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	Obsolete
Core Processor	80C51
Core Size	8-Bit
Speed	40MHz
Connectivity	CANbus, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	36
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	2K x 8
RAM Size	2.25K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	44-VQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at89c51cc03u-rlrim

- **On-chip Emulation Logic (Enhanced Hook System)**
- **Power Saving Modes**
  - **Idle Mode**
  - **Power-down Mode**
- **Power Supply: 3 volts to 5.5 volts**
- **Temperature Range: Industrial (-40° to +85°C), Automotive (-40°C to +125°C)**
- **Packages: VQFP44, PLCC44, VQFP64, PLCC52**

Description

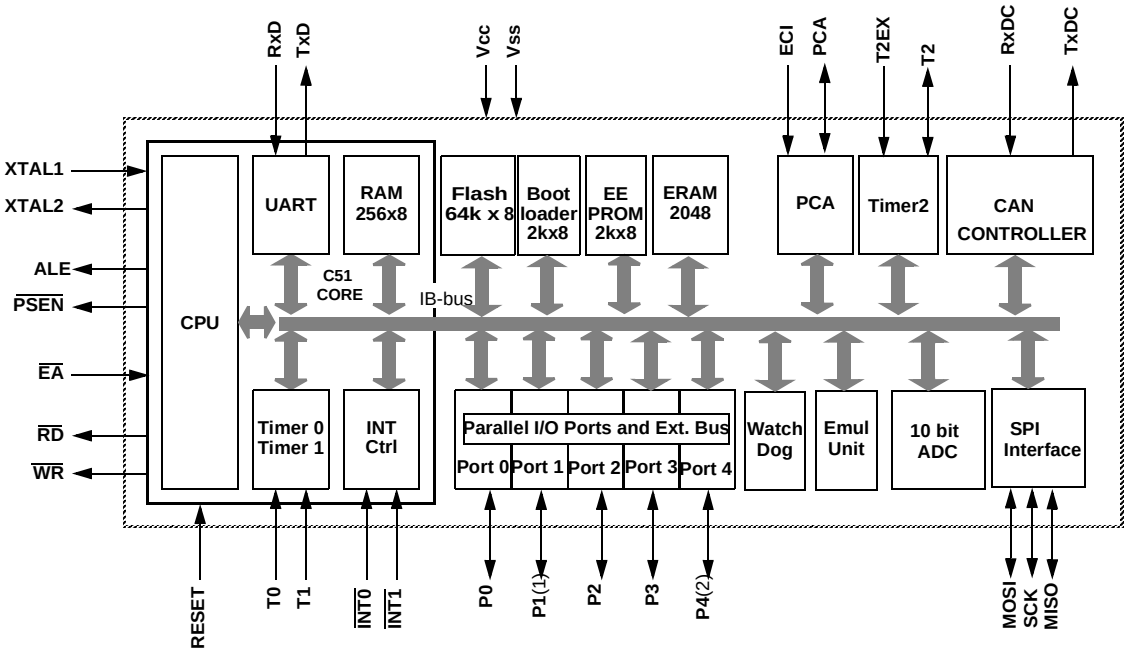
The AT89C51CC03 is a member of the family of 8-bit microcontrollers dedicated to CAN network applications.

In X2 mode a maximum external clock rate of 20 MHz reaches a 300 ns cycle time.

Besides the full CAN controller AT89C51CC03 provides 64K Bytes of Flash memory including In-System Programming (ISP), 2K Bytes Boot Flash Memory, 2K Bytes EEPROM and 2048 byte ERAM.

Primary attention is paid to the reduction of the electro-magnetic emission of AT89C51CC03.

Block Diagram



Notes: 1. 8 analog Inputs/8 Digital I/O
2. 5-Bit I/O Port

SFR Mapping

The Special Function Registers (SFRs) of the AT89C51CC03 fall into the following categories:

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
ACC	E0h	Accumulator	–	–	–	–	–	–	–	–
B	F0h	B Register	–	–	–	–	–	–	–	–
PSW	D0h	Program Status Word	CY	AC	F0	RS1	RS0	OV	F1	P
SP	81h	Stack Pointer	–	–	–	–	–	–	–	–
DPL	82h	Data Pointer Low byte LSB of DPTR	–	–	–	–	–	–	–	–
DPH	83h	Data Pointer High byte MSB of DPTR	–	–	–	–	–	–	–	–

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
P0	80h	Port 0	–	–	–	–	–	–	–	–
P1	90h	Port 1	–	–	–	–	–	–	–	–
P2	A0h	Port 2	–	–	–	–	–	–	–	–
P3	B0h	Port 3	–	–	–	–	–	–	–	–
P4	C0h	Port 4 (x5)	–	–	–	P4.4 / MOSI	P4.3 / SCK	P4.2 / MISO	P4.1 / RxDC	P4.0 / TxDC

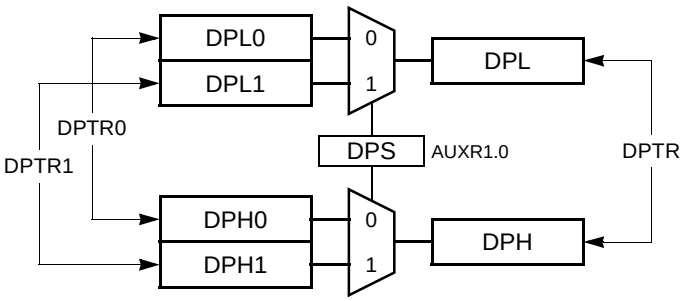
Mnemonic	Add	Name	7	6	5	4	3	2	1	0
TH0	8Ch	Timer/Counter 0 High byte	–	–	–	–	–	–	–	–
TL0	8Ah	Timer/Counter 0 Low byte	–	–	–	–	–	–	–	–
TH1	8Dh	Timer/Counter 1 High byte	–	–	–	–	–	–	–	–
TL1	8Bh	Timer/Counter 1 Low byte	–	–	–	–	–	–	–	–
TH2	CDh	Timer/Counter 2 High byte	–	–	–	–	–	–	–	–
TL2	CCh	Timer/Counter 2 Low byte	–	–	–	–	–	–	–	–
TCON	88h	Timer/Counter 0 and 1 control	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
TMOD	89h	Timer/Counter 0 and 1 Modes	GATE1	C/T1#	M11	M01	GATE0	C/T0#	M10	M00

Dual Data Pointer

Description

The AT89C51CC03 implements a second data pointer for speeding up code execution and reducing code size in case of intensive usage of external memory accesses.
DPTR 0 and DPTR 1 are seen by the CPU as DPTR and are accessed using the SFR addresses 83h and 84h that are the DPH and DPL addresses. The DPS bit in AUXR1 register (see Figure 8) is used to select whether DPTR is the data pointer 0 or the data pointer 1 (see Figure 13).

Figure 13. Dual Data Pointer Implementation



Application

Software can take advantage of the additional data pointers to both increase speed and reduce code size, for example, block operations (copy, compare…) are well served by using one data pointer as a “source” pointer and the other one as a “destination” pointer. Hereafter is an example of block move implementation using the two pointers and coded in assembler. The latest C compiler takes also advantage of this feature by providing enhanced algorithm libraries.

The INC instruction is a short (2 Bytes) and fast (6 machine cycle) way to manipulate the DPS bit in the AUXR1 register. However, note that the INC instruction does not directly force the DPS bit to a particular state, but simply toggles it. In simple routines, such as the block move example, only the fact that DPS is toggled in the proper sequence matters, not its actual value. In other words, the block move routine works the same whether DPS is '0' or '1' on entry.

```
; ASCII block move using dual data pointers
; Modifies DPTR0, DPTR1, A and PSW
; Ends when encountering NULL character
; Note: DPS exits opposite to the entry state unless an extra INC AUXR1 is added

AUXR1EQU0A2h

move:movDPTR,#SOURCE ; address of SOURCE
   incAUXR1 ; switch data pointers
   movDPTR,#DEST ; address of DEST
mv_loop:incAUXR1; switch data pointers
   movxA,@DPTR; get a byte from SOURCE
   incDPTR; increment SOURCE address
   incAUXR1; switch data pointers
   movx@DPTR,A; write the byte to DEST
   incDPTR; increment DEST address
   jnzmv_loop; check for NULL terminator
end_move:
```

Cross Flash Memory Access

Code executing from		Action	FM0 (user Flash)	FM1 (boot Flash)
	FM0 (user Flash)	Read	ok	-
		Load column latch	ok	-
		Write	-	-
	FM1 (boot Flash)	Read	ok	ok
		Load column latch	ok	-
		Write	ok	-
	External memory EA = 0	Read	(a)	-
		Load column latch	-	-
		Write	-	-

(a) Depend upon general lock bit configuration.

Given Address

Each device has an individual address that is specified in the SADDR register; the SADEN register is a mask byte that contains don't-care bits (defined by zeros) to form the device's given address. The don't-care bits provide the flexibility to address one or more slaves at a time. The following example illustrates how a given address is formed. To address a device by its individual address, the SADEN mask byte must be `1111 1111b`.

For example:

SADDR0101 0110b
SADEN1111 1100b

Given0101 01XXb

Here is an example of how to use given addresses to address different slaves:

Slave A:SADDR1111 0001b
SADEN1111 1010b

Given1111 0X0Xb

Slave B:SADDR1111 0011b
SADEN1111 1001b

Given1111 0XX1b

Slave C:SADDR1111 0011b
SADEN1111 1101b

Given1111 00X1b

The SADEN byte is selected so that each slave may be addressed separately.
For slave A, bit 0 (the LSB) is a don't-care bit; for slaves B and C, bit 0 is a 1. To communicate with slave A only, the master must send an address where bit 0 is clear (e.g. `1111 0000b`).
For slave A, bit 1 is a 0; for slaves B and C, bit 1 is a don't care bit. To communicate with slaves A and B, but not slave C, the master must send an address with bits 0 and 1 both set (e.g. `1111 0011b`).
To communicate with slaves A, B and C, the master must send an address with bit 0 set, bit 1 clear, and bit 2 clear (e.g. `1111 0001b`).

Broadcast Address

A broadcast address is formed from the logical OR of the SADDR and SADEN registers with zeros defined as don't-care bits, e.g.:

SADDR0101 0110b
SADEN1111 1100b

SADDR OR SADEN1111 111Xb

The use of don't-care bits provides flexibility in defining the broadcast address, however in most applications, a broadcast address is FFh. The following is an example of using broadcast addresses:

Slave A:SADDR1111 0001b
SADEN1111 1010b

Given1111 1X11b,

Slave B:SADDR1111 0011b
SADEN1111 1001b

Given1111 1X11B,

Slave C:SADDR=1111 0010b
SADEN1111 1101b

Given1111 1111b

Timer 1

Timer 1 is identical to Timer 0 excepted for Mode 3 which is a hold-count mode. The following comments help to understand the differences:

- Timer 1 functions as either a Timer or event Counter in three modes of operation. Figure 35 to Figure 37 show the logical configuration for modes 0, 1, and 2. Timer 1's mode 3 is a hold-count mode.
- Timer 1 is controlled by the four high-order bits of TMOD register (see Figure 31) and bits 2, 3, 6 and 7 of TCON register (see Figure 30). TMOD register selects the method of Timer gating (GATE1), Timer or Counter operation (C/T1#) and mode of operation (M11 and M01). TCON register provides Timer 1 control functions: overflow flag (TF1), run control bit (TR1), interrupt flag (IE1) and interrupt type control bit (IT1).
- Timer 1 can serve as the Baud Rate Generator for the Serial Port. Mode 2 is best suited for this purpose.
- For normal Timer operation (GATE1 = 0), setting TR1 allows TL1 to be incremented by the selected input. Setting GATE1 and TR1 allows external pin INT1# to control Timer operation.
- Timer 1 overflow (count rolls over from all 1s to all 0s) sets the TF1 flag generating an interrupt request.
- When Timer 0 is in mode 3, it uses Timer 1's overflow flag (TF1) and run control bit (TR1). For this situation, use Timer 1 only for applications that do not require an interrupt (such as a Baud Rate Generator for the Serial Port) and switch Timer 1 in and out of mode 3 to turn it off and on.
- It is important to stop Timer/Counter before changing mode.

Mode 0 (13-bit Timer)

Mode 0 configures Timer 1 as a 13-bit Timer, which is set up as an 8-bit Timer (TH1 register) with a modulo-32 prescaler implemented with the lower 5 bits of the TL1 register (see Figure 35). The upper 3 bits of TL1 register are ignored. Prescaler overflow increments TH1 register.

Mode 1 (16-bit Timer)

Mode 1 configures Timer 1 as a 16-bit Timer with TH1 and TL1 registers connected in cascade (see Figure 36). The selected input increments TL1 register.

Mode 2 (8-bit Timer with Auto-Reload)

Mode 2 configures Timer 1 as an 8-bit Timer (TL1 register) with automatic reload from TH1 register on overflow (see Figure 37). TL1 overflow sets TF1 flag in TCON register and reloads TL1 with the contents of TH1, which is preset by software. The reload leaves TH1 unchanged.

Mode 3 (Halt)

Placing Timer 1 in mode 3 causes it to halt and hold its count. This can be used to halt Timer 1 when TR1 run control bit is not available i.e. when Timer 0 is in mode 3.

Table 39. TL2 Register

TL2 (S:CCh)
Timer 2 Low Byte Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

Bit Number	Bit Mnemonic	Description
7-0		Low Byte of Timer 2.

Reset Value = 0000 0000b
Not bit addressable

Table 40. RCAP2H Register

RCAP2H (S:CBh)
Timer 2 Reload/Capture High Byte Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

Bit Number	Bit Mnemonic	Description
7-0		High Byte of Timer 2 Reload/Capture.

Reset Value = 0000 0000b
Not bit addressable

Table 41. RCAP2L Register

RCAP2L (S:CA$_H$)
TIMER 2 RELOAD/CAPTURE Low Byte Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

Bit Number	Bit Mnemonic	Description
7-0		Low Byte of Timer 2 Reload/Capture.

Reset Value = 0000 0000b
Not bit addressable

Watchdog Timer During Power-down Mode and Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode, the user does not need to service the WDT. There are 2 methods of exiting Power-down mode: by a hardware reset or via a level activated external interrupt which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, the Watchdog is disabled. Exiting Power-down with an interrupt is significantly different. The interrupt shall be held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down.

To ensure that the WDT does not overflow within a few states of exiting powerdown, it is best to reset the WDT just before entering powerdown.

In the Idle mode, the oscillator continues to run. To prevent the WDT from resetting AT89C51CC03 while in Idle mode, the user should always set up a timer that will periodically exit Idle, service the WDT, and re-enter Idle mode.

Register

Table 44. WDTPRG Register

WDTPRG (S:A7h)
Watchdog Timer Duration Programming Register

7	6	5	4	3	2	1	0
–	–	–	–	–	S2	S1	S0

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
2	S2	**Watchdog Timer Duration selection bit 2** Work in conjunction with bit 1 and bit 0.
1	S1	**Watchdog Timer Duration selection bit 1** Work in conjunction with bit 2 and bit 0.
0	S0	**Watchdog Timer Duration selection bit 0** Work in conjunction with bit 1 and bit 2.

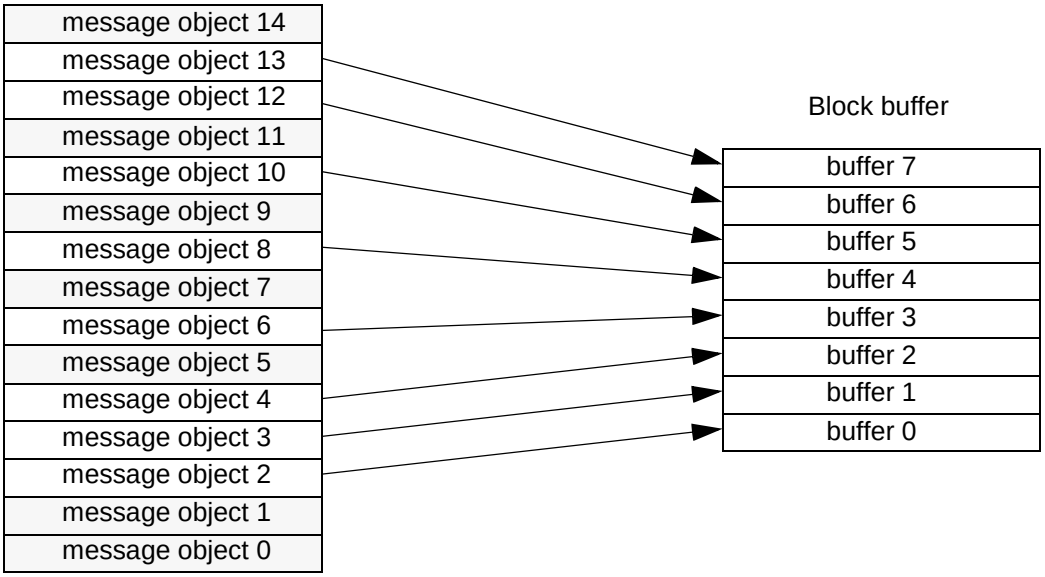
Reset Value = XXXX X000b

Buffer Mode

Any message object can be used to define one buffer, including non-consecutive message objects, and with no limitation in number of message objects used up to 15.

Each message object of the buffer must be initialized CONCH2 = 1 and CONCH1 = 1;

Figure 49. Buffer mode



The same acceptance filter must be defined for each message objects of the buffer. When there is no mask on the identifier or the IDE, all messages are accepted.

A received frame will always be stored in the lowest free message object.

When the flag Rxok is set on one of the buffer message objects, this message object can then be read by the application. This flag must then be cleared by the software and the message object re-enabled in buffer reception in order to free the message object.

The OVRBUF flag in the CANGIT register is set when the buffer is full. This flag can generate an interrupt.

The frames following the buffer-full interrupt will not stored and no status will be overwritten in the CANSTCH registers involved in the buffer until at least one of the buffer message objects is re-enabled in reception.

This flag must be cleared by the software in order to acknowledge the interrupt.

Acceptance Filter

Upon a reception hit (i.e., a good comparison between the ID+RTR+RB+IDE received and an ID+RTR+RB+IDE specified while taking the comparison mask into account) the ID+RTR+RB+IDE received are written over the ID TAG Registers.

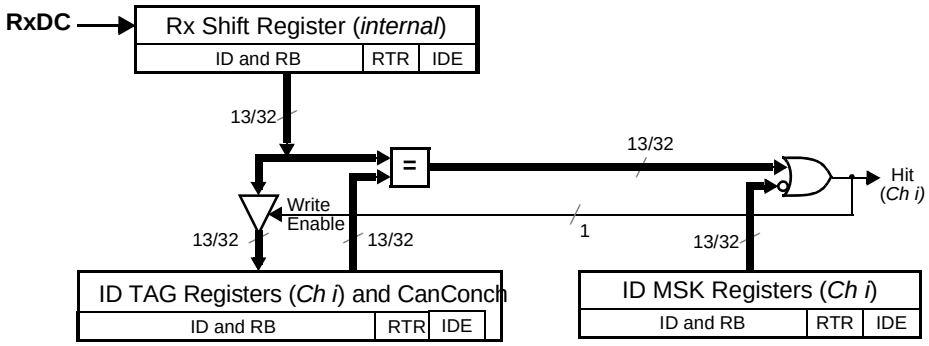
ID => IDT0-29

RTR => RTRTAG

RB => RB0-1TAG

IDE => IDE in CANCONCH register

Figure 54. Acceptance filter block diagram



example:
To accept only ID = 318h in part A.
ID MSK = 111 1111 1111 b
ID TAG = 011 0001 1000 b

CAN SFRs

CAN SFR's

Table 47. CAN SFR's With Reset Values

	0/8[1]	1/9	2/A	3/B	4/C	5/D	6/E	7/F	
F8h	IPL1 xxxx x000	CH 0000 0000	CCAP0H 0000 0000	CCAP1H 0000 0000	CCAP2H 0000 0000	CCAP3H 0000 0000	CCAP4H 0000 0000		**FFh**
F0h	B 0000 0000		ADCLK xx00 x000	ADCON 0000 0000	ADDL xxxx xx00	ADDH 0000 0000	ADCF 0000 0000	IPH1 xxxx x000	**F7h**
E8h	IEN1 xxxx x000	CL 0000 0000	CCAP0L 0000 0000	CCAP1L 0000 0000	CCAP2L 0000 0000	CCAP3L 0000 0000	CCAP4L 0000 0000		**EFh**
E0h	ACC 0000 0000								**E7h**
D8h	CCON 00xx xx00	CMOD 00xx x000	CCAPM0 x000 0000	CCAPM1 x000 0000	CCAPM2 x000 0000	CCAPM3 x000 0000	CCAPM4 x000 0000		**DFh**
D0h	PSW 0000 0000	FCON 0000 0000	EECON xxxx xx00	FSTA xxxx xx00	SPCON 0001 0100	SPSCR 0000 0000	SPDAT xxxx xxxx		**D7h**
C8h	T2CON 0000 0000	T2MOD xxxx xx00	RCAP2L 0000 0000	RCAP2H 0000 0000	TL2 0000 0000	TH2 0000 0000	CANEN1 xx00 0000	CANEN2 0000 0000	**CFh**
C0h	P4 xxxx xx11	CANGIE 0000 0000	CANIE1 xx00 0000	CANIE2 0000 0000	CANIDM1 xxxx xxxx	CANIDM2 xxxx xxxx	CANIDM3 xxxx xxxx	CANIDM4 xxxx xxxx	**C7h**
B8h	IPL0 x000 0000	SADEN 0000 0000	CANSIT1 0x00 0000	CANSIT2 0000 0000	CANIDT1 xxxx xxxx	CANIDT2 xxxx xxxx	CANIDT3 xxxx xxxx	CANIDT4 xxxx xxxx	**BFh**
B0h	P3 1111 1111	CANPAGE 0000 0000	CANSTCH xxxx xxxx	CANCONCH xxxx xxxx	CANBT1 xxxx xxxx	CANBT2 xxxx xxxx	CANBT3 xxxx xxxx	IPH0 x000 0000	**B7h**
A8h	IEN0 0000 0000	SADDR 0000 0000	CANGSTA 0000 0000	CANGCON 0000 x000	CANTIML 0000 0000	CANTIMH 0000 0000	CANSTMPL 0000 0000	CANSTMPH 0000 0000	**AFh**
A0h	P2 1111 1111	CANTCON 0000 0000	AUXR1 xxxx 00x0	CANMSG xxxx xxxx	CANTTCL 0000 0000	CANTTCH 0000 0000	WDTRST 1111 1111	WDTPRG xxxx x000	**A7h**
98h	SCON 0000 0000	SBUF 0000 0000		CANGIT 0x00 0000	CANTEC 0000 0000	CANREC 0000 0000		CKCON1 xxxx xxx0	**9Fh**
90h	P1 1111 1111								**97h**
88h	TCON 0000 0000	TMOD 0000 0000	TL0 0000 0000	TL1 0000 0000	TH0 0000 0000	TH1 0000 0000	AUXR X001 0100	CKCON 0000 0000	**8Fh**
80h	P0 1111 1111	SP 0000 0111	DPL 0000 0000	DPH 0000 0000				PCON 0000 0000	**87h**
	0/8[1]	**1/9**	**2/A**	**3/B**	**4/C**	**5/D**	**6/E**	**7/F**	

Table 56. CANSIT1 Register

CANSIT1 (S:BAh Read Only)
CAN Status Interrupt Message Object Registers 1

7	6	5	4	3	2	1	0
-	SIT14	SIT13	SIT12	SIT11	SIT10	SIT9	SIT8

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The values read from this bit is indeterminate. Do not set this bit.
6-0	SIT14:8	**Status of Interrupt by Message Object** 0 - no interrupt. 1 - IT turned on. Reset when interrupt condition is cleared by user. SIT14:8 = 0b 0000 1001 -> IT's on message objects 11 and 8. see Figure 50.

Reset Value = x000 0000b

Table 57. CANSIT2 Register

CANSIT2 (S:BBh Read Only)

CAN Status Interrupt Message Object Registers 2

7	6	5	4	3	2	1	0
SIT7	SIT6	SIT5	SIT4	SIT3	SIT2	SIT1	SIT0

Bit Number	Bit Mnemonic	Description
7-0	SIT7:0	**Status of Interrupt by Message Object** 0 - no interrupt. 1 - IT turned on. Reset when interrupt condition is cleared by user. SIT7:0 = 0b 0000 1001 -> IT's on message objects 3 and 0 see Figure 50.

Reset Value = 0000 0000b

Table 63. CANPAGE Register

CANPAGE (S:B1h)
CAN Message Object Page Register

7	6	5	4	3	2	1	0
CHNB 3	CHNB 2	CHNB 1	CHNB 0	AINC	INDX2	INDX1	INDX0

Bit Number	Bit Mnemonic	Description
7-4	CHNB3:0	**Selection of Message Object Number** The available numbers are: 0 to 14 (see Figure 48).
3	AINC	**Auto Increment of the Index (active low)** 0 - auto-increment of the index (default value). 1 - non-auto-increment of the index.
2-0	INDX2:0	**Index** Byte location of the data field for the defined message object (see Figure 48).

Reset Value = 0000 0000b

Table 64. CANCONCH Register

CANCONCH (S:B3h)
CAN Message Object Control and DLC Register

7	6	5	4	3	2	1	0
CONCH 1	CONCH 0	RPLV	IDE	DLC 3	DLC 2	DLC 1	DLC 0

Bit Number	Bit Mnemonic	Description
7-6	CONCH1:0	**Configuration of Message Object** **CONCH1 CONCH0** 0 0: disable 0 1: Launch transmission 1 0: Enable Reception 1 1: Enable Reception Buffer Note: The user must re-write the configuration to enable the corresponding bit in the CANEN1:2 registers.
5	RPLV	**Reply Valid** Used in the automatic reply mode after receiving a remote frame 0 - reply not ready. 1 - reply ready and valid.
4	IDE	**Identifier Extension** 0 - CAN standard rev 2.0 A (ident = 11 bits). 1 - CAN standard rev 2.0 B (ident = 29 bits).
3-0	DLC3:0	**Data Length Code** Number of Bytes in the data field of the message. The range of DLC is from 0 up to 8. This value is updated when a frame is received (data or remote frame). If the expected DLC differs from the incoming DLC, a warning appears in the CANSTCH register.

No default value after reset

Table 66. CANIDT1 Register for V2.0 part A

CANIDT1 for V2.0 part A (S:BCh)
CAN Identifier Tag Registers 1

7	6	5	4	3	2	1	0
IDT 10	IDT 9	IDT 8	IDT 7	IDT 6	IDT 5	IDT 4	IDT 3

Bit Number	Bit Mnemonic	Description
7-0	IDT10:3	**IDentifier tag value** See Figure 54.

No default value after reset.

Table 67. CANIDT2 Register for V2.0 part A

CANIDT2 for V2.0 part A (S:BDh)
CAN Identifier Tag Registers 2

7	6	5	4	3	2	1	0
IDT 2	IDT 1	IDT 0	-	-	-	-	-

Bit Number	Bit Mnemonic	Description
7-5	IDT2:0	**IDentifier tag value** See Figure 54.
4-0	-	**Reserved** The values read from these bits are indeterminate. Do not set these bits.

No default value after reset.

Table 68. CANIDT3 Register for V2.0 part A

CANIDT3 for V2.0 part A (S:BEh)
CAN Identifier Tag Registers 3

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

Bit Number	Bit Mnemonic	Description
7-0	-	**Reserved** The values read from these bits are indeterminate. Do not set these bits.

No default value after reset.

Table 79. CANIDM2 Register for V2.0 part B

CANIDM2 for V2.0 part B (S:C5h)
CAN Identifier Mask Registers 2

7	6	5	4	3	2	1	0
IDMSK 20	IDMSK 19	IDMSK 18	IDMSK 17	IDMSK 16	IDMSK 15	IDMSK 14	IDMSK 13

Bit Number	Bit Mnemonic	Description
7-0	IDMSK20:13	**IDentifier Mask Value** 0 - comparison true forced. 1 - bit comparison enabled. See Figure 54.

Note: The ID Mask is only used for reception.

No default value after reset.

Table 80. CANIDM3 Register for V2.0 part B

CANIDM3 for V2.0 part B (S:C6h)
CAN Identifier Mask Registers 3

7	6	5	4	3	2	1	0
IDMSK 12	IDMSK 11	IDMSK 10	IDMSK 9	IDMSK 8	IDMSK 7	IDMSK 6	IDMSK 5

Bit Number	Bit Mnemonic	Description
7-0	IDMSK12:5	**IDentifier Mask Value** 0 - comparison true forced. 1 - bit comparison enabled. See Figure 54.

Note: The ID Mask is only used for reception.

No default value after reset.

Table 86. CANSTMPH Register

CANSTMPH (S:AFh Read Only)
CAN Stamp Timer High

7	6	5	4	3	2	1	0
TIMSTMP 15	TIMSTMP 14	TIMSTMP 13	TIMSTMP 12	TIMSTMP 11	TIMSTMP 10	TIMSTMP 9	TIMSTMP 8

Bit Number	Bit Mnemonic	Description
7-0	TIMSTMP15:8	**High byte of Time Stamp** See Figure 55.

No default value after reset

Table 87. CANSTMPL Register

CANSTMPL (S:AEh Read Only)
CAN Stamp Timer Low

7	6	5	4	3	2	1	0
TIMSTMP 7	TIMSTMP 6	TIMSTMP 5	TIMSTMP 4	TIMSTMP 3	TIMSTMP 2	TIMSTMP 1	TIMSTMP 0

Bit Number	Bit Mnemonic	Description
7-0	TIMSTMP7:0	**Low byte of Time Stamp** See Figure 55.

No default value after reset

Table 88. CANTTCH Register

CANTTCH (S:A5h Read Only)
CAN TTC Timer High

7	6	5	4	3	2	1	0
TIMTTC 15	TIMTTC 14	TIMTTC 13	TIMTTC 12	TIMTTC 11	TIMTTC 10	TIMTTC 9	TIMTTC 8

Bit Number	Bit Mnemonic	Description
7-0	TIMTTC15:8	**High byte of TTC Timer** See Figure 55.

Reset Value = 0000 0000b

Serial Port Interface (SPI)

The Serial Peripheral Interface Module (SPI) allows full-duplex, synchronous, serial communication between the MCU and peripheral devices, including other MCUs.

Features

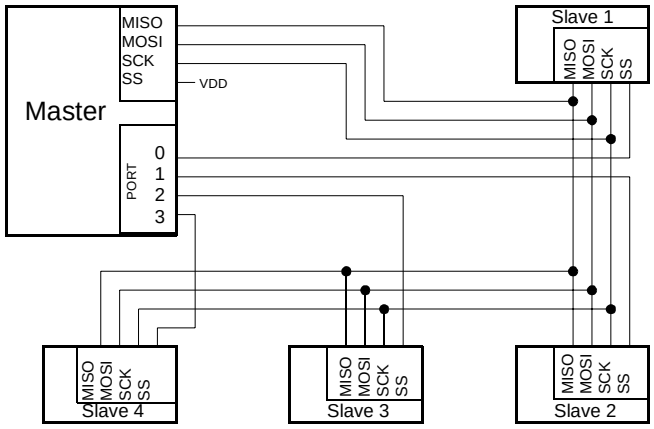
Features of the SPI Module include the following:

- Full-duplex, three-wire synchronous transfers
- Master or Slave operation
- Six programmable Master clock rates in master mode
- Serial clock with programmable polarity and phase
- Master Mode fault error flag with MCU interrupt capability

Signal Description

Figure 57 shows a typical SPI bus configuration using one Master controller and many Slave peripherals. The bus is made of three wires connecting all the devices.

Figure 57. SPI Master/Slaves Interconnection



The Master device selects the individual Slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the Slave devices.

Master Output Slave Input (MOSI)

This 1-bit signal is directly connected between the Master Device and a Slave Device. The MOSI line is used to transfer data in series from the Master to the Slave. Therefore, it is an output signal from the Master, and an input signal to a Slave. A Byte (8-bit word) is transmitted most significant bit (MSB) first, least significant bit (LSB) last.

Master Input Slave Output (MISO)

This 1-bit signal is directly connected between the Slave Device and a Master Device. The MISO line is used to transfer data in series from the Slave to the Master. Therefore, it is an output signal from the Slave, and an input signal to the Master. A Byte (8-bit word) is transmitted most significant bit (MSB) first, least significant bit (LSB) last.

SPI Serial Clock (SCK)

This signal is used to synchronize the data transmission both in and out of the devices through their MOSI and MISO lines. It is driven by the Master for eight clock cycles which allows to exchange one Byte on the serial lines.

Slave Select ($\overline{SS}$)

Each Slave peripheral is selected by one Slave Select pin ($\overline{SS}$). This signal must stay low for any message for a Slave. It is obvious that only one Master ($\overline{SS}$ high level) can drive the network. The Master may select each Slave device by software through port pins (Figure 58). To prevent bus conflicts on the MISO line, only one slave should be selected at a time by the Master for a transmission.

Bit Number	Bit Mnemonic	Description
4	MODF	**Mode Fault** - Set by hardware to indicate that the $\overline{SS}$ pin is in inappropriate logic level (in both master and slave modes). - Cleared by hardware when reading SPSCR When MODF error occurred: - In slave mode: SPI interface ignores all transmitted data while $\overline{SS}$ remains high. A new transmission is perform as soon as SS returns low. - In master mode: SPI interface is disabled (SPEN=0, see description for SPEN bit in SPCON register).
3	SPTE	**Serial Peripheral Transmit register Empty** - Set by hardware when transmit register is empty (if needed, SPDAT can be loaded with another data). - Cleared by hardware when transmit register is full (no more data should be loaded in SPDAT).
2	UARTM	**Serial Peripheral UART mode** Set and cleared by software: - Clear: Normal mode, data are transmitted MSB first (default) - Set: UART mode, data are transmitted LSB first.
1	SPTEIE	**Interrupt Enable for SPTE** Set and cleared by software: - Set to enable SPTE interrupt generation (when SPTE goes high, an interrupt is generated). - Clear to disable SPTE interrupt generation Caution: When SPTEIE is set no interrupt generation occurred when SPIF flag goes high. To enable SPIF interrupt again, SPTEIE should be cleared.
0	MODFIE	**Interrupt Enable for MODF** Set and cleared by software: - Set to enable MODF interrupt generation - Clear to disable MODF interrupt generation

Reset Value = 00X0 XXXXb

Not Bit addressable

Serial Peripheral DATa Register (SPDAT)

The Serial Peripheral Data Register (Table 94) is a read/write buffer for the receive data register. A write to SPDAT places data directly into the shift register. No transmit buffer is available in this model.

A Read of the SPDAT returns the value located in the receive buffer and not the content of the shift register.

Table 94. SPDAT Register

SPDAT - Serial Peripheral Data Register (0D6H)

7	6	5	4	3	2	1	0
R7	R6	R5	R4	R3	R2	R1	R0

Reset Value = Indeterminate

R7:R0: Receive data bits

Bit Number	Bit Mnemonic	Description
7-2	-	**Reserved** The value read from these bits are indeterminate. Do not set these bits.
1-0	ADAT1:0	**ADC result** bits 1-0

Reset Value = 00h

Table 115. IPH1 Register

IPH1 (S:F7h)
Interrupt High Priority Register 1

7	6	5	4	3	2	1	0
-	-	-	-	SPIH	POVRH	PADCH	PCANH

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	SPIH	**SPI Interrupt Priority Level Most Significant bit** SPIH SPIL Priority level 0 0 Lowest 0 1 1 0 1 1 Highest
2	POVRH	**Timer overrun Interrupt Priority Level Most Significant bit** POVRH POVRL Priority level 0 0 Lowest 0 1 1 0 1 1 Highest
1	PADCH	**ADC Interrupt Priority Level Most Significant bit** PADCH PADCL Priority level 0 0 Lowest 0 1 1 0 1 1 Highest
0	PCANH	**CAN Interrupt Priority Level Most Significant bit** PCANH PCANL Priority level 0 0 Lowest 0 1 1 0 1 1 Highest

Reset Value = XXXX 0000b