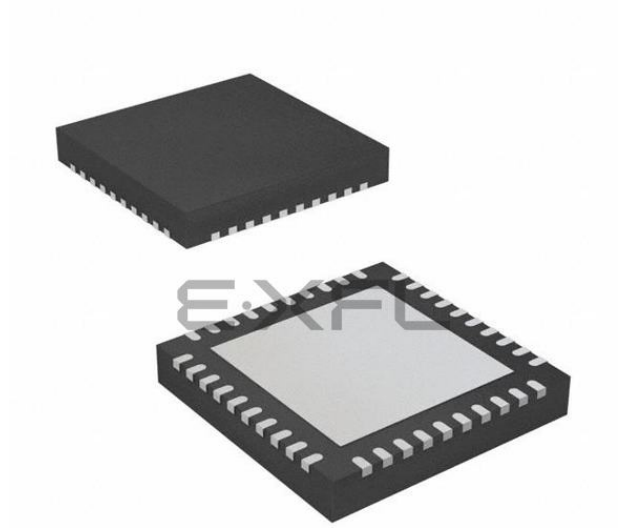


Welcome to E-XFL.COM

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), CANbus, LINbus, SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	33
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2.25K 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/c8051f568-imr

Table 5.4. Reset Electrical Characteristics

–40 to +125 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
$\overline{RST}$ Output Low Voltage	VIO = 5 V; IOL = 70 µA	—	—	40	mV
$\overline{RST}$ Input High Voltage		0.7 x V_{IO}	—	—	
$\overline{RST}$ Input Low Voltage		—	—	0.3 x V_{IO}	
$\overline{RST}$ Input Pullup Current	$\overline{RST}$ = 0.0 V, VIO = 5 V	—	49	115	µA
V_{DD} RST Threshold ($V_{RST-LOW}$)		1.65	1.75	1.80	V
V_{DD} RST Threshold ($V_{RST-HIGH}$)		2.25	2.30	2.45	V
V_{REGIN} Ramp Time for Power On	V_{REGIN} Ramp 0–1.8 V	—	—	1	ms
Missing Clock Detector Timeout	Time from last system clock rising edge to reset initiation V_{DD} = 2.1 V V_{DD} = 2.5 V	 200 200	 340 250	 600 600	µs
Reset Time Delay	Delay between release of any reset source and code execution at location 0x0000	—	155	175	µs
Minimum $\overline{RST}$ Low Time to Generate a System Reset		6	—	—	µs
V_{DD} Monitor Turn-on Time		—	60	100	µs
V_{DD} Monitor Supply Current		—	1	2	µA

Table 5.5. Flash Electrical Characteristics

V_{DD} = 1.8 to 2.75 V, –40 to +125 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Flash Size	C8051F550-3, ‘F560-3, ‘F568-9, and ‘F570-1	32768[1]			Bytes
	C8051F554-7, ‘F564-7, and ‘F572-5	16384			
Endurance		20 k	150 k	—	Erase/Write
Retention	125 °C	10	—	—	Years
Erase Cycle Time	25 MHz System Clock	28	30	45	ms
Write Cycle Time	25 MHz System Clock	79	84	125	µs
V_{DD}	Write/Erase operations	$V_{RST-HIGH}$[2]	—	—	V
Temperature during Programming Operations	–I Devices –A Devices	0 –40	— —	+125 +125	°C

1. On the 32 kB Flash devices, 1024 bytes at addresses 0x7C00 to 0x7FFF are reserved.
2. See Table 5.4 for the $V_{RST-HIGH}$ specification.

Gain Register Definition 6.1. ADC0GNH: ADC0 Selectable Gain High Byte

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

Indirect Address = 0x04;

Bit	Name	Function
7:0	GAINH[7:0]	**ADC0 Gain High Byte.** See Section 6.3.1 for details on calculating the value for this register.
Note: This register is accessed indirectly; See Section 6.3.2 for details for writing this register.		

Gain Register Definition 6.2. ADC0GNL: ADC0 Selectable Gain Low Byte

Bit	7	6	5	4	3	2	1	0
Name	GAINL[3:0]				Reserved	Reserved	Reserved	Reserved
Type	W				W	W	W	W
Reset	0	0	0	0	0	0	0	0

Indirect Address = 0x07;

Bit	Name	Function
7:4	GAINL[3:0]	**ADC0 Gain Lower 4 Bits.** See Figure 6.3.1 for details for setting this register. This register is only accessed indirectly through the ADC0H and ADC0L register.
3:0	Reserved	Must Write 0000b
Note: This register is accessed indirectly; See Section 6.3.2 for details for writing this register.		

6.5. ADC0 Analog Multiplexer

ADC0 includes an analog multiplexer to enable multiple analog input sources. Any of the following may be selected as an input: P0.0–P3.7, the on-chip temperature sensor, the core power supply (V_{DD}), or ground (GND). **ADC0 is single-ended and all signals measured are with respect to GND.** The ADC0 input channels are selected using the ADC0MX register as described in SFR Definition 6.13.

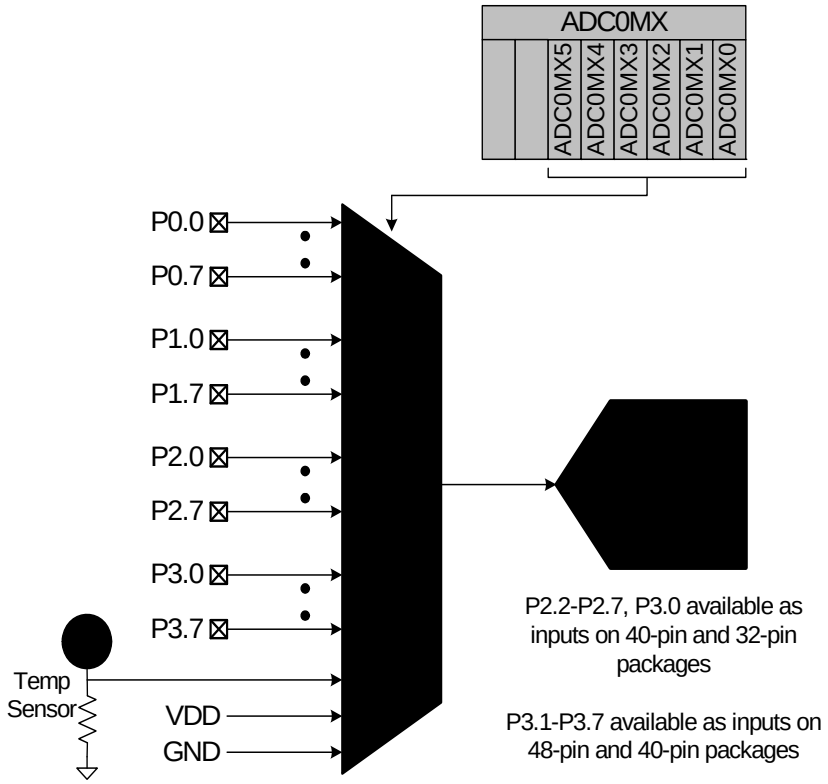


Figure 6.8. ADC0 Multiplexer Block Diagram

Important Note About ADC0 Input Configuration: Port pins selected as ADC0 inputs should be configured as analog inputs, and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, set to 0 the corresponding bit in register PnMDIN. To force the Crossbar to skip a Port pin, set to 1 the corresponding bit in register PnSKIP. See Section “19. Port Input/Output” on page 169 for more Port I/O configuration details.

10. CIP-51 Microcontroller

The MCU system controller core is the CIP-51 microcontroller. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The MCU family has a superset of all the peripherals included with a standard 8051. The CIP-51 also includes on-chip debug hardware (see description in Section 27), and interfaces directly with the analog and digital subsystems providing a complete data acquisition or control-system solution in a single integrated circuit.

The CIP-51 Microcontroller core implements the standard 8051 organization and peripherals as well as additional custom peripherals and functions to extend its capability (see Figure 10.1 for a block diagram). The CIP-51 includes the following features:

- Fully Compatible with MCS-51 Instruction Set
- 50 MIPS Peak Throughput with 50 MHz Clock
- 0 to 50 MHz Clock Frequency
- Extended Interrupt Handler
- Reset Input
- Power Management Modes
- On-chip Debug Logic
- Program and Data Memory Security

10.1. Performance

The CIP-51 employs a pipelined architecture that greatly increases its instruction throughput over the standard 8051 architecture. In a standard 8051, all instructions except for MUL and DIV take 12 or 24 system clock cycles to execute, and usually have a maximum system clock of 12 MHz. By contrast, the CIP-51 core executes 70% of its instructions in one or two system clock cycles, with no instructions taking more than eight system clock cycles.

SFR Definition 10.1. DPL: Data Pointer Low Byte

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

SFR Address = 0x82; SFR Page = All Pages

Bit	Name	Function
7:0	DPL[7:0]	**Data Pointer Low.** The DPL register is the low byte of the 16-bit DPTR. DPTR is used to access indirectly addressed Flash memory or XRAM.

SFR Definition 10.2. DPH: Data Pointer High Byte

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

SFR Address = 0x83; SFR Page = All Pages

Bit	Name	Function
7:0	DPH[7:0]	**Data Pointer High.** The DPH register is the high byte of the 16-bit DPTR. DPTR is used to access indirectly addressed Flash memory or XRAM.

Table 12.3. Special Function Registers (Continued)

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

Register	Address	Description	Page
IT01CF	0xE4	INT0/INT1 Configuration	123
LIN0ADR	0xD3	LIN0 Address	200
LIN0CF	0xC9	LIN0 Configuration	200
LIN0DAT	0xD2	LIN0 Data	201
OSCICN	0xA1	Internal Oscillator Control	160
OSCICRS	0xA2	Internal Oscillator Coarse Control	161
OSCIFIN	0x9E	Internal Oscillator Fine Calibration	161
OSCXCN	0x9F	External Oscillator Control	165
P0	0x80	Port 0 Latch	183
P0MASK	0xF2	Port 0 Mask Configuration	179
P0MAT	0xF1	Port 0 Match Configuration	179
P0MDIN	0xF1	Port 0 Input Mode Configuration	184
P0MDOUT	0xA4	Port 0 Output Mode Configuration	184
P0SKIP	0xD4	Port 0 Skip	185
P1	0x90	Port 1 Latch	185
P1MASK	0xF4	Port 1 Mask Configuration	180
P1MAT	0xF3	Port 1 Match Configuration	180
P1MDIN	0xF2	Port 1 Input Mode Configuration	186
P1MDOUT	0xA5	Port 1 Output Mode Configuration	186
P1SKIP	0xD5	Port 1 Skip	187
P2	0xA0	Port 2 Latch	187
P2MASK	0xB2	Port 2 Mask Configuration	181
P2MAT	0xB1	Port 2 Match Configuration	181
P2MDIN	0xF3	Port 2 Input Mode Configuration	188
P2MDOUT	0xA6	Port 2 Output Mode Configuration	188
P2SKIP	0xD6	Port 2 Skip	189
P3	0xB0	Port 3 Latch	189
P3MASK	0xAF	Port 3 Mask Configuration	182
P3MAT	0xAE	Port 3 Match Configuration	182
P3MDIN	0xF4	Port 3 Input Mode Configuration	190
P3MDOUT	0xAE	Port 3 Output Mode Configuration	190
P3SKIP	0xD7	Port 3 Skip	191
P4	0xB5	Port 4 Latch	191
P4MDOUT	0xAF	Port 4 Output Mode Configuration	192
PCA0CN	0xD8	PCA Control	294
PCA0CPH0	0xFC	PCA Capture 0 High	299

Table 13.1. Interrupt Summary

Interrupt Source	Interrupt Vector	Priority Order	Pending Flag	Bit addressable?	Cleared by HW?	Enable Flag	Priority Control
Reset	0x0000	Top	None	N/A	N/A	Always Enabled	Always Highest
External Interrupt 0 (INT0)	0x0003	0	IE0 (TCON.1)	Y	Y	EX0 (IE.0)	PX0 (IP.0)
Timer 0 Overflow	0x000B	1	TF0 (TCON.5)	Y	Y	ET0 (IE.1)	PT0 (IP.1)
External Interrupt 1 (INT1)	0x0013	2	IE1 (TCON.3)	Y	Y	EX1 (IE.2)	PX1 (IP.2)
Timer 1 Overflow	0x001B	3	TF1 (TCON.7)	Y	Y	ET1 (IE.3)	PT1 (IP.3)
UART0	0x0023	4	RI0 (SCON0.0) TI0 (SCON0.1)	Y	N	ES0 (IE.4)	PS0 (IP.4)
Timer 2 Overflow	0x002B	5	TF2H (TMR2CN.7) TF2L (TMR2CN.6)	Y	N	ET2 (IE.5)	PT2 (IP.5)
SPI0	0x0033	6	SPIF (SPI0CN.7) WCOL (SPI0CN.6) MODF (SPI0CN.5) RXOVRN (SPI0CN.4)	Y	N	ESPI0 (IE.6)	PSPI0 (IP.6)
SMB0	0x003B	7	SI (SMB0CN.0)	Y	N	ESMB0 (EIE1.0)	PSMB0 (EIP1.0)
ADC0 Window Compare	0x0043	8	AD0WINT (ADC0CN.3)	Y	N	EWADC0 (EIE1.1)	PWADC0 (EIP1.1)
ADC0 Conversion Complete	0x004B	9	AD0INT (ADC0CN.5)	Y	N	EADC0 (EIE1.2)	PADC0 (EIP1.2)
Programmable Counter Array	0x0053	10	CF (PCA0CN.7) CCFn (PCA0CN.n) COVF (PCA0PWM.6)	Y	N	EPCA0 (EIE1.3)	PPCA0 (EIP1.3)
Comparator0	0x005B	11	CP0FIF (CPT0CN.4) CP0RIF (CPT0CN.5)	N	N	ECP0 (EIE1.4)	PCP0 (EIP1.4)
Comparator1	0x0063	12	CP1FIF (CPT1CN.4) CP1RIF (CPT1CN.5)	N	N	ECP1 (EIE1.5)	PCP1 (EIP1.5)
Timer 3 Overflow	0x006B	13	TF3H (TMR3CN.7) TF3L (TMR3CN.6)	N	N	ET3 (EIE1.6)	PT3 (EIP1.6)
LIN0	0x0073	14	LIN0INT (LINST.3)	N	N*	ELIN0 (EIE1.7)	PLIN0 (EIP1.7)
Voltage Regulator Dropout	0x007B	15	N/A	N/A	N/A	EREG0 (EIE2.0)	PREG0 (EIP2.0)
CAN0	0x0083	16	CAN0INT (CAN0CN.7)	N	Y	ECAN0 (EIE2.1)	PCAN0 (EIP2.1)
Port Match	0x008B	17	None	N/A	N/A	EMAT (EIE2.2)	PMAT (EIP2.2)

***Note:** The LIN0INT bit is cleared by setting RSTINT (LINCTRL.3)

13.3. External Interrupts $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$

The $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ external interrupt sources are configurable as active high or low, edge or level sensitive. The IN0PL ($\overline{\text{INT0}}$ Polarity) and IN1PL ($\overline{\text{INT1}}$ Polarity) bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON (Section “25.1. Timer 0 and Timer 1” on page 261) select level or edge sensitive. The table below lists the possible configurations.

IT0	IN0PL	$\overline{\text{INT0}}$ Interrupt
1	0	Active low, edge sensitive
1	1	Active high, edge sensitive
0	0	Active low, level sensitive
0	1	Active high, level sensitive

IT1	IN1PL	$\overline{\text{INT1}}$ Interrupt
1	0	Active low, edge sensitive
1	1	Active high, edge sensitive
0	0	Active low, level sensitive
0	1	Active high, level sensitive

$\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ are assigned to Port pins as defined in the IT01CF register (see SFR Definition 13.7). Note that $\overline{\text{INT0}}$ and $\overline{\text{INT0}}$ Port pin assignments are independent of any Crossbar assignments. $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ will monitor their assigned Port pins without disturbing the peripheral that was assigned the Port pin via the Crossbar. To assign a Port pin only to $\overline{\text{INT0}}$ and/or $\overline{\text{INT1}}$, configure the Crossbar to skip the selected pin(s). This is accomplished by setting the associated bit in register XBR0 (see Section “19.3. Priority Crossbar Decoder” on page 172 for complete details on configuring the Crossbar).

IE0 (TCON.1) and IE1 (TCON.3) serve as the interrupt-pending flags for the $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ external interrupts, respectively. If an $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$ external interrupt is configured as edge-sensitive, the corresponding interrupt-pending flag is automatically cleared by the hardware when the CPU vectors to the ISR. When configured as level sensitive, the interrupt-pending flag remains logic 1 while the input is active as defined by the corresponding polarity bit (IN0PL or IN1PL); the flag remains logic 0 while the input is inactive. The external interrupt source must hold the input active until the interrupt request is recognized. It must then deactivate the interrupt request before execution of the ISR completes or another interrupt request will be generated.

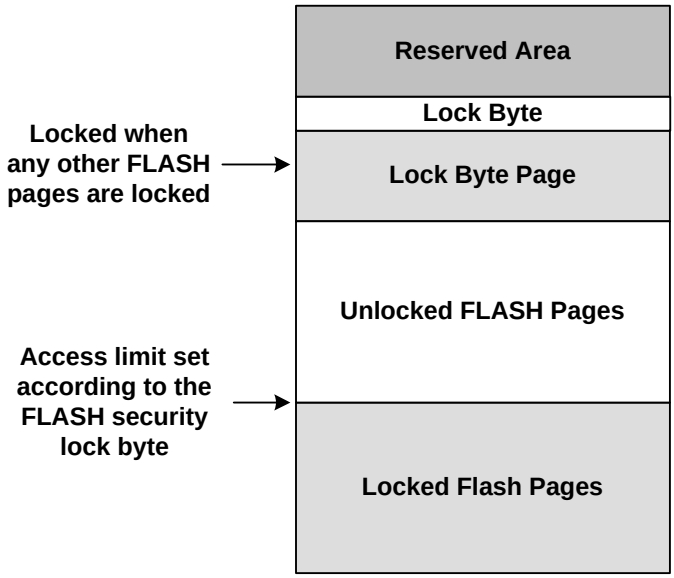
14.2. Non-volatile Data Storage

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: MOVX read instructions always target XRAM.

14.3. 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 14.1.



Security Lock Byte:	11111101b
1s Complement:	00000010b
Flash pages locked:	3 (First two Flash pages + Lock Byte Page)

Figure 14.1. Flash Program Memory Map

Table 17.1. EMIF Pinout (C8051F568-9 and ‘F570-5)

Multiplexed Mode	
Signal Name	**Port Pin**
$\overline{RD}$	P1.6
$\overline{WR}$	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

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. When the WEAKPUD bit in XBR2 is 0, a weak pullup is enabled for all Port I/O configured as open-drain. WEAKPUD does not affect the push-pull Port I/O. Furthermore, the weak pullup is turned off on an output that is driving a 0 to avoid unnecessary power dissipation.

Registers XBR0, XBR1, and XBR2 must be loaded with the appropriate values to select the digital I/O functions required by the design. Setting the XBARE bit in XBR2 to 1 enables the Crossbar. Until the Crossbar is enabled, the external pins remain as standard Port I/O (in input mode), regardless of the XBRn Register settings. For given XBRn Register settings, one can determine the I/O pin-out using the Priority Decode Table; as an alternative, the Configuration Wizard utility of the Silicon Labs IDE software will determine the Port I/O pin-assignments based on the XBRn Register settings.

The Crossbar must be enabled to use Port pins as standard Port I/O in output mode. Port output drivers are disabled while the Crossbar is disabled.

$$multiplier = \frac{20000}{baud\_rate} - 1$$

$$prescaler = \ln\left[\frac{SYSCLK}{(multiplier + 1) \times baud\_rate \times 200}\right] \times \frac{1}{\ln 2} - 1$$

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

In all of these equations, the results must be rounded down to the nearest integer.

The following example shows the steps for calculating the baud rate values for a Master node running at 24 MHz and communicating at 19200 bits/sec. First, calculate the multiplier:

$$multiplier = \frac{20000}{19200} - 1 = 0.0417 \cong 0$$

Next, calculate the prescaler:

$$prescaler = \ln\frac{24000000}{(0 + 1) \times 19200 \times 200} \times \frac{1}{\ln 2} - 1 = 1.644 \cong 1$$

Finally, calculate the divider:

$$divider = \frac{24000000}{2^{(1 + 1)} \times (0 + 1) \times 19200} = 312.5 \cong 312$$

These values lead to the following baud rate:

$$baud\_rate = \frac{24000000}{2^{(1 + 1)} \times (0 + 1) \times 312} \cong 19230.77$$

The following code programs the interface in Master mode, using the Enhanced Checksum and enables the interface to operate at 19230 bits/sec using a 24 MHz system clock.

```
LIN0CF   = 0x80;                        // Activate the interface
LIN0CF  |= 0x40;                        // Set the node as a Master

LIN0ADR  = 0x0D;                        // Point to the LIN0MUL register
// Initialize the register (prescaler, multiplier and bit 8 of divider)
LIN0DAT  = ( 0x01 << 6 ) + ( 0x00 << 1 ) + ( ( 0x138 & 0x0100 ) >> 8 );
LIN0ADR   = 0x0C;                       // Point to the LIN0DIV register
LIN0DAT   = (unsigned char)_0x138;      // Initialize LIN0DIV

LIN0ADR   = 0x0B;                       // Point to the LIN0SIZE register
LIN0DAT  |= 0x80;                       // Initialize the checksum as Enhanced

LIN0ADR   = 0x08;                       // Point to LIN0CTRL register
LIN0DAT   = 0x0C;                       // Reset any error and the interrupt
```

Table 20.2 includes the configuration values required for the typical system clocks and baud rates:

20.7. LIN Registers

The following Special Function Registers (SFRs) and indirect registers are available for the LIN controller.

20.7.1. LIN Direct Access SFR Registers Definitions

SFR Definition 20.1. LIN0ADR: LIN0 Indirect Address Register

Bit	7	6	5	4	3	2	1	0
Name	LIN0ADR[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD3; SFR Page = 0x00

Bit	Name	Function
7:0	LIN0ADR[7:0]	**LIN Indirect Address Register Bits.** This register hold an 8-bit address used to indirectly access the LIN0 core registers. Table 20.4 lists the LIN0 core registers and their indirect addresses. Reads and writes to LIN0DAT will target the register indicated by the LIN0ADR bits.

SFR Definition 20.2. LIN0DAT: LIN0 Indirect Data Register

Bit	7	6	5	4	3	2	1	0
Name	LIN0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD2; SFR Page = 0x00

Bit	Name	Function
7:0	LIN0DAT[7:0]	**LIN Indirect Data Register Bits.** When this register is read, it will read the contents of the LIN0 core register pointed to by LIN0ADR. When this register is written, it will write the value to the LIN0 core register pointed to by LIN0ADR.

LIN Register Definition 20.4. 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.

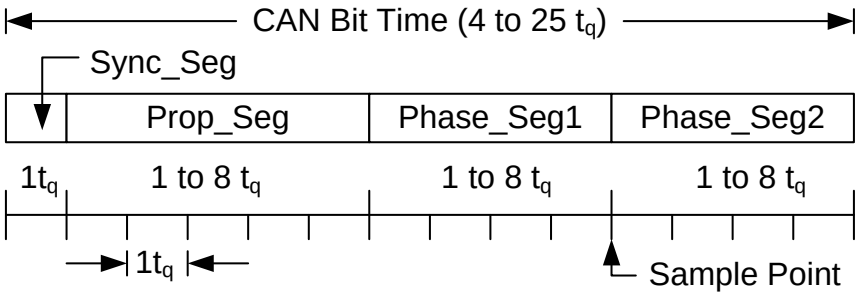


Figure 21.3. Four segments of a CAN Bit

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 tq.

Equation 21.1. 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 21.2. 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 21.3. Calculating the Bit Timing Register Value

22.5.3. 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 22.7 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.

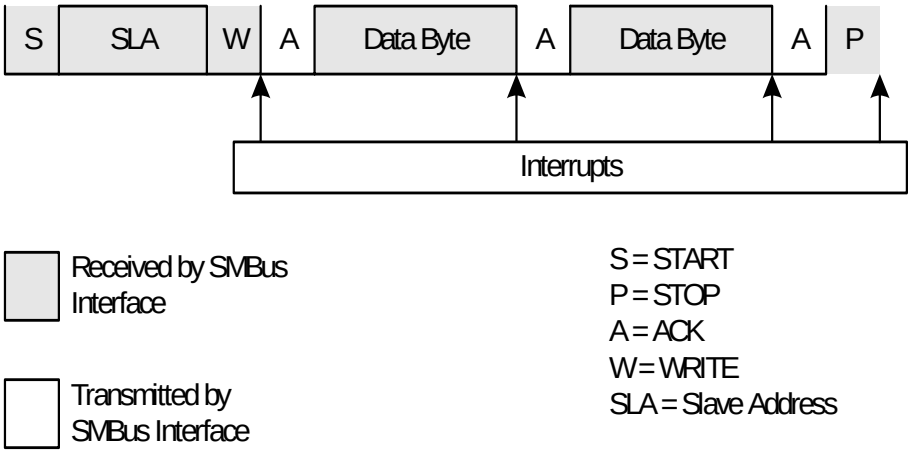


Figure 22.7. Typical Slave Write Sequence

$$\text{Baud Rate} = \frac{\text{SYSCLK}}{(65536 - (\text{SBRLH0:SBRLL0}))} \times \frac{1}{2} \times \frac{1}{\text{Prescaler}}$$

Equation 23.1. UART0 Baud Rate

A quick reference for typical baud rates and clock frequencies is given in Table 23.1.

Table 23.1. Baud Rate Generator Settings for Standard Baud Rates

	Target Baud Rate (bps)	Actual Baud Rate (bps)	Baud Rate Error	Oscillator Divide Factor	SB0PS[1:0] (Prescaler Bits)	Reload Value in SBRLH0:SBRLL0
SYSCLK = 48	230400	230769	0.16%	208	11	0xFF98
	115200	115385	0.16%	416	11	0xFF30
	57600	57554	0.08%	834	11	0xFE5F
	28800	28812	0.04%	1666	11	0xFCBF
	14400	14397	0.02%	3334	11	0xF97D
	9600	9600	0.00%	5000	11	0xF63C
	2400	2400	0.00%	20000	11	0xD8F0
	1200	1200	0.00%	40000	11	0xB1E0
SYSCLK = 24	230400	230769	0.16%	104	11	0xFFCC
	115200	115385	0.16%	208	11	0xFF98
	57600	57692	0.16%	416	11	0xFF30
	28800	28777	0.08%	834	11	0xFE5F
	14400	14406	0.04%	1666	11	0xFCBF
	9600	9600	0.00%	2500	11	0xFB1E
	2400	2400	0.00%	10000	11	0xEC78
	1200	1200	0.00%	20000	11	0xD8F0
SYSCLK = 12	230400	230769	0.16%	52	11	0xFFE6
	115200	115385	0.16%	104	11	0xFFCC
	57600	57692	0.16%	208	11	0xFF98
	28800	28846	0.16%	416	11	0xFF30
	14400	14388	0.08%	834	11	0xFE5F
	9600	9600	0.00%	1250	11	0xFD8F
	2400	2400	0.00%	5000	11	0xF63C
	1200	1200	0.00%	10000	11	0xEC78

23.2. Data Format

UART0 has a number of available options for data formatting. Data transfers begin with a start bit (logic low), followed by the data bits (sent LSB-first), a parity or extra bit (if selected), and end with one or two stop bits (logic high). The data length is variable between 5 and 8 bits. A parity bit can be appended to the data, and automatically generated and detected by hardware for even, odd, mark, or space parity. The stop bit length is selectable between 1 and 2 bit times, and a multi-processor communication mode is available for implementing networked UART buses. All of the data formatting options can be configured using the SMOD0 register, shown in SFR Definition 23.2. Figure 23.2 shows the timing for a UART0 transaction without parity or an extra bit enabled. Figure 23.3 shows the timing for a UART0 transaction with parity enabled (PE0 = 1). Figure 23.4 is an example of a UART0 transaction when the extra bit is enabled (XBE0 = 1). Note that the extra bit feature is not available when parity is enabled, and the second stop bit is only an option for data lengths of 6, 7, or 8 bits.

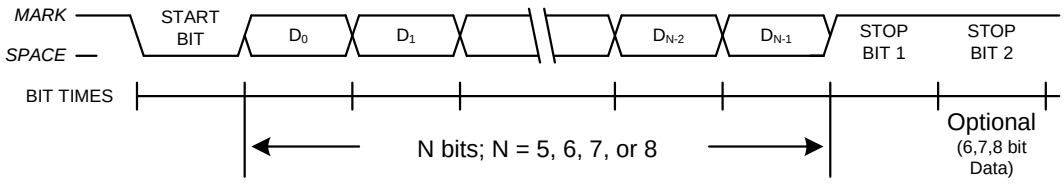


Figure 23.2. UART0 Timing Without Parity or Extra Bit

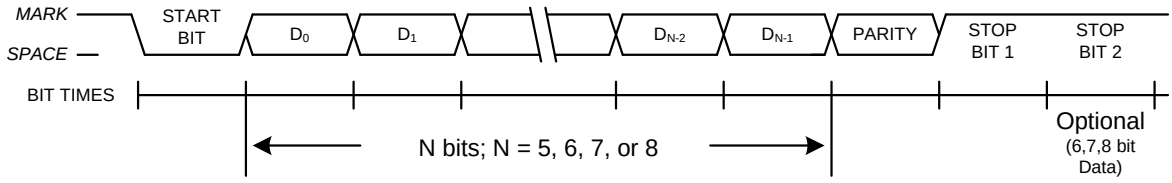


Figure 23.3. UART0 Timing With Parity

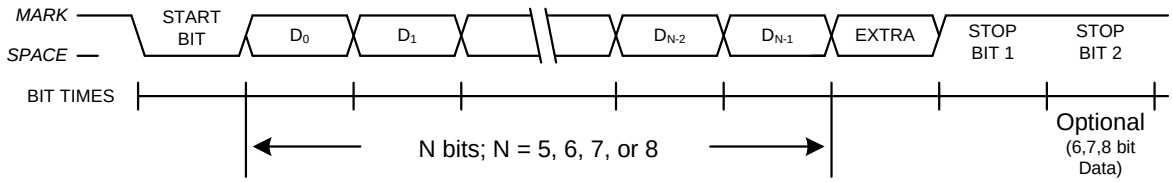


Figure 23.4. UART0 Timing With Extra Bit

23.3. Configuration and Operation

UART0 provides standard asynchronous, full duplex communication. It can operate in a point-to-point serial communications application, or as a node on a multi-processor serial interface. To operate in a point-to-point application, where there are only two devices on the serial bus, the MCE0 bit in SMOD0 should be cleared to 0. For operation as part of a multi-processor communications bus, the MCE0 and XBE0 bits should both be set to 1. In both types of applications, data is transmitted from the microcontroller on the TX0 pin, and received on the RX0 pin. The TX0 and RX0 pins are configured using the crossbar and the Port I/O registers, as detailed in Section “19. Port Input/Output” on page 169.

In typical UART communications, The transmit (TX) output of one device is connected to the receive (RX) input of the other device, either directly or through a bus transceiver, as shown in Figure 23.5.

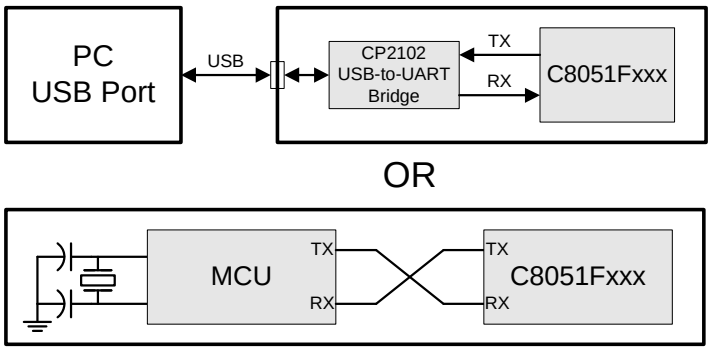


Figure 23.5. Typical UART Interconnect Diagram

23.3.1. Data Transmission

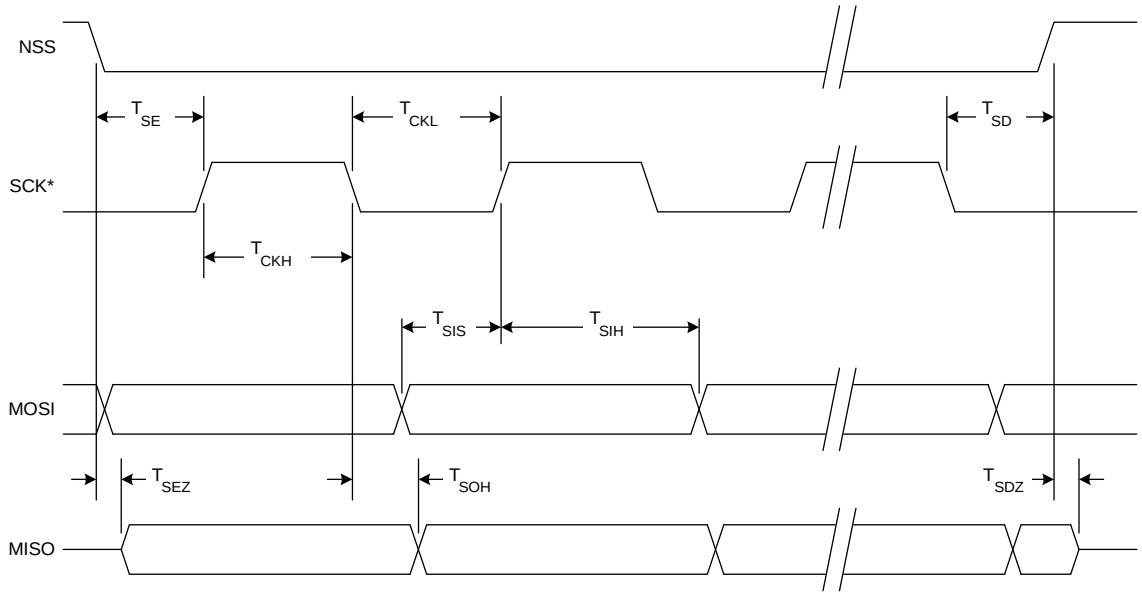
Data transmission is double-buffered and begins when software writes a data byte to the SBUF0 register. Writing to SBUF0 places data in the Transmit Holding Register, and the Transmit Holding Register Empty flag (THRE0) will be cleared to 0. If the UART’s shift register is empty (i.e., no transmission in progress), the data will be placed in the Transmit Holding Register until the current transmission is complete. The TI0 Transmit Interrupt Flag (SCON0.1) will be set at the end of any transmission (the beginning of the stop-bit time). If enabled, an interrupt will occur when TI0 is set.

Note: THRE0 can have a momentary glitch high when the UART Transmit Holding Register is not empty. The glitch will occur some time after SBUF0 was written with the previous byte and does not occur if THRE0 is checked in the instruction(s) immediately following the write to SBUF0. When firmware writes SBUF0 and SBUF0 is not empty, TX0 will be stuck low until the next device reset. Firmware should use or poll on TI0 rather than THRE0 for asynchronous UART writes that may have a random delay in between transactions.

If the extra bit function is enabled (XBE0 = 1) and the parity function is disabled (PE0 = ‘0’), the value of the TBX0 (SCON0.3) bit will be sent in the extra bit position. When the parity function is enabled (PE0 = 1), hardware will generate the parity bit according to the selected parity type (selected with S0PT[1:0]), and append it to the data field. Note: when parity is enabled, the extra bit function is not available.

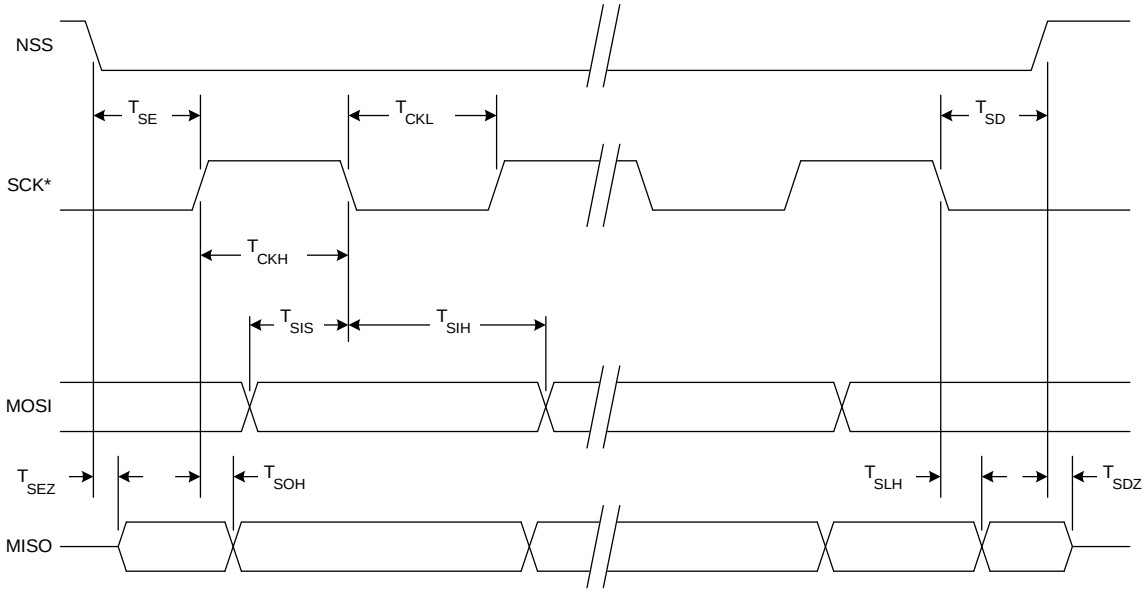
23.3.2. Data Reception

Data reception can begin any time after the REN0 Receive Enable bit (SCON0.4) is set to logic 1. After the stop bit is received, the data byte will be stored in the receive FIFO if the following conditions are met: the receive FIFO (3 bytes deep) must not be full, and the stop bit(s) must be logic 1. In the event that the receive FIFO is full, the incoming byte will be lost, and a Receive FIFO Overrun Error will be generated (OVR0 in register SCON0 will be set to logic 1). If the stop bit(s) were logic 0, the incoming data will not be stored in the receive FIFO. If the reception conditions are met, the data is stored in the receive FIFO, and



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 24.10. SPI Slave Timing (CKPHA = 0)



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 24.11. SPI Slave Timing (CKPHA = 1)