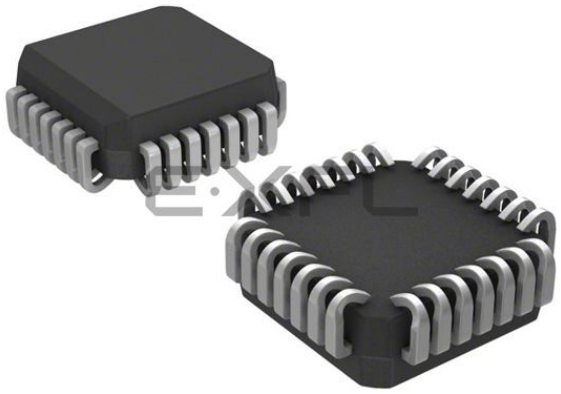


Welcome to E-XFL.COM

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	80C51
Core Size	8-Bit
Speed	40MHz
Connectivity	CANbus, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	20
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	2K x 8
RAM Size	512 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	28-LCC (J-Lead)
Supplier Device Package	28-PLCC (11.51x11.51)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at89c51cc02ca-sisum

Table 9. CAN SFRs (Continued)

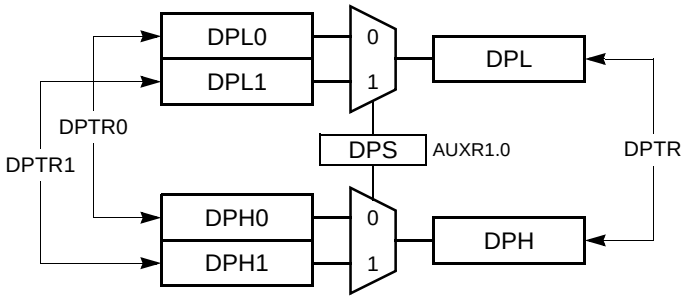
Mnemonic	Add	Name	7	6	5	4	3	2	1	0
CANEN	CFh	CAN Enable Channel byte					ENCH3	ENCH2	ENCH1	ENCH0
CANGIE	C1h	CAN General Interrupt Enable			ENRX	ENTX	ENERCH	ENBUF	ENERG	
CANIE	C3h	CAN Interrupt Enable Channel byte					IECH3	IECH2	IECH1	IECH0
CANSIT	BBh	CAN Status Interrupt Channel byte					SIT3	SIT2	SIT1	SIT0
CANTCON	A1h	CAN Timer Control	TPRESC 7	TPRESC 6	TPRESC 5	TPRESC 4	TPRESC 3	TPRESC 2	TPRESC 1	TPRESC 0
CANTIMH	ADh	CAN Timer high	CANTIM 15	CANTIM 14	CANTIM 13	CANTIM 12	CANTIM 11	CANTIM 10	CANTIM 9	CANTIM 8
CANTIML	ACh	CAN Timer low	CANTIM 7	CANTIM 6	CANTIM 5	CANTIM 4	CANTIM 3	CANTIM 2	CANTIM 1	CANTIM 0
CANSTMPH	AFh	CAN Timer Stamp high	TIMSTMP 15	TIMSTMP 14	TIMSTMP 13	TIMSTMP 12	TIMSTMP 11	TIMSTMP 10	TIMSTMP 9	TIMSTMP 8
CANSTMPL	AEh	CAN Timer Stamp low	TIMSTMP7	TIMSTMP 6	TIMSTMP 5	TIMSTMP 4	TIMSTMP 3	TIMSTMP 2	TIMSTMP 1	TIMSTMP 0
CANTTCH	A5h	CAN Timer TTC high	TIMTTC 15	TIMTTC 14	TIMTTC 13	TIMTTC 12	TIMTTC 11	TIMTTC 10	TIMTTC 9	TIMTTC 8
CANTTCL	A4h	CAN Timer TTC low	TIMTTC 7	TIMTTC 6	TIMTTC 5	TIMTTC 4	TIMTTC 3	TIMTTC 2	TIMTTC 1	TIMTTC 0
CANTEC	9Ch	CAN Transmit Error Counter	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0
CANREC	9Dh	CAN Receive Error Counter	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0
CANPAGE	B1h	CAN Page	-	-	CHNB1	CHNB0	AINC	INDX2	INDX1	INDX0
CANSTCH	B2h	CAN Status Channel	DLCW	TXOK	RXOK	BERR	SERR	CERR	FERR	AERR
CANCONCH	B3h	CAN Control Channel	CONCH1	CONCH0	RPLV	IDE	DLC3	DLC2	DLC1	DLC0
CANMSG	A3h	CAN Message Data	MSG7	MSG6	MSG5	MSG4	MSG3	MSG2	MSG1	MSG0
CANIDT1	BCh	CAN Identifier Tag byte 1(Part A) CAN Identifier Tag byte 1(PartB)	IDT10 IDT28	IDT9 IDT27	IDT8 IDT26	IDT7 IDT25	IDT6 IDT24	IDT5 IDT23	IDT4 IDT22	IDT3 IDT21
CANIDT2	BDh	CAN Identifier Tag byte 2 (PartA) CAN Identifier Tag byte 2 (PartB)	IDT2 IDT20	IDT1 IDT19	IDT0 IDT18	- IDT17	- IDT16	- IDT15	- IDT14	- IDT13
CANIDT3	BEh	CAN Identifier Tag byte 3(PartA) CAN Identifier Tag byte 3(PartB)	- IDT12	- IDT11	- IDT10	- IDT9	- IDT8	- IDT7	- IDT6	- IDT5
CANIDT4	BFh	CAN Identifier Tag byte 4(PartA) CAN Identifier Tag byte 4(PartB)	- IDT4	- IDT3	- IDT2	- IDT1	- IDT0	RTRTAG	- RB1TAG	RB0TAG
CANIDM1	C4h	CAN Identifier Mask byte 1(PartA) CAN Identifier Mask byte 1(PartB)	IDMSK10 IDMSK28	IDMSK9 IDMSK27	IDMSK8 IDMSK26	IDMSK7 IDMSK25	IDMSK6 IDMSK24	IDMSK5 IDMSK23	IDMSK4 IDMSK22	IDMSK3 IDMSK21

Dual Data Pointer

Description

The T89C51CC02 implements a second data pointer for speeding up code execution and reducing code size in case of intensive usage of external memory accesses.
DPTR0 and DPTR1 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 19) is used to select whether DPTR is the data pointer 0 or the data pointer 1 (See Figure 11).

Figure 11. 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:
```

Overview of FM0 Operations

The CPU interfaces the Flash memory through the FCON register and AUXR1 register.

These registers are used to:

- Map the memory spaces in the adressable space
- Launch the programming of the memory spaces
- Get the status of the Flash memory (busy/not busy)

Mapping of the Memory Space

By default, the user space is accessed by MOVC instruction for read only. The column latches space is made accessible by setting the FPS bit in FCON register. Writing is possible from 0000h to 3FFFh, address bits 6 to 0 are used to select an address within a page while bits 14 to 7 are used to select the programming address of the page.
Setting FPS bit takes precedence on the EEE bit in EECON register.

The other memory spaces (user, extra row, hardware security) are made accessible in the code segment by programming bits FMOD0 and FMOD1 in FCON register in accordance with Table 22. A MOVC instruction is then used for reading these spaces.

Table 22. FM0 blocks Select bits

FMOD1	FMOD0	FM0 Adressable Space
0	0	User (0000h-3FFFh)
0	1	Extra Row(FF80h-FFFFh)
1	0	Hardware Security Byte (0000h)
1	1	Reserved

Launching Programming

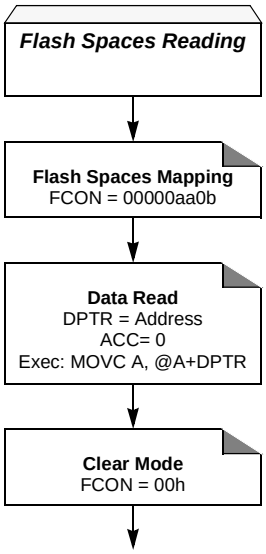
FPL3:0 bits in FCON register are used to secure the launch of programming. A specific sequence must be written in these bits to unlock the write protection and to launch the programming. This sequence is 5xh followed by Axh. Table 23 summarizes the memory spaces to program according to FMOD1:0 bits.

Table 23. Programming Spaces

	Write to FCON				
	FPL3:0	FPS	FMOD1	FMOD0	Operation
User	5	x	0	0	No action
	A	x	0	0	Write the column latches in user space
Extra Row	5	x	0	1	No action
	A	x	0	1	Write the column latches in extra row space
Hardware Security Byte	5	x	1	0	No action
	A	x	1	0	Write the fuse bits space
Reserved	5	x	1	1	No action
	A	x	1	1	No action

Note: The sequence 5xh and Axh must be executing without instructions between them otherwise the programming is aborted.
Interrupts that may occur during programming time must be disabled to avoid any spurious exit of the programming mode.

Figure 17. Reading Procedure



Note: aa = 10 for the Hardware Security Byte.

Flash Protection from Parallel Programming

The three lock bits in Hardware Security Byte (See 'In-System Programming' section) are programmed according to Table 24 provide different level of protection for the on-chip code and data located in FM0 and FM1.

The only way to write this bits are the parallel mode. They are set by default to level 3.

Table 24. Program Lock bit

Program Lock bits				
Security Level	LB0	LB1	LB2	**Protection Description**
1	U	U	U	No program lock features enabled.
2	P	U	U	Parallel programming of the Flash is disabled.
3	U	P	U	Same as 2, also verify through parallel programming interface is disabled. This is the factory defaul programming.
4	U	U	P	Same as 3

Note: 1. Program Lock bits
U: unprogrammed
P: programmed

WARNING: Security level 2, 3 and 4 should only be programmed after Flash and Core verification.

Preventing Flash Corruption

See Section "Power Management".

XROW Bytes

The EXTRA ROW (XROW) includes 128 bytes. Some of these bytes are used for specific purpose in conjonction with the bootloader.

Table 31. XROW Mapping

Description	Default Value	Address
Copy of the Manufacturer Code	58h	30h
Copy of the Device ID#1: Family code	D7h	31h
Copy of the Device ID#2: Memories size and type	BBh	60h
Copy of the Device ID#3: Name and Revision	FFh	61h

Hardware Conditions

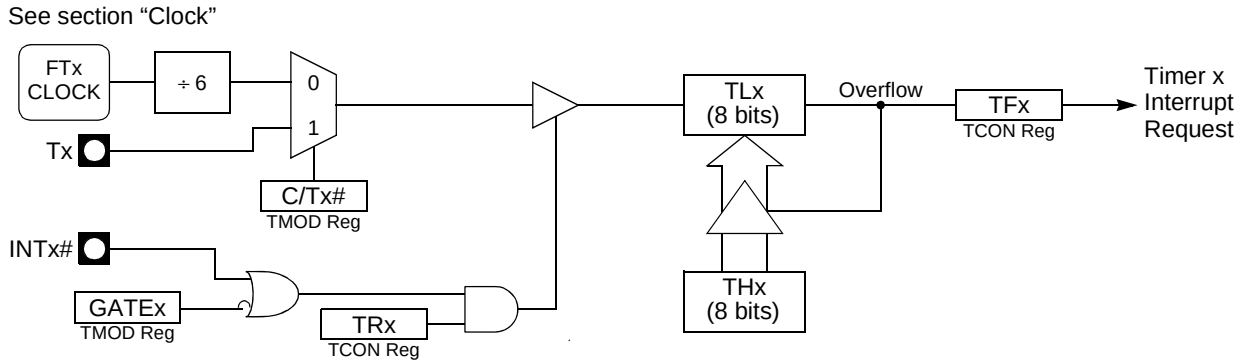
It is possible to force the controller to execute the bootloader after a Reset with hardware conditions.

During the first programming, the user can define a configuration on Port1 that will be recognized by the chip as the hardware conditions during a Reset. If this condition is met, the chip will start executing the bootloader at the end of the Reset.

See a detailed description in the applicable Document.

- Datasheet Bootloader CAN T89C51CC02.
- Datasheet Bootloader UART T89C51CC02.

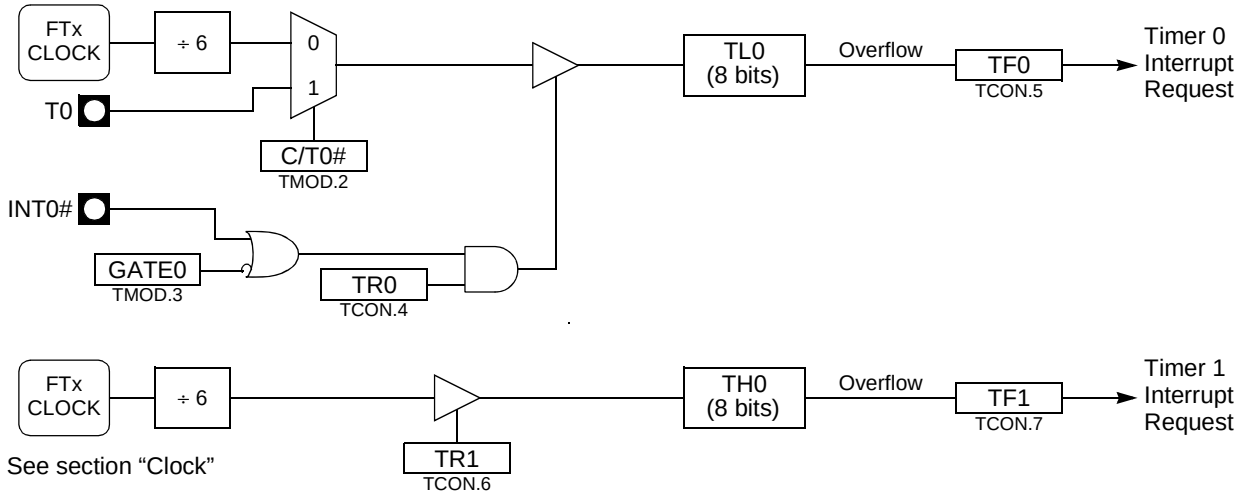
Figure 26. Timer/Counter x (x= 0 or 1) in Mode 2



Mode 3 (Two 8-bit Timers)

Mode 3 configures Timer 0 such that registers TL0 and TH0 operate as separate 8-bit Timers (See Figure 27). This mode is provided for applications requiring an additional 8-bit Timer or Counter. TL0 uses the Timer 0 control bits C/T0# and GATE0 in TMOD register, and TR0 and TF0 in TCON register in the normal manner. TH0 is locked into a Timer function (counting F_{PER} /6) and takes over use of the Timer 1 interrupt (TF1) and run control (TR1) bits. Thus, operation of Timer 1 is restricted when Timer 0 is in mode 3.

Figure 27. Timer/Counter 0 in Mode 3: Two 8-bit Counters



Timer 1

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

- Timer 1 functions as either a Timer or event Counter in three modes of operation. Figure 24 to Figure 26 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 39) and bits 2, 3, 6 and 7 of TCON register (See Figure 38). 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 24). 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 25). 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 26). 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.

Interrupt

Each Timer handles one interrupt source that is the timer overflow flag TF0 or TF1. This flag is set every time an overflow occurs. Flags are cleared when vectoring to the Timer interrupt routine. Interrupts are enabled by setting ETx bit in IEN0 register. This assumes interrupts are globally enabled by setting EA bit in IEN0 register.

Figure 28. Timer Interrupt System

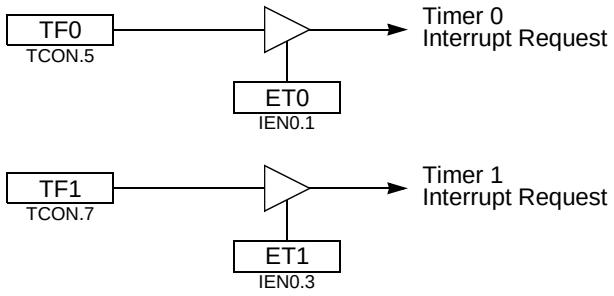


Table 39. TMOD Register
TMOD (S:89h)
Timer/Counter Mode Control Register

7	6	5	4	3	2	1	0
GATE1	**C/T1#**	**M11**	**M01**	**GATE0**	**C/T0#**	**M10**	**M00**

Bit Number	Bit Mnemonic	Description
7	GATE1	**Timer 1 Gating Control bit** Clear to enable Timer 1 whenever TR1 bit is set. Set to enable Timer 1 only while INT1# pin is high and TR1 bit is set.
6	C/T1#	**Timer 1 Counter/Timer Select bit** Clear for Timer operation: Timer 1 counts the divided-down system clock. Set for Counter operation: Timer 1 counts negative transitions on external pin T1.
5	M11	**Timer 1 Mode Select bits** M11 M01 Operating mode 0 0 Mode 0: 8-bit Timer/Counter (TH1) with 5bit prescaler (TL1). 0 1 Mode 1: 16-bit Timer/Counter. 1 1 Mode 3: Timer 1 halted. Retains count. 1 0 Mode 2: 8-bit auto-reload Timer/Counter (TL1).(1)
4	M01	
3	GATE0	**Timer 0 Gating Control bit** Clear to enable Timer 0 whenever TR0 bit is set. Set to enable Timer/Counter 0 only while INT0# pin is high and TR0 bit is set.
2	C/T0#	**Timer 0 Counter/Timer Select bit** Clear for Timer operation: Timer 0 counts the divided-down system clock. Set for Counter operation: Timer 0 counts negative transitions on external pin T0.
1	M10	**Timer 0 Mode Select bit** M10 M00 Operating mode 0 0 Mode 0: 8-bit Timer/Counter (TH0) with 5bit prescaler (TL0). 0 1 Mode 1: 16-bit Timer/Counter. 1 0 Mode 2: 8-bit auto-reload Timer/Counter (TL0).(2) 1 1 Mode 3: TL0 is an 8-bit Timer/Counter. TH0 is an 8-bit Timer using Timer 1's TR0 and TF0 bits.
0	M00	

Reset Value = 0000 0000b

Notes: 1. Reloaded from TH1 at overflow.
2. Reloaded from TH0 at overflow.

Table 40. TH0 Register
TH0 (S:8Ch)
Timer 0 High Byte Register

7	6	5	4	3	2	1	0

Bit Number	Bit Mnemonic	Description
7:0		**High Byte of Timer 0**

Reset Value = 0000 0000b

Bit Shortening

If, on the other hand, the transmitter oscillator is faster than the receiver one, the next falling edge used for resynchronization may be too early. So Phase Segment 2 in bit N is shortened in order to adjust the sample point for bit N+1 and the end of the bit time

Synchronization Jump Width

The limit to the amount of lengthening or shortening of the Phase Segments is set by the Resynchronization Jump Width.

This segment may not be longer than Phase Segment 2.

Programming the Sample Point

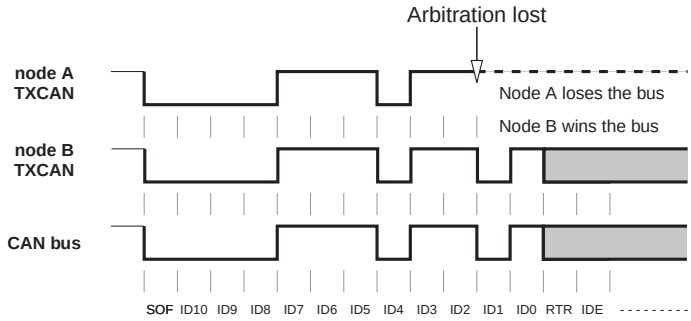
Programming of the sample point allows "tuning" of the characteristics to suit the bus.

Early sampling allows more Time Quanta in the Phase Segment 2 so the Synchronization Jump Width can be programmed to its maximum. This maximum capacity to shorten or lengthen the bit time decreases the sensitivity to node oscillator tolerances, so that lower cost oscillators such as ceramic resonators may be used.

Late sampling allows more Time Quanta in the Propagation Time Segment which allows a poorer bus topology and maximum bus length.

Arbitration

Figure 35. Bus Arbitration



The CAN protocol handles bus accesses according to the concept called “Carrier Sense Multiple Access with Arbitration on Message Priority”.

During transmission, arbitration on the CAN bus can be lost to a competing device with a higher priority CAN Identifier. This arbitration concept avoids collisions of messages whose transmission was started by more than one node simultaneously and makes sure the most important message is sent first without time loss.

The bus access conflict is resolved during the arbitration field mostly over the identifier value. If a data frame and a remote frame with the same identifier are initiated at the same time, the data frame prevails over the remote frame (c.f. RTR bit).

Errors

The CAN protocol signals any errors immediately as they occur. Three error detection mechanisms are implemented at the message level and two at the bit level:

Error at Message Level

- Cyclic Redundancy Check (CRC)
  The CRC safeguards the information in the frame by adding redundant check bits at the transmission end. At the receiver these bits are re-computed and tested against the received bits. If they do not agree there has been a CRC error.
- Frame Check
  This mechanism verifies the structure of the transmitted frame by checking the bit

fields against the fixed format and the frame size. Errors detected by frame checks are designated "format errors".

- ACK Errors
  As already mentioned frames received are acknowledged by all receivers through positive acknowledgement. If no acknowledgement is received by the transmitter of the message an ACK error is indicated.

Error at Bit Level

- Monitoring
  The ability of the transmitter to detect errors is based on the monitoring of bus signals. Each node which transmits also observes the bus level and thus detects differences between the bit sent and the bit received. This permits reliable detection of global errors and errors local to the transmitter.
- Bit Stuffing
  The coding of the individual bits is tested at bit level. The bit representation used by CAN is "Non Return to Zero (NRZ)" coding, which guarantees maximum efficiency in bit coding. The synchronization edges are generated by means of bit stuffing.

Error Signalling

If one or more errors are discovered by at least one node using the above mechanisms, the current transmission is aborted by sending an "error flag". This prevents other nodes accepting the message and thus ensures the consistency of data throughout the network. After transmission of an erroneous message that has been aborted, the sender automatically re-attempts transmission.

CAN Controller Description

The CAN controller accesses are made through SFR.
Several operations are possible by SFR:

- arithmetic and logic operations, transfers and program control (SFR is accessible by direct addressing).
- 4 independent message objects are implemented, a pagination system manages their accesses.

Any message object can be programmed in a reception buffer block (even non-consecutive buffers). For the reception of defined messages one or several receiver message objects can be masked without participating in the buffer feature. An IT is generated when the buffer is full. The frames following the buffer-full interrupt will not be taken into account until at least one of the buffer message objects is re-enabled in reception. Higher priority of a message object for reception or transmission is given to the lower message object number.

The programmable 16-bit Timer (CANTIMER) is used to stamp each received and sent message in the CANSTMP register. This timer starts counting as soon as the CAN controller is enabled by the ENA bit in the CANGCON register.

The Time Trigger Communication (TTC) protocol is supported by the T89C51CC02.

Registers

Table 56. CANGCON Register

CANGCON (S:ABh)
CAN General Control Register

7	6	5	4	3	2	1	0
ABRQ	OVRQ	TTC	SYNCTTC	AUTOBAUD	TEST	ENA	GRES

Bit Number	Bit Mnemonic	Description
7	ABRQ	**Abort Request** Not an auto-resetable bit. A reset of the ENCH bit (message object control & DLC register) is done for each message object. The pending transmission communications are immediately aborted but the on-going communication will be terminated normally, setting the appropriate status flags, TxOk or RxOk.
6	OVRQ	**Overload Frame Request (Initiator)**. Auto-resetable bit. Set to send an overload frame after the next received message. Cleared by the hardware at the beginning of transmission of the overload frame.
5	TTC	**Network in Timer Trigger Communication** set to select node in TTC. clear to disable TTC features.
4	SYNCTTC	**Synchronization of TTC** When this bit is set the TTC timer is caught on the last bit of the End Of Frame. When this bit is clear the TTC timer is caught on the Start Of Frame. This bit is only used in the TTC mode.
3	AUTOBAUD	**AUTOBAUD** set to active listening mode. Clear to disable listening mode
2	TEST	Test mode. The test mode is intended for factory testing and not for customer use.
1	ENA/$\overline{\text{STB}}$	**Enable/Standby CAN Controller** When this bit is set, it enables the CAN controller and its input clock. When this bit is clear, the on-going communication is terminated normally and the CAN controller state of the machine is frozen (the ENCH bit of each message object does not change). In the standby mode, the transmitter constantly provides a recessive level; the receiver is not activated and the input clock is stopped in the CAN controller. During the disable mode, the registers and the mailbox remain accessible. Note that two clock periods are needed to start the CAN controller state of the machine.
0	GRES	**General Reset (Software Reset).** Auto-resetable bit. This reset command is ‘ORed’ with the hardware reset in order to reset the controller. After a reset, the controller is disabled.

Reset Value = 0000 0000b

Table 71. 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 43.

No default value after reset.

Table 72. 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 43.
4-0	-	**Reserved** The values read from these bits are indeterminate. Do not set these bits.

No default value after reset.

Table 73. 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 82. CANIDM4 Register for V2.0 part A
CANIDM4 for V2.0 part A (S:C7h)
CAN Identifier Mask Registers 4

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

Bit Number	Bit Mnemonic	Description
7 - 3	-	**Reserved** The values read from these bits are indeterminate. Do not set these bits.
2	RTRMSK	**Remote transmission request Mask Value** 0 - comparison true forced. 1 - bit comparison enabled.
1	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
0	IDEMSK	**IDentifier Extension Mask Value** 0 - comparison true forced. 1 - bit comparison enabled.

Note: The ID Mask is only used for reception.

No default value after reset.

Table 83. CANIDM1 Register for V2.0 Part B
CANIDM1 for V2.0 Part B (S:C4h)
CAN Identifier Mask Registers 1

7	6	5	4	3	2	1	0
IDMSK 28	IDMSK 27	IDMSK 26	IDMSK 25	IDMSK 24	IDMSK 23	IDMSK 22	IDMSK 21

Bit Number	Bit Mnemonic	Description
7 - 0	IDMSK28:21	**IDentifier Mask Value** 0 - comparison true forced. 1 - bit comparison enabled. See Figure 43.

Note: The ID Mask is only used for reception.

No default value after reset.

Table 84. 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**[1] 0 - comparison true forced. 1 - bit comparison enabled. See Figure 43.

Note: 1. The ID Mask is only used for reception.

No default value after reset.

Table 85. 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 43.

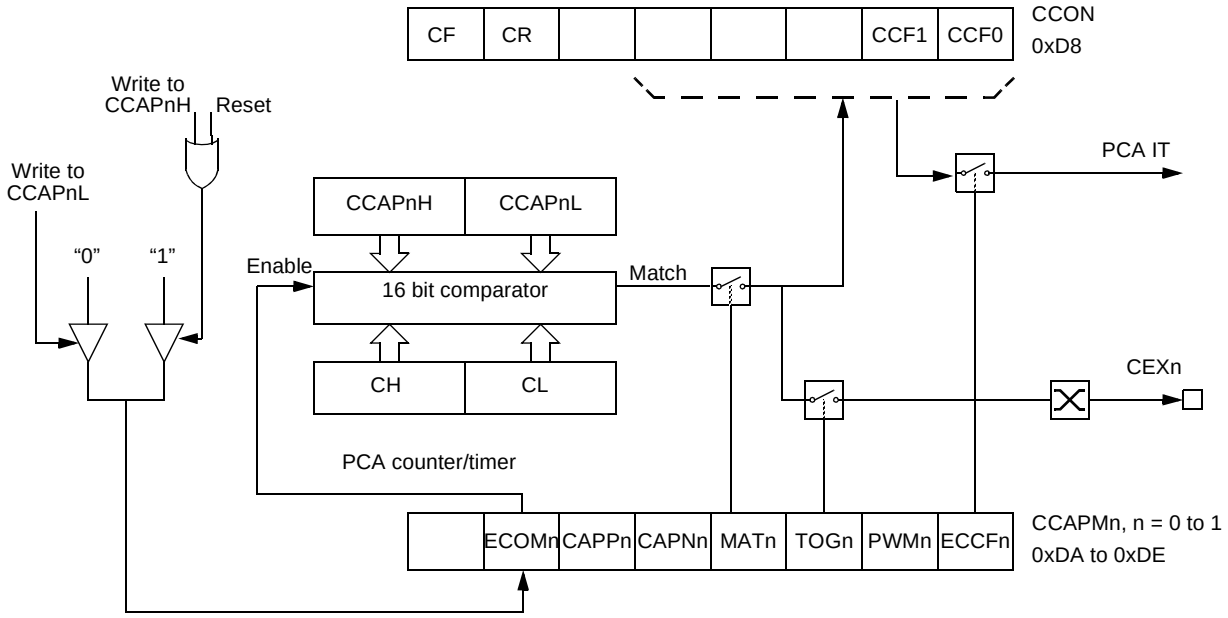
Note: The ID Mask is only used for reception.

No default value after reset.

High Speed Output Mode

In this mode the CEX output (on port 1) associated with the PCA module will toggle each time a match occurs between the PCA counter and the module's capture registers. To activate this mode the TOG, MAT, and ECOM bits in the module's CCAPMn SFR must be set.

Figure 50. PCA High Speed Output Mode



Pulse Width Modulator Mode

All the PCA modules can be used as PWM outputs. The output frequency depends on the source for the PCA timer. All the modules will have the same output frequency because they all share the PCA timer. The duty cycle of each module is independently variable using the module's capture register CCAPLn. When the value of the PCA CL SFR is less than the value in the module's CCAPLn SFR the output will be low, when it is equal to or greater than it, the output will be high. When CL overflows from FF to 00, CCAPLn is reloaded with the value in CCAPHn. the allows the PWM to be updated without glitches. The PWM and ECOM bits in the module's CCAPMn register must be set to enable the PWM mode.

Figure 51. PCA PWM Mode

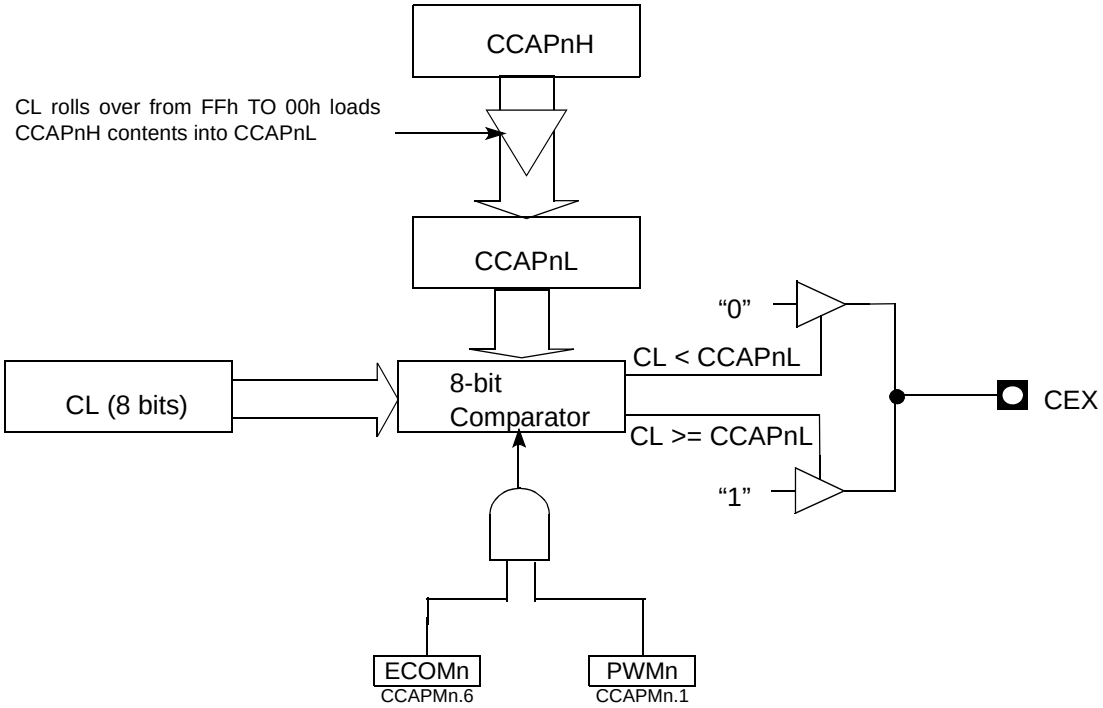


Table 100. CH Register

CH (S:F9h)
PCA Counter Register High value

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

Bit Number	Bit Mnemonic	Description
7:0	CH 7:0	High byte of Timer/Counter

Reset Value = 0000 00000b

Table 101. CL Register

CL (S:E9h)
PCA counter Register Low value

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

Bit Number	Bit Mnemonic	Description
7:0	CL0 7:0	Low byte of Timer/Counter

Reset Value = 0000 00000b

Figure 58. IEN1 Register
IEN1 (S:E8h)
Interrupt Enable Register

7	6	5	4	3	2	1	0
-	-	-	-		ETIM	EADC	ECAN

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	ETIM	**TImer overrun Interrupt Enable bit** Clear to disable the timer overrun interrupt. Set to enable the timer overrun interrupt.
1	EADC	**ADC Interrupt Enable bit** Clear to disable the ADC interrupt. Set to enable the ADC interrupt.
0	ECAN	**CAN Interrupt Enable bit** Clear to disable the CAN interrupt. Set to enable the CAN interrupt.

Reset Value = xxxx x000b
bit addressable

Table 112. IPH0 Register
IPH0 (B7h)
Interrupt High Priority Register

7	6	5	4	3	2	1	0
-	PPCH	PT2H	PSH	PT1H	PX1H	PT0H	PX0H

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	PPCH	**PCA Interrupt Priority Level Most Significant bit** **PPCH PPC Priority level** 0 0 Lowest 0 1 1 0 1 1 Highest priority
5	PT2H	**Timer 2 Overflow Interrupt High Priority bit** **PT2H PT2 Priority Level** 0 0 Lowest 0 1 1 0 1 1 Highest
4	PSH	**Serial Port High Priority bit** **PSH PS Priority Level** 0 0 Lowest 0 1 1 0 1 1 Highest
3	PT1H	**Timer 1 Overflow Interrupt High Priority bit** **PT1H PT1 Priority Level** 0 0 Lowest 0 1 1 0 1 1 Highest
2	PX1H	**External Interrupt 1 High Priority bit** **PX1H PX1 Priority Level** 0 0 Lowest 0 1 1 0 1 1 Highest
1	PT0H	**Timer 0 Overflow Interrupt High Priority bit** **PT0H PT0 Priority Level** 0 0 Lowest 0 1 1 0 1 1 Highest
0	PX0H	**External Interrupt 0 High Priority bit** **PX0H PX0 Priority Level** 0 0 Lowest 0 1 1 0 1 1 Highest

Reset Value = X000 0000b

Table 113. IPH1 Register
IPH1 (S:F7h)
Interrupt high priority Register 1

7	6	5	4	3	2	1	0
-	-	-	-		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	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
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 X000b