E·XFL



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 "<u>Embedded -</u> <u>Microcontrollers</u>"

Details

Product Status	Active
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	EBI/EMI, SMBus (2-Wire/I ² C), CANbus, LINbus, SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	40
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4.25K 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	48-TQFP
Supplier Device Package	48-TQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f504-iq

Email: info@E-XFL.COM

Address: Room A, 16/F, Full Win Commercial Centre, 573 Nathan Road, Mongkok, Hong Kong

Name	Pin 'F500/1/4/5 (48-pin)	Pin F508/9- F510/1 (40-pin)	Pin 'F502/3/6/7 (32-pin)	Туре	Description
P0.6	44	36	28	D I/O or A In	Port 0.6
P0.7	43	35	27	D I/O or A In	Port 0.7
P1.0	42	34	26	D I/O or A In	Port 1.0. See SFR Definition 20.16 for a description.
P1.1	41	33	25	D I/O or A In	Port 1.1.
P1.2	40	32	24	D I/O or A In	Port 1.2.
P1.3	39	31	23	D I/O or A In	Port 1.3.
P1.4	38	30	22	D I/O or A In	Port 1.4.
P1.5	37	29	21	D I/O or A In	Port 1.5.
P1.6	36	28	20	D I/O or A In	Port 1.6.
P1.7	35	27	19	D I/O or A In	Port 1.7.
P2.0	34	26	18	D I/O or A In	Port 2.0. See SFR Definition 20.20 for a description.
P2.1	33	25	17	D I/O or A In	Port 2.1.
P2.2	32	24	16	D I/O or A In	Port 2.2.
P2.3	31	23	15	D I/O or A In	Port 2.3.
P2.4	30	22	14	D I/O or A In	Port 2.4.
P2.5	29	21	13	D I/O or A In	Port 2.5.
P2.6	28	20	12	D I/O or A In	Port 2.6.
P2.7	27	19	11	D I/O or A In	Port 2.7.
P3.0	26	18	—	D I/O or A In	Port 3.0. See SFR Definition 20.24 for a description.
P3.1	25	17	—	D I/O or A In	Port 3.1.
P3.2	24	16	—	D I/O or A In	Port 3.2.
P3.3	23	15	—	D I/O or A In	Port 3.3.
P3.4	22	14	—	D I/O or A In	Port 3.4.
P3.5	21	13	—	D I/O or A In	Port 3.5.
P3.6	20	12	—	D I/O or A In	Port 3.6.

Table 3.1. Pin Definitions for the C8051F50x/F51x(Continued)



been accumulated. Similarly, the Window Comparator will not compare the result to the greater-than and less-than registers until "repeat count" conversions have been accumulated.

Note: When using Burst Mode, care must be taken to issue a convert start signal no faster than once every four SYSCLK periods. This includes external convert start signals.

System Clock							
Convert Start (AD0BUSY or Timer Overflow)	>	[
Post-Tracking AD0TM = 01 AD0EN = 0	Powered Down	Power-Up and Idle	т с т с	т с т с	Powered Down	Pov an	ver-Up d Idle T C
Dual-Tracking AD0TM = 11 AD0EN = 0	Powered Down	Power-Up and Track	тстс	тстс	Powered Down	Pov and	ver-Up Track T C
		AD0PWR ⁺	•				
Post-Tracking AD0TM = 01 AD0EN = 1	Idle	тстс	тстс	I	dle	тс	T C T C
Dual-Tracking AD0TM = 11 AD0EN = 1	Track	тстс	т с т с	Т	rack	тс	T C T C
	T = Tracking C = Converti	ing					
Convert Start (CNVSTR)		[
Post-Tracking AD0TM = 01 AD0EN = 0	Powered Down	Power-Up and Idle	TC	Power	ed า	Pov an	ver-Up d Idle T C
Dual-Tracking AD0TM = 11 AD0EN = 0	Powered Down	Power-Up and Track	тс	Power Dowr	ed า	Pov and	ver-Up Track T C
		AD0PWR ⁺	•				
Post-Tracking AD0TM = 01 AD0EN = 1	Idle	тс		Idle		тс	Idle
Dual-Tracking AD0TM = 11 AD0EN = 1	Track	ТС		Track		тс	Track
	T = Tracking C = Converti	ina					





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

Mnemonic	Description	Bytes	Clock Cycles
Arithmetic Operations	1		
ADD A, Rn	Add register to A	1	1
ADD A, direct	Add direct byte to A	2	2
ADD A, @Ri	Add indirect RAM to A	1	2
ADD A, #data	Add immediate to A	2	2
ADDC A, Rn	Add register to A with carry	1	1
ADDC A, direct	Add direct byte to A with carry	2	2
ADDC A, @Ri	Add indirect RAM to A with carry	1	2
ADDC A, #data	Add immediate to A with carry	2	2
SUBB A, Rn	Subtract register from A with borrow	1	1
SUBB A, direct	Subtract direct byte from A with borrow	2	2
SUBB A, @Ri	Subtract indirect RAM from A with borrow	1	2
SUBB A, #data	Subtract immediate from A with borrow	2	2
INC A	Increment A	1	1
INC Rn	Increment register	1	1
INC direct	Increment direct byte	2	2
INC @Ri	Increment indirect RAM	1	2
DEC A	Decrement A	1	1
DEC Rn	Decrement register	1	1
DEC direct	Decrement direct byte	2	2
DEC @Ri	Decrement indirect RAM	1	2
INC DPTR	Increment Data Pointer	1	1
MUL AB	Multiply A and B	1	4
DIV AB	Divide A by B	1	8
DA A	Decimal adjust A	1	1
Logical Operations	•		
ANL A, Rn	AND Register to A	1	1
ANL A, direct	AND direct byte to A	2	2
ANL A, @Ri	AND indirect RAM to A	1	2
ANL A, #data	AND immediate to A	2	2
ANL direct, A	AND A to direct byte	2	2
ANL direct, #data	AND immediate to direct byte	3	3
ORL A, Rn	OR Register to A	1	1
ORL A, direct	OR direct byte to A	2	2
ORL A, @Ri	OR indirect RAM to A	1	2
ORL A, #data	OR immediate to A	2	2
ORL direct, A	OR A to direct byte	2	2
ORL direct, #data	OR immediate to direct byte	3	3
XRL A, Rn	Exclusive-OR Register to A	1	1
XRL A, direct	Exclusive-OR direct byte to A	2	2
XRL A, @Ri	Exclusive-OR indirect RAM to A	1	2
Note: Certain instructions tak the FLRT setting (SFR	e a variable number of clock cycles to execute depending Definition 15.3).	g on instruction a	lignment and



12.1. Program Memory

The CIP-51 core has a 64 kB program memory space. The C8051F50x/F51x devices implement 64 kB or 32 kB of this program memory space as in-system, re-programmable Flash memory, organized in a contiguous block from addresses 0x0000 to 0xFFFF in 64 kB devices and addresses 0x0000 to 0x7FFF in 32 kB devices. The address 0xFBFF in 64 kB devices and 0x7FFF in 32 kB devices serves as the security lock byte for the device. Addresses above 0xFDFF are reserved in the 64 kB devices.



Figure 12.2. Flash Program Memory Map

12.1.1. MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the C8051F50x/F51x devices, the MOVX instruction is normally used to read and write on-chip XRAM, but can be re-configured to write and erase on-chip Flash memory space. MOVC instructions are always used to read Flash memory, while MOVX write instructions are used to erase and write Flash. This Flash access feature provides a mechanism for the C8051F50x/F51x to update program code and use the program memory space for non-volatile data storage. Refer to Section "15. Flash Memory" on page 129 for further details.

12.2. Data Memory

The C8051F50x/F51x devices include 4352 bytes of RAM data memory. 256 bytes of this memory is mapped into the internal RAM space of the 8051. The other 4096 bytes of this memory is on-chip "external" memory. The data memory map is shown in Figure 12.1 for reference.

12.2.1. Internal RAM

There are 256 bytes of internal RAM mapped into the data memory space from 0x00 through 0xFF. The lower 128 bytes of data memory are used for general purpose registers and scratch pad memory. Either direct or indirect addressing may be used to access the lower 128 bytes of data memory. Locations 0x00 through 0x1F are addressable as four banks of general purpose registers, each bank consisting of eight





Figure 13.1. SFR Page Stack

Automatic hardware switching of the SFR Page on interrupts may be enabled or disabled as desired using the SFR Automatic Page Control Enable Bit located in the SFR Page Control Register (SFR0CN). This function defaults to "enabled" upon reset. In this way, the autoswitching function will be enabled unless disabled in software.

A summary of the SFR locations (address and SFR page) are provided in Table 13.3 in the form of an SFR memory map. Each memory location in the map has an SFR page row, denoting the page in which that SFR resides. Certain SFRs are accessible from ALL SFR pages, and are denoted by the "(ALL PAGES)" designation. For example, the Port I/O registers P0, P1, P2, and P3 all have the "(ALL PAGES)" designation, indicating these SFRs are accessible from all SFR pages regardless of the SFRPAGE register value.

13.3. SFR Page Stack Example

The following is an example that shows the operation of the SFR Page Stack during interrupts. In this example, the SFR Control register is left in the default enabled state (i.e., SFRPGEN = 1), and the CIP-51 is executing in-line code that is writing values to SPI Data Register (SFR "SPI0DAT", located at address 0xA3 on SFR Page 0x00). The device is also using the CAN peripheral (CAN0) and the Programmable Counter Array (PCA0) peripheral to generate a PWM output. The PCA is timing a critical control function in its interrupt service round so its associated ISR that is set to low priority. At this point, the SFR page is set to access the SPI0DAT SFR (SFRPAGE = 0x00). See Figure 13.2.





Figure 13.2. SFR Page Stack While Using SFR Page 0x0 To Access SPI0DAT

While CIP-51 executes in-line code (writing values to SPI0DAT in this example), the CAN0 Interrupt occurs. The CIP-51 vectors to the CAN0 ISR and pushes the current SFR Page value (SFR Page 0x00) into SFRNEXT in the SFR Page Stack. The SFR page needed to access CAN's SFRs is then automatically placed in the SFRPAGE register (SFR Page 0x0C). SFRPAGE is considered the "top" of the SFR Page Stack. Software can now access the CAN0 SFRs. Software may switch to any SFR Page by writing a new value to the SFRPAGE register at any time during the CAN0 ISR to access SFRs that are not on SFR Page 0x0C. See Figure 13.3.





Figure 13.3. SFR Page Stack After CAN0 Interrupt Occurs

While in the CAN0 ISR, a PCA interrupt occurs. Recall the PCA interrupt is configured as a high priority interrupt, while the CAN0 interrupt is configured as a *low* priority interrupt. Thus, the CIP-51 will now vector to the high priority PCA ISR. Upon doing so, the CIP-51 will automatically place the SFR page needed to access the PCA's special function registers into the SFRPAGE register, SFR Page 0x00. The value that was in the SFRPAGE register before the PCA interrupt (SFR Page 0x0C for CAN0) is pushed down the stack into SFRNEXT. Likewise, the value that was in the SFRNEXT register before the PCA interrupt (in this case SFR Page 0x00 for SPI0DAT) is pushed down to the SFRLAST register, the "bottom" of the stack. Note that a value stored in SFRLAST (via a previous software write to the SFRLAST register) will be overwritten. See Figure 13.4.



				-				
	0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)
F8			CAN0IF2DA2L	CAN0IF2DA2H	CAN0IF2DB1L	CAN0IF2DB1H	CAN0IF2DB2L	CAN0IF2DB2H
F0	B (All Pages)		CAN0IF2A2L	CAN0IF2A2H			CAN0IF2DA1L	CAN0IF2DA1H
E8			CAN0IF2M1L	CAN0IF2M1H	CAN0IF2M2L	CAN0IF2M2H	CAN0IF2A1L	CAN0IF2A1H
E0	ACC (All Pages)		CAN0IF2CML	CAN0IF2CMH			EIE1 (All Pages)	EIE2 (All Pages)
D8			CAN0IF1DB1L	CAN0IF1DB1H	CAN0IF1DB2L	CAN0IF1DB2H	CAN0IF2CRL	CAN0IF2CRH
D0	PSW (All Pages)		CAN0IF1MCL	CAN0IF1MCH	CAN0IF1DA1L	CAN0IF1DA1H	CAN0IF1DA2L	CAN0IF1DA2H
C8			CAN0IF1A1L	CAN0IF1A1H	CAN0IF1A2L	CAN0IF1A2H	CAN0IF2MCL	CAN0IF2MCH
C0	CAN0CN		CAN0IF1CML	CAN0IF1CMH	CAN0IF1M1L	CAN0IF1M1H	CAN0IF1M2L	CAN0IF1M2H
B8	IP (All Pages)		CAN0MV1L	CAN0MV1H	CAN0MV2L	CAN0MV2H	CAN0IF1CRL	CAN0IF1CRH
B0	P3 (All Pages)		CAN0IP2L	CAN0IP2H		P4 (All Pages)	FLSCL (All Pages)	FLKEY (All Pages)
A8	IE (All Pages)		CAN0ND1L	CAN0ND1H	CAN0ND2L	CAN0ND2H	CAN0IP1L	CAN0IP1H
A0	P2 (All Pages)	CAN0BRPE	CAN0TR1L	CAN0TR1H	CAN0TR2L	CAN0TR2H		SFRPAGE (All Pages)
98	SCON0 (All Pages)		CAN0BTL	CAN0BTH	CAN0IIDL	CANOIIDH	CAN0TST	
90	P1 (All Pages)		CAN0CFG		CAN0STAT		CAN0ERRL	CAN0ERRH
88	TCON (All Pages)	TMOD (All Pages)	TL0 (All Pages)	TL1 (All Pages)	TH0 (All Pages)	TH1 (All Pages)	CKCON (All Pages)	
80	P0 (All Pages)	SP (All Pages)	DPL (All Pages)	DPH (All Pages)		SFRNEXT (All Pages)	SFRLAST (All Pages)	PCON (All Pages)
	0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)
	(bit addres	sable)						

Table 13.2. Special Function Register (SFR) Memory Map for Page 0xC

(bi dressable)



15.4.3. System Clock

- 1. If operating from an external crystal, be advised that crystal performance is susceptible to electrical interference and is sensitive to layout and to changes in temperature. If the system is operating in an electrically noisy environment, use the internal oscillator or use an external CMOS clock.
- 2. If operating from the external oscillator, switch to the internal oscillator during Flash write or erase operations. The external oscillator can continue to run, and the CPU can switch back to the external oscillator after the Flash operation has completed.

Additional Flash recommendations and example code can be found in "AN201: Writing to Flash from Firm-ware" available from the Silicon Laboratories web site.

SFR Definition 15.1. PSCTL: Program Store R/W Control

Bit	7	6	5	4	3	2	1	0
Name							PSEE	PSWE
Туре	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8F; SFR Page = 0x00

Bit	Name	Function
7:2	Unused	Read = 000000b, Write = don't care.
1	PSEE	 Program Store Erase Enable. Setting this bit (in combination with PSWE) allows an entire page of Flash program memory to be erased. If this bit is logic 1 and Flash writes are enabled (PSWE is logic 1), a write to Flash memory using the MOVX instruction will erase the entire page that contains the location addressed by the MOVX instruction. The value of the data byte written does not matter. 0: Flash program memory erasure disabled. 1: Flash program memory erasure enabled.
0	PSWE	 Program Store Write Enable. Setting this bit allows writing a byte of data to the Flash program memory using the MOVX write instruction. The Flash location should be erased before writing data. 0: Writes to Flash program memory disabled. 1: Writes to Flash program memory enabled; the MOVX write instruction targets Flash memory.



SFR Definition 18.1. EMI0CN: External Memory Interface Control

0xFF: 0xFF00 to 0xFFFF

Bit	7	6	5	4	3	2	1	0		
Nam	e	- 1	PGSEL[7:0]							
Тур	e		R/W							
Rese	et 0	0	0 0 0 0 0 0 0							
SFR /	Address = 0x	AA; SFR Page	λ; SFR Page = 0x00							
Bit	Name				Function					
7:0	PGSEL[7:0]	XRAM Page The XRAM F address whe RAM. 0x00: 0x000 0x01: 0x010 0xFE: 0xFEC	Select Bits Page Select on using an 8 0 to 0x00FF 0 to 0x01FF	Bits provide β-bit MOVX c	the high byte command, ef	e of the 16-bi fectively sele	it external da acting a 256-	ata memory byte page of		



18.4. Multiplexed and Non-multiplexed Selection

The External Memory Interface is capable of acting in a Multiplexed mode or a Non-multiplexed mode, depending on the state of the EMD2 (EMI0CF.4) bit.

18.4.1. Multiplexed Configuration

In Multiplexed mode, the Data Bus and the lower 8-bits of the Address Bus share the same Port pins: AD[7:0]. In this mode, an external latch (74HC373 or equivalent logic gate) is used to hold the lower 8-bits of the RAM address. The external latch is controlled by the ALE (Address Latch Enable) signal, which is driven by the External Memory Interface logic. An example of a Multiplexed Configuration is shown in Figure 18.1.

In Multiplexed mode, the external MOVX operation can be broken into two phases delineated by the state of the ALE signal. During the first phase, ALE is high and the lower 8-bits of the Address Bus are presented to AD[7:0]. During this phase, the address latch is configured such that the Q outputs reflect the states of the 'D' inputs. When ALE falls, signaling the beginning of the second phase, the address latch outputs remain fixed and are no longer dependent on the latch inputs. Later in the second phase, the Data Bus controls the state of the AD[7:0] port at the time RD or WR is asserted.



See Section "18.6.2. Multiplexed Mode" on page 161 for more information.

Figure 18.1. Multiplexed Configuration Example





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

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

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

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

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

SFR Definition 20.12. P0: Port 0

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

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

Bit	Name	Description	Write	Read
7:0	P0[7:0]	Port 0 Data. Sets the Port latch logic value or reads the Port pin logic state in Port cells con- figured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P0.n Port pin is logic LOW. 1: P0.n Port pin is logic HIGH.



SFR Definition 20.17. P1MDIN: Port 1 Input Mode

Bit	7	6	5	4	3	2	1	0
Name		P1MDIN[7:0]						
Туре		R/W						
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF2; SFR Page = 0x0F

Bit	Name	Function
7:0	P1MDIN[7:0]	Analog Configuration Bits for P1.7–P1.0 (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 P1MDOUT register. 0: Corresponding P1.n pin is configured for analog mode. 1: Corresponding P1.n pin is not configured for analog mode.

SFR Definition 20.18. P1MDOUT: Port 1 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P1MDOUT[7:0]							
Туре	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA5; SFR Page = 0x0F

Bit	Name	Function
7:0	P1MDOUT[7:0]	Output Configuration Bits for P1.7–P1.0 (respectively).
		These bits are ignored if the corresponding bit in register P1MDIN is logic 0. 0: Corresponding P1.n Output is open-drain. 1: Corresponding P1.n Output is push-pull.



SFR Definition 20.21. P2MDIN: Port 2 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P2MDIN[7:0]							
Туре	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF3; SFR Page = 0x0F

Bit	Name	Function
7:0	P2MDIN[7:0]	Analog Configuration Bits for P2.7–P2.0 (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 P2MDOUT register. 0: Corresponding P2.n pin is configured for analog mode. 1: Corresponding P2.n pin is not configured for analog mode.

SFR Definition 20.22. P2MDOUT: Port 2 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P2MDOUT[7:0]							
Туре	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA6; SFR Page = 0x0F

Bit	Name	Function
7:0	P2MDOUT[7:0]	Output Configuration Bits for P2.7–P2.0 (respectively).
		These bits are ignored if the corresponding bit in register P2MDIN is logic 0. 0: Corresponding P2.n Output is open-drain. 1: Corresponding P2.n Output is push-pull.



22. Controller Area Network (CAN0)

Important Documentation Note: The Bosch CAN Controller is integrated in the C8051F500/2/4/6/8-F510 devices. This section of the data sheet gives a description of the CAN controller as an overview and offers a description of how the Silicon Labs CIP-51 MCU interfaces with the on-chip Bosch CAN controller. In order to use the CAN controller, refer to Bosch's C_CAN User's Manual as an accompanying manual to the Silicon Labs' data sheet.

The C8051F500/2/4/6/8-F510 devices feature a Control Area Network (CAN) controller that enables serial communication using the CAN protocol. Silicon Labs CAN facilitates communication on a CAN network in accordance with the Bosch specification 2.0A (basic CAN) and 2.0B (full CAN). The CAN controller consists of a CAN Core, Message RAM (separate from the CIP-51 RAM), a message handler state machine, and control registers. Silicon Labs CAN is a protocol controller and does not provide physical layer drivers (i.e., transceivers). Figure 22.1 shows an example typical configuration on a CAN bus.

Silicon Labs CAN operates at bit rates of up to 1 Mbit/second, though this can be limited by the physical layer chosen to transmit data on the CAN bus. The CAN processor has 32 Message Objects that can be configured to transmit or receive data. Incoming data, message objects and their identifier masks are stored in the CAN message RAM. All protocol functions for transmission of data and acceptance filtering is performed by the CAN controller and not by the CIP-51 MCU. In this way, minimal CPU bandwidth is needed to use CAN communication. The CIP-51 configures the CAN controller, accesses received data, and passes data for transmission via Special Function Registers (SFRs) in the CIP-51.



Figure 22.1. Typical CAN Bus Configuration



23. 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 23.1.



Figure 23.1. SMBus Block Diagram



All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 23.3 illustrates a typical SMBus transaction.



Figure 23.3. SMBus Transaction

23.3.1. Transmitter vs. Receiver

On the SMBus communications interface, a device is the "transmitter" when it is sending an address or data byte to another device on the bus. A device is a "receiver" when an address or data byte is being sent to it from another device on the bus. The transmitter controls the SDA line during the address or data byte. After each byte of address or data information is sent by the transmitter, the receiver sends an ACK or NACK bit during the ACK phase of the transfer, during which time the receiver controls the SDA line.

23.3.2. Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see Section "23.3.5. SCL High (SMBus Free) Timeout" on page 229). In the event that two or more devices attempt to begin a transfer at the same time, an arbitration scheme is employed to force one master to give up the bus. The master devices continue transmitting until one attempts a HIGH while the other transmits a LOW. Since the bus is open-drain, the bus will be pulled LOW. The master attempting the HIGH will detect a LOW SDA and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave and receives the rest of the transfer if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

23.3.3. Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to I²C, which allows devices with different speed capabilities to coexist on the bus. A clock-low extension is used during a transfer in order to allow slower slave devices to communicate with faster masters. The slave may temporarily hold the SCL line LOW to extend the clock low period, effectively decreasing the serial clock frequency.

23.3.4. SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a "timeout" condition. Devices that have detected the timeout condition must reset the communication no later than 10 ms after detecting the timeout condition.

When the SMBTOE bit in SMB0CF is set, Timer 3 is used to detect SCL low timeouts. Timer 3 is forced to reload when SCL is high, and allowed to count when SCL is low. With Timer 3 enabled and configured to



25.5. Serial Clock Phase and Polarity

Four combinations of serial clock phase and polarity can be selected using the clock control bits in the SPI0 Configuration Register (SPI0CFG). The CKPHA bit (SPI0CFG.5) selects one of two clock phases (edge used to latch the data). The CKPOL bit (SPI0CFG.4) selects between an active-high or active-low clock. Both master and slave devices must be configured to use the same clock phase and polarity. SPI0 should be disabled (by clearing the SPIEN bit, SPI0CN.0) when changing the clock phase or polarity. The clock and data line relationships for master mode are shown in Figure 25.5. For slave mode, the clock and data relationships are shown in Figure 25.6 and Figure 25.7. CKPHA must be set to 0 on both the master and slave SPI when communicating between two of the following devices: C8051F04x, C8051F06x, C8051F12x, C8051F31x, C8051F32x, and C8051F33x.

The SPI0 Clock Rate Register (SPI0CKR) as shown in SFR Definition 25.3 controls the master mode serial clock frequency. This register is ignored when operating in slave mode. When the SPI is configured as a master, the maximum data transfer rate (bits/sec) is one-half the system clock frequency or 12.5 MHz, whichever is slower. When the SPI is configured as a slave, the maximum data transfer rate (bits/sec) for full-duplex operation is 1/10 the system clock frequency, provided that the master issues SCK, NSS (in 4-wire slave mode), and the serial input data synchronously with the slave's system clock. If the master issues SCK, NSS, and the serial input data asynchronously, the maximum data transfer rate (bits/sec) must be less than 1/10 the system clock frequency. In the special case where the master only wants to transmit data to the slave and does not need to receive data from the slave (i.e. half-duplex operation), the SPI slave can receive data at a maximum data transfer rate (bits/sec) of 1/4 the system clock frequency. This is provided that the master issues SCK, NSS, and the serial input data transfer rate (bits/sec) of 1/4 the system clock frequency. This is provided that the master issues SCK, NSS, and the serial input data synchronously with the slave's system clock frequency.



Figure 25.5. Master Mode Data/Clock Timing



SFR Definition 26.16. TMR3L: Timer 3 Low Byte

Bit	7	6	5	4	3	2	1	0
Name		TMR3L[7:0]						
Туре		R/W						
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x94; SFR Page = 0x00

Bit	Name	Function
7:0	TMR3L[7:0]	Timer 3 Low Byte.
		In 16-bit mode, the TMR3L register contains the low byte of the 16-bit Timer 3. In 8- bit mode, TMR3L contains the 8-bit low byte timer value.

SFR Definition 26.17. TMR3H Timer 3 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3H[7:0]							
Туре	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x95; SFR Page = 0x00

Bit	Name	Function
7:0	TMR3H[7:0]	Timer 3 High Byte.
		In 16-bit mode, the TMR3H register contains the high byte of the 16-bit Timer 3. In 8- bit mode, TMR3H contains the 8-bit high byte timer value.

