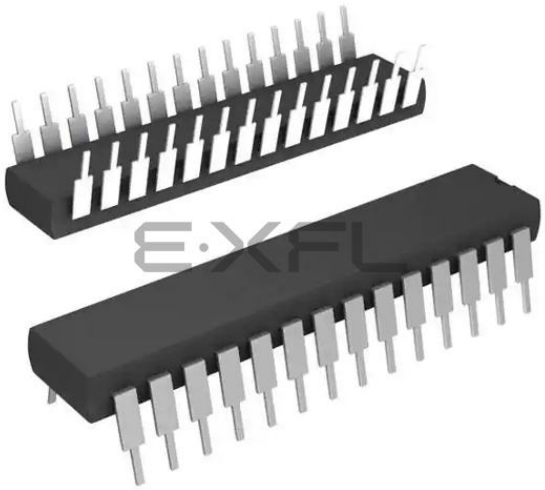

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	M8C
Core Size	8-Bit
Speed	24MHz
Connectivity	SPI, UART/USART
Peripherals	LVD, POR, PWM, WDT
Number of I/O	24
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.25V
Data Converters	A/D 1x8b, 1x11b, 1x12b; D/A 1x9b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy8c26443-24pxi

The PSoC™ CY8C25122/CY8C26233/CY8C26443/CY8C26643 family of programmable system-on-chip devices replace multiple MCU-based system components with one single-chip, configurable device. A PSoC device includes configurable analog and digital peripheral blocks, a fast CPU, Flash program memory, and SRAM data memory in a range of convenient pin-outs and memory sizes. The driving force behind this innovative programmable system-on-chip comes from user configurability of the analog and digital arrays: the PSoC blocks.

Programmable System-on-Chip (PSoC™) Blocks

- On-chip, user configurable analog and digital peripheral blocks
- PSoC blocks can be used individually or in combination
- 12 Analog PSoC blocks provide:
  - Up to 11-bit Delta-Sigma ADC
  - Up to 8-bit Successive Approximation ADC
  - Up to 12-bit Incremental ADC
  - Up to 9-bit DAC
  - Programmable gain amplifier
  - Programmable filters
  - Differential comparators
- 8 Digital PSoC blocks provide:
  - Multipurpose timers: event timing, real-time clock, pulse width modulation (PWM) and PWM with deadband
  - CRC modules
  - Full-duplex UARTs
  - SPI™ master or slave configuration
  - Flexible clocking sources for analog PSoC blocks

Powerful Harvard Architecture Processor with Fast Multiply/Accumulate

- M8C processor instruction set
- Processor speeds to 24 MHz
- Register speed memory transfers
- Flexible addressing modes
- Bit manipulation on I/O and memory
- 8x8 multiply, 32-bit accumulate

Flexible On-Chip Memory

- Flash program storage, 4K to 16K bytes, depending on device
- 50,000 erase/write cycles
- 256 bytes SRAM data storage
- In-System Serial Programming (ISSP™)
- Partial Flash updates
- Flexible protection modes
- EEPROM emulation in Flash, up to 2,304 bytes

Programmable Pin Configurations

- Schmitt trigger TTL I/O pins
- Logic output drive to 25 mA with internal pull-up or pull-down resistors, High Z, or strong driver
- Interrupt on pin change
- Analog output drive to 40 mA

Precision, Programmable Clocking

- Internal 24/48 MHz Oscillator (+/- 2.5%, no external components)
- External 32.768 kHz Crystal Oscillator (optional precision source for PLL)
- Internal Low Speed Oscillator for Watchdog and Sleep

Dedicated Peripherals

- Watchdog and Sleep Timers
- Low Voltage Detection with user-configurable threshold voltages
- On-chip voltage reference

Fully Static CMOS Devices using advanced Flash technology

- Low power at high speed
- Operating voltage from 3.0 to 5.25 V
- Operating voltage down to 1.0 V using on-chip switch mode voltage pump
- Wide temperature range: -40 °C to + 85 °C

Complete Development Tools

- Powerful integrated development environment (PSoC™ Designer)
- Low-cost, in-circuit emulator and programmer

Table 6: Pin-out 48 Pin, continued

P5[0]	I/O	29	Port 5[0]
P5[2]	I/O	30	Port 5[2]
P4[0]	I/O	31	Port 4[0]
P4[2]	I/O	32	Port 4[2]
P4[4]	I/O	33	Port 4[4]
P4[6]	I/O	34	Port 4[6]
XRES	I	35	External Reset
P3[0]	I/O	36	Port 3[0]
P3[2]	I/O	37	Port 3[2]
P3[4]	I/O	38	Port 3[4]
P3[6]	I/O	39	Port 3[6]
P2[0]	I/O	40	Port 2[0] (Non-Multiplexed Analog Input)
P2[2]	I/O	41	Port 2[2] (Non-Multiplexed Analog Input)
P2[4]	I/O	42	Port 2[4] / External AGNDIn
P2[6]	I/O	43	Port 2[6] / External VREFIn
P0[0]	I/O	44	Port 0[0] (Analog Input)
P0[2]	I/O	45	Port 0[2] (Analog Input/Output)
P0[4]	I/O	46	Port 0[4] (Analog Input/Output)
P0[6]	I/O	47	Port 0[6] (Analog Input)
Vcc	Power	48	Supply Voltage

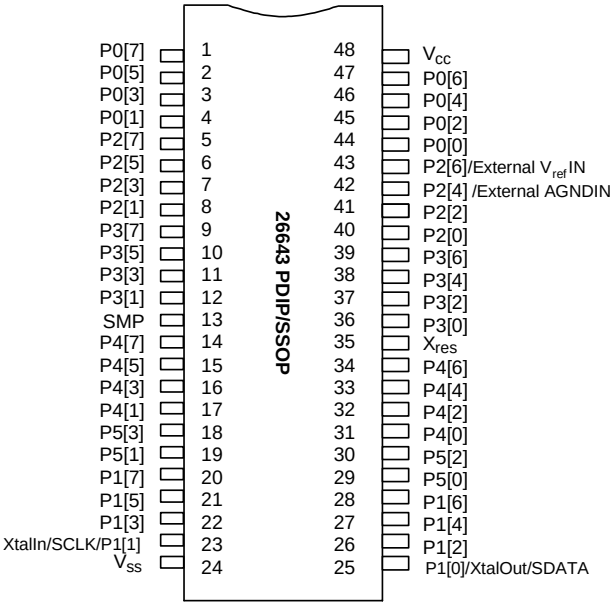

Figure 6: 26643 PDIP/SSOP

Examples:

```
                        ;In this case, the
                        ;value in the memory
                        ;location at address
                        ;7 is added with the
ADD    [7],     A      ;Accumulator, and the
                        ;result is placed in
                        ;the memory location
                        ;at address 7. The
                        ;Accumulator is
                        ;unchanged.
                        ;In this case, the
                        ;Accumulator is moved
                        ;to the register
MOV    REG[8], A        ;space location at
                        ;address 8. The
                        ;Accumulator is
                        ;unchanged.
```

2.3.5 Destination Indexed

The result of an instruction using this addressing mode is placed within either the RAM memory space or the register space. Operand 1 is added to the X register forming the address that points to the location of the result. The source for the instruction is the A register. Arithmetic instructions require two sources, the second source is the location specified by Operand 1 added with the X register. Instructions using this addressing mode are two bytes in length.

Table 17: Destination Indexed

Opcode	Operand 1
Instruction	Destination Index

Example:

```
                        ;In this case, the value
                        ;in the memory location
                        ;at address X+7 is added
                        ;with the Accumulator,
ADD  [X+7],   A         ;and the result is placed
                        ;in the memory location
                        ;at address x+7. The
                        ;Accumulator is
                        ;unchanged.
```

2.3.6 Destination Direct Immediate

The result of an instruction using this addressing mode is placed within either the RAM memory space or the register space. Operand 1 is the address of the result. The source for the instruction is Operand 2, which is an immediate value. Arithmetic instructions require two sources, the second source is the location specified by Operand 1. Instructions using this addressing mode are three bytes in length.

Table 18: Destination Direct Immediate

Opcode	Operand 1	Operand 2
Instruction	Destination Address	Immediate Value

Examples:

```
                        ;In this case, value in
                        ;the memory location at
                        ;address 7 is added to
ADD  [7],     5         ;the immediate value of
                        ;5, and the result is
                        ;placed in the memory
                        ;location at address 7.
                        ;In this case, the
                        ;immediate value of 6 is
MOV  REG[8], 6          ;moved into the register
                        ;space location at
                        ;address 8.
```

2.3.7 Destination Indexed Immediate

The result of an instruction using this addressing mode is placed within either the RAM memory space or the register space. Operand 1 is added to the X register to form the address of the result. The source for the instruction is Operand 2, which is an immediate value. Arithmetic instructions require two sources, the second source is the location specified by Operand 1 added with the X register. Instructions using this addressing mode are three bytes in length.

Table 19: Destination Indexed Immediate

Opcode	Operand 1	Operand 2
Instruction	Destination Index	Immediate Value

3.0 Memory Organization

3.1 Flash Program Memory Organization

Table 24: Flash Program Memory Map

Address	Description
0x0000	Reset Vector
0x0004	Supply Monitor Interrupt Vector
0x0008	DBA 00 PSoC Block Interrupt Vector
0x000C	DBA 01 PSoC Block Interrupt Vector
0x0010	DBA 02 PSoC Block Interrupt Vector
0x0014	DBA 03 PSoC Block Interrupt Vector
0x0018	DCA 04 PSoC Block Interrupt Vector
0x001C	DCA 05 PSoC Block Interrupt Vector
0x0020	DCA 06 PSoC Block Interrupt Vector
0x0024	DCA 07 PSoC Block Interrupt Vector
0x0028	Analog Column 0 Interrupt Vector
0x002C	Analog Column 1 Interrupt Vector
0x0030	Analog Column 2 Interrupt Vector
0x0034	Analog Column 3 Interrupt Vector
0x0038	GPIO Interrupt Vector
0x003C	Sleep Timer Interrupt Vector
0x0040	On-Chip User Program Memory Starts Here
	***
	***
	***
0x3FFF	16K Flash Maximum Depending on Version

3.2 RAM Data Memory Organization

The stack on this device grows from low addresses to high addresses. The Linker function within PSoC Designer locates the bottom of the stack after the end of Global Variables. This allows the stack to grow from just after the Global Variables until 0xFF. The stack will wrap back to 0x00 on an overflow condition.

Table 25: RAM Data Memory Map

Address	Description
0x00	First General Purpose RAM Location
0xXX	General Purpose RAM
0xXY	General Purpose RAM
0xXZ	Last General Purpose RAM Location
0xYX	Bottom of Hardware Stack
0xYY	⇓ Stack Grows This Way ⇓
0xFF	Top of Hardware Stack

4.0 Register Organization

4.1 Introduction

There are two register banks implemented on these devices. Each bank contains 256 addresses. The purpose of these register banks is to personalize and parameterize the on-chip resources as well as read and write data values.

The user selects between the two banks by setting the XIO bit in the CPU_F Flag Register.

In some cases, the same register is available on either bank, for convenience. These registers (71h to 9fh) can be accessed from either bank.

Note: All register addresses not shown are reserved and should never be written. In addition, unused or reserved bits in any register should always be written to 0.

9.0 Digital PSoC Blocks

9.1 Introduction

PSoC blocks are user configurable system resources. On-chip digital PSoC blocks reduce the need for many MCU part types and external peripheral components. Digital PSoC blocks can be configured to provide a wide variety of peripheral functions. PSoC Designer Software Integrated Development Environment provides automated configuration of PSoC blocks by simply selecting the desired functions. PSoC Designer then generates the proper configuration information and can print a device data sheet unique to that configuration.

Digital PSoC blocks provide up to eight, 8-bit multipurpose timers/counters supporting multiple event timers, real-time clocks, Pulse Width Modulators (PWM), and CRCs. In addition to all PSoC block functions, communication PSoC blocks support full-duplex UARTs and SPI master or slave functions.

As shown in Figure 12:, there are a total of eight 8-bit digital PSoC blocks in this device family configured as a linear array. Four of these are the Digital Basic Type A blocks and four are the Digital Communications Type A blocks. Each of these digital PSoC blocks can be configured independently, or used in combination.

Each digital PSoC block has a unique Interrupt Vector and Interrupt Enable bit. Functions can be stopped or started with a user-accessible Enable bit.

The Timer/Counter/CRC/PRS/Deadband functions are available on the Digital Basic Type A blocks and also the Digital Communications Type A blocks. The UART and SPI communications functions are only available on the Digital Communications Type A blocks.

There are three configuration registers: the Function Register (DBA00FN-DCA07FN) to select the block function and mode, the Input Register (DBA00IN-DCA07IN) to select data input and clock selection, and the Output Register (DBA00OU-DCA07OU) to select and enable function outputs.

The three data registers are designated Data 0 (DBA00DR0-DCA07DR0), Data 1 (DBA00DR1-DCA07DR1), and Data 2 (DBA00DR2-DCA07DR2). The function of these registers and their bit mapping is dependent on the overall block function selected by the user.

The one Control Register (DBA00CR0-DCA07CR0) is designated Control 0. The function of this register and its bit mapping is dependent on the overall block function selected by the user.

If the CPU frequency is 24 MHz and a PSoC timer/counter of 24-bits or longer is operating at 48 MHz, a write to the block Control Register to enable it (for example, a call to Timer_1_Start) may not start the block properly. In the failure case, the first count will typically be indeterminate as the upper bytes fail to make the first count correctly. However, on the first terminal count, the correct period will be loaded and counted thereafter.

Digital Communications Type A Block 06 Input Register (DCA06IN, Address = Bank 1, 39h)
Digital Communications Type A Block 07 Input Register (DCA07IN, Address = Bank 1, 3Dh)

The Data/Enable source select [3:0] bits select between multiple inputs to the Digital PSoC Blocks. These inputs serve as Clock Enables or Data Input depending on the Digital PSoC Block's programmed function. If "Chain Function to Previous" data input is selected for Data/Enable then the selected Digital PSoC block receives its Data, Enable, Zero Detect, and all chaining information from the previous digital PSoC block. The data inputs that are selected from the GPIO pins (through the Global Input Bus) are synchronized to the 24 MHz clock. The following table shows the function dependent meaning of the data input.

Table 49: Digital Function Data Input Definitions

Function	Data Input
Timer	Positive Edge Capture
Counter	Count Enable (Active High)
CRC	Data Input
PRS	N/A
Deadband	Kill Signal (Active High)
TX UART	N/A
RX UART	RX Data In
SPI Master	MISO (Master In/Slave Out)
SPI Slave	MOSI (Master Out/Slave In)

The Clock[3:0] bits select multiple sources for the clock for each digital PSoC block. The sources for each digital PSoC block clock are selected from the Global Input Bus, System Clocks, and other neighboring digital PSoC blocks. As shown in the table, Digital PSoC Blocks 0-3 can interface to Global I/Os 00-03, and Digital PSoC block 04-07 can interface to Global I/Os 4-7. It is important to note that clock inputs selected from the GPIO pins (through the Global Input Bus) are not synchronized. This may cause indeterminate results if the CPU reads a block register as it is changing in response to an external clock. CPU reads must be manually synchronized, either through the block interrupt, or through a multiple read and voting scheme.

9.2.3 Digital Basic Type A / Communications Type A Block xx Output Register

The digital PSoC block's outputs can be selected to drive associated Global Output Bus signals via the Output Select bits. In addition, the output drive can be selectively enabled in this register. The SPI Slave has an auxiliary input which is also controlled by selections in this register.

4. Capture vs. Compare

   A capture event will overwrite Data Register 2. This is also the register that holds the compare value. Therefore, using the capture function may not be compatible with using the timer compare function.

9.5.2 Counter with Optional Compare (Pulse-Width) Output

9.5.2.1 Summary

Conceptually, a counter measures the number of events between “ticks,” however, this distinction between counter and timer blurs because both functions provide a complete range of clock selections. The counter trades the timer’s hardware capture for a clock gate or ”enable” and provides a means of adjusting the duty cycle of its output so that it can double as a pulse-width modulator. A down counter lies at the heart of the counter function. Counter-configured PSoC blocks may be chained to arbitrary lengths in 8 bit increments.

In a Counter User Module, the data input is an enable for counting. Normally, when the enable goes low, the counter will hold the current count. However, if the enable happens to go low in the same clock period as Terminal Count (count of all 0's), one additional count will occur that will reload the counter from the Period Register. Once the counter is reloaded from the Period Register, counting will stop.

9.5.2.2 Registers

Data Register 1 establishes the period of the counter. Data Register 0 holds the current state of the down counter. If the function is disabled, writing a period into Data Register 1, will automatically load Data Register 0. It is also automatically reloaded on the clock cycle after it reaches zero, the terminal count value. The value in Data Register 2 (compare value) is continually compared to Data Register 0 (count value) to establish the output pulse-width (duty cycle). Reading Data Register 0 to obtain the current value of the down counter may occur only when the function is disabled. When read, this transfers the value from Data Register 0 to Data Register 2 and returns a 0 on the data bus. The value transferred to Data Register 2 can then be directly read by the CPU. However, reading the count value in this manner will overwrite any previously written compare value in Data Register 2. Control Register 0 contains one bit to enable/disable the function.

9.5.2.3 Inputs

There are two primary inputs, the Source Clock and the Enable signal. When the Enable signal is high, the down counter is decremented on the rising-edge of the Source Clock. The multiplexers selecting these inputs are controlled by the PSoC block Input Register (DBA00IN-DCA07IN).

9.5.2.4 Outputs

The counter function drives its primary output signal, Compare True, high on the falling edge of the Source Clock when the value in Data Register 0 is less (or less than or equal to) the value in Data Register 2. The duty cycle of the pulse-width modulator formed in this way is the ratio of Data Register 2 (or Data Register 2 minus one) to Data Register 1. The choice of compare operators is determined by the MODE[1] bit. The Compare value can be routed to additional analog or digital PSoC blocks or via Global Output lines The auxiliary output signal is the Terminal Count signal which can be routed via Global Output lines. The PSoC block Output Register (DBA00OU-DCA07OU) controls output options.

9.5.2.5 Interrupts

Interrupts may be generated in either of two ways. First, the PSoC block may optionally generate an interrupt on the rising edge of Terminal Count or the rising edge of the Compare signal. The selection of interrupt source is determined by the MODE[0] bit of the PSoC block Function Register (DBA00FN-DCA07FN). The MODE[1] bit controls whether the comparison operation is “less than” or “less than or equal to.”

9.5.2.6 Usage Notes

1. Enable Input

   The enable input is synchronous and when low forces the counter into a ‘hold’ state. Outputs are unaffected by the state of the enable input. If an external source is selected as the enable input, it is synchronized to the 24 MHz clock.

If the SPI Master block is being used to receive data, “dummy” bytes must be written to the TX Data Register in order to initiate transmission/reception of each byte.

9.5.8.3 Inputs

MISO (master-in, slave-out) is selected by the input multiplexer. The clock input multiplexer selects a clock that runs at twice the desired data rate. The SPIM function divides the input clock by 2 to obtain the 50% duty-cycle required for proper timing. The input multiplexer is controlled by the PSoC block Input Register (DCA04IN-DCA07IN).

9.5.8.4 Outputs

There are two outputs, both of which can be enabled onto the Global Output bus. The MOSI (master-out, slave-in) data line provides the output serial data. The second output is the bit-clock derived by dividing the input clock by 2 to ensure a 50% duty-cycle. The PSoC block Output Register (DCA04OU-DCA07OU) controls output options.

Note: The SPIM function does not provide the SS_ signal that may be used by a corresponding SPI Slave. However, this can be implemented with a GPIO pin and supporting firmware if desired.

9.5.8.5 Interrupts

When enabled, the function generates an interrupt on TX Reg Empty status (Data Register 1 empty). If Mode[1] in the Function Register is set, the SPI Master will generate an interrupt on SPI Complete.

9.5.8.6 Usage Notes

1. Reading the Status

   Reading Control Register 0, which contains the status bits, automatically resets the status bits to 0 with the exception of TX Reg Empty, which is cleared when a byte is written to the TX Data Register (Data Register 1), and the RX Reg Full, which is cleared when a byte is read from the RX Data Register (Data Register 2).

2. Using Interrupts

   TX Reg Empty status or optionally SPI Complete status generates the block interrupt. Executing the interrupt routine does not automatically clear status. If SPI Complete is selected as the interrupt source, Control Register 0 (status) must be read in the interrupt routine to clear the status. If TX Reg Empty status is selected, a byte must be written to the TX Data Register (Data Register 1) to clear the status. If the interrupting status is not cleared further interrupts will be suppressed.

9.5.9 SPI Slave - Serial Peripheral Interface (SPIS)

9.5.9.1 Summary

The SPI Slave function provides a full-duplex bi-directional synchronous data transceiver that requires an externally provided bit clock for the data. This function requires a Digital Communications Type PSoC block. It cannot be chained for longer data words. This Digital Communications Type PSoC block supports SPI modes for 0, 1, 2, and 3. See Figure 15: for waveforms of the supported modes.

9.5.9.2 Registers

Data Register 0 provides a shift register for both incoming and outgoing data. Output data is written to Data Register 1 (TX Data Register). Input data is read from Data Register 2 (RX Data Register). When Data Register 0 is empty, its value is updated from Data Register 1. As new data bits are shifted in, the transmit bits are shifted out. After the 8 bits are transmitted and received by Data Register 0, the received byte is transferred into Data Register 2 from which it can be read. Simultaneously, the next byte to transmit, if available, is transferred from Data Register 1 into Data Register 0. Control Register 0 (DCA04CR0-DCA07CR0) provides status information and configures the function for one of the four standard modes, which configure the interface based on clock polarity and phase with respect to data.

9.5.9.3 Inputs

The SPIS function has three inputs. The Input Register (DCA04IN-DCA07IN) controls the input multiplexer, which selects the MOSI data stream. It also controls the clock selection multiplexer from which the function obtains the master’s bit clock. The AUX-IO bits of the Output Register (DCA04OU-DCA07OU) select a Global Input signal from which the SS_ (Slave Select) signal is obtained. It is important to note that the SS_ signal can

10.5 Analog PSoC Block Clocking Options

All analog PSoC blocks in a particular Analog Column share the same clock signal. Choosing the clocking for an analog PSoC block is a two-step process.

1. First, if the user wants to use the **ACLK0** and **ACLK1** system-clocking signals, the digital PSoC blocks that serve as the source for these signals must be selected. This selection is made in the Analog Clock Select Register (CLK_CR1).
2. Next, the user must select the source for the **Acolumn0**, **Acolumn1**, **Acolumn2**, and **Acolumn3** system-clocking signals. The user will choose the clock for Acolumnx[1:0] bits in the Analog Column Clock Select Register (CLK_CR0). Each analog PSoC block in a particular Analog Column is clocked from the **Acolumn[x]** system-clocking signal for that column. (Note that the Acolumn[x] signals have a 1:4 divider on them.)

10.5.1 Analog Column Clock Select Register

Table 64: Analog Column Clock Select Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/ Write	RW	RW	RW	RW	RW	RW	RW	RW
Bit Name	Acolumn3 [1]	Acolumn3 [0]	Acolumn2 [1]	Acolumn2 [0]	Acolumn1 [1]	Acolumn1 [0]	Acolumn0 [1]	Acolumn0 [0]

Bit [7:6]: **Acolumn3 [1:0]**
0 0 = 24V1
0 1 = 24V2
1 0 = ACLK0
1 1 = ACLK1

Bit [5:4]: **Acolumn2 [1:0]**
0 0 = 24V1
0 1 = 24V2
1 0 = ACLK0
1 1 = ACLK1

Bit [3:2]: **Acolumn1 [1:0]**
0 0 = 24V1
0 1 = 24V2
1 0 = ACLK0
1 1 = ACLK1

Bit [1:0]: **Acolumn0 [1:0]**
0 0 = 24V1
0 1 = 24V2
1 0 = ACLK0
1 1 = ACLK1

Analog Column Clock Select Register (CLK_CR0, Address = Bank 1, 60h)

Table 71: Analog Switch Cap Type A Block xx Control 2 Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/ Write	RW	RW	RW	RW	RW	RW	RW	RW
Bit Name	AnalogBus	CompBus	AutoZero	CCap[4]	CCap[3]	CCap[2]	CCap[1]	CCap[0]

Bit 7: **AnalogBus** Enable output to the analog bus
0 = Disable output to analog column bus
1 = Enable output to analog column bus
(The output on the analog column bus is affected by the state of the ClockPhase bit in Control 0 Register (ASA10CR0, ASA12CR0, ASA21CR0, ASA23CR0). If AnalogBus is set to 0, the output to the analog column bus is tri-stated. If AnalogBus is set to 1, the signal that is output to the analog column bus is selected by the ClockPhase bit. If the ClockPhase bit is 0, the block output is gated by sampling clock on last part of PHI2. If the ClockPhase bit is 1, the block output continuously drives the analog column bus.)

Bit 6: **CompBus** Enable output to the comparator bus
0 = Disable output to comparator bus
1 = Enable output to comparator bus

Bit 5: **AutoZero** Bit for controlling gated switches
0 = Shorting switch is not active. Input cap branches shorted to op-amp input
1 = Shorting switch is enabled during internal PHI1. Input cap branches shorted to analog ground during internal PHI1 and to op-amp input during internal PHI2.

Bit [4:0]: **CCap [4:0]** Binary encoding for 32 possible capacitor sizes for C Capacitor:

0 0 0 0 0 = 0 Capacitor units in array	1 0 0 0 0 = 16 Capacitor units in array
0 0 0 0 1 = 1 Capacitor units in array	1 0 0 0 1 = 17 Capacitor units in array
0 0 0 1 0 = 2 Capacitor units in array	1 0 0 1 0 = 18 Capacitor units in array
0 0 0 1 1 = 3 Capacitor units in array	1 0 0 1 1 = 19 Capacitor units in array
0 0 1 0 0 = 4 Capacitor units in array	1 0 1 0 0 = 20 Capacitor units in array
0 0 1 0 1 = 5 Capacitor units in array	1 0 1 0 1 = 21 Capacitor units in array
0 0 1 1 0 = 6 Capacitor units in array	1 0 1 1 0 = 22 Capacitor units in array
0 0 1 1 1 = 7 Capacitor units in array	1 0 1 1 1 = 23 Capacitor units in array
0 1 0 0 0 = 8 Capacitor units in array	1 1 0 0 0 = 24 Capacitor units in array
0 1 0 0 1 = 9 Capacitor units in array	1 1 0 0 1 = 25 Capacitor units in array
0 1 0 1 0 = 10 Capacitor units in array	1 1 0 1 0 = 26 Capacitor units in array
0 1 0 1 1 = 11 Capacitor units in array	1 1 0 1 1 = 27 Capacitor units in array
0 1 1 0 0 = 12 Capacitor units in array	1 1 1 0 0 = 28 Capacitor units in array
0 1 1 0 1 = 13 Capacitor units in array	1 1 1 0 1 = 29 Capacitor units in array
0 1 1 1 0 = 14 Capacitor units in array	1 1 1 1 0 = 30 Capacitor units in array
0 1 1 1 1 = 15 Capacitor units in array	1 1 1 1 1 = 31 Capacitor units in array

Analog Switch Cap Type A Block 10 Control 2 Register (ASA10CR2, Address = Bank 0/1, 82h)
Analog Switch Cap Type A Block 12 Control 2 Register (ASA12CR2, Address = Bank 0/1, 8Ah)
Analog Switch Cap Type A Block 21 Control 2 Register (ASA21CR2, Address = Bank 0/1, 96h)
Analog Switch Cap Type A Block 23 Control 2 Register (ASA23CR2, Address = Bank 0/1, 9Eh)

10.9.2.2 Analog Switch Cap Type B Block xx Control 1 Register

AMux controls the input muxing for the A capacitor branch.

The BCap bits set the value of the capacitor in the B path.

Table 74: Analog Switch Cap Type B Block xx Control 1 Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/ Write	RW	RW	RW	RW	RW	RW	RW	RW
Bit Name	AMux[2]	AMux[1]	AMux[0]	BCap[4]	BCap[3]	BCap[2]	BCap[1]	BCap[0]

Bit [7:5]: **AMux [2:0]** Input muxing select for A capacitor branch. (Note that available mux inputs vary by individual PSoC block.)

ASB11	ASB13	ASB20	ASB22
0 0 0 = ACA01	ACA03	ASA10	ASA12
0 0 1 = ASA12	P2.2	P2.1	ASA21
0 1 0 = ASA10	ASA12	ASA21	ASA23
0 1 1 = ASA21	ASA23	ABUS0	ABUS2
1 0 0 = REFHI	REFHI	REFHI	REFHI
1 0 1 = ACA00	ACA02	ASB11	ASB13
1 1 0 = Reserved	Reserved	Reserved	Reserved
1 1 1 = Reserved	Reserved	Reserved	Reserved

Bit [4:0]: BCap [4:0] Binary encoding for 32 possible capacitor sizes for B Capacitor:

0 0 0 0 0 = 0 Capacitor units in array	1 0 0 0 0 = 16 Capacitor units in array
0 0 0 0 1 = 1 Capacitor units in array	1 0 0 0 1 = 17 Capacitor units in array
0 0 0 1 0 = 2 Capacitor units in array	1 0 0 1 0 = 18 Capacitor units in array
0 0 0 1 1 = 3 Capacitor units in array	1 0 0 1 1 = 19 Capacitor units in array
0 0 1 0 0 = 4 Capacitor units in array	1 0 1 0 0 = 20 Capacitor units in array
0 0 1 0 1 = 5 Capacitor units in array	1 0 1 0 1 = 21 Capacitor units in array
0 0 1 1 0 = 6 Capacitor units in array	1 0 1 1 0 = 22 Capacitor units in array
0 0 1 1 1 = 7 Capacitor units in array	1 0 1 1 1 = 23 Capacitor units in array
0 1 0 0 0 = 8 Capacitor units in array	1 1 0 0 0 = 24 Capacitor units in array
0 1 0 0 1 = 9 Capacitor units in array	1 1 0 0 1 = 25 Capacitor units in array
0 1 0 1 0 = 10 Capacitor units in array	1 1 0 1 0 = 26 Capacitor units in array
0 1 0 1 1 = 11 Capacitor units in array	1 1 0 1 1 = 27 Capacitor units in array
0 1 1 0 0 = 12 Capacitor units in array	1 1 1 0 0 = 28 Capacitor units in array
0 1 1 0 1 = 13 Capacitor units in array	1 1 1 0 1 = 29 Capacitor units in array
0 1 1 1 0 = 14 Capacitor units in array	1 1 1 1 0 = 30 Capacitor units in array
0 1 1 1 1 = 15 Capacitor units in array	1 1 1 1 1 = 31 Capacitor units in array

Analog Switch Cap Type B Block 11 Control 1 Register (ASB11CR1, Address = Bank 0/1, 85h)
Analog Switch Cap Type B Block 13 Control 1 Register (ASB13CR1, Address = Bank 0/1, 8Dh)
Analog Switch Cap Type B Block 20 Control 1 Register (ASB20CR1, Address = Bank 0/1, 91h)
Analog Switch Cap Type B Block 22 Control 1 Register (ASB22CR1, Address = Bank 0/1, 99h)

The SAR hardware accelerator is a block of specialized hardware designed to sequence the SAR algorithm for efficient A/D conversion. A SAR ADC is implemented conceptually with a DAC of the desired precision, and a comparator. This functionality can be configured from one or more PSoC blocks. For each conversion, the firmware should initialize the ASY_CR register as defined below, and set the sign bit of the DAC as the first guess in the algorithm. A sequence of OR instructions (Read, Modify, Write) to the DAC (CR0) register is then executed. Each of these OR instructions causes the SAR hardware to read the current state of the comparator, checking the validity of the previous guess. It either clears it or leaves it set, accordingly. The next LSB in the DAC register is also set as the next guess. Six OR instructions will complete the conversion of a 6-bit DAC. The resulting DAC code, which matches the input voltage to within 1 LSB, is then read back from the DAC CR0 register.

10.11.1 Analog Stall and Analog Stall Lockup

Stall lockup affects the operation of stalled IO writes, such as DAC writes and the stalled IOR of the SAR hardware accelerator. The DAC and SAR User Modules operate in this mode. The analog column clock frequency must not be a power of two multiple (2, 4, 8...) higher than the CPU clock frequency. Under this condition, the CPU will never recover from a stall.

See the list of relationships (in MHz) that will fail:

Table 78: Analog Frequency Relationships

Analog Column Clock	CPU Clock
3.	1.5, 0.75, .018, 0.093
1.5	0.75, 0.18, 0.093
0.75	0.18, 0.093
0.37	0.18, 0.093
0.18	0.093

You can still run the CPU clock slower than the column clock if the relationship is not a power of two multiple. For example, you can run at 0.6 MHz, which is not a power of two multiple of any CPU frequency and therefore any CPU frequency can be selected. If the CPU frequency is greater than or equal to the analog column clock, there is not a problem.

Table 79: Analog Synchronization Control Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/ Write	--	W	W	W	RW	RW	RW	RW
Bit Name	Reserved	SARCOUNT [2]	SARCOUNT [1]	SARCOUNT [0]	SAR-SIGN	SARCOL [1]	SARCOL [0]	SYN-CEN

Bit 7: **Reserved**

Bit [6:4]: **SARCOUNT [2:0]** Initial SAR count. Load this field with the number of bits to process. In a typical 6-bit SAR, the value would be 6

Bit 3: **SARSIGN** Adjust the SAR comparator based on the type of block addressed. In a DAC configuration with more than one PSoC block (more than 6-bits), this bit would be 0 when processing the most significant block and 1 when processing the least significant block. This is because the least significant block of a DAC is an inverting input to the most significant block

Bit [2:1]: **SARCOL [1:0]** Column select for SAR comparator input. The DAC portion of the SAR can reside in any of the appropriate positions in the analog PSOC block array. However, once the comparator block is positioned (and it is possible to have the DAC and comparator in the same block), this should be the column selected

Bit 0: **SYNCEN** Set to 1, will stall the CPU until the rising edge of PHI1, if a write to a register within an analog Switch Cap block takes place

Analog Synchronization Control Register (ASY_CR, Address = Bank 0, 65h)

10.12 Analog I/O

10.12.1 Analog Input Muxing

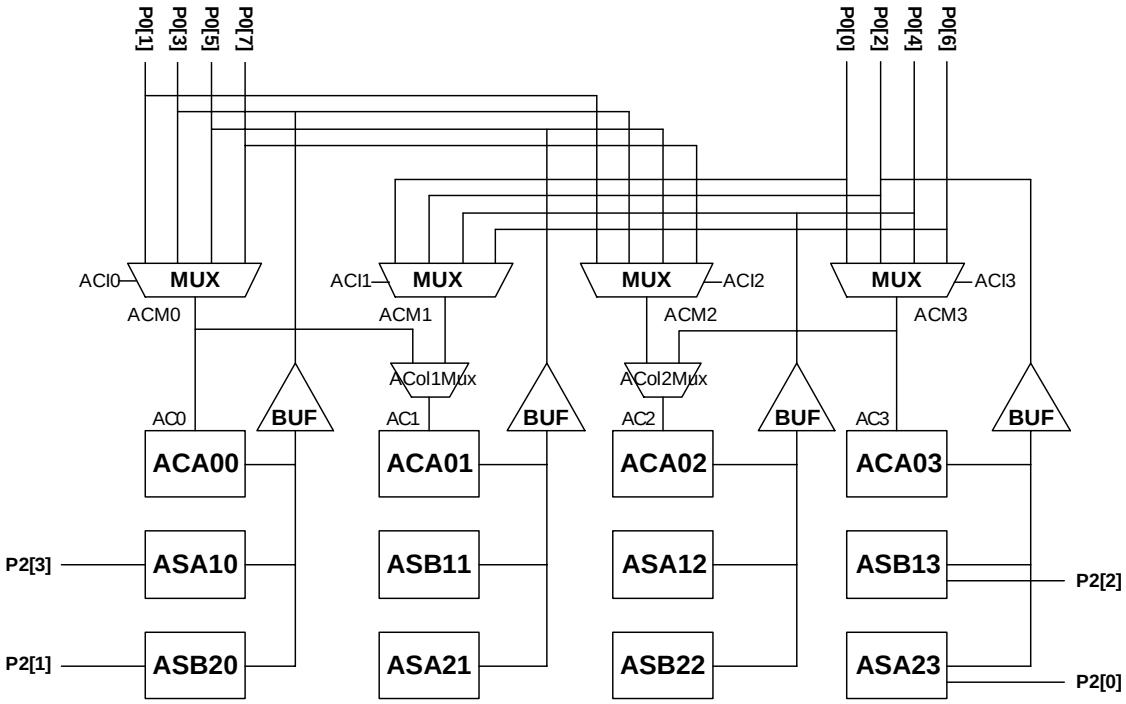

Figure 27: Analog Input Muxing

10.12.2 Analog Input Select Register

This register controls the analog muxes that feed signals in from port pins into each Analog Column. Each of the Analog Columns can have up to four port bits connected to its muxed input. Analog Columns 01 and 02 (ACI1 and ACI2) have additional muxes that allow selection between separate column multiplexers (see Analog Input Muxing diagram above). The AC1Mux and AC2Mux bit fields control the bits for those muxes and are located in the Analog Output Buffer Control Register (ABF_CR). There are four additional analog inputs that go directly into the Switch Capacitor PSoC blocks.

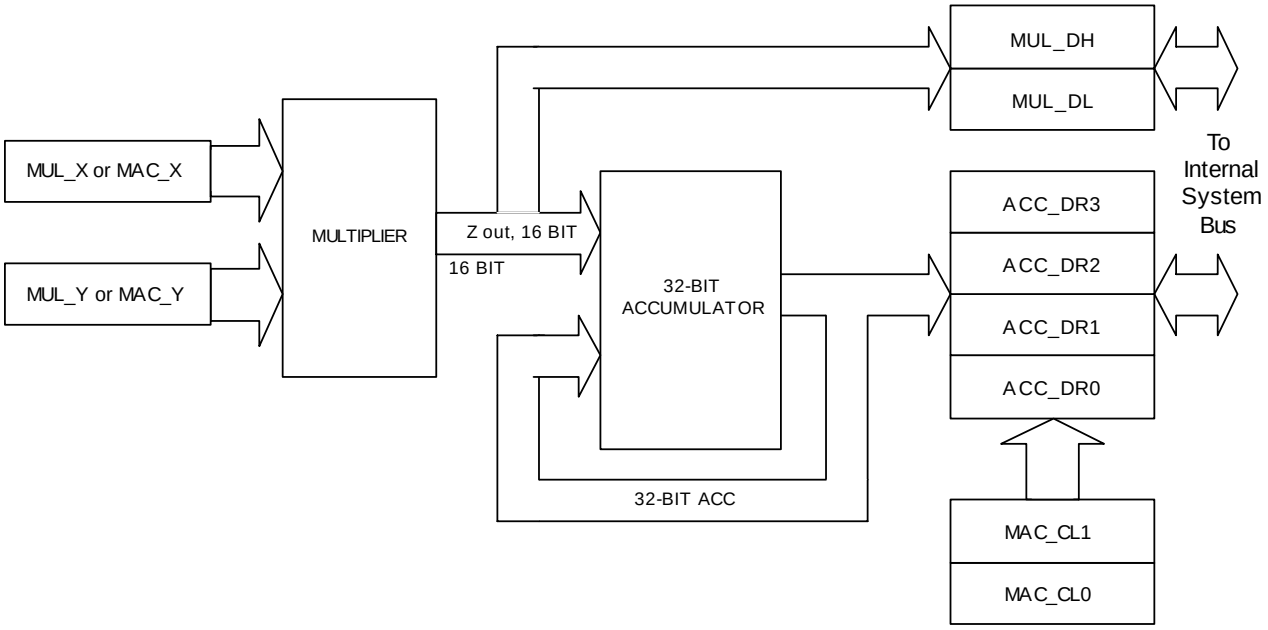

Figure 29: Multiply/Accumulate Block Diagram

Table 83: Multiply Input X Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Name	Data [7]	Data [6]	Data [5]	Data [4]	Data [3]	Data [2]	Data [1]	Data [0]
Bit [7:0]: **Data [7:0]** 8-bit data is the input value for X multiplier								

Multiply Input X Register (MUL_X, Address = Bank 0, E8h)

Table 84: Multiply Input Y Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Name	Data [7]	Data [6]	Data [5]	Data [4]	Data [3]	Data [2]	Data [1]	Data [0]
Bit [7:0]: **Data [7:0]** 8-bit data is the input value for Y multiplier								

Multiply Input Y Register (MUL_Y, Address = Bank 0, E9h)

Table 85: Multiply Result High Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R
Bit Name	Data [7]	Data [6]	Data [5]	Data [4]	Data [3]	Data [2]	Data [1]	Data [0]

Bit [7:0]: **Data [7:0]** 8-bit data value is the high order result of the multiply function

Multiply Result High Register (MUL_DH, Address = Bank 0, EAh)

Table 86: Multiply Result Low Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R
Bit Name	Data [7]	Data [6]	Data [5]	Data [4]	Data [3]	Data [2]	Data [1]	Data [0]

Bit [7:0]: **Data [7:0]** 8-bit data value is the low order result of the multiply function

Multiply Result Low Register (MUL_DL, Address = Bank 0, EBh)

Table 87: Accumulator Result 1 / Multiply/Accumulator Input X Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/Write	RW	RW	RW	RW	RW	RW	RW	RW
Bit Name	Data [7]	Data [6]	Data [5]	Data [4]	Data [3]	Data [2]	Data [1]	Data [0]

Bit [7:0]: **Data [7:0]**
8-bit data value when read is the next to lowest order result of the multiply/accumulate function
8-bit data value when written is the X multiplier input to the multiply/accumulate function

Accumulator Result 1 / Multiply/Accumulator Input X Register (ACC_DR1 / MAC_X, Address = Bank 0, ECh)

Table 88: Accumulator Result 0 / Multiply/Accumulator Input Y Register

Bit #	7	6	5	4	3	2	1	0
POR	0	0	0	0	0	0	0	0
Read/Write	RW	RW	RW	RW	RW	RW	RW	RW
Bit Name	Data [7]	Data [6]	Data [5]	Data [4]	Data [3]	Data [2]	Data [1]	Data [0]

Bit [7:0]: **Data [7:0]**
8-bit data value when read is the lowest order result of the multiply/accumulate function
8-bit data value when written is the Y multiplier input to the multiply/accumulate function

Accumulator Result 0 / Multiply/Accumulator Input Y Register (ACC_DR0 / MAC_Y, Address = Bank 0, EDh)

12.0 Development Tools

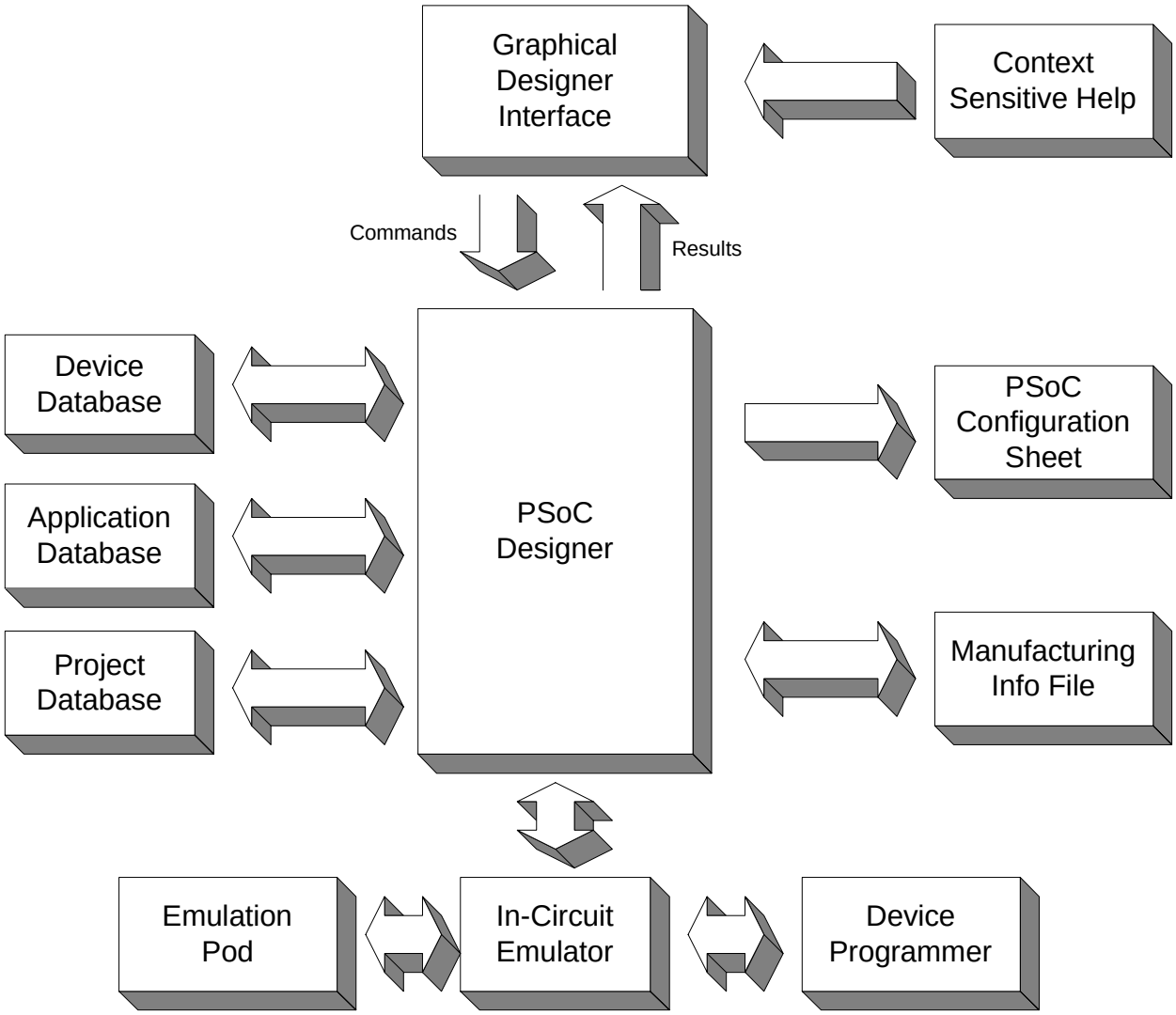

Figure 35: PSoC Designer Functional Flow

12.1 Overview

The Cypress MicroSystems PSoC Designer is a Microsoft® Windows-based, integrated development environment for the Programmable System-on-Chip (PSoC) devices. The PSoC Designer runs on Windows 98, Windows NT 4.0, Windows 2000, Windows Millennium (Me), or Windows XP.

PSoC Designer helps the customer to select an operating configuration for the microcontroller, write application code that uses the microcontroller, and debug the application. This system provides design database management by project, an integrated debugger with In-Circuit Emulator, in-system programming support, and the CYASM macro assembler for the CPUs.

PSoC Designer also supports a high-level C language compiler developed specifically for the devices in the family.

The following table lists guaranteed maximum and minimum specifications for the voltage and temperature ranges, 3.3V +/- 10% and -40°C <= T_A <= 85°C. Typical parameters apply to 5V at 25°C and are for design guidance only. For 5V operation, see Table 109 on page 132.

Table 110: 3.3V DC Analog Output Buffer Specifications

Symbol	3.3V DC Analog Output Buffer Specifications	Minimum	Typical	Maximum	Unit
	Input Offset Voltage (Absolute Value)	-	3	12	mV
	Average Input Offset Voltage Drift	-	+6	-	µV/°C
	Common-Mode Input Voltage Range	.5	-	V_{cc} - 1.0	V
	Output Resistance Bias = Low Bias = High	 - -	 1 1	 - -	 Ω Ω
	High Output Voltage Swing (Load = 32 ohms to V_{cc}/2) Bias = Low Bias = High	 .5 x V_{cc} + 1.3 .5 x V_{cc} + 1.3	 - -	 - -	 V V
	Low Output Voltage Swing (Load = 32 ohms to V_{cc}/2) Bias = Low Bias = High	 - -	 - -	 .5 x V_{cc} - 1.3 .5 x V_{cc} - 1.3	 V V
	Supply Current Including Bias Cell (No Load) Bias = Low Bias = High	 - -	 0.8 2.0	 2.0 4.3	 mA mA
	Supply Voltage Rejection Ratio	80	-	-	dB

Table 118: 3.3V AC Operational Amplifier Specifications

Symbol	3.3V AC Operational Amplifier Specifications	Minimum	Typical	Maximum	Unit
	Rising Settling Time to 0.1%				
	Bias = Low	-	-	3.0	µs
	Bias = Medium	-	-	1.6	µs
	Bias = High	-	-	1.5	µs
	Falling Settling Time to 0.1%				
	Bias = Low	-	-	2.6	µs
	Bias = Medium	-	-	1.7	µs
	Bias = High	-	-	1.6	µs
	Rising Slew Rate (20% to 80%)				
	Bias = Low	0.2	-	-	V/µs
	Bias = Medium	0.3	-	-	V/µs
	Bias = High	0.3	-	-	V/µs
	Falling Slew Rate (80% to 20%)				
	Bias = Low	0.3	-	-	V/µs
	Bias = Medium	0.3	-	-	V/µs
	Bias = High	0.3	-	-	V/µs
	Gain Bandwidth Product				
	Bias = Low	1.5	-	-	MHz
	Bias = Medium	4.4	-	-	MHz
	Bias = High	8.7	-	-	MHz

Table 120: 3.3V AC Analog Output Buffer Specifications

Symbol	3.3V AC Analog Output Buffer Specifications	Minimum	Typical	Maximum	Unit
	Rising Settling Time to 0.1%, 1V Step, 100pF Load				
	Bias = Low	-	-	3.2	µs
	Bias = High	-	-	3.2	µs
	Falling Settling Time to 0.1%, 1V Step, 100pF Load				
	Bias = Low	-	-	2.6	µs
	Bias = High	-	-	2.6	µs
	Rising Slew Rate (20% to 80%), 1V Step, 100pF Load				
	Bias = Low	.5	-	-	V/µs
	Bias = High	.5	-	-	V/µs
	Falling Slew Rate (80% to 20%), 1V Step, 100pF Load				
	Bias = Low	.5	-	-	V/µs
	Bias = High	.5	-	-	V/µs
	Small Signal Bandwidth, 20mV$_{pp}$, 3dB BW, 100pF Load				
	Bias = Low	1.3	-	-	MHz
	Bias = High	1.3	-	-	MHz
	Large Signal Bandwidth, 1V$_{pp}$, 3dB BW, 100pF Load				
	Bias = Low	360	-	-	kHz
	Bias = High	360	-	-	kHz

13.3.3 AC Programming Specifications

Table 121: AC Programming Specifications

Symbol	AC Programming Specifications	Minimum	Typical	Maximum	Unit
T_{rsclk}	Rise Time of SCLK	1	-	20	ns
T_{fsclk}	Fall Time of SCLK	1	-	20	ns
T_{ssclk}	Data Set up Time to Rising Edge of SCLK	25	-	-	ns
T_{hsclk}	Data Hold Time from Rising Edge of SCLK	25	-	-	ns
F_{sclk}	Frequency of SCLK	2	-	20	MHz
T_{eraseb}	Flash Erase Time (Block)	-	10	-	ms
T_{erasef}	Flash Erase Time (Full)	-	40	-	ms
T_{write}	Flash Block Write Time	2	10	20	ms

14.0 Packaging Information

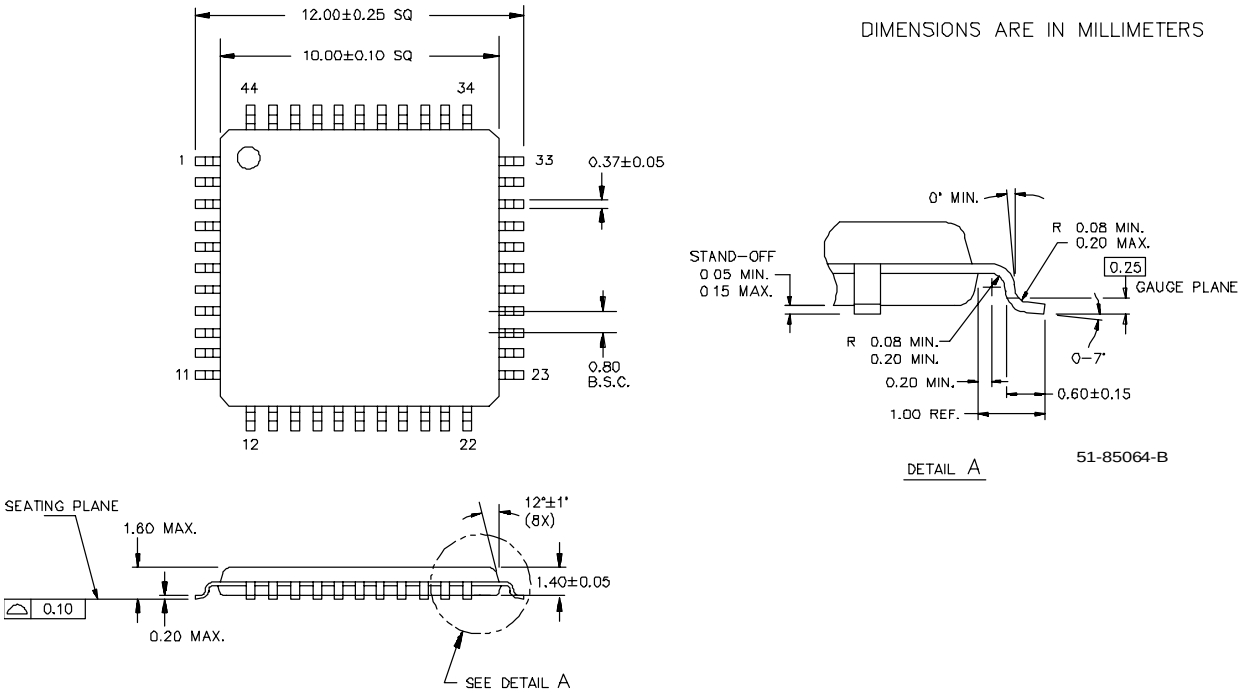

Figure 37: 44-Lead Thin Plastic Quad Flat Pack A44