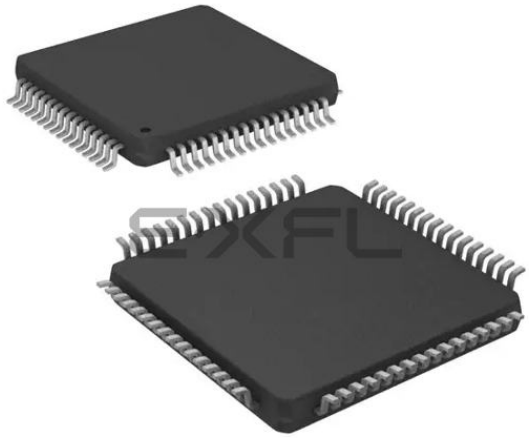


Welcome to E-XFL.COM

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	8051
Core Size	8-Bit
Speed	25MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	24
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4.25K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 2x16b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f067r

1.6. Controller Area Network

The C8051F060/1/2/3 devices feature a Controller Area Network (CAN) controller that implements serial communication using the CAN protocol. The CAN controller facilitates communication on a CAN network in accordance with the Bosch specification 2.0A (basic CAN) and 2.0B (full CAN). The CAN controller consists of a CAN Core, Message RAM (separate from the C8051 RAM), a message handler state machine, and control registers.

The CAN controller can operate at bit rates up to 1 Mbit/second. Silicon Labs CAN has 32 message objects each having its own identifier mask used for acceptance filtering of received messages. Incoming data, message objects and identifier masks are stored in the CAN message RAM. All protocol functions for transmission of data and acceptance filtering is performed by the CAN controller and not by the C8051 MCU. In this way, minimal CPU bandwidth is used for CAN communication. The C8051 configures the CAN controller, accesses received data, and passes data for transmission via Special Function Registers (SFR) in the C8051.

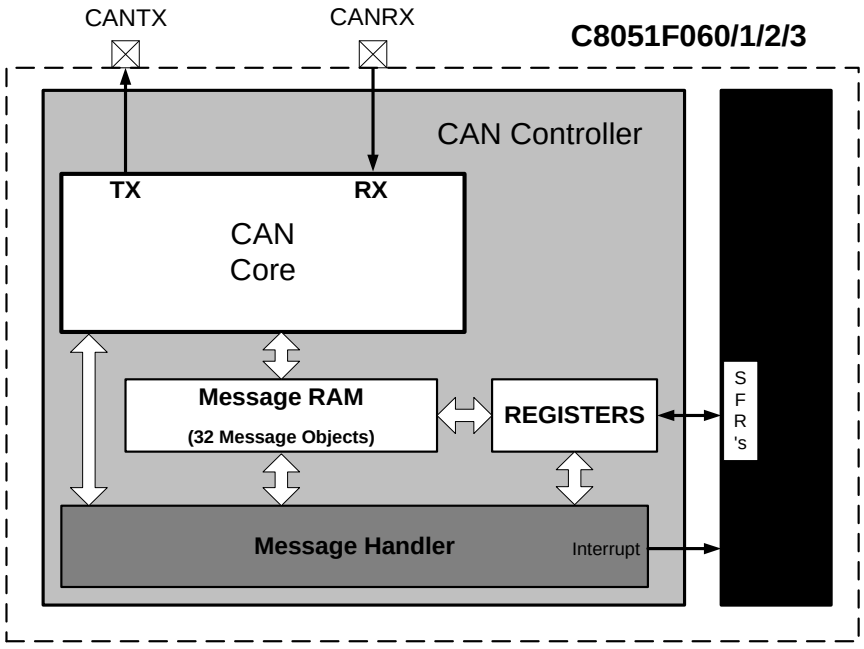


Figure 1.11. CAN Controller Overview

5. 16-Bit ADCs (ADC0 and ADC1)

The ADC subsystem for the C8051F060/1/2/3/4/5/6/7 consists of two 1 Msps, 16-bit successive-approximation-register ADCs with integrated track-and-hold, a Programmable Window Detector, and a DMA interface (see block diagrams in Figure 5.1 and Figure 5.2). The ADCs can be configured as two separate, single-ended ADCs, or as a differential pair. The Data Conversion Modes, Window Detector, and DMA interface are all configurable under software control via the Special Function Registers shown in Figure 5.1 and Figure 5.2. The voltage references used by ADC0 and ADC1 are selected as described in Section 5.2. The ADCs and their respective track-and-hold circuitry can be independently enabled or disabled with the Special Function Registers. Either ADC can be enabled by setting the ADnEN bit in the ADC's Control register (ADCnCN) to logic 1. The ADCs are in low power shutdown when these bits are logic 0.

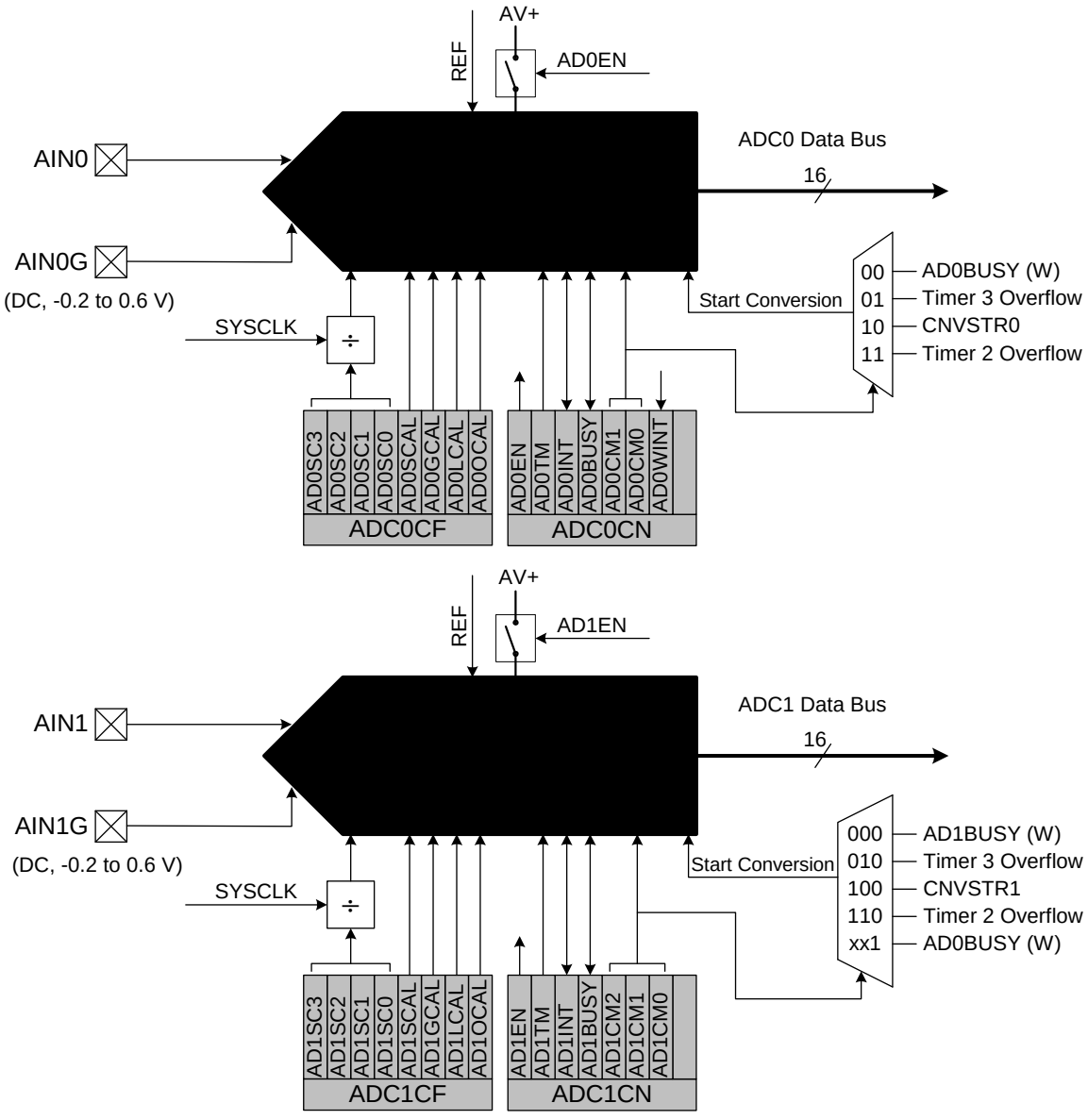


Figure 5.1. 16-Bit ADC0 and ADC1 Control Path Diagram

Figure 5.7. ADC0CF: ADC0 Configuration Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
AD0SC3	AD0SC2	AD0SC1	AD0SC0	AD0SCAL	AD0GCAL	AD0LCAL	AD0OCAL	11110000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xBC
SFR Page: 0

Bits 7-4: AD0SC3-0: ADC0 SAR Conversion Clock Period Bits.
SAR Conversion clock is divided down from the system clock according to the AD0SC bits (AD0SC3-0). The number of system clocks used for each SAR conversion clock is equal to AD0SC + 1. (Note: the ADC0 SAR Conversion Clock should be less than or equal to 25 MHz). See Table 5.1 for conversion timing details.

Bit 3: AD0SCAL: System Calibration Enable.
0: Internal ground and reference voltage are used during offset and gain calibration.
1: External voltages can be used during offset and gain calibration.

Bit 2: AD0GCAL: Gain Calibration.
Read:
0: Gain Calibration is completed or not yet started.
1: Gain Calibration is in progress.
Write:
0: No Effect.
1: Initiates a gain calibration if ADC0 is idle.

Bit 1: AD0LCAL: Linearity Calibration
Read
0: Linearity Calibration is completed or not yet started
1: Linearity Calibration is in progress
Write
0: No Effect
1: Initiates a linearity calibration if ADC0 is idle

Bit 0: AD0OCAL: Offset Calibration.
Read:
0: Offset Calibration is completed or not yet started.
1: Offset Calibration is in progress.
Write:
0: No Effect.
1: Initiates an offset calibration if ADC0 is idle.

Figure 5.8. ADC1CF: ADC1 Configuration Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
AD1SC3	AD1SC2	AD1SC1	AD1SC0	AD1SCAL	AD1GCAL	AD1LCAL	AD1OCAL	11110000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xBC
SFR Page: 1

Bits 7-4: AD1SC3-0: ADC1 SAR Conversion Clock Period Bits.
SAR Conversion clock is divided down from the system clock according to the AD1SC bits (AD1SC3-0). The number of system clocks used for each SAR conversion clock is equal to AD1SC + 1. (Note: the ADC1 SAR Conversion Clock should be less than or equal to 25 MHz). See Table 5.1 for conversion timing details.

Bit 3: AD1SCAL: System Calibration Enable.
0: Internal ground and reference voltage are used for offset and gain calibration.
1: External voltages can be used for offset and gain calibration.

Bit 2: AD1GCAL: Gain Calibration.
Read:
0: Gain Calibration is completed or not yet started.
1: Gain Calibration is in progress.
Write:
0: No Effect.
1: Initiates a gain calibration if ADC1 is idle.

Bit 1: AD1LCAL: Linearity Calibration
Read
0: Linearity Calibration is completed or not yet started
1: Linearity Calibration is in progress
Write
0: No Effect
1: Initiates a linearity calibration if ADC1 is idle

Bit 0: AD1OCAL: Offset Calibration.
Read:
0: Offset Calibration is completed or not yet started.
1: Offset Calibration is in progress.
Write:
0: No Effect.
1: Initiates an offset calibration if ADC1 is idle.

Figure 5.9. ADC0CN: ADC0 Control Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
AD0EN	AD0TM	AD0INT	AD0BUSY	AD0CM1	AD0CM0	AD0WINT	-	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Bit Addressable

SFR Address: 0xE8
SFR Page: 0

Bit 7: AD0EN: ADC0 Enable Bit.
0: ADC0 Disabled. ADC0 is in low-power shutdown.
1: ADC0 Enabled. ADC0 is active and ready for data conversions or calibrations.

Bit 6: AD0TM: ADC Track Mode Bit.
0: When the ADC is enabled, tracking is continuous unless a conversion is in process.
1: Tracking Defined by AD0CM1-0 bits.

Bit 5: AD0INT: ADC0 Conversion Complete Interrupt Flag.
This flag must be cleared by software.
0: ADC0 has not completed a data conversion since the last time this flag was cleared.
1: ADC0 has completed a data conversion.

Bit 4: AD0BUSY: ADC0 Busy Bit.
Read:
0: ADC0 Conversion is complete or a conversion is not currently in progress. AD0INT is set to logic 1 on the falling edge of AD0BUSY.
1: ADC0 Conversion is in progress.
Write:
0: No Effect.
1: Initiates ADC0 Conversion if AD0CM1-0 = 00b.

Bits 3-2: AD0CM1-0: ADC0 Start of Conversion Mode Select.
If AD0TM = 0:
00: ADC0 conversion initiated on every write of ‘1’ to AD0BUSY.
01: ADC0 conversion initiated on overflow of Timer 3.
10: ADC0 conversion initiated on rising edge of external CNVSTR0.
11: ADC0 conversion initiated on overflow of Timer 2.
If AD0TM = 1:
00: Tracking starts with the write of ‘1’ to AD0BUSY and is followed by the conversion.
01: Tracking started by the overflow of Timer 3 and is followed by the conversion.
10: ADC0 conversion starts on rising CNVSTR0 edge.
11: Tracking started by the overflow of Timer 2 and is followed by the conversion.
See Figure 5.4 and Table 5.1 for conversion timing parameters.

Bit 1: AD0WINT: ADC0 Window Compare Interrupt Flag.
This bit must be cleared by software.
0: ADC0 Window Comparison Data match has not occurred since this flag was last cleared.
1: ADC0 Window Comparison Data match has occurred.

Bit 0: RESERVED: Write to 0b.

Figure 5.26. ADC0LTH: ADC0 Less-Than Data High Byte Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xC7
SFR Page: 0

Bits 7-0: High byte of ADC0 Less-Than Data Word.

Figure 5.27. ADC0LTL: ADC0 Less-Than Data Low Byte Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xC6
SFR Page: 0

Bits 7-0: Low byte of ADC0 Less-Than Data Word.

.

Table 8.1. DAC Electrical Characteristics

VDD = 3.0 V, AV+ = 3.0 V, VREF = 2.40 V (REFBE = 0), No Output Load unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Static Performance					
Resolution			12		bits
Integral Nonlinearity			±1.5		LSB
Differential Nonlinearity				±1	LSB
Output Noise	No Output Filter 100 kHz Output Filter 10 kHz Output Filter		250 128 41		µVrms
Offset Error	Data Word = 0x014		±3	±30	mV
Offset Tempco			6		ppm/°C
Full-Scale Error			±20	±60	mV
Full-Scale Error Tempco			10		ppm/°C
VDD Power Supply Rejection Ratio			-60		dB
Output Impedance in Shutdown Mode	DACnEN = 0		100		kΩ
Output Sink Current			300		µA
Output Short-Circuit Current	Data Word = 0xFFF		15		mA
Dynamic Performance					
Voltage Output Slew Rate	Load = 40pF		0.44		V/µs
Output Settling Time to 1/2 LSB	Load = 40pF, Output swing from code 0xFFF to 0x014		10		µs
Output Voltage Swing		0		VREF-1LSB	V
Startup Time			10		µs
Analog Outputs					
Load Regulation	I_L = 0.01mA to 0.3mA at code 0xFFF		60		ppm
Power Consumption (each DAC)					
Power Supply Current (AV+ supplied to DAC)	Data Word = 0x7FF		300	500	µA

Table 13.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
XRL direct, #data	Exclusive-OR immediate to direct byte	3	3
CLR A	Clear A	1	1
CPL A	Complement A	1	1
RL A	Rotate A left	1	1
RLC A	Rotate A left through Carry	1	1
RR A	Rotate A right	1	1
RRC A	Rotate A right through Carry	1	1
SWAP A	Swap nibbles of A	1	1
Data Transfer			
MOV A, Rn	Move Register to A	1	1
MOV A, direct	Move direct byte to A	2	2
MOV A, @Ri	Move indirect RAM to A	1	2
MOV A, #data	Move immediate to A	2	2
MOV Rn, A	Move A to Register	1	1
MOV Rn, direct	Move direct byte to Register	2	2
MOV Rn, #data	Move immediate to Register	2	2
MOV direct, A	Move A to direct byte	2	2
MOV direct, Rn	Move Register to direct byte	2	2
MOV direct, direct	Move direct byte to direct byte	3	3
MOV direct, @Ri	Move indirect RAM to direct byte	2	2
MOV direct, #data	Move immediate to direct byte	3	3
MOV @Ri, A	Move A to indirect RAM	1	2
MOV @Ri, direct	Move direct byte to indirect RAM	2	2
MOV @Ri, #data	Move immediate to indirect RAM	2	2
MOV DPTR, #data16	Load DPTR with 16-bit constant	3	3
MOVC A, @A+DPTR	Move code byte relative DPTR to A	1	3
MOVC A, @A+PC	Move code byte relative PC to A	1	3
MOVX A, @Ri	Move external data (8-bit address) to A	1	3
MOVX @Ri, A	Move A to external data (8-bit address)	1	3
MOVX A, @DPTR	Move external data (16-bit address) to A	1	3
MOVX @DPTR, A	Move A to external data (16-bit address)	1	3
PUSH direct	Push direct byte onto stack	2	2
POP direct	Pop direct byte from stack	2	2
XCH A, Rn	Exchange Register with A	1	1
XCH A, direct	Exchange direct byte with A	2	2
XCH A, @Ri	Exchange indirect RAM with A	1	2
XCHD A, @Ri	Exchange low nibble of indirect RAM with A	1	2
Boolean Manipulation			
CLR C	Clear Carry	1	1
CLR bit	Clear direct bit	2	2
SETB C	Set Carry	1	1
SETB bit	Set direct bit	2	2
CPL C	Complement Carry	1	1
CPL bit	Complement direct bit	2	2
ANL C, bit	AND direct bit to Carry	2	2

17.4. Multiplexed and Non-multiplexed Selection

The External Memory Interface is capable of acting in a Multiplexed mode or a Non-multiplexed mode, depending on the state of the EMD2 (EMI0CF.4) bit.

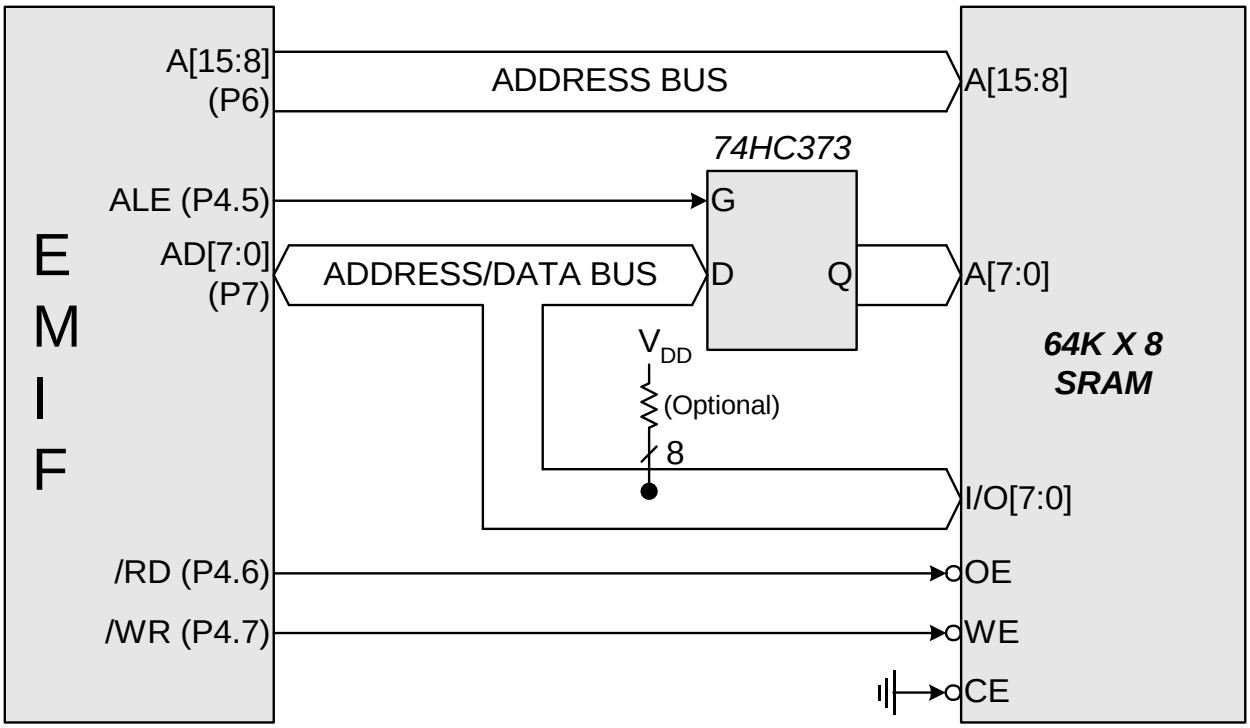
17.4.1. Multiplexed Configuration

In Multiplexed mode, the Data Bus and the lower 8-bits of the Address Bus share the same Port pins: AD[7:0]. In this mode, an external latch (74HC373 or equivalent logic gate) is used to hold the lower 8-bits of the RAM address. The external latch is controlled by the ALE (Address Latch Enable) signal, which is driven by the External Memory Interface logic. An example of a Multiplexed Configuration is shown in Figure 17.3.

In Multiplexed mode, the external MOVX operation can be broken into two phases delineated by the state of the ALE signal. During the first phase, ALE is high and the lower 8-bits of the Address Bus are presented to AD[7:0]. During this phase, the address latch is configured such that the 'Q' outputs reflect the states of the 'D' inputs. When ALE falls, signaling the beginning of the second phase, the address latch outputs remain fixed and are no longer dependent on the latch inputs. Later in the second phase, the Data Bus controls the state of the AD[7:0] port at the time /RD or /WR is asserted.

See Section "17.6.2. Multiplexed Mode" on page 199 for more information.

Figure 17.3. Multiplexed Configuration Example



18. Port Input/Output

The C8051F06x family of devices are fully integrated mixed-signal System on a Chip MCUs with 59 digital I/O pins (C8051F060/2/4/6) or 24 digital I/O pins (C8051F061/3/5/7), organized as 8-bit Ports. All ports are both bit- and byte-addressable through their corresponding Port Data registers. All Port pins support configurable Open-Drain or Push-Pull output modes and weak pull-ups. Additionally, Port 0 pins are 5 V-tolerant. A block diagram of the Port I/O cell is shown in Figure 18.1. Complete Electrical Specifications for the Port I/O pins are given in Table 18.1.

Figure 18.1. Port I/O Cell Block Diagram

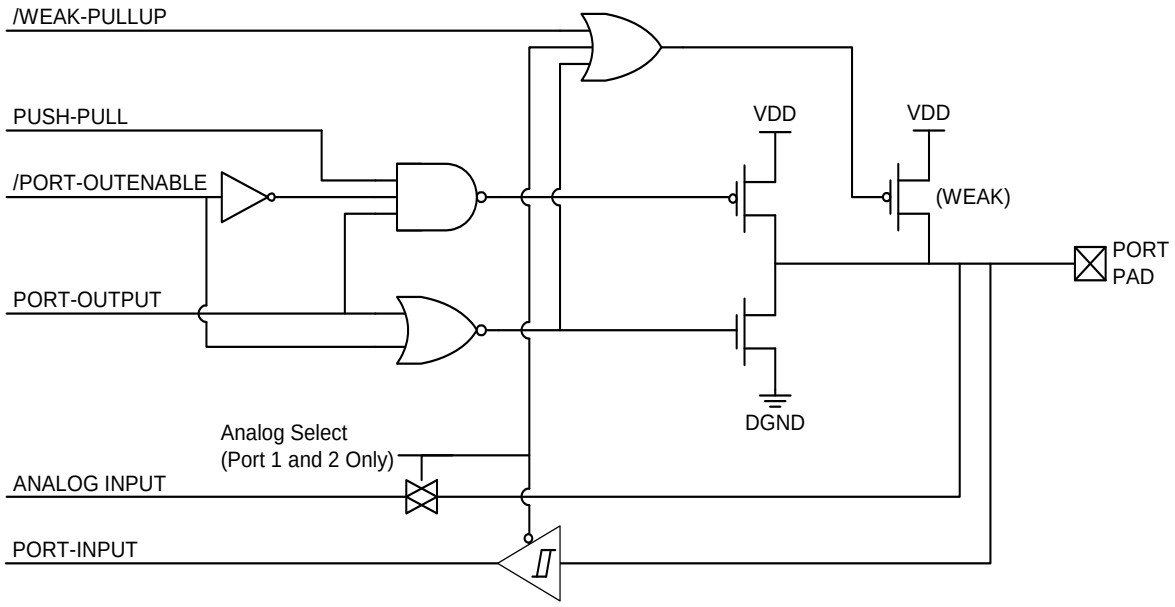


Table 18.1. Port I/O DC Electrical Characteristics

VDD = 2.7 to 3.6 V, -40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Output High Voltage (V_{OH})	I_{OH} = -3 mA, Port I/O Push-Pull I_{OH} = -10 µA, Port I/O Push-Pull	VDD - 0.7 VDD - 0.1			V
Output Low Voltage (V_{OL})	I_{OL} = 8.5 mA I_{OL} = 10 µA			0.6 0.1	V
Input High Voltage (VIH)		0.7 x VDD			
Input Low Voltage (VIL)				0.3 x VDD	
Input Leakage Current	DGND < Port Pin < VDD, Pin Tri-state Weak Pull-up Off Weak Pull-up On		 10	± 1	 µA µA
Input Capacitance			5		pF

19. Controller Area Network (CAN0, C8051F060/1/2/3)

IMPORTANT DOCUMENTATION NOTE: *The Bosch CAN Controller is integrated in the C8051F060/1/2/3 devices. This section of the data sheet gives a description of the CAN controller as an overview and offers a description of how the Silicon Labs CIP-51 MCU interfaces with the on-chip Bosch CAN controller. In order to use the CAN controller, please refer to Bosch's C_CAN User's Manual (revision 1.2) as an accompanying manual to Silicon Labs' C8051F060/1/2/3/4/5/6/7 Data sheet.*

The C8051F060/1/2/3 family of devices feature a Control Area Network (CAN) controller that enables serial communication using the CAN protocol. Silicon Labs CAN controller facilitates communication on a CAN network in accordance with the Bosch specification 2.0A (basic CAN) and 2.0B (full CAN). The CAN controller consists of a CAN Core, Message RAM (separate from the CIP-51 RAM), a message handler state machine, and control registers. Silicon Labs CAN is a protocol controller and does not provide physical layer drivers (i.e., transceivers). Figure 19.2 shows an example typical configuration on a CAN bus.

Silicon Labs CAN operates at bit rates of up to 1 Mbit/second, though this can be limited by the physical layer chosen to transmit data on the CAN bus. The CAN processor has 32 Message Objects that can be configured to transmit or receive data. Incoming data, message objects and their identifier masks are stored in the CAN message RAM. All protocol functions for transmission of data and acceptance filtering is performed by the CAN controller and not by the CIP-51 MCU. In this way, minimal CPU bandwidth is needed to use CAN communication. The CIP-51 configures the CAN controller, accesses received data, and passes data for transmission via Special Function Registers (SFR) in the CIP-51. The CAN controller's clock (f_{sys}, or CAN_CLK in the C_CAN User's Guide) is equal to the CIP-51 MCU's clock (SYSCLK).

Please refer to the Bosch CAN User's Guide for information on the function and use of the Message Handler Registers.

19.2.4. CIP-51 MCU Special Function Registers

C8051F060/1/2/3 peripherals are modified, monitored, and controlled using Special Function Registers (SFRs). Most of the CAN Controller registers cannot be accessed *directly* using the SFRs. Three of the CAN Controller's registers may be accessed directly with SFRs. All other CAN Controller registers are accessed indirectly using three CIP-51 MCU SFRs: the CAN Data Registers (CAN0DATH and CAN0DATL) and CAN Address Register (CAN0ADR). In this way, there are a total of five CAN registers used to configure and run the CAN Controller.

19.2.5. Using CAN0ADR, CAN0DATH, and CANDATL To Access CAN Registers

Each CAN Controller Register has an index number (see Table below). The CAN register address space is 128 words (256 bytes). A CAN register is accessed via the CAN Data Registers (CAN0DATH and CAN0DATL) when a CAN register's index number is placed into the CAN Address Register (CAN0ADR). For example, if the Bit Timing Register is to be configured with a new value, CAN0ADR is loaded with 0x03. The low byte of the desired value is accessed using CAN0DATL and the high byte of the bit timing register is accessed using CAN0DATH. CAN0DATL is bit addressable for convenience. To load the value 0x2304 into the Bit Timing Register:

```
CAN0ADR = 0x03;     // Load Bit Timing Register's index (Table 18.1)
CAN0DATH = 0x23;    // Move the upper byte into data reg high byte
CAN0DATL = 0x04;    // Move the lower byte into data reg low byte
```

Note: CAN0CN, CAN0STA, and CAN0TST may be accessed either by using the index method, or by direct access with the CIP-51 MCU SFRs. CAN0CN is located at SFR location 0xF8/SFR page 1 (Figure 19.6), CAN0TST at 0xDB/SFR page 1 (Figure 19.7), and CAN0STA at 0xC0/SFR page 1 (Figure 19.8).

19.2.6. CAN0ADR Autoincrement Feature

For ease of programming message objects, CAN0ADR features autoincrementing for the index ranges 0x08 to 0x12 (Interface Registers 1) and 0x20 to 0x2A (Interface Registers 2). When the CAN0ADR register has an index in these ranges, **the CAN0ADR will autoincrement by 1 to point to the next CAN register 16-bit word upon a read/write of CAN0DATL**. This speeds programming of the frequently programmed interface registers when configuring message objects.

NOTE: Table below supersedes Figure 5 in section 3, "Programmer's Model" of the Bosch CAN User's Guide.

Table 19.1. CAN Register Index and Reset Values

CAN Register Index	Register name	Reset Value	Notes
0x00	CAN Control Register	0x0001	Accessible in CIP-51 SFR Map
0x01	Status Register	0x0000	Accessible in CIP-51 SFR Map
0x02	Error Register	0x0000	Read Only
0x03	Bit Timing Register	0x2301	Write Enabled by CCE Bit in CAN0CN

Table 19.1. CAN Register Index and Reset Values (Continued)

CAN Register Index	Register name	Reset Value	Notes
0x59	Message Valid 2	0x0000	Message valid flags for message objects (read only)

Figure 19.3. CAN0DATH: CAN Data Access Register High Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xD9
SFR Page: 1

Bit7-0: CAN0DATH: CAN Data Access Register High Byte.
The CAN0DAT Registers are used to read/write register values and data to and from the CAN Registers pointed to with the index number in the CAN0ADR Register.
The CAN0ADR Register is used to point the [CAN0DATH:CAN0DATL] to a desired CAN Register. The desired CAN Register's index number is moved into CAN0ADR. The CAN0DAT Register can then read/write to and from the CAN Register.

Figure 19.4. CAN0DATL: CAN Data Access Register Low Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000001
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xD8
SFR Page: 1

Bit7-0: CAN0DATL: CAN Data Access Register Low Byte.
The CAN0DAT Registers are used to read/write register values and data to and from the CAN Registers pointed to with the index number in the CAN0ADR Register.
The CAN0ADR Register is used to point the [CAN0DATH:CAN0DATL] to a desired CAN Register. The desired CAN Register's index number is moved into CAN0ADR. The CAN0DAT Register can then read/write to and from the CAN Register.

Figure 21.9. SPI0CN: SPI0 Control Register

R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	Reset Value
SPIF	WCOL	MODF	RXOVRN	NSSMD1	NSSMD0	TXBMT	SPIEN	00000110
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Bit Addressable

SFR Address: 0xF8
SFR Page: 0

Bit 7: SPIF: SPI0 Interrupt Flag.
This bit is set to logic 1 by hardware at the end of a data transfer. If interrupts are enabled, setting this bit causes the CPU to vector to the SPI0 interrupt service routine. This bit is not automatically cleared by hardware. It must be cleared by software.

Bit 6: WCOL: Write Collision Flag.
This bit is set to logic 1 by hardware (and generates a SPI0 interrupt) to indicate a write to the SPI0 data register was attempted while a data transfer was in progress. It must be cleared by software.

Bit 5: MODF: Mode Fault Flag.
This bit is set to logic 1 by hardware (and generates a SPI0 interrupt) when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01). This bit is not automatically cleared by hardware. It must be cleared by software.

Bit 4: RXOVRN: Receive Overrun Flag (Slave Mode only).
This bit is set to logic 1 by hardware (and generates a SPI0 interrupt) when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI0 shift register. This bit is not automatically cleared by hardware. It must be cleared by software.

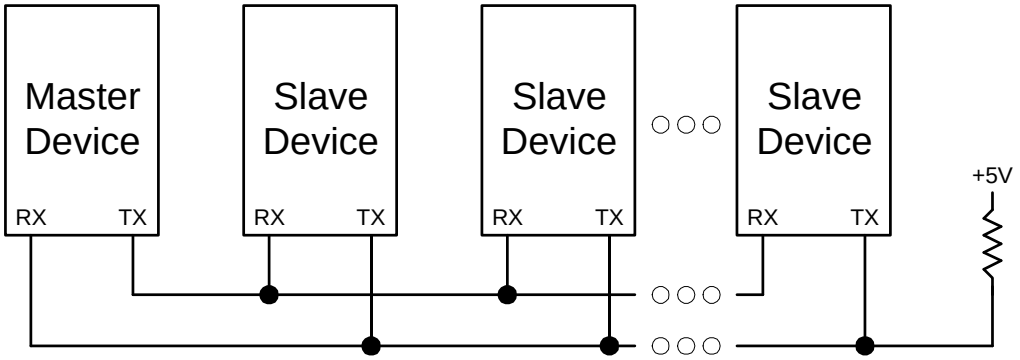
Bits 3-2: NSSMD1-NSSMD0: Slave Select Mode.
Selects between the following NSS operation modes:
(See Section “21.2. SPI0 Master Mode Operation” on page 253 and Section “21.3. SPI0 Slave Mode Operation” on page 255).
00: 3-Wire Slave or 3-wire Master Mode. NSS signal is not routed to a port pin.
01: 4-Wire Slave or Multi-Master Mode (Default). NSS is always an input to the device.
1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0.

Bit 1: TXBMT: Transmit Buffer Empty.
This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer.

Bit 0: SPIEN: SPI0 Enable.
This bit enables/disables the SPI.
0: SPI disabled.
1: SPI enabled.

address as valid. If a master were to then send an address of “11111111”, all three slave devices would recognize the address as a valid broadcast address.

Figure 22.7. UART Multi-Processor Mode Interconnect Diagram



22.3. Frame and Transmission Error Detection

All Modes:
The Transmit Collision bit (TXCOL0 bit in register SCON0) reads ‘1’ if user software writes data to the SBUF0 register while a transmit is in progress. Note that the TXCOL0 bit is also used as the SM20 bit when written by user software. This bit does not generate an interrupt.

Modes 1, 2, and 3:

The Receive Overrun bit (RXOV0 in register SCON0) reads ‘1’ if a new data byte is latched into the receive buffer before software has read the previous byte. Note that the RXOV0 bit is also used as the SM10 bit when written by user software. The Frame Error bit (FE0 in register SSTA0) reads ‘1’ if an invalid (low) STOP bit is detected. Note that the FE0 bit is also used as the SM00 bit when written by user software. The RXOV0 and FE0 bits do not generate interrupts.

23. UART1

UART1 is an asynchronous, full duplex serial port offering modes 1 and 3 of the standard 8051 UART. Enhanced baud rate support allows a wide range of clock sources to generate standard baud rates (details in Section “23.1. Enhanced Baud Rate Generation” on page 278). Received data buffering allows UART1 to start reception of a second incoming data byte before software has finished reading the previous data byte.

UART1 has two associated SFRs: Serial Control Register 1 (SCON1) and Serial Data Buffer 1 (SBUF1). The single SBUF1 location provides access to both transmit and receive registers. Reading SBUF1 accesses the buffered Receive register; writing SBUF1 accesses the Transmit register.

With UART1 interrupts enabled, an interrupt is generated each time a transmit is completed (TI1 is set in SCON1), or a data byte has been received (RI1 is set in SCON1). The UART1 interrupt flags are not cleared by hardware when the CPU vectors to the interrupt service routine. They must be cleared manually by software, allowing software to determine the cause of the UART1 interrupt (transmit complete or receive complete).

Figure 23.1. UART1 Block Diagram

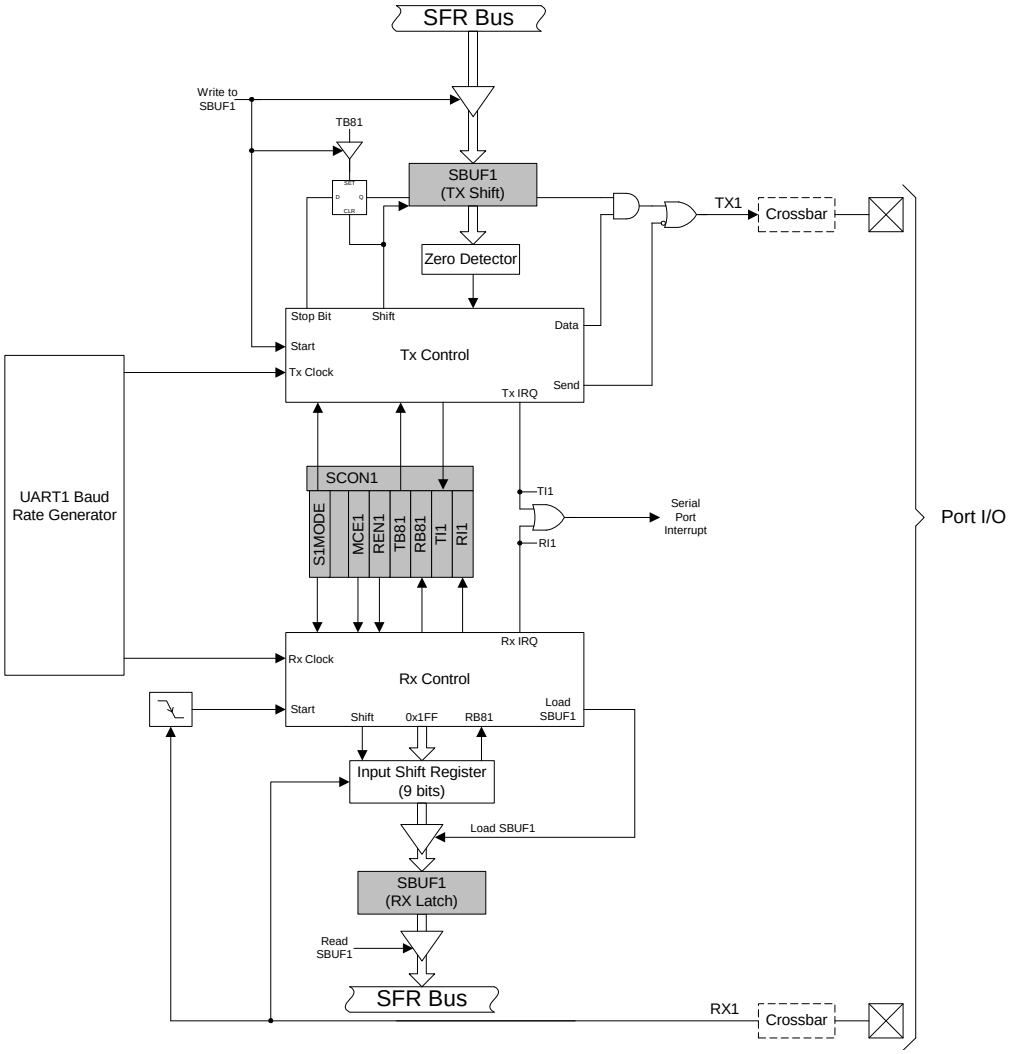


Table 23.3. Timer Settings for Standard Baud Rates Using an External Oscillator

	Frequency: 22.1184 MHz						
	Target Baud Rate (bps)	Baud Rate % Error	Oscilla-tor Divide Factor	Timer Clock Source	SCA1-SCA0 (pre-scale select)†	T1M†	Timer 1 Reload Value (hex)
SYSCLK from External Osc.	230400	0.00%	96	SYSCLK	XX	1	0xD0
	115200	0.00%	192	SYSCLK	XX	1	0xA0
	57600	0.00%	384	SYSCLK	XX	1	0x40
	28800	0.00%	768	SYSCLK / 12	00	0	0xE0
	14400	0.00%	1536	SYSCLK / 12	00	0	0xC0
	9600	0.00%	2304	SYSCLK / 12	00	0	0xA0
	2400	0.00%	9216	SYSCLK / 48	10	0	0xA0
	1200	0.00%	18432	SYSCLK / 48	10	0	0x40
SYSCLK from Internal Osc.	230400	0.00%	96	EXTCLK / 8	11	0	0xFA
	115200	0.00%	192	EXTCLK / 8	11	0	0xF4
	57600	0.00%	384	EXTCLK / 8	11	0	0xE8
	28800	0.00%	768	EXTCLK / 8	11	0	0xD0
	14400	0.00%	1536	EXTCLK / 8	11	0	0xA0
	9600	0.00%	2304	EXTCLK / 8	11	0	0x70

X = Don't care

†SCA1-SCA0 and T1M bit definitions can be found in Section 24.1.

Table 23.4. Timer Settings for Standard Baud Rates Using an External Oscillator

	Frequency: 18.432 MHz						
	Target Baud Rate (bps)	Baud Rate % Error	Oscilla-tor Divide Factor	Timer Clock Source	SCA1-SCA0 (pre-scale select)†	T1M†	Timer 1 Reload Value (hex)
SYSCLK from External Osc.	230400	0.00%	80	SYSCLK	XX	1	0xD8
	115200	0.00%	160	SYSCLK	XX	1	0xB0
	57600	0.00%	320	SYSCLK	XX	1	0x60
	28800	0.00%	640	SYSCLK / 4	01	0	0xB0
	14400	0.00%	1280	SYSCLK / 4	01	0	0x60
	9600	0.00%	1920	SYSCLK / 12	00	0	0xB0
	2400	0.00%	7680	SYSCLK / 48	10	0	0xB0
	1200	0.00%	15360	SYSCLK / 48	10	0	0x60
SYSCLK from Internal Osc.	230400	0.00%	80	EXTCLK / 8	11	0	0xFB
	115200	0.00%	160	EXTCLK / 8	11	0	0xF6
	57600	0.00%	320	EXTCLK / 8	11	0	0xEC
	28800	0.00%	640	EXTCLK / 8	11	0	0xD8
	14400	0.00%	1280	EXTCLK / 8	11	0	0xB0
	9600	0.00%	1920	EXTCLK / 8	11	0	0x88

X = Don't care

†SCA1-SCA0 and T1M bit definitions can be found in Section 24.1.

25.2.6. 16-Bit Pulse Width Modulator Mode

Each PCA0 module may also be operated in 16-Bit PWM mode. In this mode, the 16-bit capture/compare module defines the number of PCA0 clocks for the low time of the PWM signal. When the PCA0 counter matches the module contents, the output on CEXn is asserted high; when the counter overflows, CEXn is asserted low. To output a varying duty cycle, new value writes should be synchronized with PCA0 CCFn match interrupts. 16-Bit PWM Mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, CCFn should also be set to logic 1 to enable match interrupts. The duty cycle for 16-Bit PWM Mode is given by Equation 25.3.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to ‘0’; writing to PCA0CPHn sets ECOMn to ‘1’.

Equation 25.3. 16-Bit PWM Duty Cycle

$$DutyCycle = \frac{(65536 - PCA0CPn)}{65536}$$

Figure 25.9. PCA 16-Bit PWM Mode

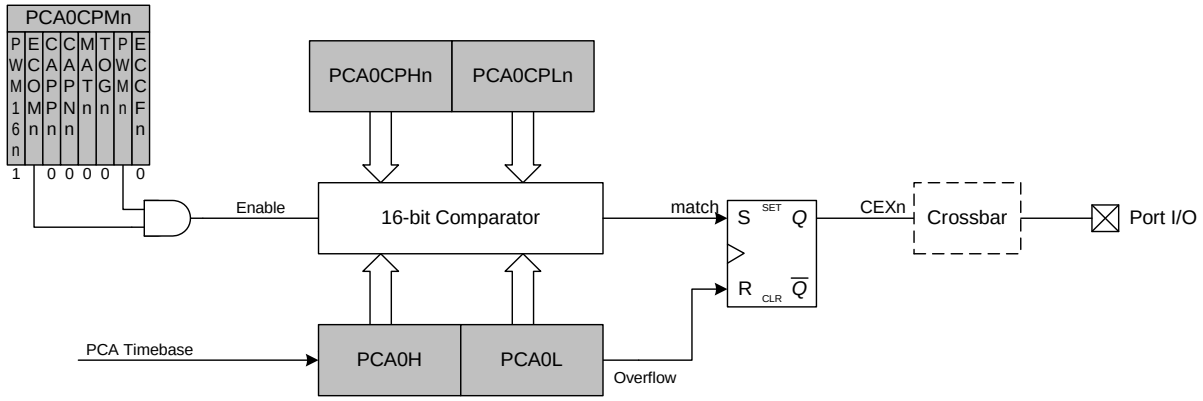


Figure 25.11. PCA0MD: PCA0 Mode Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
CIDL	-	-	-	CPS2	CPS1	CPS0	ECF	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xD9
SFR Page: 0

Bit7: CIDL: PCA0 Counter/Timer Idle Control.
Specifies PCA0 behavior when CPU is in Idle Mode.
0: PCA0 continues to function normally while the system controller is in Idle Mode.
1: PCA0 operation is suspended while the system controller is in Idle Mode.

Bits6-4: UNUSED. Read = 000b, Write = don't care.

Bits3-1: CPS2-CPS0: PCA0 Counter/Timer Pulse Select.
These bits select the timebase source for the PCA0 counter

CPS2	CPS1	CPS0	Timebase
0	0	0	System clock divided by 12
0	0	1	System clock divided by 4
0	1	0	Timer 0 overflow
0	1	1	High-to-low transitions on ECI (max rate = system clock divided by 4)
1	0	0	System clock
1	0	1	External clock divided by 8†
1	1	0	Reserved
1	1	1	Reserved

Bit0: ECF: PCA Counter/Timer Overflow Interrupt Enable.
This bit sets the masking of the PCA0 Counter/Timer Overflow (CF) interrupt.
0: Disable the CF interrupt.
1: Enable a PCA0 Counter/Timer Overflow interrupt request when CF (PCA0CN.7) is set.

†Note: External clock divided by 8 is synchronized with the system clock, and external clock must be less than or equal to the system clock frequency to operate in this mode.

26.3. Debug Support

Each MCU has on-chip JTAG and debug logic that provides non-intrusive, full speed, in-circuit debug support using the production part installed in the end application, via the four pin JTAG I/F. Silicon Labs' debug system supports inspection and modification of memory and registers, breakpoints, and single stepping. No additional target RAM, program memory, or communications channels are required. All the digital and analog peripherals are functional and work correctly (remain synchronized) while debugging. The Watchdog Timer (WDT) is disabled when the MCU is halted during single stepping or at a breakpoint.

The C8051F060DK is a development kit with all the hardware and software necessary to develop application code and perform in-circuit debug with each MCU in the C8051F06x family. Each kit includes development software for the PC, a Serial Adapter (for connection to JTAG) and a target application board with a C8051F060 installed. Serial cables and wall-mount power supply are also included.