



Welcome to [E-XFL.COM](https://www.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	Active
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	SMBus (2-Wire/I ² C), SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	25
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8.25K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.25V
Data Converters	A/D 25x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	32-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f583-iqr

C8051F58x/F59x

12.2.1.2. Bit Addressable Locations	105
12.2.1.3. Stack	105
13. Special Function Registers.....	106
13.1. SFR Paging	106
13.2. Interrupts and SFR Paging	106
13.3. SFR Page Stack Example	107
14. Interrupts	126
14.1. MCU Interrupt Sources and Vectors.....	126
14.1.1. Interrupt Priorities.....	127
14.1.2. Interrupt Latency	127
14.2. Interrupt Register Descriptions	129
14.3. External Interrupts INT0 and INT1.....	136
15. Flash Memory	138
15.1. Programming The Flash Memory	138
15.1.1. Flash Lock and Key Functions	138
15.1.2. Flash Erase Procedure	138
15.1.3. Flash Write Procedure	139
15.1.4. Flash Write Optimization	139
15.2. Non-volatile Data Storage	140
15.3. Security Options	140
15.4. Flash Write and Erase Guidelines.....	142
15.4.1. V _{DD} Maintenance and the V _{DD} monitor	142
15.4.2. PSWE Maintenance	142
15.4.3. System Clock	143
16. Power Management Modes.....	147
16.1. Idle Mode.....	147
16.2. Stop Mode	148
16.3. Suspend Mode	148
17. Reset Sources	150
17.1. Power-On Reset.....	151
17.2. Power-Fail Reset/VDD Monitor	152
17.3. External Reset.....	153
17.4. Missing Clock Detector Reset	153
17.5. Comparator0 Reset	154
17.6. PCA Watchdog Timer Reset	154
17.7. Flash Error Reset	154
17.8. Software Reset	154
18. External Data Memory Interface and On-Chip XRAM	156
18.1. Accessing XRAM.....	156
18.1.1. 16-Bit MOVX Example	156
18.1.2. 8-Bit MOVX Example	156
18.2. Configuring the External Memory Interface	157
18.3. Port Configuration.....	157
18.4. Multiplexed and Non-multiplexed Selection.....	162
18.4.1. Multiplexed Configuration.....	162

C8051F58x/F59x

21.4. LIN Slave Mode Operation	217
21.5. Sleep Mode and Wake-Up	218
21.6. Error Detection and Handling	218
21.7. LIN Registers.....	219
21.7.1. LIN Direct Access SFR Registers Definitions	219
21.7.2. LIN Indirect Access SFR Registers Definitions	221
22. Controller Area Network (CAN0)	229
22.1. Bosch CAN Controller Operation.....	230
22.1.1. CAN Controller Timing	230
22.1.2. CAN Register Access.....	231
22.1.3. Example Timing Calculation for 1 Mbit/Sec Communication	231
22.2. CAN Registers.....	233
22.2.1. CAN Controller Protocol Registers.....	233
22.2.2. Message Object Interface Registers	233
22.2.3. Message Handler Registers.....	233
22.2.4. CAN Register Assignment	234
23. SMBus.....	237
23.1. Supporting Documents	238
23.2. SMBus Configuration.....	238
23.3. SMBus Operation	238
23.3.1. Transmitter Vs. Receiver.....	239
23.3.2. Arbitration.....	239
23.3.3. Clock Low Extension.....	239
23.3.4. SCL Low Timeout.....	239
23.3.5. SCL High (SMBus Free) Timeout	240
23.4. Using the SMBus.....	240
23.4.1. SMBus Configuration Register.....	240
23.4.2. SMB0CN Control Register	244
23.4.3. Data Register	247
23.5. SMBus Transfer Modes.....	247
23.5.1. Write Sequence (Master)	248
23.5.2. Read Sequence (Master)	249
23.5.3. Write Sequence (Slave)	250
23.5.4. Read Sequence (Slave)	251
23.6. SMBus Status Decoding.....	251
24. UART0	254
24.1. Baud Rate Generator	254
24.2. Data Format.....	256
24.3. Configuration and Operation	257
24.3.1. Data Transmission	257
24.3.2. Data Reception	257
24.3.3. Multiprocessor Communications	258
25. UART1	263
25.1. Enhanced Baud Rate Generation.....	264
25.2. Operational Modes	265

C8051F58x/F59x

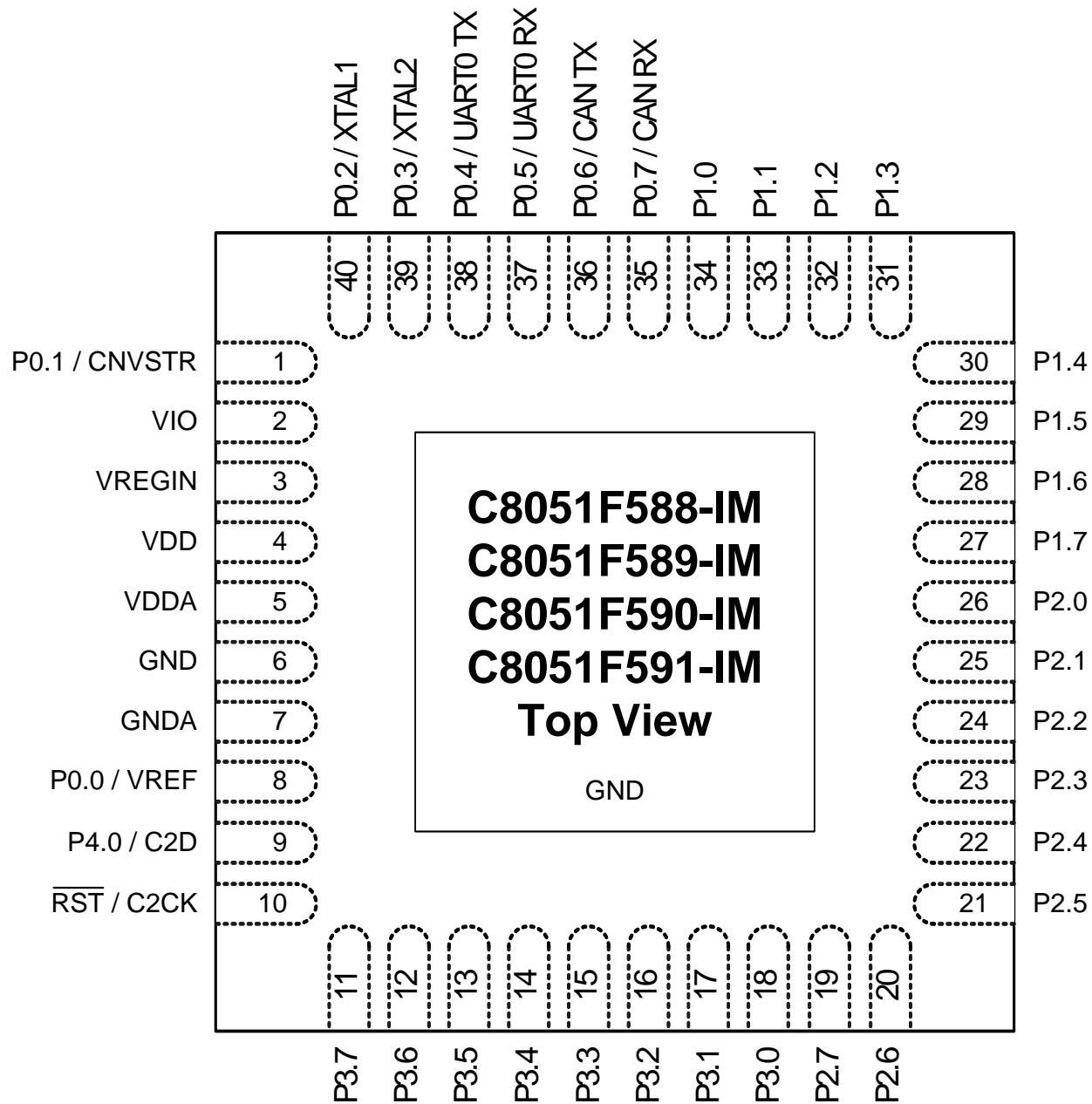


Figure 3.3. QFN-40 Pinout Diagram (Top View)

C8051F58x/F59x

SFR Definition 9.9. CPT2MX: Comparator2 MUX Selection

Bit	7	6	5	4	3	2	1	0
Name	CMX2N[3:0]				CMX2P[3:0]			
Type	R/W				R/W			
Reset	0	1	1	1	0	1	1	1

SFR Address = 0x9C; SFR Page = 0x10

Bit	Name	Function
7:4	CMX2N[3:0]	Comparator2 Negative Input MUX Selection. 0000: P0.1 0001: P0.3 0010: P0.5 0011: P0.7 0100: P1.1 0101: P1.3 0110: P1.5 0111: P1.7 1000: P2.1 1001: P2.3 1010: P2.5 1011: P2.7 1100–1111: None
3:0	CMX2P[3:0]	Comparator2 Positive Input MUX Selection. 0000: P0.0 0001: P0.2 0010: P0.4 0011: P0.6 0100: P1.0 0101: P1.2 0110: P1.4 0111: P1.6 1000: P2.0 1001: P2.2 1010: P2.4 1011: P2.6 1100–1111: None

SFR Definition 12.1. PSBANK: Program Space Bank Select

Bit	7	6	5	4	3	2	1	0
Name			COBANK[1:0]				IFBANK[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	0	1

SFR Address = 0xF5; SFR Page = All Pages

Bit	Name	Function
7:6	Reserved	Read = 00b, Must Write = 00b.
5:4	COBANK[1:0]	Constant Operations Bank Select. These bits select which Flash bank is targeted during constant operations (MOVC and Flash MOVX) involving address 0x8000 to 0xFFFF. 00: Constant Operations Target Bank 0 (note that Bank 0 is also mapped between 0x0000 to 0x7FFF). 01: Constant operations target Bank 1. 10: Constant operations target Bank 2. 11: Constant operations target Bank 3.
3:2	Reserved	Read = 00b, Must Write = 00b.
1:0	IFBANK[1:0]	Instruction Fetch Operations Bank Select. These bits select which Flash bank is used for instruction fetches involving address 0x8000 to 0xFFFF. These bits can only be changed from code in Bank 0. 00: Instructions fetch from Bank 0 (note that Bank 0 is also mapped between 0x0000 to 0x7FFF). 01: Instructions fetch from Bank 1. 10: Instructions fetch from Bank 2. 11: Instructions fetch from Bank 3.
Note: COBANK[1:0] and IFBANK[1:0] should not be set to select Bank 3 (11b) on the C8051F584/5/6/7-F590/1 devices.		

12.1.1. MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the C8051F58x/F59x devices, the MOVX instruction is normally used to read and write on-chip XRAM, but can be re-configured to write and erase on-chip Flash memory space. MOVC instructions are always used to read Flash memory, while MOVX write instructions are used to erase and write Flash. This Flash access feature provides a mechanism for the C8051F58x/F59x to update program code and use the program memory space for non-volatile data storage. Refer to Section “15. Flash Memory” on page 138 for further details.

12.2. Data Memory

The C8051F58x/F59x devices include 8448 bytes of RAM data memory. 256 bytes of this memory is mapped into the internal RAM space of the 8051. The other 8192 bytes of this memory is on-chip “external” memory. The data memory map is shown in Figure 12.1 for reference.

15.2. Non-volatile Data Storage

The Flash memory can be used for non-volatile data storage as well as program code. This allows data such as calibration coefficients to be calculated and stored at run time. Data is written using the MOVX write instruction and read using the MOVC instruction. Note that MOVX read instructions always target XRAM.

15.3. Security Options

The CIP-51 provides security options to protect the Flash memory from inadvertent modification by software as well as to prevent the viewing of proprietary program code and constants. The Program Store Write Enable (bit PSWE in register PSCTL) and the Program Store Erase Enable (bit PSEE in register PSCTL) bits protect the Flash memory from accidental modification by software. PSWE must be explicitly set to 1 before software can modify the Flash memory; both PSWE and PSEE must be set to 1 before software can erase Flash memory. Additional security features prevent proprietary program code and data constants from being read or altered across the C2 interface.

A Security Lock Byte located at the last byte of Flash user space offers protection of the Flash program memory from access (reads, writes, or erases) by unprotected code or the C2 interface. The Flash security mechanism allows the user to lock n 512-byte Flash pages, starting at page 0 (addresses 0x0000 to 0x01FF), where n is the ones complement number represented by the Security Lock Byte. **Note that the page containing the Flash Security Lock Byte is unlocked when no other Flash pages are locked (all bits of the Lock Byte are 1) and locked when any other Flash pages are locked (any bit of the Lock Byte is 0).** See example in Figure 15.1.

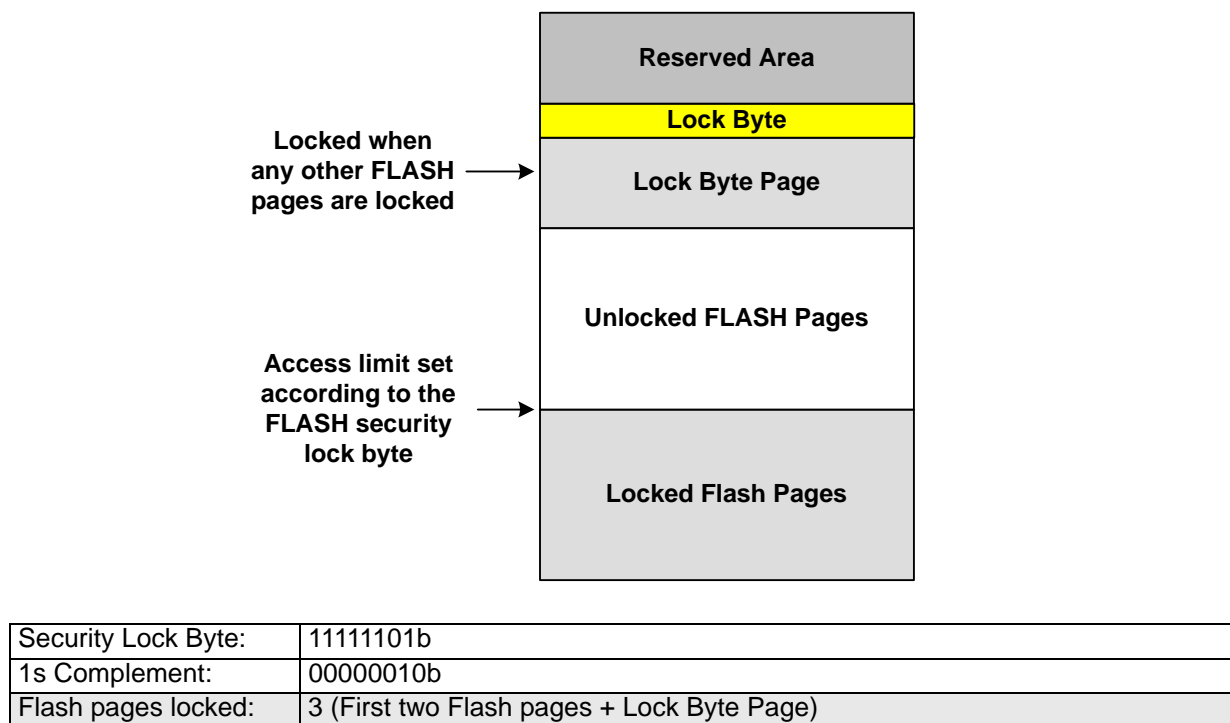


Figure 15.1. Flash Program Memory Map

The level of Flash security depends on the Flash access method. The three Flash access methods that can be restricted are reads, writes, and erases from the C2 debug interface, user firmware executing on unlocked pages, and user firmware executing on locked pages. Table 15.1 summarizes the Flash security features of the C8051F58x/F59x devices.

Table 15.1. Flash Security Summary

Action	C2 Debug Interface	User Firmware executing from:	
		an unlocked page	a locked page
Read, Write or Erase unlocked pages (except page with Lock Byte)	Permitted	Permitted	Permitted
Read, Write or Erase locked pages (except page with Lock Byte)	Not Permitted	Flash Error Reset	Permitted
Read or Write page containing Lock Byte (if no pages are locked)	Permitted	Permitted	Permitted
Read or Write page containing Lock Byte (if any page is locked)	Not Permitted	Flash Error Reset	Permitted
Read contents of Lock Byte (if no pages are locked)	Permitted	Permitted	Permitted
Read contents of Lock Byte (if any page is locked)	Not Permitted	Flash Error Reset	Permitted
Erase page containing Lock Byte (if no pages are locked)	Permitted	Flash Error Reset	Flash Error Reset
Erase page containing Lock Byte—Unlock all pages (if any page is locked)	C2 Device Erase Only	Flash Error Reset	Flash Error Reset
Lock additional pages (change 1s to '0's in the Lock Byte)	Not Permitted	Flash Error Reset	Flash Error Reset
Unlock individual pages (change 0s to 1s in the Lock Byte)	Not Permitted	Flash Error Reset	Flash Error Reset
Read, Write or Erase Reserved Area	Not Permitted	Flash Error Reset	Flash Error Reset
<p>C2 Device Erase—Erases all Flash pages including the page containing the Lock Byte.</p> <p>Flash Error Reset—Not permitted; Causes Flash Error Device Reset (FERROR bit in RSTSRC is '1' after reset).</p> <ul style="list-style-type: none"> - All prohibited operations that are performed via the C2 interface are ignored (do not cause device reset). - Locking any Flash page also locks the page containing the Lock Byte. - Once written to, the Lock Byte cannot be modified except by performing a C2 Device Erase. - If user code writes to the Lock Byte, the Lock does not take effect until the next device reset. 			

17.1. Power-On Reset

During power-up, the device is held in a reset state and the $\overline{\text{RST}}$ pin is driven low until V_{DD} settles above V_{RST} . A delay occurs before the device is released from reset; the delay decreases as the V_{DD} ramp time increases (V_{DD} ramp time is defined as how fast V_{DD} ramps from 0 V to V_{RST}). Figure 17.2. plots the power-on and V_{DD} monitor reset timing. The maximum V_{DD} ramp time is 1 ms; slower ramp times may cause the device to be released from reset before V_{DD} reaches the V_{RST} level. For ramp times less than 1 ms, the power-on reset delay (T_{PORDelay}) is typically less than 0.3 ms.

On exit from a power-on reset, the PORSF flag (RSTSRC.1) is set by hardware to logic 1. When PORSF is set, all of the other reset flags in the RSTSRC Register are indeterminate (PORSF is cleared by all other resets). Since all resets cause program execution to begin at the same location (0x0000) software can read the PORSF flag to determine if a power-up was the cause of reset. The content of internal data memory should be assumed to be undefined after a power-on reset. The V_{DD} monitor is enabled following a power-on reset.

Note: For devices with a date code before year 2011, work week 24 (1124), if the $\overline{\text{RST}}$ pin is held low for more than 1 second while power is applied to the device, and then $\overline{\text{RST}}$ is released, a percentage of devices may lock up and fail to execute code. Toggling the $\overline{\text{RST}}$ pin does not clear the condition. The condition is cleared by cycling power. Most devices that are affected will show the lock up behavior only within a narrow range of temperatures (a 5 to 10 °C window). Parts with a date code of year 2011, work week 24 (1124) or later do not have any restrictions on $\overline{\text{RST}}$ low time. The date code is included in the bottom-most line of the package top side marking. The date code is a four-digit number with the format YYWW, where YY is the two-digit calendar year and WW is the two digit work week.

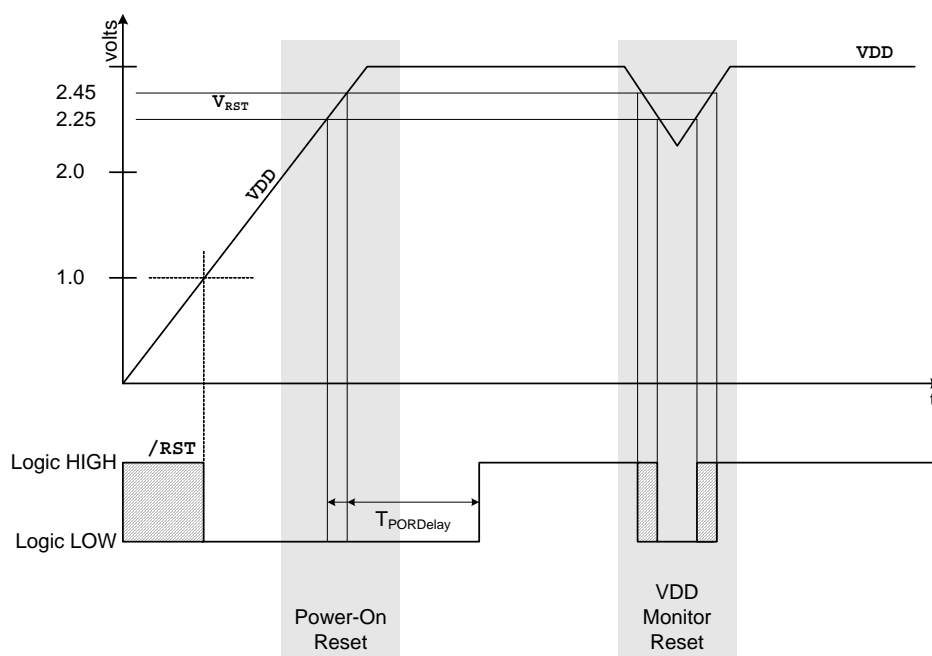


Figure 17.2. Power-On and V_{DD} Monitor Reset Timing

Table 18.2. EMIF Pinout (C8051F588/9-F590/1)

Multiplexed Mode	
Signal Name	Port Pin
$\overline{\text{RD}}$	P1.6
$\overline{\text{WR}}$	P1.7
ALE	P1.5
D0/A0	P3.0
D1/A1	P3.1
D2/A2	P3.2
D3/A3	P3.3
D4/A4	P3.4
D5/A5	P3.5
D6/A6	P3.6
D7/A7	P3.7
A8	P2.0
A9	P2.1
A10	P2.2
A11	P2.3
A12	P2.4
A13	P2.5
A14	P2.6
A15	P2.7

SFR Definition 18.1. EMI0CN: External Memory Interface Control

Bit	7	6	5	4	3	2	1	0
Name	PGSEL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xAA; SFR Page = 0x00

Bit	Name	Function
7:0	PGSEL[7:0]	XRAM Page Select Bits. The XRAM Page Select Bits provide the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. 0x00: 0x0000 to 0x00FF 0x01: 0x0100 to 0x01FF ... 0xFE: 0xFE00 to 0xFEFF 0xFF: 0xFF00 to 0xFFFF

3. The LIN controller does not directly support LIN Version 1.3 Extended Frames. If the application detects an unknown identifier (e.g. extended identifier), it has to write a 1 to the STOP bit (LIN0CTRL.7) instead of setting the DTACK (LIN0CTRL.4) bit. At that time, steps 2 through 5 can then be skipped. In this situation, the LIN controller stops the processing of LIN communication until the next SYNC BREAK is received.
4. Changing the configuration of the checksum during a transaction will cause the interface to reset and the transaction to be lost. To prevent this, the checksum should not be configured while a transaction is in progress. The same applies to changes in the LIN interface mode from slave mode to master mode and from master mode to slave mode.

21.5. Sleep Mode and Wake-Up

To reduce the system's power consumption, the LIN Protocol Specification defines a Sleep Mode. The message used to broadcast a Sleep Mode request must be transmitted by the LIN master application in the same way as a normal transmit message. The LIN slave application must decode the Sleep Mode Frame from the Identifier and data bytes. After that, it has to put the LIN slave node into the Sleep Mode by setting the SLEEP bit (LIN0CTRL.6).

If the SLEEP bit (LIN0CTRL.6) of the LIN slave application is not set and there is no bus activity for four seconds (specified bus idle timeout), the IDLTOUT bit (LIN0ST.6) is set and an interrupt request is generated. After that the application may assume that the LIN bus is in Sleep Mode and set the SLEEP bit (LIN0CTRL.6).

Sending a wake-up signal from the master or any slave node terminates the Sleep Mode of the LIN bus. To send a wake-up signal, the application has to set the WUPREQ bit (LIN0CTRL.1). After successful transmission of the wake-up signal, the DONE bit (LIN0ST.0) of the master node is set and an interrupt request is generated. The LIN slave does not generate an interrupt request after successful transmission of the wake-up signal but it generates an interrupt request if the master does not respond to the wake-up signal within 150 milliseconds. In that case, the ERROR bit (LIN0ST.2) and TOUT bit (LIN0ERR.2) are set. The application then has to decide whether or not to transmit another wake-up signal.

All LIN nodes that detect a wake-up signal will set the WAKEUP (LIN0ST.1) and DONE bits (LIN0ST.0) and generate an interrupt request. After that, the application has to clear the SLEEP bit (LIN0CTRL.6) in the LIN slave.

21.6. Error Detection and Handling

The LIN controller generates an interrupt request and stops the processing of the current frame if it detects an error. The application has to check the type of error by processing LIN0ERR. After that, it has to reset the error register and the ERROR bit (LIN0ST.2) by writing a 1 to the RSTERR bit (LIN0CTRL.2). Starting a new message with the LIN controller selected as master or sending a Wakeup signal with the LIN controller selected as a master or slave is possible only if the ERROR bit (LIN0ST.2) is set to 0.

C8051F58x/F59x

21.7.2. LIN Indirect Access SFR Registers Definitions

Table 21.4 lists the 15 indirect registers used to configured and communicate with the LIN controller.

Table 21.4. LIN Registers* (Indirectly Addressable)

Name	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LIN0DT1	0x00	DATA1[7:0]							
LIN0DT2	0x01	DATA2[7:0]							
LIN0DT3	0x02	DATA3[7:0]							
LIN0DT4	0x03	DATA4[7:0]							
LIN0DT5	0x04	DATA5[7:0]							
LIN0DT6	0x05	DATA6[7:0]							
LIN0DT7	0x06	DATA7[7:0]							
LIN0DT8	0x07	DATA8[7:0]							
LIN0CTRL	0x08	STOP(s)	SLEEP(s)	TXRX	DTACK(s)	RSTINT	RSTERR	WUPREQ	STREQ(m)
LIN0ST	0x09	ACTIVE	IDLTOU	ABORT(s)	DTREQ(s)	LININT	ERROR	WAKEUP	DONE
LIN0ERR	0x0A				SYNCH(s)	PRTY(s)	TOUT	CHK	BITERR
LIN0SIZE	0x0B	ENHCHK				LINSIZE[3:0]			
LIN0DIV	0x0C	DIVLSB[7:0]							
LIN0MUL	0x0D	PRESCL[1:0]		LINMUL[4:0]					DIV9
LIN0ID	0x0E			ID5	ID4	ID3	ID2	ID1	ID0
*Note: These registers are used in both master and slave mode. The register bits marked with (m) are accessible only in Master mode while the register bits marked with (s) are accessible only in slave mode. All other registers are accessible in both modes.									

C8051F58x/F59x

LIN Register Definition 21.7. LIN0ERR: LIN0 Error Register

Bit	7	6	5	4	3	2	1	0
Name				SYNCH	PRTY	TOUT	CHK	BITERR
Type	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Indirect Address = 0x0A

Bit	Name	Function
7:5	Unused	Read = 000b; Write = Don't Care
4	SYNCH	Synchronization Error Bit (slave mode only). 0: No error with the SYNCH FIELD has been detected. 1: Edges of the SYNCH FIELD are outside of the maximum tolerance.
3	PRTY	Parity Error Bit (slave mode only). 0: No parity error has been detected. 1: A parity error has been detected.
2	TOUT	Timeout Error Bit. 0: A timeout error has not been detected. 1: A timeout error has been detected. This error is detected whenever one of the following conditions is met: <ul style="list-style-type: none"> • The master is expecting data from a slave and the slave does not respond. • The slave is expecting data but no data is transmitted on the bus. • A frame is not finished within the maximum frame length. • The application does not set the DTACK bit (LIN0CTRL.4) or STOP bit (LIN0CTRL.7) until the end of the reception of the first byte after the identifier.
1	CHK	Checksum Error Bit. 0: Checksum error has not been detected. 1: Checksum error has been detected.
0	BITERR	Bit Transmission Error Bit. 0: No error in transmission has been detected. 1: The bit value monitored during transmission is different than the bit value sent.

22. Controller Area Network (CAN0)

Important Documentation Note: The Bosch CAN Controller is integrated in the C8051F580/2/4/6/8-F590 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, refer to Bosch's C_CAN User's Manual as an accompanying manual to the Silicon Labs' data sheet.

The C8051F580/2/4/6/8-F590 devices feature a Control Area Network (CAN) controller that enables serial communication using the CAN protocol. Silicon Labs CAN 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 22.1 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 (SFRs) in the CIP-51.

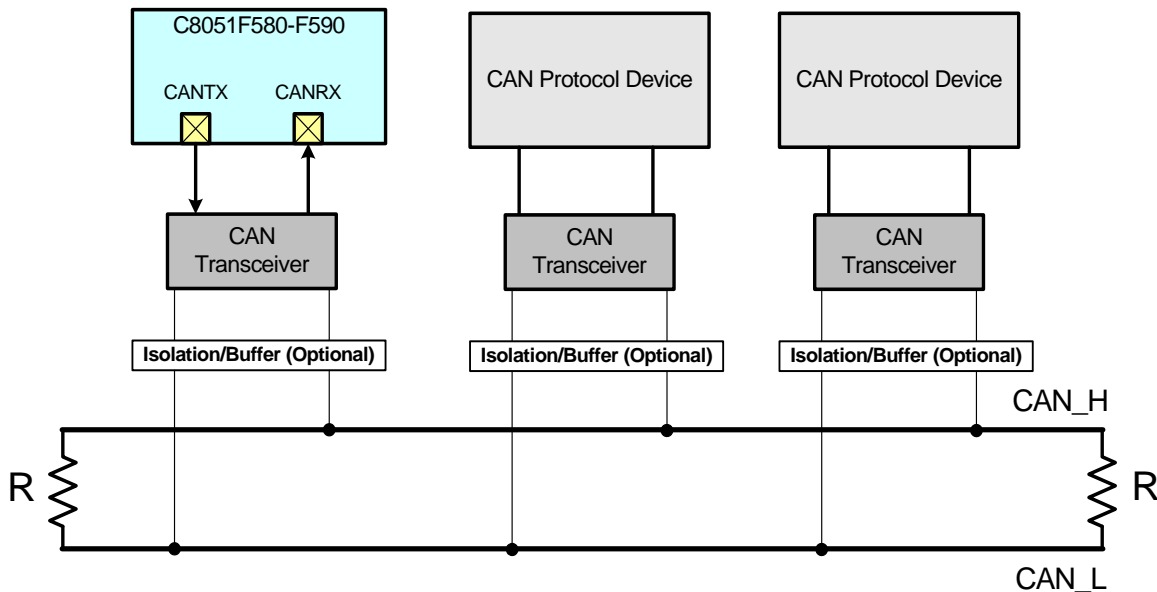


Figure 22.1. Typical CAN Bus Configuration

The CAN controller clock must be less than or equal to 25 MHz. If the CIP-51 system clock is above 25 MHz, the divider in the CAN0CFG register must be set to divide the CAN controller clock down to an appropriate speed.

22.1.2. CAN Register Access

The CAN controller clock divider selected in the CAN0CFG SFR affects how the CAN registers can be accessed. If the divider is set to 1, then a CAN SFR can immediately be read after it is written. If the divider is set to a value other than 1, then a read of a CAN SFR that has just been written must be delayed by a certain number of cycles. This delay can be performed using a NOP or some other instruction that does not attempt to read the register. This access limitation applies to read and read-modify-write instructions that occur immediately after a write. The full list of affected instructions is ANL, ORL, MOV, XCH, and XRL.

For example, with the CAN0CFG divider set to 1, the CAN0CN SFR can be accessed as follows:

```
MOV CAN0CN, #041      ; Enable access to Bit Timing Register
MOV R7, CAN0CN         ; Copy CAN0CN to R7
```

With the CAN0CFG divider set to /2, the same example code requires an additional NOP:

```
MOV CAN0CN, #041      ; Enable access to Bit Timing Register
NOP                   ; Wait for write to complete
MOV R7, CAN0CN         ; Copy CAN0CN to R7
```

The number of delay cycles required is dependent on the divider setting. With a divider of 2, the read must wait for 1 system clock cycle. With a divider of 4, the read must wait 3 system clock cycles, and with the divider set to 8, the read must wait 7 system clock cycles. The delay only needs to be applied when reading the same register that was written. The application can write and read other CAN SFRs without any delay.

22.1.3. Example Timing Calculation for 1 Mbit/Sec Communication

This example shows how to configure the CAN controller timing parameters for a 1 Mbit/Sec bit rate. Table 18.1 shows timing-related system parameters needed for the calculation.

Table 22.1. Background System Information

Parameter	Value	Description
CIP-51 system clock (SYSCLK)	24 MHz	Internal Oscillator Max
CAN controller clock (f _{sys})	24 MHz	CAN0CFG divider set to 1
CAN clock period (t _{sys})	41.667 ns	Derived from 1/f _{sys}
CAN time quantum (t _q)	41.667 ns	Derived from t _{sys} x BRP ^{1,2}
CAN bus length	10 m	5 ns/m signal delay between CAN nodes
Propagation delay time ³	400 ns	2 x (transceiver loop delay + bus line delay)
Notes: <ol style="list-style-type: none"> 1. The CAN time quantum is the smallest unit of time recognized by the CAN controller. Bit timing parameters are specified in integer multiples of the time quantum. 2. The Baud Rate Prescaler (BRP) is defined as the value of the BRP Extension Register plus 1. The BRP extension register has a reset value of 0x0000. The BRP has a reset value of 1. 3. Based on an ISO-11898 compliant transceiver. CAN does not specify a physical layer. 		

Each bit transmitted on a CAN network has 4 segments (Sync_Seg, Prop_Seg, Phase_Seg1, and Phase_Seg2), as shown in Figure 18.3. The sum of these segments determines the CAN bit time (1/bit rate). In this example, the desired bit rate is 1 Mbit/sec; therefore, the desired bit time is 1000 ns.

22.2.4. CAN Register Assignment

The standard Bosch CAN registers are mapped to SFR space as shown below and their full definitions are available in the CAN User's Guide. The name shown in the Name column matches what is provided in the CAN User's Guide. One additional SFR which is not a standard Bosch CAN register, CAN0CFG, is provided to configure the CAN clock. All CAN registers are located on SFR Page 0x0C.

Table 22.2. Standard CAN Registers and Reset Values

CAN Addr.	Name	SFR Name (High)	SFR Addr.	SFR Name (Low)	SFR Addr.	16-bit SFR	Reset Value
0x00	CAN Control Register	—	—	CAN0CN	0xC0	—	0x01
0x02	Status Register	—	—	CAN0STAT	0x94	—	0x00
0x04	Error Counter ¹	CAN0ERRH	0x97	CAN0ERRL	0x96	CAN0ERR	0x0000
0x06	Bit Timing Register ²	CAN0BTH	0x9B	CAN0BTL	0x9A	CAN0BT	0x2301
0x08	Interrupt Register ¹	CAN0IIDH	0x9D	CAN0IIDL	0x9C	CAN0IID	0x0000
0x0A	Test Register	—	—	CAN0TST	0x9E	—	0x00 ^{3,4}
0x0C	BRP Extension Register ²	—	—	CAN0BRPE	0xA1	—	0x00
0x10	IF1 Command Request	CAN0IF1CRH	0xBF	CAN0IF1CRL	0xBE	CAN0IF1CR	0x0001
0x12	IF1 Command Mask	CAN0IF1CMH	0xC3	CAN0IF1CML	0xC2	CAN0IF1CM	0x0000
0x14	IF1 Mask 1	CAN0IF1M1H	0xC5	CAN0IF1M1L	0xC4	CAN0IF1M1	0xFFFF
0x16	IF1 Mask 2	CAN0IF1M2H	0xC7	CAN0IF1M2L	0xC6	CAN0IF1M2	0xFFFF
0x18	IF1 Arbitration 1	CAN0IF1A1H	0xCB	CAN0IF1A1L	0xCA	CAN0IF1A1	0x0000
0x1A	IF1 Arbitration 2	CAN0IF1A2H	0xCD	CAN0IF1A2L	0xCC	CAN0IF1A2	0x0000
0x1C	IF1 Message Control	CAN0IF1MCH	0xD3	CAN0IF1MCL	0xD2	CAN0IF1MC	0x0000
0x1E	IF1 Data A 1	CAN0IF1DA1H	0xD5	CAN0IF1DA1L	0xD4	CAN0IF1DA1	0x0000
0x20	IF1 Data A 2	CAN0IF1DA2H	0xD7	CAN0IF1DA2L	0xD6	CAN0IF1DA2	0x0000
0x22	IF1 Data B 1	CAN0IF1DB1H	0xDB	CAN0IF1DB1L	0xDA	CAN0IF1DB1	0x0000
0x24	IF1 Data B 2	CAN0IF1DB2H	0xDD	CAN0IF1DB2L	0xDC	CAN0IF1DB2	0x0000
0x40	IF2 Command Request	CAN0IF2CRH	0xDF	CAN0IF2CRL	0xDE	CAN0IF2CR	0x0001
0x42	IF2 Command Mask	CAN0IF2CMH	0xE3	CAN0IF2CML	0xE2	CAN0IF2CM	0x0000
0x44	IF2 Mask 1	CAN0IF2M1H	0xEB	CAN0IF2M1L	0xEA	CAN0IF2M1	0xFFFF
0x46	IF2 Mask 2	CAN0IF2M2H	0xED	CAN0IF2M2L	0xEC	CAN0IF2M2	0xFFFF
0x48	IF2 Arbitration 1	CAN0IF2A1H	0xEF	CAN0IF2A1L	0xEE	CAN0IF2A1	0x0000
0x4A	IF2 Arbitration 2	CAN0IF2A2H	0xF3	CAN0IF2A2L	0xF2	CAN0IF2A2	0x0000
0x4C	IF2 Message Control	CAN0IF2MCH	0xCF	CAN0IF2MCL	0xCE	CAN0IF2MC	0x0000
0x4E	IF2 Data A 1	CAN0IF2DA1H	0xF7	CAN0IF2DA1L	0xF6	CAN0IF2DA1	0x0000

Notes:

1. Read-only register.
2. Write-enabled by CCE.
3. The reset value of CAN0TST could also be r0000000b, where r signifies the value of the CAN RX pin.
4. Write-enabled by Test.

Table 22.2. Standard CAN Registers and Reset Values (Continued)

CAN Addr.	Name	SFR Name (High)	SFR Addr.	SFR Name (Low)	SFR Addr.	16-bit SFR	Reset Value
0x50	IF2 Data A 2	CAN0IF2DA2H	0xFB	CAN0IF2DA2L	0xFA	CAN0IF2DA2	0x0000
0x52	IF2 Data B 1	CAN0IF2DB1H	0xFD	CAN0IF2DB1L	0xFC	CAN0IF2DB1	0x0000
0x54	IF2 Data B 2	CAN0IF2DB2H	0xFF	CAN0IF2DB2L	0xFE	CAN0IF2DB2	0x0000
0x80	Transmission Request 1 ¹	CAN0TR1H	0xA3	CAN0TR1L	0xA2	CAN0TR1	0x0000
0x82	Transmission Request 2 ¹	CAN0TR2H	0xA5	CAN0TR2L	0xA4	CAN0TR2	0x0000
0x90	New Data 1 ¹	CAN0ND1H	0xAB	CAN0ND1L	0xAA	CAN0ND1	0x0000
0x92	New Data 2 ¹	CAN0ND2H	0xAD	CAN0ND2L	0xAC	CAN0ND2	0x0000
0xA0	Interrupt Pending 1 ¹	CAN0IP1H	0xAF	CAN0IP1L	0xAE	CAN0IP1	0x0000
0xA2	Interrupt Pending 2 ¹	CAN0IP2H	0xB3	CAN0IP2L	0xB2	CAN0IP2	0x0000
0xB0	Message Valid 1 ¹	CAN0MV1H	0xBB	CAN0MV1L	0xBA	CAN0MV1	0x0000
0xB2	Message Valid 2 ¹	CAN0MV2H	0xBD	CAN0MV2L	0xBC	CAN0MV2	0x0000

Notes:

1. Read-only register.
2. Write-enabled by CCE.
3. The reset value of CAN0TST could also be r0000000b, where r signifies the value of the CAN RX pin.
4. Write-enabled by Test.

C8051F58x/F59x

SFR Definition 27.1. CKCON: Clock Control

Bit	7	6	5	4	3	2	1	0
Name	T3MH	T3ML	T2MH	T2ML	T1M	T0M	SCA[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8E; SFR Page = All Pages

Bit	Name	Function
7	T3MH	Timer 3 High Byte Clock Select. Selects the clock supplied to the Timer 3 high byte (split 8-bit timer mode only). 0: Timer 3 high byte uses the clock defined by the T3XCLK bit in TMR3CN. 1: Timer 3 high byte uses the system clock.
6	T3ML	Timer 3 Low Byte Clock Select. Selects the clock supplied to Timer 3. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode. 0: Timer 3 low byte uses the clock defined by the T3XCLK bit in TMR3CN. 1: Timer 3 low byte uses the system clock.
5	T2MH	Timer 2 High Byte Clock Select. Selects the clock supplied to the Timer 2 high byte (split 8-bit timer mode only). 0: Timer 2 high byte uses the clock defined by the T2XCLK bit in TMR2CN. 1: Timer 2 high byte uses the system clock.
4	T2ML	Timer 2 Low Byte Clock Select. Selects the clock supplied to Timer 2. If Timer 2 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer. 0: Timer 2 low byte uses the clock defined by the T2XCLK bit in TMR2CN. 1: Timer 2 low byte uses the system clock.
3	T1	Timer 1 Clock Select. Selects the clock source supplied to Timer 1. Ignored when C/T1 is set to 1. 0: Timer 1 uses the clock defined by the prescale bits SCA[1:0]. 1: Timer 1 uses the system clock.
2	T0	Timer 0 Clock Select. Selects the clock source supplied to Timer 0. Ignored when C/T0 is set to 1. 0: Counter/Timer 0 uses the clock defined by the prescale bits SCA[1:0]. 1: Counter/Timer 0 uses the system clock.
1:0	SCA[1:0]	Timer 0/1 Prescale Bits. These bits control the Timer 0/1 Clock Prescaler: 00: System clock divided by 12 01: System clock divided by 4 10: System clock divided by 48 11: External clock divided by 8 (synchronized with the system clock)



Timers 4 and 5 have the capability to toggle the state of their respective output port pins (T4 or T5) to produce a 50% duty cycle waveform output. The port pin state will change upon the overflow or underflow of the respective timer (depending on whether the timer is counting *up* or *down*). The toggle frequency is determined by the clock source of the timer and the values loaded into TMRnCAPH and TMRnCAPL. When counting down, the auto-reload value for the timer is 0xFFFF, and underflow will occur when the value in the timer matches the value stored in TMRnCAPH:TMRCAPL. When counting up, the auto-reload value for the timer is TMRnCAPH:TMRCAPL, and overflow will occur when the value in the timer transitions from 0xFFFF to the reload value.

To output a square wave, the timer is placed in reload mode (the Capture/Reload Select Bit in TMRnCN and the Timer/Counter Select Bit in TMRnCN are cleared to 0). The timer output is enabled by setting the Timer Output Enable Bit in TMRnCF to 1. The timer should be configured via the timer clock source and reload/underflow values such that the timer overflow/underflows at 1/2 the desired output frequency. The port pin assigned by the crossbar as the timer's output pin should be configured as a digital output (see [Section “20. Port Input/Output” on page 188](#)). Setting the timer's Run Bit (TRn) to 1 will start the toggle of the pin. A Read/Write of the Timer's Toggle Output State Bit (TMRnCF.2) is used to read the state of the toggle output, or to force a value of the output. This is useful when it is desired to start the toggle of a pin in a known state, or to force the pin into a desired state when the toggle mode is halted.

$$F_{sq} = \frac{F_{TCLK}}{2 \times (65536 - TMRnCAP)}$$

Equation 27.1. Square Wave Frequency

29.3.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEX_n pin causes PCA1 to capture the value of the PCA1 counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA1CPL_n and PCA1CPH_n). The CAPP_{1n} and CAPN_{1n} bits in the PCA1CPM_n register are used to select the type of transition that triggers the capture: low-to-high transition (positive edge), high-to-low transition (negative edge), or either transition (positive or negative edge). When a capture occurs, the Capture/Compare Flag (CCF_n) in PCA1CN is set to logic 1. An interrupt request is generated if the CCF_n interrupt for that module is enabled. The CCF_n bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. If both CAPP_{1n} and CAPN_{1n} bits are set to logic 1, then the state of the Port pin associated with CEX_n can be read directly to determine whether a rising-edge or falling-edge caused the capture.

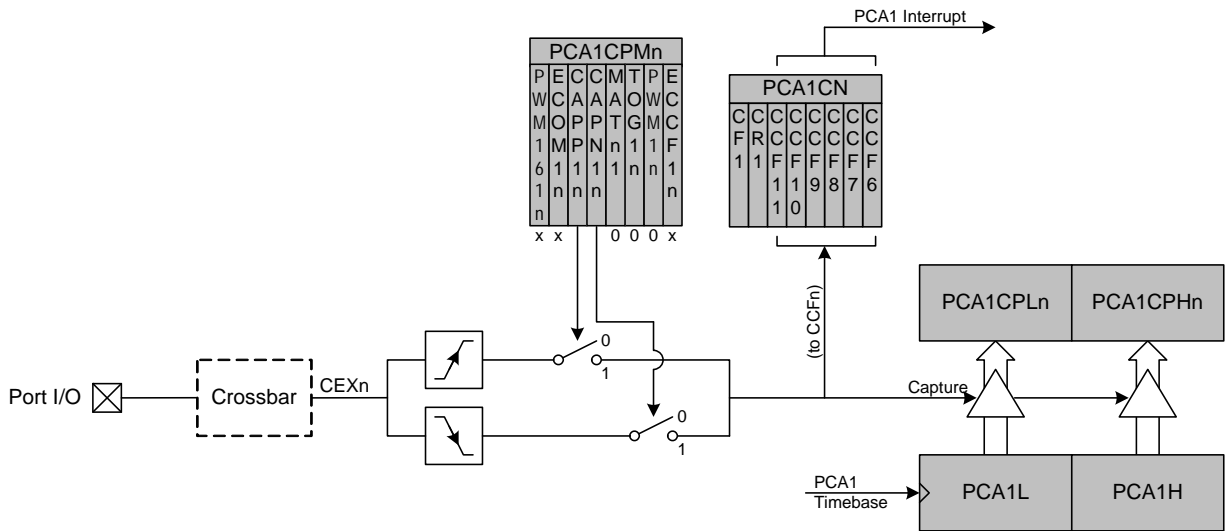


Figure 29.4. PCA1 Capture Mode Diagram

Note: The CEX_n input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.