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 (16kB)
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/cy8ctmg201a-32lqxi

1. Pin Information

This chapter lists, describes, and illustrates all pins and pinout configurations for the CY8CTMG20x, CY8CTMG20xA, CY8CTST200, and CY8CTST200A PSoC devices. For up-to-date ordering, pinout, and packaging information, refer to the individual PSoC device's data sheet or go to http://www.cypress.com.

1.1 Pinouts

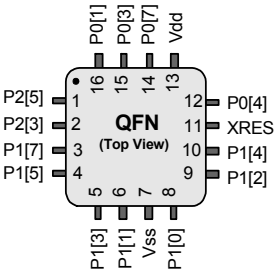
The CY8CTMG20x, CY8CTMG20xA, CY8CTST200, and CY8CTST200A PSoC devices are available in a variety of packages. Every ***port*** pin (labeled with a "P"), except for ***Vss***, ***Vdd***, and XRES in the following tables and illustrations, is capable of Digital I/O.

1.1.1 CY8CTMG200-16LGXI, CY8CTMG200A-16LGXI, CY8CTST200-16LGXI, CY8CTST200A-16LGXI PSoC 16-Pin Part Pinout

Table 1-1. 16-Pin QFN/COL Part Pinout

Pin No.	Type		Name	Description
	Digital	Analog		
1	IO	I	P2[5]	XTAL Out
2	IO	I	P2[3]	XTAL In
3	IOHR	I	P1[7]	I2C SCL, SPI SS
4	IOHR	I	P1[5]	I2C SDA, SPI MISO
5	IOHR	I	P1[3]	SPI CLK
6	IOHR	I	P1[1]	TC CLK*, I2C SCL, SPI MOSI
7	Power		Vss	Ground pin
8	IOHR	I	P1[0]	TC DATA*, I2C SDA, SPI CLK
9	IOHR	I	P1[2]	
10	IOHR	I	P1[4]	EXTCLK
11	Input		XRES	Active high external reset with internal pull down
12	IOH	I	P0[4]	
13	Power		Vdd	Power pin
14	IOH	I	P0[7]	
15	IOH	I	P0[3]	Integrating input
16	IOH	I	P0[1]	Integrating input

CY8CTMG200-16LGXI, CY8CTMG200A-16LGXI, CY8CTST200-16LGXI CY8CTST200A-16LGXI PSoC Devices



LEGEND A = Analog, I = Input, O = Output, H = 5 mA High Output Drive, R = Regulated Output Option.

* These are the ISSP pins, which are not High Z at POR (Power On Reset).

Table 2-2. Instruction Set Summary Sorted Alphabetically by Mnemonic

Opcode Hex	Cycles	Bytes	Instruction Format	Flags	Opcode Hex	Cycles	Bytes	Instruction Format	Flags	Opcode Hex	Cycles	Bytes	Instruction Format	Flags
09	4	2	ADC A, expr	C, Z	76	7	2	INC [expr]	C, Z	20	5	1	POP X	
0A	6	2	ADC A, [expr]	C, Z	77	8	2	INC [X+expr]	C, Z	18	5	1	POP A	Z
0B	7	2	ADC A, [X+expr]	C, Z	Fx	13	2	INDEX	Z	10	4	1	PUSH X	
0C	7	2	ADC [expr], A	C, Z	Ex	7	2	JACC		08	4	1	PUSH A	
0D	8	2	ADC [X+expr], A	C, Z	Cx	5	2	JC		7E	10	1	RETI	C, Z
0E	9	3	ADC [expr], expr	C, Z	8x	5	2	JMP		7F	8	1	RET	
0F	10	3	ADC [X+expr], expr	C, Z	Dx	5	2	JNC		6A	4	1	RLC A	C, Z
01	4	2	ADD A, expr	C, Z	Bx	5	2	JNZ		6B	7	2	RLC [expr]	C, Z
02	6	2	ADD A, [expr]	C, Z	Ax	5	2	JZ		6C	8	2	RLC [X+expr]	C, Z
03	7	2	ADD A, [X+expr]	C, Z	7C	13	3	LCALL		28	11	1	ROMX	Z
04	7	2	ADD [expr], A	C, Z	7D	7	3	LJMP		6D	4	1	RRC A	C, Z
05	8	2	ADD [X+expr], A	C, Z	4F	4	1	MOV X, SP		6E	7	2	RRC [expr]	C, Z
06	9	3	ADD [expr], expr	C, Z	50	4	2	MOV A, expr	Z	6F	8	2	RRC [X+expr]	C, Z
07	10	3	ADD [X+expr], expr	C, Z	51	5	2	MOV A, [expr]	Z	19	4	2	SBB A, expr	C, Z
38	5	2	ADD SP, expr		52	6	2	MOV A, [X+expr]	Z	1A	6	2	SBB A, [expr]	C, Z
21	4	2	AND A, expr	Z	53	5	2	MOV [expr], A		1B	7	2	SBB A, [X+expr]	C, Z
22	6	2	AND A, [expr]	Z	54	6	2	MOV [X+expr], A		1C	7	2	SBB [expr], A	C, Z
23	7	2	AND A, [X+expr]	Z	55	8	3	MOV [expr], expr		1D	8	2	SBB [X+expr], A	C, Z
24	7	2	AND [expr], A	Z	56	9	3	MOV [X+expr], expr		1E	9	3	SBB [expr], expr	C, Z
25	8	2	AND [X+expr], A	Z	57	4	2	MOV X, expr		1F	10	3	SBB [X+expr], expr	C, Z
26	9	3	AND [expr], expr	Z	58	6	2	MOV X, [expr]		00	15	1	SSC	
27	10	3	AND [X+expr], expr	Z	59	7	2	MOV X, [X+expr]		11	4	2	SUB A, expr	C, Z
70	4	2	AND F, expr	C, Z	5A	5	2	MOV [expr], X		12	6	2	SUB A, [expr]	C, Z
41	9	3	AND reg[expr], expr	Z	5B	4	1	MOV A, X	Z	13	7	2	SUB A, [X+expr]	C, Z
42	10	3	AND reg[X+expr], expr	Z	5C	4	1	MOV X, A		14	7	2	SUB [expr], A	C, Z
64	4	1	ASL A	C, Z	5D	6	2	MOV A, reg[expr]	Z	15	8	2	SUB [X+expr], A	C, Z
65	7	2	ASL [expr]	C, Z	5E	7	2	MOV A, reg[X+expr]	Z	16	9	3	SUB [expr], expr	C, Z
66	8	2	ASL [X+expr]	C, Z	5F	10	3	MOV [expr], [expr]		17	10	3	SUB [X+expr], expr	C, Z
67	4	1	ASR A	C, Z	60	5	2	MOV reg[expr], A		4B	5	1	SWAP A, X	Z
68	7	2	ASR [expr]	C, Z	61	6	2	MOV reg[X+expr], A		4C	7	2	SWAP A, [expr]	Z
69	8	2	ASR [X+expr]	C, Z	62	8	3	MOV reg[expr], expr		4D	7	2	SWAP X, [expr]	
9x	11	2	CALL		63	9	3	MOV reg[X+expr], expr		4E	5	1	SWAP A, SP	Z
39	5	2	CMP A, expr	if (A=B) Z=1 if (A<B) C=1	3E	10	2	MVI A, [[expr]++]	Z	47	8	3	TST [expr], expr	Z
3A	7	2	CMP A, [expr]		3F	10	2	MVI [[expr]++], A		48	9	3	TST [X+expr], expr	Z
3B	8	2	CMP A, [X+expr]		40	4	1	NOP		49	9	3	TST reg[expr], expr	Z
3C	8	3	CMP [expr], expr		29	4	2	OR A, expr	Z	4A	10	3	TST reg[X+expr], expr	Z
3D	9	3	CMP [X+expr], expr		2A	6	2	OR A, [expr]	Z	72	4	2	XOR F, expr	C, Z
73	4	1	CPL A	Z	2B	7	2	OR A, [X+expr]	Z	31	4	2	XOR A, expr	Z
78	4	1	DEC A	C, Z	2C	7	2	OR [expr], A	Z	32	6	2	XOR A, [expr]	Z
79	4	1	DEC X	C, Z	2D	8	2	OR [X+expr], A	Z	33	7	2	XOR A, [X+expr]	Z
7A	7	2	DEC [expr]	C, Z	2E	9	3	OR [expr], expr	Z	34	7	2	XOR [expr], A	Z
7B	8	2	DEC [X+expr]	C, Z	2F	10	3	OR [X+expr], expr	Z	35	8	2	XOR [X+expr], A	Z
30	9	1	HALT		43	9	3	OR reg[expr], expr	Z	36	9	3	XOR [expr], expr	Z
74	4	1	INC A	C, Z	44	10	3	OR reg[X+expr], expr	Z	37	10	3	XOR [X+expr], expr	Z
75	4	1	INC X	C, Z	71	4	2	OR F, expr	C, Z	45	9	3	XOR reg[expr], expr	Z
										46	10	3	XOR reg[X+expr], expr	Z

Note 1 Interrupt acknowledge to Interrupt Vector table = 13 cycles.

Note 2 The number of cycles required by an instruction is increased by one for instructions that span 128 byte page boundaries in the Flash memory space.

4.2.6 MVW_PP Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D5h	MVW_PP						Page Bits[2:0]			RW : 0

The MVI Write Page Pointer Register (MVW_PP) sets the effective SRAM page for MVI write memory accesses in a multi-SRAM page PSoC device.

Bits 2 to 0: Page Bits[2:0]. These bits are only used by the `MVI [expr], A` instruction, not to be confused with the `MVI A, [expr]` instruction covered by the MVR_PP register. This instruction is considered a write because data is transferred from the microprocessor's A register (CPU_A) to SRAM.

When an `MVI [expr], A` instruction is executed in a device with more than one page of SRAM, the SRAM address that is written by the instruction is determined by the value of the least significant bits in this register. However, the pointer for the `MVI [expr], A` instruction is always located in the current SRAM page. See the *PSoC Designer Assembly Language User Guide* for more information on the `MVI [expr], A` instruction.

The function of this register and the MVI instructions are independent of the SRAM Paging bits in the CPU_F register. For additional information, refer to the MVW_PP register on page 238.

4.2.7 Related Registers

- CPU_F Register on page 32.

6.2.3 PRTxDMx Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,xxh	PRTxDM0	Drive Mode 0[7:0]								RW : 00
1,xxh	PRTxDM1	Drive Mode 1[7:0]								RW : FF

LEGEND
xx An “x” after the comma in the address field indicates that there are multiple instances of the register. For an expanded address listing of these registers, refer to the Core Register Summary on page 24.

The Port Drive Mode Bit Registers (PRTxDM0 and PRTxDM1) specify the Drive mode for GPIO pins.

Bits 7 to 0: Drive Mode x[7:0]. In the PRTxDMx registers there are four possible drive modes for each port pin. Two mode bits are required to select one of these modes, and these two bits are spread into two different registers (PRTxDM0 and PRTxDM1). The bit position of the affected port pin (for example, Pin[2] in Port 0) is the same as the bit position of each of the two drive mode register bits that control the Drive mode for that pin (for example, bit[2] in PRT0DM0 and bit[2] in PRT0DM1). The two bits from the two registers are treated as a group. These are referred to as DM1 and DM0, or together as DM[1:0]. Drive modes are shown in Table 6-3.

For analog I/O, set the drive mode to the High Z analog mode, 10b. The 10b mode disables the block’s digital input buffer so no crowbar current flows, even when the analog input is not close to either power rail. If the 10b drive mode is used, the pin is always read as a zero by the CPU and the pin cannot generate a useful interrupt. (It is not strictly required that you select High Z mode for analog operation.)

When digital inputs are needed on the same pin as analog inputs, use the 11b Drive mode with the corresponding data bit (in the PRTxDR register) set high.

Drive Modes		Pin State	Description
DM1	DM0		
0	0	Resistive pull up	Resistive high, strong low
0	1	Strong drive	Strong high, strong low
1	0	High impedance, analog (**reset state**)	High Z high and low, digital input disabled (for zero power) (**reset state**)
1	1	Open drain low	High Z high (digital input enabled), strong low.

The GPIO provides a default drive mode of high impedance, analog (High Z). This is achieved by forcing the reset state of all PRTxDM1 registers to FFh.

For additional information, refer to the PRTxDM0 register on page 259, and the PRTxDM1 register on page 260.

9.3.4 Related Registers

- OSC_CR0 Register, on page 113.
- PRTxDR Registers register on page 59.
- PRTxIE Registers register on page 59.

10.3.3 SLP_CFG2 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,ECh	SLP_CFG2					ALT_Buzz [1:0]		I2C_ON	LSO_OFF	RW : 00

The Sleep Configuration Register (SLP_CFG2) holds the configuration for I2C sleep, deep sleep, and buzz.

Bits 3 and 2: ALT_Buzz[1:0]. These bits control additional selections for POR/LVD buzz rates. These are lower rates than the compatibility mode to provide for lower average power.
‘00’ - Compatibility mode, buzz rate determined by PSSDC bits.
‘01’ - Duty cycle is 1/32768.
‘10’ - Duty cycle is 1/8192.
‘11’ - Reserved.

Bit 1: I2C_ON. This bit enables the standby regulator in sleep at a level sufficient to supply the I2C circuitry. It is independent of the LSO_OFF bit.

Bit 0: LSO_OFF: This bit disables the LSO oscillator when in sleep state. By default, the LSO oscillator runs in sleep. When this bit is ‘0’, the standby regulator is active at a power level to supply the LSO and Sleep timer circuitry and the LSO is enabled. When this bit is ‘1’, the LSO is disabled in sleep, which in turn, disables the Sleep Timer, Watchdog Timer, and POR/LVD buzzing activity in sleep. If I2C_ON is not enabled and this bit is set, the device is in the lowest power deep sleep mode. Only a GPIO interrupt awakens the device from deep sleep mode.

For additional information, refer to the SLP_CFG2 register on page 284.

10.3.4 SLP_CFG3 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,EDh	SLP_CFG3		DBL_TAPS	T2TAP [1:0]		T1TAP [1:0]		T0TAP [1:0]		RW : 0x7F

The Sleep Configuration Register (SLP_CFG3) holds the configuration of the wakeup sequence taps.

It is strongly recommended to not alter this register setting.

Bit 6: DBL_TAPS. When this bit is set, all the tap values (T0, T1, and T2) are doubled for the wakeup sequence.

Bits 5 and 4: T2TAP[1:0]. These bits control the duration of the T2-T4 sequence (see Figure 10-2 on page 75) by selecting a tap from the WakeupTimer. Note The T2 delay is only valid for the wakeup sequence. It is not used for the buzz sequence.

‘00’ - 1 µs
‘01’ - 2 µs
‘10’ - 5 µs
‘11’ - 10 µs

Bits 3 and 2: T1TAP[1:0]. These bits control the duration of the T1-T2 sequence (see Figure 10-2 on page 75) by selecting a tap from the Wakeup Timer.

‘00’ - 3 µs
01’ - 4 µs
‘10’ - 5 µs
‘11’ - 10 µs

Bits 1 and 0: T0TAP[1:0]. These bits control the duration of the T0-T1 sequence (see Figure 10-2 on page 75) by selecting a tap from the Wakeup Timer.
‘00’ - 10 µs
‘01’ - 14 µs
‘10’ - 20 µs
‘11’ - 30 µs

For additional information, refer to the SLP_CFG3 register on page 285.

10.3.5 Related Registers

- INT_MSK0 Register on page 51.
- OSC_CR0 Register on page 113.
- ILO_TR Register on page 68.
- CPU_SCR0 Register on page 138.
- CPU_SCR1 Register on page 137.

The comparator digital interface performs logic processing on one or more comparator signals, provides a latching capability, and routes the result to other chip subsystems. The comparator signal is routed through a lookup table (LUT) function. The other input to the LUT is the neighboring comparator output. The LUT implements 1 of 16 functions on the two inputs, as selected by the CMP_LUT register. The LUT output also feeds the set input upon a reset/set (RS) latch. The latch is cleared by writing a ‘0’ to the appropriate bit in the CMP_RDC register or by a rising edge from the other comparator LUT.

The primary output for each comparator is the LUT output or its latched version. These are routed to the TrueTouch logic and to the interrupt controller. The comparator LUT output state and latched state are directly read by the CPU through the CMP_RDC register. A selection of comparator state may also be driven to an output pin.

When disabled, the comparators consume no power. Two active modes provide a full rail-to-rail input range or a somewhat lower power option with limited input range.

14.2.5 OSC_CR2 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,E2h	OSC_CR2				CLK48MEN		EXTCLKEN	IMODIS		RW : 00

The Oscillator Control Register 2 (OSC_CR2) configures various features of internal clock sources and clock nets.

Bit 4: CLK48MEN. This is the 48 MHz clock enable bit.

'0' disables the bit and '1' enables the bit. This register setting applies only when the device is **not** in OCD mode. When in OCD mode, the 48 MHz clock is always active.

15.3.8 I2C_BUF Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,CFh	I2C_BUF	Data Buffer[7:0]								RW : 00

The I^2C Data Buffer Register (I2C_BUF) is the CPU read/write interface to the data buffer. Whenever this register is read, the data at the location pointed to by the CPU current pointer (CPU_CP) is returned. Similarly, whenever this register is written, the data is transferred to the buffer and written at the location pointed to by the CPU current pointer (CPU_CP). Whenever this register is read, without initializing the RAM contents either through the I^2C or CPU interface, no valid value is returned.

The M8C accesses the data buffer through this register interface (I2C_BUF). Since the M8C issues an I/O read signal before an I/O write for some opcodes, the pointers (CPU_CP) to the data buffer increment unnecessarily because of an extra I/O read before an I/O write. Hence, all the I/O reads that occur in FIRST phase are ignored. The I/O reads that occur only in EXEC phase are taken as actual I/O reads for this register.

So, only the following basic M8C register access instructions may be used for accessing this register:

```
MOV A, reg[expr]
MOV A, reg[X+expr]
MOV [expr], [expr]
MOV reg[expr], A
MOV reg[X+expr], A
MOV reg[expr], expr
MOV reg[X+expr], expr
```

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

Bits 7 to 0: Data Buffer[7:0]. The I^2C Data Buffer Register (I2C_BUF) is the CPU read/write interface to the data buffer. Whenever this register is read, the data at the location pointed to by the CPU current pointer (CPU_CP) is returned. Similarly, whenever this register is written, the data is transferred to the buffer and written at the location pointed to by the CPU current pointer (CPU_CP). Whenever this register is read, without initializing the RAM contents either through the I^2C or CPU interface, no valid value is returned.

For additional information, refer to the I2C_BUF register on page 233.

15.3.10 I2C_SCR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D7h	I2C_SCR	Bus Error		Stop Status	ACK	Address	Transmit	LRB	Byte Complete	# : 00

LEGEND
Access is bit specific.

The I^2C Status and Control Register (I2C_SCR) is used by the slave to control the flow of data bytes and to keep track of the bus state during a transfer.

This register contains status bits to determine the state of the current I^2C transfer, and control bits to determine the actions for the next byte transfer. At the end of each byte transfer, the I^2C hardware interrupts the M8C microcontroller and stalls the I^2C bus on the subsequent low of the clock, until the PSoC device intervenes with the next command. This register may be read as many times as necessary; but on a subsequent write to this register, the bus stall is released and the current transfer continues.

There are five status bits: Byte Complete, LRB, Address, Stop Status, and Bus Error. These bits have Read/Clear (RC) access, which means that they are set by hardware but may be cleared by a write of '0' to the bit position. Under certain conditions, status is cleared automatically by the hardware.

There are two control bits: Transmit and ACK. These bits have RW access and may be cleared by hardware.

Bit 7: Bus Error. The Bus Error status detects misplaced Start or Stop conditions on the bus. These may be due to noise, rogue devices, or other devices that are not yet synchronized with the I^2C bus traffic. According to the I^2C specification, all compatible devices must reset their interface upon a received Start or Stop. This is a natural thing to do in slave mode because a Start initiates an address reception and a Stop idles the slave.

A bus error is defined as follows. A Start is only valid if the block is idle or a slave receiver is ready to receive the first bit of a new byte after an ACK. Any other timing for a Start condition sets the Bus Error bit. A Stop is only valid if the block is idle or a slave receiver is ready to receive the first bit of a new byte after an ACK. Any other timing for a Stop condition sets the Bus Error bit.

Bit 5: Stop Status. Stop status is set upon detection of an I^2C Stop condition. This bit is sticky, which means that it remains set until a '0' is written back to it by the firmware. This bit may only be cleared if the Byte Complete status bit is set. If the Stop Interrupt Enable bit is set, an interrupt is also generated upon Stop detection. It is never automatically cleared. Using this bit, a slave can distinguish between a previous Stop or Restart upon a given address byte interrupt.

The selections are shown in the following table:

Bit	Access	Description
7	RC	Bus Error 1 = A misplaced Start or Stop condition was detected. This status bit must be cleared by firmware with a write of '0' to the bit position. It is never cleared by the hardware.
5	RC	Stop Status 1 = A Stop condition was detected. This status bit must be cleared by firmware with a write of '0' to the bit position. It is never cleared by the hardware.
4	RW	ACK: Acknowledge Out 0 = NACK the last received byte. 1 = ACK the last received byte. This bit is automatically cleared by hardware upon the following byte complete event.
3	RC	Address 1 = The transmitted or received byte is an address. This status bit must be cleared by firmware with a write of '0' to the bit position.
2	RW	Transmit 0 = Receive Mode. 1 = Transmit Mode. This bit is set by firmware to define the direction of the byte transfer. Any Start detect automatically clears this bit.
1	RC	LRB: Last Received Bit The value of the ninth bit in a transmit sequence, which is the acknowledge bit from the receiver. 0 = Last transmitted byte was ACK'ed by the receiver. 1 = Last transmitted byte was NACK'ed by the receiver. Any Start detect automatically clears this bit.
0	RC	Byte Complete Transmit Mode: 1 = 8 bits of data have been transmitted and an ACK or NACK has been received. Receive Mode: 1 = 8 bits of data have been received. Any Start detect automatically clears this bit.

Bit 4: ACK.

This control bit defines the acknowledge data bit that is transmitted out in response to a received byte. When receiving, a byte complete interrupt is generated after the eighth data bit is received. Upon the subsequent write to this register to continue (or terminate) the transfer, the state of this bit determines the next transmitted data bit. It is ***active high***. A '1' sends an ACK and a '0' sends a NACK. A slave receiver sends a NACK to inform the master that it cannot receive any more bytes.

Bit 3: Address.

This bit is set when an address is received. This consists of a Start or Restart, and an address byte.

In slave mode when this status is set, firmware reads the received address from the data register and compares it with its own address. If the address does not match, the firmware writes a NACK indication to this register. No further interrupts occur until the next address is received. If the address does match, firmware must ACK the received byte, then byte complete interrupts are generated on subsequent bytes of the transfer.

When in I^2C compatibility mode the Address bit should not be cleared between a start and the next byte complete. If it is cleared during this time the start status is cleared internally.

Bit 2: Transmit. This bit sets the direction of the shifter for a subsequent byte transfer. The shifter is always shifting in data from the I^2C bus, but a write of '1' enables the output of the shifter to drive the SDA output line. Since a write to this register initiates the next transfer, data must be written to the data register before writing this bit. In receive mode, the previously received data must have been read from the data register before this write. Firmware derives this direction from the RW bit in the received slave address.

This direction control is only valid for data transfers. The direction of address bytes is determined by the hardware.

Bit 1: LRB. Last Received Bit. This is the last received bit in response to a previously transmitted byte. In transmit mode, the hardware sends a byte from the data register and clock in an acknowledge bit from the receiver. Upon the subsequent byte complete interrupt, firmware checks the value of this bit. A '0' is the ACK value and a '1' is a NACK value.

The meaning of LRB depends upon the current operating mode.

'0': ACK. The master wants to read another byte. The slave loads the next byte into the I2C_DR Register and sets the Transmit bit in the I2C_SCR register to continue the transfer.

'1': NACK. The master is done reading bytes. The slave reverts to IDLE state on the subsequent I2C_SCR write (regardless of the value written).

Bit 0: Byte Complete. The I^2C hardware operates on a byte basis. In transmit mode, this bit is set and an interrupt is generated at the end of nine bits (the transmitted byte + the received ACK). In receive mode, the bit is set after the eight bits of data are received. When this bit is set, an interrupt is generated at these data sampling points, which are associated with the SCL input clock rising (see details in Timing Diagrams on page 130). If the PSoC device responds with a write back to this register before the subsequent falling edge of SCL (which is approximately one-half bit time), the transfer continues without interruption. However, if the device cannot respond within that time, the hardware holds the SCL line low, stalling the I^2C bus. A subsequent write to the I2C_SCR register releases the stall.

For additional information, refer to the I2C_SCR register on page 240.

15.3.11 I2C_DR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D8h	I2C_DR	Data[7:0]								RW : 00

The I^2C Data Register (I2C_DR) provides read/write access to the Shift register.

Bits 7 to 0: Data[7:0]. This register is not buffered; therefore, writes and valid data reads occur at specific points in the transfer. These cases are outlined as follows:

- **Slave Receiver** – Data in the I2C_DR register is only valid for reading when the Byte Complete status bit is set. Data bytes must be read from the I2C_DR register before writing to the I2C_SCR Register, which continues the transfer.
- **Slave Transmitter** – Data bytes must be written to the I2C_DR register before the Transmit bit is set in the I2C_SCR Register, which continues the transfer.

For additional information, refer to the I2C_DR register on page 241.

16.3.2 CPU_SCR0 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
x,FFh	CPU_SCR0	GIES		WDRS	PORS	Sleep			STOP	# : XX

LEGEND
Access is bit specific. Refer to register detail for additional information.
XX The reset value is 10h after POR/XRES and 20h after a watchdog reset.

The System Status and Control Register 0 (CPU_SCR0) is used to convey the status and control of events for various functions of a PSoC device.

Bit 7: GIES. Global Interrupt Enable Status. This bit is a read only status bit and its use is discouraged. The GIES bit is a legacy bit that was used to provide the ability to read the GIE bit of the CPU_F register. However, the CPU_F register is now readable. When this bit is set, it indicates that the GIE bit in the CPU_F register is also set, which in turn, indicates that the microprocessor services interrupts.

Bit 5: WDRS. WatchDog Reset Status. This bit may not be set. It is normally '0' and automatically set whenever a watchdog reset occurs. The bit is readable and clearable by writing a '0' to its bit position in the CPU_SCR0 register.

Bit 4: PORS. Power On Reset Status. This bit, which is the watchdog enable bit, is set automatically by a POR or External Reset (XRES). If the bit is cleared by user code, the watchdog timer is enabled. After cleared, the only way to reset the PORS bit is to go through a POR or XRES. Thus, there is no way to disable the watchdog timer other than to go through a POR or XRES.

Bit 3: Sleep. This bit is used to enter Low Power Sleep mode when set. To wake up the system, this register bit is cleared asynchronously by any enabled interrupt. There are two special features of this bit that ensure proper sleep operation. First, the write to set the register bit is blocked if an interrupt is about to be taken on that instruction boundary (immediately after the write). Second, there is a hardware interlock to ensure that, once set, the SLEEP bit may not be cleared by an incoming interrupt until the sleep circuit has finished performing the sleep sequence and the system-wide power down signal has been asserted. This prevents the sleep circuit from being interrupted in the middle of the process of system power down, possibly leaving the system in an indeterminate state.

Bit 0: STOP. This bit is readable and writeable. When set, the PSoC M8C stops executing code until a reset event occurs. This can be either a POR, WDR, or XRES. If an application wants to stop code execution until a reset, the preferred method is to use the `HALT` instruction rather than a register write to this bit.

For additional information, refer to the CPU_SCR0 register on page 258.

20. Full-Speed USB

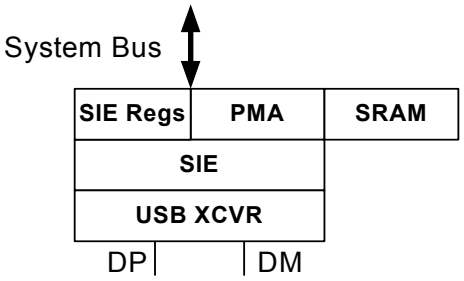
This chapter explains the Full-Speed USB (Universal Serial Bus) resource and its associated registers. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 187.

20.1 Architectural Description

The PSoC USB system resource adheres to the USB 2.0 Specification for full-speed devices operating at 12 Mbps with one upstream port and one USB address. PSoC USB consists of these components:

- Serial Interface Engine (SIE) block
- PSoC Memory Arbiter (PMA) block
- 512 bytes of dedicated SRAM
- A Full-Speed USB Transceiver with internal regulator and two dedicated USB pins

Figure 20-1. USB Block Diagram



At the PSoC system level, the full-speed USB system resource interfaces to the rest of the PSoC by way of the M8C's register access instructions and to the outside world by way of the two USB pins. The SIE supports nine endpoints including a bidirectional control endpoint (endpoint 0) and eight uni-directional data endpoints (endpoints 1 to 8). The uni-directional data endpoints are individually configurable as either IN or OUT.

20.2 Application Description

The individual components and issues of the USB system are described in detail in the following sections.

20.2.1 USB SIE

The USB Serial Interface Engine (SIE) allows the PSoC device to communicate with the USB host at full-speed data rates (12 Mbps). The SIE simplifies the interface to USB traffic by automatically handling the following USB processing tasks without firmware intervention:

- Translates the encoded received data and formats the data to be transmitted on the bus.
- Generates and checks CRCs. Incoming packets failing checksum verification are ignored.
- Checks addresses. Ignores all transactions not addressed to the device.
- Sends appropriate ACK/NAK/Stall handshakes.
- Identifies token type (SETUP, IN, OUT) and sets the appropriate token bit once a valid token in received.
- Identifies Start-of-Frame (SOF) and saves the frame count.
- Sends data to or retrieves data from the USB SRAM, by way of the PSoC Memory Arbiter (PMA).

Firmware is required to handle various parts of the USB interface. The SIE issues interrupts after key USB events to direct firmware to appropriate tasks:

- Fill and empty the USB data buffers in USB SRAM.
- Enable PMA channels appropriately.
- Coordinate enumeration by decoding USB device requests.
- Suspend and resume coordination.
- Verify and select data toggle values.

Table 20-1. Mode Encoding for Control and Non-Control Endpoints

Mode	Encoding	SETUP	IN	OUT	Comments
Disable	0000	Ignore	Ignore	Ignore	Ignore all USB traffic to this endpoint.
NAK IN/OUT	0001	Accept	NAK	NAK	NAK IN and OUT token.
Status OUT Only	0010	Accept	STALL	Check	For control endpoint, STALL IN and ACK zero byte OUT.
STALL IN/OUT	0011	Accept	STALL	STALL	For control endpoint, STALL IN and OUT token.
Reserved	0100	Ignore	Ignore	Ignore	
ISO OUT	0101	Ignore	Ignore	Always	Isochronous OUT.
Status IN Only	0110	Accept	TX 0 Byte	STALL	For control endpoint, STALL OUT and send zero byte data for IN token.
ISO IN	0111	Ignore	TX Count	Ignore	Isochronous IN.
NAK OUT	1000	Ignore	Ignore	NAK	Send NAK handshake to OUT token.
ACK OUT (STALL = 0)	1001	Ignore	Ignore	ACK	This mode is changed by the SIE to mode 1000 on issuance of ACK handshake to an OUT.
ACK OUT (STALL = 1)	1001	Ignore	Ignore	STALL	STALL the OUT transfer.
Reserved	1010	Ignore	Ignore	Ignore	
ACK OUT – STATUS IN	1011	Accept	TX0 Byte	ACK	ACK the OUT token or send zero byte data for IN token.
NAK IN	1100	Ignore	NAK	Ignore	Send NAK handshake for IN token.
ACK IN (STALL = 0)	1101	Ignore	TX Count	Ignore	This mode is changed by the SIE to mode 1100 after receiving ACK handshake to an IN data.
ACK IN (STALL = 1)	1101	Ignore	STALL	Ignore	STALL the IN transfer.
Reserved	1110	Ignore	Ignore	Ignore	
ACK IN – Status OUT	1111	Accept	TX Count	Check	Respond to IN data or Status OUT.

20.2.2 USB SRAM

The PSoC USB System Resource contains dedicated 512 bytes SRAM. This SRAM is identical to 2 SRAM pages used in the PSoC Core; however, it is not accessible by way of the M8C memory access instructions. The USB's dedicated SRAM may only be accessed by way of the PMA registers. For more information on how to use the USB's dedicated SRAM, see the next section, PSoC Memory Arbiter (PMA).

The USB SRAM contents are not directly affected by any reset, but must be treated as unknown after any POR, WDR, and XRES.

20.2.2.1 PSoC Memory Arbiter

The PSoC Memory Arbiter (PMA) is the interface between the USB's dedicated SRAM and the two blocks that access the SRAM: the M8C and the USB SIE. The PMA provides 16 channels to manage data with four Endpoints/8 channels mapped to Page 0 of USB dedicated SRAM and other four Endpoints/8 channels mapped to Page 1 of USB dedicated SRAM. All of the channel registers may be used by the M8C, but the eight non-control USB endpoints are each allocated to a specific set of PMA channel registers. It is the responsibility of the firmware to ensure that the M8C is not accessing a set of channel registers that are in use by the USB SIE.

If the M8C wants to access the same data that an SIE channel is using, two channels must be configured to access the same SRAM address ranges. Table 20-2 shows the mapping between PMA channels and which blocks can use them.

Table 20-2. PMA Channel Assignments

PMA Channel	USB SIE	M8C	Channel Registers (PMAx_xx)
0		Page 0	PMA0_DR, PMA0_RA, PMA0_WA
1	EP1	Page 0	PMA1_DR, PMA1_RA, PMA1_WA
2	EP2	Page 0	PMA2_DR, PMA2_RA, PMA2_WA
3	EP3	Page 0	PMA3_DR, PMA3_RA, PMA3_WA
4	EP4	Page 0	PMA4_DR, PMA4_RA, PMA4_WA
5		Page 0	PMA5_DR, PMA5_RA, PMA5_WA
6		Page 0	PMA6_DR, PMA6_RA, PMA6_WA
7		Page 0	PMA7_DR, PMA7_RA, PMA7_WA
8		Page 1	PMA8_DR, PMA8_RA, PMA8_WA
9	EP5	Page 1	PMA9_DR, PMA9_RA, PMA9_WA
10	EP6	Page 1	PMA10_DR, PMA10_RA, PMA10_WA
11	EP7	Page 1	PMA11_DR, PMA11_RA, PMA11_WA
12	EP8	Page 1	PMA12_DR, PMA12_RA, PMA12_WA
13		Page 1	PMA13_DR, PMA13_RA, PMA13_WA
14		Page 1	PMA14_DR, PMA14_RA, PMA14_WA
15		Page 1	PMA15_DR, PMA15_RA, PMA15_WA

20.3.5 EP0_CR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,36h	EP0_CR	Setup Received	IN Received	OUT Received	ACK'ed Transaction	Mode[3:0]				# : 00

The Endpoint Control Register (EP0_CR) is used to configure endpoint 0.

Because both firmware and the SIE are allowed to write to the Endpoint 0 Control and Count registers, the SIE provides an interlocking mechanism to prevent accidental overwriting of data. When the SIE writes to these registers they are locked and the processor cannot write to them until after reading the EP0_CR register. Writing to this register clears the upper four bits regardless of the value written.

Note The register lock is removed when the following M8C instructions are used to write the EP0_CR register as these instructions generate a read (IOR_) signal, which causes the lock to be removed even without the firmware actually reading the EP0_CR register.

```
mov reg[expr], expr
mov reg[X+expr], expr
```

Bit 7: Setup Received. When set, this bit indicates a valid setup packet was received and ACK'ed. This bit is forced high from the start of the data packet phase of the setup transaction, until the start of the ACK packet returned by the SIE. The CPU is prevented from clearing this bit during this interval. After this interval, the bit remains set until cleared by firmware. While this bit is set to '1', the CPU cannot write to the EP0_DRx registers. This prevents firmware from overwriting an incoming setup transaction before firmware has a chance to read the setup data. This bit is cleared by any non-locked writes to the register.

Bit 6: IN Received. When set, this bit indicates a valid IN packet has been received. This bit is updated to '1' after the host acknowledges an IN data packet. When clear, this bit indicates either no IN has been received or that the host did not acknowledge the IN data by sending an ACK handshake. It is cleared by any non-locked writes to the register.

Bit 5: OUT Received. When set, this bit indicates a valid OUT packet has been received and ACK'ed. This bit is updated to '1' after the last received packet in an OUT transaction. When clear, this bit indicates no OUT has been received. It is cleared by any non-locked writes to the register.

Bit 4: ACK'ed Transaction. This bit is set whenever the SIE engages in a transaction to the register's endpoint that completes with an ACK packet. This bit is cleared by any non-locked writes to the register.

Bits 3 to 0: Mode[3:0]. The mode bits control how the USB SIE responds to traffic and how the USB SIE changes the mode of that endpoint as a result of host packets to the endpoint.

For additional information, refer to the EP0_CR register on page 198.

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.

21.3.7 USB_SOF1

USB Start-of-Frame Register 1

Individual Register Names and Addresses:

USB_SOF1 : 0,32h

	7	6	5	4	3	2	1	0
Access : POR						R : 0		
Bit Name						Frame Number[10:8]		

This register is a USB Start-of-Frame register 1.

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 171 in the Full-Speed USB chapter.

Bit	Name	Description
2:0	**Frame Number[10:8]**	Contains the upper three bits of the frame number.

21.3.46 CUR_PP

Current Page Pointer Register

Individual Register Names and Addresses:

CUR_PP : 0,D0h

	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 normal memory accesses in a multi-SRAM page device. It is only used when a device has more than one SRAM page.

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 is used for generic SRAM access. See the RAM Paging chapter on page 39 for more information. 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

21.4.8 TMP_DRx

Temporary Data Registers

Individual Register Names and Addresses:

TMP_DR0 : x,6Ch TMP_DR1 : x,6Dh TMP_DR2 : x,6Eh TMP_DR3 : x,6Fh

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

These registers enhance the performance in multiple SRAM page PSoC devices.

All bits in this register are reserved for PSoC devices with 256 bytes of SRAM. For additional information, refer to the Register Definitions on page 42 in the RAM Paging chapter.

Bit	Name	Description
7:0	**Data[7:0]**	General purpose register space

bit rate (BR) The number of bits occurring per unit of time in a bit stream, usually expressed in bits per second (bps).

block
1. A functional unit that performs a single function, such as an oscillator.
2. A functional unit that may be configured to perform one of several functions, such as a digital block or an analog block.

Boolean Algebra In mathematics and computer science, Boolean algebras or Boolean lattices, are algebraic structures which "capture the essence" of the logical operations AND, OR and NOT as well as set the theoretic operations union, intersection, and complement. Boolean algebra also defines a set of theorems that describe how Boolean equations can be manipulated. For example, these theorems are used to simplify Boolean equations which reduces the number of logic elements needed to implement the equation.

The operators of Boolean algebra may be represented in various ways. Often they are simply written as AND, OR, and NOT. In describing circuits, NAND (NOT AND), NOR (NOT OR), XNOR (exclusive NOT OR), and XOR (exclusive OR) may also be used. Mathematicians often use + (for example, A+B) for OR and • for AND (for example, A*B) (since in some ways those operations are analogous to addition and multiplication in other algebraic structures) and represent NOT by a line drawn above the expression being negated (for example, ~A, A_, !A).

break-before-make The elements involved go through a disconnected state entering (‘break”) before the new connected state (“make”).

broadcast net A signal that is routed throughout the microcontroller and is accessible by many blocks or systems.

buffer
1. A storage area for data that is used to compensate for a speed difference, when transferring data from one device to another. Usually refers to an area reserved for I/O operations into which data is read or from which data is written.
2. A portion of memory set aside to store data, often before it is sent to an external device or as it is received from an external device.
3. An amplifier used to lower the output impedance of a system.

bus
1. A named connection of nets. Bundling nets together in a bus makes it easier to route nets with similar routing patterns.
2. A set of signals performing a common function and carrying similar data. Typically represented using vector notation; for example, address[7:0].
3. One or more conductors that serve as a common connection for a group of related devices.

byte A digital storage unit consisting of 8 bits.

C

C A high level programming language.

capacitance A measure of the ability of two adjacent conductors, separated by an insulator, to hold a charge when a voltage differential is applied between them. Capacitance is measured in units of Farads.

capture To extract information automatically through the use of software or hardware, as opposed to hand-entering of data into a computer file.