupt skips
- SW force CCR interrupt
- CCR interrupt type:
  - Rising, falling, both
- CCR latch modes:
  - Compare/PWM mode: counter = 0, counter = period, immediate
  - Capture input: rising edge, falling edge, both

- Force compare event
- Invert CCR output
- CCR phase delay for phase shifted drive topologies
- ADC trigger outputs:
  - PWM rising or falling edge

Dead-Time Generator (DTG) Features:

- DTG enable or bypass
- 12-bit rising edge delay
- 12-bit falling edge delay

QEP Decoder Features:

- QEP decoder enabled
- Direction status
- Configurable interrupts:
  - Phase A rising edge
  - Phase B rising edge
  - Index event
  - Counter wrap
- 4 Different counting modes for best resolution, range and speed performance

22.4 Functional Description

22.4.1 Timer Clock Structure

The timer peripheral input clock can be selected as either PCLK or ACLK. The input clock may be selected by the **TDCTL.CLKSRC** register.

Each timer peripheral has a 4-bit divider that can be used to divide the selected input clock. The user may set this divider by the **TDCTL.CLKDIV** register. There are 16 settings between /1 and /128 for the input clock divider.

The base timer also supplies the clock for the Dead-Time Generators (DTGs) to allow for different range and resolution for dead-time. The timer may select the DTGCLK to be the timer clock before or after the clock divider. The DTG input clock may be selected by the **TDCTL.DTGCLK** register.

To use the timer clock before the input clock divider, set **TDCTL.DTGCLK** to 0b. To use the timer clock after the input clock divider, set **TDCTL.DTGCLK** to 1b.

22.4.2 Timer Counter

The base timer is a 16-bit timer that can count either up, or up then down to support both edge and center aligned and asymmetric PWM output types.

The timer period is stored in the **TDPRD** register. The current value of the timer counter is updated at every timer tick and is stored in the **TDCTR** register. The **TDCTR** register is a RW register, so it may be updated at any time and is changed immediately.

The **TDPRD** register and all of the **TDCCTRn** registers have shadow copies that are updated at a user-specified time. See the section below on Base Timer **TDPRD** Latching and CCR Timer Latching for more information.

When **TDCTL.MODE** = 00b, the timer is disabled. Even when disabled, all timer registers are accessible via the APB bus.

To enable the timer, set the **TDCTL.MODE** to 01b (up mode), 10b (up/down mode) or 11b (up/down asymmetric mode).

When **TDCTL.MODE** = 01b, the timer is configured in up mode. The timer will count from 0 to **TDPRD**. The counter is updated in **TDCTR** at every timer tick. If auto-reload is active (**TDCTL.SINGLE** = 0b), then the timer count will automatically count from **TDPRD** to 0 and continue counting up. If auto-reload is not active (**TDCTL.SINGLE** = 1b), then the timer will count from 0 to **TDPRD** and stop. When this happens, the timer will set the **TDCTL.MODE** to 00b (disabled).

When the timer is configured for up/down mode (**TDCTL.MODE** = 10b) or configured for up/down asymmetric mode (**TDCTL.MODE** = 11b), the timer will count from 0 to **TDPRD** and then back down to 0. If auto-reload is not active (**TDCTL.SINGLE** = 1b), the time timer will stop and set **TDCTL.MODE** to 00b (disabled). If auto-reload is active (**TDCTL.SINGLE** = 0b), then the timer count will then continue counting back up after it reaches 0.

22.4.3 Up Mode

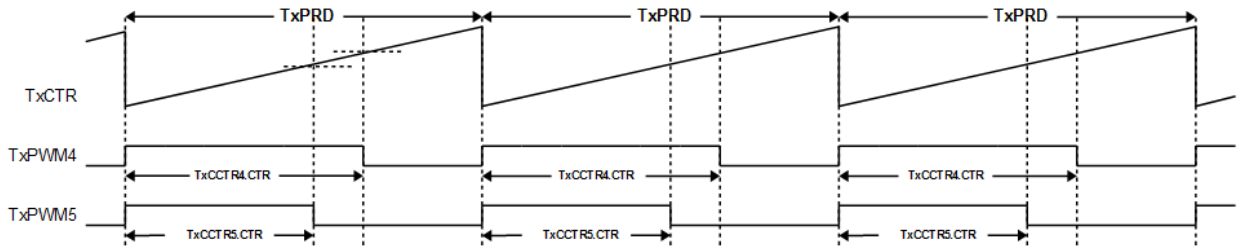
When the **TDCTL.MODE** is set to 01b (up mode) the timer will begin counting up from 0. The **TDCTR** will count to the value of **TDPRD** and then will reset to 0.

If the timer is configured for auto-reload mode (**TDCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it reaches 0.

The **TDCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate an edge aligned PWM as shown below.

Figure 22-3 Up Mode PWM Waveform



In this mode, when **TDCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TDCTR** register reaches the **TDCCTRn.CTR** threshold. When the **TDCTR** counts from **TDPRD** to 0, the PWM output is transitioned to high.

22.4.4 Up/Down Mode

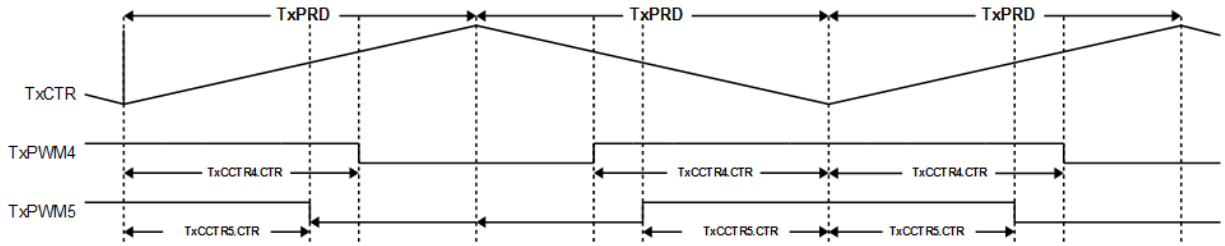
When the **TDCTL.MODE** is set to 10b (up/down mode) the timer will begin counting up from 0. The **TDCTR** will count to the value of **TDPRD** and then will begin counting back down to 0.

If the timer is configured for auto-reload mode (**TDCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it counts from 1 to 0.

The **TDCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate a center aligned PWM as shown below.

Figure 22-4 Up/Down Mode PWM Waveform



In this mode, when **TDCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TDCTR** register reaches the **TDCCTRn.CTR** threshold. When the **TDCTR** counts from **TDPRD** - 1 to **TDPRD**, it will begin counting down. When the **TDCTR** counts down to **TDCCTRn.CTR** the PWM output is transitioned to high.

22.4.5 Up/Down Asymmetric mode

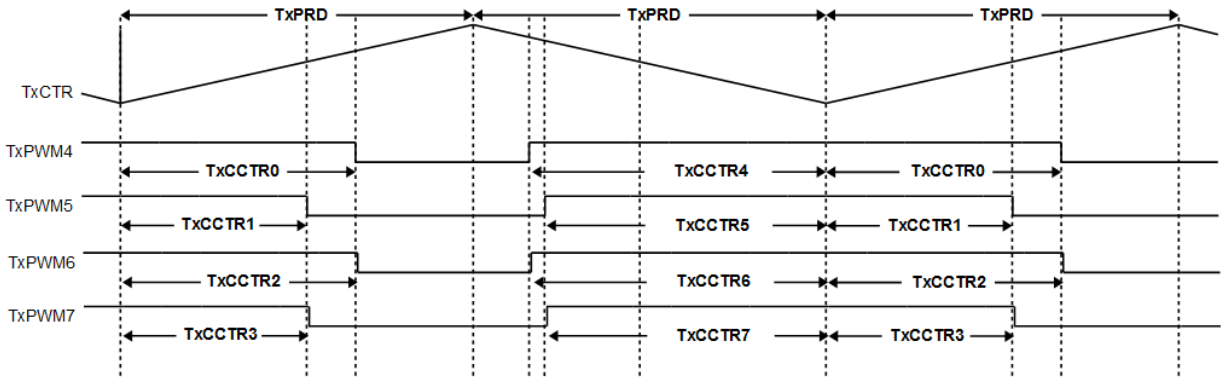
When the **TDCTL.MODE** is set to 11b (up/down asymmetric mode) the timer will begin counting up from 0. The **TDCTR** will count to the value of **TDPRD** and then will begin counting back down to 0.

If the timer is configured for auto-reload mode (**TDCTL.SINGLE** = 0b), then the timer will continue counting up on the next timer clock cycle. If the timer is not configured for auto-reload, then the timer will stop when it counts from 1 to 0.

The **TDCTR** register may be read or written at any time. This feature may be used to “pre-load” the timer counter with a user-specified value before it is enabled and begins counting.

This timer mode can generate a center aligned asymmetric PWM as shown below.

Figure 22-5 Up/Down Asymmetric Mode PWM Waveform



In this mode, when **TDCTR** starts counting from 0, the PWM output is high. The PWM output is transitioned to low then the **TDCTR** register reaches the **TDCCTRn.CTR** threshold. When the

TDCTR counts from **TDPRD** - 1 to **TDPRD**, it will begin counting down. When the **TDCTR** counts down to **TDCCTRn.CTR** the PWM output is transitioned to high.

In this mode, the CCR outputs are generated such that 2 **TDCCTRn.CTR** registers are used to generate a single PWM output, so that the center-aligned PWM can be generated with asymmetric on-time.

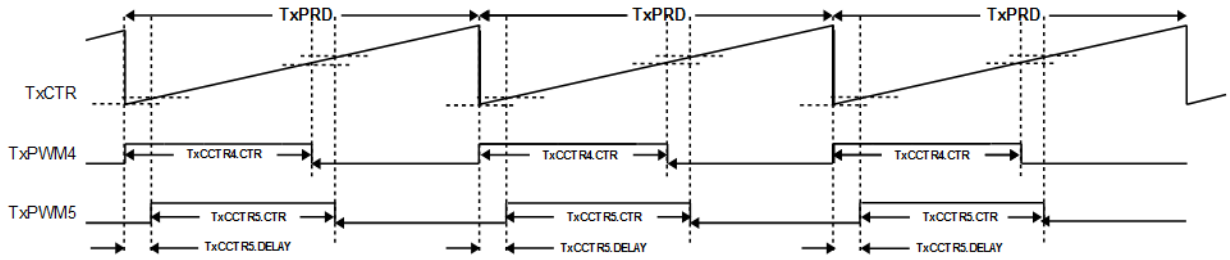
In this mode, the PWM outputs of the timer are generated as follows:

- TDPWM4 uses **TDCCTR0.CTR** for the first portion of the on time and **TDCCTR4.CTR** for the second portion of the on time
- TDPWM5 uses **TDCCTR1.CTR** for the first portion of the on time and **TDCCTR5.CTR** for the second portion of the on time
- TDPWM6 uses **TDCCTR2.CTR** for the first portion of the on time and **TDCCTR6.CTR** for the second portion of the on time
- TDPWM7 uses **TDCCTR3.CTR** for the first portion of the on time and **TDCCTR7.CTR** for the second portion of the on time

22.4.6 Up Mode with Phase Delay

When the timer is configured for up mode (**TDMODE** = 01b) the CCR allows a delay to be applied to the output PWM period, to support phase delay control topologies. If the value of **TDCCTRn.DELAY** > 0, then the CCR will delay this number of ticks before starting the PWM period (transitioning the output signal from high to low).

Figure 22-6 Up Mode with Phase Delay



22.4.7 Timer Synchronization

The timer peripheral allows synchronization between the timers in the PAC55XX. The timers may be synchronized as follows:

- Timer A (master), Timer B (slave)
- Timer A (master), Timer B (slave), Timer C (slave)
- Timer A (master), Timer B (slave), Timer C (slave), Timer D (slave)

The timer master's SYNC_OUT signals are connected to the downstream timer slave's SYNC_IN signal.

When configured, the timers can synchronize their clocks so they are sharing the same time base. In order to do this, the timers must be configured with the same clock source (**TDCTL.CLKSRC**) and clock divider (**TDCTL.CLKDIV**).

To configure the timers for synchronization, follow these steps:

- While the timer master is disabled (**TDCTL.MODE** = 00b), set the timer master **TxCTL.SSYNC** = 0b
- For each of the downstream timer slaves, while the timer is disabled **TDCTL.MODE** = 00b, set the **TDCTL.SSYNC** to 1b. Make sure that **TDCTL.CLKDIV** and **TDCTL.CLKSRC** for each of the timer slaves is the same as the timer master.
- Enable each of the slave timers by setting **TDCTL.MODE** to 01b (up), 10b (up/down) or 11b (up/down asymmetric). The slave timers will not start counting since the **TDCTL.SSYNC** is set to a 1b.
- Enable the timer master by setting **TxCTL.MODE** to 01b (up), 10b (up/down) or 11b (up/down asymmetric).

At this point, the master and all configured timer slaves will begin counting.

22.4.8 Base Timer TAPRD Latching

The timer peripherals have a shadow copy of the **TDPRD** register that is used for the counting operations. The latching of the data from the **TDPRD** register to the shadow register is controlled by the timer configuration.

There are several options for latching of this register into the shadow copy. This behavior is controlled by the **TDCTL.PRDLATCH** setting and depends on the setting of the **TDCTL.MODE** register as well.

22.4.8.1 TDPRD Latch When TCCTR = 0

- If **TDCTL.MODE** = 01b (up mode) and **TDCTL.PRDLATCH** = 00b (**TDCTR** = 0), then the **TDPRD** register is copied into the shadow register when **TDCTR** counts from **TCPRD** to 0.
- If **TDCTL.MODE** = 10b (up/down mode) or if **TCCTL.MODE** = 11b (up/down asymmetric mode) and **TCCTL.PRDLATCH** = 00b (**TCCTR** = 0), then the **TCPRD** register is copied into the shadow register when the **TCCTR** counts from 1 to 0.

22.4.8.2 TDPRD Latch When TCCTR = TCPRD

- If **TDCTL.MODE** = 01b (up mode) or if **TDCTL.MODE** = 10b (up/down mode) or if **TDCTL.MODE** = 11b (up/down asymmetric mode) and **TDCTL.PRDLATCH** = 01b

(**TDCTR** = **TBPRD**), then the **TDPRD** register is copied into the shadow register when **TDCTR** counts from **TDPRD** – 1 to **TDPRD**.

22.4.8.3 TDPRD Latch Immediate

- If **TDCTL.PRDLATCH** = 11b (immediate), then the **TDPRD** register is copied into the shadow register as soon as the **TDPRD** register is written.

22.4.9 CCR Timer Latching

When in compare mode, the CCR units support configurable latching of the **TDCCRn.CTR** values into a shadow register that is used for timer operation. The shadow register is used for generating the PWM output when the CCR is in compare mode.

This feature can be used to control when the new duty cycle is applied, during the PWM period.

When the CCR is in compare mode (**TDCCTLn.CCMODE** = 0b), the shadow register is latched as follows:

- If **TDCCTLn.CCLATCH** = 00b (**TDCTR** = 0)
  - If **TDCTL.MODE** = 01b (up mode), **TDCCTRn.CTR** is copied into the shadow register when **TDCTR** counts from **TDPRD** to 0. If **TDCTL.MODE** = 10b (up/down mode) or 11b (up/down asymmetric mode), **TDCCTRn.CTR** is copied into the shadow register when **TDCTR** counts from 1 to 0.
- If **TDCCTLn.CCLATCH** = 01b (**TDCTR = TCPRD**), then the **TDCCTRn.CTR** is copied into the shadow register when **TDCTR** counts from **TDPRD** – 1 to **TDPRD**.
- If **TDCCTLn.CCLATCH** = 10b (latch immediate), then the **TDCCTRn.CTR** is copied into the shadow register as soon as it is written.

When the CCR is in capture mode (**TDCCTLn.CCMODE** = 1b), the configuration of **TDCCTLn.CCLATCH** controls when the **TDCTR** is copied into the **TDCCTRn.CTR** register.

- If **TDCCTLn.CCLATCH** = 00b (rising edge), then the value of **TDCTR** is copied into the **TDCCTLn.CTR** register upon a rising edge in the input signal.
- If **TDCCTLn.CCLATCH** = 01b (falling edge), then the value of the **TDCTR** is copied into the **TDCCTLn.CTR** register upon a falling edge in the input signal.
- If **TDCCTLn.CCLATCH** = 10b (both), then the value of the **TDCTR** is copied into the **TDCCTLn.CTR** register on both a rising and falling edge of the input signal.

22.4.10 Timer Whole Latching

It is sometimes convenient to latch both the **TDPRD** (period) and all **TDCCTRn.CTR** (duty cycle) registers into the shadow registers at one time.

If the **TDCTL.LATCH** bit is written to a 1b, then all the **TDPRD** and all **TDCCTRn.CTR** registers will be copied into the shadow registers on the same clock cycle.

The **TDCTL.LATCH** bit is self-clearing, and will always be read as a 0b.

22.4.11 Inverting CCR PWM Output

When the CCR is in compare mode (**TDCCTRn.CCMODE** = 0b), the output of the CCR (the input to the DTG) may be inverted. To invert this output, set the **TDCCTRn.CCOUTINV** = 1b.

This mode is useful for some control topologies, especially full-bridge.

22.4.12 Base Timer Interrupts

The base timer may be configured to generate an interrupt to the NVIC.

If **TDCTL.MODE** = 01b (up mode) and **TDCTR** counts from **TDPRD** to 0 then the **TCINT.BASEIF** bit is set to 1b. If the **TDCTL.BASEIE** interrupt enable bit is set to a 1b, then the TIMERD_IRQ signal to the NVIC is asserted.

If **TDCTL.MODE** = 10b (up/down mode) or 11b (up/down asymmetric mode) and **TDCTR** counts from 1 to 0, then the **TDINT.BASEIF** bit is set to 1b. If the **TDCTL.BASEIE** interrupt enable bit is set to a 1b, then the TIMERD_IRQ signal to the NVIC is asserted.

The **TDINT.BASEIF** may be cleared by writing a 1b to it.

22.4.13 CCR Compare Interrupts

When **TxCCTLn.CCMODE** = 0b (compare mode), the CCR unit may be configured to generate an interrupt to the NVIC.

When the PWM edge transitions, then the CCR interrupt flag in the **TxINT** register is set as follows:

- If **TxCCTLn.CCINTEDGE** = 00b (rising edge) and a rising edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and TIMER_IRQ to the NVIC is asserted.
- If **TxCCTLn. CCINTEDGE** = 01b (falling edge) and a falling edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and TIMER_IRQ to the NVIC is asserted. **TxINT.CCRnIF** may be cleared by writing it to a 1b.
- If **TxCCTLn. CCINTEDGE** = 10b (rising or falling edge) and a rising or falling edge is detected in the TxPWM output signal, then the **TxINT.CCRnIF** interrupt flag is set and

TIMER_IRQ to the NVIC is asserted. **TxINT.CCRnIF** may be cleared by writing it to a 1b.

TxINT.CCRnIF may be cleared by writing it to a 1b.

22.4.14 CCR Capture Interrupts

When **TxCCTLn.CCMODE** = 1b (capture mode), the CCR unit may be configured to generate an interrupt to the NVIC based on the input signal.

When the PWM edge transitions, then the CCR interrupt flag in the **TxINT** register is set as follows:

- If **TxCCTLn.CCINTEDGE** = 00b (rising edge) and a rising edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is set.
- If **TxCCTLn.CCINTEDGE** = 01b (falling edge) and a falling edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is set.
- If **TxCCTLn.CCINTEDGE** = 10b (rising or falling edge) and a rising or falling edge is detected in the TxPWM input signal, then the **TxINT.CCRnIF** interrupt flag is.

If the **TxINT.CCRnIF** interrupt flag is set and the corresponding **TxCCTLn.CCINTEN** bit is set, then the TIMERA_IRQ signal to the NVIC will be asserted.

The **TxINT.CCRnIF** interrupt flag can be cleared by writing it to a 1b.

22.4.15 Timer IRQ signal

The timer unit has one IRQ output signal to the NVIC (TIMERD_IRQ).

This signal is asserted when any of the following conditions are true:

- **TDINT.BASEIF** = 1b and **TDCTL.BASEIE** = 1b
- **TDINT.CCR0IF** = 1b and **TDCCTL0.CCINTEN** = 1b
- **TDINT.CCR1IF** = 1b and **TDCCTL1.CCINTEN** = 1b
- **TDINT.CCR2IF** = 1b and **TDCCTL2.CCINTEN** = 1b
- **TDINT.CCR3IF** = 1b and **TDCCTL3.CCINTEN** = 1b
- **TDINT.CCR4IF** = 1b and **TDCCTL4.CCINTEN** = 1b
- **TDINT.CCR5IF** = 1b and **TDCCTL5.CCINTEN** = 1b
- **TDINT.CCR6IF** = 1b and **TDCCTL6.CCINTEN** = 1b
- **TDINT.CCR7IF** = 1b and **TDCCTL7.CCINTEN** = 1b

The TIMERD_IRQ signal is de-asserted when all of the above conditions are false.

22.4.16 Skipping CCR Interrupts

Sometimes it is useful to not generate a CCR interrupt to the NVIC every time the **TDCTR** counts to the **TDCCTRn.CTR** value. For example, in a control application when the PWM frequency is fast, but you only need a MCU interrupt every 5 PWM periods.

Each CCR in the timer has the ability to skip interrupts to the NVIC to allow this capability.

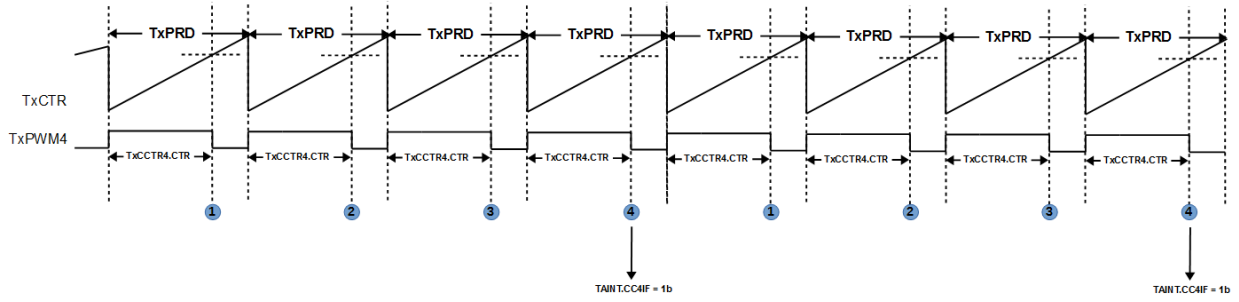
The CCR will generate interrupts to the NVIC according to the **TDCCTL.CCINTSKIP** field as follows:

- If **TDCCTLn.CCINTSKIP** = 0000b, then the CCR will generate an interrupt each time
- If **TDCCTLn.CCINTSKIP** = 0001b, then the CCR will skip 1 CCR cycle before generating an interrupt to the NVIC.
- If **TDCCTLn.CCINTSKIP** = 0010b, then the CCR will skip 2 CCR cycles before generating an interrupt to the NVIC.
- If **TDCCTLn.CCINTSKIP** = 0011b, then the CCR will skip 3 CCR cycles before generating an interrupt to the NVIC.
- …

- If **TDCCTLn.CCINTSKIP** = 1110b, then the CCR will skip 14 CCR cycles before generating an interrupt to the NVIC.
- If **TDCCTLn.CCINTSKIP** = 1111b, then the CCR will skip 15 CCR cycles before generating an interrupt to the NVIC.

The diagram below shows an example of a CCR4 with interrupt skipping configured with a value of 4 (**TDCCTL.CCINTSKIP** = 4).

Figure 22-7 CCR Interrupt Skipping Diagram



22.4.17 Timer ADC Triggers

Each timer CCR unit may be configured to be an ADC trigger, to allow the DTSE to automatically begin conversion sequences.

For more information on this feature, see the section on the ADC and DTSE in this user guide.

22.4.18 Dead-Time Generators (DTG)

Each timer has 4 DTG units. Each DTG unit is capable of generating a pair of complementary signals that can be configured with dead-time to drive an inverter for half-bridge topologies.

The input clock to the DTG can be configured to be the timer clock before or after the **TDCTL.CLKDIV** input clock divider. If **TDCTL.DTGCLK** is 0b, then the DTGCLK is the clock before the **TDCTL.CLKDIV** clock divider. If **TDCTL.DTGCLK** is 1b, then the DTGCLK is the clock after the **TDCTL.CLKDIV** clock divider.

When **TDDTGCTLn.DTEN** = 0b, the DTG is disabled (bypassed). In this mode, the timer signals have the following behavior (see the simplified block diagram above):

DTG0:

- CCR0 output is connected to TDPWM0
- CCR4 output is connected to TDPWM4

DTG1:

- CCR1 output is connected to TDPWM1
- CCR5 output is connected to TDPWM5

DTG2:

- CCR2 output is connected to TDPWM2
- CCR6 output is connected to TDPWM6

DTG3:

- CCR3 output is connected to TDPWM3
- CCR7 output is connected to TDPWM7

When **TDDTGCTL0.DTEN** = 1b, the DTG is enabled. In this mode, only one CCR output is used to generate the two complementary outputs. In this mode, the timer signals have the following behavior:

DTG0:

- CCR0 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR4 output is used to generate the two complementary output signals: TDPWM0 and TDPWM4.

- TDPWM0 becomes the low-side complementary signal.
- TDPWM4 becomes the high-side complementary signal.
- The **TDDTGCTL0.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, to when the low-side signal rises).
- The **TDDTGCTL0.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, to when the high-side signal rises).

DTG1:

- CCR1 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR5 output is used to generate the two complementary output signals: TDPWM1 and TDPWM5.
- TDPWM1 becomes the low-side complementary signal.
- TDPWM5 becomes the high-side complementary signal.
- The **TDDTGCTL1.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TDDTGCTL1.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

DTG2:

- CCR2 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR6 output is used to generate the two complementary output signals: TDPWM2 and TDPWM6.
- TDPWM2 becomes the low-side complementary signal.
- TDPWM6 becomes the high-side complementary signal.
- The **TDDTGCTL2.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TDDTGCTL2.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

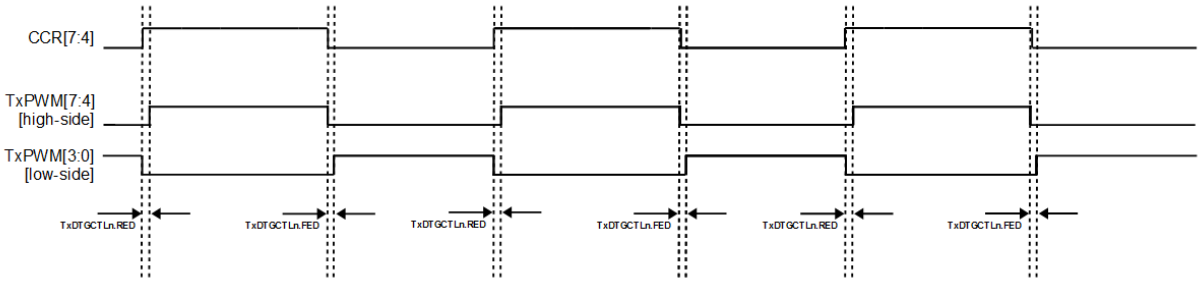
DTG3:

- CCR3 output is unused. This CCR output may still be used to generate CCR interrupts or ADC timer triggers, but will not be connected to the Digital Peripheral MUX.
- CCR7 output is used to generate the two complementary output signals: TDPWM3 and TDPWM4.
- TDPWM3 becomes the low-side complementary signal.
- TDPWM7 becomes the high-side complementary signal.
- The **TDDTGCTL3.FED** specifies the number of DTGCLK ticks that are applied for falling edge delay (the delay from when the high-side signal falls, and when the low-side signal rises).
- The **TDDTGCTL0.RED** specifies the number of DTGCLK ticks that are applied for rising edge delay (the delay from when the low-side signals falls, and when the high-side signal rises).

The PWM waveform below shows an example of the dead-time applied to the signals through the DTG units.

22.4.19 Timer D DTG Output

Figure 22-8 Timer D DTG Output



22.5 QEP

22.5.1 Overview

Each PAC55XX PWM Timer contains a QEP decoder that can be used to determine the position, speed, and direction of a motor with a QEP encoder.

The inputs to the QEP peripheral are TxQEPPHA, TxQEPPHB and TxQEPIDX (phase A, phase B and index). These three inputs have a glitch filter applied to them, to filter out noise and are used for calculation of the motor position, speed, and direction.

22.5.2 Configuration

The QEP peripheral can be enabled by setting the **TxQEPCTL.QEPEN** bit to a 1b.[20] The input clock to the QEP peripheral may be configured between /1 and /128 using the **TxQEPCTL.CLKDIV** field.

If the **TxQEPCTL.CNTAB** bit is 0b, only phase A edges are processed. If the **TxQEPCTL.CNTAB** bit is set to a 1b, then both phase A and phase B edges are processed. If the **TxQEPCTL.CNTEDGE** bit is set to 0b, then only the rising edges are processed. If the **TxQEPCTL.CNTEDGE** bit is set to a 1b, then both rising and falling edges are processed.

If the direction is determined to be clockwise (CW), then the **TxQEPCTL.DIR** bit is set to 0b. If the direction is determined to be counterclockwise (CCW), then the **TxQEPCTL.DIR** bit is set to 1b. The direction will get updated only on edges that are processed.

When an edge is processed, the QEP peripheral will first determine the direction and then increment or decrement the counter value in **TxQEPCTR.CTR** accordingly. **TxQEPCTR.CTR** may be written at any time.

If an edge is processed while the motor is turning clockwise, the counter value in **TxQEPCTR.CTR** is incremented. If the value of **TxQEPCTR.CTR** was **TxQEPMAX.CTRMAX** during this event, then **TxQEPCTR.CTR** is set to 0.

If an edge is processed while the motor is turning counterclockwise, the counter value in **TxQEPCTR.CTR** is decremented. If the value of **TxQEPCTR.CTR** was 0 during this event, then **TxQEPCTR.CTR** is set to **TxQEPMAX.CTRMAX**.

[20] Before enabling the QEP peripheral, be sure to set up the Digital Peripheral MUX, so that no false edges are detected by the QEP state machine.

22.5.3 Index Reset

If the **TxQEPCTL.IDXRST** bit is set to a 1b, then an index event clears the **TxQEPCTR.CTR** counter to 0. If the motor is turning in the clockwise direction during this event, **TxQEPCTR.CTR** is set to 0. If the motor is turning in the counterclockwise direction during this event the **TxQEPCTR.CTR** is set to 0 (if not coincident with the phase edge) or **TxQEPMAX.CTRMAX** (if coincident with the phase edge).

If this bit is set to a 0b, then an index event does not reset the **TAQEPCTL.TICKS** counter.

22.5.4 Interrupts

If a change to the direction is detected (0 to 1 or 1 to 0), then the **TxQEPIF.DIRIF** bit is set to a 1b. If the **TxQEPIER.DIRIE** bit is set to 1b, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.DIRIF** bit can be cleared by writing it to 1b.

When a rising edge on phase A is detected the **TxQEPIF.PHAIF** bit is set to 1b. If the **TxQEPIER.PHAIE** bit is set and the **TxQEPIF.PHAIF** bit is set, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.PHAIF** bit can be cleared by writing a 1b to it.

When a rising edge on phase B is detected the **TxQEPIF.PHBIF** bit is set. If the **TxQEPIER.PHBIE** bit is set and the **TxQEPIF.PHBIF** bit is set, then the QEPx_IRQ signal is asserted to the NVIC. The **TxQEPIF.PHBIF** bit can be cleared by writing a 1b to it.

If an overflow or underflow in the **TxQEPCTR.CTR** is detected, the **TxQEPIF.WRIF** bit is set (counter wrap interrupt flag). If the **TxQEPIER.WRIE** bit is set and the **TxQEPIF.WRIF** bits are set, then the QEPx_IRQ signal is asserted to the NVIC.

If an index event is detected, the **TxQEPIF.IDXEVIF** bit is set. If the **TxQEPIER.IDXEVIE** bit is set and the **TxQEPIF.IDXEVIF** bit is set, then the QEPx_IRQ signal to the NVIC is asserted. The **TxQEPIF.IDXEVIF** bit can be cleared by writing a 1b to it.

22.6 Peripheral IO Mapping

The Timer D peripheral signal inputs and outputs (CCR, QEP) are connected to the digital peripheral MUX so they can be used by various IO pins.

The table below shows which peripheral outputs may be connected to which IO pins.

Table 22-1 Timer D Peripheral IO Mapping

TIMER SIGNAL	IO PIN
TDPWM0	PE0, PF0, PG0
TDPWM1	PE1, PF1, PG1
TDPWM2	PE2, PF2, PG2
TDPWM3	PE3, PF3, PG3
TDPWM4	PE4, PF4, PG4
TDPWM5	PE5, PF5, PG5
TDPWM6	PE6, PF6, PG6
TDPWM7	PE7, PF7
TDQEPIDX	PD0, PD4, PE4, PG4, PG7
TDQEPPHA	PD1, PD5, PE5,PG5
TDQEPPHB	PD2, PD6, PE6, PG6

For more information on how to use the Digital Peripheral MUX to connect peripheral signals to IO, see GPIO and DPM.

22.7 Register Summary

Table 22-2 Timer D Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
TDCTL	4009 0000h	Timer D control	RW	0000 0000h
TDINT	4009 0004h	Timer D interrupt control	RW	0000 0000h
TDPRD	4009 0008h	Timer D period	RW	0000 0000h
TDCTR	4009 000Ch	Timer D counter	RW	0000 0000h
TDCCTL0	4009 0100h	Timer D CC control 0	RW	0000 0000h
TDCCTR0	4009 0104h	Timer D CC counter 0	RW	0000 0000h
TDCCTL1	4009 0108h	Timer D CC control 1	RW	0000 0000h
TDCCTR1	4009 010Ch	Timer D CC counter 1	RW	0000 0000h
TDCCTL2	4009 0110h	Timer D CC control 2	RW	0000 0000h
TDCCTR2	4009 0114h	Timer D CC counter 2	RW	0000 0000h
TDCCTL3	4009 0118h	Timer D CC control 3	RW	0000 0000h
TDCCTR3	4009 011Ch	Timer D CC counter 3	RW	0000 0000h
TDCCTL4	4009 0120h	Timer D CC control 4	RW	0000 0000h
TDCCTR4	4009 0124h	Timer D CC counter 4	RW	0000 0000h
TDCCTL5	4009 0128h	Timer D CC control 5	RW	0000 0000h
TDCCTR5	4009 012Ch	Timer D CC counter 5	RW	0000 0000h
TDCCTL6	4009 0130h	Timer D CC control 6	RW	0000 0000h
TDCCTR6	4009 0134h	Timer D CC counter 6	RW	0000 0000h
TDCCTL7	4009 0138h	Timer D CC control 7	RW	0000 0000h
TDCCTR7	4009 013Ch	Timer D CC counter 7	RW	0000 0000h
TDDTGCTL0	4009 0200h	Timer D DTG control 0	RW	0000 0000h
TDDTGCTL1	4009 0204h	Timer D DTG control 1	RW	0000 0000h
TDDTGCTL2	4009 0208h	Timer D DTG control 2	RW	0000 0000h
TDDTGCTL3	4009 020Ch	Timer D DTG control 3	RW	0000 0000h
TDQEPCTL	4009 0300h	Timer D QEP control	RW	0000 0000h
TDQEPCTR	4009 0304h	Timer D QEP counter	RW	0000 0000h
TDQEPMAX	4009 0308h	Timer D QEP max counter value	RW	0000 FFFFh
TDQEPIER	4009 030Ch	Timer D QEP interrupt enable	RW	0000 0000h
TDQEPIFR	4009 0310h	Timer D QEP interrupt flag	RW	0000 0000h

22.8 Register Detail

22.8.1 TDCTL

Register 22-1 TDCTL (Timer D Control, 4009 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:14	**Reserved**	RO	0	Reserved
13	**BASEIE**	RW	0	Base timer interrupt enable: 0b: not enabled 1b: enabled
12	**CLR**	RW	0	Base timer clear: 0b: do not clear 1b: clear timer counter
11	**LATCH**	RW	0	When written to 1b, this will latch the **TDPRD** and all **TDCCTRn** registers on the same clock cycle. Writing this bit to 0b has no effect. This is a self-clearing bit.
10	**DTGCLK**	RW	0	DTG Clock Source: 0b: Before input clock divider 1b: After input clock divider
9	**CLKSRC**	RW	0	Timer Clock Source: 0b: PCLK 1b: ACLK
8:6	**CLKDIV**	RW	0	Timer Input Clock Divider: 000b: /1 001b: /2 010b: /4 011b: /8 100b: /16 101b: /32 110b: /64 111b: /128
5	**SINGLE**	RW	0	Single Shot Timer: 0b: disabled (auto-reload) 1b: enabled (single shot timer)
4	**SSYNC**	RW	0	Timer Slave Synchronization: 0b: disabled 1b: enabled
3:2	**PRDLATCH**	RW	0	Timer Period Latch Mode: 00b: Latch **TDPRD** when **TDCTR** = 0 01b: Latch **TDPRD** when **TDCTR** = **TDPRD** 10b: Latch **TDPRD** immediately upon register write 11b: reserved
1:0	**MODE**	RW	0	Timer Mode: 00b: disabled 01b: up mode 10b: up/down mode 11b: asymmetric mode

22.8.2 TDINT

Register 22-2 TDINT (Timer D Interrupt Control, 4009 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:9	**Reserved**	RO	0	Reserved
8	**BASEIF**	W1C	0	Base timer interrupt flag: 0b: no interrupt flag 1b: interrupt flag
7	**CCR7IF**	W1C	0	CCR7 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
6	**CCR6IF**	W1C	0	CCR6 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
5	**CCR5IF**	W1C	0	CCR5 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
4	**CCR4IF**	W1C	0	CCR4 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
3	**CCR3IF**	W1C	0	CCR3 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
2	**CCR2IF**	W1C	0	CCR2 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
1	**CCR1IF**	W1C	0	CCR1 interrupt flag: 0b: no interrupt flag 1b: interrupt flag
0	**CCR0IF**	W1C	0	CCR0 interrupt flag: 0b: no interrupt flag 1b: interrupt flag

22.8.3 TCPRD

Register 22-3 TDPRD (Timer D Period, 4009 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**PERIOD**	RW	0	Timer period value

22.8.4 TDCTR

Register 22-4 TDCTR (Timer D Counter, 4009 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**COUNTER**	RW	0	Timer counter

22.8.5 TDCCTL0

Register 22-5 TDCCTL0 (Timer D CCR Control 0, 4009 0100h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TDCCTL0.CCMODE** = 0b (compare): 00b: Latch **TDCCTR0** registers when **TDCTR** = 0 01b: Latch **TDCCTR0** registers when **TDCTR** = **TDPRD** 10b: Latch **TDCCTR0** registers immediately 11b: Reserved If **TDCCTL0.CCMODE** = 1b (capture): 00b: Latch **TDCTR** into **TDCCTR0.CTR** on rising edge 01b: Latch **TDCTR** into **TDCCTR0.CTR** on falling edge 10b: Latch **TDCTR** into **TDCCTR0.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TDCCTL0.CCMODE** = 1b (capture), set **TDINT.CCR0IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

22.8.6 TDCCTR0

Register 22-6 TDCCTR0 (Timer D CCR Counter 0, 4009 0104h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TDCCTL0.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TDCCTRL0.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TDCCTL0.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

22.8.7 TDCCTL1

Register 22-7 TDCCTL1 (Timer D CCR Control 1, 4009 0108h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TDCCTL1.CCMODE** = 0b (compare): 00b: Latch **TDCCTR1** registers when **TDCTR** = 0 01b: Latch **TDCCTR1** registers when **TDCTR** = **TDPRD** 10b: Latch **TDCCTR1** registers immediately 11b: Reserved If **TDCCTL1.CCMODE** = 1b (capture): 00b: Latch **TDCTR** into **TDCCTR1.CTR** on rising edge 01b: Latch **TDCTR** into **TDCCTR1.CTR** on falling edge 10b: Latch **TDCTR** into **TDCCTR1.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TDCCTL1.CCMODE** = 1b (capture), set **TDINT.CCR1IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

22.8.8 TDCCTR1

Register 22-8 TDCCTR1 (Timer D CCR Counter 1, 4009 010Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TDCCTL1.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TDCCTRL1.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TDCCTL1.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

22.8.9 TDCCTL2

Register 22-9 TDCCTL2 (Timer D CCR Control 2, 4009 0110h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TDCCTL2.CCMODE** = 0b (compare): 00b: Latch **TDCCTR2** registers when **TDCTR** = 0 01b: Latch **TDCCTR2** registers when **TDCTR** = **TDPRD** 10b: Latch **TDCCTR2** registers immediately 11b: Reserved If **TDCCTL2.CCMODE** = 1b (capture): 00b: Latch **TDBCTR** into **TDCCTR2.CTR** on rising edge 01b: Latch **TDCTR** into **TDCCTR2.CTR** on falling edge 10b: Latch **TDCTR** into **TDCCTR2.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TDCCTL2.CCMODE** = 1b (capture), set **TDINT.CCR2IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

22.8.10 TDCCTR2

Register 22-10 TDCCTR2 (Timer D CCR Counter 2, 4009 0114h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TDCCTL2.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TDCCTRL2.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TDCCTL2.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

22.8.11 TDCCTL3

Register 22-11 TDCCTL3 (Timer D CCR Control 3, 4009 0118h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TDCCTL3.CCMODE** = 0b (compare): 00b: Latch **TDCCTR3** registers when **TDCTR** = 0 01b: Latch **TDCCTR3** registers when **TDCTR** = **TDPRD** 10b: Latch **TDCCTR3** registers immediately 11b: Reserved If **TDCCTL3.CCMODE** = 1b (capture): 00b: Latch **TDCTR** into **TDCCTR3.CTR** on rising edge 01b: Latch **TDCTR** into **TDCCTR3.CTR** on falling edge 10b: Latch **TDCTR** into **TDCCTR3.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TDCCTL3.CCMODE** = 1b (capture), set **TDINT.CCR3IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

22.8.12 TDCCTR3

Register 22-12 TDCCTR3 (Timer D CCR Counter 3, 4009 011Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TDCCTL3.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TDCCTRL3.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TDCCTL3.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

22.8.13 TDCCTL4

Register 22-13 TDCCTL4 (Timer D CCR Control 4, 4009 0120h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TDCCTL4.CCMODE** = 0b (compare): 00b: Latch **TDCCTR4** registers when **TDCTR** = 0 01b: Latch **TDCCTR4** registers when **TDCTR** = **TDPRD** 10b: Latch **TDCCTR4** registers immediately 11b: Reserved If **TDCCTL4.CCMODE** = 1b (capture): 00b: Latch **TDCTR** into **TDCCTR4.CTR** on rising edge 01b: Latch **TDCTR** into **TDCCTR4.CTR** on falling edge 10b: Latch **TDCTR** into **TDCCTR4.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TDCCTL4.CCMODE** = 1b (capture), set **TDINT.CCR4IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

22.8.14 TDCCTR4

Register 22-14 TDCCTR4 (Timer D CCR Counter 4, 4009 0124h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TDCCTL4.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TDCCTRL4.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TDCCTL4.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

22.8.15 TDCCTL5

Register 22-15 TDCCTL5 (Timer D CCR Control 5, 4009 0128h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TDCCTL5.CCMODE** = 0b (compare): 00b: Latch **TDCCTR5** registers when **TDCTR** = 0 01b: Latch **TDCCTR5** registers when **TDCTR** = **TDPRD** 10b: Latch **TDCCTR5** registers immediately 11b: Reserved If **TDCCTL5.CCMODE** = 1b (capture): 00b: Latch **TDCTR** into **TDCCTR5.CTR** on rising edge 01b: Latch **TDCTR** into **TDCCTR5.CTR** on falling edge 10b: Latch **TDCTR** into **TDCCTR5.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TDCCTL5.CCMODE** = 1b (capture), set **TDINT.CCR5IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

22.8.16 TDCCTR5

Register 22-16 TDCCTR5 (Timer D CCR Counter 5, 4009 012Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TDCCTL5.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TDCCTRL5.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TDCCTL5.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

22.8.17 TDCCTL6

Register 22-17 TDCCTL6 (Timer D CCR Control 6, 4009 0130h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TDCCTL6.CCMODE** = 0b (compare): 00b: Latch **TDCCTR6** registers when **TDCTR** = 0 01b: Latch **TDCCTR6** registers when **TDCTR** = **TDPRD** 10b: Latch **TDCCTR6** registers immediately 11b: Reserved If **TDCCTL6.CCMODE** = 1b (capture): 00b: Latch **TDCTR** into **TDCCTR6.CTR** on rising edge 01b: Latch **TDCTR** into **TDCCTR6.CTR** on falling edge 10b: Latch **TDCTR** into **TDCCTR6.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TDCCTL6.CCMODE** = 1b (capture), set **TDINT.CCR6IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

22.8.18 TDCCTR6

Register 22-18 TDCCTR6 (Timer D CCR Counter 6, 4009 0134h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TDCCTL6.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TDCCTRL6.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TDCCTL6.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

22.8.19 TDCCTL7

Register 22-19 TDCCTL7 (Timer D CCR Control 7, 4009 0138h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:8	**CCINTSKIP**	RW	0	CC Interrupt Skip Counter: 0000b: Don't skip CCR matches before interrupt 0001b: Skip 1 CCR match before interrupt 0010b: Skip 2 CCR matches before interrupt 0011b: Skip 3 CCR matches before interrupt … 1110b: Skip 14 CCR matches before interrupt 1111b: Skip 15 CCR matches before interrupt
7	**CCFORCE**	RW	0	Write this bit to a 1b to force this compare event. This is a self-clearing bit.
6:5	**CCLATCH**	RW	0	CCR register latch mode. If **TDCCTL7.CCMODE** = 0b (compare): 00b: Latch **TDCCTR7** registers when **TDCTR** = 0 01b: Latch **TDCCTR7** registers when **TDCTR** = **TDPRD** 10b: Latch **TDCCTR7** registers immediately 11b: Reserved If **TDCCTL7.CCMODE** = 1b (capture): 00b: Latch **TDCTR** into **TDCCTR7.CTR** on rising edge 01b: Latch **TDCTR** into **TDCCTR7.CTR** on falling edge 10b: Latch **TDCTR** into **TDCCTR7.CTR** on both edges 11b: Reserved
4	**CCOUTINV**	RW	0	Invert CCR output: 0b: Do not invert CCR output 1b: Invert CCR output
3:2	**CCINTEDGE**	RW	0	When **TDCCTL7.CCMODE** = 1b (capture), set **TDINT.CCR7IF** = 1b when input signal detects: 00b: rising edge 01b: falling edge 10b: either rising or falling edges 11b: reserved
1	**CCINTEN**	RW	0	CCR interrupt enable: 0b: disabled 1b: enabled
0	**CCMODE**	RW	0	CCR mode: 0b: compare 1b: capture

22.8.20 TDCCTR7

Register 22-20 TDCCTR7 (Timer D CCR Counter 7, 4009 013Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**DELAY**	RW	0	If **TDCCTL7.CCMODE** = 0b (compare mode), this is the delay count to use before compare events that can be used for PWM output of phase shifted topologies.
15:0	**CTR**	RW	0	If **TDCCTRL7.CCMOD**E = 0b (compare mode), this is the compare value used for this CCR. If **TDCCTL7.CCMODE** = 1b (capture mode), this is the capture value for this CCR.

22.8.21 TDDTGCTL0

Register 22-21 TDDTGCTL0 (Timer D Dead-Time Generator Control 0, 4009 0200h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG0.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG0.

22.8.22 TDDTGCTL1

Register 22-22 TDDTGCTL1 (Timer D Dead-Time Generator Control 1, 4009 0204h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG1.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG1.

22.8.23 TDDTGCTL2

Register 22-23 TDDTGCTL2 (Timer D Dead-Time Generator Control 2, 4009 0208h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG2.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG2.

22.8.24 TDDTGCTL3

Register 22-24 TDDTGCTL3 (Timer D Dead-Time Generator Control 3, 4009 020Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**DTEN**	RW	0	Dead-time generator enable: 0b: not enabled 1b: enabled
30:28	**Reserved**	RO	0	Reserved
27:16	**FED**	RW	0	Falling edge delay: The number of DTGCLK ticks used for falling edge dead-time delay for DTG3.
15:12	**Reserved**	RW	0	Reserved
11:0	**RED**	RW	0	Rising edge delay: The number of DTGCLK ticks used for rising edge dead-time delay for DTG3.

22.8.25 TDQEPCTL

Register 22-25 TDQEPCTL (Timer D QEP Control, 4009 0300h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:11	**Reserved**	RO	0	Reserved
10:8	**CLKDIV**	RW	0	QEP Input Clock Divider: 000b: /1 001b: /2 010b: /4 011b: /8 100b: /16 101b: /32 110b: /64 111b: /128
7:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXRST**	RW	0	Index reset mode. Used for resetting counter. 0b: No reset 1b: Index edge reset. When direction is CW, reset counter to 0 when index rising edge event occurs. When direction is CCW, reset when index falling edge occurs (reset to 0 if not coincident with phase edge; reset to Tx**QEPMAX** if coincident with phase edge).
3	**DIR**	RO	0	Motor direction: 0b: CW (clockwise) 1b: CCW (counter-clockwise)
2	**CNTEDGE**	RW	0	Count on edge: 0b: Rising edge only 1b: Rising and falling edge
1	**CNTAB**	RW	0	Count on A/B: 0b: Count phase A only 1b: Count phase A and phase B
0	**QEPEN**	RW	0	QEP peripheral enable: 0b: disabled 1b: enabled

22.8.26 TDQEPCTR

Register 22-26 TDQEPCTR (Timer D QEP Counter, 4009 0304h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**CTR**	RW	0	Counter Value

22.8.27 TDQEPMAX

Register 22-27 TDQEPMAX (Timer D QEP Maximum Counter, 4009 0308h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**CTRMAX**	RW	0	Maximum Counter Value.

22.8.28 TDQEPIER

Register 22-28 TDQEPIER (Timer D QEP Interrupt Enable, 4009 030Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXEVIE**	RW	0	Index event interrupt enable: 0b: disabled 1b: enabled
3	**WRIE**	RW	0	Counter wrap interrupt enable: 0b: disabled 1b: enabled
2	**PHBIE**	RW	0	Phase B rising edge interrupt enable: 0b: disabled 1b: enabled
1	**PHAIE**	RW	0	Phase A rising edge interrupt enable: 0b: disabled 1b: enabled
0	**DIRIE**	RW	0	Direction change interrupt enable: 0b: disabled 1b: enabled

22.8.29 TDQEPIF

Register 22-29 TDQEPIF (Timer D QEP Interrupt Flag, 4009 0310h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved, write as 0.
4	**IDXEVIF**	W1C	0	Index event interrupt flag: 0b: disabled 1b: enabled
3	**WRIF**	W1C	0	Counter wrap interrupt flag: 0b: disabled 1b: enabled
2	**PHBIF**	W1C	0	Phase B rising edge interrupt flag: 0b: disabled 1b: enabled
1	**PHAIF**	W1C	0	Phase A rising edge interrupt flag: 0b: disabled 1b: enabled
0	**DIRIF**	W1C	0	Direction change interrupt flag: 0b: disabled 1b: enabled

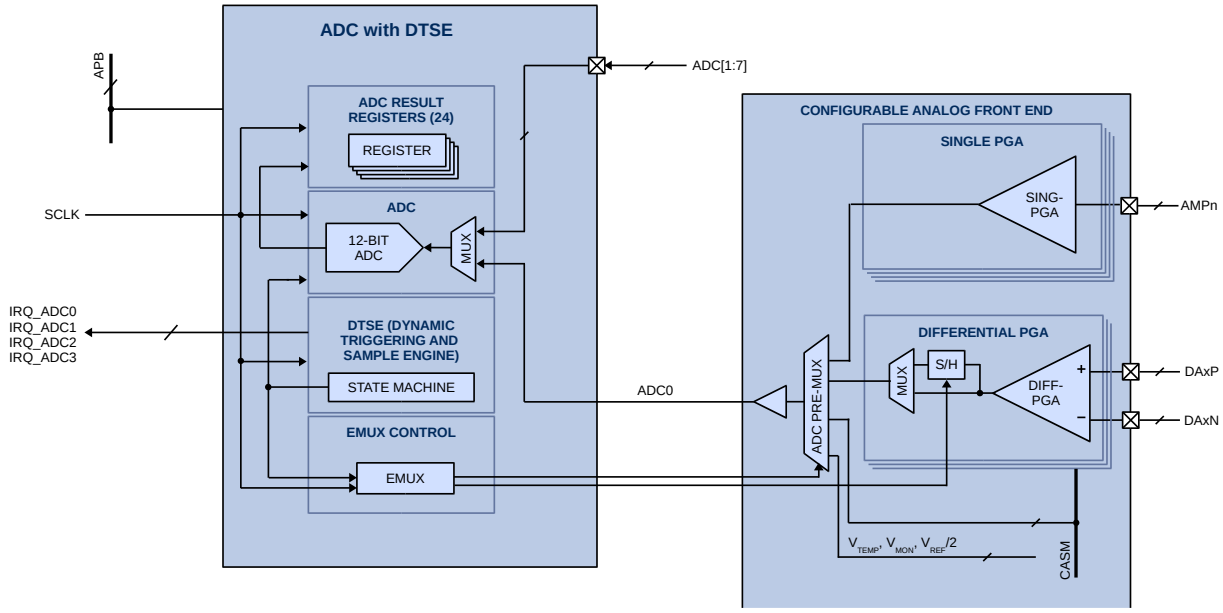
23 ADC AND DTSE

23.1 Overview

The PAC55XX contains a powerful Analog to Digital Converter (ADC) and Dynamic Triggering and Sampling Engine (DTSE).

The ADC is an 8-channel 12-bit 2.5MSPS SAR architecture Analog to Digital Converter. The ADC is controlled by a highly flexible and programmable sequencing engine called the DTSE.

Figure 23-1 ADC and DTSE Simplified Block Diagram



The ADC is an APB bus client. The ADC and DTSE are clocked by the ADC Clock which is a divided down version of SCLK (system clock) as determined by **ADCCTL.ADCDIV**. The EMUX is also clocked by the EMUX clock which is a divided down version of SCLK as determined by **EMUXCTL.EMUXDIV.**

The ADC may be programmed to generate up to 4 different interrupts to the NVIC.

The ADC works with the Configurable Analog Front-End™ (CAFE) to provide additional ADC channels as well as signal conditioning capabilities.

23.2 Features

The PAC55XX family ADC and DTSE peripheral has the following features:

- ADC Features:
  - 2.5 MSPS, 12-bit SAR ADC

- 8 Input Channels
  - Expandable through the CAFE
- Up to 40MHz Input Clock
- 4-bit Input Clock Divider
- Power-down mode
- EMUX serial port for low-latency communication to the CAFE
  - Up to 25MHz
  - Input clock divider
  - Busy status bit
  - Interrupt for EMUX done
- Manual or automatic (DTSE) conversion mode

- DTSE Features:
  - Up to 32 Input triggers
  - PWM output or capture input triggers for start of sequence
  - Software triggers for conversions
  - Multiple triggers may share the same conversion series for maximum flexibility and efficiency
  - Any conversion sequence may generate up to 4 IRQ signals to the NVIC
  - Automatic copying of conversion results to dedicated results registers (24)
  - Sending of EMUX command by hardware using sequencing engine

23.3 Functional Description

23.3.1 ADC

The ADC is a 12-bit 2.5MSPS SAR ADC. The ADC is powered by the VCORE 1.2V input for the digital logic (DTSE) and VCC33 3.3V input for the analog portion.

The PAC55XX generates its own voltage reference for the ADC, which can be either 2.5V or 3.0V.

The ADC has 8 analog input channels named ADC0 to ADC7. ADC0 is a special-purpose channel which is used to provide an analog input from the CAFE sub-system, which can perform additional signal conditioning, sample and hold via programmable gain single ended an differential amplifiers. For more information on the capabilities of the PAC55XX CAFE system, see the device Analog User Guide.

23.3.2 Enabling the ADC

To enable the ADC, set the **ADCCTL.ENABLE** to 1b. The ADC will not be able to perform any conversions in manual or DTSE modes unless this bit is set to 1b.

23.3.3 ADC Power-Down Mode

When not in use, it is recommended that the ADC put into ‘power-down’ mode. Even when the ADC is not being used by the rest of the PAC55XX, the analog portion of the ADC may still consume power.

In order to turn off the analog portion of the ADC when it is not in use, set **ADCCTL.ENABLE** to 0b. When this bit is set to 0b, the ADC registers will still be accessible, but the ADC will not be able to be used.

During normal operation, the **ADCCTL.ENABLE** will need to be set to 1b.

23.3.4 ADC Clock

The ADC and DTSE peripheral use an input clock for ADC conversions and to run the DTSE. The ADC and DTSE use the SCLK system clock as the input clock.

The maximum clock frequency that the ADC supports for conversions is 40MHz. The SCLK input clock may be much faster than this (up to 150MHz), so there is an ADC clock divider to reduce the input frequency to a maximum of 40MHz.

The **ADCCTL.ADCDIV** is a 4-bit divider that may be configured to divide the SCLK input clock from between 1 to 16. The table below shows how to configure the divider to the desired setting.

Table 23-1 ADC Input Clock Divider

ADCCTL.ADCDIV	ADCCLK
000b	SCLK /1
001b	SCLK /2
010b	SCLK /3
011b	SCLK /4
100b	SCLK /5
101b	SCLK /6
110b	SCLK /7
111b	SCLK /8

23.3.5 EMUX

In order to enable low-latency communication with the CAFE, the ADC and DTSE uses an EMUX peripheral. The EMUX is a dedicated, fast, low-latency serial interface used for quickly communicating data needed for signal sampling to the CAFE.

The EMUX may be used manually or in DTSE mode.

When the ADC is configured in manual mode (**ADCCTL.MODE** = 0b), the EMUX must also be set in manual mode. To set the EMUX into manual mode, set the **EMUXCTL.EMUXMODE** to 0b. In this mode, whenever the **EMUXDATA** register is written, the EMUX will write this data onto the bus to the CAFE sub-system.

When the ADC is configured in DTSE mode (**ADCCTL.MODE** = 01b), the EMUX must also be set into DTSE mode. To set the EMUX into DTSE mode, set the **EMUXCTL.EMUXMODE** to 1b. In this mode, the EMUX is written to automatically by the DTSE. Writing data into the **EMUXDATA** register has no effect.

When the EMUX is busy, the **EMUXCTL.EMUXBUSY** bit will be set to 1b. If the EMUX is not in use, then this bit will be a 0b.

The EMUX clock may be as high as 25MHz and is supplied from the SCLK peripheral clock. Since the SCLK frequency may be much higher than the EMUX clock, the EMUX has an input clock divider that may be used to reduce the input clock rate to 25MHz or lower.

The **EMUXCTL.EMUXDIV** contains the value to be used for the EMUX input clock divider. See the table below for how this may be configured to produce the desired EMUX clock divider.

Table 23-2 EMUX Input Clock Divider

EMUXCTL.EMUXDIV	EMUXCLK
000b	SCLK /1
001b	SCLK /2
010b	SCLK /3
011b	SCLK /4
100b	SCLK /5
101b	SCLK /6
110b	SCLK /7
111b	SCLK /8

23.3.6 EMUX Interrupts

The EMUX may be configured to generate an interrupt when it has completed writing data.

Whenever the EMUX completes writing data, it will set the **ADCINT.INTFEMUX** to a 1b. If the EMUX interrupt is enabled (**ADCCTL.INTENEMUX** = 1b), then the IRQ_ADC0 signal to the NVIC will be asserted.

Writing **ADCINT.INTFEMUX** to a 1b will clear this bit and de-assert the IRQ_ADC0, if asserted.

23.3.7 Manual ADC Conversions

To perform manual ADC conversions, follow the steps below:

- Enable the ADC by setting **ADCCTL.ENABLE** to 1b.
- Set the ADC mode to manual mode by setting **ADCCTL.MODE** to 0b.
- Select the ADC input channel by writing the **ADCCTL.CHANNEL** to the desired ADC input channel number (ADC0 to ADC7).
- Start the ADC conversion by setting the **ADCCTL.START** to a 1b. This bit is self-clearing and will always be read as 0b.
    - The ADC may have its channel selected and start the conversion at the same time by writing both the **ADCCTL.START** and **ADCCTL.CHANNEL** fields in the same register write to **ADCCTL**.
- At this point, the ADC will set the **ADCCTL.ADBUSY** bit to 1b. This bit will be 1b until the conversion has completed and the ADC has written the digital result into **ADCRES**.
- The ADC will set the **ADCCTL.ADBUSY** while the conversion is in progress.
- When the conversion has completed, the ADC will clear the **ADCCTL.ADBUSY** bit and set the **ADCINT.FMAN** bit to 1b indicating that the conversion has completed.

The user may read the result of the ADC conversion from the **ADCRES** register.

23.3.8 Manual ADC Conversions with Repeat

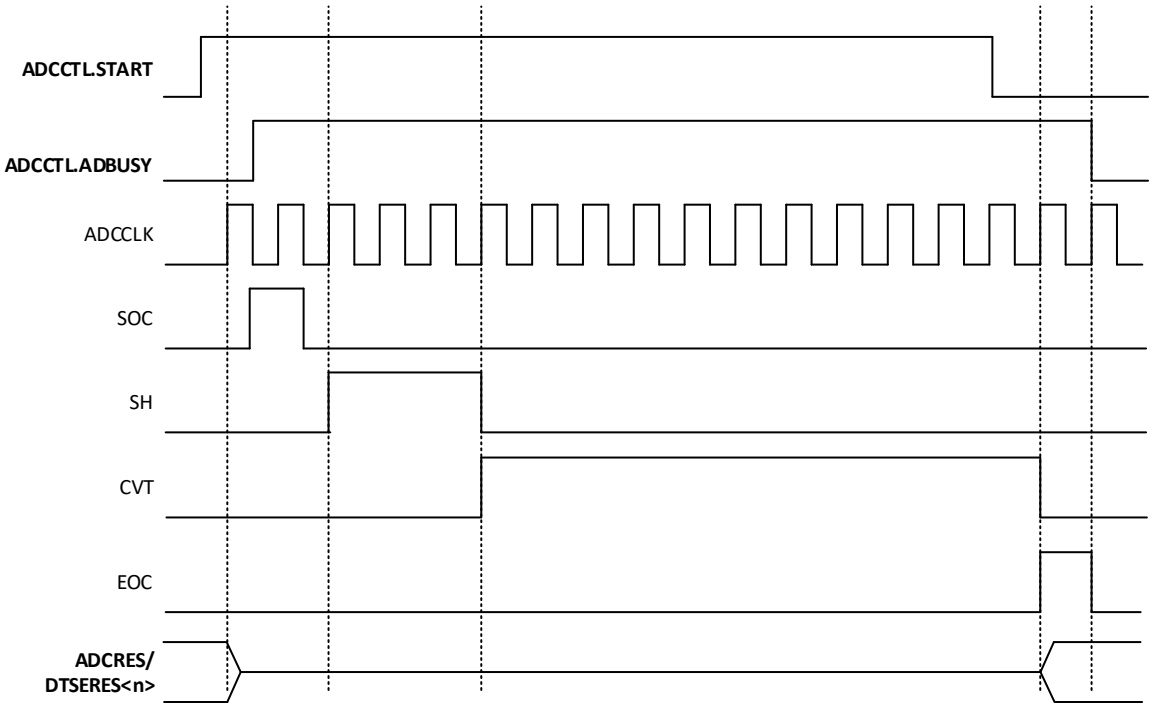
The ADC may continuously repeat ADC conversions in manual mode by setting the **ADCCTL.REPEAT** to a 1b.

When **ADCCTL.REPEAT** is set to a 1b and the **ADCCTL.START** it written to a 1b, the ADC will continue converting the channel in **ADCTL.CHANNEL** until the **ADCCTL.START** bit is cleared.

23.3.9 ADC Conversion Timing

The diagram below shows the timing of a single ADC conversion.

Figure 23-2 ADC Conversion Timing Diagram



The SOC (start of conversion) signal to the ADC is asserted when the **ADCCTL.START** bit is set. This bit may be set in manual mode by the user, or by the DTSE. The ADC then begins its sample and hold of 3 ADC clock cycles.

After the sampling period has ended, the ADC begins the digitization process of the analog signal. This will take 12 ADCCLK cycles, one for each bit of resolution of the digitized result.

After the conversion period, the EOC signal is asserted and the ADC copies the digital result to the **ADCRES** or **DTSERES<n>** register, depending on the ADC mode. The results are available one cycle after EOC is asserted.

When EOC is asserted, the ADC clears the **ADCCTL.ADBUSY** bit sets and the **ADCINT.INTFMAN** to 1b.

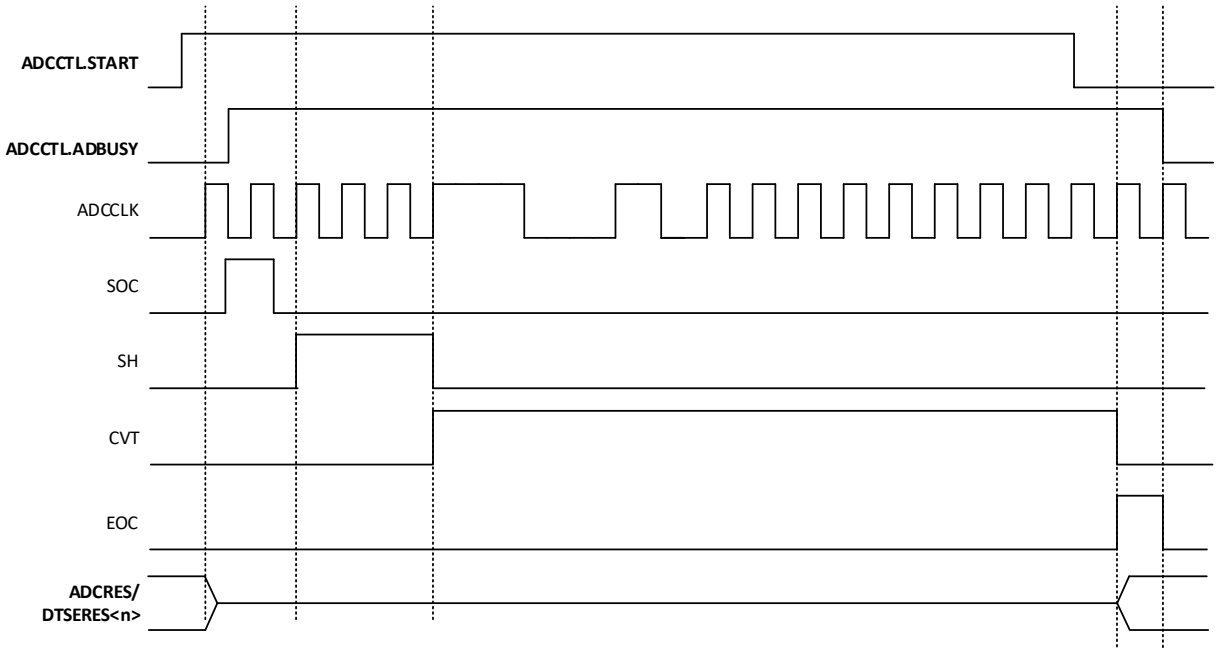
23.3.10 Enhanced ADC Accuracy Mode[21]

If the user sets **ADCCTL.ENACC** to 1b, the ADC will improve the accuracy of the conversion results. This mode may be useful if the ADCCLK is fast and at high temperatures.

When enabled, this mode will extend the conversion time of the first two digitized bits. This additional conversion time can improve the accuracy of the digitized result.

The diagram below shows the timing of a single ADC conversion when **ADCCTL.ENACC** is set to 1b.

Figure 23-3 ADC Accuracy Mode Conversion Timing Diagram



[21] This feature is not available in VER1 of the MCU.

23.3.11 ADC Manual Mode Interrupts

When the ADC is configured for manual mode (**ADCCTL.MODE** = 0b), when the ADC completes a conversion, the **ADCINT.INTFMAN** is set to 1b.

If the **ADCCTL.INTENMAN** is set to 1b and the **ADCINT.INTFMAN** is set to 1b, the ADC will assert the IRQ_ADC0 signal to the NVIC. The **ADCINT.INTFMAN** may be cleared when written to a 1b.

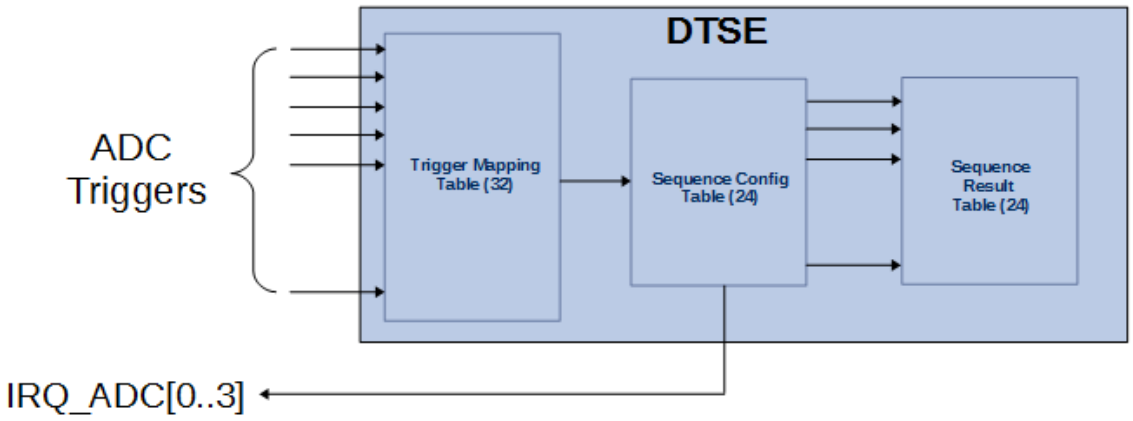
The IRQ_ADC0 signal will be de-asserted when the **ADCINT.INTFMAN** bit is clear and **ADCINT.ADCIRQ0IF** is cleared.

23.4 DTSE

23.4.1 Overview

The Dynamic Triggering and Sampling Engine (DTSE) is a power and highly configurable auto-sequencer that allows application-specific automatic ADC conversions without MCU intervention.

Figure 23-4 DTSE Simplified Block Diagram



There are three main components to the DTSE:

- Trigger Mapping Table
- Sequence Configuration Table
- Sequence Result Table

The DTSE allows the user to specify up to 32 possible input triggers (any PWM output from any timer) to start a series of ADC conversions. The *Trigger Mapping Table* allows the user to configure which of these triggers will trigger a given series of conversions.

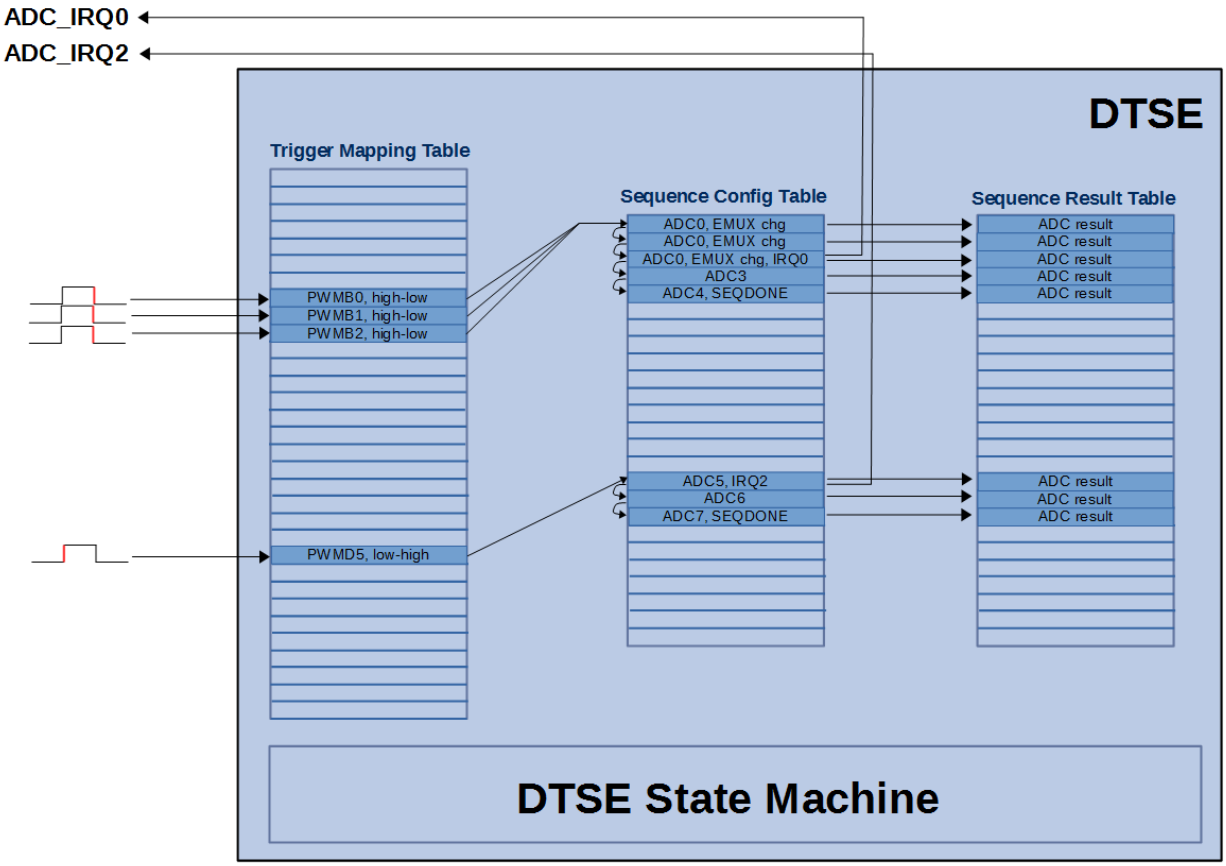
The *Sequence Configuration Table* allows the user to specify a series of conversions that the DTSE will perform automatically, without MCU intervention. This table allows the user to specify information about a single conversion such as the ADC channel, if an interrupt should be asserted, etc.

The *Sequence Result Table* are where the digital ADC conversion results are copied using DMA. There is a dedicated results register for each ADC sequence in the Sequence Configuration Table.

23.4.2 DTSE Conversion Sequence Example

The figure below shows an example of a DTSE conversion sequence.

Figure 23-5 DTSE Conversion Sequence Example



In this example, the *Trigger Mapping Table* is configured for 4 input triggers: a high to low transition on PWMB0, PWMB1 and PWMB2 and a low to high trigger on PWMD5.

There are two series of conversions configured in the *Sequence Configuration Table*, one for PWMB0/PWMB1/PWMB2 triggers and one for PWMD5 triggers. The first series is 5 conversion sequences long and asserts IRQ_ADC0 after the third conversion. The second series is 3 conversion sequence long and asserts IRQ_ADC2 after the first conversion in the series.

In both cases, the ADC results are copied to the results table after each individual conversion has been completed.

23.4.3 Trigger Mapping Table

The *Trigger Mapping Table* can be configured to hunt for triggers from any of the PWM outputs (or capture inputs) from Timers A, B, C or D. The user may configure the behavior of all of the DTSE triggers using the **DTSETRIGENTxTOy** registers.

Each trigger entry can be configured to look for the following signal transitions by writing the **DTSETRIGENTxTOy.TRIGnEDGE** field.

- None (00b)
- Low to high edge (01b)
- High to low edge (10b)
- Either Low to high or high to low edge (11b)

The user can force a trigger event in software by writing the **DTSETRIGENTxTOy.FORCE** field to a 1b. This will force the DTSE to process the sequence as if the trigger configured in **DTSETRIGENTxTOy.TRIGEDGE** was received.

The table below shows the *Trigger Mapping Table* and how to configure each of the PWM triggers.

Table 23-3 DTSE Trigger Mapping Table

REGISTER	FIELD	DTSE TRIGGER
DTSETRIGENT0TO3	**TRIG0EDGE**	Timer A, PWMA0/CCR0
	TRIG1EDGE	Timer A, PWMA1/CCR1
	TRIG2EDGE	Timer A, PWMA2/CCR2
	TRIG3EDGE	Timer A, PWMA3/CCR3
DTSETRIGENT4TO7	**TRIG4EDGE**	Timer A, PWMA4/CCR4
	TRIG5EDGE	Timer A, PWMA5/CCR5
	TRIG6EDGE	Timer A, PWMA6/CCR6
	TRIG7EDGE	Timer A, PWMA7/CCR7
DTSETRIGENT8TO11	**TRIG8EDGE**	Timer B, PWMB0/CCR0
	TRIG9EDGE	Timer B, PWMB1/CCR1
	TRIG10EDGE	Timer B, PWMB2/CCR2
	TRIG11EDGE	Timer B, PWMB3/CCR3
DTSETRIGENT12TO15	**TRIG12EDGE**	Timer B, PWMB4/CCR4
	TRIG13EDGE	Timer B, PWMB5/CCR5
	TRIG14EDGE	Timer B, PWMB6/CCR6
	TRIG15EDGE	Timer B, PWMB7/CCR7

DTSETRIGENT16TO19	**TRIG16EDGE**	Timer C, PWMC0/CCR0
	TRIG17EDGE	Timer C, PWMC1/CCR1
	TRIG18EDGE	Timer C, PWMC2/CCR2
	TRIG19EDGE	Timer C, PWMC3/CCR3
DTSETRIGENT20TO23	**TRIG20EDGE**	Timer C, PWMC4/CCR4
	TRIG21EDGE	Timer C, PWMC5/CCR5
	TRIG22EDGE	Timer C, PWMC6/CCR6
	TRIG23EDGE	Timer C, PWMC7/CCR7
DTSETRIGENT24TO27	**TRIG24EDGE**	Timer D, PWMD0/CCR0
	TRIG25EDGE	Timer D, PWMD1/CCR1
	TRIG26EDGE	Timer D, PWMD2/CCR2
	TRIG27EDGE	Timer D, PWMD3/CCR3
DTSETRIGENT28TO31	**TRIG28EDGE**	Timer D, PWMD4/CCR4
	TRIG29EDGE	Timer D, PWMD5/CCR5
	TRIG30EDGE	Timer D, PWMD6/CCR6
	TRIG31EDGE	Timer D, PWMD7/CCR7

If the DTSE detects an input trigger, then it uses the **DTSETRIGENTxTOY.TRIGnCFGIDX** to access the *Sequence Configuration Table* entry to perform the first conversion.

For example, if the *Trigger Mapping Table* has the following configuration:

- **DTSETRIG8TO11.TRIG10EDGE** = 10b
- **DTSETRIG8TO11.TRIG10CFGIDX** = 101b

If the DTSE detects a high-to-low transition on PWMB2, then the DTSE will use index 5 for the *Sequence Configuration Table* to determine which ADC conversion to perform to start the series.

23.4.4 Sequence Configuration Table

The *Sequence Configuration Table* contains a series of 24 entries. Each entry in this table configures a single ADC conversion. It is possible to chain together up to 24 ADC conversions in single series if all entries in this table are used.

Each entry in the Sequence Configuration Table allows the user to configure the following information:

- ADC Channel to convert

- EMUX Command Control (when to send EMUX command)
- EMUX data to send
- IRQ enable
- IRQ number to assert
- Last sequence in the conversion
- No conversion flag

The *Trigger Mapping Table* will point to the first sequence index in this table to convert. After each conversion the DTSE determines if it will perform addition conversions (**DTSESEQCFGn.SEQDONE** = 0b) or if it has completed ADC conversions for the series (**DTSESEQCFGn.SEQDONE** = 1b).

If the DTSE will perform more conversions, then it will advance to the next entry in the *Sequence Configuration Table*.

The table below shows a single entry from the *Sequence Configuration Table* and what the different fields control.

Table 23-4 DTSE Sequence Configuration Table Entry

FIELD	DESCRIPTION
IRQEN	IRQ Enabled. If set to a 1b, this will cause the DTSE to assert the IRQ signal to the NVIC according to the setting of the **IRQNUM** field in this register.
IRQNUM	IRQ Number to assert. If the IRQEN field is set to 1b, the DTSE will assert an interrupt to the NVIC according to the contents of this field as shown below. 00b: IRQ_ADC0 01b: IRQ_ADC1 10b: IRQ_ADC2 11b: IRQ_ADC3
EMUXD	EMUX Data to send. If the **EMUXC** field is set to 01b (send before sample) or 10b (send after conversion), the DTSE will write this field to EMUX according to the time specified in **EMUXC**.
SEQDONE	DTSE sequence done. 0b: Sequence not complete. Advance to the next entry in this table for the next conversion. 1b: Sequence is complete.
CHANNEL	ADC Channel. This is the ADC channel to convert for this conversion sequence: 000b: ADC0 001b: ADC1

	010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
EMUXC	EMUX Control. This field indicates when the DTSE will send the EMUX command for this conversion: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.
NOCVT[22]	No conversion flag. This field indicates if the DTSE should perform or not perform an ADC conversion during this sequence. If a conversion is not performed, only the EMUX command will be sent and then the DTSE will continue to the next sequence. 0b: Perform ADC conversion as configured 1b: Do not perform ADC conversion This feature is useful if the user wants to send an EMUX command quickly without taking the time for the full ADC conversion.

23.4.5 Result Table

When the DTSE *Sequence Configuration Table* has completed a conversion, the DTSE writes the digital result of the conversion into the **DTSERESn** register and sets the appropriate **ADCINT.INFn** field.

For example, if entry 11 in the *Sequence Configuration Table* has just been converted, the DTSE will write the ADC result into the **DTSERES11** register and set the **ADCINT.INTF11** bit to 1b.

23.4.6 DTSE Conversion State Machine

When the DTSE is enabled and configured, the following steps are executed by the DTSE when performing a series of conversions:

1. The DTSE hunts for triggers that are configured in the *Trigger Mapping Table*
2. When a trigger is received, the DTSE checks the **DTSETRIGENTxTOy.TRIGEDGE** to see if the edge of the trigger is configured.

[22] The no conversion configuration is not available in VER1 of the MCU.

3. If the trigger is received, the DTSE looks up the entry in the *Sequence Configuration Table* according to the index found in the *Trigger Mapping Table* (**DTSETRIGENTxTOy.TRIGnCFGIDX**).
4. Set **ADCCTL.ADBUSY** to 1b
5. Label: *do_conversion*
    a. If **DTSESEQCFGn.EMUXC** = 01b (send EMUX command before sampling), send the EMUX data from **DTSESEQCFGn.EMUXD**.
    b. Set the ADC channel to **DTSESEQCFGn.CHANNEL** start the ADC conversion.
    c. Wait for sample and hold phase to complete
    d. Wait for ADC conversion to complete
    e. Write the result of this conversion to the **DTSERESn** register.
    f. If **DTSESEQCFGn.EMUXC** = 10b (send EMUX command after conversion), send the EMUX data from **DTSESEQCFGn.EMUXD**.
    g. Set the **ADCINT.INTFn** flag for this sequence entry
    h. If the **DTSESEQCFGn.IRQEN** is set to 1b, assert the IRQ signal as specified in **DTSESEQCFGn.IRQNUM**.
    i. If **DTSESEQCFGn.SEQDONE** = 0b, advance to next entry in the *Sequence Configuration Table* and goto *do_conversion* above. Otherwise, continue below.
6. Set **ADCCTL.ADBUSY** = 0

23.4.7 Clearing DTSE Interrupts

The **ADCINT** register may be used to clear any of the interrupts (IRQ_ADC0, IRQ_ADC1, IRQ_ADC2 or IRQ_ADC3).

To clear any of these IRQs, clear the interrupt flags as shown below:

- To clear IRQ_ADC0, set **ADCINT.IRQ0IF** to 1b
- To clear IRQ_ADC1, set **ADCINT.IRQ1IF** to 1b
- To clear IRQ_ADC2, set **ADCINT.IRQ2IF** to 1b
- To clear IRQ_ADC3, set **ADCINT.IRQ3IF** to 1b

23.5 Peripheral IO Mapping

The ADC and DTSE ADC inputs are connected by an Analog MUX so they can be used by various IO pins.

The tables below show which IO pins are available for each ADC channel.

Table 23-5 ADC Peripheral IO Mapping

ADC CHANNEL	IO PIN
ADC0	PG7
ADC1	PD3, PG5

ADC2	PD2, PG6
ADC3	PD1
ADC4	PD0, PF4
ADC5	PF5
ADC6	PF6
ADC7	PF7

For more information on how to use the Digital Peripheral MUX to connect peripheral signals to IO, see **Error! Reference source not found.**.

23.6 DTSE Clock Configuration

In VER1 of the MCU, there are limitations for the relationship between the4 ADC and EMUX clock as described in this section.

For the ADC DTSE to work properly, the frequency of the ADC clock and EMUX must be configured to follow the equation:

- EMUX clock <= ½ * ADC clock

The ADC clock and the EMUX clock are both derived from SCLK (the system clock). The EMUX clock frequency is selected by configuring the EMUX input clock divider in **EMUXCTL.EMUXDIV**, and the ADC clock frequency is selected by configuring the ADC input clock divider in **ADCCTL.ADCDIV**.

When using the DTSE, the ADC/EMUX clock limitation will result in the EMUX transaction completing and switching the EMUX channel right at the start of the next ADC conversion cycle. Therefore, the input to the sample and hold (S/H) before the ADC will not be settled, and the next ADC conversion will be invalid.

The table below shows several examples of legal configurations of the EMUX and ADC clock for the DTSE for VER1 of the MCU.

Table 23-6 DTSE Valid Clock Configurations

SCLK (MHz)	HCLK Divider	HCLK (MHz)	ADCCLK Divider	ADCCLK (MHz)	EMUXCLK Divider	EMUXCLK (MHz)	ADC Throughput (MSPS)
300	2	150	8	37.5	16	18.75	2.34
240	2	120	6	40	12	20	2.5
150	1	150	5	30	10	15	1.875
200	2	100	5	40	10	20	2.5

23.7 Register Summary

Table 23-7 ADC and DTSE Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
EMUXCTL	4000 0000h	EMUX Control	RW	0000 0000h
EMUXDATA	4000 0004h	EMUX Data	RW	0000 0000h
ADCCTL	4000 0008h	ADC Control	RW	0000 0000h
ADCRES	4000 000Ch	ADC Result	RW	0000 0000h
ADCINT	4000 0010h	ADC Interrupt Control	RW	0000 0000h
DTSETRIGENT0TO3	4000 0040h	DTSE Trigger Entry 0 to 3	RW	0000 0000h
DTSETRIGENT4TO7	4000 0044h	DTSE Trigger Entry 4 to 7	RW	0000 0000h
DTSETRIGENT8TO11	4000 0048h	DTSE Trigger Entry 8 to 11	RW	0000 0000h
DTSETRIGENT12TO15	4000 004Ch	DTSE Trigger Entry 12 to 15	RW	0000 0000h
DTSETRIGENT16TO19	4000 0050h	DTSE Trigger Entry 16 to 19	RW	0000 0000h
DTSETRIGENT20TO23	4000 0054h	DTSE Trigger Entry 20 to 23	RW	0000 0000h
DTSETRIGENT24TO27	4000 0058h	DTSE Trigger Entry 24 to 27	RW	0000 0000h
DTSETRIGENT28TO31	4000 005Ch	DTSE Trigger Entry 28 to 31	RW	0000 0000h
DTSESEQCFG0	4000 0080h	DTSE Sequence Config 0	RW	0000 0000h
DTSESEQCFG1	4000 0084h	DTSE Sequence Config 1	RW	0000 0000h
DTSESEQCFG2	4000 0088h	DTSE Sequence Config 2	RW	0000 0000h
DTSESEQCFG3	4000 008Ch	DTSE Sequence Config 3	RW	0000 0000h
DTSESEQCFG4	4000 0090h	DTSE Sequence Config 4	RW	0000 0000h
DTSESEQCFG5	4000 0094h	DTSE Sequence Config 5	RW	0000 0000h
DTSESEQCFG6	4000 0098h	DTSE Sequence Config 6	RW	0000 0000h
DTSESEQCFG7	4000 009Ch	DTSE Sequence Config 7	RW	0000 0000h
DTSESEQCFG8	4000 00A0h	DTSE Sequence Config 8	RW	0000 0000h
DTSESEQCFG9	4000 00A4h	DTSE Sequence Config 9	RW	0000 0000h
DTSESEQCFG10	4000 00A8h	DTSE Sequence Config 10	RW	0000 0000h
DTSESEQCFG11	4000 00ACh	DTSE Sequence Config 11	RW	0000 0000h
DTSESEQCFG12	4000 00B0h	DTSE Sequence Config 12	RW	0000 0000h
DTSESEQCFG13	4000 00B4h	DTSE Sequence Config 13	RW	0000 0000h
DTSESEQCFG14	4000 00B8h	DTSE Sequence Config 14	RW	0000 0000h
DTSESEQCFG15	4000 00BCh	DTSE Sequence Config 15	RW	0000 0000h
DTSESEQCFG16	4000 00C0h	DTSE Sequence Config 16	RW	0000 0000h
DTSESEQCFG17	4000 00C4h	DTSE Sequence Config 17	RW	0000 0000h

DTSESEQCFG18	4000 00C8h	DTSE Sequence Config 18	RW	0000 0000h
DTSESEQCFG19	4000 00CCh	DTSE Sequence Config 19	RW	0000 0000h
DTSESEQCFG20	4000 00D0h	DTSE Sequence Config 20	RW	0000 0000h
DTSESEQCFG21	4000 00D4h	DTSE Sequence Config 21	RW	0000 0000h
DTSESEQCFG22	4000 00D8h	DTSE Sequence Config 22	RW	0000 0000h
DTSESEQCFG23	4000 00DCh	DTSE Sequence Config 23	RW	0000 0000h
DTSERES0	4000 0100h	DTSE Result 0	RO	0000 0000h
DTSERES1	4000 0104h	DTSE Result 1	RO	0000 0000h
DTSERES2	4000 0108h	DTSE Result 2	RO	0000 0000h
DTSERES3	4000 010Ch	DTSE Result 3	RO	0000 0000h
DTSERES4	4000 0110h	DTSE Result 4	RO	0000 0000h
DTSERES5	4000 0114h	DTSE Result 5	RO	0000 0000h
DTSERES6	4000 0118h	DTSE Result 6	RO	0000 0000h
DTSERES7	4000 011Ch	DTSE Result 7	RO	0000 0000h
DTSERES8	4000 0120h	DTSE Result 8	RO	0000 0000h
DTSERES9	4000 0124h	DTSE Result 9	RO	0000 0000h
DTSERES10	4000 0128h	DTSE Result 10	RO	0000 0000h
DTSERES11	4000 012Ch	DTSE Result 11	RO	0000 0000h
DTSERES12	4000 0130h	DTSE Result 12	RO	0000 0000h
DTSERES13	4000 0134h	DTSE Result 13	RO	0000 0000h
DTSERES14	4000 0138h	DTSE Result 14	RO	0000 0000h
DTSERES15	4000 013Ch	DTSE Result 15	RO	0000 0000h
DTSERES16	4000 0140h	DTSE Result 16	RO	0000 0000h
DTSERES17	4000 0144h	DTSE Result 17	RO	0000 0000h
DTSERES18	4000 0148h	DTSE Result 18	RO	0000 0000h
DTSERES19	4000 014Ch	DTSE Result 19	RO	0000 0000h
DTSERES20	4000 0150h	DTSE Result 20	RO	0000 0000h
DTSERES21	4000 0154h	DTSE Result 21	RO	0000 0000h
DTSERES22	4000 0158h	DTSE Result 22	RO	0000 0000h
DTSERES23	4000 015Ch	DTSE Result 23	RO	0000 0000h

23.8 Register Detail

23.8.1 EMUXCTL

Register 23-1 EMUXCTL (EMUX Control, 4000 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:6	**Reserved**	RO	0	Reserved
5	**EMUXMODE**	RW	0	EMUX Mode: 0b: Write EMUX data from **EMUXDATA** register 1b: Write EMUX data from DTSE sequencer commands
4	**EMUXBUSY**	RO	0	EMUX Busy Status: 0b: EMUX not busy 1b: EMUX busy
3:0	**EMUXDIV**	RW	0	EMUX Clock Divider: 0000b: SCLK /1 0001b: SCLK /2 0010b: SCLK /3 0011b: SCLK /4 0100b: SCLK /5 0101b: SCLK /6 0110b: SCLK /7 0111b: SCLK /8 1000b: SCLK /9 1001b: SCLK /10 1010b: SCLK /11 1011b: SCLK /12 1100b: SCLK /13 1101b: SCLK /14 1110b: SCLK /15 1111b: SCLK /16

23.8.2 EMUXDATA

Register 23-2 EMUXDATA (EMUX Data, 4000 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**EMUXDATA**	RW	0	When **EMUXCTL.EMUXMODE** = 0b, the EMUX writes this data onto the EMUX. Writing this data when **EMUXCTL.EMUXMODE** = 1b has no effect.

23.8.3 ADCCTL

Register 23-3 ADCCTL (ADC Configuration, 4000 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:19	**Reserved**	RO	0	Reserved
18	**ENACC**	RW	0	ADC Accuracy Mode: 0b: not enabled 1b: enabled
17:16	**Reserved**	RW	0	Reserved, must write to 0.
15	**INTENCOL**	RW	0	Sequence Collision Interrupt Enable: 0b: not enabled 1b: enabled
14	**INTENEMUX**	RW	0	EMUX Complete Interrupt Enable: 0b: not enabled 1b: enabled
13	**INTENMAN**	RW	0	ADC Manual Mode Conversion Complete: 0b: not enabled 1b: enabled
12	**ENABLE**	RW	0	ADC Enable: 0b: ADC module disabled (PD mode) 1b: ADC module enabled
11	**START**	W1C	0	ADC Start Conversion: 1b: Start the ADC conversion or conversion sequence This bit is automatically cleared when set.
10	**REPEAT**	RW	0	Enabled Repeated Conversions: 0b: Disable repeated conversions 1b: Enable repeated conversions
9:8	**MODE**	RW	0	ADC Mode: 00b: Manual mode 01b: DTSE mode
7	**ADBUSY**	RO	0	ADC Busy: 0b: ADC not performing conversion 1b: ADC performing conversion
6:4	**CHANNEL**	RW	0	ADC Channel. When **ADCCTL.MODE** = 0b (manual mode), this field sets the ADC channel to convert. 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
3:0	**ADCDIV**	RW	0	ADC Clock Divider: 0000b: SCLK /1 0001b: SCLK /2 0010b: SCLK /3 0011b: SCLK /4 0100b: SCLK /5

				0101b: SCLK /6 0110b: SCLK /7 0111b: SCLK /8 1000b: SCLK /9 1001b: SCLK /10 1010b: SCLK /11 1011b: SCLK /12 1100b: SCLK /13 1101b: SCLK /14 1110b: SCLK /15 1111b: SCLK / 16

23.8.4 ADCRES

Register 23-4 ADCRES (ADC Result, 4000 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	When **ADCCTL.MODE** = 0b (manual mode) and an ADC conversion has completed, the 12-bit digitized results is available here.

23.8.5 ADCINT

Register 23-5 ADCINT (ADC Interrupt Control, 4000 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**ADCIRQ3IF**	W1C	0	ADCIRQ3 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
30	**ADCIRQ2IF**	W1C	0	ADCIRQ2 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
29	**ADCIRQ1IF**	W1C	0	ADCIRQ1 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
28	**ADCIRQ0IF**	W1C	0	ADCIRQ0 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
27	**Reserved**	RO	0	Reserved
26	**INTFCOL**	W1C	0	DTSE Sequence Collision Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
25	**INTFEMUX**	W1C	0	DTSE EMUX Complete Interrupt Flag- 0b: no interrupt flag 1b: interrupt flag
24	**INTFMAN**	W1C	0	DTSE Manual Mode Conversion Complete Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
23	**INTF23**	W1C	0	DTSE Sequence 23 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
22	**INTF22**	W1C	0	DTSE Sequence 22 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
21	**INTF21**	W1C	0	DTSE Sequence 21 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
20	**INTF20**	W1C	0	DTSE Sequence 20 Interrupt Flag: 0b: no interrupt flag

				1b: interrupt flag
19	**INTF19**	W1C	0	DTSE Sequence 19 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
18	**INTF18**	W1C	0	DTSE Sequence 18 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
17	**INTF17**	W1C	0	DTSE Sequence 17 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
16	**INTF16**	W1C	0	DTSE Sequence 16 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
15	**INTF15**	W1C	0	DTSE Sequence 15 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
14	**INTF14**	W1C	0	DTSE Sequence 14 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
13	**INTF13**	W1C	0	DTSE Sequence 13 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
12	**INTF12**	W1C	0	DTSE Sequence 12 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
11	**INTF11**	W1C	0	DTSE Sequence 11 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
10	**INTF10**	W1C	0	DTSE Sequence 10 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
9	**INTF9**	W1C	0	DTSE Sequence 9 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
8	**INTF8**	W1C	0	DTSE Sequence 8 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
7	**INTF7**	W1C	0	DTSE Sequence 7 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
6	**INTF6**	W1C	0	DTSE Sequence 6 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
5	**INTF5**	W1C	0	DTSE Sequence 5 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
4	**INTF4**	W1C	0	DTSE Sequence 4 Interrupt Flag: 0b: no interrupt flag

				1b: interrupt flag
3	**INTF3**	W1C	0	DTSE Sequence 3 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
2	**INTF2**	W1C	0	DTSE Sequence 2 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
1	**INTF1**	W1C	0	DTSE Sequence 1 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag
0	**INTF0**	W1C	0	DTSE Sequence 0 Interrupt Flag: 0b: no interrupt flag 1b: interrupt flag

23.8.6 DTSETRIGENT0TO3

Register 23-6 DTSETRIGENT0TO3 (DTSE Trigger Entry 0 to 3, 4000 0040h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**FORCE3**	W1C	0	Force Trigger 3. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 3. This bit is cleared when set and writing 0 to this bit has no effect.
30:29	**TRIG3EDGE**	RW	0	DTSE Trigger 3 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
28:24	**TRIG3CFGIDX**	RW	0	DTSE Trigger 3 Sequence Configuration Entry Index.
23	**FORCE2**	W1C	0	Force Trigger 2. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 2. This bit is cleared when set and writing 0 to this bit has no effect.
22:21	**TRIG2EDGE**	RW	0	DTSE Trigger 2 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
20:16	**TRIG2CFGIDX**	RW	0	DTSE Trigger 2 Sequence Configuration Entry Index.
15	**FORCE1**	W1C	0	Force Trigger 1. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 1. This bit is cleared when set and writing 0 to this bit has no effect.
14:13	**TRIG1EDGE**	RW	0	DTSE Trigger 1 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
12:8	**TRIG1CFGIDX**	RW	0	DTSE Trigger 1 Sequence Configuration Entry Index.
7	**FORCE0**	W1C	0	Force Trigger 0. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 0. This bit is cleared when set and writing 0 to this bit has no effect.
6:5	**TRIG0EDGE**	RW	0	DTSE Trigger 0 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges

4:0	**TRIG0CFGIDX**	RW	0	DTSE Trigger 0 Sequence Configuration Entry Index.

23.8.7 DTSETRIGENT4TO7

Register 23-7 DTSETRIGENT4TO7 (DTSE Trigger Entry 4 to 7, 4000 0044h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**FORCE7**	W1C	0	Force Trigger 7. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 7. This bit is cleared when set and writing 0 to this bit has no effect.
30:29	**TRIG7EDGE**	RW	0	DTSE Trigger 7 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
28:24	**TRIG7CFGIDX**	RW	0	DTSE Trigger 7 Sequence Configuration Entry Index.
23	**FORCE6**	W1C	0	Force Trigger 6. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 6. This bit is cleared when set and writing 0 to this bit has no effect.
22:21	**TRIG6EDGE**	RW	0	DTSE Trigger 6 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
20:16	**TRIG6CFGIDX**	RW	0	DTSE Trigger 6 Sequence Configuration Entry Index.
15	**FORCE5**	W1C	0	Force Trigger 5. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 5. This bit is cleared when set and writing 0 to this bit has no effect.
14:13	**TRIG5EDGE**	RW	0	DTSE Trigger 5 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
12:8	**TRIG5CFGIDX**	RW	0	DTSE Trigger 5 Sequence Configuration Entry Index.
7	**FORCE4**	W1C	0	Force Trigger 4. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 4. This bit is cleared when set and writing 0 to this bit has no effect.

6:5	**TRIG4EDGE**	RW	0	DTSE Trigger 4 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
4:0	**TRIG4CFGIDX**	RW	0	DTSE Trigger 4 Sequence Configuration Entry Index.

23.8.8 DTSETRIGENT8TO11

Register 23-8 DTSETRIGENT8TO11 (DTSE Trigger Entry 8 to 11, 4000 0048h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**FORCE11**	W1C	0	Force Trigger 11. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 11. This bit is cleared when set and writing 0 to this bit has no effect.
30:29	**TRIG11EDGE**	RW	0	DTSE Trigger 11 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
28:24	**TRIG11CFGIDX**	RW	0	DTSE Trigger 11 Sequence Configuration Entry Index.
23	**FORCE10**	W1C	0	Force Trigger 10. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 10. This bit is cleared when set and writing 0 to this bit has no effect.
22:21	**TRIG10EDGE**	RW	0	DTSE Trigger 10 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
20:16	**TRIG10CFGIDX**	RW	0	DTSE Trigger 10 Sequence Configuration Entry Index.
15	**FORCE9**	W1C	0	Force Trigger 9. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 9. This bit is cleared when set and writing 0 to this bit has no effect.
14:13	**TRIG9EDGE**	RW	0	DTSE Trigger 9 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
12:8	**TRIG9CFGIDX**	RW	0	DTSE Trigger 9 Sequence Configuration Entry Index.
7	**FORCE8**	W1C	0	Force Trigger 8. When ADCCTL.MODE = 01b (DTSE mode) and this bit is

				written to a 1b, it forces DTSE trigger 8. This bit is cleared when set and writing 0 to this bit has no effect.
6:5	**TRIG8EDGE**	RW	0	DTSE Trigger 8 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
4:0	**TRIG8CFGIDX**	RW	0	DTSE Trigger 8 Sequence Configuration Entry Index.

23.8.9 DTSETRIGENT12TO15

Register 23-9 DTSETRIGENT12TO15 (DTSE Trigger Entry 12 to 15, 4000 004Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**FORCE15**	W1C	0	Force Trigger 15. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 15. This bit is cleared when set and writing 0 to this bit has no effect.
30:29	**TRIG15EDGE**	RW	0	DTSE Trigger 15 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
28:24	**TRIG15CFGIDX**	RW	0	DTSE Trigger 15 Sequence Configuration Entry Index.
23	**FORCE14**	W1C	0	Force Trigger 14. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 14. This bit is cleared when set and writing 0 to this bit has no effect.
22:21	**TRIG14EDGE**	RW	0	DTSE Trigger 14 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
20:16	**TRIG14CFGIDX**	RW	0	DTSE Trigger 14 Sequence Configuration Entry Index.
15	**FORCE13**	W1C	0	Force Trigger 13. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 13. This bit is cleared when set and writing 0 to this bit has no effect.
14:13	**TRIG13EDGE**	RW	0	DTSE Trigger 13 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges

12:8	**TRIG13CFGIDX**	RW	0	DTSE Trigger 13 Sequence Configuration Entry Index.
7	**FORCE12**	W1C	0	Force Trigger 12. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 12. This bit is cleared when set and writing 0 to this bit has no effect.
6:5	**TRIG12EDGE**	RW	0	DTSE Trigger 12 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
4:0	**TRIG12CFGIDX**	RW	0	DTSE Trigger 12 Sequence Configuration Entry Index.

23.8.10 DTSETRIGENT16TO19

Register 23-10 DTSETRIGENT16TO19 (DTSE Trigger Entry 16 to 19, 4000 0050h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**FORCE19**	W1C	0	Force Trigger 19. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 19. This bit is cleared when set and writing 0 to this bit has no effect.
30:29	**TRIG19EDGE**	RW	0	DTSE Trigger 19 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
28:24	**TRIG19CFGIDX**	RW	0	DTSE Trigger 19 Sequence Configuration Entry Index.
23	**FORCE18**	W1C	0	Force Trigger 18. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 18. This bit is cleared when set and writing 0 to this bit has no effect.
22:21	**TRIG18EDGE**	RW	0	DTSE Trigger 18 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
20:16	**TRIG18CFGIDX**	RW	0	DTSE Trigger 18 Sequence Configuration Entry Index.
15	**FORCE17**	W1C	0	Force Trigger 17. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 17. This bit is cleared when set and writing 0 to this bit has no effect.
14:13	**TRIG17EDGE**	RW	0	DTSE Trigger 17 Edge Configuration:

				00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
12:8	**TRIG17CFGIDX**	RW	0	DTSE Trigger 17 Sequence Configuration Entry Index.
7	**FORCE16**	W1C	0	Force Trigger 16. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 16. This bit is cleared when set and writing 0 to this bit has no effect.
6:5	**TRIG16EDGE**	RW	0	DTSE Trigger 16 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
4:0	**TRIG16CFGIDX**	RW	0	DTSE Trigger 16 Sequence Configuration Entry Index.

23.8.11 DTSETRIGENT20TO23

Register 23-11 DTSETRIGENT20TO23 (DTSE Trigger Entry 20 to 23, 4000 0054h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**FORCE23**	W1C	0	Force Trigger 23. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 23. This bit is cleared when set and writing 0 to this bit has no effect.
30:29	**TRIG23EDGE**	RW	0	DTSE Trigger 23 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
28:24	**TRIG23CFGIDX**	RW	0	DTSE Trigger 23 Sequence Configuration Entry Index.
23	**FORCE22**	W1C	0	Force Trigger 22. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 22. This bit is cleared when set and writing 0 to this bit has no effect.
22:21	**TRIG22EDGE**	RW	0	DTSE Trigger 22 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
20:16	**TRIG22CFGIDX**	RW	0	DTSE Trigger 22 Sequence Configuration Entry Index.
15	**FORC21**	W1C	0	Force Trigger 21. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 21.

				This bit is cleared when set and writing 0 to this bit has no effect.
14:13	**TRIG21EDGE**	RW	0	DTSE Trigger 21 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
12:8	**TRIG21CFGIDX**	RW	0	DTSE Trigger 21 Sequence Configuration Entry Index.
7	**FORCE20**	W1C	0	Force Trigger 20. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 20. This bit is cleared when set and writing 0 to this bit has no effect.
6:5	**TRIG20EDGE**	RW	0	DTSE Trigger 20 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
4:0	**TRIG20CFGIDX**	RW	0	DTSE Trigger 20 Sequence Configuration Entry Index.

23.8.12 DTSETRIGENT24TO27

Register 23-12 DTSETRIGENT24TO27 (DTSE Trigger Entry 24 to 27, 4000 0058h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**FORCE27**	W1C	0	Force Trigger 27. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 27. This bit is cleared when set and writing 0 to this bit has no effect.
30:29	**TRIG27EDGE**	RW	0	DTSE Trigger 27 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
28:24	**TRIG27CFGIDX**	RW	0	DTSE Trigger 27 Sequence Configuration Entry Index.
23	**FORCE26**	W1C	0	Force Trigger 26. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 26. This bit is cleared when set and writing 0 to this bit has no effect.
22:21	**TRIG26EDGE**	RW	0	DTSE Trigger 26 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
20:16	**TRIG26CFGIDX**	RW	0	DTSE Trigger 26 Sequence Configuration Entry Index.

15	**FORCE25**	W1C	0	Force Trigger 25. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 25. This bit is cleared when set and writing 0 to this bit has no effect.
14:13	**TRIG25EDGE**	RW	0	DTSE Trigger 25 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
12:8	**TRIG25CFGIDX**	RW	0	DTSE Trigger 25 Sequence Configuration Entry Index.
7	**FORCE24**	W1C	0	Force Trigger \24. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 24. This bit is cleared when set and writing 0 to this bit has no effect.
6:5	**TRIG24EDGE**	RW	0	DTSE Trigger 24 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
4:0	**TRIG24CFGIDX**	RW	0	DTSE Trigger 24 Sequence Configuration Entry Index.

23.8.13 DTSETRIGENT28TO31

Register 23-13 DTSETRIGENT28TO31 (DTSE Trigger Entry 28 to 31, 4000 005Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31	**FORCE31**	W1C	0	Force Trigger 31. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 31. This bit is cleared when set and writing 0 to this bit has no effect.
30:29	**TRIG31EDGE**	RW	0	DTSE Trigger 31 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
28:24	**TRIG31CFGIDX**	RW	0	DTSE Trigger 31 Sequence Configuration Entry Index.
23	**FORCE30**	W1C	0	Force Trigger 30. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 30. This bit is cleared when set and writing 0 to this bit has no effect.
22:21	**TRIG30EDGE**	RW	0	DTSE Trigger 30 Edge Configuration: 00b: unused 01b: trigger rising edge

				10b: trigger falling edge 11b: trigger both rising and falling edges
20:16	**TRIG30CFGIDX**	RW	0	DTSE Trigger 30 Sequence Configuration Entry Index.
15	**FORCE29**	W1C	0	Force Trigger 29. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 29. This bit is cleared when set and writing 0 to this bit has no effect.
14:13	**TRIG29EDGE**	RW	0	DTSE Trigger 29 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
12:8	**TRIG29CFGIDX**	RW	0	DTSE Trigger 29 Sequence Configuration Entry Index.
7	**FORCE28**	W1C	0	Force Trigger 28. When ADCCTL.MODE = 01b (DTSE mode) and this bit is written to a 1b, it forces DTSE trigger 28. This bit is cleared when set and writing 0 to this bit has no effect.
6:5	**TRIG28EDGE**	RW	0	DTSE Trigger 28 Edge Configuration: 00b: unused 01b: trigger rising edge 10b: trigger falling edge 11b: trigger both rising and falling edges
4:0	**TRIG28CFGIDX**	RW	0	DTSE Trigger 28 Sequence Configuration Entry Index.

23.8.14 DTSESEQCFG0

Register 23-14 DTSESEQCFG0 (DTSE Sequence Configuration 0, 4000 0080h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG0.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG0.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSEQCFG0.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.15 DTSESEQCFG1

Register 23-15 DTSESEQCFG1 (DTSE Sequence Configuration 1, 4000 0084h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG1.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG1.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSEQCFG1.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.16 DTSESEQCFG2

Register 23-16 DTSESEQCFG2 (DTSE Sequence Configuration 2, 4000 0088h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG2.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG2.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSEQCFG2.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.17 DTSESEQCFG3

Register 23-17 DTSESEQCFG3 (DTSE Sequence Configuration 3, 4000 008Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG3.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG3.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG3.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.18 DTSESEQCFG4

Register 23-18 DTSESEQCFG4 (DTSE Sequence Configuration 4, 4000 0090h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG4.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG4.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG4.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.19 DTSESEQCFG5

Register 23-19 DTSESEQCFG5 (DTSE Sequence Configuration 5, 4000 0094h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG5.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG5.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG5.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.20 DTSESEQCFG6

Register 23-20 DTSESEQCFG6 (DTSE Sequence Configuration 6, 4000 0098h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG6.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG6.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG6.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.21 DTSESEQCFG7

Register 23-21 DTSESEQCFG7 (DTSE Sequence Configuration 7, 4000 009Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG7.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG7.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG7.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.22 DTSESEQCFG8

Register 23-22 DTSESEQCFG8 (DTSE Sequence Configuration 8, 4000 00A0h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX)
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG8.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG8.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG8.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.23 DTSESEQCFG9

Register 23-23 DTSESEQCFG9 (DTSE Sequence Configuration 9, 4000 00A4h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG9.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG9.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG9.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.24 DTSESEQCFG10

Register 23-24 DTSESEQCFG10 (DTSE Sequence Configuration 10, 4000 00A8h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG10.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG10.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG10.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.25 DTSESEQCFG11

Register 23-25 DTSESEQCFG11 (DTSE Sequence Configuration 11, 4000 00ACh)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG11.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG11.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG11.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.26 DTSESEQCFG12

Register 23-26 DTSESEQCFG12 (DTSE Sequence Configuration 12, 4000 00B0h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG12.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG12.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG12.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.27 DTSESEQCFG13

Register 23-27 DTSESEQCFG13 (DTSE Sequence Configuration 13, 4000 00B4h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG13.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG13.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG13.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.28 DTSESEQCFG14

Register 23-28 DTSESEQCFG14 (DTSE Sequence Configuration 14, 4000 00B8h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG14.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG14.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG14.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.29 DTSESEQCFG15

Register 23-29 DTSESEQCFG15 (DTSE Sequence Configuration 15, 4000 00BCh)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG15.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG15.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG15.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.30 DTSESEQCFG16

Register 23-30 DTSESEQCFG16 (DTSE Sequence Configuration 16, 4000 00C0h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG16.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG16.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG16.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.31 DTSESEQCFG17

Register 23-31 DTSESEQCFG17 (DTSE Sequence Configuration 17, 4000 00C4h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG17.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG17.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG17.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.32 DTSESEQCFG18

Register 23-32 DTSESEQCFG18 (DTSE Sequence Configuration 18, 4000 00C8h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX)
18	**IRQEN**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG8.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG8.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG8.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.33 DTSESEQCFG19

Register 23-33 DTSESEQCFG19 (DTSE Sequence Configuration 19, 4000 00CCh)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG19.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG19.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG19.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.34 DTSESEQCFG20

Register 23-34 DTSESEQCFG20 (DTSE Sequence Configuration 20, 4000 00D0h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG20.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG20.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG20.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.35 DTSESEQCFG21

Register 23-35 DTSESEQCFG21 (DTSE Sequence Configuration 21, 4000 00D4h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG21.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG21.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG21.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.36 DTSESEQCFG22

Register 23-36 DTSESEQCFG22 (DTSE Sequence Configuration 22, 4000 00D8h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG22.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG22.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG22.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.37 DTSESEQCFG23

Register 23-37 DTSESEQCFG23 (DTSE Sequence Configuration 23, 4000 00DCh)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:20	**Reserved**	RO	0	Reserved
19	**NOCVT**	RW	0	No ADC conversion. 0b: Perform ADC conversion and send EMUX as configured 1b: Do not perform ADC conversion (only send EMUX) This feature is not available in VER1 of the MCU.
18	**IRQEN**	RW	0	Assert IRQ after converting this sequence. 0b: Do not assert IRQ 1b: Assert IRQ
17:16	**IRQNUM**	RW	0	IRQ number to assert. If **DTSESEQCFG23.IRQEN** = 1b, assert this IRQ signal to the NVIC when this sequence has completed conversion: 00b: ADC0_IRQ 01b: ADC1_IRQ 10b: ADC2_IRQ 11b: ADC3_IRQ
15:8	**EMUXD**	RW	0	EMUX data to send. If **DTSESEQCFG23.EMUXC** = 01b (Send before sample and hold) or 10b (Send after sample and hold), this 8-bit data is written to the EMUX. If **DTSESEQCFG23.EMUXC** = 00b, then EMUX data is not sent.
7	**SEQDONE**	RW	0	Final sequence of series. Indicates if this sequence is the final sequence for this conversion series. 0b: Not the final sequence, proceed to next sequence entry 1b: Final sequence in this series
6:5	**Reserved**	RO	0	Reserved
4:2	**CHANNEL**	RW	0	ADC Channel. This is the ADC channel to convert during this sequence: 000b: ADC0 001b: ADC1 010b: ADC2 011b: ADC3 100b: ADC4 101b: ADC5 110b: ADC6 111b: ADC7
1:0	**EMUXC**	RW	0	EMUX Control. This controls the behavior of the EMUX during this conversion sequence: 00b: Do not send EMUX command 01b: Start sending the EMUX command at the beginning of the ADC conversion cycle[1] 10b: Start sending the EMUX command after sample and hold 11b: Reserved [1] The ADC Conversion cycle consists of 4 ADC clocks for sample and hold followed by 12 ADC clocks for ADC conversion.

23.8.38 DTSERES0

Register 23-38 DTSERES0 (DTSE Conversion Result 0, 4000 0100h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 0.

23.8.39 DTSERES1

Register 23-39 DTSERES1 (DTSE Conversion Result 1, 4000 0104h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 1.

23.8.40 DTSERES2

Register 23-40 DTSERES2 (DTSE Conversion Result 2, 4000 0108h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 2.

23.8.41 DTSERES3

Register 23-41 DTSERES3 (DTSE Conversion Result 3, 4000 010Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 3.

23.8.42 DTSERES4

Register 23-42 DTSERES4 (DTSE Conversion Result 4, 4000 0110h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 4.

23.8.43 DTSERES5

Register 23-43 DTSERES5 (DTSE Conversion Result 5, 4000 0114h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 5.

23.8.44 DTSERES6

Register 23-44 DTSERES6 (DTSE Conversion Result 6, 4000 0118h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 6.

23.8.45 DTSERES7

Register 23-45 DTSERES7 (DTSE Conversion Result 7, 4000 011Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 7.

23.8.46 DTSERES8

Register 23-46 DTSERES8 (DTSE Conversion Result 8, 4000 0120h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 0.

23.8.47 DTSERES9

Register 23-47 DTSERES9 (DTSE Conversion Result 9, 4000 0124h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 9.

23.8.48 DTSERES10

Register 23-48 DTSERES10 (DTSE Conversion Result 10, 4000 0128h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 10.

23.8.49 DTSERES11

Register 23-49 DTSERES11 (DTSE Conversion Result 11, 4000 012Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 11.

23.8.50 DTSERES12

Register 23-50 DTSERES12 (DTSE Conversion Result 12, 4000 0130h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 12.

23.8.51 DTSERES13

Register 23-51 DTSERES13 (DTSE Conversion Result 13, 4000 0134h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 13.

23.8.52 DTSERES14

Register 23-52 DTSERES14 (DTSE Conversion Result 14, 4000 0138h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 14.

23.8.53 DTSERES15

Register 23-53 DTSERES15 (DTSE Conversion Result 15, 4000 013Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 15.

23.8.54 DTSERES16

Register 23-54 DTSERES16 (DTSE Conversion Result 16, 4000 1240h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 16.

23.8.55 DTSERES17

Register 23-55 DTSERES17 (DTSE Conversion Result 17, 4000 0144h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 17.

23.8.56 DTSERES18

Register 23-56 DTSERES18 (DTSE Conversion Result 18, 4000 0148h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 18.

23.8.57 DTSERES19

Register 23-57 DTSERES19 (DTSE Conversion Result 19, 4000 014Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 19.

23.8.58 DTSERES20

Register 23-58 DTSERES20 (DTSE Conversion Result 20, 4000 0150h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 20.

23.8.59 DTSERES21

Register 23-59 DTSERES21 (DTSE Conversion Result 21, 4000 0154h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 21.

23.8.60 DTSERES22

Register 23-60 DTSERES22 (DTSE Conversion Result 22, 4000 0158h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 22.

23.8.61 DTSERES23

Register 23-61 DTSERES23 (DTSE Conversion Result 23, 4000 015Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:12	**Reserved**	RO	0	Reserved
11:0	**RES**	RO	0	DTSE conversion result for sequence 23.

24 CRC

24.1 Overview

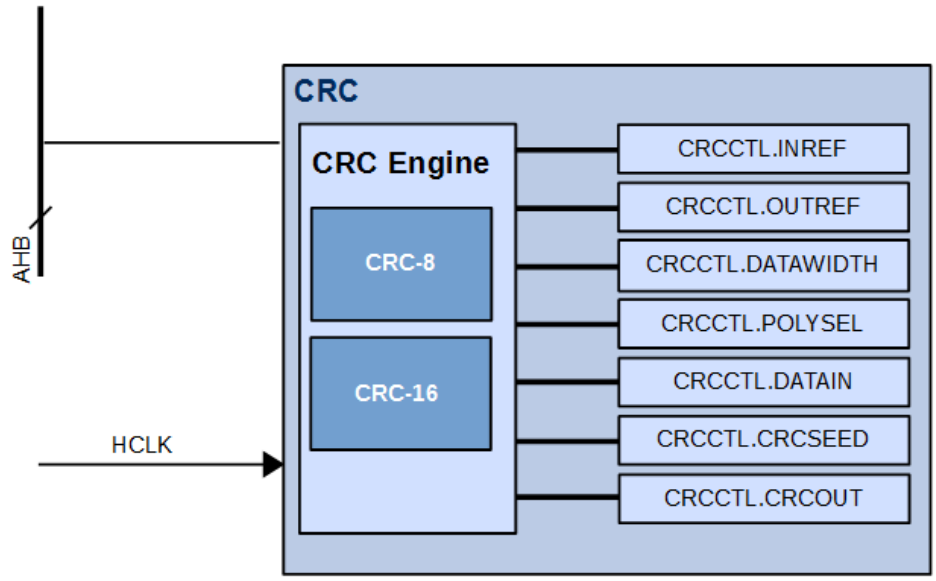
The PAC55XX contains a CRC peripheral that can be used by applications to perform data integrity validation. Examples of these types of applications could be:

- Memory Testing
- IEC60730 Class B Safety
- Communication Protocol CRC

This peripheral allows the user to write data into a CRC accumulator using the AHB bus, and the peripheral can quickly accumulate the CRC based on a selection of standard CRC polynomials.

24.2 Block Diagram

Figure 24-1 CRC Block Diagram



24.3 Features

- Polynomial Selection
  - CRC-16 (2) or CRC-8
- Configurable Data Input Width:
  - 8-bit or 32-bit
- Configurable Data Input Reflection

- Configurable Data Output Reflection
- User Defined Seed Value

24.4 Functional Description

24.4.1 CRC Polynomial

The user may select the CRC polynomial used for the CRC calculation from three choices. The table below shows the available CRC polynomials and the **CRCCTL.POLYSEL** value to choose for a given polynomial.

Table 24-1 CRC Polynomial Selections

NAME	POLYNOMIAL REPRESENTATION	CRCCTL.POLYSEL	NOTES
CRC-16-CCITT	0x1021	00b	Also known as CRC-CCITT
CRC-16-IBM	0x8005	01b	Also known as CRC-16 and CRC-16-ANSI
CRC-8-Dallas/Maxim	0x31	10b	

The user may set the seed for the CRC calculation by writing it to the **CRCSEED** register.

In order to save firmware instructions and processing time, data may be written into the CRC engine either as 8-bit data or 32-bit data into the **CRCDATAIN** register. To enter data into this register as 32-bit data, set the **CRCCTL.DATAWIDTH** to 0b. To enter data into this register as 8-bit data, set the **CRCCTL.DATAWIDTH** to 1b.

The CRC engine can reflect the input data or output CRC value. To reflect the input data from the **CRCDATIN** register, set the **CRCCTL.INREF** bit to 1b. To reflect the output data to the **CRCOUT** register, set the **CRCCTL.OUTREF** bit to 1b.

The CRC will accumulate into the read-only **CRCOUT** register.

24.5 Register Summary

Table 24-2 CRC Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
CRCCTL	400D 1000h	CRC Control	RW	0000 0000h
CRCDATAIN	400D 1004h	CRC Data Input Value	RW	0000 0000h
CRCSEED	400D 1008h	CRC Seed Value	RW	0000 0000h
CRCDATAOUT	400D 100Ch	CRC Data Output Value	RO	0000 0000h

24.6 Register Detail

24.6.1 CRCCTL

Register 24-1 CRCCTL (CRC Control, 400D 1000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**INREF**	RW	0	Reflect **DATAIN** input data to CRC engine: 0b: Do not reflect **DATAIN** input data 1b: Reflect **DATAIN** input data
3	**OUTREF**	RW	0	Reflect **DATAOUT** output data from CRC engine: 0b: Do not reflect **DATAOUT** output data 1b: Reflect **DATAOUT** output data
2	**DATAWIDTH**	RW	0b	DATAIN input data width: 0b: 32-bits – **DATAIN**[31:0] 1b: 8-bits – **DATAIN** [7:0]
1:0	**POLYSEL**	RW	00b	CRC Polynomial Select: 00b: CRC-16-CCITT 01b: CRC-16-IBM 10b: CRC-8-Dallas/Maxim 11b: Reserved

24.6.2 CRCDATAIN

Register 24-2 CRCDATAIN (CRC Data Input, 400D 1004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0		RW	0	CRC data input value. When **CRCCTL.DATAWIDTH** is set to 0b, data written to 31:0 used for CRC accumulation. When **CRCCTL.DATAWIDTH** is set to 1b, data written 7:0 used for CRC accumulation.

24.6.3 CRCSEED

Register 24-3 CRCSEED (CRC Seed Value, 400D 1008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**CRCSEED**	RW	0	CRC seed value. When **CRCCTL.POLYSEL** set to 00b (CRC-16-CCITT) or set to 01b (CRC-16-IBM), seed is located in 15:0. When **CRCCTL.POLYSEL** is set to 10b (CRC-8-Dallas/Maxim), seed is located in 7:0.

24.6.4 CRCOUT

Register 24-4 CRCOUT (CRC Data Output, 400D 100Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**CRCOUT**	RO	0	CRC data output value. When **CRCCTL.POLYSEL** set to 00b (CRC-16-CCITT) or set to 01b (CRC-16-IBM), data output value is located in 15:0. When **CRCCTL.POLYSEL** is set to 10b (CRC-8-Dallas/Maxim), data output value is located in 7:0.

25 USART A

25.1 Overview

The PAC55XX family contains support for four Universal Synchronous Asynchronous Receive Transmit (USART) peripherals.

A USART is a serial communications engine that may be configured for UART or SSP through a mode selection register.

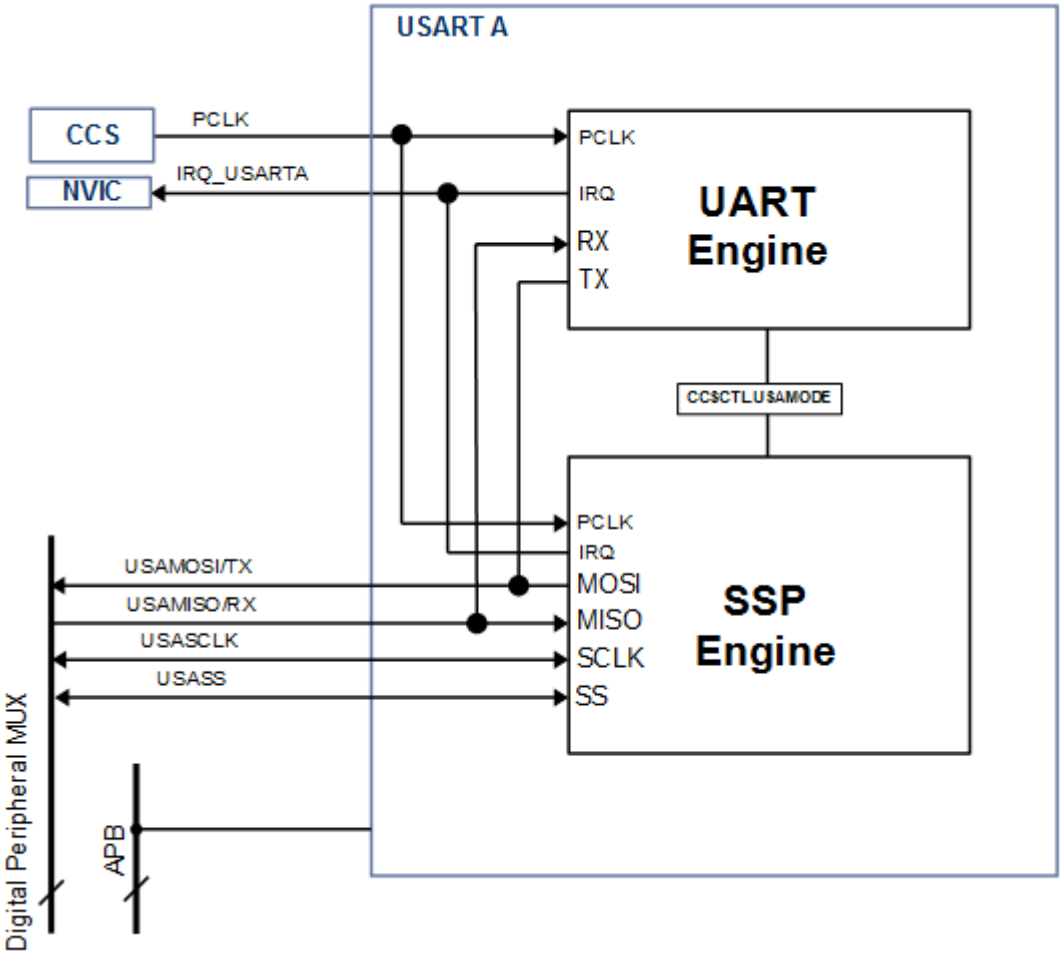
The USART is an APB client and is clocked by the Peripheral Clock (PCLK) system clock.

25.2 Features

- UART Mode:
  - 5-bit to 8-bit data interface
  - Optional parity
  - 1, 1.5 or 2 stop bits
  - 16-bit programmable baud-rate generator
  - 8-bit scratch pad
  - Independent RX and TX FIFOs
  - Configurable RX and TX interrupts
- SSP Mode:
  - Motorola SPI, TI Synchronous serial or National Semiconductor Microwire support
  - Master and Slave mode support
  - Independent RX and TX FIFOs
  - Configurable clock pre-scaler
  - Programmable Interrupts

25.3 System Block Diagram

Figure 25-1 USART A System Block Diagram



25.4 Functional Description

The USART may be configured for either UART or SSP mode via the **CCSCTL.USAMODE** register field.

In UART mode, the peripheral can receive or transmit UART serial data over the USAMOSI/TX (UART TX) or USAMISO/RX serial lines. In this mode, the USART signals for USASCLK and USASS are unused. The UART may assert the IRQ_USARTA0 signal to the NVIC for interrupts and is clocked using the PCLK system clock and connected to the APB bus.

In UART mode, only the UART registers may be accessed, starting at register offset 0.

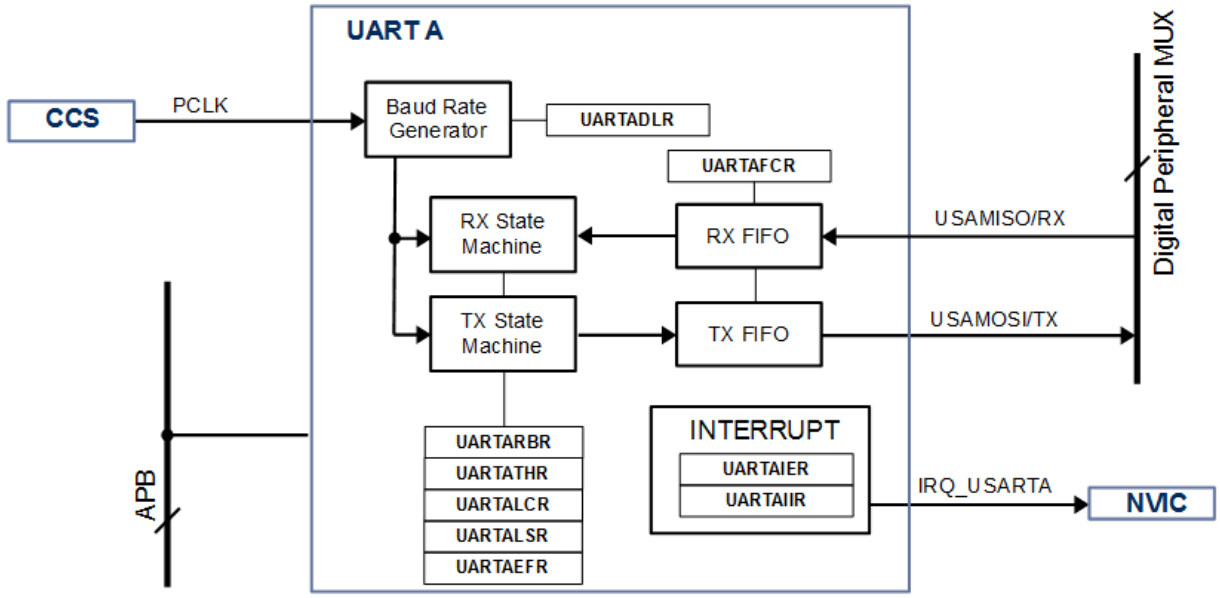
In SSP mode, the peripheral can receive or transmit serial data over the USAMOSI/TX (SPI MOSI) or USAMISO/RX (SPI MISO) serial lines. The USASCLK and USASS are used by the peripheral in either master or slave mode for the SSP clock and Slave Select signals. The SSP may assert the IRQ_USARTA signal to the NVIC for interrupts, and is clocked using the PCLK system clock and connected to the APB bus.

In SPI mode, only the SPI registers may be accessed, starting at register offset 0.

25.5 UART Mode Functional Description

25.5.1 UART Mode System Block Diagram

Figure 25-2 UART A Mode System Block Diagram



25.5.2 Mode Configuration

The UART mode of the USART must first be selected by writing **CCSCTL.USAMODE** to 1b. This will set the USART into UART mode and allow all configuration registers to be used in this mode.

The configuration and status registers that are available when **CCSCTL.USAMODE** is 1b are below.

Table 25-1 USART A UART Mode Registers

REGISTER	DESCRIPTION
UARTARBR	UART A Receive Buffer Register
UARTATHR	UART A Transmit Holding Register
UARTADLR	UART A Divisor Latch Register
UARTAIER	UART A Interrupt Enable Register
UARTAIIR	UART A Interrupt Identification Register
UARTAFCR	UART A FIFO Control Register
UARTALCR	UART A Line control Register
UARTALSR	UART A Line Status Register
UARTASCR	UART A Scratch Pad Register

25.5.3 Baud Rate Configuration

To configure the baud rate for RX/TX operations, the **UARTADLR** may be used.

To baud rate may be configured by the following formula:

Baud Rate = PCLK / (16 * **UARTADLR**)

Note that UART peripherals sometimes need an accurate timing base in order to function properly. Because of that, it is suggested that the CLKREF or EXTCLK be used as the clock source for PCLK when using the UART.

The ROSCCLK may not provide enough accuracy for UART applications.

25.5.4 FIFO Reset

The RX and TX FIFOs may be reset independently by using the **UARTAFC** register.

To change any of the bits in the **UARTAFCR**, the **UARTAFCR.FIFOEN** must be set to 1b (**UARTAFCR** access).

To reset the RX FIFO, write the **UARTAFCR.RXFIFORST** bit to 1b. This will clear all bytes in the FIFO and reset the pointer to the top of the FIFO for the RX state machine.

To reset the TX FIFO, write the **UARTAFCR.TXFIFORST** bit to 1b. This will clear all bytes in the FIFO and reset the pointer to the top of the FIFO for the TX state machine.

When the **UARTAFCR** register is done being used to configure the FIFOs, set the **UARTAFCR.FIFOEN** to 1b (application mode).

25.5.5 UART Configuration

In UART mode, the UART may be have the parity, break control, stop bits and word length configured by the user.

The parity checking may be enabled or disabled. To enable parity checking, set the **UARTALCR.PEN** to 1b. To disable, set this field to 0b. If enabled, the user may configure the type of parity checking as follows.

Table 25-2 UART A Parity Modes

UARTALCR.PEN	UARTALCR.PSEL	Parity	Description
0b	n/a	None	
1b	00b	Odd	The number of ones in the transmitted character plus the attached parity bit will be odd.
	01b	Even	The number of ones in the transmitted character plus the attached parity bit will be even.
	10b	Forced 1 stick	Force 1 for parity
	11b	Forced 0 stick	Force 0 for parity

To set the number of stop bits, the user may write the **UARTALCR.SBS** field. To select 1 stop bits, set **UARTALCR.SBS** to 0b. To select 2 stop bits, set **UARTALCR.SBS** to 1b. If the **UARTALCR.WLS** is set to 00b (5-bit character length) and **UARTALCR.SBS** is set to 1b, then there will be 1.5 stop bits.

To configure break control, the user can set the **UARTALCR.BCON** field. To disable break transmission, set **UARTALCR.BCON** to 0b. To force the TX signal to logic 0, set the **UARTALCR.BCON** to 1b.

To set the word length for the UART, the user can set the **UARTALCR.WLS** field as shown below.

Table 25-3 UART A Word Length

UARTALCR.WLS	UART word length
00b	5-bit
01b	6-bit
10b	7-bit
11b	8-bit

25.5.6 UART Scratch Pad

There is an 8-bit general purpose register that may be used in the USART.

The user may use the **USASCR** register for an 8-bit scratch pad.

25.5.7 UART Interrupts

The status of interrupts may be read at any time by the **UARTAIIR** register. If the **UARTAIIR.INTSTATUS** field is set to 0b, at least one interrupt is pending. The interrupt type can be read by the **UARTAIIR.INTID** field as shown below.

Table 25-4 UART A Interrupt ID

UARTAIIR.INTID	Interrupt Type
000b	Reserved
001b	TX Holding Register Empty
010b	Receive Data Available
011b	Receive Line Status
100b	Reserved
101b	Reserved
110b	Receive FIFO character time-out
111b	Reserved

25.5.8 Transmit Operation

Transmission may be initiated by writing **UARTATHR** with the data desired to be sent.

If the TX FIFO is enabled (**UARTAFCR.FIFOEN** = 1b), the data will be written into the TX FIFO. The contents of the TX FIFO will be transferred to the TX shift register one character at a time, until the TX FIFO is empty. The depth of the TX FIFO is 16-bytes.

If the TX FIFO is not enabled (**UARTAFCR.FIFOEN** = 0b), the data will be transferred to the TX shift register.

The bits to be transmitted are then shifted out of the TX shift register in the order of start bit, data bits (LSB first), parity bit, stop bit, using the configuration from the **UARTALCR** register.

The baud rate used will be set by the **UARTADLR** register.

25.5.9 Transmit Interrupts

If the TX FIFO is enabled (**UARTAFCR.FIFOEN** = 1b), the UART may be configured to generate interrupts after characters have been successfully transmitted. The UART may be configured to generate an interrupt after 1, 4, 8 or 14 characters have been transmitted. To set the interrupt threshold, the user should set the **UARTAEFR.ENMODE** = 1b and set the **UARTAFCR.TXTL** to the interrupt FIFO depth that is desired.

To enable transmit interrupts, set the **UARTAIER.THREIE** (TX holding register empty interrupt enable). When the transmit holding register has been emptied and the TX FIFO is empty (if **UARTAFCR.FIFOEN** = 1b), the **UARTALSR.THRE** will be set to 1b. This field will be cleared the next time the **UARTATHR** register is written with data to transmit. During this event, the **UARTAIIR.INTSTATUS** will be set to 1b (interrupt pending) and the **UARTAIIR.INTID** will be set to 001b (TX holding register empty).

If the **UARTAIER.THREIE** is set to 1b and any of the flags in **UARTALSR** are set, the IRQ_USARTA signal to the NVIC will be asserted.

25.5.10 Receive Operation

Data is sampled into the RX shift register at a sampling rate of PCLK / 16. A filter is used to remove spurious inputs that last for less than two periods of the sampling rate.

If the RX FIFO is enabled (**UARTAFCR.FIFOEN** = 1b), the data from the shift register will be loaded into the RX FIFO. The RX FIFO will load each character from the FIFO into the RX Buffer Register, until the FIFO is empty. The depth of the RX FIFO is 16-bytes.

If the RX FIFO is not enabled (**UARTAFCR.FIFOEN** = 0b), the received data will be loaded directly into the RX Buffer Register.

When the complete character has been loaded into the RX Buffer Register, it may be red using the **UARTARBR** register. The receiver checks the parity and stop bits as specified in the **UARTALCR** register.

25.5.11 Receive Interrupts

When characters are received from the UART, the user may configure the peripheral to indicate RX FIFO character time-out, incorrect parity, a missing stop bit (frame error) or other line status registers. To enable these interrupts, set the **UARTAIER.RLSIE** to 1b. If these conditions occur, the **UARTAIIR.INTSTATUS** will be set to 1b (interrupt pending) and the **UARTAIIR.INTID** will be set accordingly.

The UART may be configured to generate interrupts after a character has been successfully received.

To enable receive interrupts, set the **UARTAIER.RBRIE** (RX buffer register interrupt enable). When the **UARTARBR** (receive buffer register) has been filled, the **UARTALSR.RDR** will be set to 1b. This field will be cleared the next time the **UARTARBR** register is read.

If the RX FIFO is enabled (**UARTAFCR.FIFOEN** = 1b), then the FIFO will generate an interrupt when 1, 4, 8 or 14 characters have been loaded into it. To set the interrupt threshold set the **UARTAFCR.RXTL** to the interrupt FIFO depth that is desired. When the FIFO contains the

configured number of characters, the **UARTAIIR.INTSTATUS** will be set to a 1b (interrupt pending) and the **UARTAIIR.INTID** will be set accordingly.

If the **UARTAIER.RBRIE** is set to 1b and the **UARTALSR.RDR** interrupt flag is set, the IRQ_USARTA signal to the NVIC will be asserted.

25.6 UART Register Summary

Table 25-5 USART A Register Summary (UART Mode)

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
UARTARBR	4002 0000h	UART A Receive Buffer Register	RO	--
UARTATHR	4002 0004h	UART A Transmit Holding Register	WO	--
UARTADLR	4002 0008h	UART A Divisor Latch Register	RW	0000 0001h
UARTAIER	4002 000Ch	UART A Interrupt Enable Register	RW	0000 0000h
UARTAIIR	4002 0010h	UART A Interrupt Identification Register	RO	0000 0001h
UARTAFCR	4002 0014h	UART A FIFO Control Register	RW	0000 0000h
UARTALCR	4002 0018h	UART A Line control Register	RW	0000 0000h
UARTALSR	4002 0020h	UART A Line Status Register	RO	0000 0060h
UARTASCR	4002 0028h	UART A Scratch Pad Register	RW	--
UARTAEFR	4002 002Ch	UART A Enhanced Mode Register	RW	0000 000h

25.7 UART Register Detail

25.7.1 UARTARBR

Register 25-1 UARTARBR (UART A Receive Buffer Register, 4002 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**RBR**	RO	0	Contains the oldest received character in the UART RX FIFO.

25.7.2 UARATHR

Register 25-2 UARTATHR (UART A Transmit Holding Register, 4002 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**THR**	WO	-	Writing to the **UARTATHR** causes the data to be stored in the UART transmit FIFO. The character will be sent when it reaches the bottom of the FIFO and the transmitter is available.

25.7.3 UARTADLR

Register 25-3 UARTADLR (UART A Divisor Latch Register, 4002 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**DLR**	RW	0000 0001b	Sets the baud rate for the module. Baud rate = PCLK / (16 * **UARTADLR**)

25.7.4 UARTAIER

Register 25-4 UARTAIER (UART A Interrupt Enable Register, 4002 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:3	**Reserved**	RO	0	Reserved
2	**RLSIE**	RW	0	RX Line Status Interrupt Enable. Enables the UART RX line status interrupts. The status of this interrupts can be read from **UARTALSR[4:1]**. 0b: Disable the RX line status interrupts 1b: Enable the RX line status interrupts
1	**THRIE**	RW	0	TX Holding Register Empty Interrupt Enable. Enables the THRE interrupt for the UART. The status of this interrupt can be read from **UARTALSR.THRE**. 0b: Disable the THRE interrupts 1b: Enable the THRE interrupts
0	**RBRIE**	RW	0	RX Buffer Register Interrupt Enable. Enables the Receive Data Available interrupt for the UART. It also controls the character receive time-out interrupt. 0b: Disable the RBR interrupts 1b: Enable the RBR interrupts

25.7.5 UARTAIIR

Register 25-5 UARTAIIR (UART A Interrupt Identification Register, 4002 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3:1	**INTID**	RO	000b	Interrupt identification: 000b: Reserved 001b: TX Holding Register Empty 010b: Receive Data Available 011b: Receive Line Status 100b: Reserved 101b: Reserved 110b: Receive FIFO Character Time-out 111b: Reserved
0	**INTSTATUS**	RO	1b	Interrupt status. Note that this bit is active low. The pending interrupt can be determined through the **UARTAIIR.INTID** field. 0b: At least one interrupt is pending 1b: No interrupt is pending

Note that this register is cleared on read.

25.7.6 UARTAFCR

Register 25-6 UARTAFCR (UART A FIFO Control Register, 4002 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:6	**RXTL**	RW	0	RX Trigger Level. These two bits determine how many receive UART FIFO characters must be written before an interrupt is activated. 00b: 1 character 01b: 4 characters 10b: 8 characters 11b: 14 characters
5:4	**TXTL**	RW	0	TX Trigger Level. These two bits determine how may transmit UART FIFO characters must be written before an interrupt is activated. 00b: 1 character 01b: 4 characters 10b: 8 characters 11b: 14 characters
3	**Reserved**	RO	0	Reserved
2	**TXFIFORST**	RW	0	TX FIFO Reset. 0b: No effect 1b: Clears all bytes in the UART TX FIFO. This bit is self-clearing
1	**RXFIFORST**	RW	0	RX FIFO Reset. 0b: No effect 1b: Clears all bytes in the UART RX FIFO. This bit is self-clearing
0	**FIFOEN**	RW	0	FIFO Enable: 0b: UART FIFOs are disabled. Reset UARTAFCR settings to their default values. 1b: UART RX and TX FIFOs are enabled. **UARTAFCR[7:1]** are accessible when this bit is set. Any transition on this bit will automatically clear the UART RX AND TX FIFOs.

25.7.7 UARTALCR

Register 25-7 UARTALCR (UART A Line Control Register, 4002 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:7	**Reserved**	RO	0	Reserved
6	**BCON**	RW	0	Break Control: 0b: Disable break transmission 1b: Enable break transmission. Output pin UARTATXD is forced to logic 0 when this bit is set.
5:4	**PSEL**	RW	0	Parity Select: 00b: Odd parity. The number of 1s in the transmitted character and the attached parity will be odd. 01b: Even parity. The number of 1s in the transmitted character and the attached parity will be even. 10b: Forced 1 stick parity. 11b: Forced 0 stick parity.
3	**PEN**	RW	0	Parity Enable: 0b: Disable parity generation and checking 1b: Enable parity generation and checking
2	**SBS**	RW	0	Stop Bit Select: 0b: 1 stop bit 1b: 2 stop bits, 1.5 stop bits if (**UARTALCR.WLS** = 00b)
1:0	**WLS**	RW	0	Word Length Select: 00b: 5-bit character length 01b: 6-bit character length 10b: 7-bit character length 11b: 8-bit character length

25.7.8 UARTALSR

Register 25-8 UARTALSR (UART A Line Status Register, 4002 0020h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**RXFE**	RO	0	Error in RX FIFO. This bit is set when a character with a RX error such as framing, parity or break is loaded into **UARTARBR**. This bit is cleared when the **UARTALSR** register is read, and there are no subsequent errors in the UART FIFO.
6	**TEMT**	RO	0	Transmitter Empty. This bit is set when both **UARTATHR** and **UARTATSR** are empty; this bit is cleared when THR contains valid data. 0b: THR contains valid data 1b: THR is empty
5	**THRE**	RO	0	Transmitter Holding Register Empty. This bit is set immediately upon detection of an empty UART THR and is cleared on a THR write. 0b: THR contains valid data 1b: THR is empty
4	**BI**	RO	0	Break Interrupt. When UARTARX is held in the spacing state (all zeroes) for one full character transmission (start, data, parity, stop), a break interrupt occurs. Once the break condition has been detected, the receiver goes idle until UARTARX goes to the marking statue (all ones). A read of the **UARTALSR** register clears this status bit. The time of break detection is dependents on **UARTAFCR.FIFOEN**. The break interrupt is associated with the character at the top of the UART RBR FIFO. 0b: Break interrupt status is inactive 1b: Break interrupt status is active
3	**FE**	RO	0	Framing Error. When the stop bit of a received character is a logic 0, a framing error occurs. Reading the **UARTALSR** register will clear this bit. The time of the framing error is dependent upon **UARTAFCR[3]**. Upon detection of a framing error, the RX will attempt to re-synchronize the data and assume that the bad stop bit is actually an early start bit. However, it cannot be assumed that the next received byte will be correct even if there is no framing error. A framing error is associated with the character at the top of the UART RBR FIFO. 0b: Framing error status is inactive 1b: Framing error status is active
2	**PE**	RO	0	Parity Error. When the parity bit of a received character is in the wrong state, a parity error occurs. Reading the **UARTALSR** register will clear this bit. The time of the parity error detection is dependent on **UARTAFCR.FIFOEN**.

				A parity error is associated with the character at the top of the UART RBR FIFO. 0b: Parity error status is inactive 1b: Parity error status is active
1	**OE**	RO	0	Overrun Error. The overrun error condition is set as soon as it occurs. Reading the **UARTALSR** register will clear this bit. This bit is set when the UART RSR has a new character assembled and the UART RBR FIFO is full. In this case, the UART RBR FIFO will not be overwritten and the character in the UART RSR will be lost. 0b: Overrun error status is inactive 1b: Overrun error status is active
0	**RDR**	RO	0	Receiver Data Ready. This bit is set when the RBR holds an unread character and is cleared when the UART RBR FIFO is empty. 0b: RBR is empty 1b: RBR contains valid data

25.7.9 UARTASCR

Register 25-9 UARTASCR (UART A Scratch Pad Register, 4002 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**PAD**	RW	-	A read-able, write-able byte.

25.7.10 UARTAEFR

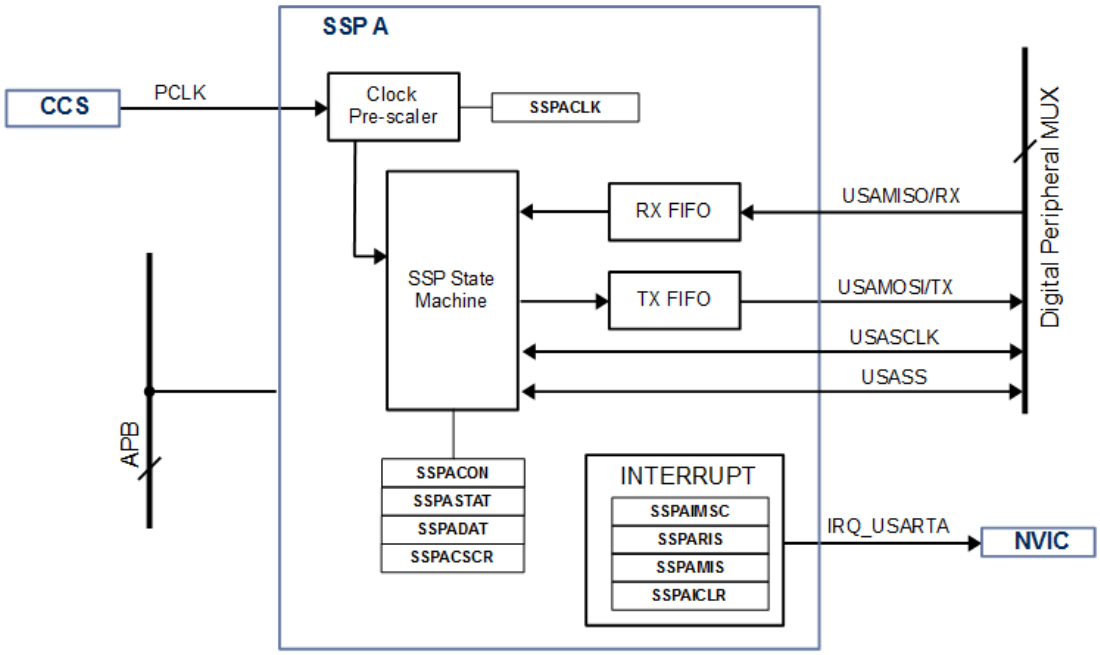
Register 25-10 UARTAEFR (UART A Enhanced Feature Register, 4002 002Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**ENMODE**	RW	0	Enhanced Mode 0b – disabled 1b – enabled
3:0	**Reserved**	RO	0	Reserved

25.8 SSP Functional Description

25.8.1 SSP Mode System Block Diagram

Figure 25-3 SSP A System Block Diagram



25.8.2 Mode Configuration

The SSP mode of the USART must first be selected by writing **CCSCTL.USAMODE** to 0b. This will set the USART into SSP mode and allow all configuration registers to be used in this mode.

The configuration and status registers that are available when **CCSCTL.USAMODE** is 0b are below.

Table 25-6 USART A UART Mode Registers

REGISTER	DESCRIPTION
SSPACON	SSP A Control Register
SSPASTAT	SSP A Status Register
SSPADAT	SSP A Data Register
SSPACLK	SSP A Clock Control Register
SSPAIMSC	SSP A Interrupt Mask Set and Clear Register
SSPARIS	SSP A Raw Interrupt Status Register
SSPAMIS	SSP A Masked Interrupt Status Register

SSPAICLR	SSP A Interrupt Clear Register
SSPASSCR	SSP A Slave Select Configuration Register

25.8.3 SSP Overview

When in SSP mode, the USART may be a master or slave interface that enables synchronous serial communication with slave or master peripherals having one of the following:

- Motorola-style SPI
- TI-style Synchronous Serial Interface
- National Semiconductor-style Microwire

In both master and slave modes, the SSP performs parallel to serial conversion on data into 32-bit wide, 8-location deep RX and TX FIFOs.

The SSP may be configured to generate interrupts for servicing the TX and RX FIFOs and error conditions such as FIFO overrun and timeout.

25.8.4 Clock Configuration

When configured for master mode, the SSP peripheral may configure the clock pre-scaler to generate the desired SCLK output clock frequency. The input clock on the SSP peripheral is the PCLK (peripheral clock) system clock.

The SCLK is generated from the PCLK input and the **SSPACLK.M** and **SSPACLK.N** parameters. The SCLK clock frequency is generated according to the following formula:

F_{SSPCLK} = PCLK / ((**SSPACLK.M** + 1) * **SSPACLK.N**)

In this formula, **SSPACLK.N** must be an even value from 2 to 254.

25.8.5 SSP Clock Constraints

The SSP clock configuration must be configured according to the guidelines below, in order for the peripheral to operate correctly.

Here are the clock constraints for the input clock SSPCLK (after the pre-scaler) and output SSP clock, USACLK:

- When in master mode, the minimum frequency of the input SSPCLK must be at least 2X the output SSP clock (USASCLK)
- When in slave mode, the minimum frequency of the input SSPCLK must be at least 12X the input SSP clock (USASCLK)

The maximum frequency of the SSPCLK should be:

- When in master mode, F_{SSPCLK}(max) <= 254 * 128 * $F_{USASCLK}$ (SSP clock output)
- When in slave mode, F_{SSPCLK}(max) <= 254 * 128 * $F_{USASCLK}$ (SSP clock input)

25.8.6 SSP Configuration

The SSP may be configured to support various features of master and slave based serial interfaces.

To enable the SSP controller, set **SSPACON.SSPEN** to 1b. When disabled, the SSP may still have registers configured for the configuration mode and interrupts, but will not operate on serial data until enabled. Once the SSP has been configured, set this bit to 1b to begin processing.

To configure the SSP mode, set **SSPACON.FRF** (frame format) to the desired value as shown below.

Table 25-7 SSP A Frame Format

SSPACON.FRF	Frame Format
00b	SPI (Motorola)
01b	Synchronous Serial Format (TI)
10b	Microwire (National Semiconductor)
11b	Reserved

To select master mode, set **SSPACON.MS** to 0b. In this mode, the controller is the bus master. It will drive the SCLK, MOSI and SS signals and will receive data on the MISO line.

To select slave mode, set **SSPACON.MS** to 1b. In this mode, the controller is the bus slave and will drive the MISO line and receive input from SCLK, MOSI and SS.

To change the serial to parallel endian order, the user may use the **SSPACON.LSBFIRST** field. To configure the SSP for LSB first, set **SSPACON.LSBFIRST** to 1b. To configure the SSP for MSB first, set **SSPACON.LSBFIRST** to a 0b.

To enable loop-back mode, set the **SSPACON.LBM** to 1b. In this mode, serial input is taken from the serial output instead of from the digital peripheral MUX.

While in slave mode, the behavior of the MISO output can be controlled. When **SSPACON.SOD** is set to 0b, the SSP can drive the MISO in slave mode. When this bit is 1b, the SSP will not drive the MISO output in slave mode.

The clock phase of the SSP may also be configured. To configure the SSP for capturing data on the first clock edge transition, set **SSPACON.CPH** to 0b. To configure the SSP for capturing data on the second clock edge transition, set **SSPACON.CPH** to 1b.

When the **SSPACON.FRF** is 00b (SPI), the clock output polarity may be configured. To configure the SSP clock to be active high, set **SSPACON.CPO** to 0b. To configure the SSP clock to be active low, set **SSPACON.CPO** to 1b.

The SSP may also configure the size of the data word size transferred in each frame. To configure the data size, see the table below.

Table 25-8 SSP A Data Size Select

SSPACON.DSS	Data Size
0 0000b	Reserved
0 0001b	Reserved
0 0010b	Reserved
0 0011b	4-bit
0 0100b	5-bit
0 0101b	6-bit
0 0110b	7-bit
0 0111b	8-bit
0 1000b	9-bit
0 1001b	10-bit
0 1010b	11-bit
0 1011b	12-bit
0 1100b	13-bit
0 1101b	14-bit
0 1110b	15-bit
0 1111b	16-bit
…	…
1 1111b	32-bit

25.8.7 SSP Slave Select Configuration

The SSP allows the behavior of the SS (slave select) signal to be configured by using the **SSPASSCR** register as follows when configured as a bus master.

The **SSPASSCR.SELCS** field must always be set to a 00b for proper operation.

To configure the SSP to automatically control the behavior of the SS signal, set **SSPASSCR.SWSEL** to 0b. If the user wants to control the SS signal by software, set **SSPASSCR.SWSEL** to 1b.

To configure the SS signal behavior after a frame transfer, use the **SSPASSCR.SPHDONTCARE** field. To configure the SSP to not pull the SS signal high after a frame transfer, set **SSPASSCR.SPHDONTCARE** to 0b. To configure the SSP to pull the SS signal high after a frame transfer, set the **SSPASSCR.SPHDONTCARE** to 1b.

25.8.8 SSP Interrupts

The SSP may configure interrupts for the following conditions.

When the SSP detects that the TX FIFO is half-full or less, the **SSPARIS.TXRIS** and **SSPAMIS.TXMIS** are set to 1b. During this condition, if the **SSPAIMSC.TXIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTA signal to the NVIC.

When the SSP detects that the RX FIFO is half-full or more, the **SSPARIS.RXRIS** and **SSPAMIS.RXMIS** are set to 1b. During this condition, if the **SSPAIMSC.RXIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTA signal to the NVIC.

The SSP contains a time for reading the RX FIFO. If the RX FIFO is not empty, and 32 F_{SSPCLK} periods have gone by, the **SSPARIS.RTIM** and **SSPAMIS.RTMIS** bits are set to 1b. During this condition, if the **SSPAIMSC.RTIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTA signal to the NVIC. This condition may be cleared by writing **SSPAICLR.RTIC** to 1b.

If the SSP detects the RX FIFO is full, when a character is attempted to be inserted into it, is sets the **SSPARIS.ROIM** and **SSPAMIS.ROMIS** to 1b. During this condition, if the **SSPAIMSC.ROIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTA signal to the NVIC. This condition may be cleared by writing **SSPAICLR.ROIC** to 1b.

25.8.9 Master Mode Operation

When configured as a bus master (**SSPACON.MS** = 0b), the SSP will insert a character into the TX FIFO when it is written into the **SSPADAT** register. The SSP will read a character from the TX FIFO and performs a parallel to serial conversion on it. Then the serial data stream and frame control signal is synchronized to the clock and are output through the USAMOSI/TX pin to the attached slaves. While the data is being transmit to the slaves, the USASCLK and USASS behave as configured in the **SSPACON** and **SSPACSCR** registers.

The master receive logic performs serial to parallel conversion on the incoming synchronous USAMISO/RX data stream, and stores the character in the RX FIFO. The master may read first entry in the RX FIFO by reading the **SSPADAT** register.

25.8.10 Slave Mode Operation

When configured as a bus slave (**SSPACON.MS** = 1b), the SCLK is provided by the attached master. The user can write the next character to send into **SSPADAT** and it will get inserted into the TX FIFO. The slave transmit logic reads a value from the TX FIFO, performs parallel to serial conversion and outputs the serial data stream on the USAMISO/RX pin to the attached master. The slave receive logic performs serial to parallel conversion on the incoming USAMOSI/TX data stream, extracting and storing values into the RX FIFO. The first character in the RX FIFO may be read by reading the **SSPADAT** register.

25.8.11 SSP Status

The SSP module maintains status information on the operation of the module. The table below shows the available SSP status conditions.

Table 25-9 SSP A Status

SSPASTAT field	Description
BSY	If set, the SSP controller is currently sending/receiving a frame and/or the TX FIFO is not empty.
RFF	If set, the RX FIFO is full.
RNE	If set, the RX FIFO is not empty.
TNF	If set, the TX FIFO is not full.
TFE	If set, the TX FIFO is empty

25.9 SSP Register Summary

Table 25-10 USART A Register Summary (SSP Mode)

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
SSPACON	4002 0000h	SSP A Control Register	RW	0000 0000h
SSPASTAT	4002 0004h	SSP A Status Register	RO	0000 0003h
SSPADAT	4002 0008h	SSP A Data Register	RW	0000 0000h
SSPACLK	4002 000Ch	SSP A Clock Control Register	RW	0000 0000h
SSPAIMSC	4002 0010h	SSP A Interrupt Mask Set and Clear Register	RW	0000 0000h
SSPARIS	4002 0014h	SSP A Raw Interrupt Status Register	RO	0000 0008h
SSPAMIS	4002 0018h	SSP A Masked Interrupt Status Register	RO	0000 0000h
SSPAICLR	4002 001Ch	SSP A Interrupt Clear Register	RW	0000 0000h
SSPASSCR	4002 0028h	SSP A Slave Select Configuration Register	RW	0000 0000h

25.10 SSP Register Detail

25.10.1 SSPACON

Register 25-11 SSPACON (SSP A Control Register, 4002 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:14	**Reserved**	RO	0	Reserved
13	**LSBFIRST**	RW	0	Endian Order: 0b: MSB transmit / receive first 1b: LSB transmit / receive first
12	**LBM**	RW	0	Loopback Mode: 0b: Normal operation 1b: Loopback mode. Serial Input is taken from the serial output (MOSI or MISO) rather than the serial in (MISO or MOSI respectively)
11	**SSPEN**	RW	0	SSP Enable: 0b: SSP Controller is disabled 1b: SSP Controller is enabled. It will interact with other devices on the serial bus. Software should write the appropriate control information to the other SPI/SSP registers and interrupt controller registers, before setting this bit.
10	**MS**	RW	0	Master/Slave Mode: 0b: The SSP controller acts as a bus master, driving the USASCLK, USAMOSI and USSS signals 1b: The SSP controller acts as a bus slave, driving the USAMISO and receiving USASCLK, USAMOSI and USASS.
9	**SOD**	RW	0	Slave Output Disable: 0b: The SSP can drive the USAMISO output in slave mode 1b: The SSP must not drive the USAMISO output in slave mode
8	**CPH**	RW	0	Clock Out Phase. This bit is only used when **SSPACON.FRF** = 0b (SPI). 0b: The SSP controller captures serial data on the first edge transition of the frame. 1b: The SSP controller captures serial data on the second edge transition of the frame.
7	**CPO**	RW	0	Clock Out Polarity (this bit is only used in SPI mode, when **SSPACON.FRF** = 00b): 0b: The SSP clock is active high. 1b: The SSP clock is active low.
6:5	**FRF**	RW	0	Frame Format: 00b: SPI 01b: TI 10b: Microwire 11b: Reserved
4:0	**DSS**	RW	0	Data Size Select: 0 0000b: Reserved 0 0001b: Reserved 0 0010b: Reserved 0 0011b: 4-bit transfer 0 0100b: 5-bit transfer

				0 0101b: 6-bit transfer 0 0110b: 7-bit transfer 0 0111b: 8-bit transfer 0 1000b: 9-bit transfer 0 1001b: 10-bit transfer 0 1010b: 11-bit transfer 0 1011b: 12-bit transfer 0 1100b: 13-bit transfer 0 1101b: 14-bit transfer 0 1110b: 15-bit transfer 0 1111b: 16-bit transfer … 1 1111b: 32-bit transfer

25.10.2 SSPASTAT

Register 25-12 SSPASTAT (SSP A Status Register, 4002 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**BSY**	RO	0	Busy Bit: 0b: SPI controller is idle 1b: SPI controller is sending/receiving a frame and/or the TX FIFO is not empty
3	**RFF**	RO	0	Receive FIFO Full: 0b: RX FIFO not full 1b: RX FIFO full
2	**RNE**	RO	0	Receive FIFO not empty: 0b: RX FIFO empty 1b: RX FIFO not empty
1	**TNF**	RO	0	Transmit FIFO not full: 0b: TX FIFO full 1b: TX FIFO not full
0	**TFE**	RO	0	Transmit FIFO empty: 0b: TX FIFO not empty 1b: TX FIFO empty

25.10.3 SSPADAT

Register 25-13 SSPADAT (SSP A Data Register, 4002 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**DATA**	RW	0	Software can writing data to be sent in a future frame to this register when the **SSPASTAT.TNF** bit is set to 1b, indicating that the TX FIFO was previously empty and the SPI controller is not busy on the bus, transmission of the data will begin immediately. Otherwise, the data written to this register will be sent as soon as all previous data has been sent (and received). If the data length is less than 32-bits, the data must be right-justified in this register. Software may read data from this register when the **SSPASTAT.RNE** bit is set to a 1b, indicating that the RX FIFO is not empty. When software reads this register, the SPI controller returns the data from the least recent frame in the RX FIFO. If the data length is less than 32-bits, the data is right-justified in this field with the MSBs set to 0.

25.10.4 SSPACLK

Register 25-14 SSPACLK (SSP A Clock Register, 4002 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:8	**M**	RW	0	These fields may set the SPI master clock rate by the formula: F_{SSPCLK} = PCLK / ((**SSPACLK.M** + 1) * **SSPACLK.N**) N must be an even value from 2 to 254.
7:0	**N**	RW	0	

25.10.5 SSPAIMSC

Register 25-15 SSPAIMSC (SSP A Interrupt Mask Set and Clear Enable Register, 4002 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXIM**	RW	0	Transmit FIFO Interrupt Mask: 0b: TX FIFO half-full or less condition interrupt disabled 1b: TX FIFO half-full or less condition interrupt enabled
2	**RXIM**	RW	0	Receive FIFO Interrupt Mask: 0b: RX FIFO half-full or more condition interrupt disabled 1b: RX FIFO half-full or more condition interrupt enabled
1	**RTIM**	RW	0	Receive Timeout Interrupt Mask: 0b: RX FIFO not empty and no read prior to timeout period interrupt disabled 1b: RX FIFO not empty and no read prior to timeout period interrupt enabled
0	**ROIM**	RW	0	Receive Overrun Interrupt Mask: 0b: RX FIFO written to while full condition interrupt disabled 1b: RX FIFO written to while full condition interrupt enabled

25.10.6 SSPARIS

Register 25-16 SSPARIS (SSP A Raw Interrupt Status Register, 4002 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXRIS**	RO	1	TX FIFO Raw Interrupt Status: 0b: TX FIFO not at least half-full 1b: TX FIFO at least half-full
2	**RXRIS**	RO	0	RX FIFO Raw Interrupt Status: 0b: RX FIFO not at least half-full 1b: RX FIFO at least half-full
1	**RTRIS**	RO	0	RX FIFO Timeout Interrupt Status: 1b: RX FIFO is not empty and has not been read for the timeout period. The timeout period is 32-bit times at F_{SSPCLK}
0	**RORIS**	RO	0	RX FIFO Overwrite Interrupt Status: 1b: RX FIFO was full when another frame was completely received. The preceding data frame is overrun by the new frame when this occurs.

25.10.7 SSPAMIS

Register 25-17 SSPAMIS (SSP A Masked Interrupt Status Register, 4002 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXMIS**	RO	1	TX FIFO Masked Interrupt Status: 0b: TX FIFO not at least half-full 1b: TX FIFO at least half-full
2	**RXMIS**	RO	0	RX FIFO Masked Interrupt Status: 0b: RX FIFO not at least half-full 1b: RX FIFO at least half-full
1	**RTMIS**	RO	0	RX FIFO Timeout Interrupt Status: 1b: RX FIFO is not empty and has not been read for the timeout period. The timeout period is 32-bit times at F_{SSPCLK}
0	**ROMIS**	RO	0	RX FIFO Overwrite Interrupt Status: 1b: RX FIFO was full when another frame was completely received. The preceding data frame is overrun by the new frame when this occurs.

25.10.8 SSPAICLR

Register 25-18 SSPAICLR (SSP A Interrupt Clear Register, 4002 001Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:2	**Reserved**	RO	0	Reserved
1	**RTIC**	RW	0	RX FIFO Timeout Interrupt Clear: 0b: No effect 1b: Clears the **SSPAxIS.RTRIS** interrupt flag This bit is self-clearing.
0	**ROIC**	RW	0	RX FIFO Overrun Interrupt Clear: 0b: No effect 1b: Clears the **SSPAxIS.RORIS** interrupt flag This bit is self-clearing.

25.10.9 SSPASSCR

Register 25-19 SSPASSCR (SSP A Slave Select Configuration Register, 4002 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**SPHDONTCARE**	RW	0	Slave Select Pull High: 0b: USASS cannot pull high after frame transfer 1b: USASS must pull high after frame transfer
3	**SWSS**	RW	0	Slave Select State: 0b: Set USASS to low 1b: Set USASS to high
2	**SWSEL**	RW	0	Slave Select Software Control: 0b: USASS is automatically controlled by the SPI module 1b: USASS is software controlled by **SSPASSCR.SWSS**
1:0	**SELSS**	RW	0	Slave Select Signal Control: 00b: USASS is enabled 01b: Reserved 10b: Reserved 11b: Reserved

26 USART B

26.1 Overview

The PAC55XX family contains support for four Universal Synchronous Asynchronous Receive Transmit (USART) peripherals.

A USART is a serial communications engine that may be configured for UART or SSP through a mode selection register.

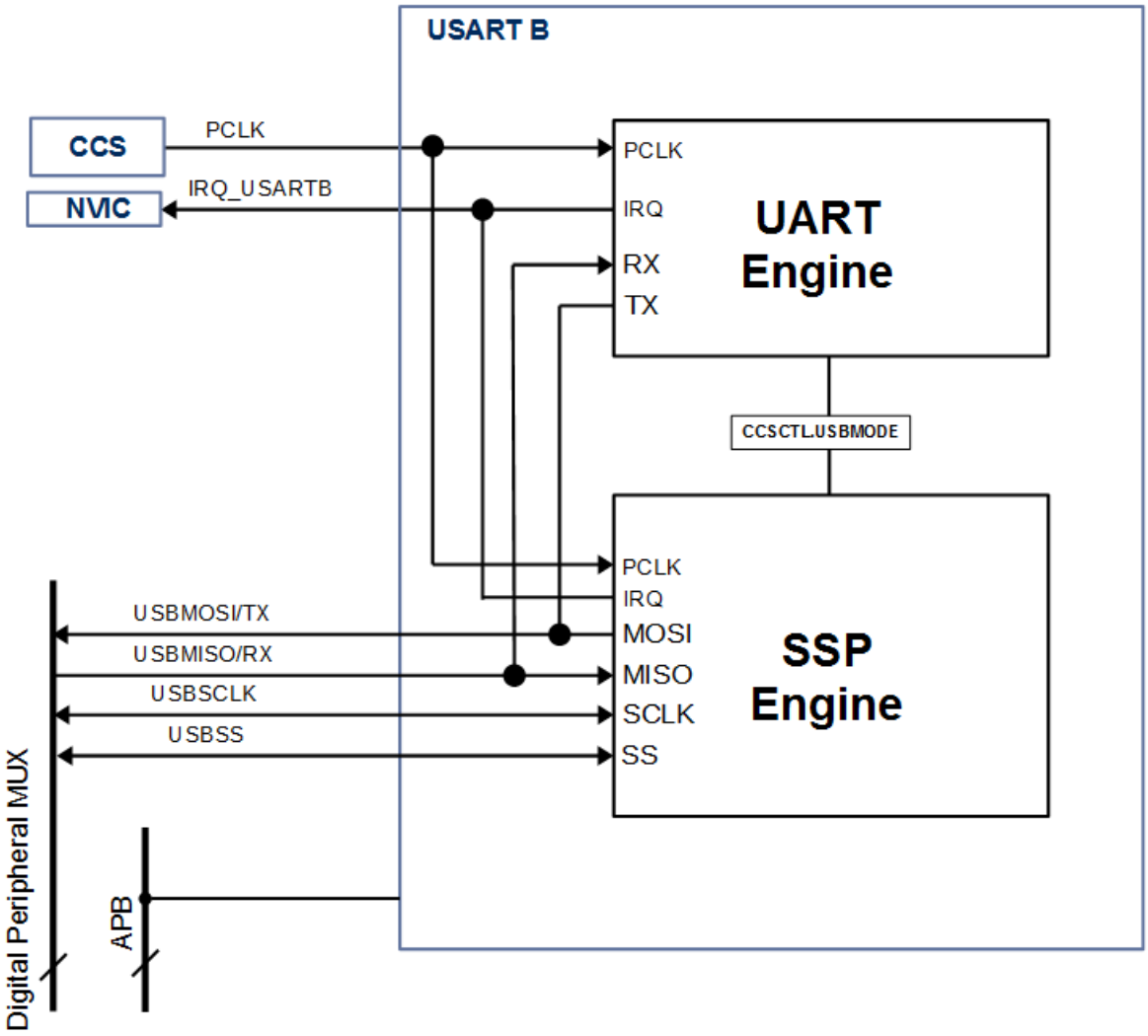
The USART is an APB client and is clocked by the Peripheral Clock (PCLK) system clock.

26.2 Features

- UART Mode:
  - 5-bit to 8-bit data interface
  - Optional parity
  - 1, 1.5 or 2 stop bits
  - 16-bit programmable baud-rate generator
  - 8-bit scratch pad
  - Independent RX and TX FIFOs
  - Configurable RX and TX interrupts
- SSP Mode:
  - Motorola SPI, TI Synchronous serial or National Semiconductor Microwire support
  - Master and Slave mode support
  - Independent RX and TX FIFOs
  - Configurable clock pre-scaler
  - Programmable Interrupts

26.3 System Block Diagram

Figure 26-1 USART B System Block Diagram



26.4 Functional Description

The USART may be configured for either UART or SSP mode via the **CCSCTL.USBMODE** register.

In UART mode, the peripheral can receive or transmit UART serial data over the USBMOSI/TX (UART TX) or USBMISO/RX serial lines. In this mode, the USART signals for USBSCLK and USBSS are unused. The UART may assert the IRQ_USARTB signal to the NVIC for interrupts and is clocked using the PCLK system clock and connected to the APB bus.

In UART mode, only the UART registers may be accessed, starting at register offset 0.

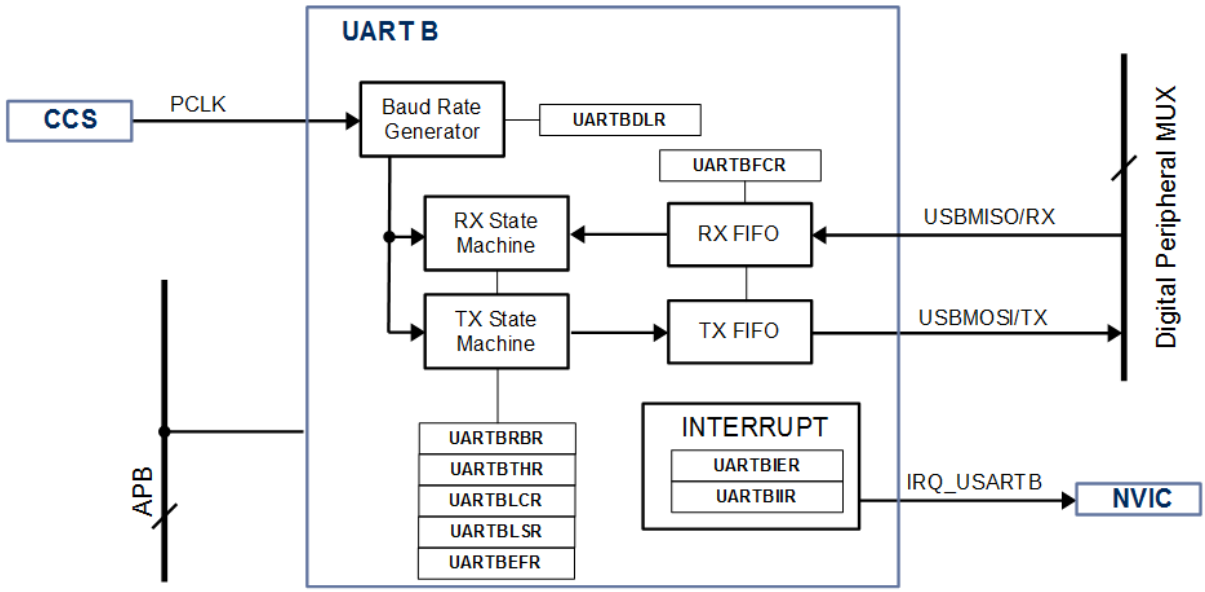
In SSP mode, the peripheral can receive or transmit serial data over the USBMOSI/TX (SPI MOSI) or USBMISO/RX (SPI MISO) serial lines. The USBSCLK and USBSS are used by the peripheral in either master or slave mode for the SSP clock and Slave Select signals. The SSP may assert the IRQ_USARTB signal to the NVIC for interrupts, and is clocked using the PCLK system clock and connected to the APB bus.

In SPI mode, only the SPI registers may be accessed, starting at register offset 0.

26.5 UART Mode Functional Description

26.5.1 UART B Mode System Block Diagram

Figure 26-2 UART B Mode System Block Diagram



26.5.2 Mode Configuration

The UART mode of the USART must first be selected by writing **CCSCTL.USBMODE** to 1b. This will set the USART into UART mode and allow all configuration registers to be used in this mode.

The configuration and status registers that are available when **CCSCTL.USBMODE** is 1b are below.

Table 26-1 USART B UART Mode Registers

REGISTER	DESCRIPTION
UARTBRBR	UART B Receive Buffer Register
UARTBTHR	UART B Transmit Holding Register
UARTBDLR	UART B Divisor Latch Register
UARTBIER	UART B Interrupt Enable Register
UARTBIIR	UART B Interrupt Identification Register
UARTBFCR	UART B FIFO Control Register
UARTBLCR	UART B Line control Register
UARTBLSR	UART B Line Status Register
UARTBSCR	UART B Scratch Pad Register

26.5.3 Baud Rate Configuration

To configure the baud rate for RX/TX operations, the **UARTBDLR** may be used.

To baud rate may be configured by the following formula:

Baud Rate = PCLK / (16 * **UARTBDLR**)

Note that UART peripherals sometimes need an accurate timing base in order to function properly. Because of that, it is suggested that the CLKREF or EXTCLK be used as the clock source for PCLK when using the UART.

The ROSCCLK may not provide enough accuracy for UART applications.

26.5.4 FIFO Reset

The RX and TX FIFOs may be reset independently by using the **UARTBFC** register.

To change any of the bits in the **UARTBFCR**, the **UARTBFCR.FIFOEN** must be set to 1b (**UARTBFCR** access).

To reset the RX FIFO, write the **UARTBFCR.RXFIFORST** bit to 1b. This will clear all bytes in the FIFO and reset the pointer to the top of the FIFO for the RX state machine.

To reset the TX FIFO, write the **UARTBFCR.TXFIFORST** bit to 1b. This will clear all bytes in the FIFO and reset the pointer to the top of the FIFO for the TX state machine.

When the **UARTBFCR** register is done being used to configure the FIFOs, set the **UARTBFCR.FIFOEN** to 1b (application mode).

26.5.5 UART Configuration

In UART mode, the UART may be have the parity, break control, stop bits and word length configured by the user.

The parity checking may be enabled or disabled. To enable parity checking, set the **UARTBLCR.PEN** to 1b. To disable, set this field to 0b. If enabled, the user may configure the type of parity checking as follows.

Table 26-2 UART A Parity Modes

UARTBLCR.PEN	UARTBLCR.PSEL	Parity	Description
0b	n/a	None	
1b	00b	Odd	The number of ones in the transmitted character plus the attached parity bit will be odd.
	01b	Even	The number of ones in the transmitted character plus the attached parity bit will be even.
	10b	Forced 1 stick	Force 1 for parity
	11b	Forced 0 stick	Force 0 for parity

To set the number of stop bits, the user may write the **UARTBLCR.SBS** field. To select 1 stop bits, set **UARTBLCR.SBS** to 0b. To select 2 stop bits, set **UARTBLCR.SBS** to 1b. If the **UARTBLCR.WLS** is set to 00b (5-bit character length) and **UARTBLCR.SBS** is set to 1b, then there will be 1.5 stop bits.

To configure break control, the user can set the **UARTBLCR.BCON** field. To disable break transmission, set **UARTBLCR.BCON** to 0b. To force the TX signal to logic 0, set the **UARTBLCR.BCON** to 1b.

To set the word length for the UART, the user can set the **UARTBLCR.WLS** field as shown below.

Table 26-3 UART B Word Length

UARTBLCR.WLS	UART word length
00b	5-bit
01b	6-bit
10b	7-bit
11b	8-bit

26.5.6 UART Scratch Pad

There is an 8-bit general purpose register that may be used in the USART.

The user may use the **USBSCR** register for an 8-bit scratch pad.

26.5.7 UART Interrupts

The status of interrupts may be read at any time by the **UARTBIIR** register. If the **UARTBIIR.INTSTATUS** field is set to 0b, at least one interrupt is pending. The interrupt type can be read by the **UARTBIIR.INTID** field as shown below.

Table 26-4 UART B Interrupt ID

UARTBIIR.INTID	Interrupt Type
000b	Reserved
001b	TX Holding Register Empty
010b	Receive Data Available
011b	Receive Line Status
100b	Reserved
101b	Reserved
110b	Receive FIFO character time-out
111b	Reserved

26.5.8 Transmit Operation

Transmission may be initiated by writing **UARTBTHR** with the data desired to be sent.

If the TX FIFO is enabled (**UARTBFCR.FIFOEN** = 1b), the data will be written into the TX FIFO. The contents of the TX FIFO will be transferred to the TX shift register one character at a time, until the TX FIFO is empty. The depth of the TX FIFO is 16-bytes.

If the TX FIFO is not enabled (**UARTBFCR.FIFOEN** = 0b), the data will be transferred to the TX shift register.

The bits to be transmitted are then shifted out of the TX shift register in the order of start bit, data bits (LSB first), parity bit, stop bit, using the configuration from the **UARTBLCR** register.

The baud rate used will be set by the **UARTBDLR** register.

26.5.9 Transmit Interrupts

If the TX FIFO is enabled (**UARTBFCR.FIFOEN** = 1b), the UART may be configured to generate interrupts after characters have been successfully transmitted. The UART may be configured to generate an interrupt after 1, 4, 8 or 14 characters have been transmitted. To set the interrupt threshold, the user should set the **UARTBEFR.ENMODE** = 1b and set the **UARTBFCR.TXTL** to the interrupt FIFO depth that is desired.

To enable transmit interrupts, set the **UARTBIER.THREIE** (TX holding register empty interrupt enable). When the transmit holding register has been emptied and the TX FIFO is empty (if **UARTBFCR.FIFOEN** = 1b), the **UARTBLSR.THRE** will be set to 1b. This field will be cleared the next time the **UARTBTHR** register is written with data to transmit. During this event, the **UARTBIIR.INTSTATUS** will be set to 1b (interrupt pending) and the **UARTBIIR.INTID** will be set to 001b (TX holding register empty).

If the **UARTBIER.THREIE** is set to 1b and any of the flags in **UARTBLSR** are set, the IRQ_USARTB signal to the NVIC will be asserted.

26.5.10 Receive Operation

Data is sampled into the RX shift register at a sampling rate of PCLK / 16. A filter is used to remove spurious inputs that last for less than two periods of the sampling rate.

If the RX FIFO is enabled (**UARTBFCR.FIFOEN** = 1b), the data from the shift register will be loaded into the RX FIFO. The RX FIFO will load each character from the FIFO into the RX Buffer Register, until the FIFO is empty. The depth of the RX FIFO is 16-bytes.

If the RX FIFO is not enabled (**UARTBFCR.FIFOEN** = 0b), the received data will be loaded directly into the RX Buffer Register.

When the complete character has been loaded into the RX Buffer Register, it may be red using the **UARTBRBR** register. The receiver checks the parity and stop bits as specified in the **UARTBLCR** register.

26.5.11 Receive Interrupts

When characters are received from the UART, the user may configure the peripheral to indicate RX FIFO character time-out, incorrect parity, a missing stop bit (frame error) or other line status registers. To enable these interrupts, set the **UARTBIER.RLSIE** to 1b. If these conditions occur, the **UARTBIIR.INTSTATUS** will be set to 1b (interrupt pending) and the **UARTBIIR.INTID** will be set accordingly.

The UART may be configured to generate interrupts after a character has been successfully received.

To enable receive interrupts, set the **UARTBIER.RBRIE** (RX buffer register interrupt enable). When the **UARTBRBR** (receive buffer register) has been filled, the **UARTBLSR.RDR** will be set to 1b. This field will be cleared the next time the **UARTBRBR** register is read.

If the RX FIFO is enabled (**UARTBFCR.FIFOEN** = 1b), then the FIFO will generate an interrupt when 1, 4, 8 or 14 characters have been loaded into it. To set the interrupt threshold, the user should set **UARTBFCR.RXTL** to the interrupt FIFO depth that is desired. When the FIFO

contains the configured number of characters, the **UARTBIIR.INTSTATUS** will be set to a 1b (interrupt pending) and the **UARTBIIR.INTID** will be set accordingly.

If the **UARTBIER.RBRIE** is set to 1b and the **UARTBLSR.RDR** interrupt flag is set, the IRQ_USARTB signal to the NVIC will be asserted.

26.6 UART Register Summary

Table 26-5 USART B Register Summary (UART Mode)

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
UARTBRBR	4003 0000h	UART B Receive Buffer Register	RO	--
UARTBTHR	4003 0004h	UART B Transmit Holding Register	WO	--
UARTBDLR	4003 0008h	UART B Divisor Latch Register	RW	0000 0001h
UARTBIER	4003 000Ch	UART B Interrupt Enable Register	RW	0000 0000h
UARTBIIR	4003 0010h	UART B Interrupt Identification Register	RO	0000 0001h
UARTBFCR	4003 0014h	UART B FIFO Control Register	RW	0000 0000h
UARTBLCR	4003 0018h	UART B Line control Register	RW	0000 0000h
UARTBLSR	4003 0020h	UART B Line Status Register	RO	0000 0060h
UARTBSCR	4003 0028h	UART B Scratch Pad Register	RW	--
UARTBEFR	4002 002Ch	UART B Enhanced Mode Register	RW	0000 000h

26.7 UART Register Detail

26.7.1 UARTBRBR

Register 26-1 UARTBRBR (UART B Receive Buffer Register, 4003 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**RBR**	RO	0	Contains the oldest received character in the UART RX FIFO.

26.7.2 UARBTHR

Register 26-2 UARTBTHR (UART B Transmit Holding Register, 4003 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**THR**	WO	-	Writing to the **UARTBTHR** causes the data to be stored in the UART transmit FIFO. The character will be sent when it reaches the bottom of the FIFO and the transmitter is available.

26.7.3 UARTBDLR

Register 26-3 UARTBDLR (UART B Divisor Latch Register, 4003 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**DLR**	RW	0000 0001b	Sets the baud rate for the module. Baud rate = PCLK / (16 * **UARTBDLR**)

26.7.4 UARTBIER

Register 26-4 UARTBIER (UART B Interrupt Enable Register, 4003 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:3	**Reserved**	RO	0	Reserved
2	**RLSIE**	RW	0	RX Line Status Interrupt Enable. Enables the UART RX line status interrupts. The status of this interrupts can be read from **UARTBLSR[4:1]**. 0b: Disable the RX line status interrupts 1b: Enable the RX line status interrupts
1	**THRIE**	RW	0	TX Holding Register Empty Interrupt Enable. Enables the THRE interrupt for the UART. The status of this interrupt can be read from **UARTBLSR.THRE**. 0b: Disable the THRE interrupts 1b: Enable the THRE interrupts
0	**RBRIE**	RW	0	RX Buffer Register Interrupt Enable. Enables the Receive Data Available interrupt for the UART. It also controls the character receive time-out interrupt. 0b: Disable the RBR interrupts 1b: Enable the RBR interrupts

26.7.5 UARTBIIR

Register 26-5 UARTBIIR (UART B Interrupt Identification Register, 4003 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3:1	**INTID**	RO	000b	Interrupt identification: 000b: Reserved 001b: TX Holding Register Empty 010b: Receive Data Available 011b: Receive Line Status 100b: Reserved 101b: Reserved 110b: Receive FIFO Character Time-out 111b: Reserved
0	**INTSTATUS**	RO	1b	Interrupt status. Note that this bit is active low. The pending interrupt can be determined through the **UARTBIIR.INTID** field. 0b: At least one interrupt is pending 1b: No interrupt is pending

Note that this register is cleared on read.

26.7.6 UARTBFCR

Register 26-6 UARTBFCR (UART B FIFO Control Register, 4003 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:6	**RXTL**	RW	0	RX Trigger Level. These two bits determine how many receive UART FIFO characters must be written before an interrupt is activated. 00b: 1 character 01b: 4 characters 10b: 8 characters 11b: 14 characters
5:4	**TXTL**	RW	0	TX Trigger Level. These two bits determine how may transmit UART FIFO characters must be written before an interrupt is activated. 00b: 1 character 01b: 4 characters 10b: 8 characters 11b: 14 characters
3	**Reserved**	RO	0	Reserved
2	**TXFIFORST**	RW	0	TX FIFO Reset. 0b: No effect 1b: Clears all bytes in the UART TX FIFO. This bit is self-clearing
1	**RXFIFORST**	RW	0	RX FIFO Reset. 0b: No effect 1b: Clears all bytes in the UART RX FIFO. This bit is self-clearing
0	**FIFOEN**	RW	0	FIFO Enable: 0b: UART FIFOs are disabled. Reset UARTAFCR settings to their default values. 1b: UART RX and TX FIFOs are enabled. **UARTBFCR[7:1]** are accessible when this bit is set. Any transition on this bit will automatically clear the UART RX AND TX FIFOs.

26.7.7 UARTBLCR

Register 26-7 UARTBLCR (UART B Line Control Register, 4003 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:7	**Reserved**	RO	0	Reserved
6	**BCON**	RW	0	Break Control: 0b: Disable break transmission 1b: Enable break transmission. Output pin UARTBTXD is forced to logic 0 when this bit is set.
5:4	**PSEL**	RW	0	Parity Select: 00b: Odd parity. The number of 1s in the transmitted character and the attached parity will be odd. 01b: Even parity. The number of 1s in the transmitted character and the attached parity will be even. 10b: Forced 1 stick parity. 11b: Forced 0 stick parity.
3	**PEN**	RW	0	Parity Enable: 0b: Disable parity generation and checking 1b: Enable parity generation and checking
2	**SBS**	RW	0	Stop Bit Select: 0b: 1 stop bit 1b: 2 stop bits, 1.5 stop bits if (**UARTBLCR.WLS** = 00b)
1:0	**WLS**	RW	0	Word Length Select: 00b: 5-bit character length 01b: 6-bit character length 10b: 7-bit character length 11b: 8-bit character length

26.7.8 UARTBLSR

Register 26-8 UARTBLSR (UART B Line Status Register, 4003 0020h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**RXFE**	RO	0	Error in RX FIFO. This bit is set when a character with a RX error such as framing, parity or break is loaded into **UARTBRBR**. This bit is cleared when the **UARTBLSR** register is read, and there are no subsequent errors in the UART FIFO.
6	**TEMT**	RO	0	Transmitter Empty. This bit is set when both **UARTBTHR** and **UARTBTSR** are empty; this bit is cleared when THR contains valid data. 0b: THR contains valid data 1b: THR is empty
5	**THRE**	RO	0	Transmitter Holding Register Empty. This bit is set immediately upon detection of an empty UART THR and is cleared on a THR write. 0b: THR contains valid data 1b: THR is empty
4	**BI**	RO	0	Break Interrupt. When UARTBRX is held in the spacing state (all zeroes) for one full character transmission (start, data, parity, stop), a break interrupt occurs. Once the break condition has been detected, the receiver goes idle until UARTBRX goes to the marking statue (all ones). A read of the **UARTBLSR** register clears this status bit. The time of break detection is dependents on **UARTBFCR.FIFOEN**. The break interrupt is associated with the character at the top of the UART RBR FIFO. 0b: Break interrupt status is inactive 1b: Break interrupt status is active
3	**FE**	RO	0	Framing Error. When the stop bit of a received character is logic 0, a framing error occurs. Reading the **UARTBLSR** register will clear this bit. The time of the framing error is dependent upon **UARTBFCR[3]**. Upon detection of a framing error, the RX will attempt to re-synchronize the data and assume that the bad stop bit is actually an early start bit. However, it cannot be assumed that the next received byte will be correct even if there is no framing error. A framing error is associated with the character at the top of the UART RBR FIFO. 0b: Framing error status is inactive 1b: Framing error status is active
2	**PE**	RO	0	Parity Error. When the parity bit of a received character is in the wrong state, a parity error occurs. Reading the **UARTBLSR** register will clear this bit. The time of the parity error detection is dependent on **UARTBFCR.FIFOEN**.

				A parity error is associated with the character at the top of the UART RBR FIFO. 0b: Parity error status is inactive 1b: Parity error status is active
1	**OE**	RO	0	Overrun Error. The overrun error condition is set as soon as it occurs. Reading the **UARTBLSR** register will clear this bit. This bit is set when the UART RSR has a new character assembled and the UART RBR FIFO is full. In this case, the UART RBR FIFO will not be overwritten and the character in the UART RSR will be lost. 0b: Overrun error status is inactive 1b: Overrun error status is active
0	**RDR**	RO	0	Receiver Data Ready. This bit is set when the RBR holds an unread character and is cleared when the UART RBR FIFO is empty. 0b: RBR is empty 1b: RBR contains valid data

26.7.9 UARTBSCR

Register 26-9 UARTBSCR (UART B Scratch Pad Register, 4003 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**PAD**	RW	-	A read-able, write-able byte.

26.7.10 UARTBEFR

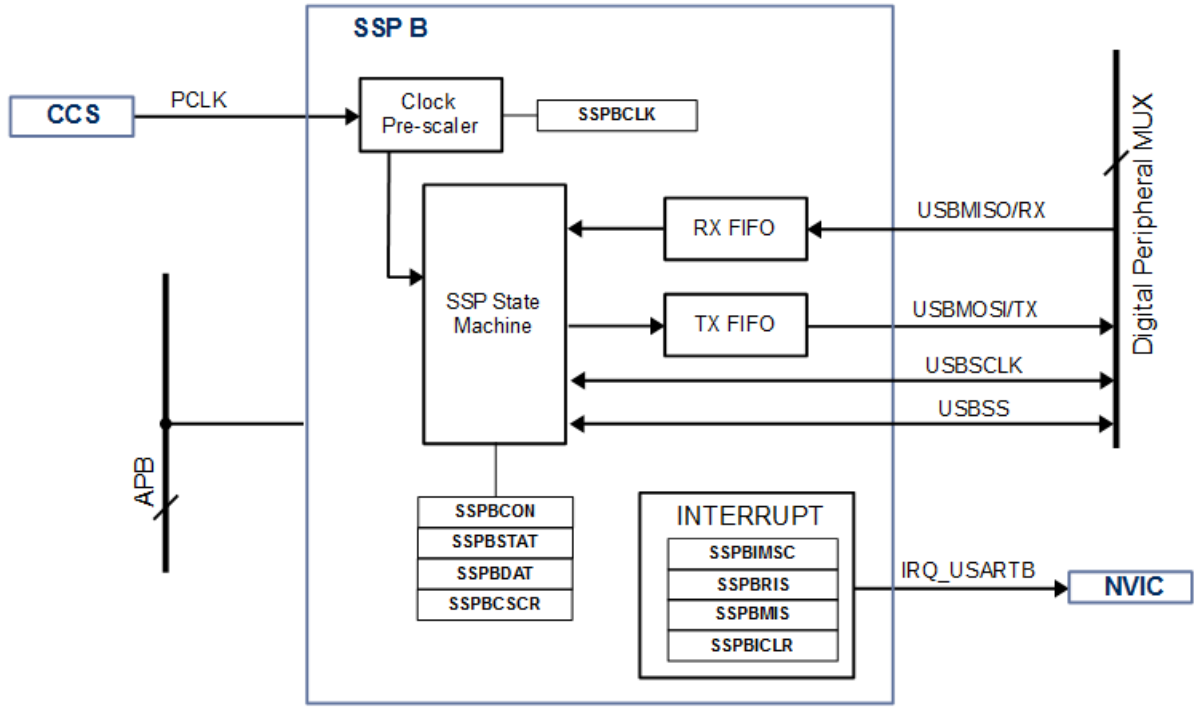
Register 26-10 UARTBEFR (UART B Enhanced Feature Register, 4003 002Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**ENMODE**	RW	0	Enhanced Mode 0b – disabled 1b – enabled
3:0	**Reserved**	RO	0	Reserved

26.8 SSP Functional Description

26.8.1 SSP Mode System Block Diagram

Figure 26-3 SSP B System Block Diagram



26.8.2 Mode Configuration

The SSP mode of the USART must first be selected by writing **CCSCTL.USBMODE** to 0b. This will set the USART into SSP mode and allow all configuration registers to be used in this mode.

The configuration and status registers that are available when **CCSCTL.USBMODE** is 0b are below.

Table 26-6 USART B UART Mode Registers

REGISTER	DESCRIPTION
SSPBCON	SSP B Control Register
SSPBSTAT	SSP B Status Register
SSPBDAT	SSP B Data Register
SSPBCLK	SSP B Clock Control Register
SSPBIMSC	SSP B Interrupt Mask Set and Clear Register
SSPBRIS	SSP B Raw Interrupt Status Register

SSPBMIS	SSP B Masked Interrupt Status Register
SSPBICLR	SSP B Interrupt Clear Register
SSPBSSCR	SSP B Slave Select Configuration Register

26.8.3 SSP Overview

When in SSP mode, the USART may be a master or slave interface that enables synchronous serial communication with slave or master peripherals having one of the following:

- Motorola-style SPI
- TI-style Synchronous Serial Interface
- National Semiconductor-style Microwire

In both master and slave modes, the SSP performs parallel to serial conversion on data into 32-bit wide, 8-location deep RX and TX FIFOs.

The SSP may be configured to generate interrupts for servicing the TX and RX FIFOs and error conditions such as FIFO overrun and timeout.

26.8.4 Clock Configuration

When configured for master mode, the SSP peripheral may configure the clock pre-scaler to generate the desired SCLK output clock frequency. The input clock on the SSP peripheral is the PCLK (peripheral clock) system clock.

The SCLK is generated from the PCLK input and the **SSPBCLK.M** and **SSPBCLK.N** parameters. The SCLK clock frequency is generated according to the following formula:

F_{SSPCLK} = PCLK / ((**SSPBCLK.M** + 1) * **SSPBCLK.N**)

In this formula, **SSPBCLK.N** must be an even value from 2 to 254.

26.8.5 SSP Clock Constraints

The SSP clock configuration must be configured according to the guidelines below, in order for the peripheral to operate correctly.

Here are the clock constraints for the input clock SSPCLK (after the pre-scaler) and output SSP clock, USBCLK:

- When in master mode, the minimum frequency of the input SSPCLK must be at least 2X the output SSP clock (USBSCLK)
- When in slave mode, the minimum frequency of the input SSPCLK must be at least 12X the input SSP clock (USBSCLK)

The maximum frequency of the SSPCLK should be:

- When in master mode, $F_{SSPCLK}(max) <= 254 * 128 * F_{USBSCLK}$ (SSP clock output)
- When in slave mode, $F_{SSPCLK}(max) <= 254 * 128 * F_{USBSCLK}$ (SSP clock input)

26.8.6 SSP Configuration

The SSP may be configured to support various features of master and slave based serial interfaces.

To enable the SSP controller, set **SSPBCON.SSPEN** to 1b. When disabled, the SSP may still have registers configured for the configuration mode and interrupts, but will not operate on serial data until enabled. Once the SSP has been configured, set this bit to 1b to begin processing.

To configure the SSP mode, set **SSPBCON.FRF** (frame format) to the desired value as shown below.

Table 26-7 SSP B Frame Format

SSPBCON.FRF	Frame Format
00b	SPI (Motorola)
01b	Synchronous Serial Format (TI)
10b	Microwire (National Semiconductor)
11b	Reserved

To select master mode, set **SSPBCON.MS** to 0b. In this mode, the controller is the bus master. It will drive the SCLK, MOSI and SS signals and will receive data on the MISO line.

To select slave mode, set **SSPBCON.MS** to 1b. In this mode, the controller is the bus slave and will drive the MISO line and receive input from SCLK, MOSI and SS.

To change the serial to parallel endian order, the user may use the **SSPBCON.LSBFIRST** field. To configure the SSP for LSB first, set **SSPBCON.LSBFIRST** to 1b. To configure the SSP for MSB first, set **SSPBCON.LSBFIRST** to a 0b.

To enable loop-back mode, set the **SSPBCON.LBM** to 1b. In this mode, serial input is taken from the serial output instead of from the digital peripheral MUX.

While in slave mode, the behavior of the MISO output can be controlled. When **SSPBCON.SOD** is set to 0b, the SSP can drive the MISO in slave mode. When this bit is 1b, the SSP will not drive the MISO output in slave mode.

The clock phase of the SSP may also be configured. To configure the SSP for capturing data on the first clock edge transition, set **SSPBCON.CPH** to 0b. To configure the SSP for capturing data on the second clock edge transition, set **SSPBCON.CPH** to 1b.

When the **SSPBCON.FRF** is 00b (SPI), the clock output polarity may be configured. To configure the SSP clock to be active high, set **SSPBCON.CPO** to 0b. To configure the SSP clock to be active low, set **SSPBCON.CPO** to 1b.

The SSP may also configure the size of the data word size transferred in each frame. To configure the data size, see the table below.

Table 26-8 SSP B Data Size Select

SSPBCON.DSS	Data Size
0 0000b	Reserved
0 0001b	Reserved
0 0010b	Reserved
0 0011b	4-bit
0 0100b	5-bit
0 0101b	6-bit
0 0110b	7-bit
0 0111b	8-bit
0 1000b	9-bit
0 1001b	10-bit
0 1010b	11-bit
0 1011b	12-bit
0 1100b	13-bit
0 1101b	14-bit
0 1110b	15-bit
0 1111b	16-bit
…	…
1 1111b	32-bit

26.8.7 SSP Slave Select Configuration

The SSP allows the behavior of the SS (slave select) signal to be configured by using the **SSPBSSCR** register as follows when configured as a bus master.

The **SSPBSSCR.SELCS** field must always be set to a 00b for proper operation.

To configure the SSP to automatically control the behavior of the SS signal, set **SSPBSSCR.SWSEL** to 0b. If the user wants to control the SS signal by software, set **SSPBSSCR.SWSEL** to 1b.

To configure the SS signal behavior after a frame transfer, use the **SSPBSSCR.SPHDONTCARE** field. To configure the SSP to not pull the SS signal high after a frame transfer, set **SSPBSSCR.SPHDONTCARE** to 0b. To configure the SSP to pull the SS signal high after a frame transfer, set the **SSPBSSCR.SPHDONTCARE** to 1b.

26.8.8 SSP Interrupts

The SSP may configure interrupts for the following conditions.

When the SSP detects that the TX FIFO is half-full or less, the **SSPBRIS.TXRIS** and **SSPBMIS.TXMIS** are set to 1b. During this condition, if the **SSPBIMSC.TXIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTB signal to the NVIC.

When the SSP detects that the RX FIFO is half-full or more, the **SSPBRIS.RXRIS** and **SSPBMIS.RXMIS** are set to 1b. During this condition, if the **SSPBIMSC.RXIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTB signal to the NVIC.

The SSP contains a time for reading the RX FIFO. If the RX FIFO is not empty, and 32 F_{SSPCLK} periods have gone by, the **SSPBRIS.RTIM** and **SSPBMIS.RTMIS** bits are set to 1b. During this condition, if the **SSPBIMSC.RTIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTB signal to the NVIC. This condition may be cleared by writing **SSPBICLR.RTIC** to 1b.

If the SSP detects the RX FIFO is full, when a character is attempted to be inserted into it, is sets the **SSPBRIS.ROIM** and **SSPBMIS.RORIS** to 1b. During this condition, if the **SSPBIMSC.ROIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTB signal to the NVIC. This condition may be cleared by writing **SSPBICLR.ROIC** to 1b.

26.8.9 Master Mode Operation

When configured as a bus master (**SSPBCON.MS** = 0b), the SSP will insert a character into the TX FIFO when it is written into the **SSPBDAT** register. The SSP will read a character from the TX FIFO and performs a parallel to serial conversion on it. Then the serial data stream and frame control signal is synchronized to the clock and are output through the USBMOSI/TX pin to the attached slaves. While the data is being transmit to the slaves, the USBSCLK and USBSS behave as configured in the **SSPBCON** and **SSPBCSCR** registers.

The master receive logic performs serial to parallel conversion on the incoming synchronous USBMISO/RX data stream, and stores the character in the RX FIFO. The master may read first entry in the RX FIFO by reading the **SSPBDAT** register.

26.8.10 Slave Mode Operation

When configured as a bus slave (**SSPBCON.MS** = 1b), the SCLK is provided by the attached master. The user can write the next character to send into **SSPBDAT** and it will get inserted into the TX FIFO. The slave transmit logic reads a value from the TX FIFO, performs parallel to serial conversion and outputs the serial data stream on the USBMISO/RX pin to the attached master. The slave receive logic performs serial to parallel conversion on the incoming USBMOSI/TX data stream, extracting and storing values into the RX FIFO. The first character in the RX FIFO may be read by reading the **SSPBDAT** register.

26.8.11 SSP Status

The SSP module maintains status information on the operation of the module. The table below shows the available SSP status conditions.

Table 26-9 SSP B Status

SSPBSTAT field	Description
BSY	If set, the SSP controller is currently sending/receiving a frame and/or the TX FIFO is not empty.
RFF	If set, the RX FIFO is full.
RNE	If set, the RX FIFO is not empty.
TNF	If set, the TX FIFO is not full.
TFE	If set, the TX FIFO is empty

26.9 SSP Register Summary

Table 26-10 USART B Register Summary (SSP Mode)

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
SSPBCON	4003 0000h	SSP B Control Register	RW	0000 0000h
SSPBSTAT	4003 0004h	SSP B Status Register	RO	0000 0003h
SSPBDAT	4003 0008h	SSP B Data Register	RW	0000 0000h
SSPBCLK	4003 000Ch	SSP B Clock Control Register	RW	0000 0000h
SSPBIMSC	4003 0010h	SSP B Interrupt Mask Set and Clear Register	RW	0000 0000h
SSPBRIS	4003 0014h	SSP B Raw Interrupt Status Register	RO	0000 0008h
SSPBMIS	4003 0018h	SSP B Masked Interrupt Status Register	RO	0000 0008h
SSPBICLR	4003 001Ch	SSP B Interrupt Clear Register	RW	0000 0000h
SSPBSSCR	4003 0028h	SSP B Slave Select Configuration Register	RW	0000 0000h

26.10 SSP Register Detail

26.10.1 SSPBCON

Register 26-11 SSPBCON (SSP B Control Register, 4003 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:14	**Reserved**	RO	0	Reserved
13	**LSBFIRST**	RW	0	Endian Order: 0b: MSB transmit / receive first 1b: LSB transmit / receive first
12	**LBM**	RW	0	Loopback Mode: 0b: Normal operation 1b: Loopback mode. Serial Input is taken from the serial output (MOSI or MISO) rather than the serial in (MISO or MOSI respectively)
11	**SSPEN**	RW	0	SSP Enable: 0b: SSP Controller is disabled 1b: SSP Controller is enabled. It will interact with other devices on the serial bus. Software should write the appropriate control information to the other SPI/SSP registers and interrupt controller registers, before setting this bit.
10	**MS**	RW	0	Master/Slave Mode: 0b: The SSP controller acts as a bus master, driving the USBSCLK, USBMOSI and USSS signals 1b: The SSP controller acts as a bus slave, driving the USBMISO and receiving USBSCLK, USBMOSI and USBSS.
9	**SOD**	RW	0	Slave Output Disable: 0b: The SSP can drive the USBMISO output in slave mode 1b: The SSP must not drive the USBMISO output in slave mode
8	**CPH**	RW	0	Clock Out Phase. This bit is only used when **SSPBCON.FRF** = 0b (SPI). 0b: The SSP controller captures serial data on the first edge transition of the frame. 1b: The SSP controller captures serial data on the second edge transition of the frame.
7	**CPO**	RW	0	Clock Out Polarity (this bit is only used in SPI mode when **SSPBCON.FRF** = 00b) 0b: The clock is active high. 1b: The clock is active low.
6:5	**FRF**	RW	0	Frame Format: 00b: SPI 01b: TI 10b: Microwire 11b: Reserved
4:0	**DSS**	RW	0	Data Size Select: 0 0000b: Reserved 0 0001b: Reserved 0 0010b: Reserved 0 0011b: 4-bit transfer 0 0100b: 5-bit transfer

				0 0101b: 6-bit transfer 0 0110b: 7-bit transfer 0 0111b: 8-bit transfer 0 1000b: 9-bit transfer 0 1001b: 10-bit transfer 0 1010b: 11-bit transfer 0 1011b: 12-bit transfer 0 1100b: 13-bit transfer 0 1101b: 14-bit transfer 0 1110b: 15-bit transfer 0 1111b: 16-bit transfer … 1 1111b: 32-bit transfer

26.10.2 SSPBSTAT

Register 26-12 SSPBSTAT (SSP B Status Register, 4003 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**BSY**	RO	0	Busy Bit: 0b: SPI controller is idle 1b: SPI controller is sending/receiving a frame and/or the TX FIFO is not empty
3	**RFF**	RO	0	Receive FIFO Full: 0b: RX FIFO not full 1b: RX FIFO full
2	**RNE**	RO	0	Receive FIFO not empty: 0b: RX FIFO empty 1b: RX FIFO not empty
1	**TNF**	RO	0	Transmit FIFO not full: 0b: TX FIFO full 1b: TX FIFO not full
0	**TFE**	RO	0	Transmit FIFO empty: 0b: TX FIFO not empty 1b: TX FIFO empty

26.10.3 SSPBDAT

Register 26-13 SSPBDAT (SSP B Data Register, 4003 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**DATA**	RW	0	Software can writing data to be sent in a future frame to this register when the **SSPBSTAT.TNF** bit is set to 1b, indicating that the TX FIFO was previously empty and the SPI controller is not busy on the bus, transmission of the data will begin immediately. Otherwise, the data written to this register will be sent as soon as all previous data has been sent (and received). If the data length is less than 32-bits, the data must be right-justified in this register. Software may read data from this register when the **SSPBSTAT.RNE** bit is set to a 1b, indicating that the RX FIFO is not empty. When software reads this register, the SPI controller returns the data from the least recent frame in the RX FIFO. If the data length is less than 32-bits, the data is right-justified in this field with the MSBs set to 0.

26.10.4 SSPBCLK

Register 26-14 SSPBCLK (SSP B Clock Register, 4003 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:8	**M**	RW	0	These fields may set the SPI master clock rate by the formula: F_{SSPCLK} = PCLK / ((**SSPBCLK.M** + 1) * **SSPBCLK.N**) N must be an even value from 2 to 254.
7:0	**N**	RW	0	

26.10.5 SSPBIMSC

Register 26-15 SSPBIMSC (SSP B Interrupt Mask Set and Clear Enable Register, 4003 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXIM**	RW	0	Transmit FIFO Interrupt Mask: 0b: TX FIFO half-full or less condition interrupt disabled 1b: TX FIFO half-full or less condition interrupt enabled
2	**RXIM**	RW	0	Receive FIFO Interrupt Mask: 0b: RX FIFO half-full or more condition interrupt disabled 1b: RX FIFO half-full or more condition interrupt enabled
1	**RTIM**	RW	0	Receive Timeout Interrupt Mask: 0b: RX FIFO not empty and no read prior to timeout period interrupt disabled 1b: RX FIFO not empty and no read prior to timeout period interrupt enabled
0	**ROIM**	RW	0	Receive Overrun Interrupt Mask: 0b: RX FIFO written to while full condition interrupt disabled 1b: RX FIFO written to while full condition interrupt enabled

26.10.6 SSPBRIS

Register 26-16 SSPBRIS (SSP B Raw Interrupt Status Register, 4003 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXRIS**	RO	1	TX FIFO Raw Interrupt Status: 0b: TX FIFO not at least half-full 1b: TX FIFO at least half-full
2	**RXRIS**	RO	0	RX FIFO Raw Interrupt Status: 0b: RX FIFO not at least half-full 1b: RX FIFO at least half-full
1	**RTRIS**	RO	0	RX FIFO Timeout Interrupt Status: 1b: RX FIFO is not empty and has not been read for the timeout period. The timeout period is 32-bit times at F_{SSPCLK}
0	**RORIS**	RO	0	RX FIFO Overwrite Interrupt Status: 1b: RX FIFO was full when another frame was completely received. The preceding data frame is overrun by the new frame when this occurs.

26.10.7 SSPBMIS

Register 26-17 SSPBMIS (SSP B Masked Interrupt Status Register, 4003 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXMIS**	RO	1	TX FIFO Masked Interrupt Status: 0b: TX FIFO not at least half-full 1b: TX FIFO at least half-full
2	**RXMIS**	RO	0	RX FIFO Masked Interrupt Status: 0b: RX FIFO not at least half-full 1b: RX FIFO at least half-full
1	**RTMIS**	RO	0	RX FIFO Timeout Interrupt Status: 1b: RX FIFO is not empty and has not been read for the timeout period. The timeout period is 32-bit times at F_{SSPCLK}
0	**ROMIS**	RO	0	RX FIFO Overrun Interrupt Status: 1b: RX FIFO was full when another frame was completely received. The preceding data frame is overrun by the new frame when this occurs.

26.10.8 SSPBICLR

Register 26-18 SSPBICLR (SSP B Interrupt Clear Register, 4003 001Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:2	**Reserved**	RO	0	Reserved
1	**RTIC**	RW	0	RX FIFO Timeout Interrupt Clear: 0b: No effect 1b: Clears the **SSPBxIS.RTRIS** interrupt flag This bit is self-clearing.
0	**ROIC**	RW	0	RX FIFO Overwrite Interrupt Clear: 0b: No effect 1b: Clears the **SSPBxIS.RORIS** interrupt flag This bit is self-clearing.

26.10.9 SSPBSSCR

Register 26-19 SSPBSSCR (SSP B Slave Select Configuration Register, 4003 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**SPHDONTCARE**	RW	0	Slave Select Pull High: 0b: USBSS cannot pull high after frame transfer 1b: USBSS must pull high after frame transfer
3	**SWSS**	RW	0	Slave Select State: 0b: Set USBSS to low 1b: Set USBSS to high
2	**SWSEL**	RW	0	Slave Select Software Control: 0b: USBSS is automatically controlled by the SPI module 1b: USBSS is software controlled by **SSPBSSCR.SWSS**
1:0	**SELSS**	RW	0	Slave Select Signal Control: 00b: USBSS is enabled 01b: Reserved 10b: Reserved 11b: Reserved

27 USART C

27.1 Overview

The PAC55XX family contains support for four Universal Synchronous Asynchronous Receive Transmit (USART) peripherals.

A USART is a serial communications engine that may be configured for UART or SSP through a mode selection register.

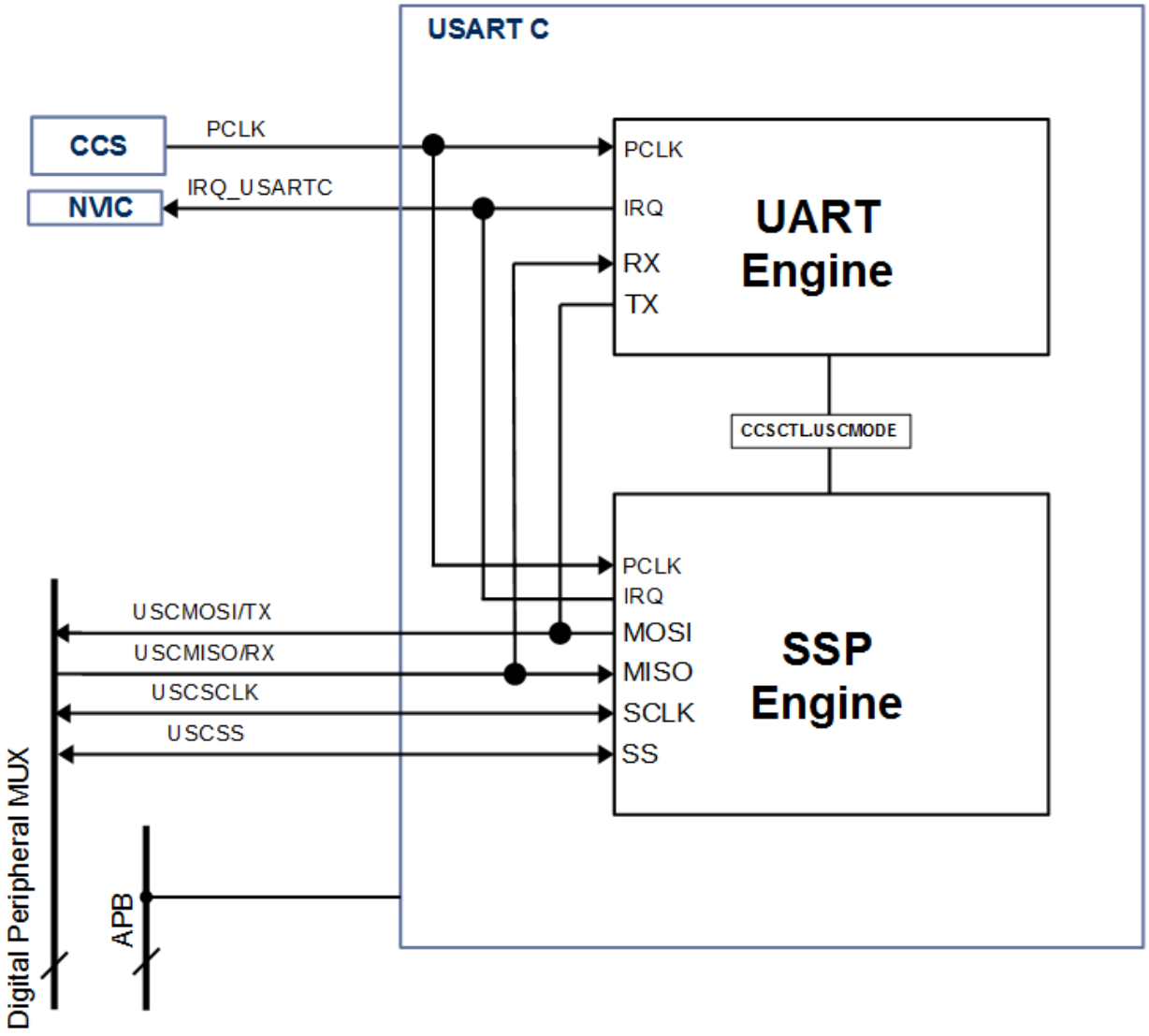
The USART is an APB client and is clocked by the Peripheral Clock (PCLK) system clock.

27.2 Features

- UART Mode:
  - 5-bit to 8-bit data interface
  - Optional parity
  - 1, 1.5 or 2 stop bits
  - 16-bit programmable baud-rate generator
  - 8-bit scratch pad
  - Independent RX and TX FIFOs
  - Configurable RX and TX interrupts
- SSP Mode:
  - Motorola SPI, TI Synchronous serial or National Semiconductor Microwire support
  - Master and Slave mode support
  - Independent RX and TX FIFOs
  - Configurable clock pre-scaler
  - Programmable Interrupts

27.3 System Block Diagram

Figure 27-1 USART C System Block Diagram



27.4 Functional Description

The USART may be configured for either UART or SSP mode via the **CCSCTL.USBMODE** register.

In UART mode, the peripheral can receive or transmit UART serial data over the USCMOSI/TX (UART TX) or USCMISO/RX serial lines. In this mode, the USART signals for USCSCLK and USBSS are unused. The UART may assert the IRQ_USARTC signal to the NVIC for interrupts and is clocked using the PCLK system clock and connected to the APB bus.

In UART mode, only the UART registers may be accessed, starting at register offset 0.

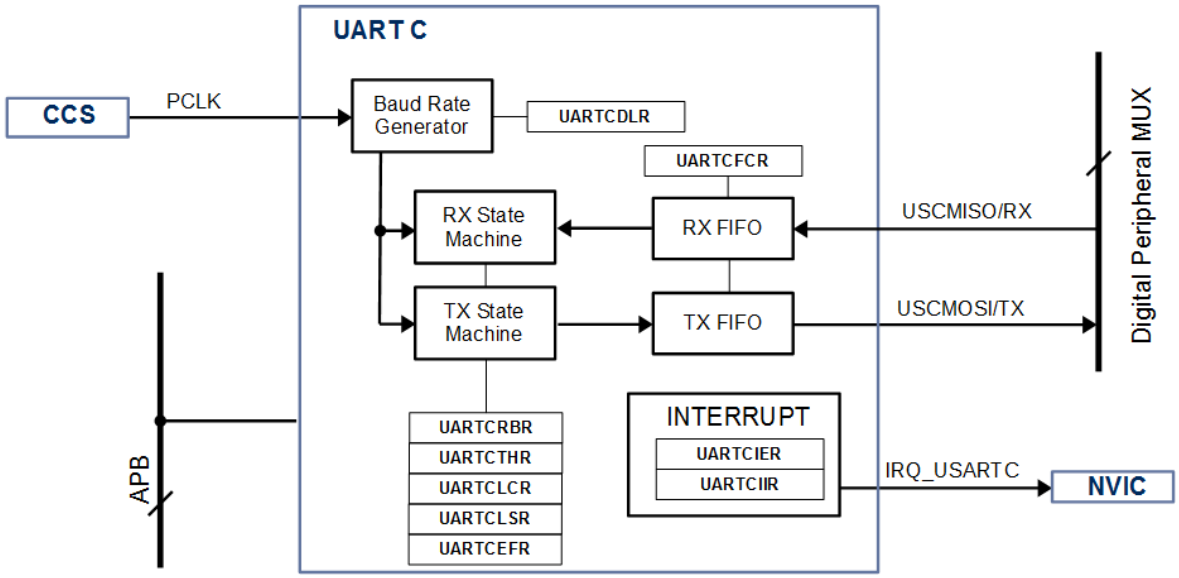
In SSP mode, the peripheral can receive or transmit serial data over the USCMOSI/TX (SPI MOSI) or USCMISO/RX (SPI MISO) serial lines. The USCSCLK and USCSS are used by the peripheral in either master or slave mode for the SSP clock and Slave Select signals. The SSP may assert the IRQ_USARTC signal to the NVIC for interrupts, and is clocked using the PCLK system clock and connected to the APB bus.

In SPI mode, only the SPI registers may be accessed, starting at register offset 0.

27.5 UART Mode Functional Description

27.5.1 UART C Mode System Block Diagram

Figure 27-2 UART C Mode System Block Diagram



27.5.2 Mode Configuration

The UART mode of the USART must first be selected by writing **CCSTL.USCMODE** to 1b. This will set the USART into UART mode and allow all configuration registers to be used in this mode.

The configuration and status registers that are available when **CCSCTL.USCMODE** is 1b are below.

Table 27-1 USART C UART Mode Registers

REGISTER	DESCRIPTION
UARTCRBR	UART C Receive Buffer Register
UARTCTHR	UART C Transmit Holding Register
UARTCDLR	UART C Divisor Latch Register
UARTCIER	UART C Interrupt Enable Register
UARTCIIR	UART C Interrupt Identification Register
UARTCFCR	UART C FIFO Control Register
UARTCLCR	UART C Line control Register
UARTCLSR	UART C Line Status Register
UARTCSCR	UART C Scratch Pad Register

27.5.3 Baud Rate Configuration

To configure the baud rate for RX/TX operations, the **UARTCDLR** may be used.

To baud rate may be configured by the following formula:

Baud Rate = PCLK / (16 * **UARTCDLR**)

Note that UART peripherals sometimes need an accurate timing base in order to function properly. Because of that, it is suggested that the CLKREF or EXTCLK be used as the clock source for PCLK when using the UART.

The ROSCCLK may not provide enough accuracy for UART applications.

27.5.4 FIFO Reset

The RX and TX FIFOs may be reset independently by using the **UARTCFC** register.

To change any of the bits in the **UARTCFCR**, the **UARTCFCR.FIFOEN** must be set to 1b (**UARTCFCR** access).

To reset the RX FIFO, write the **UARTCFCR.RXFIFORST** bit to 1b. This will clear all bytes in the FIFO and reset the pointer to the top of the FIFO for the RX state machine.

To reset the TX FIFO, write the **UARTCFCR.TXFIFORST** bit to 1b. This will clear all bytes in the FIFO and reset the pointer to the top of the FIFO for the TX state machine.

When the **UARTCFCR** register is done being used to configure the FIFOs, set the **UARTCFCR.FIFOEN** to 1b (application mode).

27.5.5 UART Configuration

In UART mode, the UART may be have the parity, break control, stop bits and word length configured by the user.

The parity checking may be enabled or disabled. To enable parity checking, set the **UARTCLCR.PEN** to 1b. To disable, set this field to 0b. If enabled, the user may configure the type of parity checking as follows.

Table 27-2 UART C Parity Modes

UARTCLCR.PEN	UARTCLCR.PSEL	Parity	Description
0b	n/a	None	
1b	00b	Odd	The number of ones in the transmitted character plus the attached parity bit will be odd.
	01b	Even	The number of ones in the transmitted character plus the attached parity bit will be even.
	10b	Forced 1 stick	Force 1 for parity
	11b	Forced 0 stick	Force 0 for parity

To set the number of stop bits, the user may write the **UARTCLCR.SBS** field. To select 1 stop bits, set **UARTCLCR.SBS** to 0b. To select 2 stop bits, set **UARTCLCR.SBS** to 1b. If the **UARTCLCR.WLS** is set to 00b (5-bit character length) and **UARTCLCR.SBS** is set to 1b, then there will be 1.5 stop bits.

To configure break control, the user can set the **UARTCLCR.BCON** field. To disable break transmission, set **UARTCLCR.BCON** to 0b. To force the TX signal to logic 0, set the **UARTCLCR.BCON** to 1b.

To set the word length for the UART, the user can set the **UARTCLCR.WLS** field as shown below.

Table 27-3 UART C Word Length

UARTCLCR.WLS	UART word length
00b	5-bit
01b	6-bit
10b	7-bit
11b	8-bit

27.5.6 UART Scratch Pad

There is an 8-bit general purpose register that may be used in the USART.

The user may use the **USCSCR** register for an 8-bit scratch pad.

27.5.7 UART Interrupts

The status of interrupts may be read at any time by the **UARTCIIR** register. If the **UARTCIIR.INTSTATUS** field is set to 0b, at least one interrupt is pending. The interrupt type can be read by the **UARTCIIR.INTID** field as shown below.

Table 27-4 UART C Interrupt ID

UARTCIIR.INTID	Interrupt Type
000b	Reserved
001b	TX Holding Register Empty
010b	Receive Data Available
011b	Receive Line Status
100b	Reserved
101b	Reserved
110b	Receive FIFO character time-out
111b	Reserved

27.5.8 Transmit Operation

Transmission may be initiated by writing **UARTCTHR** with the data desired to be sent.

If the TX FIFO is enabled (**UARTCFCR.FIFOEN** = 1b), the data will be written into the TX FIFO. The contents of the TX FIFO will be transferred to the TX shift register one character at a time, until the TX FIFO is empty. The depth of the TX FIFO is 16-bytes.

If the TX FIFO is not enabled (**UARTCFCR.FIFOEN** = 0b), the data will be transferred to the TX shift register.

The bits to be transmitted are then shifted out of the TX shift register in the order of start bit, data bits (LSB first), parity bit, stop bit, using the configuration from the **UARTCLCR** register.

The baud rate used will be set by the **UARTCDLR** register.

27.5.9 Transmit Interrupts

If the TX FIFO is enabled (**UARTCFCR.FIFOEN** = 1b), the UART may be configured to generate interrupts after characters have been successfully transmitted. The UART may be configured to generate an interrupt after 1, 4, 8 or 14 characters have been transmitted. To set the interrupt threshold, the user should set the **UARTCEFR.ENMODE** = 1b and set the **UARTCFCR.TXTL** to the interrupt FIFO depth that is desired.

To enable transmit interrupts, set the **UARTCIER.THREIE** (TX holding register empty interrupt enable). When the transmit holding register has been emptied and the TX FIFO is empty (if **UARTCFCR.FIFOEN** = 1b), the **UARTCLSR.THRE** will be set to 1b. This field will be cleared the next time the **UARTCTHR** register is written with data to transmit. During this event, the **UARTCIIR.INTSTATUS** will be set to 1b (interrupt pending) and the **UARTCIIR.INTID** will be set to 001b (TX holding register empty).

If the **UARTCIER.THREIE** is set to 1b and any of the flags in **UARTCLSR** are set, the IRQ_USARTC signal to the NVIC will be asserted.

27.5.10 Receive Operation

Data is sampled into the RX shift register at a sampling rate of PCLK / 16. A filter is used to remove spurious inputs that last for less than two periods of the sampling rate.

If the RX FIFO is enabled (**UARTCFCR.FIFOEN** = 1b), the data from the shift register will be loaded into the RX FIFO. The RX FIFO will load each character from the FIFO into the RX Buffer Register, until the FIFO is empty. The depth of the RX FIFO is 16-bytes.

If the RX FIFO is not enabled (**UARTCFCR.FIFOEN** = 0b), the received data will be loaded directly into the RX Buffer Register.

When the complete character has been loaded into the RX Buffer Register, it may be red using the **UARTCRBR** register. The receiver checks the parity and stop bits as specified in the **UARTCLCR** register.

27.5.11 Receive Interrupts

When characters are received from the UART, the user may configure the peripheral to indicate RX FIFO character time-out, incorrect parity, a missing stop bit (frame error) or other line status registers. To enable these interrupts, set the **UARTCIER.RLSIE** to 1b. If these conditions occur, the **UARTCIIR.INTSTATUS** will be set to 1b (interrupt pending) and the **UARTCIIR.INTID** will be set accordingly.

The UART may be configured to generate interrupts after a character has been successfully received.

To enable receive interrupts, set the **UARTCIER.RBRIE** (RX buffer register interrupt enable). When the **UARTCRBR** (receive buffer register) has been filled, the **UARTCLSR.RDR** will be set to 1b. This field will be cleared the next time the **UARTCRBR** register is read.

If the RX FIFO is enabled (**UARTCFCR.FIFOEN** = 1b), then the FIFO will generate an interrupt when 1, 4, 8 or 14 characters have been loaded into it. To set the interrupt threshold, set the **UARTCFCR.RXTL** to the interrupt FIFO depth that is desired. When the FIFO contains the

configured number of characters, the **UARTCIIR.INTSTATUS** will be set to a 1b (interrupt pending) and the **UARTCIIR.INTID** will be set accordingly.

If the **UARTCAIER.RBRIE** is set to 1b and the **UARTCLSR.RDR** interrupt flag is set, the IRQ_USARTC signal to the NVIC will be asserted.

27.6 UART Register Summary

Table 27-5 USART C Register Summary (UART Mode)

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
UARTCRBR	4004 0000h	UART C Receive Buffer Register	RO	--
UARTCTHR	4004 0004h	UART C Transmit Holding Register	WO	--
UARTCDLR	4004 0008h	UART C Divisor Latch Register	RW	0000 0001h
UARTCIER	4004 000Ch	UART C Interrupt Enable Register	RW	0000 0000h
UARTCIIR	4004 0010h	UART C Interrupt Identification Register	RO	0000 0001h
UARTCFCR	4004 0014h	UART C FIFO Control Register	RW	0000 0000h
UARTCLCR	4004 0018h	UART C Line control Register	RW	0000 0000h
UARTCLSR	4004 0020h	UART C Line Status Register	RO	0000 0060h
UARTCSCR	4004 0028h	UART C Scratch Pad Register	RW	--
UARTCEFR	4002 002Ch	UART C Enhanced Mode Register	RW	0000 000h

27.7 UART Register Detail

27.7.1 UARTCRBR

Register 27-1 UARTCRBR (UART C Receive Buffer Register, 4004 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**RBR**	RO	0	Contains the oldest received character in the UART RX FIFO.

27.7.2 UARTCTHR

Register 27-2 UARTCTHR (UART C Transmit Holding Register, 4004 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**THR**	WO	-	Writing to the **UARTBTHR** causes the data to be stored in the UART transmit FIFO. The character will be sent when it reaches the bottom of the FIFO and the transmitter is available.

27.7.3 UARTCDLR

Register 27-3 UARTCDLR (UART C Divisor Latch Register, 4004 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**DLR**	RW	0000 0001b	Sets the baud rate for the module. Baud rate = PCLK / (16 * **UARTCDLR**)

27.7.4 UARTCIER

Register 27-4 UARTCIER (UART C Interrupt Enable Register, 4004 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:3	**Reserved**	RO	0	Reserved
2	**RLSIE**	RW	0	RX Line Status Interrupt Enable. Enables the UART RX line status interrupts. The status of this interrupts can be read from **UARTCLSR[4:1]**. 0b: Disable the RX line status interrupts 1b: Enable the RX line status interrupts
1	**THRIE**	RW	0	TX Holding Register Empty Interrupt Enable. Enables the THRE interrupt for the UART. The status of this interrupt can be read from **UARTCLSR.THRE**. 0b: Disable the THRE interrupts 1b: Enable the THRE interrupts
0	**RBRIE**	RW	0	RX Buffer Register Interrupt Enable. Enables the Receive Data Available interrupt for the UART. It also controls the character receive time-out interrupt. 0b: Disable the RBR interrupts 1b: Enable the RBR interrupts

27.7.5 UARTCIIR

Register 27-5 UARTCIIR (UART C Interrupt Identification Register, 4004 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3:1	**INTID**	RO	000b	Interrupt identification: 000b: Reserved 001b: TX Holding Register Empty 010b: Receive Data Available 011b: Receive Line Status 100b: Reserved 101b: Reserved 110b: Receive FIFO Character Time-out 111b: Reserved
0	**INTSTATUS**	RO	1b	Interrupt status. Note that this bit is active low. The pending interrupt can be determined through the **UARTCIIR.INTID** field. 0b: At least one interrupt is pending 1b: No interrupt is pending

Note that this register is cleared on read.

27.7.6 UARTCFCR

Register 27-6 UARTCFCR (UART C FIFO Control Register, 4004 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:6	**RXTL**	RW	0	RX Trigger Level. These two bits determine how many receive UART FIFO characters must be written before an interrupt is activated. 00b: 1 character 01b: 4 characters 10b: 8 characters 11b: 14 characters
5:4	**TXTL**	RW	0	TX Trigger Level. These two bits determine how may transmit UART FIFO characters must be written before an interrupt is activated. 00b: 1 character 01b: 4 characters 10b: 8 characters 11b: 14 characters
3	**Reserved**	RO	0	Reserved
2	**TXFIFORST**	RW	0	TX FIFO Reset. 0b: No effect 1b: Clears all bytes in the UART TX FIFO. This bit is self-clearing
1	**RXFIFORST**	RW	0	RX FIFO Reset. 0b: No effect 1b: Clears all bytes in the UART RX FIFO. This bit is self-clearing
0	**FIFOEN**	RW	0	FIFO Enable: 0b: UART FIFOs are disabled. Reset UARTAFCR settings to their default values. 1b: UART RX and TX FIFOs are enabled. **UARTCFCR[7:1]** are accessible when this bit is set. Any transition on this bit will automatically clear the UART RX AND TX FIFOs.

27.7.7 UARTCLCR

Register 27-7 UARTCLCR (UART C Line Control Register, 4004 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:7	**Reserved**	RO	0	Reserved
6	**BCON**	RW	0	Break Control: 0b: Disable break transmission 1b: Enable break transmission. Output pin UARTCTXD is forced to logic 0 when this bit is set.
5:4	**PSEL**	RW	0	Parity Select: 00b: Odd parity. The number of 1s in the transmitted character and the attached parity will be odd. 01b: Even parity. The number of 1s in the transmitted character and the attached parity will be even. 10b: Forced 1 stick parity. 11b: Forced 0 stick parity.
3	**PEN**	RW	0	Parity Enable: 0b: Disable parity generation and checking 1b: Enable parity generation and checking
2	**SBS**	RW	0	Stop Bit Select: 0b: 1 stop bit 1b: 2 stop bits, 1.5 stop bits if (**UARTCLCR.WLS** = 00b)
1:0	**WLS**	RW	0	Word Length Select: 00b: 5-bit character length 01b: 6-bit character length 10b: 7-bit character length 11b: 8-bit character length

27.7.8 UARTCLSR

Register 27-8 UARTCLSR (UART C Line Status Register, 4004 0020h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**RXFE**	RO	0	Error in RX FIFO. This bit is set when a character with a RX error such as framing, parity or break is loaded into **UARTCRBR**. This bit is cleared when the **UARTCLSR** register is read, and there are no subsequent errors in the UART FIFO.
6	**TEMT**	RO	0	Transmitter Empty. This bit is set when both **UARTCTHR** and **UARTCTSR** are empty; this bit is cleared when THR contains valid data. 0b: THR contains valid data 1b: THR are empty
5	**THRE**	RO	0	Transmitter Holding Register Empty. This bit is set immediately upon detection of an empty UART THR and is cleared on a THR write. 0b: THR contains valid data 1b: THR is empty
4	**BI**	RO	0	Break Interrupt. When UARTCRX is held in the spacing state (all zeroes) for one full character transmission (start, data, parity, stop), a break interrupt occurs. Once the break condition has been detected, the receiver goes idle until UARTCRX goes to the marking statue (all ones). A read of the **UARTCLSR** register clears this status bit. The time of break detection is dependents on **UARTCFCR.FIFOEN**. The break interrupt is associated with the character at the top of the UART RBR FIFO. 0b: Break interrupt status is inactive 1b: Break interrupt status is active
3	**FE**	RO	0	Framing Error. When the stop bit of a received character is logic 0, a framing error occurs. Reading the **UARTCLSR** register will clear this bit. The time of the framing error is dependent upon **UARTCFCR[3]**. Upon detection of a framing error, the RX will attempt to re-synchronize the data and assume that the bad stop bit is actually an early start bit. However, it cannot be assumed that the next received byte will be correct even if there is no framing error. A framing error is associated with the character at the top of the UART RBR FIFO. 0b: Framing error status is inactive 1b: Framing error status is active
2	**PE**	RO	0	Parity Error. When the parity bit of a received character is in the wrong state, a parity error occurs. Reading the **UARTCLSR** register will clear this bit. The time of the parity error detection is dependent on **UARTCFCR.FIFOEN**.

				A parity error is associated with the character at the top of the UART RBR FIFO. 0b: Parity error status is inactive 1b: Parity error status is active
1	**OE**	RO	0	Overrun Error. The overrun error condition is set as soon as it occurs. Reading the **UARTCLSR** register will clear this bit. This bit is set when the UART RSR has a new character assembled and the UART RBR FIFO is full. In this case, the UART RBR FIFO will not be overwritten and the character in the UART RSR will be lost. 0b: Overrun error status is inactive 1b: Overrun error status is active
0	**RDR**	RO	0	Receiver Data Ready. This bit is set when the RBR holds an unread character and is cleared when the UART RBR FIFO is empty. 0b: RBR is empty 1b: RBR contains valid data

27.7.9 UARTCSCR

Register 27-9 UARTCSCR (UART C Scratch Pad Register, 4004 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**PAD**	RW	-	A read-able, write-able byte.

27.7.10 UARTCEFR

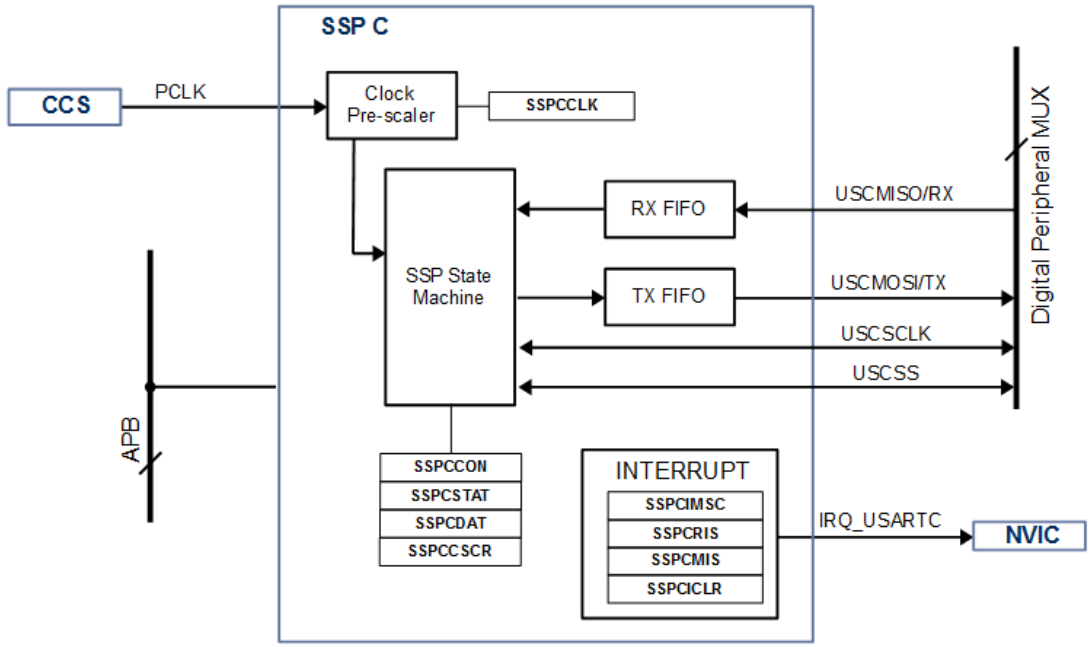
Register 27-10 UARTCEFR (UARTC Enhanced Feature Register, 4004 002Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**ENMODE**	RW	0	Enhanced Mode 0b – disabled 1b – enabled
3:0	**Reserved**	RO	0	Reserved

27.8 SSP Functional Description

27.8.1 SSP Mode System Block Diagram

Figure 27-3 SSP C System Block Diagram



27.8.2 Mode Configuration

The SSP mode of the USART must first be selected by writing **CCSCTL.USCMODE** to 0b. This will set the USART into SSP mode and allow all configuration registers to be used in this mode.

The configuration and status registers that are available when **CCSCTL.USCMODE** is 0b are below.

Table 27-6 USART C UART Mode Registers

REGISTER	DESCRIPTION
SSPCCON	SSP C Control Register
SSPCSTAT	SSP C Status Register
SSPCDAT	SSP C Data Register
SSPCCLK	SSP C Clock Control Register
SSPCIMSC	SSP C Interrupt Mask Set and Clear Register
SSPCRIS	SSP C Raw Interrupt Status Register
SSPCMIS	SSP C Masked Interrupt Status Register
SSPCICLR	SSP C Interrupt Clear Register

SSPCSSCR	SSP C Slave Select Configuration Register

27.8.3 SSP Overview

When in SSP mode, the USART may be a master or slave interface that enables synchronous serial communication with slave or master peripherals having one of the following:

- Motorola-style SPI
- TI-style Synchronous Serial Interface
- National Semiconductor-style Microwire

In both master and slave modes, the SSP performs parallel to serial conversion on data into 32-bit wide, 8-location deep RX and TX FIFOs.

The SSP may be configured to generate interrupts for servicing the TX and RX FIFOs and error conditions such as FIFO overrun and timeout.

27.8.4 Clock Configuration

When configured for master mode, the SSP peripheral may configure the clock pre-scaler to generate the desired SCLK output clock frequency. The input clock on the SSP peripheral is the PCLK (peripheral clock) system clock.

The SCLK is generated from the PCLK input and the **SSPCCLK.M** and **SSPCCLK.N** parameters. The SCLK clock frequency is generated according to the following formula:

F_{SSPCLK} = PCLK / ((**SSPCCLK.M** + 1) * **SSPCCLK.N**)

In this formula, **SSPCCLK.N** must be an even value from 2 to 254.

27.8.5 SSP Clock Constraints

The SSP clock configuration must be configured according to the guidelines below, in order for the peripheral to operate correctly.

Here are the clock constraints for the input clock SSPCLK (after the pre-scaler) and output SSP clock, USCCLK:

- When in master mode, the minimum frequency of the input SSPCLK must be at least 2X the output SSP clock (USCSCLK)
- When in slave mode, the minimum frequency of the input SSPCLK must be at least 12X the input SSP clock (USCSCLK)

The maximum frequency of the SSPCLK should be:

- When in master mode, F_{SSPCLK}(max) <= 254 * 128 * $F_{USCSCLK}$ (SSP clock output)

- When in slave mode, F_{SSPCLK}(max) <= 254 * 128 * $F_{USCSCLK}$ (SSP clock input)

27.8.6 SSP Configuration

The SSP may be configured to support various features of master and slave based serial interfaces.

To enable the SSP controller, set **SSPCCON.SSPEN** to 1b. When disabled, the SSP may still have registers configured for the configuration mode and interrupts, but will not operate on serial data until enabled. Once the SSP has been configured, set this bit to 1b to begin processing.

To configure the SSP mode, set **SSPCCON.FRF** (frame format) to the desired value as shown below.

Table 27-7 SSP C Frame Format

SSPCCON.FRF	Frame Format
00b	SPI (Motorola)
01b	Synchronous Serial Format (TI)
10b	Microwire (National Semiconductor)
11b	Reserved

To select master mode, set **SSPCCON.MS** to 0b. In this mode, the controller is the bus master. It will drive the SCLK, MOSI and SS signals and will receive data on the MISO line.

To select slave mode, set **SSPCCON.MS** to 1b. In this mode, the controller is the bus slave and will drive the MISO line and receive input from SCLK, MOSI and SS.

To change the serial to parallel endian order, the user may use the **SSPCCON.LSBFIRST** field. To configure the SSP for LSB first, set **SSPCCON.LSBFIRST** to 1b. To configure the SSP for MSB first, set **SSPCCON.LSBFIRST** to a 0b.

To enable loop-back mode, set the **SSPCCON.LBM** to 1b. In this mode, serial input is taken from the serial output instead of from the digital peripheral MUX.

While in slave mode, the behavior of the MISO output can be controlled. When **SSPCCON.SOD** is set to 0b, the SSP can drive the MISO in slave mode. When this bit is 1b, the SSP will not drive the MISO output in slave mode.

The clock phase of the SSP may also be configured. To configure the SSP for capturing data on the first clock edge transition, set **SSPCCON.CPH** to 0b. To configure the SSP for capturing data on the second clock edge transition, set **SSPCCON.CPH** to 1b.

When the **SSPCCON.FRF** is 00b (SPI), the clock output polarity may be configured. To configure the SSP clock to be active high, set **SSPCCON.CPO** to 0b. To configure the SSP clock to be active low, set **SSPCCON.CPO** to 1b.

The SSP may also configure the size of the data word size transferred in each frame. To configure the data size, see the table below.

Table 27-8 SSP C Data Size Select

SSPCCON.DSS	Data Size
0 0000b	Reserved
0 0001b	Reserved
0 0010b	Reserved
0 0011b	4-bit
0 0100b	5-bit
0 0101b	6-bit
0 0110b	7-bit
0 0111b	8-bit
0 1000b	9-bit
0 1001b	10-bit
0 1010b	11-bit
0 1011b	12-bit
0 1100b	13-bit
0 1101b	14-bit
0 1110b	15-bit
0 1111b	16-bit
…	…
1 1111b	32-bit

27.8.7 SSP Slave Select Configuration

The SSP allows the behavior of the SS (slave select) signal to be configured by using the **SSPCSSCR** register as follows when configured as a bus master.

The **SSPCSSCR.SELCS** field must always be set to a 00b for proper operation.

To configure the SSP to automatically control the behavior of the SS signal, set **SSPCSSCR.SWSEL** to 0b. If the user wants to control the SS signal by software, set **SSPCSSCR.SWSEL** to 1b.

To configure the SS signal behavior after a frame transfer, use the **SSPCSSCR.SPHDONTCARE** field. To configure the SSP to not pull the SS signal high after a frame transfer, set **SSPCSSCR.SPHDONTCARE** to 0b. To configure the SSP to pull the SS signal high after a frame transfer, set the **SSPCSSCR.SPHDONTCARE** to 1b.

27.8.8 SSP Interrupts

The SSP may configure interrupts for the following conditions.

When the SSP detects that the TX FIFO is half-full or less, the **SSPCRIS.TXRIS** and **SSPCMIS.TXMIS** are set to 1b. During this condition, if the **SSPCIMSC.TXIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTC signal to the NVIC.

When the SSP detects that the RX FIFO is half-full or more, the **SSPCRIS.RXRIS** and **SSPCMIS.RXMIS** are set to 1b. During this condition, if the **SSPCIMSC.RXIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTC signal to the NVIC.

The SSP contains a time for reading the RX FIFO. If the RX FIFO is not empty, and 32 F_{SSPCLK} periods have gone by, the **SSPCRIS.RTIM** and **SSPCMIS.RTMIS** bits are set to 1b. During this condition, if the **SSPCIMSC.RTIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTC signal to the NVIC. This condition may be cleared by writing **SSPCICLR.RTIC** to 1b.

If the SSP detects the RX FIFO is full, when a character is attempted to be inserted into it, is sets the **SSPCRIS.ROIM** and **SSPCMIS.ROMIS** to 1b. During this condition, if the **SSPCIMSC.ROIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTC signal to the NVIC. This condition may be cleared by writing **SSPCICLR.ROIC** to 1b.

27.8.9 Master Mode Operation

When configured as a bus master (**SSPCCON.MS** = 0b), the SSP will insert a character into the TX FIFO when it is written into the **SSPCDAT** register. The SSP will read a character from the TX FIFO and performs a parallel to serial conversion on it. Then the serial data stream and frame control signal is synchronized to the clock and are output through the USCMOSI/TX pin to the attached slaves. While the data is being transmit to the slaves, the USCSCLK and USCSS behave as configured in the **SSPCCON** and **SSPCCSCR** registers.

The master receive logic performs serial to parallel conversion on the incoming synchronous USCMISO/RX data stream and stores the character in the RX FIFO. The master may read first entry in the RX FIFO by reading the **SSPCDAT** register.

27.8.10 Slave Mode Operation

When configured as a bus slave (**SSPCCON.MS** = 1b), the SCLK is provided by the attached master. The user can write the next character to send into **SSPCDAT** and it will get inserted into the TX FIFO. The slave transmit logic reads a value from the TX FIFO, performs parallel to serial conversion and outputs the serial data stream on the USCMISO/RX pin to the attached master. The slave receive logic performs serial to parallel conversion on the incoming USCMOSI/TX data stream, extracting and storing values into the RX FIFO. The first character in the RX FIFO may be read by reading the **SSPCDAT** register.

27.8.11 SSP Status

The SSP module maintains status information on the operation of the module. The table below shows the available SSP status conditions.

Table 27-9 SSP C Status

SSPCSTAT field	Description
BSY	If set, the SSP controller is currently sending/receiving a frame and/or the TX FIFO is not empty.
RFF	If set, the RX FIFO is full.
RNE	If set, the RX FIFO is not empty.
TNF	If set, the TX FIFO is not full.
TFE	If set, the TX FIFO is empty

27.9 SSP Register Summary

Table 27-10 USART C Register Summary (SSP Mode)

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
SSPCCON	4004 0000h	SSP C Control Register	RW	0000 0000h
SSPCSTAT	4004 0004h	SSP C Status Register	RO	0000 0003h
SSPCDAT	4004 0008h	SSP C Data Register	RW	0000 0000h
SSPCCLK	4004 000Ch	SSP C Clock Control Register	RW	0000 0000h
SSPCIMSC	4004 0010h	SSP C Interrupt Mask Set and Clear Register	RW	0000 0000h
SSPCRIS	4004 0014h	SSP C Raw Interrupt Status Register	RO	0000 0008h
SSPCMIS	4004 0018h	SSP C Masked Interrupt Status Register	RO	0000 0008h
SSPCICLR	4004 001Ch	SSP C Interrupt Clear Register	RW	0000 0000h
SSPCSSCR	4004 0028h	SSP C Slave Select Configuration Register	RW	0000 0000h

27.10 SSP Register Detail

27.10.1 SSPCCON

Register 27-11 SSPCCON (SSP C Control Register, 4004 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:14	**Reserved**	RO	0	Reserved
13	**LSBFIRST**	RW	0	Endian Order: 0b: MSB transmit / receive first 1b: LSB transmit / receive first
12	**LBM**	RW	0	Loopback Mode: 0b: Normal operation 1b: Loopback mode. Serial Input is taken from the serial output (MOSI or MISO) rather than the serial in (MISO or MOSI respectively)
11	**SSPEN**	RW	0	SSP Enable: 0b: SSP Controller is disabled 1b: SSP Controller is enabled. It will interact with other devices on the serial bus. Software should write the appropriate control information to the other SPI/SSP registers and interrupt controller registers, before setting this bit.
10	**MS**	RW	0	Master/Slave Mode: 0b: The SSP controller acts as a bus master, driving the USCSCLK, USCMOSI and USCSS signals 1b: The SSP controller acts as a bus slave, driving the USCMISO and receiving USCSCLK, USCMOSI and USCSS.
9	**SOD**	RW	0	Slave Output Disable: 0b: The SSP can drive the USCMISO output in slave mode 1b: The SSP must not drive the USCMISO output in slave mode
8	**CPH**	RW	0	Clock Out Phase. This bit is only used when **SSPCCON.FRF** = 0b (SPI). 0b: The SSP controller captures serial data on the first edge transition of the frame. 1b: The SSP controller captures serial data on the second edge transition of the frame.
7	**CPO**	RW	0	Clock Out Polarity (this bit is only used in SPI mode when **SSCCON.FRF** = 00b) 0b: The clock is active high. 1b: The clock is active low.
6:5	**FRF**	RW	0	Frame Format: 00b: SPI 01b: TI 10b: Microwire 11b: Reserved
4:0	**DSS**	RW	0	Data Size Select: 0 0000b: Reserved 0 0001b: Reserved 0 0010b: Reserved 0 0011b: 4-bit transfer 0 0100b: 5-bit transfer

				0 0101b: 6-bit transfer 0 0110b: 7-bit transfer 0 0111b: 8-bit transfer 0 1000b: 9-bit transfer 0 1001b: 10-bit transfer 0 1010b: 11-bit transfer 0 1011b: 12-bit transfer 0 1100b: 13-bit transfer 0 1101b: 14-bit transfer 0 1110b: 15-bit transfer 0 1111b: 16-bit transfer … 1 1111b: 32-bit transfer

27.10.2 SSPCSTAT

Register 27-12 SSPCSTAT (SSP C Status Register, 4004 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**BSY**	RO	0	Busy Bit: 0b: SPI controller is idle 1b: SPI controller is sending/receiving a frame and/or the TX FIFO is not empty
3	**RFF**	RO	0	Receive FIFO Full: 0b: RX FIFO not full 1b: RX FIFO full
2	**RNE**	RO	0	Receive FIFO not empty: 0b: RX FIFO empty 1b: RX FIFO not empty
1	**TNF**	RO	0	Transmit FIFO not full: 0b: TX FIFO full 1b: TX FIFO not full
0	**TFE**	RO	0	Transmit FIFO empty: 0b: TX FIFO not empty 1b: TX FIFO empty

27.10.3 SSPCDAT

Register 27-13 SSPCDAT (SSP C Data Register, 4004 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**DATA**	RW	0	Software can writing data to be sent in a future frame to this register when the **SSPCSTAT.TNF** bit is set to 1b, indicating that the TX FIFO was previously empty and the SPI controller is not busy on the bus, transmission of the data will begin immediately. Otherwise, the data written to this register will be sent as soon as all previous data has been sent (and received). If the data length is less than 32-bits, the data must be right-justified in this register. Software may read data from this register when the **SSPCSTAT.RNE** bit is set to a 1b, indicating that the RX FIFO is not empty. When software reads this register, the SPI controller returns the data from the least recent frame in the RX FIFO. If the data length is less than 32-bits, the data is right-justified in this field with the MSBs set to 0.

27.10.4 SSPCCLK

Register 27-14 SSPCCLK (SSP C Clock Register, 4004 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:8	**M**	RW	0	These fields may set the SPI master clock rate by the formula: F_{SSPCLK} = PCLK / ((**SSPCCLK.M** + 1) * **SSPCCLK.N**) N must be an even value from 2 to 254.
7:0	**N**	RW	0	

27.10.5 SSPCIMSC

Register 27-15 SSPCIMSC (SSP C Interrupt Mask Set and Clear Enable Register, 4004 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXIM**	RW	0	Transmit FIFO Interrupt Mask: 0b: TX FIFO half-full or less condition interrupt disabled 1b: TX FIFO half-full or less condition interrupt enabled
2	**RXIM**	RW	0	Receive FIFO Interrupt Mask: 0b: RX FIFO half-full or more condition interrupt disabled 1b: RX FIFO half-full or more condition interrupt enabled
1	**RTIM**	RW	0	Receive Timeout Interrupt Mask: 0b: RX FIFO not empty and no read prior to timeout period interrupt disabled 1b: RX FIFO not empty and no read prior to timeout period interrupt enabled
0	**ROIM**	RW	0	Receive Overrun Interrupt Mask: 0b: RX FIFO written to while full condition interrupt disabled 1b: RX FIFO written to while full condition interrupt enabled

27.10.6 SSPCRIS

Register 27-16 SSPCRIS (SSP C Raw Interrupt Status Register, 4004 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXRIS**	RO	1	TX FIFO Raw Interrupt Status: 0b: TX FIFO not at least half-full 1b: TX FIFO at least half-full
2	**RXRIS**	RO	0	RX FIFO Raw Interrupt Status: 0b: RX FIFO not at least half-full 1b: RX FIFO at least half-full
1	**RTRIS**	RO	0	RX FIFO Timeout Interrupt Status: 1b: RX FIFO is not empty and has not been read for the timeout period. The timeout period is 32-bit times at F_{SSPCLK}
0	**RORIS**	RO	0	RX FIFO Overrun Interrupt Status: 1b: RX FIFO was full when another frame was completely received. The preceding data frame is overrun by the new frame when this occurs.

27.10.7 SSPCMIS

Register 27-17 SSPCMIS (SSP C Masked Interrupt Status Register, 4004 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXMIS**	RO	1	TX FIFO Masked Interrupt Status: 0b: TX FIFO not at least half-full 1b: TX FIFO at least half-full
2	**RXMIS**	RO	0	RX FIFO Masked Interrupt Status: 0b: RX FIFO not at least half-full 1b: RX FIFO at least half-full
1	**RTMIS**	RO	0	RX FIFO Timeout Interrupt Status: 1b: RX FIFO is not empty and has not been read for the timeout period. The timeout period is 32-bit times at F_{SSPCLK}
0	**ROMIS**	RO	0	RX FIFO Overrun Interrupt Status: 1b: RX FIFO was full when another frame was completely received. The preceding data frame is overrun by the new frame when this occurs.

27.10.8 SSPCICLR

Register 27-18 SSPCICLR (SSP C Interrupt Clear Register, 4004 001Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:2	**Reserved**	RO	0	Reserved
1	**RTIC**	RW	0	RX FIFO Timeout Interrupt Clear: 0b: No effect 1b: Clears the **SSPCxIS.RTRIS** interrupt flag This bit is self-clearing.
0	**ROIC**	RW	0	RX FIFO Overwrite Interrupt Clear: 0b: No effect 1b: Clears the **SSPCxIS.RORIS** interrupt flag This bit is self-clearing.

27.10.9 SSPCSSCR

Register 27-19 SSPCSSCR (SSP C Slave Select Configuration Register, 4004 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**SPHDONTCARE**	RW	0	Slave Select Pull High: 0b: USCSS cannot pull high after frame transfer 1b: USCSS must pull high after frame transfer
3	**SWSS**	RW	0	Slave Select State: 0b: Set USCSS to low 1b: Set USCSS to high
2	**SWSEL**	RW	0	Slave Select Software Control: 0b: USCSS is automatically controlled by the SPI module 1b: USCSS is software controlled by **SSPCSSCR.SWSS**
1:0	**SELSS**	RW	0	Slave Select Signal Control: 00b: USCSS is enabled 01b: Reserved 10b: Reserved 11b: Reserved

28 USART D

28.1 Overview

The PAC55XX family contains support for four Universal Synchronous Asynchronous Receive Transmit (USART) peripherals.

A USART is a serial communications engine that may be configured for UART or SSP through a mode selection register.

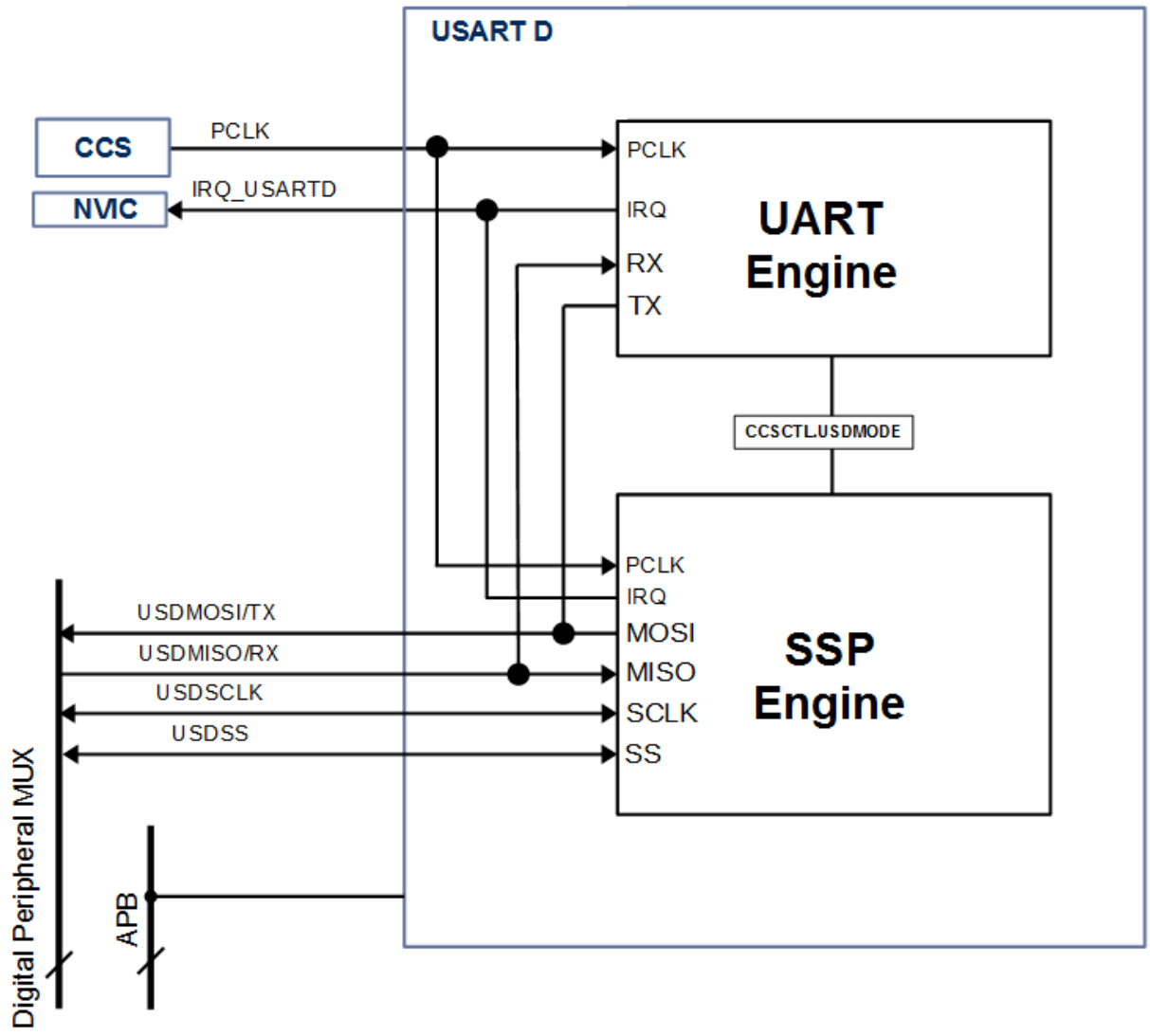
The USART is an APB client and is clocked by the Peripheral Clock (PCLK) system clock.

28.2 Features

- UART Mode:
  - 5-bit to 8-bit data interface
  - Optional parity
  - 1, 1.5 or 2 stop bits
  - 16-bit programmable baud-rate generator
  - 8-bit scratch pad
  - Independent RX and TX FIFOs
  - Configurable RX and TX interrupts
- SSP Mode:
  - Motorola SPI, TI Synchronous serial or National Semiconductor Microwire support
  - Master and Slave mode support
  - Independent RX and TX FIFOs
  - Configurable clock pre-scaler
  - Programmable Interrupts

28.3 System Block Diagram

Figure 28-1 USART D System Block Diagram



28.4 Functional Description

The USART may be configured for either UART or SSP mode via the **CCSCTL.USDMODE** register field.

In UART mode, the peripheral can receive or transmit UART serial data over the USDMOSI/TX (UART TX) or USDMISO/RX serial lines. In this mode, the USART signals for USDSCLK and USBSS are unused. The UART may assert the IRQ_USARTD signal to the NVIC for interrupts and is clocked using the PCLK system clock and connected to the APB bus.

In UART mode, only the UART registers may be accessed, starting at register offset 0.

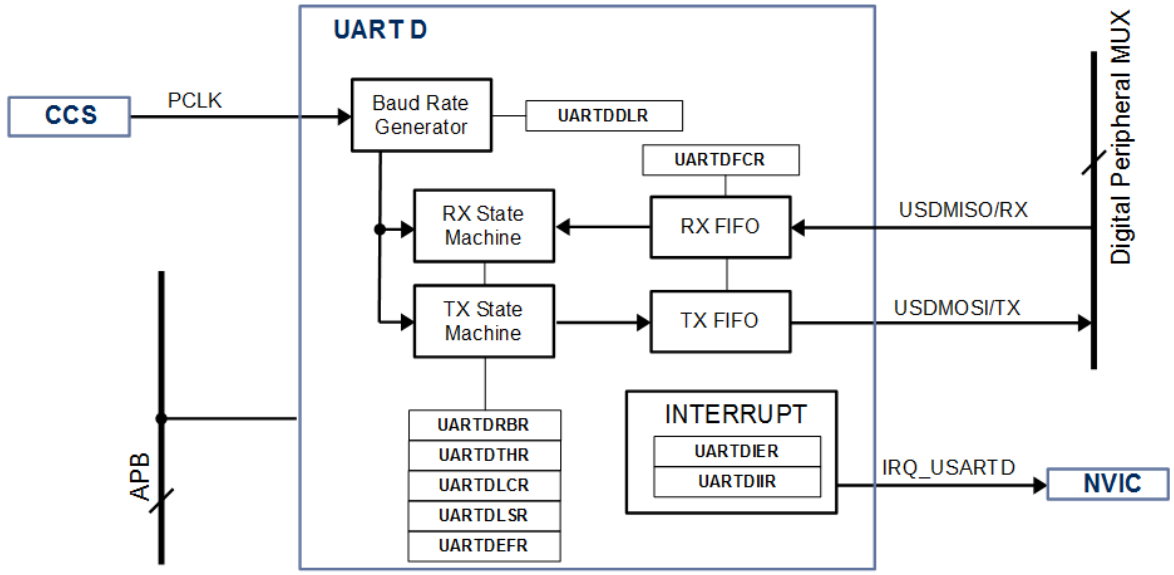
In SSP mode, the peripheral can receive or transmit serial data over the USDMOSI/TX (SPI MOSI) or USDMISO/RX (SPI MISO) serial lines. The USDSCLK and USDSS are used by the peripheral in either master or slave mode for the SSP clock and Slave Select signals. The SSP may assert the IRQ_USARTD signal to the NVIC for interrupts, and is clocked using the PCLK system clock and connected to the APB bus.

In SPI mode, only the SPI registers may be accessed, starting at register offset 0.

28.5 UART Mode Functional Description

28.5.1 UART D Mode System Block Diagram

Figure 28-2 UART D Mode System Block Diagram



28.5.2 Mode Configuration

The UART mode of the USART must first be selected by writing **CCSCTL.USDMODE** to 1b. This will set the USART into UART mode and allow all configuration registers to be used in this mode.

The configuration and status registers that are available when **CCSCTL.USDMODE** is 1b are below.

Table 28-1 USART D UART Mode Registers

REGISTER	DESCRIPTION
UARTDRBR	UART D Receive Buffer Register
UARTDTHR	UART D Transmit Holding Register
UARTDDLR	UART D Divisor Latch Register
UARTDIER	UART D Interrupt Enable Register
UARTDIIR	UART D Interrupt Identification Register
UARTDFCR	UART D FIFO Control Register
UARTDLCR	UART D Line control Register
UARTDLSR	UART D Line Status Register
UARTDSCR	UART D Scratch Pad Register

28.5.3 Baud Rate Configuration

To configure the baud rate for RX/TX operations, the **UARTDDLR** may be used.

To baud rate may be configured by the following formula:

Baud Rate = PCLK / (16 * **UARTDDLR**)

Note that UART peripherals sometimes need an accurate timing base in order to function properly. Because of that, it is suggested that the CLKREF or EXTCLK be used as the clock source for PCLK when using the UART.

The ROSCCLK may not provide enough accuracy for UART applications.

28.5.4 FIFO Reset

The RX and TX FIFOs may be reset independently by using the **UARTDFC** register.

To change any of the bits in the **UARTDFCR**, the **UARTDFCR.FIFOEN** must be set to 1b (**UARTDFCR** access).

To reset the RX FIFO, write the **UARTDFCR.RXFIFORST** bit to 1b. This will clear all bytes in the FIFO and reset the pointer to the top of the FIFO for the RX state machine.

To reset the TX FIFO, write the **UARTDFCR.TXFIFORST** bit to 1b. This will clear all bytes in the FIFO and reset the pointer to the top of the FIFO for the TX state machine.

When the **UARTDFCR** register is done being used to configure the FIFOs, set the **UARTDFCR.FIFOEN** to 1b (application mode).

28.5.5 UART Configuration

In UART mode, the UART may be have the parity, break control, stop bits and word length configured by the user.

The parity checking may be enabled or disabled. To enable parity checking, set the **UARTDLCR.PEN** to 1b. To disable, set this field to 0b. If enabled, the user may configure the type of parity checking as follows.

Table 28-2 UART D Parity Modes

UARTDLCR.PEN	UARTDLCR.PSEL	Parity	Description
0b	n/a	None	
1b	00b	Odd	The number of ones in the transmitted character plus the attached parity bit will be odd.
	01b	Even	The number of ones in the transmitted character plus the attached parity bit will be even.
	10b	Forced 1 stick	Force 1 for parity
	11b	Forced 0 stick	Force 0 for parity

To set the number of stop bits, the user may write the **UARTDLCR.SBS** field. To select 1 stop bits, set **UARTDLCR.SBS** to 0b. To select 2 stop bits, set **UARTDLCR.SBS** to 1b. If the **UARTDLCR.WLS** is set to 00b (5-bit character length) and **UARTDLCR.SBS** is set to 1b, then there will be 1.5 stop bits.

To configure break control, the user can set the **UARTDLCR.BCON** field. To disable break transmission, set **UARTDLCR.BCON** to 0b. To force the TX signal to logic 0, set the **UARTDLCR.BCON** to 1b.

To set the word length for the UART, the user can set the **UARTDLCR.WLS** field as shown below.

Table 28-3 UART D Word Length

UARTCLCR.WLS	UART word length
00b	5-bit
01b	6-bit
10b	7-bit
11b	8-bit

28.5.6 UART Scratch Pad

There is an 8-bit general purpose register that may be used in the USART.

The user may use the **USDSCR** register for an 8-bit scratch pad.

28.5.7 UART Interrupts

The status of interrupts may be read at any time by the **UARTDIIR** register. If the **UARTDIIR.INTSTATUS** field is set to 0b, at least one interrupt is pending. The interrupt type can be read by the **UARTDIIR.INTID** field as shown below.

Table 28-4 UART D Interrupt ID

UARTCIIR.INTID	Interrupt Type
000b	Reserved
001b	TX Holding Register Empty
010b	Receive Data Available
011b	Receive Line Status
100b	Reserved
101b	Reserved
110b	Receive FIFO character time-out
111b	Reserved

28.5.8 Transmit Operation

Transmission may be initiated by writing **UARTDTHR** with the data desired to be sent.

If the TX FIFO is enabled (**UARTDFCR.FIFOEN** = 1b), the data will be written into the TX FIFO. The contents of the TX FIFO will be transferred to the TX shift register one character at a time, until the TX FIFO is empty. The depth of the TX FIFO is 16-bytes.

If the TX FIFO is not enabled (**UARTDFCR.FIFOEN** = 0b), the data will be transferred to the TX shift register.

The bits to be transmitted are then shifted out of the TX shift register in the order of start bit, data bits (LSB first), parity bit, stop bit, using the configuration from the **UARTDLCR** register.

The baud rate used will be set by the **UARTDDLR** register.

28.5.9 Transmit Interrupts

If the TX FIFO is enabled (**UARTDFCR.FIFOEN** = 1b), the UART may be configured to generate interrupts after characters have been successfully transmitted. The UART may be configured to generate an interrupt after 1, 4, 8 or 14 characters have been transmitted. To set the interrupt threshold, the user should set the **UARTDEFR.ENMODE** = 1b and set the **UARTDFCR.TXTL** to the interrupt FIFO depth that is desired.

To enable transmit interrupts, set the **UARTDIER.THREIE** (TX holding register empty interrupt enable). When the transmit holding register has been emptied and the TX FIFO is empty (if **UARTDFCR.FIFOEN** = 1b), the **UARTDLSR.THRE** will be set to 1b. This field will be cleared the next time the **UARTDTHR** register is written with data to transmit. During this event, the **UARTDIIR.INTSTATUS** will be set to 1b (interrupt pending) and the **UARTDIIR.INTID** will be set to 001b (TX holding register empty).

If the **UARTDIER.THREIE** is set to 1b and any of the flags in **UARTDLSR** are set, the IRQ_USARTD signal to the NVIC will be asserted.

28.5.10 Receive Operation

Data is sampled into the RX shift register at a sampling rate of PCLK / 16. A filter is used to remove spurious inputs that last for less than two periods of the sampling rate.

If the RX FIFO is enabled (**UARTDFCR.FIFOEN** = 1b), the data from the shift register will be loaded into the RX FIFO. The RX FIFO will load each character from the FIFO into the RX Buffer Register, until the FIFO is empty. The depth of the RX FIFO is 16-bytes.

If the RX FIFO is not enabled (**UARTDFCR.FIFOEN** = 0b), the received data will be loaded directly into the RX Buffer Register.

When the complete character has been loaded into the RX Buffer Register, it may be red using the **UARTDRBR** register. The receiver checks the parity and stop bits as specified in the **UARTDLCR** register.

28.5.11 Receive Interrupts

When characters are received from the UART, the user may configure the peripheral to indicate RX FIFO character time-out, incorrect parity, a missing stop bit (frame error) or other line status registers. To enable these interrupts, set the **UARTDIER.RLSIE** to 1b. If these conditions occur, the **UARTDIIR.INTSTATUS** will be set to 1b (interrupt pending) and the **UARTDIIR.INTID** will be set accordingly.

The UART may be configured to generate interrupts after a character has been successfully received.

To enable receive interrupts, set the **UARTDIER.RBRIE** (RX buffer register interrupt enable). When the **UARTDRBR** (receive buffer register) has been filled, the **UARTDLSR.RDR** will be set to 1b. This field will be cleared the next time the **UARTDRBR** register is read.

If the RX FIFO is enabled (**UARTDFCR.FIFOEN** = 1b), then the FIFO will generate an interrupt when 1, 4, 8 or 14 characters have been loaded into it. To set the interrupt threshold, set the **UARTDFCR.RXTL** to the interrupt FIFO depth that is desired. When the FIFO contains the

configured number of characters, the **UARTDIIR.INTSTATUS** will be set to a 1b (interrupt pending) and the **UARTDIIR.INTID** will be set accordingly.

If the **UARTDIER.RBRIE** is set to 1b and the **UARTDLSR.RDR** interrupt flag is set, the IRQ_USARTD signal to the NVIC will be asserted.

28.6 UART Register Summary

Table 28-5 USART D Register Summary (UART Mode)

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
UARTDRBR	4005 0000h	UART D Receive Buffer Register	RO	--
UARTDTHR	4005 0004h	UART D Transmit Holding Register	WO	--
UARTDDLR	4005 0008h	UART D Divisor Latch Register	RW	0000 0001h
UARTDIER	4005 000Ch	UART D Interrupt Enable Register	RW	0000 0000h
UARTDIIR	4005 0010h	UART D Interrupt Identification Register	RO	0000 0001h
UARTDFCR	4005 0014h	UART D FIFO Control Register	RW	0000 0000h
UARTDLCR	4005 0018h	UART D Line control Register	RW	0000 0000h
UARTDLSR	4005 0020h	UART D Line Status Register	RO	0000 0060h
UARTDSCR	4005 0028h	UART D Scratch Pad Register	RW	--
UARTDEFR	4002 002Ch	UART D Enhanced Mode Register	RW	0000 000h

28.7 UART Register Detail

28.7.1 UARTDRBR

Register 28-1 UARTDRBR (UART D Receive Buffer Register, 4005 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**RBR**	RO	0	Contains the oldest received character in the UART RX FIFO.

28.7.2 UARTDTHR

Register 28-2 UARTDTHR (UART D Transmit Holding Register, 4005 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**THR**	WO	-	Writing to the **UARTBTHR** causes the data to be stored in the UART transmit FIFO. The character will be sent when it reaches the bottom of the FIFO and the transmitter is available.

28.7.3 UARTDDLR

Register 28-3 UARTDDLR (UART D Divisor Latch Register, 4005 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:0	**DLR**	RW	0000 0001b	Sets the baud rate for the module. Baud rate = PCLK / (16 * **UARTDDLR**)

28.7.4 UARTDIER

Register 28-4 UARTDIER (UART D Interrupt Enable Register, 4005 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:3	**Reserved**	RO	0	Reserved
2	**RLSIE**	RW	0	RX Line Status Interrupt Enable. Enables the UART RX line status interrupts. The status of this interrupts can be read from **UARTDLSR[4:1]**. 0b: Disable the RX line status interrupts 1b: Enable the RX line status interrupts
1	**THRIE**	RW	0	TX Holding Register Empty Interrupt Enable. Enables the THRE interrupt for the UART. The status of this interrupt can be read from **UARTDLSR.THRE**. 0b: Disable the THRE interrupts 1b: Enable the THRE interrupts
0	**RBRIE**	RW	0	RX Buffer Register Interrupt Enable. Enables the Receive Data Available interrupt for the UART. It also controls the character receive time-out interrupt. 0b: Disable the RBR interrupts 1b: Enable the RBR interrupts

28.7.5 UARTDIIR

Register 28-5 UARTDIIR (UART D Interrupt Identification Register, 4005 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3:1	**INTID**	RO	000b	Interrupt identification: 000b: Reserved 001b: TX Holding Register Empty 010b: Receive Data Available 011b: Receive Line Status 100b: Reserved 101b: Reserved 110b: Receive FIFO Character Time-out 111b: Reserved
0	**INTSTATUS**	RO	1b	Interrupt status. Note that this bit is active low. The pending interrupt can be determined through the **UARTDIIR.INTID** field. 0b: At least one interrupt is pending 1b: No interrupt is pending

Note that this register is cleared on read.

28.7.6 UARTDFCR

Register 28-6 UARTDFCR (UART D FIFO Control Register, 4005 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:6	**RXTL**	RW	0	RX Trigger Level. These two bits determine how many receive UART FIFO characters must be written before an interrupt is activated. 00b: 1 character 01b: 4 characters 10b: 8 characters 11b: 14 characters
5:4	**TXTL**	RW	0	TX Trigger Level. These two bits determine how may transmit UART FIFO characters must be written before an interrupt is activated. 00b: 1 character 01b: 4 characters 10b: 8 characters 11b: 14 characters
3	**Reserved**	RO	0	Reserved
2	**TXFIFORST**	RW	0	TX FIFO Reset. 0b: No effect 1b: Clears all bytes in the UART TX FIFO. This bit is self-clearing
1	**RXFIFORST**	RW	0	RX FIFO Reset. 0b: No effect 1b: Clears all bytes in the UART RX FIFO. This bit is self-clearing
0	**FIFOEN**	RW	0	FIFO Enable: 0b: UART FIFOs are disabled. Reset UARTAFCR settings to their default values. 1b: UART RX and TX FIFOs are enabled. **UARTDFCR[7:1]** are accessible when this bit is set. Any transition on this bit will automatically clear the UART RX AND TX FIFOs.

28.7.7 UARTDLCR

Register 28-7 UARTDLCR (UART D Line Control Register, 4005 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:7	**Reserved**	RO	0	Reserved
6	**BCON**	RW	0	Break Control: 0b: Disable break transmission 1b: Enable break transmission. Output pin UARTDTXD is forced to logic 0 when this bit is set.
5:4	**PSEL**	RW	0	Parity Select: 00b: Odd parity. The number of 1s in the transmitted character and the attached parity will be odd. 01b: Even parity. The number of 1s in the transmitted character and the attached parity will be even. 10b: Forced 1 stick parity. 11b: Forced 0 stick parity.
3	**PEN**	RW	0	Parity Enable: 0b: Disable parity generation and checking 1b: Enable parity generation and checking
2	**SBS**	RW	0	Stop Bit Select: 0b: 1 stop bit 1b: 2 stop bits, 1.5 stop bits if (**UARTDLCR.WLS** = 00b)
1:0	**WLS**	RW	0	Word Length Select: 00b: 5-bit character length 01b: 6-bit character length 10b: 7-bit character length 11b: 8-bit character length

28.7.8 UARTDLSR

Register 28-8 UARTDLSR (UART D Line Status Register, 4005 0020h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**RXFE**	RO	0	Error in RX FIFO. This bit is set when a character with a RX error such as framing, parity or break is loaded into **UARTDRBR**. This bit is cleared when the **UARTDLSR** register is read, and there are no subsequent errors in the UART FIFO.
6	**TEMT**	RO	0	Transmitter Empty. This bit is set when both **UARTDTHR** and **UARTDTSR** are empty; this bit is cleared when THR contains valid data. 0b: THR contains valid data 1b: THR is empty
5	**THRE**	RO	0	Transmitter Holding Register Empty. This bit is set immediately upon detection of an empty UART THR and is cleared on a THR write. 0b: THR contains valid data 1b: THR is empty
4	**BI**	RO	0	Break Interrupt. When UARTDRX is held in the spacing state (all zeroes) for one full character transmission (start, data, parity, stop), a break interrupt occurs. Once the break condition has been detected, the receiver goes idle until UARTDRX goes to the marking statue (all ones). A read of the **UARTDLSR** register clears this status bit. The time of break detection is dependents on **UARTDFCR.FIFOEN**. The break interrupt is associated with the character at the top of the UART RBR FIFO. 0b: Break interrupt status is inactive 1b: Break interrupt status is active
3	**FE**	RO	0	Framing Error. When the stop bit of a received character is logic 0, a framing error occurs. Reading the **UARTDLSR** register will clear this bit. The time of the framing error is dependent upon **UARTDFCR[3]**. Upon detection of a framing error, the RX will attempt to re-synchronize the data and assume that the bad stop bit is actually an early start bit. However, it cannot be assumed that the next received byte will be correct even if there is no framing error. A framing error is associated with the character at the top of the UART RBR FIFO. 0b: Framing error status is inactive 1b: Framing error status is active
2	**PE**	RO	0	Parity Error. When the parity bit of a received character is in the wrong state, a parity error occurs. Reading the **UARTDLSR** register will clear this bit. The time of the parity error detection is dependent on **UARTDFCR.FIFOEN**.

				A parity error is associated with the character at the top of the UART RBR FIFO. 0b: Parity error status is inactive 1b: Parity error status is active
1	**OE**	RO	0	Overrun Error. The overrun error condition is set as soon as it occurs. Reading the **UARTDLSR** register will clear this bit. This bit is set when the UART RSR has a new character assembled and the UART RBR FIFO is full. In this case, the UART RBR FIFO will not be overwritten and the character in the UART RSR will be lost. 0b: Overrun error status is inactive 1b: Overrun error status is active
0	**RDR**	RO	0	Receiver Data Ready. This bit is set when the RBR holds an unread character and is cleared when the UART RBR FIFO is empty. 0b: RBR is empty 1b: RBR contains valid data

28.7.9 UARTDSCR

Register 28-9 UARTDSCR (UART D Scratch Pad Register, 4005 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**PAD**	RW	-	A read-able, write-able byte.

28.7.10 UARTDEFR

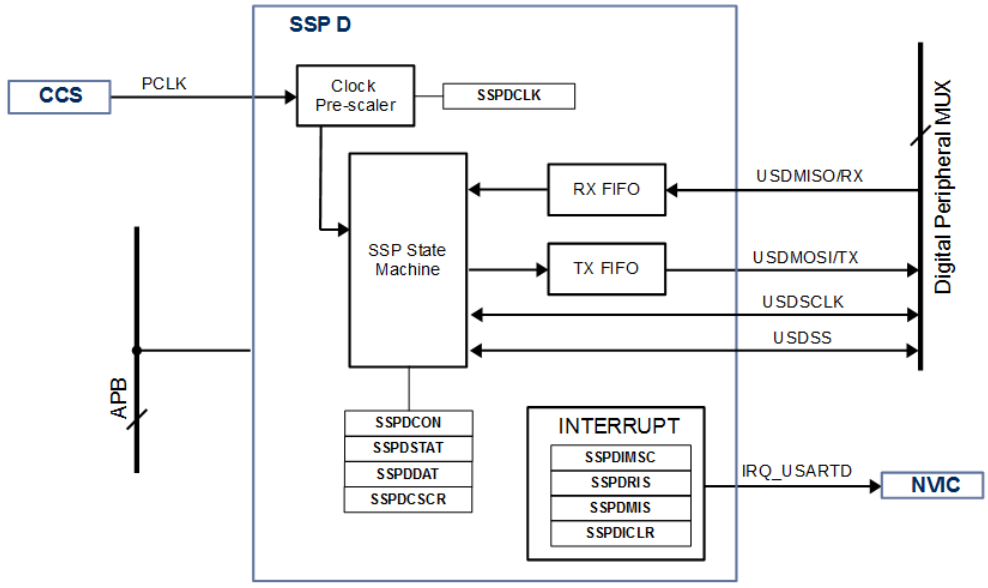
Register 28-10 UARTDEFR (UART D Enhanced Feature Register, 4005 002Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**ENMODE**	RW	0	Enhanced Mode 0b – disabled 1b – enabled
3:0	**Reserved**	RO	0	Reserved

28.8 SSP Functional Description

28.8.1 SSP Mode System Block Diagram

Figure 28-3 SSP D System Block Diagram



28.8.2 Mode Configuration

The SSP mode of the USART must first be selected by writing **CCSCTL.USDMODE** to 0b. This will set the USART into SSP mode and allow all configuration registers to be used in this mode.

The configuration and status registers that are available when **CCSCTL.USDMODE** is 0b are below.

Table 28-6 USART D UART Mode Registers

REGISTER	DESCRIPTION
SSPDCON	SSP D Control Register
SSPDSTAT	SSP D Status Register
SSPDDAT	SSP D Data Register
SSPDCLK	SSP D Clock Control Register
SSPDIMSC	SSP D Interrupt Mask Set and Clear Register
SSPDRIS	SSP D Raw Interrupt Status Register
SSPDMIS	SSP D Masked Interrupt Status Register
SSPDICLR	SSP D Interrupt Clear Register
SSPDSSCR	SSP D Slave Select Configuration Register

28.8.3 SSP Overview

When in SSP mode, the USART may be a master or slave interface that enables synchronous serial communication with slave or master peripherals having one of the following:

- Motorola-style SPI
- TI-style Synchronous Serial Interface
- National Semiconductor-style Microwire

In both master and slave modes, the SSP performs parallel to serial conversion on data into 32-bit wide, 8-location deep RX and TX FIFOs.

The SSP may be configured to generate interrupts for servicing the TX and RX FIFOs and error conditions such as FIFO overrun and timeout.

28.8.4 Clock Configuration

When configured for master mode, the SSP peripheral may configure the clock pre-scaler to generate the desired SCLK output clock frequency. The input clock on the SSP peripheral is the PCLK (peripheral clock) system clock.

The SCLK is generated from the PCLK input and the **SSPDCLK.M** and **SSPDCLK.N** parameters. The SCLK clock frequency is generated according to the following formula:

F_{SSPCLK} = PCLK / ((**SSPDCLK.M** + 1) * **SSPDCLK.N**)

In this formula, **SSPDCLK.N** must be an even value from 2 to 254.

28.8.5 SSP Clock Constraints

The SSP clock configuration must be configured according to the guidelines below, in order for the peripheral to operate correctly.

Here are the clock constraints for the input clock SSPCLK (after the pre-scaler) and output SSP clock, USDCLK:

- When in master mode, the minimum frequency of the input SSPCLK must be at least 2X the output SSP clock (USDSCLK)
- When in slave mode, the minimum frequency of the input SSPCLK must be at least 12X the input SSP clock (USDSCLK)

The maximum frequency of the SSPCLK should be:

- When in master mode, $F_{SSPCLK}(max) <= 254 * 128 * F_{USDSCLK}$ (SSP clock output)
- When in slave mode, $F_{SSPCLK}(max) <= 254 * 128 * F_{USDSCLK}$ (SSP clock input)

28.8.6 SSP Configuration

The SSP may be configured to support various features of master and slave based serial interfaces.

To enable the SSP controller, set **SSPDCON.SSPEN** to 1b. When disabled, the SSP may still have registers configured for the configuration mode and interrupts, but will not operate on serial data until enabled. Once the SSP has been configured, set this bit to 1b to begin processing.

To configure the SSP mode, set **SSPDCON.FRF** (frame format) to the desired value as shown below.

Table 28-7 SSP D Frame Format

SSPDCON.FRF	Frame Format
00b	SPI (Motorola)
01b	Synchronous Serial Format (TI)
10b	Microwire (National Semiconductor)
11b	Reserved

To select master mode, set **SSPDCON.MS** to 0b. In this mode, the controller is the bus master. It will drive the SCLK, MOSI and SS signals and will receive data on the MISO line.

To select slave mode, set **SSPDCON.MS** to 1b. In this mode, the controller is the bus slave and will drive the MISO line and receive input from SCLK, MOSI and SS.

To change the serial to parallel endian order, the user may use the **SSPDCON.LSBFIRST** field. To configure the SSP for LSB first, set **SSPDCON.LSBFIRST** to 1b. To configure the SSP for MSB first, set **SSPDCON.LSBFIRST** to a 0b.

To enable loop-back mode, set the **SSPDCON.LBM** to 1b. In this mode, serial input is taken from the serial output instead of from the digital peripheral MUX.

While in slave mode, the behavior of the MISO output can be controlled. When **SSPDCON.SOD** is set to 0b, the SSP can drive the MISO in slave mode. When this bit is 1b, the SSP will not drive the MISO output in slave mode.

The clock phase of the SSP may also be configured. To configure the SSP for capturing data on the first clock edge transition, set **SSPDCON.CPH** to 0b. To configure the SSP for capturing data on the second clock edge transition, set **SSPDCON.CPH** to 1b.

When the **SSPDCON.FRF** is 00b (SPI), the clock output polarity may be configured. To configure the SSP clock to be active high, set **SSPDCON.CPO** to 0b. To configure the SSP clock to be active low, set **SSPDCON.CPO** to 1b.

The SSP may also configure the size of the data word size transferred in each frame. To configure the data size, see the table below.

Table 28-8 SSP C Data Size Select

SSPDCON.DSS	Data Size
0 0000b	Reserved
0 0001b	Reserved
0 0010b	Reserved
0 0011b	4-bit
0 0100b	5-bit
0 0101b	6-bit
0 0110b	7-bit
0 0111b	8-bit
0 1000b	9-bit
0 1001b	10-bit
0 1010b	11-bit
0 1011b	12-bit
0 1100b	13-bit
0 1101b	14-bit
0 1110b	15-bit
0 1111b	16-bit
…	…
1 1111b	32-bit

28.8.7 SSP Slave Select Configuration

The SSP allows the behavior of the SS (slave select) signal to be configured by using the **SSPDSSCR** register as follows when configured as a bus master.

The **SSPDSSCR.SELCS** field must always be set to a 00b for proper operation.

To configure the SSP to automatically control the behavior of the SS signal, set **SSPDSSCR.SWSEL** to 0b. If the user wants to control the SS signal by software, set **SSPDSSCR.SWSEL** to 1b.

To configure the SS signal behavior after a frame transfer, use the **SSPDSSCR.SPHDONTCARE** field. To configure the SSP to not pull the SS signal high after a frame transfer, set **SSPDSSCR.SPHDONTCARE** to 0b. To configure the SSP to pull the SS signal high after a frame transfer, set the **SSPDSSCR.SPHDONTCARE** to 1b.

28.8.8 SSP Interrupts

The SSP may configure interrupts for the following conditions.

When the SSP detects that the TX FIFO is half-full or less, the **SSPDRIS.TXRIS** and **SSPDMIS.TXMIS** are set to 1b. During this condition, if the **SSPDIMSC.TXIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTD signal to the NVIC.

When the SSP detects that the RX FIFO is half-full or more, the **SSPDRIS.RXRIS** and **SSPDMIS.RXMIS** are set to 1b. During this condition, if the **SSPDIMSC.RXIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTD signal to the NVIC.

The SSP contains a time for reading the RX FIFO. If the RX FIFO is not empty, and 32 F_{SSPCLK} periods have gone by, the **SSPDRIS.RTIM** and **SSPDMIS.RTMIS** bits are set to 1b. During this condition, if the **SSPDIMSC.RTIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTD signal to the NVIC. This condition may be cleared by writing **SSPDICLR.RTIC** to 1b.

If the SSP detects the RX FIFO is full, when a character is attempted to be inserted into it, is sets the **SSPDRIS.ROIM** and **SSPDMIS.ROMIS** to 1b. During this condition, if the **SSPDIMSC.ROIM** interrupt enable is set to 1b, then the SSP will assert the IRQ_USARTD signal to the NVIC. This condition may be cleared by writing **SSPDICLR.ROIC** to 1b.

28.8.9 Master Mode Operation

When configured as a bus master (**SSPDCON.MS** = 0b), the SSP will insert a character into the TX FIFO when it is written into the **SSPDDAT** register. The SSP will read a character from the TX FIFO and performs a parallel to serial conversion on it. Then the serial data stream and frame control signal is synchronized to the clock and are output through the USDMOSI/TX pin to the attached slaves. While the data is being transmit to the slaves, the USDSCLK and USDSS behave as configured in the **SSPDCON** and **SSPDCSCR** registers.

The master receive logic performs serial to parallel conversion on the incoming synchronous USDMISO/RX data stream, and stores the character in the RX FIFO. The master may read first entry in the RX FIFO by reading the **SSPDDAT** register.

28.8.10 Slave Mode Operation

When configured as a bus slave (**SSPDCON.MS** = 1b), the SCLK is provided by the attached master. The user can write the next character to send into **SSPDDAT** and it will get inserted into the TX FIFO. The slave transmit logic reads a value from the TX FIFO, performs parallel to serial conversion and outputs the serial data stream on the USDMISO/RX pin to the attached master. The slave receive logic performs serial to parallel conversion on the incoming USDMOSI/TX data stream, extracting and storing values into the RX FIFO. The first character in the RX FIFO may be read by reading the **SSPDDAT** register.

28.8.11 SSP Status

The SSP module maintains status information on the operation of the module. The table below shows the available SSP status conditions.

Table 28-9 SSP D Status

SSPDSTAT field	Description
BSY	If set, the SSP controller is currently sending/receiving a frame and/or the TX FIFO is not empty.
RFF	If set, the RX FIFO is full.
RNE	If set, the RX FIFO is not empty.
TNF	If set, the TX FIFO is not full.
TFE	If set, the TX FIFO is empty

28.9 SSP Register Summary

Table 28-10 USART D Register Summary (SSP Mode)

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
SSPDCON	4005 0000h	SSP D Control Register	RW	0000 0000h
SSPDSTAT	4005 0004h	SSP D Status Register	RO	0000 0003h
SSPDDAT	4005 0008h	SSP D Data Register	RW	0000 0000h
SSPDCLK	4005 000Ch	SSP D Clock Control Register	RW	0000 0000h
SSPDIMSC	4005 0010h	SSP D Interrupt Mask Set and Clear Register	RW	0000 0000h
SSPDRIS	4005 0014h	SSP D Raw Interrupt Status Register	RO	0000 0008h
SSPDMIS	4005 0018h	SSP D Masked Interrupt Status Register	RO	0000 0008h
SSPDICLR	4005 001Ch	SSP D Interrupt Clear Register	RW	0000 0000h
SSPDSSCR	4005 0028h	SSP D Slave Select Configuration Register	RW	0000 0000h

28.10 SSP Register Detail

28.10.1 SSPDCON

Register 28-11 SSPDCON (SSP D Control Register, 4005 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:14	**Reserved**	RO	0	Reserved
13	**LSBFIRST**	RW	0	Endian Order: 0b: MSB transmit / receive first 1b: LSB transmit / receive first
12	**LBM**	RW	0	Loopback Mode: 0b: Normal operation 1b: Loopback mode. Serial Input is taken from the serial output (MOSI or MISO) rather than the serial in (MISO or MOSI respectively)
11	**SSPEN**	RW	0	SSP Enable: 0b: SSP Controller is disabled 1b: SSP Controller is enabled. It will interact with other devices on the serial bus. Software should write the appropriate control information to the other SPI/SSP registers and interrupt controller registers, before setting this bit.
10	**MS**	RW	0	Master/Slave Mode: 0b: The SSP controller acts as a bus master, driving the USDSCLK, USDMOSI and USDSS signals 1b: The SSP controller acts as a bus slave, driving the USDMISO and receiving USDSCLK, USDMOSI and USDSS.
9	**SOD**	RW	0	Slave Output Disable: 0b: The SSP can drive the USDMISO output in slave mode 1b: The SSP must not drive the USDMISO output in slave mode
8	**CPH**	RW	0	Clock Out Phase. This bit is only used when **SSPDCON.FRF** = 0b (SPI). 0b: The SSP controller captures serial data on the first edge transition of the frame. 1b: The SSP controller captures serial data on the second edge transition of the frame.
7	**CPO**	RW	0	Clock Out Polarity (this is only used in SPI mode when **SSDCON.FRF** = 00b) 0b: The clock is active high. 1b: The clock is active low.
6:5	**FRF**	RW	0	Frame Format: 00b: SPI 01b: TI 10b: Microwire 11b: Reserved
4:0	**DSS**	RW	0	Data Size Select: 0 0000b: Reserved 0 0001b: Reserved 0 0010b: Reserved 0 0011b: 4-bit transfer 0 0100b: 5-bit transfer

				0 0101b: 6-bit transfer 0 0110b: 7-bit transfer 0 0111b: 8-bit transfer 0 1000b: 9-bit transfer 0 1001b: 10-bit transfer 0 1010b: 11-bit transfer 0 1011b: 12-bit transfer 0 1100b: 13-bit transfer 0 1101b: 14-bit transfer 0 1110b: 15-bit transfer 0 1111b: 16-bit transfer … 1 1111b: 32-bit transfer

28.10.2 SSPDSTAT

Register 28-12 SSPDSTAT (SSP D Status Register, 4005 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**BSY**	RO	0	Busy Bit: 0b: SPI controller is idle 1b: SPI controller is sending/receiving a frame and/or the TX FIFO is not empty
3	**RFF**	RO	0	Receive FIFO Full: 0b: RX FIFO not full 1b: RX FIFO full
2	**RNE**	RO	0	Receive FIFO not empty: 0b: RX FIFO empty 1b: RX FIFO not empty
1	**TNF**	RO	0	Transmit FIFO not full: 0b: TX FIFO full 1b: TX FIFO not full
0	**TFE**	RO	0	Transmit FIFO empty: 0b: TX FIFO not empty 1b: TX FIFO empty

28.10.3 SSPDDAT

Register 28-13 SSPDDAT (SSP D Data Register, 4005 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**DATA**	RW	0	Software can writing data to be sent in a future frame to this register when the **SSPDSTAT.TNF** bit is set to 1b, indicating that the TX FIFO was previously empty and the SPI controller is not busy on the bus, transmission of the data will begin immediately. Otherwise, the data written to this register will be sent as soon as all previous data has been sent (and received). If the data length is less than 32-bits, the data must be right-justified in this register. Software may read data from this register when the **SSPDSTAT.RNE** bit is set to a 1b, indicating that the RX FIFO is not empty. When software reads this register, the SPI controller returns the data from the least recent frame in the RX FIFO. If the data length is less than 32-bits, the data is right-justified in this field with the MSBs set to 0.

28.10.4 SSPDCLK

Register 28-14 SSPDCLK (SSP D Clock Register, 4005 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:16	**Reserved**	RO	0	Reserved
15:8	**M**	RW	0	These fields may set the SPI master clock rate by the formula: F_{SSPCLK} = PCLK / ((**SSPDCLK.M** + 1) * **SSPDCLK.N**) N must be an even value from 2 to 254.
7:0	**N**	RW	0	

28.10.5 SSPDIMSC

Register 28-15 SSPDIMSC (SSP D Interrupt Mask Set and Clear Enable Register, 4005 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXIM**	RW	0	Transmit FIFO Interrupt Mask: 0b: TX FIFO half-full or less condition interrupt disabled 1b: TX FIFO half-full or less condition interrupt enabled
2	**RXIM**	RW	0	Receive FIFO Interrupt Mask: 0b: RX FIFO half-full or more condition interrupt disabled 1b: RX FIFO half-full or more condition interrupt enabled
1	**RTIM**	RW	0	Receive Timeout Interrupt Mask: 0b: RX FIFO not empty and no read prior to timeout period interrupt disabled 1b: RX FIFO not empty and no read prior to timeout period interrupt enabled
0	**ROIM**	RW	0	Receive Overrun Interrupt Mask: 0b: RX FIFO written to while full condition interrupt disabled 1b: RX FIFO written to while full condition interrupt enabled

28.10.6 SSPDRIS

Register 28-16 SSPDRIS (SSP D Raw Interrupt Status Register, 4005 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXRIS**	RO	1	TX FIFO Raw Interrupt Status: 0b: TX FIFO not at least half-full 1b: TX FIFO at least half-full
2	**RXRIS**	RO	0	RX FIFO Raw Interrupt Status: 0b: RX FIFO not at least half-full 1b: RX FIFO at least half-full
1	**RTRIS**	RO	0	RX FIFO Timeout Interrupt Status: 1b: RX FIFO is not empty and has not been read for the timeout period. The timeout period is 32-bit times at F_{SSPCLK}
0	**RORIS**	RO	0	RX FIFO Overrun Interrupt Status: 1b: RX FIFO was full when another frame was completely received. The preceding data frame is overrun by the new frame when this occurs.

28.10.7 SSPDMIS

Register 28-17 SSPDMIS (SSP D Masked Interrupt Status Register, 4005 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:4	**Reserved**	RO	0	Reserved
3	**TXMIS**	RO	1	TX FIFO Masked Interrupt Status: 0b: TX FIFO not at least half-full 1b: TX FIFO at least half-full
2	**RXMIS**	RO	0	RX FIFO Masked Interrupt Status: 0b: RX FIFO not at least half-full 1b: RX FIFO at least half-full
1	**RTMIS**	RO	0	RX FIFO Timeout Interrupt Status: 1b: RX FIFO is not empty and has not been read for the timeout period. The timeout period is 32-bit times at F_{SSPCLK}
0	**ROMIS**	RO	0	RX FIFO Overrun Interrupt Status: 1b: RX FIFO was full when another frame was completely received. The preceding data frame is overrun by the new frame when this occurs.

28.10.8 SSPDICLR

Register 28-18 SSPDICLR (SSP D Interrupt Clear Register, 4005 001Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:2	**Reserved**	RO	0	Reserved
1	**RTIC**	RW	0	RX FIFO Timeout Interrupt Clear: 0b: No effect 1b: Clears the **SSPDxIS.RTRIS** interrupt flag This bit is self-clearing.
0	**ROIC**	RW	0	RX FIFO Overwrite Interrupt Clear: 0b: No effect 1b: Clears the **SSPDxIS.RORIS** interrupt flag This bit is self-clearing.

28.10.9 SSPDSSCR

Register 28-19 SSPDSSCR (SSP D Slave Select Configuration Register, 4005 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:5	**Reserved**	RO	0	Reserved
4	**SPHDONTCARE**	RW	0	Slave Select Pull High: 0b: USDSS cannot pull high after frame transfer 1b: USDSS must pull high after frame transfer
3	**SWSS**	RW	0	Slave Select State: 0b: Set USDSS to low 1b: Set USDSS to high
2	**SWSEL**	RW	0	Slave Select Software Control: 0b: USDSS is automatically controlled by the SPI module 1b: USDSS is software controlled by **SSPDSSCR.SWSS**
1:0	**SELSS**	RW	0	Slave Select Signal Control: 00b: USDSS is enabled 01b: Reserved 10b: Reserved 11b: Reserved

29 I2C

29.1 Overview

The PAC55XX contains one I2C peripheral that supports version 2.1 of the I2C specification.

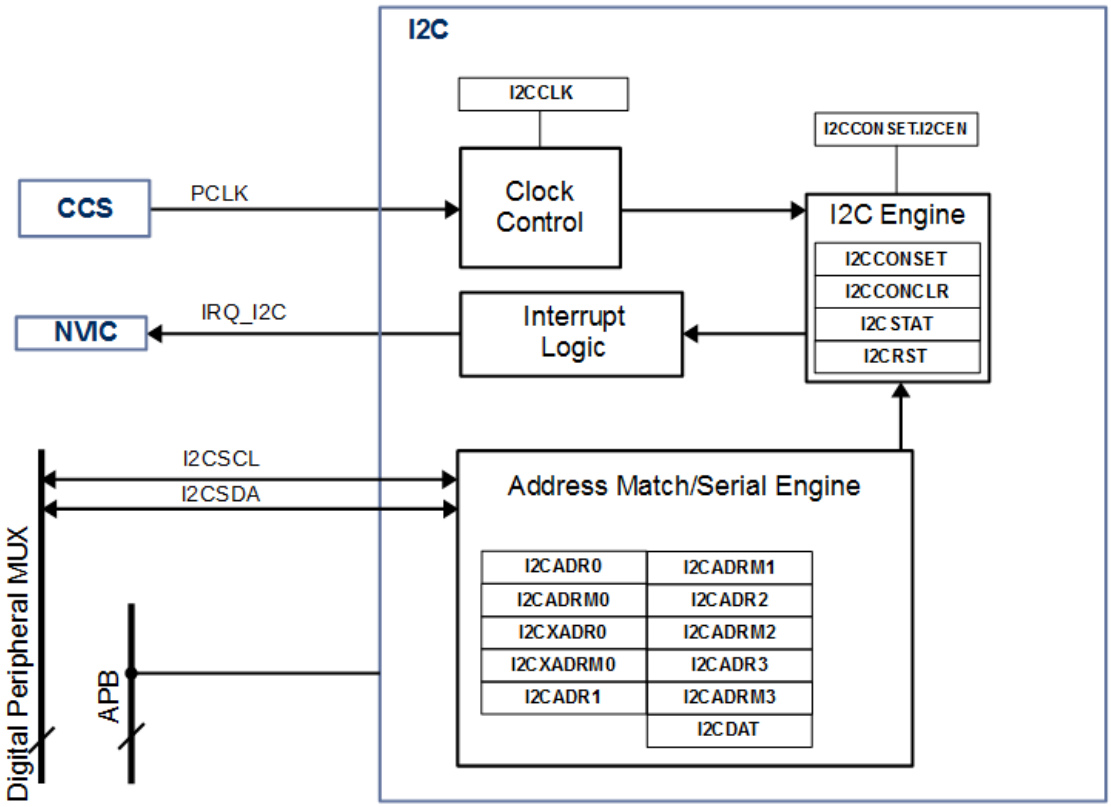
The I2C module is an APB client, and is clocked by the PCLK (peripheral clock) and can operate as a 7-bit or 10-bit master or slave controller.

29.2 Features

- I2C Master or slave peripheral
- 7-bit or 10-bit addressing modes
- Support High-speed, fast or standard speed transfer rates (3.4Mbps, 400kHz, 100kHz)
- Support for General Call Address detection
- Interrupt for transmit and receive data

29.3 System Block Diagram

Figure 29-1 I2C System Block Diagram



29.4 Functional Description

29.4.1 Enabling I2C

To enable the I2C module, set the **I2CCONSET.I2CEN** field to 1b. The module may be disabled by setting this field to a 0b. In addition, the user may set the **I2CCONCLR.I2CENC** bit to 1b to clear **I2CCONSET.I2CEN**. Writing a 0b to **I2CCONCLR.I2CENC** has no effect.

29.4.2 I2C Clocking

The I2C engine has its clock supplied from the peripheral clock (PCLK).

The I2C sampling frequency (FSAMP) and the SCL clock period (FSCL) may be configured using the I2CCLK register. These frequencies can be set as shown below:

- F_{SAMP} = PCLK / ($2^{\mathbf{I2CCLK.M}}$)
- F_{SCL} = PCLK / ($2^{\mathbf{I2CCLK.M}} \times (\mathbf{I2CCLK.N} + 1) \times 10$)

The table below shows some common configurations that can be used to generate F_{SAMP} and F_{SCL}.

Table 29-1 I2C F_{SAMP} and F_{CLK} configuration examples

PCLK	I2CCLK.M	I2CCLK.N	F_{SAMP}	F_{SCL}
48MHz	2	2	12MHz	400kHz
80MHz	2	1	20MHz	1MHz
136MHz	2	0	34MHz	3.4MHz

29.4.3 I2C Master Write

To perform an I2C Write transaction as an I2C master, follow these steps:

- Write **I2CCONSET.STA** to create an I2C START condition
  - The **I2CSTAT** register will contain a 08h if successful
- Write **I2CDAT** with the I2C target address:
  - Bit 0 should contain a 0b for a write operation
  - If configured, the I2C engine should generate an interrupt after writing the address
  - The **I2CSTAT** register will contain a 18h if the address is successfully written and an ACK is received
  - The **I2CSTAT** register will contain a 20h if the address is successfully written and a NACK is received
- For each data byte to write onto the bus:
  - Write the **I2CDAT** register with the 8-bit data
  - If configured, the I2C engine should generate an interrupt after writing the data
  - The **I2CSTAT** register will contain a 28h after the data is transmitted, and an ACK was received
  - The **I2CSTAT** register will contain a 30h after the data is transmitted, and a NACK was received
- Write **I2CCONSET.STO** to create an I2C STOP condition
  - Note that this bit must be set before clearing the I2C interrupt flag (**I2CCONSET.SI**)

29.4.4 I2C Master Read[23]

- Write **I2CCONSET.STA** to create an I2C START condition
  - The **I2CSTAT** register will contain a 08h if successful
- Write **I2CDAT** with the I2C target address:

[23] For a single data read, a repeated start condition is not required.

- Bit 0 should contain a 1b for a read operation
- If configured, the I2C engine should generate an interrupt after writing the address
- The **I2CSTAT** register will contain a 40h if the address is successfully written and an ACK is received
- The **I2CSTAT** register will contain a 48h if the address is successfully written and a NACK is received

- Write **I2CCONSET.STA** to create a repeated START condition
  - The I2CSTAT register will contain a 10h if the repeated START condition was successfully written
- Write **I2CDAT** with the I2C target address:
  - Bit 0 should contain a 1b for a read operation
  - If configured, the I2C engine should generate an interrupt after writing the address
  - The **I2CSTAT** register will contain a 40h if the address is successfully written and an ACK is received
  - The **I2CSTAT** register will contain a 48h if the address is successfully written and a NACK is received
- For each data byte to read from the bus:
  - If configured, wait for an I2C interrupt for a received data byte
  - Read the 8-bit data from the **I2CDAT** register
  - The **I2CSTAT** register will contain a 50h after the data is read from the bus, and an ACK was received
  - The **I2CSTAT** register will contain a 58h after the data is read from the bus, and a NACK was received
- Write **I2CCONSET.STO** to create an I2C STOP condition
  - Note that this bit must be set before clearing the I2C interrupt flag (**I2CCONSET.SI**)

29.4.5 I2C Slave Write

- Receive I2C interrupt upon received I2C slave address:
  - **I2CSTAT** will be 60h if matches slave address and write bit received, and ACK transmitted
- For every I2C interrupt after data byte received (for address phase):
  - **I2CSTAT** will be set to 80h, if ACK transmitted
  - **I2CSTAT** will be set to 88h, if NACK transmitted
- Set **I2CCONSET.AA** to 0b to NACK or 1b to ACK address
- **I2CSTAT** will be set to A0h if a repeated START was received
- For every I2C interrupt for every data byte received after address:
  - I2CSTAT will be set to 80h, if ACK transmitted

- I2CSTAT will be set to 88h, if NACK transmitted

- Set **I2CCONSET.AA** to 0b to NACK or 1b to ACK data byte

29.4.6 I2C Slave Read

- Receive I2C interrupt upon received I2C slave address:
  - **I2CSTAT** will be A8h if matches slave address and read bit received, and ACK transmitted
- For every I2C interrupt after data byte received (for address phase):
  - **I2CSTAT** will be set to 80h, if ACK transmitted
  - **I2CSTAT** will be set to 88h, if NACK transmitted
- **I2CSTAT** will be set to A0h if a repeated START was received
- Write **I2CDAT** with data byte to write onto I2C bus
- For every I2C interrupt for every data byte received after address:
  - **I2CSTAT** will be set to 80h, if ACK transmitted
  - **I2CSTAT** will be set to 88h, if NACK transmitted

Set **I2CCONSET.AA** to 0b to NACK or 1b to ACK data byte

29.4.7 I2C Slave Addressing

The I2C engine allows up to 4 7-bit slave addresses and 1 10-bit slave address to be configured that may be used when the device is operating as a I2C slave. Each slave address register has a matching mask register that can be used to mask slave addresses with when checking for a match.

The table below shows which registers should be programmed to perform slave address matching for both 7-bit and 10-bit addressing modes.

Table 29-2 I2C Slave Address Matching

Slave Address Register	Slave Address Mask	7-bit Address	10-bit Address
I2CADR0	**I2CADRM0**	X	
I2CADR1	**I2CADRM1**	X	
I2CADR2	**I2CADRM2**	X	
I2CXADR0	**I2CXADRM0**		X

In each of the I2C slave address registers, **I2CxADRx.GC** is the general call enable bit, and can be set if the registers should be configured to use the general call address.

29.4.8 I2C Interrupts

The I2C module has interrupts that may be enabled or disabled by the user. To enable interrupts, set the **I2CCONSET.I2CIE** to a 1b. To disable interrupts, set this field to a 0b. The user may also set the **I2CCONCLR.I2CIEC** bit to 1b to clear the **I2CCONSET.I2CIE** bit. Writing 0b to **I2CCONCLR.I2CIEC** has no effect.

The I2C module will generate an interrupt condition after any address or data byte is sent or received. During any of these conditions, the **I2CCONSET.SI** interrupt flag will be set. If the **I2CSETCON.I2CIE** interrupt enable bit is set to a 1b during any of these conditions, the IRQ_I2C signal to the NVIC will be asserted.

To clear the interrupt condition, write **I2CCONCLR.SIC** bit to a 1b will clear the interrupt flag in **I2CCONSET.SI** which will de-assert the IRQ_I2C signal to the NVIC.

29.4.9 I2C Status

The status of the I2C engine is kept in the **I2CSTAT** register. The contents of this register will show the status of the engine. The status codes for this register are shown below.

Table 29-3 I2C Status Codes

I2CSTATUS Code	Description
00h	Bus error (master mode only)
08h	START condition transmitted
10h	Repeated START condition transmitted
18h	Address and Write bit transmitted, ACK received
20h	Address and Write bit transmitted, NACK received
28h	Data byte transmitted in master mode, ACK received
30h	Data byte transmitted in master mode, NACK received
38h	Arbitration lost in address or data byte
40h	Address and read bit transmitted, ACK received
48h	Address and read bit transmitted, NACK received
50h	Data byte received in master mode, ACK transmitted
58h	Data byte received in master mode, NACK transmitted
60h	Slave address and write bit received, ACK transmitted
68h	Arbitration lost in address as master, slave address and write bit received, ACK transmitted
70h	General call address received, ACK transmitted
78h	Arbitration lost in address as master, general call address received, ACK transmitted
80h	Data byte received after slave address received, ACK transmitted
88h	Data byte received after slave address received, NACK transmitted
90h	Data byte received after general call address received, ACK transmitted
A0h	STOP or repeated START condition received in slave mode
A8h	Slave address and read bit received, ACK transmitted
B0h	Arbitration lost in address as master, slave address and read bit received, ACK transmitted
B8h	Data byte transmitted in slave mode, ACK received
C0h	Data byte transmitted in slave mode, NACK received
C8h	Last byte transmitted in slave mode, ACK received
D0h	Last byte transmitted in slave mode, NACK received
E0h	Second address byte transmitted, ACK received
E8h	Second address byte transmitted, NACK received
F8h	No relevant status information

29.4.10 I2C Reset

The I2C module may have software reset applied to it. After a reset is commanded, all I2C state machines and registers will reset to their default values.

To execute a software reset of the module, write **I2CRST** with the value 0x07.

29.5 Register Summary

Table 29-4 I2C Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
I2CCONSET	4001 0000h	I2C Control Set Register	RW	0000 0000h
I2CCONCLR	4001 0004h	I2C Control Clear Register	WO	--
I2CSTAT	4001 0008h	I2C Status Register	RO	0000 00F8h
I2CDAT	4001 000Ch	I2C Data Register	RW	0000 0000h
I2CCLK	4001 0010h	I2C Clock Control Register	RW	0000 0000h
I2CADR0	4001 0014h	I2C Slave Address Register 0	RW	0000 0000h
I2CADRM0	4001 0018h	I2C Slave Address Mask Register 0	RW	0000 00FEh
I2CXADR0	4001 001C	I2C Extended Slave Address Register 0	RW	0000 0000h
I2CXADRM0	4001 0020h	I2C Extended Slave Address Mask Register 0	RW	0000 07FEh
I2CRST	4001 0024h	I2C Software Reset Register	RW	0000 0000h
I2CADR1	4001 0028h	I2C Slave Address Register 1	RW	0000 0000h
I2CADRM1	4001 002Ch	I2C Slave Address Mask Register 1	RW	0000 00FEh
I2CADR2	4001 0030h	I2C Slave Address Register 2	RW	0000 0000h
I2CADRM2	4001 0034h	I2C Slave Address Mask Register 2	RW	0000 00FEh
I2CADR3	4001 0038h	I2C Slave Address Register 3	RW	0000 0000h
I2CADRM3	4001 003Ch	I2C Slave Address Mask Register 3	RW	0000 00FEh

29.6 Register Detail

29.6.1 I2CCONSET

Register 29-1 I2CCONSET (I2C Control Set, 4001 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:9	**Reserved**	RO	0	Reserved
8	**GCF**	RO	0	General Call Flag: 0b: I2C general call address no match 1b: I2C general call address match
7	**I2CIE**	RW	0	Interrupt Enabled: 0b: I2C disabled 1b: I2C enabled
6	**I2CEN**	RW	0	I2C Enabled: 0b: I2C Disabled 1b: I2C Enabled
5	**STA**	W1C	0	START flag: 0b: No effect 1b: I2C enters master mode and sends START condition. This bit is self-clearing.
4	**STO**	W1C	0	STOP flag: 0b: No effect 1b: Send STOP condition (if in master mode) or behave like an I2C STOP sent (if in slave mode). This bit is self-clearing.
3	**SI**	RO	0	I2C Interrupt flag: 0b: no flag 1b: flag
2	**AA**	RW	0	Assert Acknowledge flag: 0b: NACK will be returned during the acknowledge clock pulse on the SCL line when data byte has been received 1b: ACK will be returned during the acknowledge clock pulse on the SCL line when: The address in the Slave Address Register has been received The General Call address has been received while the GC (general call) bit in the ADR register is set A data byte has been received while I2C is in master or slave mode
1	**XADRF**	RO	0	I2C Extended Slave Address Flag (10-bit addressing): 0b: No slave address match 1b: Slave address match with 10-bit address. This bit is cleared when new data is transmit/received
0	**ADRF**	RO	0	I2C Slave Address Flag: 0b: No slave address match 1b: Slave address match with 7-bit address. This bit is cleared when new data is transmit/received.

29.6.2 I2CCONCLR

Register 29-2 I2CCONCLR (I2C Control Clear, 4001 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7	**I2CIEC**	RW	0	I2C Interrupt Disable: 0b: No effect 1b: Clears **I2CCONSET.I2CIE**. This bit is self-clearing.
6	**I2CENC**	RW	0	I2C Interface Disable: 0b: No effect 1b: Clears **I2CCONSET.I2CEN.** This bit is self-clearing.
5	**STAC**	RW	0	START Flag Clear: 0b: No effect 1b: Clears **I2CCONSET.STA**. This bit is self-clearing.
4	**Reserved**	RO	0	Reserved
3	**SIC**	RW	0	I2C Interrupt Clear: 0b: No effect 1b: Clears **I2CCONSET.SI**. This bit is self-clearing. Clearing the **I2CCONSET.SI** bit will reset the **I2CSTATUS** register to 0xF8, so be sure not to writes this bit until ready for the I2C module to proceed.
2	**AAC**	RW	0	Assert Acknowledge Clear: 0b: No effect 1b: Clears **I2CCONSET.AA**. This bit is self-clearing.
1:0	**Reserved**	RO	0	Reserved

29.6.3 I2CSTAT

Register 29-3 I2CSTAT (I2C Status, 4001 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:3	**Status**	RO	1 1111b	Status code for I2C engine.
2:0	**Reserved**	RO	0	Reserved

29.6.4 I2CDAT

Register 29-4 I2CDAT (I2C Data, 4001 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**Data**	RW	0	Data values received or to be transmit

29.6.5 I2CCLK

Register 29-5 I2CCLK (I2C Clock, 4001 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:7	**Reserved**	RO	0	Reserved
6:4	**M**	RW	0	F_{SAMP} = PCLK / 2^M
3:0	**N**	RW	0	F_{SCL} = PCLK / (2^M x (N+1) x 10)

29.6.6 I2CADR0

Register 29-6 I2CADR0 (I2C Slave Address 0, 4001 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:1	**Address**	RW	0	The I2C device address for slave mode.
0	**GC**	RW	0	General Call Enable: 0b: General call disabled 1b: General call enabled

29.6.7 I2CADRM0

Register 29-7 I2CADRM0 (I2C Slave Address Mask 0, 4001 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:1	**Mask**	RW	111 1111b	Slave address mask bits. Bits that are set to 0 are don't care. Bits that are set to 1 should exactly match the address register. The mask register has on effect on comparison to the GC address.
0	**Reserved**	RO	0	Reserved

29.6.8 I2CXADR0

Register 29-8 I2CXADR0 (I2C Extended Slave Address 0, 4001 001Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:11	**Reserved**	RO	0	Reserved
10:1	**Address**	RW	0	The I2C device address for slave mode used for extended addressing.
0	**Reserved**	RO	0	Reserved

29.6.9 I2CXADRM0

Register 29-9 I2CXADRM0 (I2C Extended Slave Address Mask 0, 4001 0020h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:11	**Reserved**	RO	0	Reserved
10:1	**Mask**	RW	11 1111 1111b	Slave address mask bits. Bits that are set to 0 are don't care. Bits that are set to 1 should exactly match the address register. The mask register has no effect on comparison to the GC address.
0	**Reserved**	RO	0	Reserved

29.6.10 I2CRST

Register 29-10 I2CRST (I2C Software Reset, 4001 0024h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:0	**RST**	RW	0	I2C Software Reset. To reset, write this field to 0x07.

29.6.11 I2CADR1

Register 29-11 I2CADR1 (I2C Slave Address 1, 4001 0028h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:1	**Address**	RW	0	The I2C device address for slave mode.
0	**GC**	RW	0	General Call Enable: 0b: General call disabled 1b: General call enabled

29.6.12 I2CADRM1

Register 29-12 I2CADRM1 (I2C Slave Address Mask 1, 4001 002Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:1	**Mask**	RW	111 1111b	Slave address mask bits. Bits that are set to 0 are don't care. Bits that are set to 1 should exactly match the address register. The mask register has on effect on comparison to the GC address.
0	**Reserved**	RO	0	Reserved

29.6.13 I2CADR2

Register 29-13 I2CADR2 (I2C Slave Address 2, 4001 0030h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:1	**Address**	RW	0	The I2C device address for slave mode.
0	**GC**	RW	0	General Call Enable: 0b: General call disabled 1b: General call enabled

29.6.14 I2CADRM2

Register 29-14 I2CADRM2 (I2C Slave Address Mask 2, 4001 0034h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:1	**Mask**	RW	111 1111b	Slave address mask bits. Bits that are set to 0 are don't care. Bits that are set to 1 should exactly match the address register. The mask register has on effect on comparison to the GC address.
0	**Reserved**	RO	0	Reserved

29.6.15 I2CADR3

Register 29-15 I2CADR3 (I2C Slave Address 3, 4001 0038h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:1	**Address**	RW	0	The I2C device address for slave mode.
0	**GC**	RW	0	General Call Enable: 0b: General call disabled 1b: General call enabled

29.6.16 I2CADRM3

Register 29-16 I2CADRM3 (I2C Slave Address Mask 3, 4001 003Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:8	**Reserved**	RO	0	Reserved
7:1	**Mask**	RW	111 1111b	Slave address mask bits. Bits that are set to 0 are don't care. Bits that are set to 1 should exactly match the address register. The mask register has on effect on comparison to the GC address.

0	**Reserved**	RO	0	Reserved

30 CAN

The PAC55XX contains a Controller Area Network (CAN) peripheral that can be used as a serial communications interface.

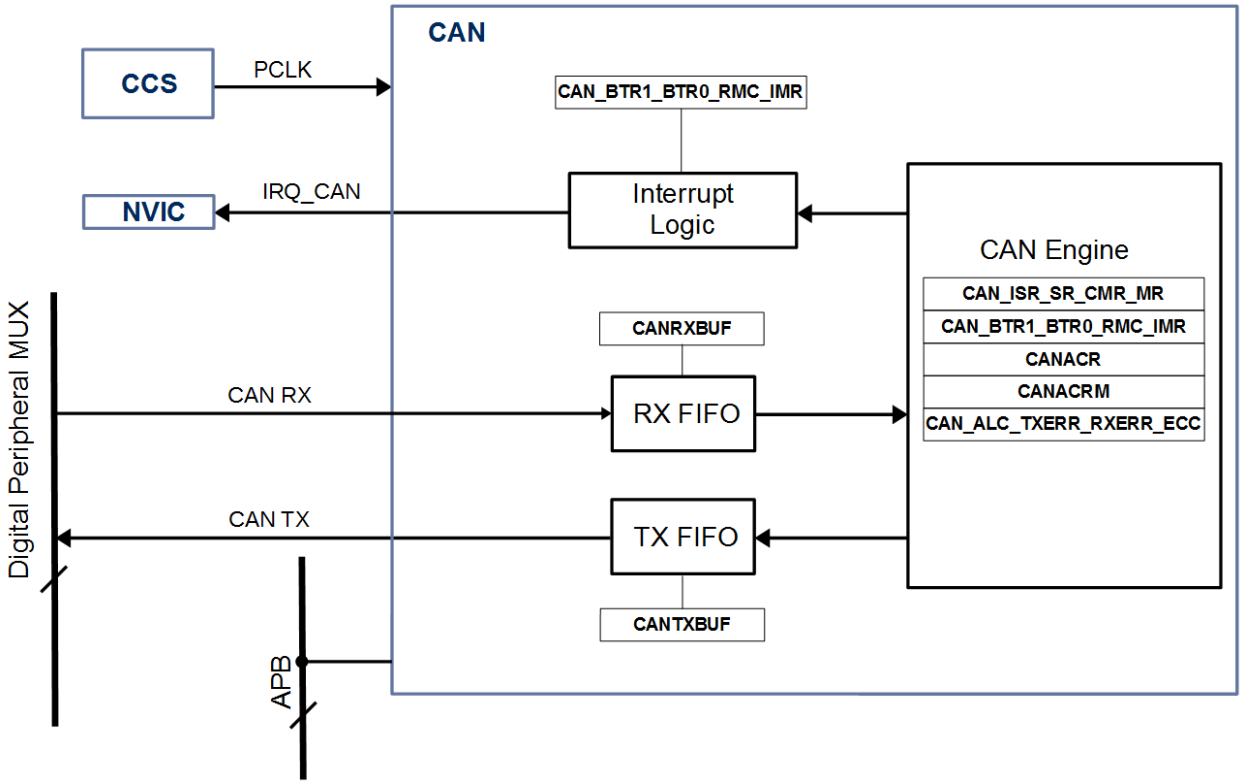
The CAN peripheral is an APB bus client and is clocked by the PCLK peripheral clock.

30.1 Features

- Conforms to Bosch CAN 2.0B Specification
  - 11 and 29-bit wide message identifiers
- Data rate up to 1Mbps
- Supports both CAN hard synchronization re-synchronization
- Hardware Message Filtering (dual/single filters)
- 64-Byte Receive FIFO
- 16-Byte Transmit FIFO
- Overload Frame generated on FIFO overflow
- Normal and Listen Only mode support
- Single Shot transmission
- Ability to abort transmission
- Readable error counters
- Last error code

30.2 System Block Diagram

Figure 30-1 CAN System Block Diagram



30.3 Overview

The Controller Area Network (CAN) is an advanced serial bus system that efficiently supports distributed control systems. It is an internationally standardized protocol by the International Standardization Organization (ISO) and Society for Automotive Engineers (SAE) for use in automotive and other applications.

The main CAN protocol features are listed below:

- Multi-master bus
- Number of nodes not limited by protocol
- Number of nodes may be changed dynamically without disturbing the communication of other nodes
- Supports broadcast and multi-cast addressing
- Sophisticated error-detection and handling mechanisms using CRC checking and other methods
- Non-return-to-zero (NRZ) coding
- Bit-stuffing for synchronization
- Up to 1Mbps data rate
- Carrier Sense Multiple Access/Collision Detection with arbitration

30.4 CAN Mode

The CAN controller supports three modes of operation:

- *Reset*
- *Normal*
- *Listen-Only*

To operate in *reset* mode, set the **CAMMR.RM** bit to 1b. In *reset* mode, frame reception and transmission is not possible. This mode is used to configure the CAN controller. After configuration is complete, the user may set the mode to *normal* or *listen-only* mode. After a MCU reset, the CAN controller is in *reset* mode.

To enter *listen-only* mode, the user must write **CAMMR.LOM** to 1b and **CAMMR.RM** while in *reset* mode. In *listen-only* mode, the CAN controller could give a no acknowledge to the CAN bus, even if a message is received successfully. If this occurs, the error counters are stopped at the current value. This mode is used mainline for automatic bit-rate detection without disturbing traffic on the network, or for design of a CAN bus analyzer.

To enter *normal* mode, the user must write **CAMMR.LOM** to 0b and **CAMMR.RM** to 0b while in *reset* mode.

After an MCU reset is performed, 11 consecutive recessive bits must be detected before the idle state may be reached.

30.5 Filtering Scheme

The user may configure the hardware acceptance filter scheme. The filtering scheme may be set to either single or dual filtering. In order to configure the filter scheme, the CAN controller must be in *reset* mode.

To set the CAN controller for single filtering, set the **CAMMR.AFM** to 0b.

To set the CAN controller for dual filtering, set the **CAMMR.AFM** bit to 1b.

30.6 Baud-Rate Pre-scaler

The user may use the **CANBTR0** register to set the Baud-Rate Pre-scaler (BRP) and synchronization jump width. This register may be read in any mode (normal or reset) but may be written only in reset mode.

The baud-rate pre-scaler value is defined by the **CANBTR0.BRP** field, and can be selected in a range of 1 to 64 by the equation:

- BRP = **CANBTR0.BRP** + 1

For example, if **CANBTR0.BRP** = 5 then the Baud-Rate Pre-Scaler would be x6.

The period of the CAN system clock t_{SCLK} is calculated by the following equation:

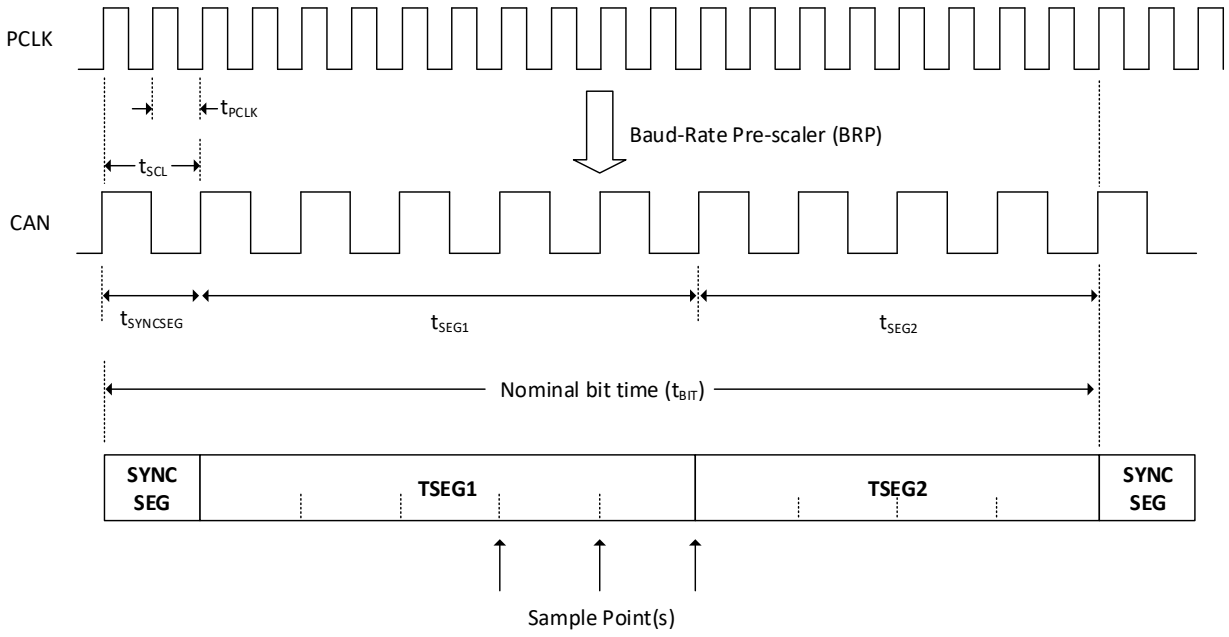
$t_{SCLK} = 2 \times t_{PCLK} \times (32 \times$ **CANBTR0.BRP** $+ 1)$, where $t_{PCLK} = 1 / f_{PCLK}$.

30.7 CAN Bus Timing

In the CAN peripheral, the user may configure the bit period, location of the sample point and number of samples taken at each sample point.

The diagram below shows the timing diagram of the CAN bit period.

Figure 30-2 CAN Bit Period Diagram



The nominal bit time consists of three non-overlapping segments with time durations shown below:

- SYNC_SEG - $t_{SYNC\_SEG} = t_{SCL}$
- TSEG1 – $t_{TSEG1} = t_{SCL} *$ (**CANBTR1.TSEG1** + 1)
- TSEG2 – $t_{TSEG2} = t_{SCL} *$ (**CANBTR1.TSEG2** + 1)

The total nominal bit time (t_{BIT}) is simply the sum of the segment durations:

- $t_{BIT} = t_{SYNC\_SEG} + t_{SEG1} + t_{SEG2}$

Each of the time segments above is specified as an integer number of the basic unit of time in a bit period, called the Time Quantum (TQ). The time duration of a Time Quantum is one period of the CAN system clock (t_{SCL}), which is derived from the PCLK (t_{PCLK}). The CAN system clock can be adjusted by the user via a programmable pre-scaler (Baud-Rate Pre-Scaler, BRP) according to the equation below.

- t_{SCL} = BRP * 2 * t_{PCLK} = 2 * BRP / F_{PCLK}

30.8 Synchronization Jump Width

To allow for compensation of phase shifts between time-bases on different controllers, the bit period must be shortened or lengthened accordingly. The Synchronization Jump Width defines the number of TQ by which a bit period may be lengthened or shortened in the event of a re-synchronization.

The maximum number of clock cycles in a bit period may be changed by the following equation:

- t_{SJW} = t_{SCLK} x (2 x **CANBTR0.SJW** + 1), where t_{SCLK} is as defined by the pre-scaler configuration above

30.9 CAN Message Formats

The CAN controller peripheral may send and receive data that are stored in TX FIFO or RX FIFO. Messages may be accessed via the **CANTXBUF** and **CANRXBUF** commands which are described below. When using the **CANTXBUF** and **CANRXBUF** registers, the data is written and read 32-bits at a time.

For data in the TX FIFO, the offset field corresponds directly to the head of the TX FIFO. For data in the RX FIFO, the offset field corresponds to the distance from the beginning of the message in the FIFO.

The following tables show the various message formats when present in the TX FIFO and RX FIFO.

30.9.1 Standard Frame Message Format

The CAN standard frame message format for the TX/RX FIFOs is shown below.

Table 30-1 Standard Frame Message Format

WORD OFFSET (32b)	BYTE OFFSET	DATA BITS	BUFFER CONTENT for TX/RX FIFO							
0	0x00	7:0	FF	RTR	X/0	X/0	DLC3	DLC2	DLC1	DLC0
	0x01	15:8	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
	0x02	23:16	ID2	ID1	ID0	X/RTR	X/0	X/0	X/0	X/0
	0x03	31:24	Data 1 (If DLC < 1, then unused)							
1	0x04	7:0	Data 2 (If DLC < 2, then unused)							
	0x05	15:8	Data 3 (If DLC < 3, then unused)							
	0x06	23:16	Data 4 (If DLC < 4, then unused)							
	0x07	31:24	Data 5 (If DLC < 5, then unused)							
2	0x08	7:0	Data 6 (If DLC < 6, then unused)							
	0x09	15:8	Data 7 (If DLC < 7, then unused)							
	0x0A	23:16	Data 8 (If DLC < 8, then unused)							
	0x0B	31:24	Unused							

The fields in this frame are defined as below:

- FF – Frame Format
  - 1: Extended frames
  - 0: Standard frames
- RTR – Remote Request Bit
  - 1: Remote frames
  - 0: Data frames
- X – Don’t care
- DLC – Data Length Code
- ID – CAN Message Identifier
- Data (1..8) – Data Bytes (7 – MSB, 0 – LSB)

30.9.1.1 Message Identifier (IDx)

The message identifier consists of 11 bits (ID10 to ID0). ID10 is the MSB, which is transmitted first on the bus during the arbitration process. This identifier acts as the message name. It is used in a receiver for acceptance filtering and to determine the bus access priority during the arbitration process. Lower identifier values indicate higher priority messages.

30.9.1.2 Remote Transmission Request (RTR)

When this bit is set, a remote frame is transmitted via the CAN bus. This means that no data bytes are included within this frame. But, it is necessary to specify the correct data length code which depends on the corresponding data frame with the same identifier coding. If the RTR bit is not set, a data frame will be sent including the number of data bytes as specified by the data length code.

30.9.1.3 Data Length Code (DLC)

The number of bytes in the data field of a message is encoded by the data length code. At the start of a remote frame transmission the data length code is not considered due to the RTR being a 1b (remote).

This forces the number of transmitted/received data bytes to be 0. But, the data length code must still be specified correctly to avoid bus errors if two CAN controllers start a remote frame transmission with the same identifier simultaneously. The range of the data byte count is 0 to 8 bytes and is coded as shown in the table below.

Table 30-2 Standard Frame DLC Encoding

NUMBER OF BYTES	DATA LENGTH CODE			
	DLC3	DLC2	DLC1	DLC0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0

Because of compatibility issues, no data length code greater than 8 may be used. If a selected value is greater than 8, the maximum number of 8 bytes is transmitted in the data frame with the data length code specified in DLC.

30.9.1.4 Data Field

The number of transferred data bytes is determined by the DLC. The first bit transmitted is the MSB of data byte 1. When the DLC is greater than 8, only 8 bytes of data are transmitted. When the RTR bit is set to 1, no data bytes are transmitted, regardless of DLC. ID10 is the first bit transmitted after a start of frame (SOF).

The received data length code located in the frame information byte represents the real sent data length code, which may be greater than 8. But, the real number of data bytes is 8. Users should note this when reading data from the RX FIFO.

30.9.2 Extended Frame Message Format

The CAN extended frame message format for the TX/RX FIFOs is shown below.

Table 30-3 Extended Frame Message Format

BYTE OFFSET	DATA BITS	BUFFER CONTENT for TX/RX FIFO							
0x00	7:0	FF	RTR	X/0	X/0	DLC3	DLC2	DLC1	DLC0
0x01	15:8	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
0x02	23:16	ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13
0x03	31:24	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5
0x04	7:0	ID4	ID3	ID2	ID1	ID0	X/RTR	X/0	X/0
0x05	15:8	Data 1 (If DLC < 1, then unused)							
0x06	23:16	Data 2 (If DLC < 2, then unused)							
0x07	31:24	Data 3 (If DLC < 3, then unused)							
0x08	7:0	Data 4 (If DLC < 4, then unused)							
0x09	15:8	Data 5 (If DLC < 5, then unused)							
0x0A	23:16	Data 6 (If DLC < 6, then unused)							
0x0B	31:24	Data 7 (If DLC < 7, then unused)							
0x0C	7:0	Data 8 (If DLC < 8, then unused)							
0x0D	15:8	Unused							
0x0E	23:16	Unused							
0x0F	31:24	Unused							

The fields in this frame are defined as below:

- FF – Frame Format
  - 1: Extended frames
  - 0: Standard frames
- RTR – Remote Request Bit
  - 1: Remote frames
  - 0: Data frames
- X – Don’t care
- DLC – Data Length Code
- ID – CAN Message Identifier
- Data (1..8) – Data Bytes (7 – MSB, 0 – LSB)

30.9.2.1 Message Identifier (IDx)

In the Extended Frame the message identifier is divided into two parts:

- Base identifier – 11 bits wide (ID28 – ID18)
- Extended identifier – 18 bits wide (ID17 – ID0)

The Base Identifier consists of 11 bits (ID28 to ID18) and is equivalent to the Standard Frame identifier. This identifies the frame's base priority.

30.9.2.2 Remote Transmission Request (RTR)

When this bit is set, a remote frame is transmitted via the CAN bus. This means that no data bytes are included within this frame. But, it is necessary to specify the correct data length code which depends on the corresponding data frame with the same identifier coding. If the RTR bit is not set, a data frame will be sent including the number of data bytes as specified by the data length code.

30.9.2.3 Data Length Code (DLC)

The number of bytes in the data field of a message is encoded by the data length code. At the start of a remote frame transmission the data length code is not considered due to the RTR being a 1b (remote).

This forces the number of transmitted/received data bytes to be 0. But, the data length code must still be specified correctly to avoid bus errors if two CAN controllers start a remote frame transmission with the same identifier simultaneously. The range of the data byte count is 0 to 8 bytes and is coded as shown in the table below.

Table 30-4 Standard Frame DLC Encoding

NUMBER OF BYTES	DATA LENGTH CODE			
	DLC3	DLC2	DLC1	DLC0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0

Because of compatibility issues, no data length code greater than 8 may be used. If a selected value is greater than 8, the maximum number of 8 bytes is transmitted in the data frame with the data length code specified in DLC.

30.9.2.4 Data Field

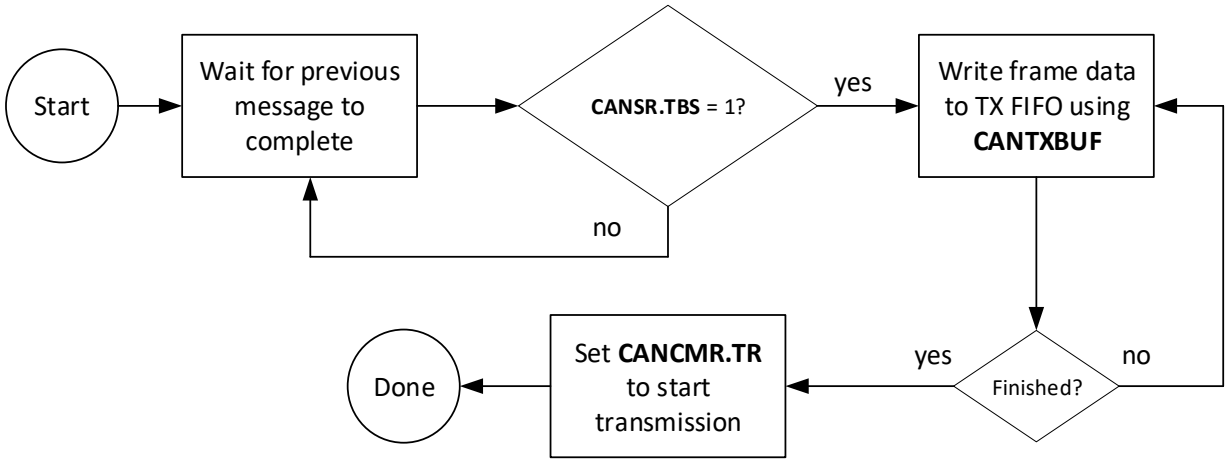
The number of transferred data bytes is determined by the DLC. The first bit transmitted is the MSB of data byte 1. When the DLC is greater than 8, only 8 bytes of data are transmitted. When the RTR bit is set to 1, no data bytes are transmitted, regardless of DLC. ID10 is the first bit transmitted after a start of frame (SOF).

The received data length code located in the frame information byte represents the real sent data length code, which may be greater than 8. But, the real number of data bytes is 8. Users should note this when reading data from the RX FIFO.

30.10 Transmitting Messages

The CAN controller transmits messages to the CAN bus from the TX FIFO. A simplified flow chart of the transmit process is shown below.

Figure 30-3 CAN Transmit Message Flow Chart



The CAN controller transmits messages from the TX FIFO. Before writing into the TX FIFO, the MCU must make sure the TX FIFO is not busy by making sure that the **CANSR.TBS** bit is set to 1b (Transmit buffer released for MCU).

The user may write the transmit message into the TX FIFO using the **CANTXBUF** register. Data can be written a word (32 bits) at a time into the tail of the TX FIFO until the entire message has been written. After the message is completely written into the TX FIFO, then CAN controller can be configured to transmit the message to the CAN bus.

30.10.1 Transmit Modes

To initiate transmission of a frame, set the **CANCMR.TR** to a 1b.

To initial a *single shot* transmission of a frame, set the **CANCMR.TR** and **CANCMR.AT** bits both to 1b. In a *single shot* transmission, in the case of a bus error or arbitration lost frame, a re-transmission is not performed.

To abort a frame transmission, the user may set **CANCMR.AT** to a 1b. A frame transmission may not be aborted by setting **CANCMR.TR** to a 0b in the middle of a message.

30.10.2 Transmit Error Counter

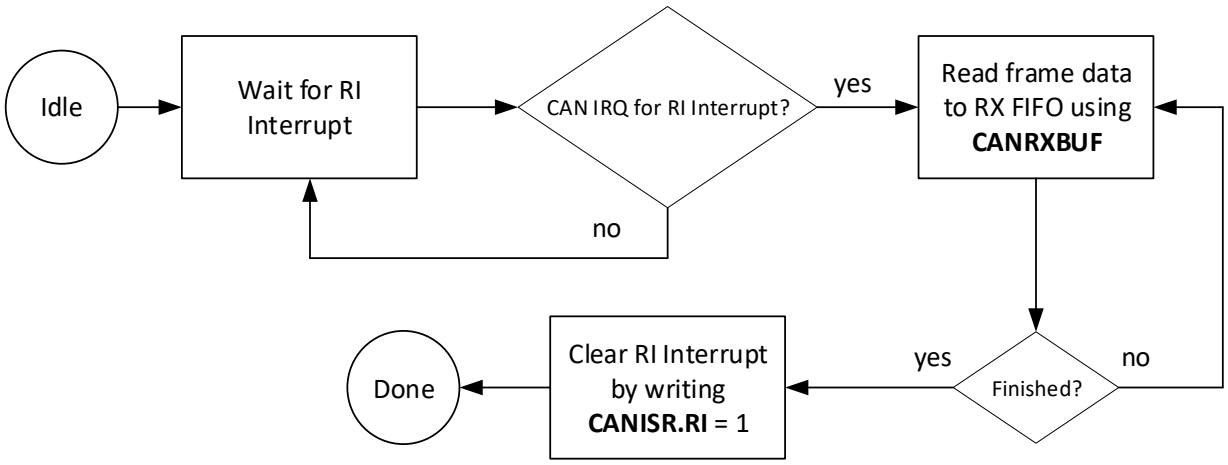
The **CANTXERR** register contains the read-only value of the transmit error counter. The transmit error counter is internally 9 bits, only 8 bits is shown in **CANTXERR**. If a bus off event occurs, this counter is initialized to 127 to count the minimum protocol defined time (128

occurrences of the bus free signal). Reading this register during this time gives information about the status of the bus off recovery.

30.11 Receiving Messages

The CAN controller receives messages to the CAN bus into the RX FIFO. A simplified flow chart of the receive process is shown below.

Figure 30-4 CAN Receive Message Flow Chart



The CAN controller receives messages from the RX FIFO. The CAN controller will interrupt the MCU using the NVIC with a Receive Interrupt (**CANISR.RI** = 1b).

After this interrupt is received, the user may read the message from the RX FIFO using the **CANRXBUF** register until the entire frame is read. The user is responsible for making sure to read the correct number of 32-bit words from the RX FIFO.

To clear the interrupt, the user may write **CANISR.RI** to 1b.

30.11.1 Bus Sampling

The CAN controller allows the user to configure the number of bus samples to take when receiving messages. The number of samples may be configured to be either 1 or 3.

If the number of bus samples is configured to be 1, the sample is taken at the end of the TSEG1 interval. To configure 1 bus sample, set **CANBTR1.SAM** to 0b. For higher speed CAN bus implementations, this mode is preferred.

If the number of bus samples is configured to be 3, the three samples are taken one TQ apart and just before the end of the TSEG1 interval. In this mode, the data is derived from the best 2 out of 3 samples. To configure 3 bus samples, set **CANBTR1.SAM** to 1b.

30.11.2 Receive Message Counter

The **CANRMC** is a read-only register that holds the current number of frames stored in the RX FIFO. The RX FIFO is able to store up to 21 messages. The following equation shows how to calculate the maximum number of messages stored in the RX FIFO:

- n = 64 / (3 + data length code)

This value is incremented on each successful frame reception and decremented when clearing the RI interrupt (writing **CANISR.RI** to 1b).

30.11.3 Receive Error Counter

The **RXERR** register contains the read-only value of the receive error counter. During a bus off event, this register is initialized to 0.

30.11.4 Acceptance Filter

The CAN controller can filter which messages on the CAN bus is passed to the RX FIFO, based on the identifier bits.

The acceptance filter allows the user to configure patterns of messages that may be received into the RX FIFO by using the Acceptance Code Register (**CANACR**) and the Acceptance Code Mask Register (**CANAMR**). The **CANACR** contain bit patterns of messages to be received while corresponding **CANAMR** defines which bit positions will be compared and which bit positions will not.

The filter configuration may be set to single or dual filter by setting the **CANMR.AFM** bit to 0b (dual) or 1b (single).

In the single filter configuration (**CANMR.AFM** = 1b), one long filter may be defined (32-bits). If a standard frame format is received, the complete identifier including the RTR bit and first two data bytes (if received) are used for acceptance filtering. Messages may also be accepted if there is no data byte. If only one data byte is received, then only bits up this data byte are compared with the filter. All single bit comparisons must signal acceptance for successful reception of the message.

Below are diagrams of the address matching using the single filter configuration for both standard and extended frames.

Figure 30-5 Acceptance Filter (single filter, standard frame)

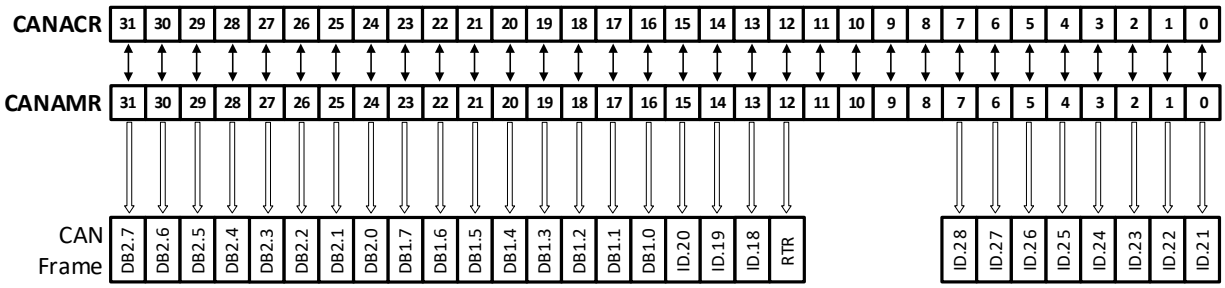
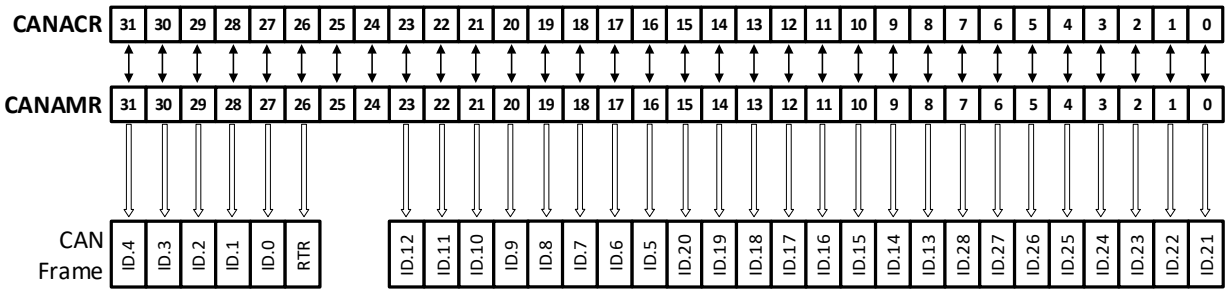


Figure 30-6 Acceptance Filter (single filter, extended frame)



In the dual filter configuration (**CANMR.AFM** = 0b), two short filters can be defined. The received message is compared to both filters to decide whether this message should be stored in the RX FIFO. If a standard frame message is received, the first filter compares the complete standard identifier including the RTR bit and the first data byte of the message. The second filter just compares the complete standard identifier including the RTR bit. For a message to be successfully received, a single bit comparison of at least one filter must signal the acceptance. If no data byte filtering is required for the first filter, **CANAMR[3:0]** must be set to 1111b.

Figure 30-7 Acceptance Filter (dual filter, standard frame)

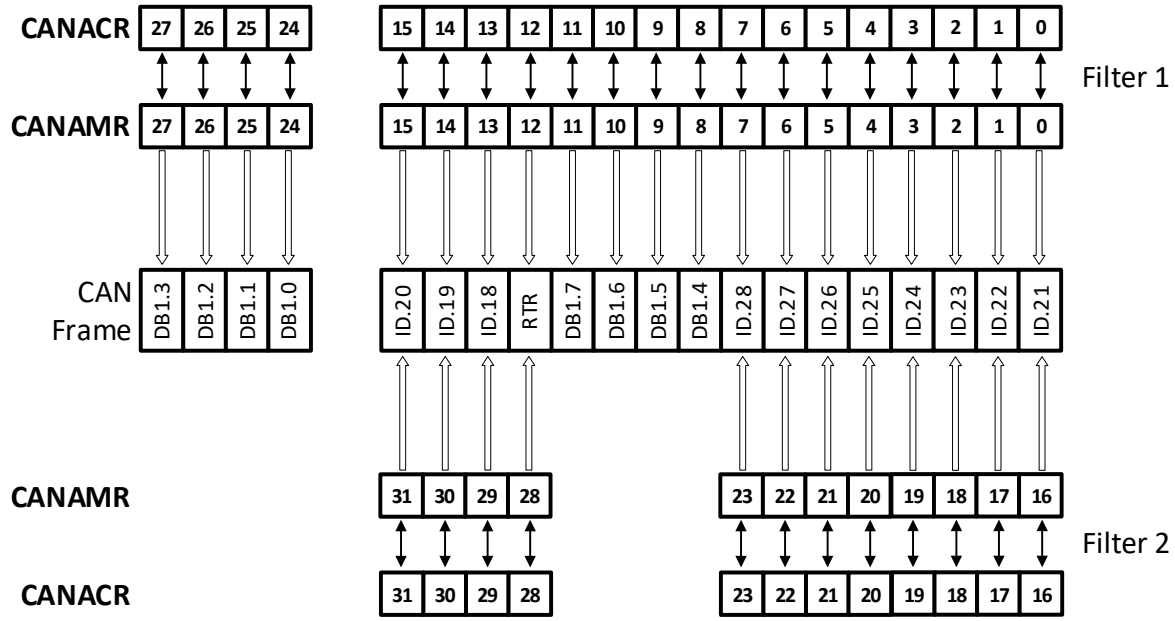
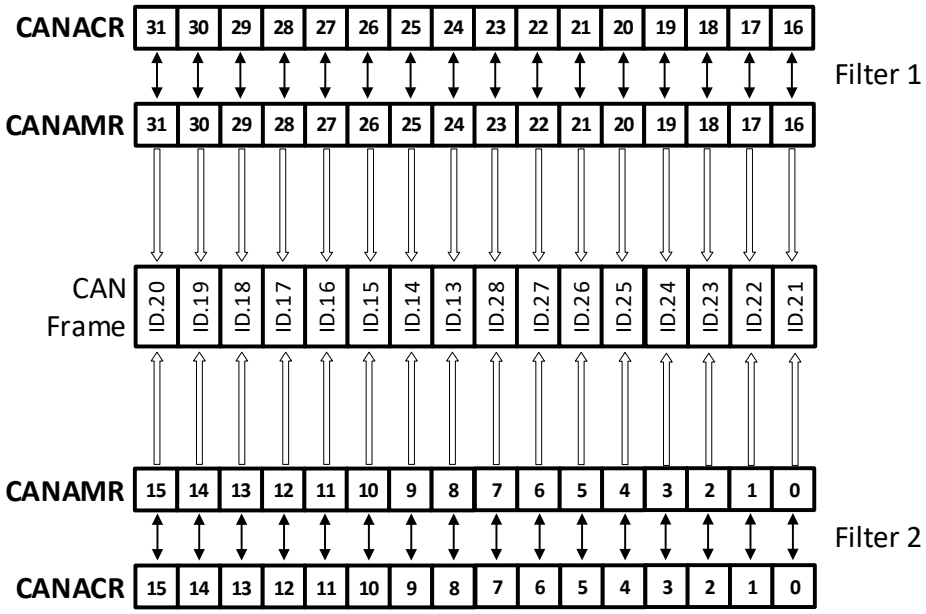


Figure 30-8 Acceptance Filter (dual filter, extended frame)



30.12 Interrupts

The CAN controller supports a series status conditions that can be polled or set to interrupt the MCU during operation.

The table below shows the set of conditions that may be polled, or have interrupts configured.

Table 30-5 CAN Controller Interrupts

CONDITION	STATUS BIT	INTERRUPT MASK	DESCRIPTION
Data Overrun	**CANISR.DOI**	**CANIMR.DOIM**	Set when RX FIFO overrun occurs.
Bus Error	**CANISR.BEI**	**CANIMR1.BEIM**	Set when CAN controller encounters a bus error while transmitting or receiving a message.
Transmission Interrupt	**CANISR.TI**	**CANIMR.TIM**	Set after a successful message transmission. This status bit must be cleared before the next message is written into the TX FIFO.
Receive Interrupt	**CANIS.RI**	**CANIMR.RIM**	Set when at least one message is in the RX FIFO. After reading a message from the RX FIFO using the **CANRXBUF** register, the CPU must clear this condition to decrement the RX Message Counter (this counter is not automatically decremented).
Error Passive Interrupt	**CANISR.EPI**	**CANIMR.EPIM**	Set when the CAN controller has reached or exited error passive level (on an active to passive or passive to active event).
Error Warning Interrupt	**CANISR.EWI**	**CANIMR.EWIM**	Set when there is a change in any of the error status or bus status bits. This can be used to detect if the CAN controller has entered or exited the bus off state.
Arbitration List Interrupt	**CANISR.ALI**	**CANIMR.ALIM**	Set when the CAN controller lost arbitration during transmission of its own message and became a receiver. The **CANALC** register may be read to check which bit in the frame arbitration was lost.

For each status condition, an interrupt may be asserted to the NVIC if the corresponding mask bit is set when the condition is true. For example, if there is a message in the RX FIFO and the **CANISR.RI** bit is set, if the **CANIMR.RIM** bit is also set then an interrupt to the NVIC will be asserted.

30.13 Error Handling

The CAN controller performs full error confinement as specified in the CAN 2.0B specification. The process of error handling is executed fully automatically by the hardware.

However, to provide the user with additional details about a certain error condition the CAN controller has an Error Code Capture function. When a CAN bus error is detected, the CAN controller forces the corresponding bus error interrupt. At the same time, the error code is stored in the **CANECC** register. The error bits are latched until read by the MCU.

The table below shows the different CAN bus errors, and which bits they are stored in.

Table 30-6 CAN Controller Interrupts

CONDITION	STATUS BIT	DESCRIPTION
RXWRN	**CANECC.RXWRN**	Set when the receive error counter (**CANRXERR**) is greater than or equal to 96.
TXWRN	**CANECC.TXWRN**	Set when the transmit error counter (**CANTXERR**) is greater than or equal to 96.
EDIR	**CANECC.EDIR**	The direction of data transfer while the error occurred: 1b: reception 0b: transmission
ACKER	**CANECC.ACKER**	Set when an ACK error occurred.
FRMER	**CANECC.FRMER**	Set when form error occurred.
CRCER	**CANECC.CRCER**	Set when CRC error occurred.
STFER	**CANECC.STFER**	Set when stuff error occurred.
BER	**CANECC.BER**	Set when bit error occurred.

The CAN controller can operate in one of three possible states, depending on the value of the error counter:

- Error Active
- Error Passive
- Bus Off

The CAN controller is in the *Error Active* state if both error counters (**CANRXERR** and **CANTXERR**) are both in the range 0 to 127. In this case, an error condition active error flag (6 dominant bits) is generated.

The CAN controller is in the *Error Passive* state if one of the error counters (**CANRXERR** or **CANTXERR**) is in the range of 128 to 255. A passive error flag (6 recessive bits) is generated upon detection of an error condition in this case.

If the transmit error counter (**CANTXERR**) is greater than 255, then the CAN controller is put into the *Bus Off* state. In this state, the reset request bit is set automatically and the CAN controller cannot influence the bus. When in this mode, the CAN controller can recover only by

the host controller command ‘Reset Request = 0’. This will start the bus off recovery procedure where the Transmit Error Counter (**CANTXERR**) is used to count 128 occurrences of a bus free signal. At the end of this time, both error counters (**CANRXERR** and **CANTXERR**) will be set to 0 and the device is put into the *Error Active* state again.

30.14 Arbitration Lost Capture

If arbitration is lost during transmission of a message, the **CANALC** register is updated with the identifier bit of the message when it was lost. This register is read-only and is not updated until the previous arbitration lost interrupt is acknowledged.

For example, when arbitration was lost at identifier bit 20 in the extended frame, the **CANALC** register will contain a value of 8 as shown in the diagram below.

Figure 30-9 Arbitration Lost Event

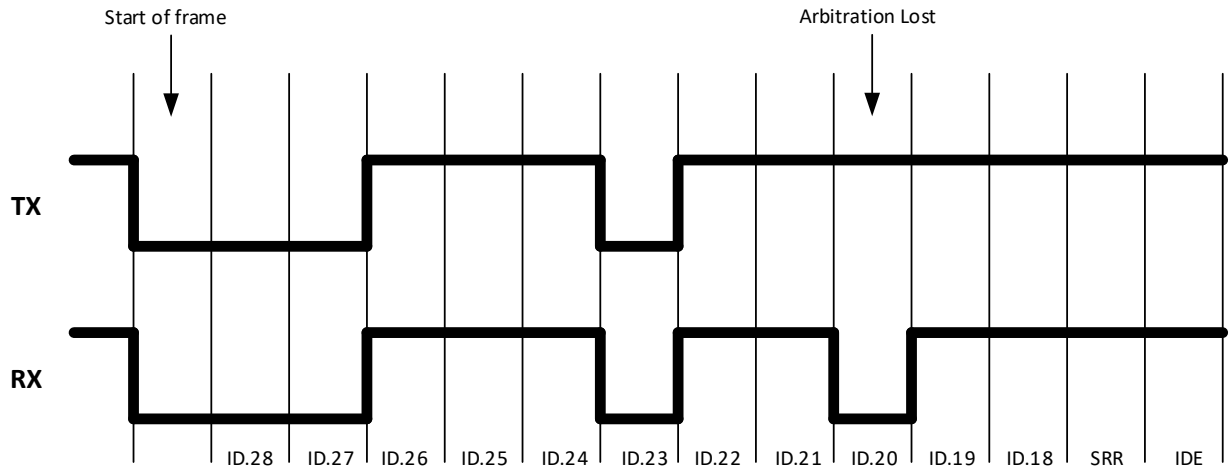
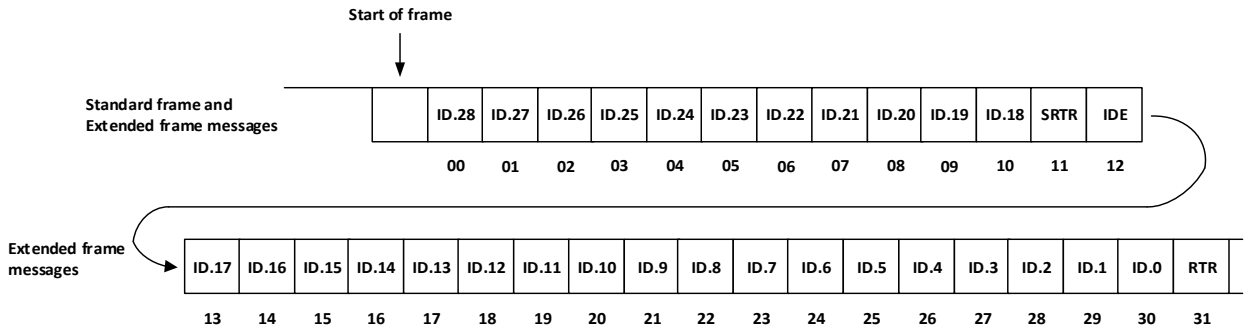


Figure 30-10 Arbitration Lost Bit-number Interpretation



The **CANALC** register field contains the bit at which the arbitration was lost according to the table below.

Table 30-7 Arbitration Lost Code

CANALC	DECIMAL VALUE	DESCRIPTION
0 0000b	0	Arbitration lost in **ID28 / 10**
0 0001b	1	Arbitration lost in **ID27 / 9**
0 0010b	2	Arbitration lost in **ID26 / 8**
0 0011b	3	Arbitration lost in **ID25 / 7**
0 0100b	4	Arbitration lost in **ID24 / 6**
0 0101b	5	Arbitration lost in **ID23 / 5**
0 0110b	6	Arbitration lost in **ID22 / 4**
0 0111b	7	Arbitration lost in **ID21 / 3**
0 1000b	8	Arbitration lost in **ID20 / 2**
0 1001b	9	Arbitration lost in **ID19 / 1**
0 1010b	10	Arbitration lost in **ID18 / 0**
0 1011b	11	Arbitration lost in **SRR / RTR**
0 1100b	12	Arbitration lost in **IDE** bit
0 1101b	13	Arbitration lost in **ID17**[24]
0 1110b	14	Arbitration lost in **ID16**[24]
0 1111b	15	Arbitration lost in **ID15**[24]
1 0000b	16	Arbitration lost in **ID14**[24]
1 0001b	17	Arbitration lost in **ID13**[24]
1 0010b	18	Arbitration lost in **ID12**[24]
1 0011b	19	Arbitration lost in **ID11**[24]
1 0100b	20	Arbitration lost in **ID10**[24]
1 0101b	21	Arbitration lost in **ID9**[24]
1 0110b	22	Arbitration lost in **ID8**[24]
1 0111b	23	Arbitration lost in **ID7**[24]
1 1000b	24	Arbitration lost in **ID6**[24]
1 1001b	25	Arbitration lost in **ID5**[24]
1 1010b	26	Arbitration lost in **ID4**[24]
1 1011b	27	Arbitration lost in **ID3**[24]
1 1100b	28	Arbitration lost in **ID2**[24]
1 1101b	29	Arbitration lost in **ID1**[24]
1 1110b	30	Arbitration lost in **ID0**[24]
1 1111b	31	Arbitration lost in **RTR**[24]

[24] Extended frame messages only

30.15 CAN Bus Status

The CAN controller provides a read-only status register that announces various conditions on the CAN bus. The table below shows the various bits in the CAN status register.

Table 30-8 CAN Register Summary

STATUS FIELD	STATUS	REGISTER	SUMMARY
BS	Bus off status	**CANSR.BS**	When this bit is set to 1b, this CAN node is in the bus off state and cannot receive or transmit frames. When this bit is cleared to 0b, this CAN node may receive and transmit frames. When the transmit error counter exceeds the limit of 255, the **CANSR.BS** bit is set to 1b (bus off), the **CANMR.RM** bit is set to 1b (reset mode) and if the bus error interrupt is enabled, an interrupt is asserted to the NVIC. The transmit error counter is then set to 127 and receive error counter is cleared. The CAN controller will then wait for 128 occurrences of the bus free signal (11 consecutive recessive bits) counting down the transmit error counter. When the interrupt is cleared (bus on), the error counters are reset and the error warning interrupt is generated, if enabled.
ES	Error status	**CANSR.ES**	When this bit is set to 1b, at least one of the CAN error counters reached the warning limit (96).
TS	Transmit status	**CANSR.TS**	When this bit is set to 1b, the CAN controller is transmitting a message.
RS	Receive status	**CANSR.RS**	When this bit is set to 1b, the CAN controller is receiving a message.
TBS	Transmit Buffer Status	**CANSR.TBS**	When this bit is set to 1b, the CAN controller will allow the MCU to write into the TX buffer. When this bit is set to 0b, the TX buffer is locked while the CAN controller is transmitting a message or the transmission is pending. If the MCU attempts to write into the transmit buffer while locked (bit set to 0b), the written data is not accepted.
DSO	Data overrun status	**CANSR.DSO**	When this bit is set to 1b, the RX FIFO has encountered an overrun. When this bit is set to 0b, no overrun has been encountered since the last clear data overrun command.
RBS	Receive buffer status	**CANSR.RBS**	When this bit is set to 1b, at least one message is in the RX FIFO. When this bit is set to 0b, no messages are in the FIFO.

30.16 Register Summary

Table 30-9 CAN Register Summary

REGISTER	ADDRESS	DESCRIPTION	ACCESS	RESET
CANMR	400A 0000h	CAN Mode	RW	04h
CANCMR	400A 0001h	CAN Command	RW	00h
CANSR	400A 0002h	CAN Status Register	RO	00h
CANISR	400A 0003h	CAN Interrupt Status/ACK Register	RW	00h
CANIMR	400A 0004h	CAN Interrupt Mask Register	RW	00h
CANRMC	400A 0005h	CAN Receive Message Counter	RO	00h
CANBTR0	400A 0006h	CAN Bus Timing 0 Register	RW	00h
CANBTR1	400A 0007h	CAN Bus Timing 1 Register	RW	00h
CANTXBUF	400A 0008h	CAN Transmit Buffer Register	RW	00000000h
CANRXBUF	400A 000Ch	CAN Receive Buffer Register	RO	00000000h
CANACR	400A 0010h	CAN Acceptance Code Register	RW	00000000h
CANAMR	400A 0014h	CAN Acceptance Mask Register	RW	00000000h
CANECC	400A 0018h	CAN Error Code Capture Register	RO	00h
CANRXERR	400A 0019h	CAN RX Error Counter Register	RO	00h
CANTXERR	400A 001Ah	CAN TX Error Counter Register	RO	00h
CANALC	400A 001Bh	CAN Arbitration Lost Code Capture Register	RO	00h

30.17 Register Detail

30.17.1 CANMR

Register 30-1 CANMR (CAN Mode Register, 400A 0000h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7:3	**Reserved**	RO	0	Reserved
2	**RM**	RW	1	Reset mode: 0b: inactive 1b: active
1	**LOM**	RW	0	Listen only mode. This register may only be written in reset mode, when **CANMR.RM** = 1b. 0b: inactive 1b: active
0	**AFM**	RW	0	Hardware acceptance filter scheme. This register may only be written in reset mode, when **CANMR.RM** = 1b. 0b: Dual filter 1b: Single filter

30.17.2 CANCMR

Register 30-2 CANCMR (CAN Command Register, 400A 0001h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7:3	**Reserved**	RO	0	Reserved
2	**TR**	RW	0	Transmit request: 0b: inactive 1b: active
1	**AT**	RW	0	Abort transmission: 0b: inactive 1b: active
0	**Reserved**	RO	0	Reserved

30.17.3 CANSR

Register 30-3 CANSR (CAN Status Register, 400A 0002h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7	**RBS**	RO	0	Receive Buffer Status
6	**DSO**	RO	0	Data Overrun Status
5	**TBS**	RO	0	Transmit Buffer Status
4	**Reserved**	RO	0	Reserved
3	**RS**	RO	0	Receive Status
2	**TS**	RO	0	Transmit Status
1	**ES**	RO	0	Error Status
0	**BS**	RO	0	Bus Off Status

30.17.4 CANISR

Register 30-4 CANISR (CAN Interrupt Status/ACK Register, 400A 0003h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7	**Reserved**	RO	0	Reserved
6	**ALI**	W1C	0	Arbitration Lost Interrupt: 0b: no interrupt 1b: interrupt
5	**EWI**	W1C	0	Error Warning Interrupt: 0b: no interrupt 1b: interrupt
4	**EPI**	W1C	0	Error Passive Interrupt: 0b: no interrupt 1b: interrupt
3	**RI**	W1C	0	Receive Interrupt: 0b: no interrupt 1b: interrupt
2	**TI**	W1C	0	Transmit Interrupt: 0b: no interrupt 1b: interrupt
1	**BEI**	W1C	0	Bus Error Interrupt: 0b: no interrupt 1b: interrupt
0	**DOI**	W1C	0	Data Overflow Interrupt: 0b: no interrupt 1b: interrupt

30.17.5 CANIMR

Register 30-5 CANIMR (CAN Interrupt Mask Register, 400A 0004h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7	**Reserved**	RO	0	Reserved
6	**ALIM**	RW	0	ALI Interrupt Mask: 0b: masked 1b: not masked
5	**EWIM**	RW	0	EWI Interrupt Mask: 0b: masked 1b: not masked
4	**EPIM**	RW	0	EPI Interrupt Mask: 0b: masked 1b: not masked
3	**RIM**	RW	0	RI Interrupt Mask: 0b: masked 1b: not masked
2	**TIM**	RW	0	TI Interrupt Mask: 0b: masked 1b: not masked
1	**BEIM**	RW	0	BEI Interrupt Mask: 0b: masked 1b: not masked
0	**DOIM**	RW	0	DOI Interrupt Mask: 0b: masked 1b: not masked

30.17.6 CANRMC

Register 30-6 CANRMC (CAN Receive Message Counter Register, 400A 0005h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7:5	**Reserved**	RO	0	Reserved
4:0	**RMC**	RO	0	Number of frames stored in the receive FIFO. Incremented on each successful frame reception and decremented by clearing RI interrupt.

30.17.7 CANBTR0

Register 30-7 CANBTR0 (CAN Bus Timing 0 Register, 400A 0006h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7:6	**SJW**	RW	0	Synchronization jump width.
5:0	**BRP**	RW	0	Baud Rate Pre-scaler.

30.17.8 CANBTR1

Register 30-8 CANBTR1 (CAN Bus Timing 1 Register, 400A 0007h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7	**SAM**	RW	0	Number of bus level samples: 0b: Bus is sampled once 1b: Bus is sampled three times
6:4	**TSEG2**	RW	0	Number of clock cycles per Time Segment 2.
3:0	**TSEG1**	RW	0	Number of clock cycles per Time Segment 1.

30.17.9 CANTXBUF

Register 30-9 CANTXBUF (CAN Transmit Buffer Register, 400A 0008h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**TXBUF**	RW	0	Transmit Buffer Data

30.17.10 CANRXBUF

Register 30-10 CANRXBUF (CAN Receive Buffer Register, 400A 000Ch)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**RXBUF0**	RO	0	Receive Buffer Data

30.17.11 CANACR

Register 30-11 CANACR (CAN Acceptance Code Register, 400A 0010h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**ACR**	RW	0	Acceptance Code

30.17.12 CANAMR

Register 30-12 CANAMR (CAN Acceptance Code Mask Register, 400A 0014h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
31:0	**AMR**	RW	0	Acceptance Code Mask

30.17.13 CANECC

Register 30-13 CANECC (CAN Error Code Capture Register, 400A 0018h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7	**RXWRN**	RO	0	Set when **CANRXERR** ≥ 96
6	**TXWRN**	RO	0	Set when **CANTXERR** ≥ 96
5	**EDIR**	RO	0	Direction of transfer while error occurred: 0b: transmission 1b: reception
4	**ACKER**	RO	0	ACK error occurred
3	**FRMER**	RO	0	Form error occurred
2	**CRCER**	RO	0	CRC error occurred
1	**STFER**	RO	0	Stuff error occurred
0	**BER**	RO	0	Bit error occurred

30.17.14 CANRXERR

Register 30-14 CANRXERR (CAN Receive Error Counter Register, 400A 0019h)

BIT	NAME	ACCESS	RESET	DESCRIPTION
15:8	**RXERR**	RO	0	Receive error counter

30.17.15 CANTXERR

Register 30-15 CANTXERR (CAN Transmit Error Counter Register, 400A 001Ah)

BIT	NAME	ACCESS	RESET	DESCRIPTION
15:8	**RXERR**	RO	0	Receive error counter

30.17.16 CANALC

Register 30-16 CANALC (CAN Arbitration Lost Code Capture Register, 400A 001Bh)

BIT	NAME	ACCESS	RESET	DESCRIPTION
7:5	**Reserved**	RO	0	Reserved
4:0	**CANALC**	RO	0	Arbitration Lost Bit Number: 00: Arbitration lost in ID28 / 10 01: Arbitration lost in ID28 / 9 02: Arbitration lost in ID28 / 8 03: Arbitration lost in ID28 / 7 04: Arbitration lost in ID28 / 6 05: Arbitration lost in ID28 / 5 06: Arbitration lost in ID28 / 4 07: Arbitration lost in ID28 / 3 08: Arbitration lost in ID28 / 2 09: Arbitration lost in ID28 / 1 10: Arbitration lost in ID28 / 0 11: Arbitration lost in SRR / RTR 12: Arbitration lost in IDE bit 13: Arbitration lost in ID17[25] 14: Arbitration lost in ID16[25] 15: Arbitration lost in ID15[25] 16: Arbitration lost in ID14[25] 17: Arbitration lost in ID13[25] 18: Arbitration lost in ID12[25] 19: Arbitration lost in ID11[25] 20: Arbitration lost in ID10[25] 21: Arbitration lost in ID9[25] 22: Arbitration lost in ID8[25] 23: Arbitration lost in ID7[25] 24: Arbitration lost in ID6[25] 25: Arbitration lost in ID5[25] 26: Arbitration lost in ID4[25] 27: Arbitration lost in ID3[25] 28: Arbitration lost in ID2[25] 29: Arbitration lost in ID1[25] 30: Arbitration lost in ID0[25] 31: Arbitration lost in RTR

[25] Extended frame messages only

31 ARM® CORTEX®-M4F REFERENCE

The PAC55XX controller contains an Arm® Cortex®-M4F MCU. The Arm® Cortex®-M4F has several configurable options. The options listed below are present on the PAC55XX family of controllers:

- IEEE754 single-precision Floating Point Unit
- Memory Protection Unit (MPU) included
- Number of interrupt priorities: 8 (3-bit priority)
- Wakeup Interrupt Controller (WIC) included
- Sleep Mode power-saving included
- Little Endian configuration
- 24-bit SysTick timer included
- Embedded Trace Module (ETM) included
    - Instruction trace only

ARM provides a full set of documentation for the Arm® Cortex®-M4F MCU.

You can retrieve the full set of documentation on the Arm® Cortex®-M4F from here:

http://infocenter.arm.com/help/index.jsp

The documents that are the most important are:

- Arm® Cortex®-M4 Processor Technical Reference Manual
- Coresight ETM-M4 Technical Reference Manual

The *Arm® Cortex®-M4 Processor Technical Reference Manual* contains documentation for the *Arm® Cortex®*-M4 processor, the programmer's model, instruction set, registers, memory map, floating point multimedia, trace and debug support.

The *Arm® Cortex®-M4 Devices Generic User Guide* contains documentation for the *Arm® Cortex®*-M4F's *Embedded Trace Macrocell™*

32 Revision History

Version	Date	Changes
v2.0	2025-07-06	- Corrected QEP MCU v1 and v2 description.
v2.1	2025-04-09	- Modified SSP (SPI) nomenclature to use “half-full” in all cases instead of “half-empty” - Corrected Schmitt trigger description to state GPIO Schmitt trigger control bit descriptions are located in the GPIO and Digital Peripheral Mux (DPM) section - Corrected DTSE sequence entry EMUXC field description to clarify when EMUX data is sent - Corrected SSP DSS (Data Size Select) field to indicate there are 32 settings up to 32-bit

33 Contact Information

For the latest specifications, additional product information, worldwide sales and distribution locations:

Web: www.qorvo.com
Tel: 1-844-890-8163
Email: customer.support@qorvo.com

34 Important Notice

The information contained herein is believed to be reliable; however, Qorvo makes no warranties regarding the information contained herein and assumes no responsibility or liability whatsoever for the use of the information contained herein. All information contained herein is subject to change without notice. Customers should obtain and verify the latest relevant information before placing orders for Qorvo products. The information contained herein or any use of such information does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other intellectual property rights, whether with regard to such information itself or anything described by such information. **THIS INFORMATION DOES NOT CONSTITUTE A WARRANTY WITH RESPECT TO THE PRODUCTS DESCRIBED HEREIN, AND QORVO HEREBY DISCLAIMS ANY AND ALL WARRANTIES WITH RESPECT TO SUCH PRODUCTS WHETHER EXPRESS OR IMPLIED BY LAW, COURSE OF DEALING, COURSE OF PERFORMANCE, USAGE OF TRADE OR OTHERWISE, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.**
Without limiting the generality of the foregoing, Qorvo products are not warranted or authorized for use as critical components in medical, life-saving, or life-sustaining applications, or other applications where a failure would reasonably be expected to cause severe personal injury or death.