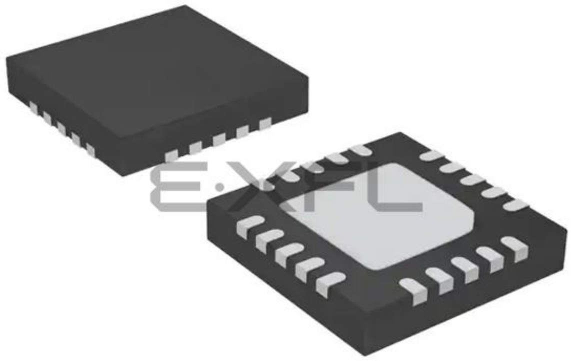

Welcome to E-XFL.COM

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	25MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	15
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.2V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	20-VFQFN Exposed Pad
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f854-c-im

2.5. Communications and other Digital Peripherals

2.5.1. Universal Asynchronous Receiver/Transmitter (UART0)

The UART uses two signals (TX and RX) and a predetermined fixed baud rate to provide asynchronous communications with other devices.

The UART module provides the following features:

- Asynchronous transmissions and receptions.
- Baud rates up to SYSCLK / 2 (transmit) or SYSCLK / 8 (receive).
- 8- or 9-bit data.
- Automatic start and stop generation.

2.5.2. Serial Peripheral Interface (SPI0)

SPI is a 3- or 4-wire communication interface that includes a clock, input data, output data, and an optional select signal.

The SPI module includes the following features:

- Supports 3- or 4-wire master or slave modes.
- Supports external clock frequencies up to SYSCLK / 2 in master mode and SYSCLK / 10 in slave mode.
- Support for all clock phase and polarity modes.
- 8-bit programmable clock rate.
- Support for multiple masters on the same data lines.

2.5.3. System Management Bus / I2C (SMBus0)

The SMBus interface is a two-wire, bi-directional serial bus compatible with both I2C and SMBus protocols. The two clock and data signals operate in open-drain mode with external pull-ups to support automatic bus arbitration.

Reads and writes to the interface are byte-oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/8th of the system clock as a master or slave, which can be faster than allowed by the SMBus / I2C specification, depending on the clock source 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.

The SMBus module includes the following features:

- Standard (up to 100 kbps) and Fast (400 kbps) transfer speeds.
- Support for master, slave, and multi-master modes.
- Hardware synchronization and arbitration for multi-master mode.
- Clock low extending (clock stretching) to interface with faster masters.
- Hardware support for 7-bit slave and general call address recognition.
- Firmware support for 10-bit slave address decoding.
- Ability to inhibit all slave states.
- Programmable data setup/hold times.

2.5.4. 16/32-bit CRC (CRC0)

The CRC module is designed to provide hardware calculations for flash memory verification and communications protocols. The CRC module supports the standard CCITT-16 16-bit polynomial (0x1021), and includes the following features:

- Support for four CCITT-16 polynomial.

2.7. Reset Sources

Reset circuitry allows the controller to be easily placed in a predefined default condition. On entry to this reset state, the following occur:

- The core halts program execution.
- Module registers are initialized to their defined reset values unless the bits reset only with a power-on reset.
- External port pins are forced to a known state.
- Interrupts and timers are disabled.

All registers are reset to the predefined values noted in the register descriptions unless the bits only reset with a power-on reset. The contents of RAM are unaffected during a reset; any previously stored data is preserved as long as power is not lost.

The Port I/O latches are reset to 1 in open-drain mode. Weak pullups are enabled during and after the reset. For VDD Supply Monitor and power-on resets, the $\overline{\text{RST}}$ pin is driven low until the device exits the reset state.

On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to the internal low-power oscillator. The Watchdog Timer is enabled with the Low Frequency Oscillator (LFO0) as its clock source. Program execution begins at location 0x0000.

2.8. On-Chip Debugging

The C8051F85x/86x 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.

10.2. Programming the Flash Memory

Writes to flash memory clear bits from logic 1 to logic 0, and can be performed on single byte locations. Flash erasures set bits back to logic 1, and occur only on full pages. The write and erase operations are automatically timed by hardware for proper execution; data polling to determine the end of the write/erase operation is not required. Code execution is stalled during a flash write/erase operation.

The simplest means of programming the flash memory is through the C2 interface using programming tools provided by Silicon Labs or a third party vendor. This is the only means for programming a non-initialized device.

To ensure the integrity of flash contents, it is strongly recommended that the on-chip supply monitor be enabled in any system that includes code that writes and/or erases flash memory from software.

10.2.1. Flash Lock and Key Functions

Flash writes and erases by user software are protected with a lock and key function. The Flash Lock and Key Register (FLKEY) must be written with the correct key codes, in sequence, before flash operations may be performed. The key codes are: 0xA5, 0xF1. The timing does not matter, but the codes must be written in order. If the key codes are written out of order, or the wrong codes are written, flash writes and erases will be disabled until the next system reset. Flash writes and erases will also be disabled if a flash write or erase is attempted before the key codes have been written properly. The flash lock resets after each write or erase; the key codes must be written again before a following flash operation can be performed.

10.2.2. Flash Erase Procedure

The flash memory can be programmed by software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before writing to flash memory using MOVX, flash write operations must be enabled by: (1) setting the PSWE Program Store Write Enable bit in the PSCTL register to logic 1 (this directs the MOVX writes to target flash memory); and (2) Writing the flash key codes in sequence to the Flash Lock register (FLKEY). The PSWE bit remains set until cleared by software.

A write to flash memory can clear bits to logic 0 but cannot set them; only an erase operation can set bits to logic 1 in flash. **A byte location to be programmed should be erased before a new value is written.** Erase operation applies to an entire page (setting all bytes in the page to 0xFF). To erase an entire page, perform the following steps:

1. Disable interrupts (recommended).
2. Set the PSEE bit (register PSCTL).
3. Set the PSWE bit (register PSCTL).
4. Write the first key code to FLKEY: 0xA5.
5. Write the second key code to FLKEY: 0xF1.
6. Using the MOVX instruction, write a data byte to any location within the page to be erased.
7. Clear the PSWE and PSEE bits.

10.2.3. Flash Write Procedure

Flash bytes are programmed by software with the following sequence:

1. Disable interrupts (recommended).
2. Erase the flash page containing the target location, as described in Section 10.2.2.
3. Set the PSWE bit (register PSCTL).
4. Clear the PSEE bit (register PSCTL).
5. Write the first key code to FLKEY: 0xA5.
6. Write the second key code to FLKEY: 0xF1.
7. Using the MOVX instruction, write a single data byte to the desired location within the desired

11. Device Identification and Unique Identifier

The C8051F85x/86x has SFRs that identify the device family, derivative, and revision. These SFRs can be read by firmware at runtime to determine the capabilities of the MCU that is executing code. This allows the same firmware image to run on MCUs with different memory sizes and peripherals, and dynamically change functionality to suit the capabilities of that MCU.

In addition to the device identification registers, a 32-bit unique identifier (UID) is pre-programmed into all Revision C and later devices. The UID resides in the last four bytes of XRAM (C8051F850/1/3/4 and C8051F860/1/3/4) or RAM (C8051F852/5 and C8051F862/5). For devices with the UID in RAM, the UID can be read by firmware using indirect data accesses. For devices with the UID in XRAM, the UID can be read by firmware using MOVX instructions. The UID can also be read through the debug port for all devices.

Firmware can overwrite the UID during normal operation, and the bytes in memory will be automatically reinitialized with the UID value after any device reset. Firmware using this area of memory should always initialize the memory to a known value, as any previous data stored at these locations will be overwritten and not retained through a reset.

Table 11.1. UID Implementation Information

Device	Memory Segment	Addresses
C8051F850 C8051F851 C8051F853 C8051F854 C8051F860 C8051F861 C8051F863 C8051F864	XRAM	**(MSB)** 0x00FF, 0x00FE, 0x00FD, 0x00FC **(LSB)**
C8051F852 C8051F855 C8051F862 C8051F865	RAM (indirect)	**(MSB)** 0xFF, 0xFE, 0xFD, 0xFC **(LSB)**

Register 11.3. REVID: Revision Identifcation

Bit	7	6	5	4	3	2	1	0
Name	REVID							
Type	R							
Reset	X	X	X	X	X	X	X	X
SFR Address: 0xB6								

Table 11.4. REVID Register Bit Descriptions

Bit	Name	Function
7:0	REVID	**Revision ID.** This read-only register returns the 8-bit revision ID. 00000000: Revision A 00000001: Revision B 00000010: Revision C 00000011-11111111: Reserved.

cycles to complete the DIV instruction and 4 clock cycles to execute the LCALL to the ISR. If the CPU is executing an ISR for an interrupt with equal or higher priority, the new interrupt will not be serviced until the current ISR completes, including the RETI and following instruction. If more than one interrupt is pending when the CPU exits an ISR, the CPU will service the next highest priority interrupt that is pending.

12.2. Interrupt Control Registers

Register 12.1. IE: Interrupt Enable

Bit	7	6	5	4	3	2	1	0
Name	EA	ESPI0	ET2	ES0	ET1	EX1	ET0	EX0
Type	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xA8 (bit-addressable)								

Table 12.2. IE Register Bit Descriptions

Bit	Name	Function
7	EA	**Enable All Interrupts.** Globally enables/disables all interrupts and overrides individual interrupt mask settings. 0: Disable all interrupt sources. 1: Enable each interrupt according to its individual mask setting.
6	ESPI0	**Enable SPI0 Interrupt.** This bit sets the masking of the SPI0 interrupts. 0: Disable all SPI0 interrupts. 1: Enable interrupt requests generated by SPI0.
5	ET2	**Enable Timer 2 Interrupt.** This bit sets the masking of the Timer 2 interrupt. 0: Disable Timer 2 interrupt. 1: Enable interrupt requests generated by the TF2L or TF2H flags.
4	ES0	**Enable UART0 Interrupt.** This bit sets the masking of the UART0 interrupt. 0: Disable UART0 interrupt. 1: Enable UART0 interrupt.
3	ET1	**Enable Timer 1 Interrupt.** This bit sets the masking of the Timer 1 interrupt. 0: Disable all Timer 1 interrupt. 1: Enable interrupt requests generated by the TF1 flag.
2	EX1	**Enable External Interrupt 1.** This bit sets the masking of External Interrupt 1. 0: Disable external interrupt 1. 1: Enable interrupt requests generated by the INT1 input.
1	ET0	**Enable Timer 0 Interrupt.** This bit sets the masking of the Timer 0 interrupt. 0: Disable all Timer 0 interrupt. 1: Enable interrupt requests generated by the TF0 flag.

Table 12.5. EIP1 Register Bit Descriptions

Bit	Name	Function
0	PSMB0	**SMBus (SMB0) Interrupt Priority Control.** This bit sets the priority of the SMB0 interrupt. 0: SMB0 interrupt set to low priority level. 1: SMB0 interrupt set to high priority level.

Register 15.2. DPH: Data Pointer High

Bit	7	6	5	4	3	2	1	0
Name	DPH							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Address: 0x83								

Table 15.3. DPH Register Bit Descriptions

Bit	Name	Function
7:0	DPH	**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.

Register 15.4. ACC: Accumulator

Bit	7	6	5	4	3	2	1	0
Name	ACC							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xE0 (bit-addressable)								

Table 15.5. ACC Register Bit Descriptions

Bit	Name	Function
7:0	ACC	**Accumulator.** This register is the accumulator for arithmetic operations.

17.3. Comparator Control Registers

Register 17.1. CPT0CN: Comparator 0 Control

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CPEN	CPOUT	CPRIF	CPFIF	CPHYP		CPHYN	
Type	RW	R	RW	RW	RW		RW	
Reset	0	0	0	0	0	0	0	0
SFR Address: 0x9B								

Table 17.5. CPT0CN Register Bit Descriptions

Bit	**Name**	**Function**
7	CPEN	**Comparator 0 Enable Bit.** 0: Comparator Disabled. 1: Comparator Enabled.
6	CPOUT	**Comparator 0 Output State Flag.** 0: Voltage on CP0P < CP0N. 1: Voltage on CP0P > CP0N.
5	CPRIF	**Comparator 0 Rising-Edge Flag. Must be cleared by software.** 0: No Comparator Rising Edge has occurred since this flag was last cleared. 1: Comparator Rising Edge has occurred.
4	CPFIF	**Comparator 0 Falling-Edge Flag. Must be cleared by software.** 0: No Comparator Falling-Edge has occurred since this flag was last cleared. 1: Comparator Falling-Edge has occurred.
3:2	CPHYP	**Comparator 0 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	CPHYN	**Comparator 0 Negative Hysteresis Control Bits.** 00: Negative Hysteresis Disabled. 01: Negative Hysteresis = 5 mV. 10: Negative Hysteresis = 10 mV. 11: Negative Hysteresis = 20 mV.

20.4.4. 16-Bit Pulse Width Modulator Mode

A PCA module may also be operated in 16-Bit PWM mode. 16-bit PWM mode is independent of the other (8 through 11-bit) PWM modes. The entire PCA0CP register is used to determine the duty cycle in 16-bit PWM mode.

To output a varying duty cycle, new value writes should be synchronized with the PCA CCFn match flag to ensure seamless updates.

16-Bit PWM mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, the match interrupt flag should be enabled (ECCFn = 1 AND MATn = 1) to help synchronize the capture/compare register writes. If the MATn bit is set to 1, the CCFn flag for the module will be set each time a match edge or up edge occurs. The CF flag in PCA0CN can be used to detect the overflow or down edge.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

Register 20.2. PCA0MD: PCA Mode

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CIDL	Reserved			CPS			ECF
Type	RW	R			RW			RW
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xD9								

Table 20.4. PCA0MD Register Bit Descriptions

Bit	Name	Function
7	CIDL	**PCA Counter/Timer Idle Control.** Specifies PCA behavior when CPU is in Idle Mode. 0: PCA continues to function normally while the system controller is in Idle Mode. 1: PCA operation is suspended while the system controller is in Idle Mode.
6:4	Reserved	Must write reset value.
3:1	CPS	**PCA Counter/Timer Pulse Select.** These bits select the timebase source for the PCA counter. 000: System clock divided by 12. 001: System clock divided by 4. 010: Timer 0 overflow. 011: High-to-low transitions on ECI (max rate = system clock divided by 4). 100: System clock. 101: External clock divided by 8 (synchronized with the system clock). 110: Low frequency oscillator divided by 8. 111: Reserved.
0	ECF	**PCA Counter/Timer Overflow Interrupt Enable.** This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt. 0: Disable the CF interrupt. 1: Enable a PCA Counter/Timer Overflow interrupt request when CF (PCA0CN.7) is set.

21. Port I/O (Port 0, Port 1, Port 2, Crossbar, and Port Match)

Digital and analog resources on the C8051F85x/86x family are externally available on the device's multi-purpose I/O pins. Port pins P0.0-P1.7 can be defined as general-purpose I/O (GPIO), assigned to one of the internal digital resources through the crossbar, or assigned to an analog function. Port pins P2.0 and P2.1 can be used as GPIO. Port pin P2.0 is shared with the C2 Interface Data signal (C2D). The designer has complete control over which functions are assigned, limited only by the number of physical I/O pins. This resource assignment flexibility is achieved through the use of a priority crossbar decoder. Note that the state of a port I/O pin can always be read in the corresponding port latch, regardless of the crossbar settings.

The crossbar assigns the selected internal digital resources to the I/O pins based on the Priority Decoder (Figure 21.2 and Figure 21.3). The registers XBR0, XBR1 and XBR2 are used to select internal digital functions.

The port I/O cells are configured as either push-pull or open-drain in the Port Output Mode registers (PnMDOUT, where n = 0,1). Additionally, each bank of port pins (P0, P1, and P2) has two selectable drive strength settings.

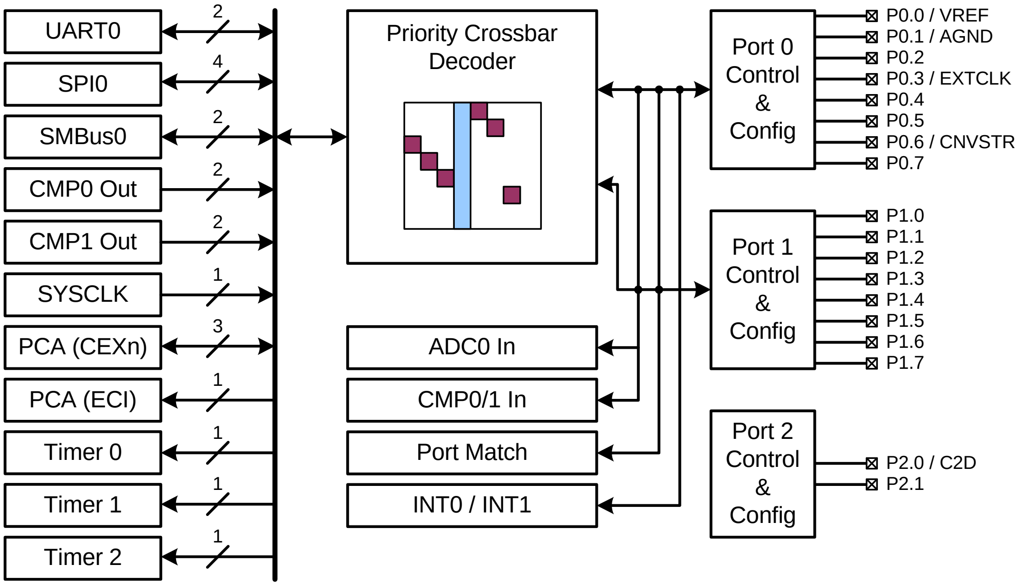

Figure 21.1. Port I/O Functional Block Diagram

Register 21.9. P0MDOUT: Port 0 Output Mode

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	P0MDOUT							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xA4								

Table 21.12. P0MDOUT Register Bit Descriptions

Bit	**Name**	**Function**
7:0	P0MDOUT	**Port 0 Output Mode.** These bits are only applicable when the pin is configured for digital mode using the P0MDIN register. 0: Corresponding P0.n Output is open-drain. 1: Corresponding P0.n Output is push-pull.

Register 21.12. P1MAT: Port 1 Match

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	P1MAT							
Type	RW							
Reset	1	1	1	1	1	1	1	1
SFR Address: 0xED								

Table 21.15. P1MAT Register Bit Descriptions

Bit	Name	Function
7:0	P1MAT	**Port 1 Match Value.** Match comparison value used on P1 pins for bits in P1MASK which are set to 1. 0: P1.x pin logic value is compared with logic LOW. 1: P1.x pin logic value is compared with logic HIGH.
Note: Port 1 consists of 8 bits (P1.0-P1.7) on QSOP24 packages and 7 bits (P1.0-P1.6) on QFN20 packages and 4 bits (P1.0-P1.3) on SOIC16 packages.		

Table 23.2. SPI0CFG Register Bit Descriptions

Bit	Name	Function
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.

24.1. Supporting Documents

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

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

24.2. SMBus Configuration

Figure 24.2 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. However, the maximum voltage on any port pin must conform to the electrical characteristics specifications. 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.

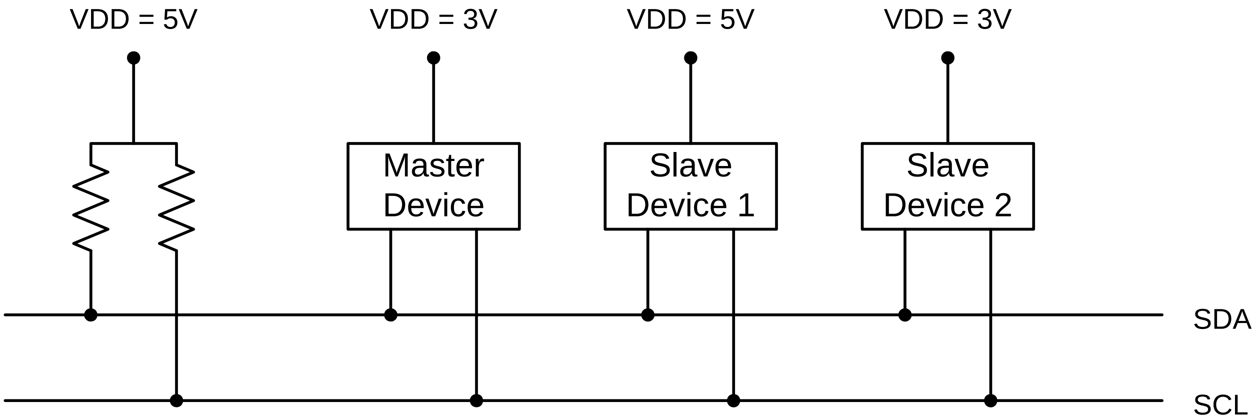

Figure 24.2. Typical SMBus Configuration

24.3. 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 24.3). 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.

27.5. Watchdog Timer Control Registers

Register 27.1. WDTCN: Watchdog Timer Control

Bit	7	6	5	4	3	2	1	0
Name	WDTCN							
Type	RW							
Reset	0	0	0	1	0	1	1	1
SFR Address: 0x97								

Table 27.1. WDTCN Register Bit Descriptions

Bit	Name	Function
7:0	WDTCN	**WDT Control.** The WDT control field has different behavior for reads and writes. Read: When reading the WDTCN register, the lower three bits (WDTCN[2:0]) indicate the current timeout interval. Bit WDTCN.4 indicates whether the WDT is active (logic 1) or inactive (logic 0). Write: Writing the WDTCN register can set the timeout interval, enable the WDT, disable the WDT, reset the WDT, or lock the WDT to prevent disabling. Writing to WDTCN with the MSB (WDTCN.7) cleared to 0 will set the timeout interval to the value in bits WDTCN[2:0]. Writing 0xA5 both enables and reloads the WDT. Writing 0xDE followed within 4 system clocks by 0xAD disables the WDT. Writing 0xFF locks out the disable feature until the next device reset.

28. Revision-Specific Behavior

C8051F85x/86x Revision B devices have differences from Revision C devices:

- Temperature Sensor offset and slope
- Flash endurance
- Latch-up performance
- Unique Identifier

28.1. Revision Identification

The Lot ID Code on the top side of the device package can be used for decoding device revision information. Figure 28.1, Figure 28.2, and Figure 28.3 show how to find the Lot ID Code on the top side of the device package.

Firmware can distinguish between a Revision B and Revision C device using the value of the REVID register described in “Device Identification and Unique Identifier” on page 68.

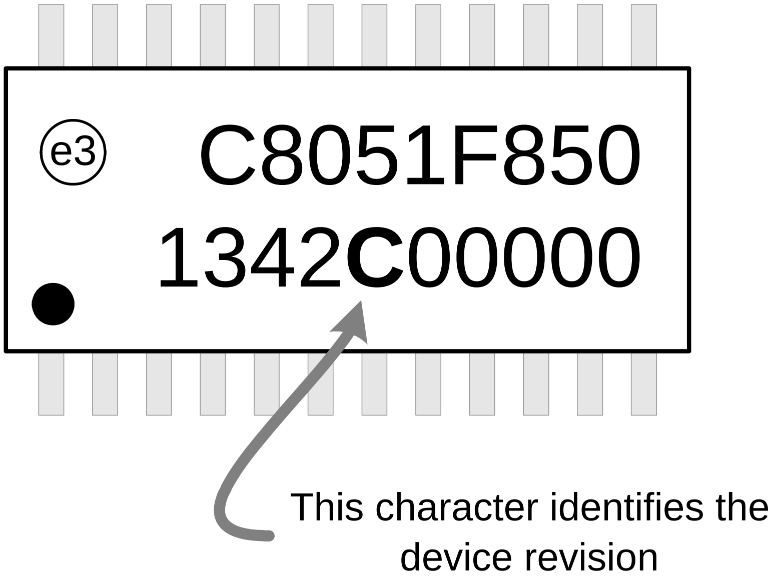

Figure 28.1. QSOP-24 Package Revision Marking