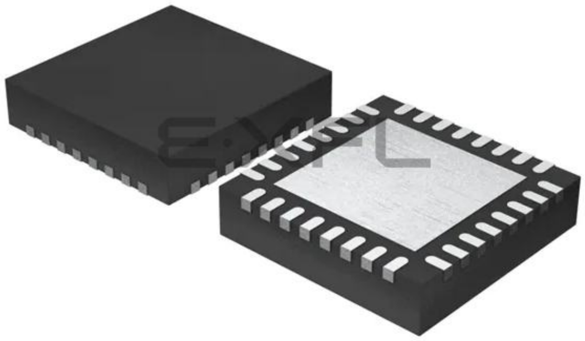
Silicon Labs - C8051F587-IMR Datasheet



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	Active
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	25
Program Memory Size	96KB (96K 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-VFQFN Exposed Pad
Supplier Device Package	32-QFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f587-imr

List of Tables

Table 5.10. ADC0 Electrical Characteristics

VDDA = 1.8 to 2.75 V, –40 to +125 °C, VREF = 1.5 V (REFSL=0) unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
DC Accuracy					
Resolution		12			bits
Integral Nonlinearity		—	±0.5	±3	LSB
Differential Nonlinearity	Guaranteed Monotonic	—	±0.5	±1	LSB
Offset Error[1]		–10	–1.6	10	LSB
Full Scale Error		–20	–4.2	20	LSB
Offset Temperature Coefficient		—	–2	—	ppm/°C
Dynamic performance (10 kHz sine-wave single-ended input, 1 dB below Full Scale, 200 ksps)					
Signal-to-Noise Plus Distortion		63	66	—	dB
Total Harmonic Distortion	Up to the 5th harmonic	—	81	—	dB
Spurious-Free Dynamic Range		—	–82	—	dB
Conversion Rate					
SAR Conversion Clock		—	—	3.6	MHz
Conversion Time in SAR Clocks[2]		13	—	—	clocks
Track/Hold Acquisition Time[3]	VDDA ≥2.0 V VDDA < 2.0 V	1.5 3.5	—	—	µs
Throughput Rate[4]	VDDA ≥2.0 V	—	—	200	ksps
Analog Inputs					
ADC Input Voltage Range[5]	gain = 1.0 (default) gain = n	0 0	—	VREF VREF / n	V
Absolute Pin Voltage with respect to GND		0	—	V_{IO}	V
Sampling Capacitance		—	29	—	pF
Input Multiplexer Impedance		—	5	—	kΩ
Power Specifications					
Power Supply Current (VDDA supplied to ADC0)	Operating Mode, 200 ksps	—	1100	1500	µA
Burst Mode (Idle)		—	1100	1500	µA
Power-On Time		5	—	—	µs
Power Supply Rejection Ratio		—	–60	—	mV/V

Notes:

1. Represents one standard deviation from the mean. Offset and full-scale error can be removed through calibration.
2. An additional 2 FCLK cycles are required to start and complete a conversion
3. Additional tracking time may be required depending on the output impedance connected to the ADC input. See Section “6.2.1. Settling Time Requirements” on page 59
4. An increase in tracking time will decrease the ADC throughput.
5. See Section “6.3. Selectable Gain” on page 60 for more information about the setting the gain.

13. Special Function Registers

The direct-access data memory locations from 0x80 to 0xFF constitute the special function registers (SFRs). The SFRs provide control and data exchange with the C8051F58x/F59x's resources and peripherals. The CIP-51 controller core duplicates the SFRs found in a typical 8051 implementation as well as implementing additional SFRs used to configure and access the sub-systems unique to the C8051F58x/F59x. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set. Table 13.3 lists the SFRs implemented in the C8051F58x/F59x device family.

The SFR registers are accessed anytime the direct addressing mode is used to access memory locations from 0x80 to 0xFF. SFRs with addresses ending in 0x0 or 0x8 (e.g., P0, TCON, SCON0, IE, etc.) are bit-addressable as well as byte-addressable. All other SFRs are byte-addressable only. Unoccupied addresses in the SFR space are reserved for future use. Accessing unoccupied addresses in the SFR space will have an indeterminate effect and should be avoided. Refer to the corresponding pages of the data sheet, as indicated in Table 13.3, for a detailed description of each register.

13.1. SFR Paging

The CIP-51 features SFR paging*,* allowing the device to map many SFRs into the 0x80 to 0xFF memory address space. The SFR memory space has 256 *pages*. In this way, each memory location from 0x80 to 0xFF can access up to 256 SFRs. The C8051F58x/F59x family of devices utilizes three SFR pages: 0x0, 0xC, and 0xF. SFR pages are selected using the Special Function Register Page Selection register, SFRPAGE (see SFR Definition 11.3). The procedure for reading and writing an SFR is as follows:

1. Select the appropriate SFR page number using the SFRPAGE register.
2. Use direct accessing mode to read or write the special function register (MOV instruction).

13.2. Interrupts and SFR Paging

When an interrupt occurs, the SFR Page Register will automatically switch to the SFR page containing the flag bit that caused the interrupt. The automatic SFR Page switch function conveniently removes the burden of switching SFR pages from the interrupt service routine. Upon execution of the RETI instruction, the SFR page is automatically restored to the SFR Page in use prior to the interrupt. This is accomplished via a three-byte *SFR Page Stack*. The top byte of the stack is SFRPAGE, the current SFR Page. The second byte of the SFR Page Stack is SFRNEXT. The third, or bottom byte of the SFR Page Stack is SFRLAST. Upon an interrupt, the current SFRPAGE value is pushed to the SFRNEXT byte, and the value of SFRNEXT is pushed to SFRLAST. Hardware then loads SFRPAGE with the SFR Page containing the flag bit associated with the interrupt. On a return from interrupt, the SFR Page Stack is popped resulting in the value of SFRNEXT returning to the SFRPAGE register, thereby restoring the SFR page context without software intervention. The value in SFRLAST (0x00 if there is no SFR Page value in the bottom of the stack) of the stack is placed in SFRNEXT register. If desired, the values stored in SFRNEXT and SFRLAST may be modified during an interrupt, enabling the CPU to return to a different SFR Page upon execution of the RETI instruction (on interrupt exit). Modifying registers in the SFR Page Stack does not cause a push or pop of the stack. Only interrupt calls and returns will cause push/pop operations on the SFR Page Stack.

On the C8051F58x/F59x devices, vectoring to an interrupt will switch SFRPAGE to page 0x00, except for the CAN0 interrupt which will switch SFRPAGE to page 0x0C, and the UART1, PCA1, Comparator2, and Timer4/5 interrupts will switch SFRPAGE to 0x10.

SFR Definition 13.3. SFRNEXT: SFR Next

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

SFR Address = 0x85; SFR Page = All Pages

Bit	Name	Function
7:0	SFRNEXT[7:0]	**SFR Page Bits.** This is the value that will go to the SFR Page register upon a return from interrupt. Write: Sets the SFR Page contained in the second byte of the SFR Stack. This will cause the SFRPAGE SFR to have this SFR page value upon a return from interrupt. Read: Returns the value of the SFR page contained in the second byte of the SFR stack. SFR page context is retained upon interrupts/return from interrupts in a 3 byte SFR Page Stack: SFRPAGE is the first entry, SFRNEXT is the second, and SFRLAST is the third entry. The SFR stack bytes may be used alter the context in the SFR Page Stack, and will not cause the stack to “push” or “pop”. Only interrupts and return from interrupts cause pushes and pops of the SFR Page Stack.

Table 13.1. Special Function Register (SFR) Memory Map for Pages 0x00, 0x10, and 0x0F

90	00 10 0F	P1 (All Pages)	TMR3CN TMR5CN	TMR3RLL TMR5CAPL	TMR3RLH TMR5CAPH	TMR3L TMR5L	TMR3H TMR5H	TMR5CF	CLKMUL
88	00 10 0F	TCON (All Pages)	TMOD (All Pages)	TL0 (All Pages)	TL1 (All Pages)	TH0 (All Pages)	TH1 (All Pages)	CKCON (All Pages)	PSCTL CLKSEL
80	00 10 0F	P0 (All Pages)	SP (All Pages)	DPL (All Pages)	DPH (All Pages)	SFR0CN	SFRNEXT (All Pages)	SFRLAST (All Pages)	PCON (All Pages)
		0(8) (bit addressable)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)

Table 13.3. Special Function Registers (Continued)

SFRs are listed in alphabetical order. All undefined SFR locations are reserved

Register	Address	Description	Page
PCA1CPL11	0xCE	PCA1 Capture 11 Low	350
PCA1CPM6	0xDA	PCA1 Module 6 Mode Register	348
PCA1CPM7	0xDB	PCA1 Module 7 Mode Register	348
PCA1CPM8	0xDC	PCA1 Module 8 Mode Register	348
PCA1CPM9	0xDD	PCA1 Module 9 Mode Register	348
PCA1CPM10	0xDE	PCA1 Module 10 Mode Register	348
PCA1CPM11	0xDF	PCA1 Module 11 Mode Register	348
PCA1H	0xFA	PCA1 Counter High	349
PCA1L	0xF9	PCA1 Counter Low	349
PCA1MD	0xD9	PCA1 Mode	346
PCA1PWM	0xDA	PCA1 PWM Configuration	347
PCON	0x87	Power Control	151
PSBANK	0xF5	Program Space Bank Select	104
PSCTL	0x8F	Program Store R/W Control	145
PSW	0xD0	Program Status Word	100
REF0CN	0xD1	Voltage Reference Control	76
REG0CN	0xD1	Voltage Regulator Control	90
RSTSRC	0xEF	Reset Source Configuration/Status	157
SBCON0	0xAB	UART0 Baud Rate Generator Control	263
SBRLH0	0xAD	UART0 Baud Rate Reload High Byte	264
SBRLL0	0xAC	UART0 Baud Rate Reload Low Byte	264
SBUF0	0x99	UART0 Data Buffer	263
SCON0	0x98	UART0 Control	261
SBUF1	0x99	UART1 Data Buffer	270
SCON1	0x98	UART1 Control	269
SFR0CN	0x84	SFR Page Control	113
SFRLAST	0x86	SFR Stack Last Page	116
SFRNEXT	0x85	SFR Stack Next Page	115
SFRPAGE	0xA7	SFR Page Select	114
SMB0CF	0xC1	SMBus0 Configuration	245
SMB0CN	0xC0	SMBus0 Control	247
SMB0DAT	0xC2	SMBus0 Data	249
SMOD0	0xA9	UART0 Mode	262
SN0	0xF9	Serial Number 0	101
SN1	0xFA	Serial Number 1	101
SN2	0xFB	Serial Number 2	101

SFR Definition 14.4. EIP1: Extended Interrupt Priority 1

Bit	7	6	5	4	3	2	1	0
Name	PLIN0	PT3	PCP1	PCP0	PPCA0	PADC0	PWADC0	PSMB0
Type	R/W	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 = 0xF6; SFR Page = All Pages

Bit	Name	Function
7	PLIN0	**LIN0 Interrupt Priority Control.** This bit sets the priority of the LIN0 interrupt. 0: LIN0 interrupts set to low priority level. 1: LIN0 interrupts set to high priority level.
6	PT3	**Timer 3 Interrupt Priority Control.** This bit sets the priority of the Timer 3 interrupt. 0: Timer 3 interrupts set to low priority level. 1: Timer 3 interrupts set to high priority level.
5	PCP1	**Comparator0 (CP1) Interrupt Priority Control.** This bit sets the priority of the CP1 interrupt. 0: CP1 interrupt set to low priority level. 1: CP1 interrupt set to high priority level.
4	PCP0	**Comparator0 (CP0) Interrupt Priority Control.** This bit sets the priority of the CP0 interrupt. 0: CP0 interrupt set to low priority level. 1: CP0 interrupt set to high priority level.
3	PPCA0	**Programmable Counter Array (PCA0) Interrupt Priority Control.** This bit sets the priority of the PCA0 interrupt. 0: PCA0 interrupt set to low priority level. 1: PCA0 interrupt set to high priority level.
2	PADC0	**ADC0 Conversion Complete Interrupt Priority Control.** This bit sets the priority of the ADC0 Conversion Complete interrupt. 0: ADC0 Conversion Complete interrupt set to low priority level. 1: ADC0 Conversion Complete interrupt set to high priority level.
1	PWADC0	**ADC0 Window Comparator Interrupt Priority Control.** This bit sets the priority of the ADC0 Window interrupt. 0: ADC0 Window interrupt set to low priority level. 1: ADC0 Window interrupt set to high priority level.
0	PSMB0	**SMBus (SMB0) Interrupt Priority Control.** This bit sets the priority of the SMB0 interrupt. 0: SMB0 interrupt set to low priority level. 1: SMB0 interrupt set to high priority level.

SFR Definition 15.4. CCH0CN: Cache Control

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	Reserved	CHPFEN	Reserved	Reserved	Reserved	Reserved	CHBLKW
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	0	0	0	0	0

SFR Address = 0xE3; SFR Page = 0x0F

Bit	**Name**	**Function**
7:6	Reserved	Must Write 00b
5	CHPFEN	**Cache Prefect Enable Bit.** 0: Prefetch engine is disabled. 1: Prefetch engine is enabled.
4:1	Reserved	Must Write 0000b.
0	CHBLKW	**Block Write Enable Bit.** This bit allows block writes to Flash memory from firmware. 0: Each byte of a software Flash write is written individually. 1: Flash bytes are written in groups of two.

SFR Definition 15.5. ONESHOT: Flash Oneshot Period

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

SFR Address = 0xBE; SFR Page = 0x0F

Bit	**Name**	**Function**
7:4	Unused	Read = 0000b. Write = don’t care.
3:0	PERIOD[3:0]	**Oneshot Period Control Bits.** These bits limit the internal Flash read strobe width as follows. When the Flash read strobe is de-asserted, the Flash memory enters a low-power state for the remainder of the system clock cycle. These bits have no effect when the system clocks is greater than 12.5 MHz and FLRT = 0. $FLASH_{RDMAX} = 5ns + (PERIOD \times 5ns)$

SFR Definition 16.1. PCON: Power Control

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

SFR Address = 0x87; SFR Page = All Pages

Bit	Name	Function
7:2	GF[5:0]	**General Purpose Flags 5–0.** These are general purpose flags for use under software control.
1	STOP	**Stop Mode Select.** Setting this bit will place the CIP-51 in Stop mode. This bit will always be read as 0. 1: CPU goes into Stop mode (internal oscillator stopped).
0	IDLE	**IDLE: Idle Mode Select.** Setting this bit will place the CIP-51 in Idle mode. This bit will always be read as 0. 1: CPU goes into Idle mode. (Shuts off clock to CPU, but clock to Timers, Interrupts, Serial Ports, and Analog Peripherals are still active.)

18.6.2.2. 8-bit MOVX without Bank Select: EMI0CF[4:2] = 001 or 011

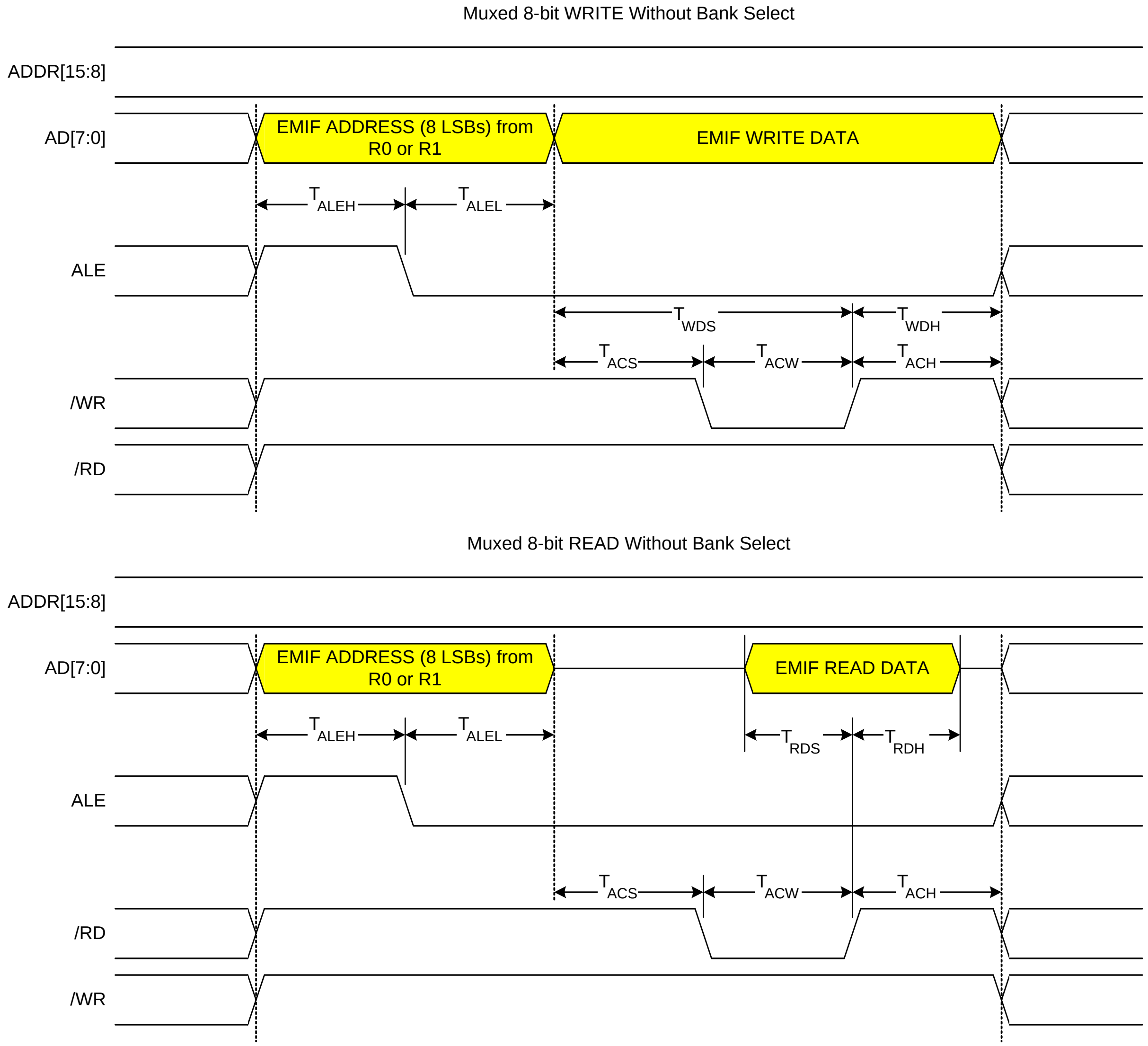


Figure 18.8. Multiplexed 8-bit MOVX without Bank Select Timing

SFR Definition 20.1. XBR0: Port I/O Crossbar Register 0

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CP1AE	CP1E	CP0AE	CP0E	SMB0E	SPI0E	CAN0E	URT0E
Type	R/W	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 = 0xE1; SFR Page = 0x0F

Bit	Name	Function
7	CP1AE	**Comparator1 Asynchronous Output Enable.** 0: Asynchronous CP1 unavailable at Port pin. 1: Asynchronous CP1 routed to Port pin.
6	CP1E	**Comparator1 Output Enable.** 0: CP1 unavailable at Port pin. 1: CP1 routed to Port pin.
5	CP0AE	**Comparator0 Asynchronous Output Enable.** 0: Asynchronous CP0 unavailable at Port pin. 1: Asynchronous CP0 routed to Port pin.
4	CP0E	**Comparator0 Output Enable.** 0: CP0 unavailable at Port pin. 1: CP0 routed to Port pin.
3	SMB0E	**SMBus I/O Enable.** 0: SMBus I/O unavailable at Port pins. 1: SMBus I/O routed to Port pins.
2	SPI0E	**SPI I/O Enable.** 0: SPI I/O unavailable at Port pins. 1: SPI I/O routed to Port pins. Note that the SPI can be assigned either 3 or 4 GPIO pins.
1	CAN0E	**CAN I/O Output Enable.** 0: CAN I/O unavailable at Port pins. 1: CAN_TX, CAN_RX routed to Port pins P0.6 and P0.7.
0	URT0E	**UART0 I/O Output Enable.** 0: UART0 I/O unavailable at Port pin. 1: UART0 TX0, RX0 routed to Port pins P0.4 and P0.5.

SFR Definition 20.28. P3SKIP: Port 3Skip

Bit	7	6	5	4	3	2	1	0
Name	P3SKIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD7; SFR Page = 0x0F

Bit	Name	Function
7:0	P3SKIP[7:0]	**Port 3 Crossbar Skip Enable Bits.** These bits select Port 3 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar. 0: Corresponding P3.n pin is not skipped by the Crossbar. 1: Corresponding P3.n pin is skipped by the Crossbar.
Note: Port P3.1–P3.7 are only available on the 48-pin and 40-pin packages.		

SFR Definition 20.29. P4: Port 4

Bit	7	6	5	4	3	2	1	0
Name	P4[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xB5; SFR Page = All Pages

Bit	Name	Description	Write	Read
7:0	P4[7:0]	**Port 4 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P4.n Port pin is logic LOW. 1: P4.n Port pin is logic HIGH.
Note: Port 4.0 is only available on the 48-pin and 40-pin packages.; P4.1-P4.7 is only available on the 48-pin packages.				

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.

23.5.2. Read Sequence (Master)

During a read sequence, an SMBus master reads data from a slave device. The master in this transfer will be a transmitter during the address byte, and a receiver during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 1 (READ). Serial data is then received from the slave on SDA while the SMBus outputs the serial clock. The slave transmits one or more bytes of serial data. An interrupt is generated after each received byte.

Software must write the ACK bit at that time to ACK or NACK the received byte. Writing a 1 to the ACK bit generates an ACK; writing a 0 generates a NACK. Software should write a 0 to the ACK bit for the last data transfer, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. The interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 23.6 shows a typical master read sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the 'data byte transferred' interrupts occur **before** the ACK cycle in this mode.

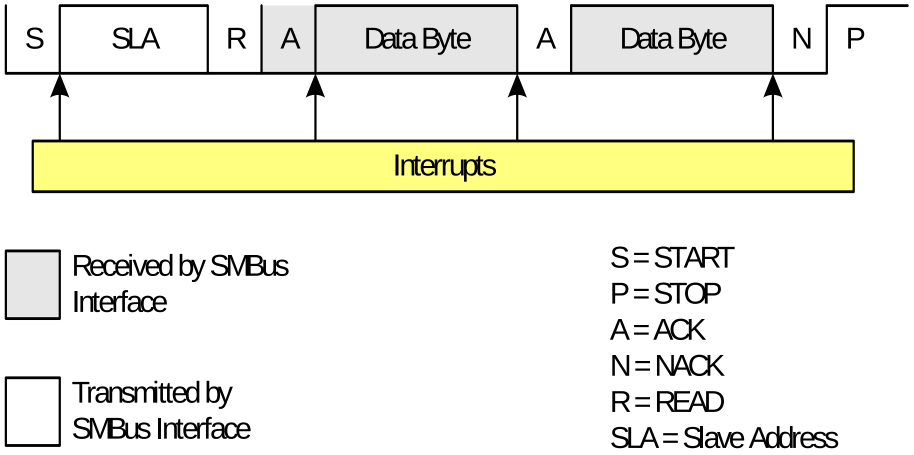


Figure 23.6. Typical Master Read Sequence

23.5.3. Write Sequence (Slave)

During a write sequence, an SMBus master writes data to a slave device. The slave in this transfer will be a receiver during the address byte, and a receiver during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode when a START followed by a slave address and direction bit (WRITE in this case) is received. Upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK.

If the received slave address is ignored, slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are received. Software must write the ACK bit at that time to ACK or NACK the received byte.

The interface exits Slave Receiver Mode after receiving a STOP. Note that the interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver. Figure 23.7 shows a typical slave write sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the 'data byte transferred' interrupts occur **before** the ACK in this mode.

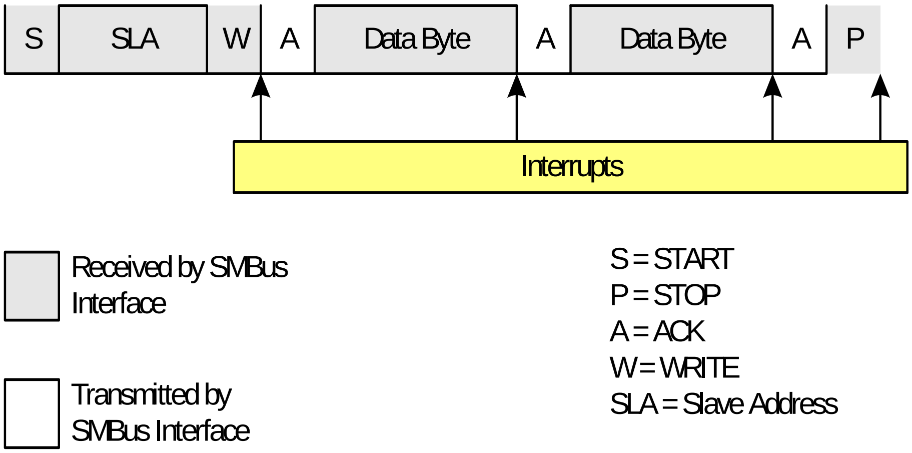


Figure 23.7. Typical Slave Write Sequence

25.2. Operational Modes

UART1 provides standard asynchronous, full duplex communication. The UART mode (8-bit or 9-bit) is selected by the S1MODE bit (SCON1.7). Typical UART connection options are shown in Figure 25.3.

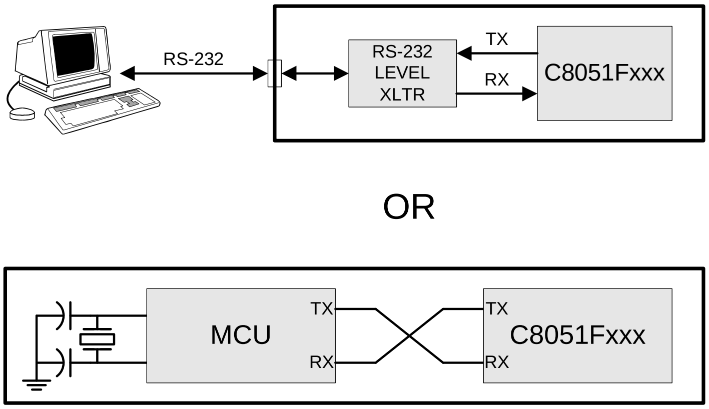


Figure 25.3. UART Interconnect Diagram

25.2.1. 8-Bit UART

8-Bit UART mode uses a total of 10 bits per data byte: one start bit, eight data bits (LSB first), and one stop bit. Data are transmitted LSB first from the TX1 pin and received at the RX1 pin. On receive, the eight data bits are stored in SBUF1 and the stop bit goes into RB81 (SCON1.2).

Data transmission begins when software writes a data byte to the SBUF1 register. The TI1 Transmit Interrupt Flag (SCON1.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN1 Receive Enable bit (SCON1.4) is set to logic 1. After the stop bit is received, the data byte will be loaded into the SBUF1 receive register if the following conditions are met: RI0 must be logic 0, and if MCE0 is logic 1, the stop bit must be logic 1. In the event of a receive data overrun, the first received 8 bits are latched into the SBUF1 receive register and the following overrun data bits are lost.

If these conditions are met, the eight bits of data is stored in SBUF1, the stop bit is stored in RB81 and the RI1 flag is set. If these conditions are not met, SBUF1 and RB81 will not be loaded and the RI1 flag will not be set. An interrupt will occur if enabled when either TI1 or RI1 is set.

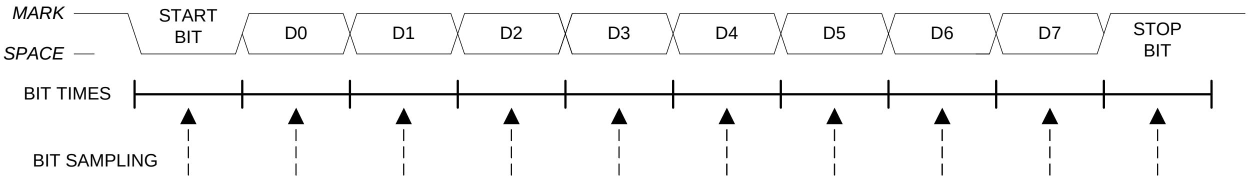


Figure 25.4. 8-Bit UART Timing Diagram

25.2.2. 9-Bit UART

9-bit UART mode uses a total of eleven bits per data byte: a start bit, 8 data bits (LSB first), a programmable ninth data bit, and a stop bit. The state of the ninth transmit data bit is determined by the value in TB81 (SCON1.3), which is assigned by user software. It can be assigned the value of the parity flag (bit P in register PSW) for error detection, or used in multiprocessor communications. On receive, the ninth data bit goes into RB81 (SCON1.2) and the stop bit is ignored.

26.3. SPI0 Slave Mode Operation

When SPI0 is enabled and not configured as a master, it will operate as a SPI slave. As a slave, bytes are shifted in through the MOSI pin and out through the MISO pin by a master device controlling the SCK signal. A bit counter in the SPI0 logic counts SCK edges. When 8 bits have been shifted through the shift register, the SPIF flag is set to logic 1, and the byte is copied into the receive buffer. Data is read from the receive buffer by reading SPI0DAT. A slave device cannot initiate transfers. Data to be transferred to the master device is pre-loaded into the shift register by writing to SPI0DAT. Writes to SPI0DAT are double-buffered, and are placed in the transmit buffer first. If the shift register is empty, the contents of the transmit buffer will immediately be transferred into the shift register. When the shift register already contains data, the SPI will load the shift register with the transmit buffer's contents after the last SCK edge of the next (or current) SPI transfer.

When configured as a slave, SPI0 can be configured for 4-wire or 3-wire operation. The default, 4-wire slave mode, is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In 4-wire mode, the NSS signal is routed to a port pin and configured as a digital input. SPI0 is enabled when NSS is logic 0, and disabled when NSS is logic 1. The bit counter is reset on a falling edge of NSS. Note that the NSS signal must be driven low at least 2 system clocks before the first active edge of SCK for each byte transfer. Figure 26.4 shows a connection diagram between two slave devices in 4-wire slave mode and a master device.

3-wire slave mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. NSS is not used in this mode, and is not mapped to an external port pin through the crossbar. Since there is no way of uniquely addressing the device in 3-wire slave mode, SPI0 must be the only slave device present on the bus. It is important to note that in 3-wire slave mode there is no external means of resetting the bit counter that determines when a full byte has been received. The bit counter can only be reset by disabling and re-enabling SPI0 with the SPIEN bit. Figure 26.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.

26.4. SPI0 Interrupt Sources

When SPI0 interrupts are enabled, the following four flags will generate an interrupt when they are set to logic 1:

All of the following bits must be cleared by software.

1. The SPI Interrupt Flag, SPIF (SPI0CN.7) is set to logic 1 at the end of each byte transfer. This flag can occur in all SPI0 modes.
2. The Write Collision Flag, WCOL (SPI0CN.6) is set to logic 1 if a write to SPI0DAT is attempted when the transmit buffer has not been emptied to the SPI shift register. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written.This flag can occur in all SPI0 modes.
3. The Mode Fault Flag MODF (SPI0CN.5) is set to logic 1 when SPI0 is configured as a master, and for multi-master mode and the NSS pin is pulled low. When a Mode Fault occurs, the MSTEN and SPIEN bits in SPI0CN are set to logic 0 to disable SPI0 and allow another master device to access the bus.
4. The Receive Overrun Flag RXOVRN (SPI0CN.4) is set to logic 1 when configured as a slave, and a transfer is completed and the receive buffer still holds an unread byte from a previous transfer. The new byte is not transferred to the receive buffer, allowing the previously received data byte to be read. The data byte which caused the overrun is lost.

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)

SFR Definition 27.4. TL0: Timer 0 Low Byte

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	TL0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8A; SFR Page = All Pages

Bit	**Name**	**Function**
7:0	TL0[7:0]	**Timer 0 Low Byte.** The TL0 register is the low byte of the 16-bit Timer 0.

SFR Definition 27.5. TL1: Timer 1 Low Byte

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	TL1[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8B; SFR Page = All Pages

Bit	**Name**	**Function**
7:0	TL1[7:0]	**Timer 1 Low Byte.** The TL1 register is the low byte of the 16-bit Timer 1.

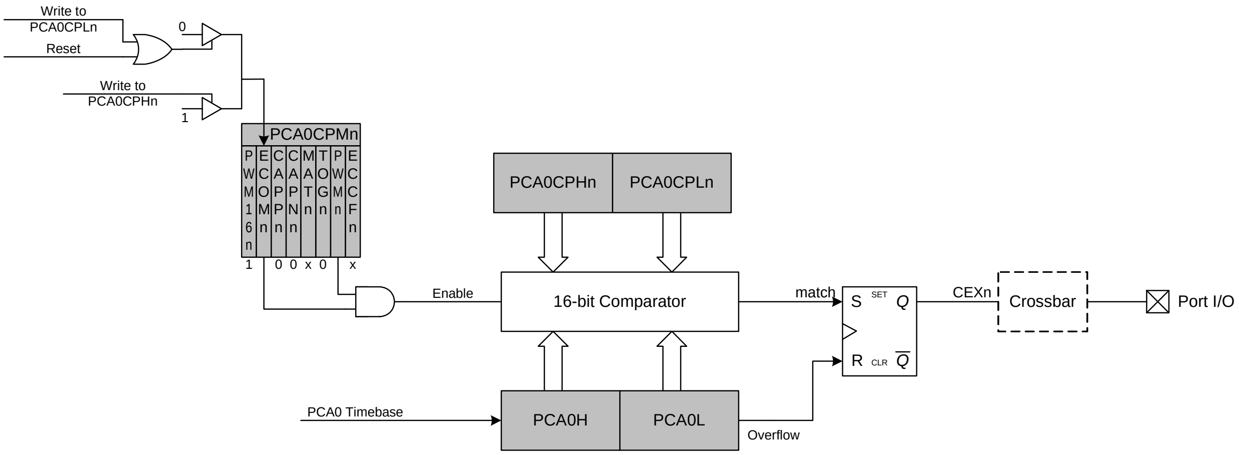


Figure 28.10. PCA0 16-Bit PWM Mode

28.4. Watchdog Timer Mode

A programmable watchdog timer (WDT) function is available through the PCA0 Module 5. The WDT is used to generate a reset if the time between writes to the WDT update register (PCA0CPH2) exceed a specified limit. The WDT can be configured and enabled/disabled as needed by software.

With the WDTE bit set in the PCA0MD register, Module 5 operates as a watchdog timer (WDT). The Module 2 high byte is compared to the PCA0 counter high byte; the Module 2 low byte holds the offset to be used when WDT updates are performed. **The Watchdog Timer is enabled on reset. Writes to some PCA0 registers are restricted while the Watchdog Timer is enabled.** The WDT will generate a reset shortly after code begins execution. To avoid this reset, the WDT should be explicitly disabled (and optionally re-configured and re-enabled if it is used in the system).

28.4.1. Watchdog Timer Operation

While the WDT is enabled:

- PCA0 counter is forced on.
- Writes to PCA0L and PCA0H are not allowed.
- PCA0 clock source bits (CPS2–CPS0) are frozen.
- PCA0 Idle control bit (CIDL) is frozen.
- PCA0 Module 5 is forced into software timer mode.
- Writes to the Module 5 mode register (PCA0CPM5) are disabled.

While the WDT is enabled, writes to the CR bit will not change the PCA0 counter state; the counter will run until the WDT is disabled. The PCA0 counter run control bit (CR) will read zero if the WDT is enabled but user software has not enabled the PCA0 counter. If a match occurs between PCA0CPH5 and PCA0H while the WDT is enabled, a reset will be generated. To prevent a WDT reset, the WDT may be updated with a write of any value to PCA0CPH5. Upon a PCA0CPH5 write, PCA0H plus the offset held in PCA0CPL5 is loaded into PCA0CPH5 (See Figure 28.11).