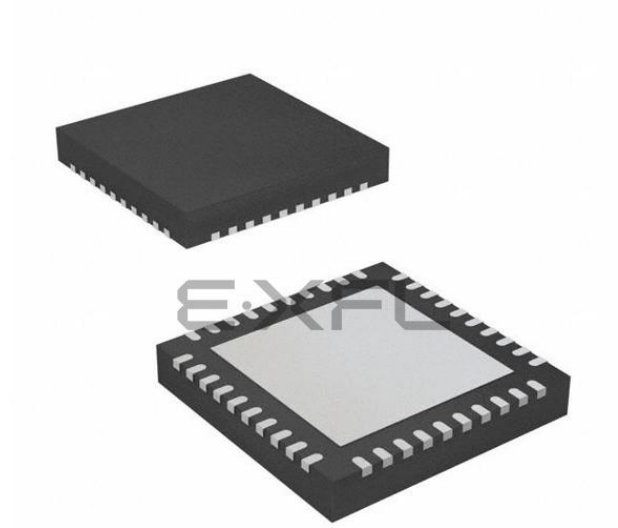

Welcome to E-XFL.COM

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	EBI/EMI, SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	33
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 32x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	40-VFQFN Exposed Pad
Supplier Device Package	40-QFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f591-im

List of Figures

List of Registers

Table 3.1. Pin Definitions for the C8051F58x/F59x (Continued)

Name	Pin F580/1/4/5 (48-pin)	Pin F588/9-F590/1 (40-pin)	Pin F582/3/6/7 (32-pin)	Type	Description
P0.6	44	36	28	D I/O or A In	Port 0.6
P0.7	43	35	27	D I/O or A In	Port 0.7
P1.0	42	34	26	D I/O or A In	Port 1.0. See SFR Definition 20.17 for a description.
P1.1	41	33	25	D I/O or A In	Port 1.1.
P1.2	40	32	24	D I/O or A In	Port 1.2.
P1.3	39	31	23	D I/O or A In	Port 1.3.
P1.4	38	30	22	D I/O or A In	Port 1.4.
P1.5	37	29	21	D I/O or A In	Port 1.5.
P1.6	36	28	20	D I/O or A In	Port 1.6.
P1.7	35	27	19	D I/O or A In	Port 1.7.
P2.0	34	26	18	D I/O or A In	Port 2.0. See SFR Definition 20.21 for a description.
P2.1	33	25	17	D I/O or A In	Port 2.1.
P2.2	32	24	16	D I/O or A In	Port 2.2.
P2.3	31	23	15	D I/O or A In	Port 2.3.
P2.4	30	22	14	D I/O or A In	Port 2.4.
P2.5	29	21	13	D I/O or A In	Port 2.5.
P2.6	28	20	12	D I/O or A In	Port 2.6.
P2.7	27	19	11	D I/O or A In	Port 2.7.
P3.0	26	18	—	D I/O or A In	Port 3.0. See SFR Definition 20.25 for a description.
P3.1	25	17	—	D I/O or A In	Port 3.1.
P3.2	24	16	—	D I/O or A In	Port 3.2.
P3.3	23	15	—	D I/O or A In	Port 3.3.
P3.4	22	14	—	D I/O or A In	Port 3.4.
P3.5	21	13	—	D I/O or A In	Port 3.5.
P3.6	20	12	—	D I/O or A In	Port 3.6.

Table 5.11. Temperature Sensor Electrical Characteristics

VDDA = 1.8 to 2.75 V, –40 to +125 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Linearity		—	± 0.1	—	°C
Slope		—	3.33	—	mV/°C
Slope Error*		—	100	—	µV/°C
Offset	Temp = 0 °C	—	856	—	mV
Offset Error*	Temp = 0 °C	—	12	—	mV
Power Supply Current		—	22	—	µA
Tracking Time		12	—	—	µs
***Note:** Represents one standard deviation from the mean.					

Table 5.12. Voltage Reference Electrical Characteristics

VDDA = 1.8 to 2.75 V, –40 to +125 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Internal Reference (REFBE = 1)					
Output Voltage	25 °C ambient (REFLV = 0)	1.45	1.50	1.55	V
	25 °C ambient (REFLV = 1), V_{DD} = 2.6 V	2.15	2.20	2.25	
VREF Short-Circuit Current		—	5	10	mA
VREF Temperature Coefficient		—	22	—	ppm/°C
Power Consumption	Internal	—	30	50	µA
Load Regulation	Load = 0 to 200 µA to AGND	—	3	—	µV/µA
VREF Turn-on Time 1	4.7 µF and 0.1 µF bypass	—	1.5	—	ms
VREF Turn-on Time 2	0.1 µF bypass	—	46	—	µs
Power Supply Rejection		—	1.2	—	mV/V
External Reference (REFBE = 0)					
Input Voltage Range		1.5	—	V_{DDA}	V
Input Current	Sample Rate = 200 ksps; VREF = 1.5 V	—	2.5	—	µA
Power Specifications					
Reference Bias Generator	REFBE = 1 or TEMPE = 1	—	21	40	µA

6. 12-Bit ADC (ADC0)

The ADC0 on the C8051F58x/F59x consists of an analog multiplexer (AMUX0) with 35/28 total input selections and a 200 ksps, 12-bit successive-approximation-register (SAR) ADC with integrated track-and-hold, programmable window detector, programmable attenuation (1:2), and hardware accumulator. The ADC0 subsystem has a special Burst Mode which can automatically enable ADC0, capture and accumulate samples, then place ADC0 in a low power shutdown mode without CPU intervention. The AMUX0, data conversion modes, and window detector are all configurable under software control via the Special Function Registers shows in Figure 6.1. ADC0 inputs are single-ended and may be configured to measure P0.0-P3.7, the Temperature Sensor output, V_{DD}, or GND with respect to GND. The voltage reference for ADC0 is selected as described in Section "7. Temperature Sensor" on page 74. ADC0 is enabled when the AD0EN bit in the ADC0 Control register (ADC0CN) is set to logic 1, or when performing conversions in Burst Mode. ADC0 is in low power shutdown when AD0EN is logic 0 and no Burst Mode conversions are taking place.

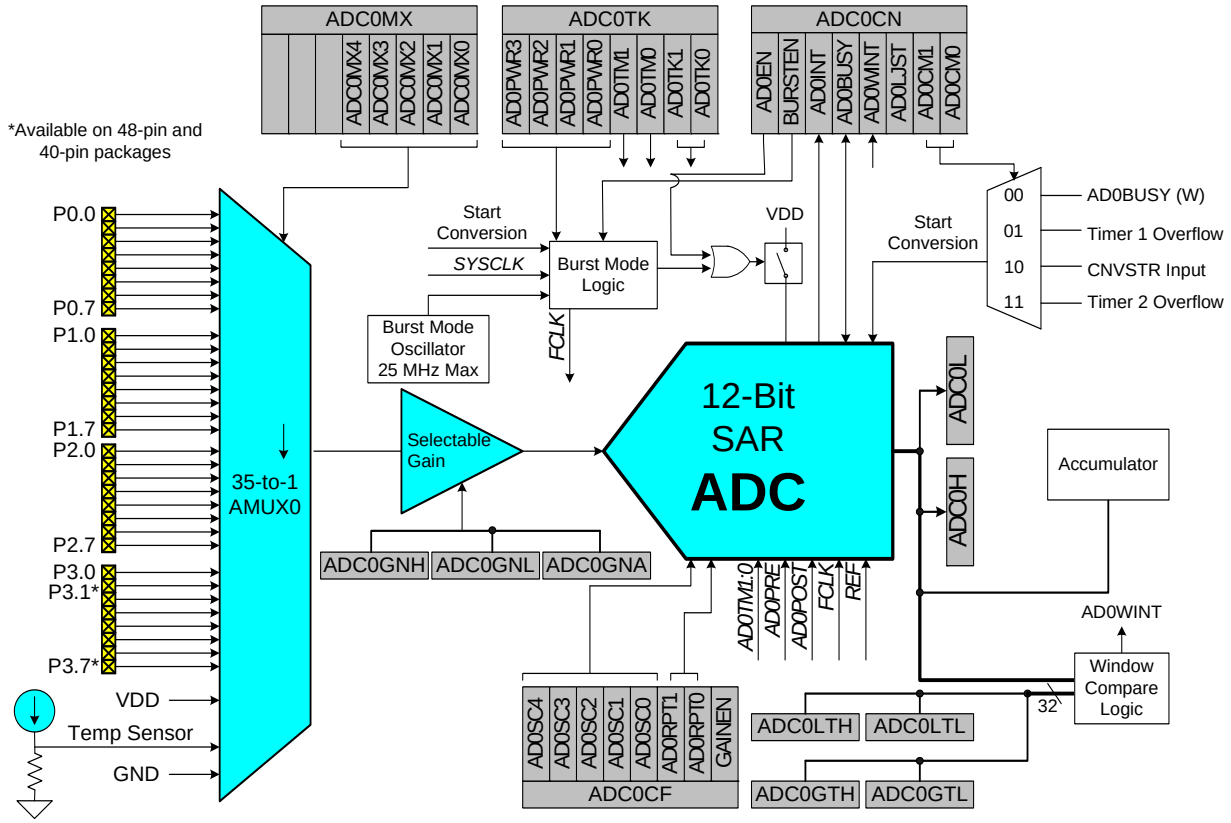

Figure 6.1. ADC0 Functional Block Diagram

Table 11.1. CIP-51 Instruction Set Summary (Prefetch-Enabled) (Continued)

Mnemonic	Description	Bytes	Clock Cycles
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
ANL C, /bit	AND complement of direct bit to Carry	2	2
ORL C, bit	OR direct bit to carry	2	2
ORL C, /bit	OR complement of direct bit to Carry	2	2
MOV C, bit	Move direct bit to Carry	2	2
MOV bit, C	Move Carry to direct bit	2	2
JC rel	Jump if Carry is set	2	2/(4-6)*
JNC rel	Jump if Carry is not set	2	2/(4-6)*
JB bit, rel	Jump if direct bit is set	3	3/(5-7)*
JNB bit, rel	Jump if direct bit is not set	3	3/(5-7)*
JBC bit, rel	Jump if direct bit is set and clear bit	3	3/(5-7)*
Program Branching			
ACALL addr11	Absolute subroutine call	2	4-6*
LCALL addr16	Long subroutine call	3	5-7*
RET	Return from subroutine	1	6-8*
RETI	Return from interrupt	1	6-8*
AJMP addr11	Absolute jump	2	4-6*
LJMP addr16	Long jump	3	5-7*
SJMP rel	Short jump (relative address)	2	4-6*
JMP @A+DPTR	Jump indirect relative to DPTR	1	3-5*
JZ rel	Jump if A equals zero	2	2/(4-6)*
JNZ rel	Jump if A does not equal zero	2	2/(4-6)*
CJNE A, direct, rel	Compare direct byte to A and jump if not equal	3	4/(6-8)*
CJNE A, #data, rel	Compare immediate to A and jump if not equal	3	3/(6-8)*
CJNE Rn, #data, rel	Compare immediate to Register and jump if not equal	3	3/(5-7)*
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal	3	4/(6-8)*
DJNZ Rn, rel	Decrement Register and jump if not zero	2	2/(4-6)*
DJNZ direct, rel	Decrement direct byte and jump if not zero	3	3/(5-7)*
NOP	No operation	1	1

Note: Certain instructions take a variable number of clock cycles to execute depending on instruction alignment and the FLRT setting (SFR Definition 15.3).

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.

Table 14.1. Interrupt Summary

Interrupt Source	Interrupt Vector	Priority Order	Pending Flag	Bit addressable?	Cleared by HW?	Enable Flag	Priority Control
UART1	0x0093	18	RI1 (SCON1.0) TI1 (SCON1.1)	Y	N	ES1 (EIE2.3)	PS1 (EIP2.3)
Programmable Counter Array 1	0x009B	19	CF (PCA1CN.n) CCFn (PCA1CN.n)	Y	N	EPCA1 (EIE2.4)	PPCA1 (EIP2.4)
Comparator2	0x00A3	20	CP2FIF (CPT2CN.4) CP2RIF (CPT2CN.5)	N	N	ECP2 (EIE2.5)	PCP2 (EIP2.5)
Timer 4 Overflow	0x00AB	21	TF4H (TMR4CN.7) TR4L (TMR4CN.6)	N	N	ET4 (EIE2.6)	PT4 (EIP2.6)
Timer 5 Overflow	0x00B3	22	TF5H (TMR5CN.7) TF5L (TMR5CN.6)	N	N	ET5 (EIE2.7)	PT5 (EIP2.7)
***Note:** The LIN0INT bit is cleared by setting RSTINT (LINCTRL.3)							

14.2. Interrupt Register Descriptions

The SFRs used to enable the interrupt sources and set their priority level are described in this section. Refer to the data sheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

SFR Definition 17.2. RSTSRC: Reset Source

Bit	7	6	5	4	3	2	1	0
Name		FERROR	C0RSEF	SWRSF	WDTRSF	MCDRSF	PORSF	PINRSF
Type	R	R	R/W	R/W	R	R/W	R/W	R
Reset	0	Varies	Varies	Varies	Varies	Varies	Varies	Varies

SFR Address = 0xEF; SFR Page = 0x00

Bit	Name	Description	Write	Read
7	Unused	**Unused.**	Don't care.	0
6	FERROR	**Flash Error Reset Flag.**	N/A	Set to 1 if Flash read/write/erase error caused the last reset.
5	C0RSEF	**Comparator0 Reset Enable and Flag.**	Writing a 1 enables Comparator0 as a reset source (active-low).	Set to 1 if Comparator0 caused the last reset.
4	SWRSF	**Software Reset Force and Flag.**	Writing a 1 forces a system reset.	Set to 1 if last reset was caused by a write to SWRSF.
3	WDTRSF	**Watchdog Timer Reset Flag.**	N/A	Set to 1 if Watchdog Timer overflow caused the last reset.
2	MCDRSF	**Missing Clock Detector Enable and Flag.**	Writing a 1 enables the Missing Clock Detector. The MCD triggers a reset if a missing clock condition is detected.	Set to 1 if Missing Clock Detector timeout caused the last reset.
1	PORSF	**Power-On/V_{DD} Monitor Reset Flag, and V_{DD} monitor Reset Enable.**	Writing a 1 enables the V_{DD} monitor as a reset source. **Writing 1 to this bit before the V_{DD} monitor is enabled and stabilized may cause a system reset.**	Set to 1 anytime a power-on or V_{DD} monitor reset occurs. **When set to 1 all other RSTSRC flags are indeterminate.**
0	PINRSF	**HW Pin Reset Flag.**	N/A	Set to 1 if $\overline{\text{RST}}$ pin caused the last reset.

Note: Do not use read-modify-write operations on this register

18. External Data Memory Interface and On-Chip XRAM

For C8051F58x/F59x devices, 8 kB of RAM are included on-chip and mapped into the external data memory space (XRAM). Additionally, an External Memory Interface (EMIF) is available on the C8051F580/1/4/5 and C8051F588/9-F590/1 devices, which can be used to access off-chip data memories and memory-mapped devices connected to the GPIO ports. The external memory space may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using the MOVX indirect addressing mode using R0 or R1. If the MOVX instruction is used with an 8-bit address operand (such as @R1), then the high byte of the 16-bit address is provided by the External Memory Interface Control Register (EMI0CN, shown in SFR Definition 18.1).

Note: The MOVX instruction can also be used for writing to the Flash memory. See Section "15. Flash Memory" on page 138 for details. The MOVX instruction accesses XRAM by default.

18.1. Accessing XRAM

The XRAM memory space is accessed using the MOVX instruction. The MOVX instruction has two forms, both of which use an indirect addressing method. The first method uses the Data Pointer, DPTR, a 16-bit register which contains the effective address of the XRAM location to be read from or written to. The second method uses R0 or R1 in combination with the EMI0CN register to generate the effective XRAM address. Examples of both of these methods are given below.

18.1.1. 16-Bit MOVX Example

The 16-bit form of the MOVX instruction accesses the memory location pointed to by the contents of the DPTR register. The following series of instructions reads the value of the byte at address 0x1234 into the accumulator A:

```
MOV    DPTR, #1234h        ; load DPTR with 16-bit address to read (0x1234)
MOVX   A, @DPTR            ; load contents of 0x1234 into accumulator A
```

The above example uses the 16-bit immediate MOV instruction to set the contents of DPTR. Alternately, the DPTR can be accessed through the SFR registers DPH, which contains the upper 8-bits of DPTR, and DPL, which contains the lower 8-bits of DPTR.

18.1.2. 8-Bit MOVX Example

The 8-bit form of the MOVX instruction uses the contents of the EMI0CN SFR to determine the upper 8-bits of the effective address to be accessed and the contents of R0 or R1 to determine the lower 8-bits of the effective address to be accessed. The following series of instructions read the contents of the byte at address 0x1234 into the accumulator A.

```
MOV    EMI0CN, #12h        ; load high byte of address into EMI0CN
MOV    R0, #34h            ; load low byte of address into R0 (or R1)
MOVX   a, @R0              ; load contents of 0x1234 into accumulator A
```

SFR Definition 19.2. OSCICN: Internal Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	IOSCEN[1:0]		SUSPEND	IFRDY	Reserved	IFCN[2:0]		
Type	R/W	R/W	R/W	R	R	R/W		
Reset	1	1	0	1	0	0	0	0

SFR Address = 0xA1; SFR Page = 0x0F;

Bit	Name	Function
7:6	IOSCEN[1:0]	**Internal Oscillator Enable Bits.** 00: Oscillator Disabled. 01: Reserved. 10: Reserved. 11: Oscillator enabled in normal mode and disabled in suspend mode.
5	SUSPEND	**Internal Oscillator Suspend Enable Bit.** Setting this bit to logic 1 places the internal oscillator in SUSPEND mode. The internal oscillator resumes operation when one of the SUSPEND mode awakening events occurs. Before entering suspend mode, firmware must set the ZTCEN bit in REF0CN.
4	IFRDY	**Internal Oscillator Frequency Ready Flag.** Note: This flag may not accurately reflect the state of the oscillator. Firmware should not use this flag to determine if the oscillator is running. 0: Internal oscillator is not running at programmed frequency. 1: Internal oscillator is running at programmed frequency.
3	Reserved	Read = 0b; Must Write = 0b.
2:0	IFCN[2:0]	**Internal Oscillator Frequency Divider Control Bits.** 000: SYSCLK derived from Internal Oscillator divided by 128. 001: SYSCLK derived from Internal Oscillator divided by 64. 010: SYSCLK derived from Internal Oscillator divided by 32. 011: SYSCLK derived from Internal Oscillator divided by 16. 100: SYSCLK derived from Internal Oscillator divided by 8. 101: SYSCLK derived from Internal Oscillator divided by 4. 110: SYSCLK derived from Internal Oscillator divided by 2. 111: SYSCLK derived from Internal Oscillator divided by 1.

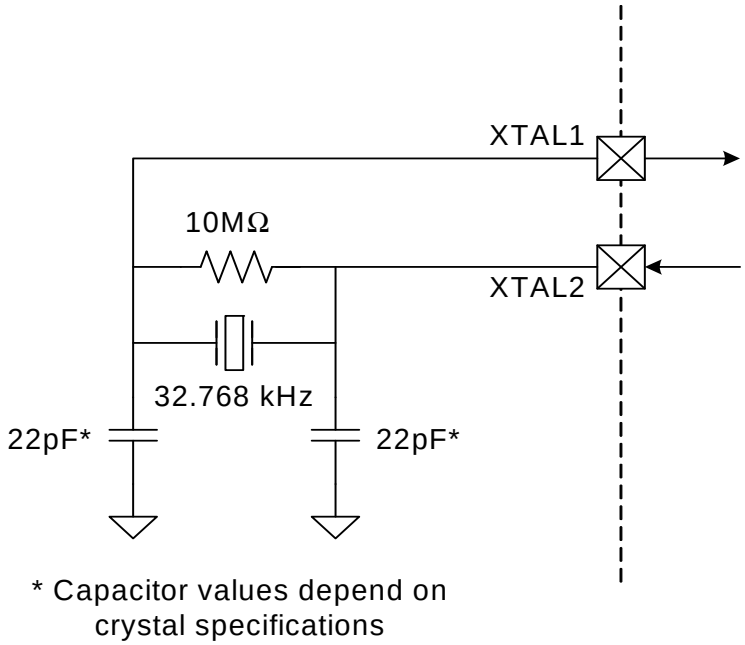

Figure 19.3. External 32.768 kHz Quartz Crystal Oscillator Connection Diagram

19.4.2. External RC Example

If an RC network is used as an external oscillator source for the MCU, the circuit should be configured as shown in Figure 19.1, Option 2. The capacitor should be no greater than 100 pF; however for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, first select the RC network value to produce the desired frequency of oscillation, according to Equation , where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and R = the pull-up resistor value in kΩ.

$$f = 1.23 \times 10^3 / (R \times C)$$

Equation 19.1. RC Mode Oscillator Frequency

For example: If the frequency desired is 100 kHz, let R = 246 kΩ and C = 50 pF:

f = 1.23(10^3)/RC = 1.23(10^3)/[246 x 50] = 0.1 MHz = 100 kHz

Referring to the table in SFR Definition 19.6, the required XFCN setting is 010b.

19.4.3. External Capacitor Example

If a capacitor is used as an external oscillator for the MCU, the circuit should be configured as shown in Figure 19.1, Option 3. The capacitor should be no greater than 100 pF; however for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, select the capacitor to be used and find the frequency of oscillation according to Equation , where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and V_{DD} = the MCU power supply in volts.

20.6. Special Function Registers for Accessing and Configuring Port I/O

All Port I/O are accessed through corresponding special function registers (SFRs) that are both byte addressable and bit addressable, except for P4 which is only byte addressable. When writing to a Port, the value written to the SFR is latched to maintain the output data value at each pin. When reading, the logic levels of the Port's input pins are returned regardless of the XBRn settings (i.e., even when the pin is assigned to another signal by the Crossbar, the Port register can always read its corresponding Port I/O pin). The exception to this is the execution of the read-modify-write instructions that target a Port Latch register as the destination. The read-modify-write instructions when operating on a Port SFR are the following: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ and MOV, CLR or SETB, when the destination is an individual bit in a Port SFR. For these instructions, the value of the latch register (not the pin) is read, modified, and written back to the SFR.

Ports 0–3 have a corresponding PnSKIP register which allows its individual Port pins to be assigned to digital functions or skipped by the Crossbar. All Port pins used for analog functions, GPIO, or dedicated digital functions such as the EMIF should have their PnSKIP bit set to 1.

The Port input mode of the I/O pins is defined using the Port Input Mode registers (PnMDIN). Each Port cell can be configured for analog or digital I/O. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is P4, which can only be used for digital I/O.

The output driver characteristics of the I/O pins are defined using the Port Output Mode registers (PnMDOUT). Each Port Output driver can be configured as either open drain or push-pull. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is the SMBus (SDA, SCL) pins, which are configured as open-drain regardless of the PnMDOUT settings.

SFR Definition 20.13. P0: Port 0

Bit	7	6	5	4	3	2	1	0
Name	P0[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0x80; SFR Page = All Pages; Bit-Addressable

Bit	Name	Description	Write	Read
7:0	P0[7:0]	**Port 0 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: P0.n Port pin is logic LOW. 1: P0.n Port pin is logic HIGH.

21.1. Software Interface with the LIN Controller

The selection of the mode (Master or Slave) and the automatic baud rate feature are done though the LIN0 Control Mode (LIN0CF) register. The other LIN registers are accessed indirectly through the two SFRs LIN0 Address (LIN0ADR) and LIN0 Data (LIN0DAT). The LIN0ADR register selects which LIN register is targeted by reads/writes of the LIN0DAT register. The full list of indirectly-accessible LIN registers is given in Table 21.4 on page 223.

21.2. LIN Interface Setup and Operation

The hardware based LIN controller allows for the implementation of both Master and Slave nodes with minimal firmware overhead and complete control of the interface status while allowing for interrupt and polled mode operation.

The first step to use the controller is to define the basic characteristics of the node:

Mode—Master or Slave

Baud Rate—Either defined manually or using the autobaud feature (slave mode only)

Checksum Type—Select between classic or enhanced checksum, both of which are implemented in hardware.

21.2.1. Mode Definition

Following the LIN specification, the controller implements in hardware both the Slave and Master operating modes. The mode is configured using the MODE bit (LIN0CF.6).

21.2.2. Baud Rate Options: Manual or Autobaud

The LIN controller can be selected to have its baud rate calculated manually or automatically. A master node must always have its baud rate set manually, but slave nodes can choose between a manual or automatic setup. The configuration is selected using the ABAUD bit (LIN0CF.5).

Both the manual and automatic baud rate configurations require additional setup. The following sections explain the different options available and their relation with the baud rate, along with the steps necessary to achieve the required baud rate.

21.2.3. Baud Rate Calculations: Manual Mode

The baud rate used by the LIN controller is a function of the System Clock (SYSCLK) and the LIN timing registers according to the following equation:

$$baud\_rate = \frac{SYSCLK}{2^{(prescaler+1)} \times divider \times (multiplier+1)}$$

The prescaler, divider and multiplier factors are part of the LIN0DIV and LIN0MUL registers and can assume values in the following range:

Table 21.1. Baud Rate Calculation Variable Ranges

Factor	Range
prescaler	0…3
multiplier	0…31
divider	200…511

Important Note: The minimum system clock (SYSCLK) to operate the LIN controller is 8 MHz.

Use the following equations to calculate the values for the variables for the baud-rate equation:

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.

LIN Register Definition 21.4. LIN0DTn: LIN0 Data Byte n

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

Indirect Address: LIN0DT1 = 0x00, LIN0DT2 = 0x01, LIN0DT3 = 0x02, LIN0DT4 = 0x03, LIN0DT5 = 0x04, LIN0DT6 = 0x05, LIN0DT7 = 0x06, LIN0DT8 = 0x07

Bit	Name	Function
7:0	DATAn[7:0]	**LIN Data Byte n.** Serial Data Byte that is received or transmitted across the LIN interface.

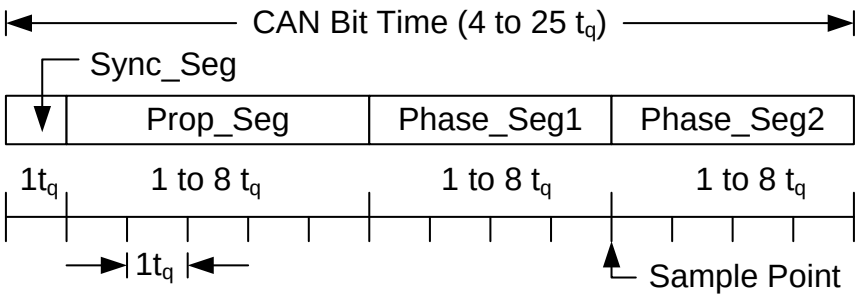

Figure 22.3. Four segments of a CAN Bit

The length of the 4 bit segments must be adjusted so that their sum is as close as possible to the desired bit time. Since each segment must be an integer multiple of the time quantum (t_q), the closest achievable bit time is 24 tq (1000.008 ns), yielding a bit rate of 0.999992 Mbit/sec. The Sync_Seg is a constant 1 tq. The Prop_Seg must be greater than or equal to the propagation delay of 400 ns and so the choice is 10 tq (416.67 ns).

The remaining time quanta (13 tq) in the bit time are divided between Phase_Seg1 and Phase_Seg2 as shown in. Based on this equation, Phase_Seg1 = 6 tq and Phase_Seg2 = 7 tq.

Phase_Seg1 + Phase_Seg2 = Bit_Time – (Synch_Seg + Prop_Seg)

1. If Phase_Seg1 + Phase_Seg2 is even, then Phase_Seg2 = Phase_Seg1. If the sum is odd, Phase_Seg2 = Phase_Seg1 + 1.
2. Phase_Seg2 should be at least 2 t_q.

Equation 22.1. Assigning the Phase Segments

The Synchronization Jump Width (SJW) timing parameter is defined by. It is used for determining the value written to the Bit Timing Register and for determining the required oscillator tolerance. Since we are using a quartz crystal as the system clock source, an oscillator tolerance calculation is not needed.

SJW = minimum (4, Phase_Seg1)

Equation 22.2. Synchronization Jump Width (SJW)

The value written to the Bit Timing Register can be calculated using Equation 18.3. The BRP Extension register is left at its reset value of 0x0000.

BRPE = BRP – 1 = BRP Extension Register = 0x0000

SJWp = SJW – 1 = minimum (4, 6) – 1 = 3

TSEG1 = Prop_Seg + Phase_Seg1 - 1 = 10 + 6 – 1 = 15

TSEG2 = Phase_Seg2 – 1 = 6

Bit Timing Register = (TSEG2 x 0x1000) + (TSEG1 x 0x0100)

Bit Timing Register = (TSEG2 x 0x1000) + (TSEG1 x 0x0100) + (SJWp x 0x0040) + BRPE = 0x6FC0

Equation 22.3. Calculating the Bit Timing Register Value

Table 23.4. SMBus Status Decoding

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Master Transmitter	1110	0	0	X	A master START was generated.	Load slave address + R/W into SMB0DAT.	0	0	X	1100
	1100	0	0	0	A master data or address byte was transmitted; NACK received.	Set STA to restart transfer.	1	0	X	1110
						Abort transfer.	0	1	X	—
		0	0	1	A master data or address byte was transmitted; ACK received.	Load next data byte into SMB0DAT.	0	0	X	1100
						End transfer with STOP.	0	1	X	—
						End transfer with STOP and start another transfer.	1	1	X	—
						Send repeated START.	1	0	X	1110
						Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT).	0	0	X	1000
Master Receiver	1000	1	0	X	A master data byte was received; ACK requested.	Acknowledge received byte; Read SMB0DAT.	0	0	1	1000
						Send NACK to indicate last byte, and send STOP.	0	1	0	—
						Send NACK to indicate last byte, and send STOP followed by START.	1	1	0	1110
						Send ACK followed by repeated START.	1	0	1	1110
						Send NACK to indicate last byte, and send repeated START.	1	0	0	1110
						Send ACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	1	1100
						Send NACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	0	1100

SFR Definition 29.3. PCA1PWM: PCA1 PWM Configuration

Bit	7	6	5	4	3	2	1	0
Name	ARSEL1	ECOV1	COVF1				CLSEL1[1:0]	
Type	R/W	R/W	R/W	R	R	R	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xDA; SFR Page = 0x0F

Bit	Name	Function
7	ARSEL1	**Auto-Reload Register Select.** This bit selects whether to read and write the normal PCA1 capture/compare registers (PCA1CPn), or the Auto-Reload registers at the same SFR addresses. This function is used to define the reload value for 9, 10, and 11-bit PWM modes. In all other modes, the Auto-Reload registers have no function. 0: Read/Write Capture/Compare Registers at PCA1CPHn and PCA1CPLn. 1: Read/Write Auto-Reload Registers at PCA1CPHn and PCA1CPLn.
6	ECOV1	**Cycle Overflow Interrupt Enable.** This bit sets the masking of the Cycle Overflow Flag (COVF1) interrupt. 0: COVF1 will not generate PCA1 interrupts. 1: A PCA1 interrupt will be generated when COVF1 is set.
5	COVF1	**Cycle Overflow Flag.** This bit indicates an overflow of the 8th, 9th, 10th, or 11th bit of the main PCA1 counter (PCA1). The specific bit used for this flag depends on the setting of the Cycle Length Select bits. The bit can be set by hardware or software, but must be cleared by software. 0: No overflow has occurred since the last time this bit was cleared. 1: An overflow has occurred since the last time this bit was cleared.
4:2	Unused	Read = 000b; Write = Don't care.
1:0	CLSEL1[1:0]	**Cycle Length Select.** When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle, between 8, 9, 10, or 11 bits. This affects all channels configured for PWM which are not using 16-bit PWM mode. These bits are ignored for individual channels configured to16-bit PWM mode. 00: 8 bits. 01: 9 bits. 10: 10 bits. 11: 11 bits.

SFR Definition 29.7. PCA1CPLn: PCA1 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA1CPn[7:0]							
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 Addresses: PCA1CPL6 = 0xFB, PCA1CPL7 = 0xE9, PCA1CPL8 = 0xEB, PCA1CPL9 = 0xED, PCA1CPL10 = 0xFD, PCA1CPL11 = 0xCE; SFR Page (all registers) = 0x10

Bit	Name	Function
7:0	PCA1CPn[7:0]	**PCA1 Capture Module Low Byte.** The PCA1CPLn register holds the low byte (LSB) of the 16-bit capture module n. This register address also allows access to the low byte of the corresponding PCA1 channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL1 bit in register PCA1PWM controls which register is accessed.
Note: A write to this register will clear the module's ECOM1n bit to a 0.		

SFR Definition 29.8. PCA1CPHn: PCA1 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA1CPn[15:8]							
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 Addresses: PCA1CPH6 = 0xFC, PCA1CPH7 = 0xEA, PCA1CPH8 = 0xEC, PCA1CPH9 = 0xEE, PCA1CPH10 = 0xFE, PCA1CPH11 = 0xCF; SFR Page (all registers) = 0x10

Bit	Name	Function
7:0	PCA1CPn[15:8]	**PCA1 Capture Module High Byte.** The PCA1CPHn register holds the high byte (MSB) of the 16-bit capture module n. This register address also allows access to the high byte of the corresponding PCA1 channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL1 bit in register PCA1PWM controls which register is accessed.
Note: A write to this register will set the module's ECOM1n bit to a 1.		