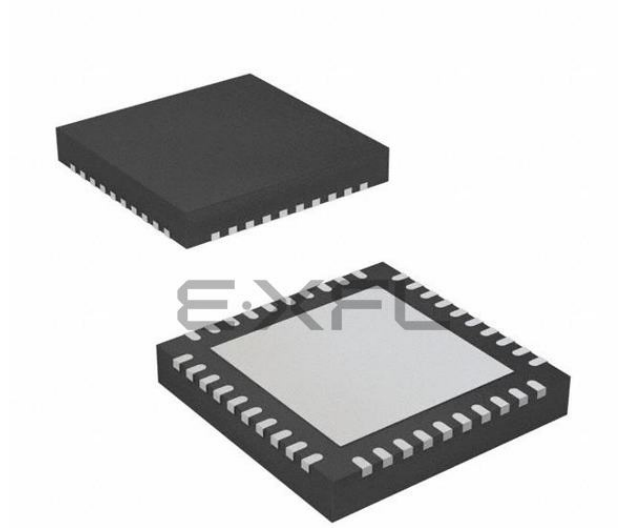

Welcome to E-XFL.COM

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	16KB (16K 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/c8051f572-im

Table 3.1. Pin Definitions for the C8051F55x/56x/57x (Continued)

Name	Pin 40-pin packages	Pin 32-pin packages	Pin 24-pin packages	Type	Description
P1.0	34	26	17	D I/O or A In	Port 1.0. See SFR Definition 19.16.
P1.1	33	25	16	D I/O or A In	Port 1.1.
P1.2	32	24	15	D I/O or A In	Port 1.2.
P1.3	31	23	14	D I/O or A In	Port 1.3.
P1.4	30	22	13	D I/O or A In	Port 1.4.
P1.5	29	21	12	D I/O or A In	Port 1.5.
P1.6	28	20	11	D I/O or A In	Port 1.6.
P1.7	27	19	10	D I/O or A In	Port 1.7.
P2.0	26	18	9	D I/O or A In	Port 2.0. See SFR Definition 19.20.
P2.1	25	17	—	D I/O or A In	Port 2.1.
P2.2	24	16	—	D I/O or A In	Port 2.2.
P2.3	23	15	—	D I/O or A In	Port 2.3.
P2.4	22	14	—	D I/O or A In	Port 2.4.
P2.5	21	13	—	D I/O or A In	Port 2.5.
P2.6	20	12	—	D I/O or A In	Port 2.6.
P2.7	19	11	—	D I/O or A In	Port 2.7.
P3.0	18	—	—	D I/O or A In	Port 3.0. See SFR Definition 19.24.
P3.1	17	—	—	D I/O or A In	Port 3.1.
P3.2	16	—	—	D I/O or A In	Port 3.2.
P3.3	15	—	—	D I/O or A In	Port 3.3.
P3.4	14	—	—	D I/O or A In	Port 3.4.
P3.5	13	—	—	D I/O or A In	Port 3.5.
P3.6	12	—	—	D I/O or A In	Port 3.6.
P3.7	11	—	—	D I/O or A In	Port 3.7.

SFR Definition 8.3. CPT1CN: Comparator1 Control

Bit	7	6	5	4	3	2	1	0
Name	CP1EN	CP1OUT	CP1RIF	CP1FIF	CP1HYP[1:0]		CP1HYN[1:0]	
Type	R/W	R	R/W	R/W	R/W		R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9D; SFR Page = 0x00

Bit	Name	Function
7	CP1EN	**Comparator1 Enable Bit.** 0: Comparator1 Disabled. 1: Comparator1 Enabled.
6	CP1OUT	**Comparator1 Output State Flag.** 0: Voltage on CP1+ < CP1–. 1: Voltage on CP1+ > CP1–.
5	CP1RIF	**Comparator1 Rising-Edge Flag. Must be cleared by software.** 0: No Comparator1 Rising Edge has occurred since this flag was last cleared. 1: Comparator1 Rising Edge has occurred.
4	CP1FIF	**Comparator1 Falling-Edge Flag. Must be cleared by software.** 0: No Comparator1 Falling-Edge has occurred since this flag was last cleared. 1: Comparator1 Falling-Edge has occurred.
3:2	CP1HYP[1:0]	**Comparator1 Positive Hysteresis Control Bits.** 00: Positive Hysteresis Disabled. 01: Positive Hysteresis = 5 mV. 10: Positive Hysteresis = 10 mV. 11: Positive Hysteresis = 20 mV.
1:0	CP1HYN[1:0]	**Comparator1 Negative Hysteresis Control Bits.** 00: Negative Hysteresis Disabled. 01: Negative Hysteresis = 5 mV. 10: Negative Hysteresis = 10 mV. 11: Negative Hysteresis = 20 mV.

SFR Definition 8.5. CPT0MX: Comparator0 MUX Selection

Bit	7	6	5	4	3	2	1	0
Name	CMX0N[3:0]				CMX0P[3:0]			
Type	R/W				R/W			
Reset	0	1	1	1	0	1	1	1

SFR Address = 0x9C; SFR Page = 0x00

Bit	Name	Function
7:4	CMX0N[3:0]	**Comparator0 Negative Input MUX Selection.** 0000: P0.1 0001: P0.3 0010: P0.5 0011: P0.7 0100: P1.1 0101: P1.3 0110: P1.5 0111: P1.7 1000: P2.1 1001: P2.3 (only available on 40-pin and 32-pin devices) 1010: P2.5 (only available on 40-pin and 32-pin devices) 1011: P2.7 (only available on 40-pin and 32-pin devices) 1100–1111: None
3:0	CMX0P[3:0]	**Comparator0 Positive Input MUX Selection.** 0000: P0.0 0001: P0.2 0010: P0.4 0011: P0.6 0100: P1.0 0101: P1.2 0110: P1.4 0111: P1.6 1000: P2.0 1001: P2.2 (only available on 40-pin and 32-pin devices) 1010: P2.4 (only available on 40-pin and 32-pin devices) 1011: P2.6 (only available on 40-pin and 32-pin devices) 1100–1111: None

SFR Definition 13.3. EIE1: Extended Interrupt Enable 1

Bit	7	6	5	4	3	2	1	0
Name	ELIN0	ET3	ECP1	ECP0	EPCA0	EADC0	EWADC0	ESMB0
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 = 0xE6; SFR Page = All Pages

Bit	Name	Function
7	ELIN0	**Enable LIN0 Interrupt.** This bit sets the masking of the LIN0 interrupt. 0: Disable LIN0 interrupts. 1: Enable interrupt requests generated by the LIN0INT flag.
6	ET3	**Enable Timer 3 Interrupt.** This bit sets the masking of the Timer 3 interrupt. 0: Disable Timer 3 interrupts. 1: Enable interrupt requests generated by the TF3L or TF3H flags.
5	ECP1	**Enable Comparator1 (CP1) Interrupt.** This bit sets the masking of the CP1 interrupt. 0: Disable CP1 interrupts. 1: Enable interrupt requests generated by the CP1RIF or CP1FIF flags.
4	ECP0	**Enable Comparator0 (CP0) Interrupt.** This bit sets the masking of the CP0 interrupt. 0: Disable CP0 interrupts. 1: Enable interrupt requests generated by the CP0RIF or CP0FIF flags.
3	EPCA0	**Enable Programmable Counter Array (PCA0) Interrupt.** This bit sets the masking of the PCA0 interrupts. 0: Disable all PCA0 interrupts. 1: Enable interrupt requests generated by PCA0.
2	EADC0	**Enable ADC0 Conversion Complete Interrupt.** This bit sets the masking of the ADC0 Conversion Complete interrupt. 0: Disable ADC0 Conversion Complete interrupt. 1: Enable interrupt requests generated by the AD0INT flag.
1	EWADC0	**Enable Window Comparison ADC0 Interrupt.** This bit sets the masking of ADC0 Window Comparison interrupt. 0: Disable ADC0 Window Comparison interrupt. 1: Enable interrupt requests generated by ADC0 Window Compare flag (AD0WINT).
0	ESMB0	**Enable SMBus (SMB0) Interrupt.** This bit sets the masking of the SMB0 interrupt. 0: Disable all SMB0 interrupts. 1: Enable interrupt requests generated by SMB0.

16.2. Power-Fail Reset/V_{DD} Monitor

When a power-down transition or power irregularity causes V_{DD} to drop below V_{RST}, the power supply monitor will drive the $\overline{RST}$ pin low and hold the CIP-51 in a reset state (see Figure 16.2). When V_{DD} returns to a level above V_{RST}, the CIP-51 will be released from the reset state. Note that even though internal data memory contents are not altered by the power-fail reset, it is impossible to determine if V_{DD} dropped below the level required for data retention. If the PORSF flag reads 1, the data may no longer be valid. The V_{DD} monitor is enabled after power-on resets. Its defined state (enabled/disabled) is not altered by any other reset source. For example, if the V_{DD} monitor is disabled by code and a software reset is performed, the V_{DD} monitor will still be disabled after the reset. **To protect the integrity of Flash contents, the V_{DD} monitor must be enabled to the higher setting (VDMLVL = 1) and selected as a reset source if software contains routines which erase or write Flash memory. If the V_{DD} monitor is not enabled and set to the high level, any erase or write performed on Flash memory will cause a Flash Error device reset.**

Important Note: If the V_{DD} monitor is being turned on from a disabled state, it should be enabled before it is selected as a reset source. Selecting the V_{DD} monitor as a reset source before it is enabled and stabilized may cause a system reset. In some applications, this reset may be undesirable. If this is not desirable in the application, a delay should be introduced between enabling the monitor and selecting it as a reset source. The procedure for enabling the V_{DD} monitor and configuring it as a reset source from a disabled state is as follows:

1. Enable the V_{DD} monitor (VDMEN bit in VDM0CN = 1).
2. If necessary, wait for the V_{DD} monitor to stabilize (see Table 5.4 for the V_{DD} Monitor turn-on time).
   Note: This delay should be omitted if software contains routines that erase or write Flash memory.
3. Select the V_{DD} monitor as a reset source (PORSF bit in RSTSRC = 1).

See Figure 16.2 for V_{DD} monitor timing; note that the power-on-reset delay is not incurred after a V_{DD} monitor reset. See Table 5.4 for complete electrical characteristics of the V_{DD} monitor.

Note: The output of the internal voltage regulator is calibrated by the MCU immediately after any reset event. The output of the un-calibrated internal regulator could be below the high threshold setting of the V_{DD} Monitor. If this is the case *and* the V_{DD} Monitor is set to the high threshold setting *and* if the MCU receives a non-power on reset (POR), the MCU will remain in reset until a POR occurs (i.e., V_{DD} Monitor will keep the device in reset). A POR will force the V_{DD} Monitor to the low threshold setting which is guaranteed to be below the un-calibrated output of the internal regulator. The device will then exit reset and resume normal operation. It is for this reason Silicon Labs strongly recommends that the V_{DD} Monitor is always left in the low threshold setting (i.e. default value upon POR).

When programming the Flash in-system, the V_{DD} Monitor must be set to the high threshold setting. For the highest system reliability, the time the V_{DD} Monitor is set to the high threshold setting should be minimized (e.g., setting the V_{DD} Monitor to the high threshold setting just before the Flash write operation and then changing it back to the low threshold setting immediately after the Flash write operation).

Note: The V_{DD} Monitor may trigger on fast changes in voltage on the VDD pin, regardless of whether the voltage increased or decreased.

18.4. External Oscillator Drive Circuit

The external oscillator circuit may drive an external crystal, ceramic resonator, capacitor, or RC network. A CMOS clock may also provide a clock input. For a crystal or ceramic resonator configuration, the crystal/resonator must be wired across the XTAL1 and XTAL2 pins as shown in Option 1 of Figure 18.1. A 10 MΩ resistor also must be wired across the XTAL2 and XTAL1 pins for the crystal/resonator configuration. In RC, capacitor, or CMOS clock configuration, the clock source should be wired to the XTAL2 pin as shown in Option 2, 3, or 4 of Figure 18.1. The type of external oscillator must be selected in the OSCXCN register, and the frequency control bits (XFCN) must be selected appropriately (see SFR Definition 18.6).

Important Note on External Oscillator Usage: Port pins must be configured when using the external oscillator circuit. When the external oscillator drive circuit is enabled in crystal/resonator mode, Port pins P0.2 and P0.3 are used as XTAL1 and XTAL2 respectively. When the external oscillator drive circuit is enabled in capacitor, RC, or CMOS clock mode, Port pin P0.3 is used as XTAL2. The Port I/O Crossbar should be configured to skip the Port pins used by the oscillator circuit; see Section “19.3. Priority Crossbar Decoder” on page 172 for Crossbar configuration. Additionally, when using the external oscillator circuit in crystal/resonator, capacitor, or RC mode, the associated Port pins should be configured as **analog inputs**. In CMOS clock mode, the associated pin should be configured as a **digital input**. See Section “19.4. Port I/O Initialization” on page 174 for details on Port input mode selection.

SFR Definition 19.13. P0MDIN: Port 0 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P0MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF1; SFR Page = 0x0F

Bit	Name	Function
7:0	P0MDIN[7:0]	**Analog Configuration Bits for P0.7–P0.0 (respectively).** Port pins configured for analog mode have their weak pull-up and digital receiver disabled. For analog mode, the pin also needs to be configured for open-drain mode in the P0MDOUT register. 0: Corresponding P0.n pin is configured for analog mode. 1: Corresponding P0.n pin is not configured for analog mode.

SFR Definition 19.14. P0MDOUT: Port 0 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P0MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA4; SFR Page = 0x0F

Bit	Name	Function
7:0	P0MDOUT[7:0]	**Output Configuration Bits for P0.7–P0.0 (respectively).** These bits are ignored if the corresponding bit in register P0MDIN is logic 0. 0: Corresponding P0.n Output is open-drain. 1: Corresponding P0.n Output is push-pull.

SFR Definition 19.23. P2SKIP: Port 2 Skip

Bit	7	6	5	4	3	2	1	0
Name	P2SKIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD6; SFR Page = 0x0F

Bit	Name	Function
7:0	P2SKIP[7:0]	**Port 2 Crossbar Skip Enable Bits.** These bits select Port 2 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 P2.n pin is not skipped by the Crossbar. 1: Corresponding P2.n pin is skipped by the Crossbar.
Note: P2.2-P2.7 are available on 40-pin and 32-pin packages.		

SFR Definition 19.24. P3: Port 3

Bit	7	6	5	4	3	2	1	0
Name	P3[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xB0; SFR Page = All Pages; Bit-Addressable

Bit	Name	Description	Write	Read
7:0	P3[7:0]	**Port 3 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: P3.n Port pin is logic LOW. 1: P3.n Port pin is logic HIGH.
Note: P3.0 is available on 40-pin and 32-pin packages. P3.1-P3.7 are available on 40-pin packages				

SFR Definition 19.27. 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: P3.0 is available on 40-pin and 32-pin packages. P3.1-P3.7 are available on 40-pin packages		

SFR Definition 19.28. 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 available on 40-pin packages.				

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

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

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

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:

Table 20.3. Autobaud Parameters Examples

System Clock (MHz)	Prescaler	Divider
25	1	312
24.5	1	306
24	1	300
22.1184	1	276
16	1	200
12.25	0	306
12	0	300
11.0592	0	276
8	0	200

20.3. LIN Master Mode Operation

The master node is responsible for the scheduling of messages and sends the header of each frame containing the SYNCH BREAK FIELD, SYNCH FIELD, and IDENTIFIER FIELD. The steps to schedule a message transmission or reception are listed below.

1. Load the 6-bit Identifier into the LIN0ID register.
2. Load the data length into the LIN0SIZE register. Set the value to the number of data bytes or "1111b" if the data length should be decoded from the identifier. Also, set the checksum type, classic or enhanced, in the same LIN0SIZE register.
3. Set the data direction by setting the TXRX bit (LIN0CTRL.5). Set the bit to 1 to perform a master transmit operation, or set the bit to 0 to perform a master receive operation.
4. If performing a master transmit operation, load the data bytes to transmit into the data buffer (LIN0DT1 to LIN0DT8).
5. Set the STREQ bit (LIN0CTRL.0) to start the message transfer. The LIN controller will schedule the message frame and request an interrupt if the message transfer is successfully completed or if an error has occurred.

This code segment shows the procedure to schedule a message in a transmission operation:

```
LIN0ADR   = 0x08;               // Point to LIN0CTRL
LIN0DAT  |= 0x20;               // Select to transmit data
LIN0ADR   = 0x0E;               // Point to LIN0ID
LIN0DAT   = 0x11;               // Load the ID, in this example 0x11
LIN0ADR   = 0x0B;               // Point to LIN0SIZE
LIN0DAT  = ( LIN0DAT & 0xF0 ) | 0x08;      // Load the size with 8

LIN0ADR  = 0x00;               // Point to Data buffer first byte
for (i=0; i<8; i++)
{
   LIN0DAT = i + 0x41;         // Load the buffer with ‘A’, ‘B’, ...
   LIN0ADR++;                  // Increment the address to the next buffer
}
LIN0ADR  = 0x08;               // Point to LIN0CTRL
LIN0DAT  = 0x01;               // Start Request
```

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.

LIN Register Definition 20.9. LIN0DIV: LIN0 Divider Register

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

Indirect Address = 0x0C

Bit	Name	Function
7:0	DIVLSB	**LIN Baud Rate Divider Least Significant Bits.** The 8 least significant bits for the baud rate divider. The 9th and most significant bit is the DIV9 bit (LIN0MUL.0). The valid range for the divider is 200 to 511.

LIN Register Definition 20.10. LIN0MUL: LIN0 Multiplier Register

Bit	7	6	5	4	3	2	1	0
Name	PRESCL[1:0]		LINMUL[4:0]					DIV9
Type	R/W		R/W					R/W
Reset	1	1	1	1	1	1	1	1

Indirect Address = 0x0D

Bit	Name	Function
7:6	PRESCL[1:0]	**LIN Baud Rate Prescaler Bits.** These bits are the baud rate prescaler bits.
5:1	LINMUL[4:0]	**LIN Baud Rate Multiplier Bits.** These bits are the baud rate multiplier bits. These bits are not used in slave mode.
0	DIV9	**LIN Baud Rate Divider Most Significant Bit.** The most significant bit of the baud rate divider. The 8 least significant bits are in LIN0DIV. The valid range for the divider is 200 to 511.

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

21.1.2. CAN Register Access

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

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

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

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

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

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

21.1.3. Example Timing Calculation for 1 Mbit/Sec Communication

This example shows how to configure the CAN controller timing parameters for a 1 Mbit/Sec bit rate. Table 21.1 shows timing-related system parameters needed for the calculation.

Table 21.1. Background System Information

Parameter	Value	Description
CIP-51 system clock (SYSCLK)	24 MHz	Internal Oscillator Max
CAN controller clock (f_{sys})	24 MHz	CAN0CFG divider set to 1
CAN clock period (t_{sys})	41.667 ns	Derived from $1/f_{sys}$
CAN time quantum (t_q)	41.667 ns	Derived from t_{sys} x BRP[1,2]
CAN bus length	10 m	5 ns/m signal delay between CAN nodes
Propogation delay time[3]	400 ns	2 x (transceiver loop delay + bus line delay)

Notes:
1. The CAN time quantum is the smallest unit of time recognized by the CAN controller. Bit timing parameters are specified in integer multiples of the time quantum.
2. The Baud Rate Prescaler (BRP) is defined as the value of the BRP Extension Register plus 1. The BRP extension register has a reset value of 0x0000. The BRP has a reset value of 1.
3. Based on an ISO-11898 compliant transceiver. CAN does not specify a physical layer.

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

22. SMBus

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the I2C serial bus. Reads and writes to the interface by the system controller are byte oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/20th of the system clock as a master or slave (this can be faster than allowed by the SMBus specification, depending on the system clock used). A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The SMBus interface may operate as a master and/or slave, and may function on a bus with multiple masters. The SMBus provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and START/STOP control and generation. A block diagram of the SMBus peripheral and the associated SFRs is shown in Figure 22.1.

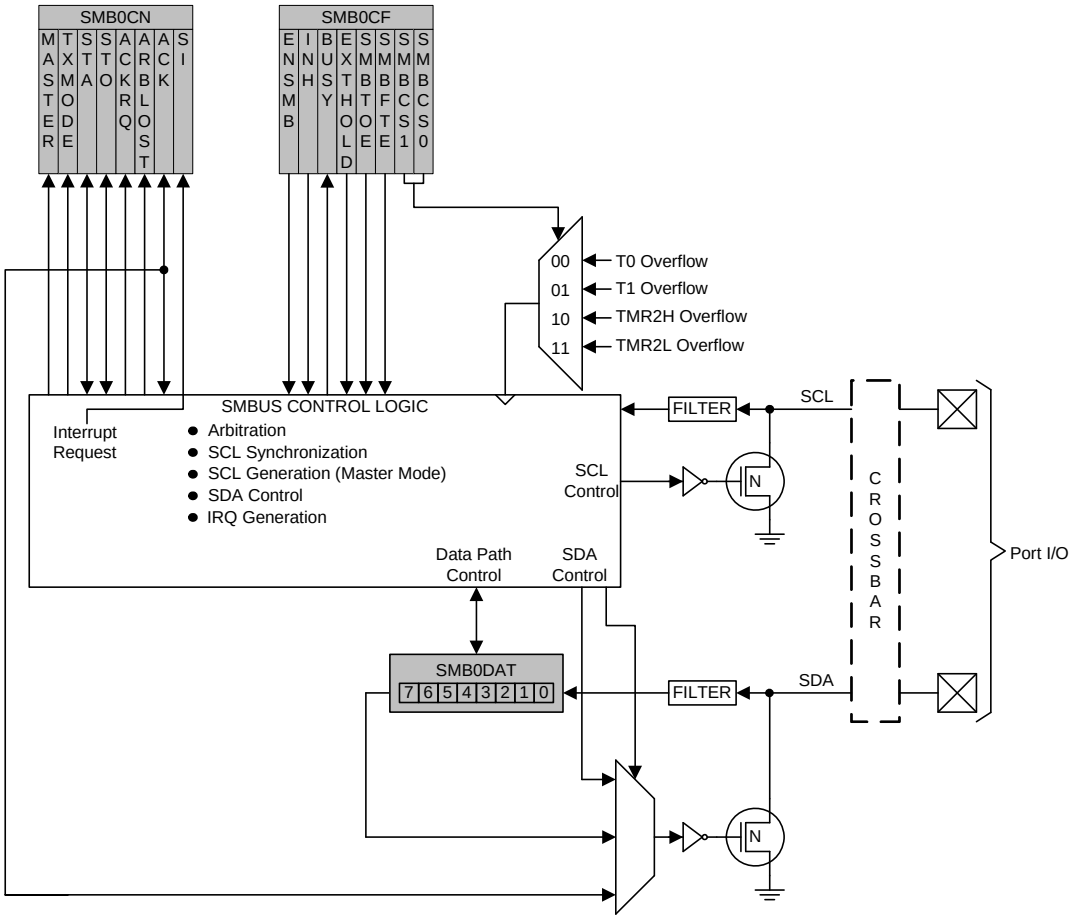

Figure 22.1. SMBus Block Diagram

22.1. Supporting Documents

It is assumed the reader is familiar with or has access to the following supporting documents:

1. The I^2C-Bus and How to Use It (including specifications), Philips Semiconductor.
2. The I^2C-Bus Specification—Version 2.0, Philips Semiconductor.
3. System Management Bus Specification—Version 1.1, SBS Implementers Forum.

22.2. SMBus Configuration

Figure 22.2 shows a typical SMBus configuration. The SMBus specification allows any recessive voltage between 3.0 V and 5.0 V; different devices on the bus may operate at different voltage levels. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pullup resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 300 ns and 1000 ns, respectively.

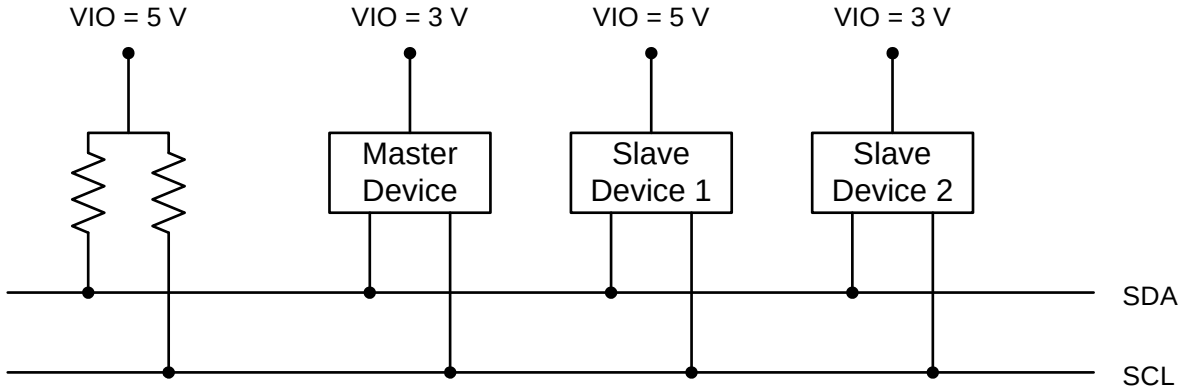

Figure 22.2. Typical SMBus Configuration

22.3. SMBus Operation

Two types of data transfers are possible: data transfers from a master transmitter to an addressed slave receiver (WRITE), and data transfers from an addressed slave transmitter to a master receiver (READ). The master device initiates both types of data transfers and provides the serial clock pulses on SCL. The SMBus interface may operate as a master or a slave, and multiple master devices on the same bus are supported. If two or more masters attempt to initiate a data transfer simultaneously, an arbitration scheme is employed with a single master always winning the arbitration. It is not necessary to specify one device as the Master in a system; any device who transmits a START and a slave address becomes the master for the duration of that transfer.

A typical SMBus transaction consists of a START condition followed by an address byte (Bits7–1: 7-bit slave address; Bit0: R/W direction bit), one or more bytes of data, and a STOP condition. Bytes that are received (by a master or slave) are acknowledged (ACK) with a low SDA during a high SCL (see Figure 22.3). If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

The direction bit (R/W) occupies the least-significant bit position of the address byte. The direction bit is set to logic 1 to indicate a "READ" operation and cleared to logic 0 to indicate a "WRITE" operation.

SFR Definition 24.1. SPI0CFG: SPI0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	SPIBSY	MSTEN	CKPHA	CKPOL	SLVSEL	NSSIN	SRMT	RXBMT
Type	R	R/W	R/W	R/W	R	R	R	R
Reset	0	0	0	0	0	1	1	1

SFR Address = 0xA1; SFR Page = 0x00

Bit	Name	Function
7	SPIBSY	**SPI Busy.** This bit is set to logic 1 when a SPI transfer is in progress (master or slave mode).
6	MSTEN	**Master Mode Enable.** 0: Disable master mode. Operate in slave mode. 1: Enable master mode. Operate as a master.
5	CKPHA	**SPI0 Clock Phase.** 0: Data centered on first edge of SCK period.* 1: Data centered on second edge of SCK period.*
4	CKPOL	**SPI0 Clock Polarity.** 0: SCK line low in idle state. 1: SCK line high in idle state.
3	SLVSEL	**Slave Selected Flag.** This bit is set to logic 1 whenever the NSS pin is low indicating SPI0 is the selected slave. It is cleared to logic 0 when NSS is high (slave not selected). This bit does not indicate the instantaneous value at the NSS pin, but rather a de-glitched version of the pin input.
2	NSSIN	**NSS Instantaneous Pin Input.** This bit mimics the instantaneous value that is present on the NSS port pin at the time that the register is read. This input is not de-glitched.
1	SRMT	**Shift Register Empty (valid in slave mode only).** This bit will be set to logic 1 when all data has been transferred in/out of the shift register, and there is no new information available to read from the transmit buffer or write to the receive buffer. It returns to logic 0 when a data byte is transferred to the shift register from the transmit buffer or by a transition on SCK. SRMT = 1 when in Master Mode.
0	RXBMT	**Receive Buffer Empty (valid in slave mode only).** This bit will be set to logic 1 when the receive buffer has been read and contains no new information. If there is new information available in the receive buffer that has not been read, this bit will return to logic 0. RXBMT = 1 when in Master Mode.

Note: In slave mode, data on MOSI is sampled in the center of each data bit. In master mode, data on MISO is sampled one SYSCLK before the end of each data bit, to provide maximum settling time for the slave device. See Table 24.1 for timing parameters.

SFR Definition 25.4. 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 25.5. 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.

26.5. Register Descriptions for PCA0

Following are detailed descriptions of the special function registers related to the operation of the PCA.

SFR Definition 26.1. PCA0CN: PCA Control

Bit	7	6	5	4	3	2	1	0
Name	CF	CR	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
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 = 0xD8; Bit-Addressable; SFR Page = 0x00

Bit	Name	Function
7	CF	**PCA Counter/Timer Overflow Flag.** Set by hardware when the PCA Counter/Timer overflows from 0xFFFF to 0x0000. When the Counter/Timer Overflow (CF) interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
6	CR	**PCA Counter/Timer Run Control.** This bit enables/disables the PCA Counter/Timer. 0: PCA Counter/Timer disabled. 1: PCA Counter/Timer enabled.
5	CCF5	**PCA Module 5 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF5 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
4	CCF4	**PCA Module 4 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF4 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
3	CCF3	**PCA Module 3 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF3 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
2	CCF2	**PCA Module 2 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF2 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
1	CCF1	**PCA Module 1 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF1 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
0	CCF0	**PCA Module 0 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF0 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.

27. C2 Interface

C8051F55x/56x/57x devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow Flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. See the C2 Interface Specification for details on the C2 protocol.

27.1. C2 Interface Registers

The following describes the C2 registers necessary to perform Flash programming through the C2 interface. All C2 registers are accessed through the C2 interface as described in the C2 Interface Specification.

C2 Register Definition 27.1. C2ADD: C2 Address

Bit	7	6	5	4	3	2	1	0
Name	C2ADD[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function	
7:0	C2ADD[7:0]	**C2 Address.** The C2ADD register is accessed via the C2 interface to select the target Data register for C2 Data Read and Data Write commands.	
		Address	**Description**
		0x00	Selects the Device ID register for Data Read instructions
		0x01	Selects the Revision ID register for Data Read instructions
		0x02	Selects the C2 Flash Programming Control register for Data Read/Write instructions
		0xB4	Selects the C2 Flash Programming Data register for Data Read/Write instructions