Welcome to E-XFL.COM

Embedded - Microcontrollers - Application Specific: Tailored Solutions for Precision and Performance

Embedded - Microcontrollers - Application Specific represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering enhanced performance, efficiency, and functionality to meet the demands of specialized applications.

What Are Embedded - Microcontrollers - Application Specific?

Application-specific microcontrollers are engineered to

Details	
Product Status	Obsolete
Applications	Touchscreen Controller
Core Processor	M8C
Program Memory Type	FLASH (32kB)
Controller Series	CY8CT
RAM Size	2K x 8
Interface	I²C, SPI, UART/USART, USB
Number of I/O	28
Voltage - Supply	1.8V
Operating Temperature	-40°C ~ 85°C
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-QFN
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy8ctmg200a-32lqxit

Top Level Architecture

The PSoC block diagram on the next page illustrates the top-level architecture of the CY8CTMG20x and CY8CTST200 devices. Each major grouping in the diagram is covered in this manual in its own section: PSoC Core, TrueTouch System, and the System Resources. Banding these three main areas together is the communication network of the system ***bus***.

PSoC Core

The PSoC Core is a powerful engine that supports a rich instruction set. It encompasses the ***SRAM*** for data storage, an ***interrupt*** controller for easy program execution to new addresses, sleep and watchdog timers, a regulated 3.0V output option is provided for Port 1 I/Os, and multiple ***clock*** sources that include the IMO (internal main oscillator) and ILO (internal low speed oscillator) for precision, programmable clocking.

The CPU core, called the M8C, is a powerful processor with speeds up to 24 MHz. The M8C is a four MIPS 8-***bit*** Harvard architecture microprocessor. Within the CPU core are the ***SROM*** and ***Flash*** memory components that provide flexible programming.

PSoC GPIOs provide connection to the CPU and the TrueTouch resources of the device. Each pin's drive mode is selectable from four options, allowing great flexibility in external interfacing. Every pin also has the capability to generate a system interrupt on low level and change from last read.

TrueTouch™ System

The TrueTouch System is composed of comparators, reference drivers, I/O multiplexers, and digital logic to support various capacitive sensing algorithms. Various reference selections are provided. Digital logic is mainly comprised of counters and timers.

System Resources

The System Resources provide additional PSoCcapability. These system resources include:

- Digital clocks to increase the flexibility of the PSoC programmable system-on-chip.
- I2C functionality with "no bus stalling."
- Various system resets supported by the M8C.
- Power-On-Reset (POR) circuit protection.
- SPI master and slave functionality.
- A programmable timer to provide periodic interrupts.
- Clock boost network providing a stronger signal to switches.
- Full-speed USB interface for USB 2.0 communication with 512 bytes of dedicated buffer memory and an internal 3V regulator.

2.5.3 Three-Byte Instructions

The three-byte instruction formats are the second most prevalent instruction formats. These instructions need three bytes because they either move data between two addresses in the user accessible address space (registers and RAM) or they hold 16-bit absolute addresses as the destination of a long jump or long call.

Table 2-5. Three-Byte Instruction Formats

Byte 0	Byte 1	Byte 2
8-Bit Opcode	16-Bit Address (MSB, LSB)	
8-Bit Opcode	8-Bit Address	8-Bit Data
8-Bit Opcode	8-Bit Address	8-Bit Address

The first instruction format, shown in the first row of Table 2-5, is used by the `LJMP` and `LCALL` instructions. These instructions change program execution unconditionally to an absolute address. The instructions use an 8-bit opcode, leaving room for a 16-bit destination address.

The second three-byte instruction format, shown in the second row of Table 2-5, is used by the following two addressing modes:

- Destination Direct Source Immediate (`ADD [7], 5`)
- Destination Indexed Source Immediate (`ADD [X+7], 5`)

The third three-byte instruction format, shown in the third row of Table 2-5, is for the Destination Direct Source Direct addressing mode, which is used by only one instruction. This instruction format uses an 8-bit opcode followed by two 8-bit addresses. The first address is the destination address in RAM, while the second address is the source address in RAM. The following is an example of this instruction:

```
MOV [7], [5]
```

The following code example puts the correct value in KEY1 and KEY2. The code is preceded by a HALT, to force the program to jump directly into the setup code and not accidentally run into it.

```
1.        halt
2.        SSCOP: mov [KEY1], 3ah
3.        mov X, SP
4.        mov A, X
5.        add A, 3
6.        mov [KEY2], A
```

3.1.1 Additional SROM Feature

Return Codes: These aid in the determination of success or failure of a particular function. The return code is stored in KEY1's position in the parameter block. The Checksum and TableRead functions do not have return codes because KEY1's position in the parameter block is used to return other data.

Table 3-3. SROM Return Code Meanings

Return Code Value	Description
00h	Success.
01h	Function not allowed because of block level protection.
02h	Software reset without hardware reset.
03h	Fatal error, SROM halted.
04h	Write and Verify error.
06h	Failure of Smartwrite parameters CheckSum

Note Read, write, and erase operations may fail if the target block is read or write protected. Block protection levels are set during device programming and cannot be modified from code in the PSoC device.

3.1.2 SROM Function Descriptions

3.1.2.1 SWBootReset Function

The SROM function SWBootReset is responsible for transitioning the device from a reset state to running user code. See Chapter "System Resets" on page 135 for more information on what events causes the SWBootReset function to execute.

The SWBootReset function executes whenever the SROM is entered with an M8C accumulator value of 00h; the SRAM parameter block is not used as an input to the function. This happens, by design, after a hardware reset because the M8C's accumulator is reset to 00h or when user code executes the SSC instruction with an accumulator value of 00h.

If the checksum of the calibration data is valid, the SWBootReset function ends by setting the internal M8C registers (CPU_SP, CPU_PC, CPU_X, CPU_F, CPU_A) to 00h, writing 00h to most SRAM addresses in SRAM Page 0, and then begins to execute user code at address 0000h. (See Table 3-5 and the following paragraphs for more information on which SRAM addresses are modified.) If the checksum is not valid, an internal reset is executed and the boot process starts over. If this condition occurs, the internal reset status bit (IRESS) is set in the CPU_SCR1 register.

In devices with more than 256 bytes of SRAM, no SRAM is modified by the SWBootReset function in SRAM pages numbered higher than '0'.

Table 3-5 documents the value of all the SRAM addresses in Page 0 after a successful SWBootReset. A value of "xx" indicates that the SRAM address is not modified by the SWBootReset function. A hex value indicates that the address always has the indicated value after a successful SWBootReset. A "??" indicates that the value, after a SWBootReset, is determined by the value of the IRAMDIS bit in the CPU_SCR1 register. If IRAMDIS is not set, these addresses are initialized to 00h. If IRAMDIS is set, these addresses are not modified by a SWBootReset after a watchdog reset.

Note The T0, T1, and T2 mentioned in the SLP_CFG3 register with respect to Figure 10-2 on page 75 are defined as follows:

- T0: Time duration between T0 and T1 in the timing diagram.
- T1: Time duration between T1 and T2 in the timing diagram.
- T2: Time duration between T3 and T4 in the timing diagram.

10.1.2 Sleep Timer

The Sleep Timer is a 15-bit up counter clocked by the 32 kHz clock source. This timer is always enabled except in deep sleep mode. The exception to this is within an ***ICE*** (**in-circuit *emulator***) in ***debugger*** mode and when the Stop bit in the CPU_SCR0 is set; the sleep timer is disabled, so that the user does not get continual watchdog resets when a breakpoint is hit in the debugger environment.

If the associated sleep timer interrupt is enabled, a periodic interrupt to the CPU is generated based upon the sleep interval selected from the OSC_CR0 register. The sleep timer functionality does not need to directly associate with the sleep state. It can be used as a general purpose timer interrupt regardless of sleep state.

The reset state of the sleep timer is a count value of all zeros. There are two ways to reset the sleep timer. Any hardware reset, (that is, POR, XRES, or Watchdog Reset (WDR)) resets the sleep timer. There is also a method that allows the user to reset the sleep timer in firmware. A write of 38h to the RES_WDT register clears the sleep timer.

Note Any write to the RES_WDT register also clears the watchdog timer.

Clearing the sleep timer is done at anytime to synchronize the sleep timer operation to CPU processing. A good example of this is after POR. The CPU hold off, due to voltage ramp and others, may be significant. In addition, a significant amount of program initialization may be required. However, the sleep timer starts counting immediately after POR and is at an arbitrary count when user code begins execution. In this case, it is desirable to clear the sleep timer before enabling the sleep interrupt initially to ensure that the first sleep period is a full interval.

10.2 Application Overview

The following are notes regarding sleep related to firmware and application issues.

Note 1 If an interrupt is pending, enabled, and scheduled to be taken at the instruction boundary after the write to the SLEEP bit, the system does not go to sleep. The instruction still executes, but it cannot set the SLEEP bit in the CPU_SCR0 register. Instead, the interrupt is taken and the effect of the sleep instruction ignored.

Note 2 There is no need to enable the Global Interrupt Enable (CPU_F register) to wake the system out of sleep state. Individual interrupt enables, as set in the interrupt mask registers, are sufficient. If the Global Interrupt Enable is not set, the CPU does not service the ISR associated with that interrupt. However, the system wakes up and continues executing instructions from the point at which it went to sleep. In this case, the user must manually clear the pending interrupt or subsequently enable the Global Interrupt Enable bit and let the CPU take the ISR. If a pending interrupt is not cleared, it is continuously asserted. Although the SLEEP bit may be written and the sleep sequence executed as soon as the device enters sleep mode, the SLEEP bit is cleared by the pending interrupt and sleep mode is exited immediately.

Note 3 Upon wakeup, the instruction immediately after the sleep instruction is executed before the interrupt service routine (if enabled). The instruction after the sleep instruction is prefetched before the system actually goes to sleep. Thus, when an interrupt occurs to wake the system up, the prefetched instruction executes and the interrupt service routine is executed. (If the Global Interrupt Enable is not set, instruction execution continues where it left off before sleep.)

Note 4 If the Global Interrupt Enable bit is disabled, it is safely enabled just before the instruction that writes the SLEEP bit. It is usually undesirable to get an interrupt on the instruction boundary just before writing the SLEEP bit. This means that upon return from the interrupt, the sleep command is executed, possibly bypassing any firmware preparations that are necessary to go to sleep. To prevent this, disable interrupts before making preparations. After sleep preparations, enable global interrupts and write the SLEEP bit with the two consecutive instructions as follows.

```
and f,~01h // disable global interrupts
// (prepare for sleep, could
// be many instructions)
or f,01h // enable global interrupts
mov reg[ffh],08h // Set the sleep bit
```

Because of the timing of the Global Interrupt Enable instruction, it is not possible for an interrupt to occur immediately after that instruction. The earliest for the interrupt to occur is after the next instruction (write to the SLEEP bit) is executed. If an interrupt is pending, the sleep instruction is executed; but as described in Note 1, the sleep instruction is ignored. The first instruction executed after the ISR is the instruction after sleep.

Summary Table of the System Resource Registers *(continued)*

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,46h	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
0,48h	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
0,4Ah	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
0,4Ch	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
0,4Eh	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
1,30h	USB_CR1						BusActiv-ity	Enable-Lock	RegEnable	RW : 0
1,34h	PMAx_WA	Write Address[7:0]								RW : 00
1,35h	PMAx_WA	Write Address[7:0]								RW : 00
1,36h	PMAx_WA	Write Address[7:0]								RW : 00
1,37h	PMAx_WA	Write Address[7:0]								RW : 00
1,38h	PMAx_WA	Write Address[7:0]								RW : 00
1,39h	PMAx_WA	Write Address[7:0]								RW : 00
1,3Ah	PMAx_WA	Write Address[7:0]								RW : 00
1,3Bh	PMAx_WA	Write Address[7:0]								RW : 00
1,44h	PMAx_WA	Write Address[7:0]								RW : 00
1,45h	PMAx_WA	Write Address[7:0]								RW : 00
1,46h	PMAx_WA	Write Address[7:0]								RW : 00
1,47h	PMAx_WA	Write Address[7:0]								RW : 00
1,48h	PMAx_WA	Write Address[7:0]								RW : 00
1,49h	PMAx_WA	Write Address[7:0]								RW : 00
1,4Ah	PMAx_WA	Write Address[7:0]								RW : 00
1,4Bh	PMAx_WA	Write Address[7:0]								RW : 00
1,3Ch	PMAx_RA	Read Address[7:0]								RW : 00
1,3Dh	PMAx_RA	Read Address[7:0]								RW : 00
1,3Eh	PMAx_RA	Read Address[7:0]								RW : 00
1,3Fh	PMAx_RA	Read Address[7:0]								RW : 00
1,40h	PMAx_RA	Read Address[7:0]								RW : 00
1,41h	PMAx_RA	Read Address[7:0]								RW : 00
1,42h	PMAx_RA	Read Address[7:0]								RW : 00
1,43h	PMAx_RA	Read Address[7:0]								RW : 00
1,4Ch	PMAx_RA	Read Address[7:0]								RW : 00
1,4Dh	PMAx_RA	Read Address[7:0]								RW : 00
1,4Eh	PMAx_RA	Read Address[7:0]								RW : 00
1,4Fh	PMAx_RA	Read Address[7:0]								RW : 00
1,50h	PMAx_RA	Read Address[7:0]								RW : 00
1,51h	PMAx_RA	Read Address[7:0]								RW : 00
1,52h	PMAx_RA	Read Address[7:0]								RW : 00
1,53h	PMAx_RA	Read Address[7:0]								RW : 00
1,54h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,55h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,56h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,57h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,58h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,59h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,5Ah	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,5Bh	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00

The I²C block controls the data (SDA) and the clock (SCL) to the external I²C interface through direct connections to two dedicated GPIO pins. When I²C is enabled, these GPIO pins are not available for general purpose use. The PSoC CPU firmware interacts with the block through I/O register reads and writes, and firmware synchronization is implemented through polling and/or interrupts.

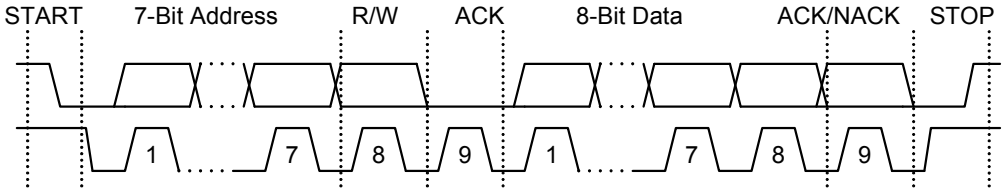
In the default operating mode, which is firmware compatible with previous versions of I²C slave modules, the I²C bus is stalled upon every received address or byte, and the CPU is required to read the data or supply data as required before the I²C bus continues. However, this I²C Slave Enhanced module provides new data buffering capability as an enhanced feature. In the EZI2C buffering mode, the I²C slave interface appears as a 32-byte RAM buffer to the external I²C master. Using a simple predefined protocol, the master controls the read and write pointers into the RAM. When this method is enabled, the slave never stalls the bus. In this protocol, the data available in the RAM (this is managed by the CPU) is valid.

15.1.1 Basic I²C Data Transfer

Figure 15-2 shows the basic form of data transfers on the I²C bus with a 7-bit address format. For a more detailed description, see the Philips Semiconductors (now NXP Semiconductors) I²C-Bus Specification, version 2.1.

A Start condition (generated by the master) is followed by a data byte, consisting of a 7-bit slave address (there is also a 10-bit address mode) and a read/write (RW) bit. The RW bit sets the direction of data transfer. The addressed slave is required to acknowledge (ACK) the bus by pulling the data line low during the ninth bit time. If the ACK is received, the transfer proceeds and the master transmits or receives an indeterminate number of bytes, depending upon the RW direction. If, for any reason, the slave does not respond with an ACK, a Stop condition is generated by the master to terminate the transfer or a Restart condition is generated for a retry attempt.

Figure 15-2. Basic I²C Data Transfer with 7-Bit Address Format



15.2 Application Overview

There are two modes of slave operation, which are differentiated by how the I²C block synchronizes CPU interaction, how and when stalling of the I²C bus is done, and data buffered.

15.2.1 Slave Operation

When Slave mode is enabled, it is continually listening on the bus for a Start condition. When detected, the transmitted address/RW byte is received and read from the I²C block by firmware. At the point where eight bits of the address/RW byte are received, a byte complete interrupt is generated. On the following low of the clock, the bus is stalled by holding the SCL line low until the PSoC device has had a chance to read the address byte and compare it to its own address. It Issues an ACK or NACK command based upon that comparison.

If there is an address match, the RW bit determines how the PSoC device sequences the data transfer in Slave mode, as shown in the two branches of Figure 15-3. I²C handshaking methodology (slave holds the SCL line low to “stall” the bus) is used, as necessary, to give the PSoC device time to respond to the events and conditions on the bus. Figure 15-3 is a graphical representation of a typical data transfer from the slave perspective.

15.3.2 I2C_XSTAT Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,C9h	I2C_XSTAT							Dir	Slave Busy	R : 0

The I²C Extended Status Register (I2C_XSTAT) reads enhanced feature status. All bits are read only. When the bits of I2C_XCFG are left in their reset state, the block is in compatibility mode, and this register is not in use.

Bit 1: Dir. This bit indicates the direction of the current transfer. A '1' indicates a master read, and a '0' indicates a master write. It is only valid when the Slave Busy bit (bit 0) is set to '1'.

Bits 0: Slave Busy. This bit is set upon a hardware address compare and is reset upon the following stop signal. Poll this bit to determine when the slave is busy and the buffer module is being accessed.

For additional information, refer to the I2C_XSTAT register on page 227.

15.3.3 I2C_ADDR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,CAh	I2C_ADDR		Slave Address[6:0]							RW : 00

The I²C Slave Address Register (I2C_ADDR) holds the slave's 7-bit address. All bits are RW.

Note When hardware address compare mode is not enabled in the I2C_XCFG register, this register is not in use.

Bits 6 to 0: Slave Address[6:0]. These 7 bits hold the slave's own device address.

For additional information, refer to the I2C_ADDR register on page 228.

15.3.4 I2C_BP Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,CBh	I2C_BP				I2C Base Pointer[4:0]					R : 00

The I²C Base Address Pointer Register (I2C_BP) contains the base address value of the RAM data buffer.

Note When in compatibility mode, this register is not in use.

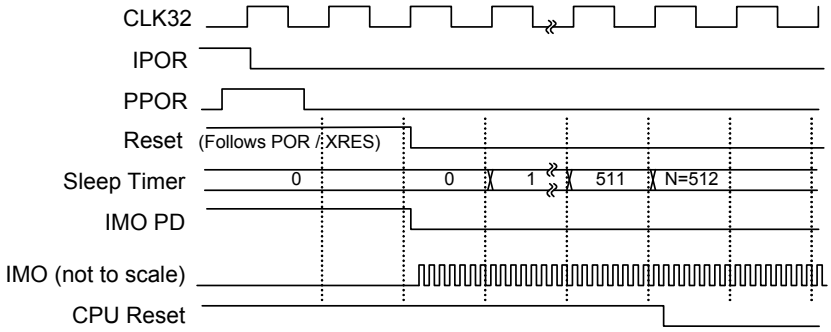
Bits 4 to 0: I2C Base Pointer[4:0]. In the EZI2C protocol, the first data byte after the slave address transaction in write mode is the base address for subsequent reads and writes and it is transferred directly into this register. If the desired transaction is a master write to the slave, subsequent bytes are written to the RAM buffer starting with this address and auto incremented (see I2C_CP Register).

In case of a read, a Start or Restart must be issued and the read location starts with this address and again subsequent read addresses are auto incremented as pointed to by the I2C_CP register value. The value of this register is modified only at the beginning of every I²C write transaction. The I²C master must always supply a value for this register in the first byte of data after the slave's address in a given write transaction. If performing reads, the master need not set the value of this register. The current value of this register is also used directly for reads.

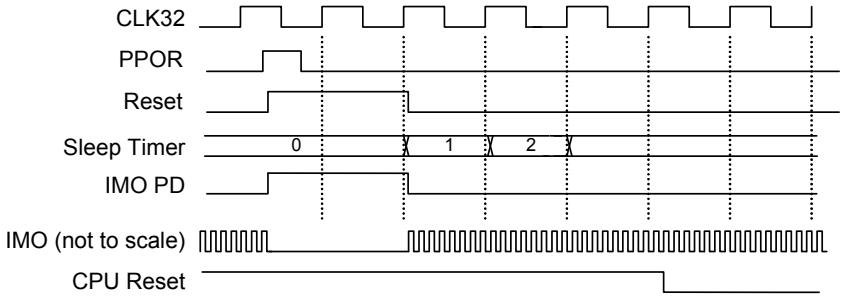
For additional information, refer to the I2C_BP register on page 229.

Figure 16-4. Key Signals During POR and XRES

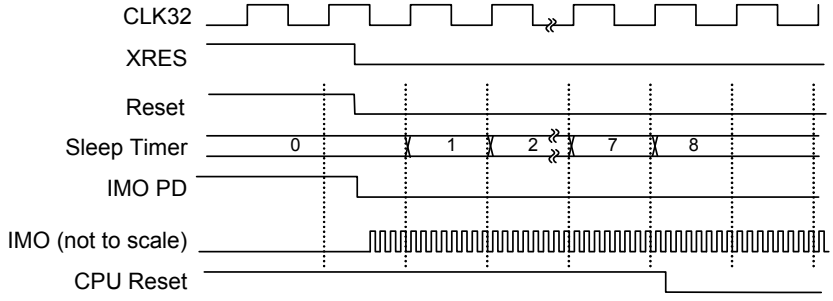
POR (IPOR followed by PPOR): Reset while POR is high (IMO off), then 511(+) cycles (IMO on), and then the CPU reset is released. **XRES** is the same, with N=8.



PPOR (with no IPOR): Reset while PPOR is high and to the end of the next 32K cycle (IMO off); 1 cycle IMO on before the CPU reset is released. Note that at the 3V level, PPOR tends to be brief because the reset clears the POR range register (VLT_CR) back to the default 2.4V setting.



XRES: Reset while XRES is high (IMO off), then 7(+) cycles (IMO on), and then the CPU reset is released.



Configuration Register

The configuration block contains 1 register. This register must not be changed while the block is enabled. Note that the SPI Configuration register is located in bank 1 of the PSoC device's memory map.

18.2.4 SPI_CFG Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,29h	SPI_CFG	Clock Sel[2:0]			Bypass	SS_	SS_EN_	Int Sel	Slave	RW : 00

The SPI Configuration Register (SPI_CFG) is used to configure the SPI.

Bits 7 to 5: Clock Sel [2:0]. Clock Selection. These bits determine the operating frequency of the SPI Master.

Bit 4: Bypass. This bit determines whether or not the inputs are synchronized to SYSCLK.

Bit 3: SS_. Slave Select. This bit determines the logic value of the SS_ signal when the SS_EN_ signal is asserted (SS_EN_ = 0).

Bit 2: SS_EN_. Slave Select Enable. This active low bit determines if the slave select (SS_) signal is driven internally. If it is driven internally, its logic level is determined by the SS_ bit. If it is driven externally, its logic level is determined by the external pin.

Bit 1: Int Sel. Interrupt Select. This bit selects which condition produces an interrupt.

Bit 0: Slave. This bit determines whether the block functions as a master or slave.

For additional information, refer to the SPI_CFG register on page 261.

18.2.4.1 SPI Configuration Register Definitions

Table 18-5. SPI Configuration Register Descriptions

Bit #	Name	Access	Mode	Description
7:5	Clock Sel	Read/Write	Master	SYSCLK 000b / 2 001b / 4 010b / 8 011b / 16 100b / 32 101b / 64 110b / 128 111b / 256
4	Bypass	Read/Write	Master/Slave	0 = All pin inputs are doubled, synchronized. 1 = Input synchronization is bypassed.
3	SS_	Read/Write	Slave	0 = Slave selected. 1 = Slave selection is determined from external SS_ pin.
2	SS_EN_	Read/Write	Slave	0 = Slave selection determined from SS_ bit. 1 = Slave selection determined from external SS_ pin.
1	Int Sel	Read/Write	Master/Slave	0 = Interrupt on TX Reg Empty. 1 = Interrupt on SPI Complete.
0	Slave	Read/Write	Master/Slave	0 = Operates as a master. 1 = Operates as a slave.

18.2.5 Related Registers

- IO_CFG1 Register on page 272.

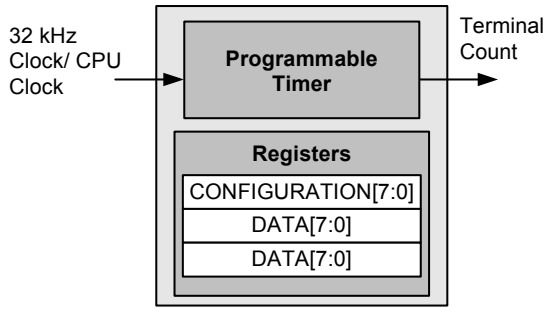
19. Programmable Timer

This chapter presents the Programmable Timer and its associated registers. For a complete table of the programmable timer registers, refer to the Summary Table of the System Resource Registers on page 106. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 187.

19.1 Architectural Description

The device has three programmable timers (TIMER0, TIMER1, TIMER2). All three timers are individually controlled. The programmable timers are 16-bit down counters for the device. TIMER0 has a terminal count output. The timers have one configuration and two data registers associated with them. You start these timers by setting the START bit in their configuration registers (PT0_CFG, PT1_CFG, PT2_CFG). When started, the timers always start counting down from the value loaded into their data registers (PT*_DATA1, PT*_DATA0). The timers have a one-shot mode, in which the timers complete one full count cycle and stop. In one-shot mode the START bit in the configuration register is cleared after completion of one full count cycle. Setting the START bit restarts the timer.

Figure 19-1. Programmable Timer Block Diagram



19.1.1 Operation

When started, the programmable timer loads the value contained in its data registers and counts down to its terminal count of zero. The timers output an active high terminal count pulse for one clock cycle upon reaching the terminal count. The low time of the terminal count pulse is equal to the loaded decimal count value, multiplied by the clock period (TC_{pw} = COUNT $VALUE_{decimal}$ * CLK_{period}). The period of the terminal count output is the pulse width of the terminal count, plus one clock period (TC_{period} = TC_{pw} + CLK_{period}). Refer to Figure 19-2 and Figure 19-3.

Only TIMER0 outputs this terminal count output. TIMER1 and TIMER2 do not have a terminal count output.

The timers work on either the 32 kHz clock or CPU clock. This clock selection is done using the CLKSEL bits in the respective configuration registers (PT0_CFG, PT1_CFG, PT2_CFG). Make clock selections before setting the START bit so that the timing is not affected and clock frequency does not change while the timer is running.

TIMER1 works on prescaled IMO clock (IMO-P) when the CSD_MODE bit in the CS_CR0 (0,A0) register is set to ‘1’. TIMER1 outputs the START signal, which used in the TrueTouch module during CSD mode. Refer to 11.1.1.5 Sigma Delta on page 89 for more details on TrueTouch CSD mode. When CSD_MODE is set to ‘0‘, it works on either the 32 kHz clock or CPU clock, depending on the CLKSEL bit setting.

Section E: Registers

The Registers section discusses the registers of the PSoC device. It lists all the registers in mapping tables, in address order. For easy reference, each register is linked to the page of a detailed description located in the next chapter. This section encompasses the following chapter:

- Register Reference chapter on page 187.

Register General Conventions

The register conventions specific to this section and the Register Reference chapter are listed in the following table:

Register Conventions

Convention	Description
Empty, grayed-out table cell	Illustrates a reserved bit or group of bits.
‘x’ before the comma in an address	Indicates the register exists in register bank 1 and register bank 2.
‘x’ in a register name	Indicates that there are multiple instances/address ranges of the same register.
R	Read register or bit(s).
W	Write register or bit(s).
O	Only a read/write register or bit(s).
L	Logical register or bit(s).
C	Clearable register or bit(s).
#	Access is bit specific.

Register Mapping Tables

The PSoC device has a total register address space of 512 bytes. The register space is also referred to as I/O space and is broken into two parts: Bank 0 (user space) and Bank 1 (configuration space). The XIO bit in the Flag register (CPU_F) determines which bank the user is currently in. When the XIO bit is set, the user is said to be in the “extended” address space or the “configuration” registers.

Refer to the individual PSoC device data sheets for device-specific register mapping information.

Register Map Bank 0 Table: User Space

Name	Addr (0,Hex)	Access	Page	Name	Addr (0,Hex)	Access	Page	Name	Addr (0,Hex)	Access	Page	Name	Addr (0,Hex)	Access	Page
PRT0DR	00	RW	188	EP1_CNT0	40	#	201		80				C0		
PRT0IE	01	RW	189	EP1_CNT1	41	RW	202		81				C1		
	02			EP2_CNT0	42	#	201		82				C2		
	03			EP2_CNT1	43	RW	202		83				C3		
PRT1DR	04	RW	188	EP3_CNT0	44	#	201		84				C4		
PRT1IE	05	RW	189	EP3_CNT1	45	RW	202		85				C5		
	06			EP4_CNT0	46	#	201		86				C6		
	07			EP4_CNT1	47	RW	202		87				C7		
PRT2DR	08	RW	188	EP5_CNT0	48	#	201		88			I2C_XCFG	C8	RW	226
PRT2IE	09	RW	189	EP5_CNT1	49	RW	202		89			I2C_XSTAT	C9	R	227
	0A			EP6_CNT0	4A	#	201		8A			I2C_ADDR	CA	RW	228
	0B			EP6_CNT1	4B	RW	202		8B			I2C_BP	CB	R	229
PRT3DR	0C	RW	188	EP7_CNT0	4C	#	201		8C			I2C_CP	CC	R	230
PRT3IE	0D	RW	189	EP7_CNT1	4D	RW	202		8D			CPU_BP	CD	RW	231
	0E			EP8_CNT0	4E	#	201		8E			CPU_CP	CE	R	232
	0F			EP8_CNT1	4F	RW	202		8F			I2C_BUF	CF	RW	233
PRT4DR	10	RW	188		50				90			CUR_PP	D0	RW	234
PRT4IE	11	RW	189		51				91			STK_PP	D1	RW	235
	12				52				92				D2		
	13				53				93			IDX_PP	D3	RW	236
	14				54				94			MVR_PP	D4	RW	237
	15				55				95			MVW_PP	D5	RW	238
	16				56				96			I2C_CFG	D6	RW	239
	17				57				97			I2C_SCR	D7	#	240
	18			PMA0_DR	58	RW	203		98			I2C_DR	D8	RW	241
	19			PMA1_DR	59	RW	203		99				D9		
	1A			PMA2_DR	5A	RW	203		9A			INT_CLR0	DA	RW	242
	1B			PMA_DR	5B	RW	203		9B			INT_CLR1	DB	RW	244
	1C			PMA4_DR	5C	RW	203		9C			INT_CLR2	DC	RW	246
	1D			PMA5_DR	5D	RW	203		9D				DD		
	1E			PMA6_DR	5E	RW	203		9E			INT_MSK2	DE	RW	248
	1F			PMA7_DR	5F	RW	203		9F			INT_MSK1	DF	RW	249
	20				60			CS_CR0	A0	RW	211	INT_MSK0	E0	RW	250
	21			AMUX_CFG	61	RW	204	CS_CR1	A1	RW	212	INT_SW_EN	E1	RW	251
	22				62			CS_CR2	A2	RW	213	INT_VC	E2	RC	252
	23				63			CS_CR3	A3	RW	214	RES_WDT	E3	W	253
	24			PMA8_DR	64	RW	203	CS_CNTL	A4	RW	215		E4		
	25			PMA9_DR	65	RW	203	CS_CNTH	A5	RW	216		E5		
	26			PMA10_DR	66	RW	203	CS_STAT	A6	#	217		E6		
	27			PMA11_DR	67	RW	203	CS_TIMER	A7	RW	218		E7		
	28			PMA12_DR	68	RW	203	CS_SLEW	A8	RW	219		E8		
SPI_TXR	29	W	190	PMA13_DR	69	RW	203	PRS_CR	A9	RW	220		E9		
SPI_RXR	2A	R	191	PMA14_DR	6A	RW	203		AA				EA		
SPI_CR	2B	#	192	PMA15_DR	6B	RW	203		AB				EB		
	2C			TMP_DR0	6C	RW	266		AC				EC		
	2D			TMP_DR1	6D	RW	266		AD				ED		
	2E			TMP_DR2	6E	RW	266		AE				EE		
	2F			TMP_DR3	6F	RW	266		AF				EF		
	30				70			PT0_CFG	B0	RW	221		F0		
USB_SOF0	31	R	193		71			PT0_DATA1	B1	RW	223		F1		
USB_SOF1	32	R	194		72			PT0_DATA0	B2	RW	223		F2		
USB_CR0	33	RW	195		73			PT1_CFG	B3	RW	224		F3		
USBIO_CR0	34	#	196		74			PT1_DATA1	B4	RW	223		F4		
USBIO_CR1	35	#	197		75			PT1_DATA0	B5	RW	223		F5		
EP0_CR	36	#	198		76			PT2_CFG	B6	RW	225		F6		
EP0_CNT0	37	#	199		77			PT2_DATA1	B7	RW	223	CPU_F	F7	RL	254
EP0_DR0	38	RW	200	CMP_RDC	78	#	205	PT2_DATA0	B8	RW	223		F8		
EP0_DR1	39	RW	200	CMP_MUX	79	RW	206		B9				F9		
EP0_DR2	3A	RW	200	CMP_CR0	7A	RW	207		BA				FA		
EP0_DR3	3B	RW	200	CMP_CR1	7B	RW	208		BB				FB		
EP0_DR4	3C	RW	200	CMP_LUT	7C	RW	210		BC				FC		
EP0_DR5	3D	RW	200		7D				BD			IDAC_D	FD	RW	256
EP0_DR6	3E	RW	200		7E				BE			CPU_SCR1	FE	#	257
EP0_DR7	3F	RW	200		7F				BF			CPU_SCR0	FF	#	258

Gray fields are reserved. # Access is bit specific.

21.3.2 PRTxIE

Port Interrupt Enable Registers

Individual Register Names and Addresses:

PRT0IE : 0,01h PRT1IE : 0,05h PRT2IE : 0,09h PRT3IE : 0,0Dh
PRT4IE : 0,11h

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	Interrupt Enables[7:0]							

These registers enable or disable interrupts from individual GPIO pins.

The upper nibble of the PRT4IE register returns the last data bus value when read and must be masked off before using this information. For additional information, refer to the Register Definitions on page 59 in the GPIO chapter.

Bit	Name	Description
7:0	**Interrupt Enables[7:0]**	These bits enable the corresponding port pin interrupt. Only four LSB are used since this port has four pins. 0 Port pin interrupt disabled for the corresponding pin. 1 Port pin interrupt enabled for the corresponding pin. Interrupt mode is determined by the IOINT bit in the IO_CFG1 register.

21.3.15 EPx_CNT1

Endpoint Count 1 Registers

Individual Register Names and Addresses:

EP1_CNT1 : 0,41h EP2_CNT1 : 0,43h EP3_CNT1 : 0,45h EP4_CNT1 : 0,47h
EP5_CNT1 : 0,49h EP6_CNT1 : 0,4Bh EP7_CNT1 : 0,4Dh EP8_CNT1 : 0,4Fh

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	Data Count[7:0]							

These registers are endpoint count 1 registers.

For additional information, refer to the Register Definitions on page 171 in the Full-Speed USB chapter.

Bit	Name	Description
7:0	**Data Count[7:0]**	These bits are the eight LSb of a 9-bit counter. The MSb is the Count MSb of the EPx_CNT0 register.

21.3.20 CMP_CR0

Comparator Control Register 0

Individual Register Names and Addresses:

CMP_CR0 : 0,7Ah

	7	6	5	4	3	2	1	0
Access : POR				RW : 0				RW : 0
Bit Name				CMP1EN				CMP0EN

This register is used to enable and configure the input range of the comparators.

In the table above, reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 103 in the Comparators chapter.

Bit	Name	Description	
4	**CMP1EN**	0	Comparator 1 disabled, powered off.
		1	Comparator 1 enabled.
0	**CMP0EN**	0	Comparator 0 disabled, powered off.
		1	Comparator 0 enabled.

21.3.23 CS_CR0

TrueTouch Control Register 0

Individual Register Names and Addresses:

CS_CR0 : 0,A0h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0		RW : 0	RW : 0	RW : 0	RW : 0		RW : 0
Bit Name	CSOUT[1:0]		CSD_ PRSCLK	CSD_CS_ CLK	CSD_ MODE	MODE[1:0]		EN

This register controls the operation of the TrueTouch counters.

Do not write bits [7:1] while the block is enabled. For additional information, refer to the Register Definitions on page 92 in the TrueTouch Module chapter.

Bit	Name	Description
7:6	CSOUT[1:0]	These bits select between a number of TrueTouch signals that can be driven to an output pin. 00b IN. 01bCS_INT. 10bCOL. 11bCOH.
5	**CSD_PRSCLK**	This bit selects between IMO-P or the PRS output as a clock source to drive the main capacitor switch. 0 Select IMO-P. 1 Select PRS Output.
4	**CSD_CS_CLK**	This bit selects between IMO or IMO-P for the TrueTouch counters to work. Depending on this bit selection either IMO or IMO-P is sent as the source clock to the clock dividers which generate CS_CLK as shown in Figure 11-13 on page 91. 0 Select IMO. 1 Select IMO-P.
3	**CSD_MODE**	This bit enables the CSD mode. When this bit is enabled, the TIMER1 block works on IMO-P (pre-scaled IMO) clock. This is also an enable for TrueTouch counters to toggle. **Note**: Once the CSD_MODE bit is enabled, the IMO-P clock is a free running divider clock that cannot be stopped and re-started. The IMO-P and the CPU clock are both derived from the IMO clock but the phase relationship between them is nondeterministic. 0 Disable CSD mode. Programmable Timer1 works on either CPUCLK/CLK32, (depends on CLKSEL bit selection in PT1_CFG (0, B3h) register). 1 Enable CSD mode. When this bit is set to 1, Programmable Timer1 works on IMO-P.
2:1	**MODE[1:0]**	TrueTouch Counter Mode. 00b Event Mode. Start in enable, stop on interrupt event. 01b Pulse Width Mode. Start on positive edge of next input. Stop on negative edge of input. 10b Period Mode. Start on positive edge of input. Stop on next positive edge of input. 11b Start in enable, continuous operation until disable.
0	**EN**	0 Counting is stopped and all counter values are reset to zero. 1 Counters are enabled for counting.

21.3.27 CS_CNTL

TrueTouch Counter Low Byte Register

Individual Register Names and Addresses:

CS_CNTL : 0,A4h

	7	6	5	4	3	2	1	0
Access : POR	RO : 00							
Bit Name	Data[7:0]							

This register contains the current count for the low byte counter and is read only.

For additional information, refer to the Register Definitions on page 92 in the TrueTouch Module chapter.

Bit	Name	Description
7:0	**Data[7:0]**	On a read of this register, the current count is returned. It may only be read when the counter is stopped. **Note** The counter must be stopped by the configured event. When the counter is disabled, the count is reset to 00h.

21.3.49 MVR_PP

MVI Read Page Pointer Register

Individual Register Names and Addresses:

MVR_PP : 0,D4h

	7	6	5	4	3	2	1	0
Access : POR						RW : 0		
Bit Name						Page Bits[2:0]		

This register is used to set the effective SRAM page for MVI read memory accesses in a multi-SRAM page PSoC device.

This register is only used when a device has more than one page of SRAM. In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Reserved bits must always be written with a value of '0'. For additional information, refer to the Register Definitions on page 42 in the RAM Paging chapter.

Bit	Name	Description
2:0	**Page Bits[2:0]**	Bits determine which SRAM page an MVI Read instruction operates on. 000b SRAM Page 0 001b SRAM Page 1 010b SRAM Page 2 011b SRAM Page 3 100b SRAM Page 4 101b SRAM Page 5 110b SRAM Page 6 111b SRAM Page 7

21.3.55 INT_CLR1

Interrupt Clear Register 1

Individual Register Names and Addresses:

INT_CLR1 : 0,DBh

	7	6	5	4	3	2	1	0
Access : POR	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0
Bit Name	Endpoint3	Endpoint2	Endpoint1	Endpoint0	USB_SOF	USB_BUS_RST	Timer2	Timer1

This register is used to enable the individual interrupt sources' ability to clear posted interrupts.

When bits in this register are read, a '1' is returned for every bit position that has a corresponding posted interrupt. When bits in this register are written with a '0' and ENSWINT is not set, posted interrupts are cleared at the corresponding bit positions. If there is no posted interrupt, there is no effect. When bits in this register are written with a '1' and ENSWINT is set, an interrupt is posted in the interrupt controller.

For additional information, refer to the Register Definitions on page 48 in the Interrupt Controller chapter.

Bit	Name	Description
7	**Endpoint3**	Read 0 No posted interrupt for USB Endpoint3. Read 1 Posted interrupt present for USB Endpoint3. Write 0 AND ENSWINT = 0 Clear posted interrupt if it exists. Write 1 AND ENSWINT = 0 No effect. Write 0 AND ENSWINT = 1 No effect. Write 1 AND ENSWINT = 1 Post an interrupt for USB Endpoint3.
6	**Endpoint2**	Read 0 No posted interrupt for USB Endpoint2. Read 1 Posted interrupt present for USB Endpoint2. Write 0 AND ENSWINT = 0 Clear posted interrupt if it exists. Write 1 AND ENSWINT = 0 No effect. Write 0 AND ENSWINT = 1 No effect. Write 1 AND ENSWINT = 1 Post an interrupt for USB Endpoint2.
5	**Endpoint1**	Read 0 No posted interrupt for USB Endpoint1. Read 1 Posted interrupt present for USB Endpoint1. Write 0 AND ENSWINT = 0 Clear posted interrupt if it exists. Write 1 AND ENSWINT = 0 No effect. Write 0 AND ENSWINT = 1 No effect. Write 1 AND ENSWINT = 1 Post an interrupt for USB Endpoint1.

(continued on next page)

21.4.19 OSC_CR2

Oscillator Control Register 2

Individual Register Names and Addresses:

OSC_CR2 : 1,E2h

	7	6	5	4	3	2	1	0
Access : POR				RW : 0		RW : 0	RW : 0	
Bit Name				CLK48MEN		EXTCLKEN	IMODIS	

This register is used to configure various features of internal clock sources and clock nets.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Reserved bits must always be written with a value of '0'. For additional information, refer to the Register Definitions on page 112 in the Digital Clocks chapter.

Bit	Name	Description
4	**CLK48MEN**	This is the 48 MHz clock enable bit. 0 Disables the 48 MHz clock. 1 Enables the 48 MHz clock.
2	**EXTCLKEN**	External Clock Mode Enable. 0 Disabled. Operate from internal main oscillator. 1 Enabled. Operate from the clock supplied at P1[4] or P1[1] based upon the TSYNC bit in CPU_SCR1.
1	**IMODIS**	Internal Oscillator Disable. This bit can be set to save power when using an external clock on P1[4]. 0 Enabled. Internal oscillator enabled. 1 Disabled. **Note** This bit must not be set high in the same instruction that sets EXTCLKEN high, but it can be set in the next instruction. Also, this bit must not be set high if the external clock frequency is less than 6 MHz. When switching from external clock to internal clock, the IMO must be enabled for at least 10 µs before the transition to internal clock. Refer to Switch Operation on page 110.