

8-bit MCU with CAN controller and Flash

1. Description

Part of the CANary™ family of microcontrollers dedicated to CAN network applications, the T89C51CC02 is a low pin count 8-bit Flash microcontroller.

While remaining fully compatible with the 80C51 it offers a superset of this standard microcontroller. In X2 mode a maximum external clock rate of 20 MHz reaches a 300 ns cycle time.

Besides the full CAN controller T89C51CC02 provides 16 Kbytes of Flash memory including In-system Programming (ISP), 2-Kbyte Boot Flash Memory, 2-Kbyte EEPROM and 512 bytes RAM.

Special attention is paid to the reduction of the electromagnetic emission of T89C51CC02.



2. Features

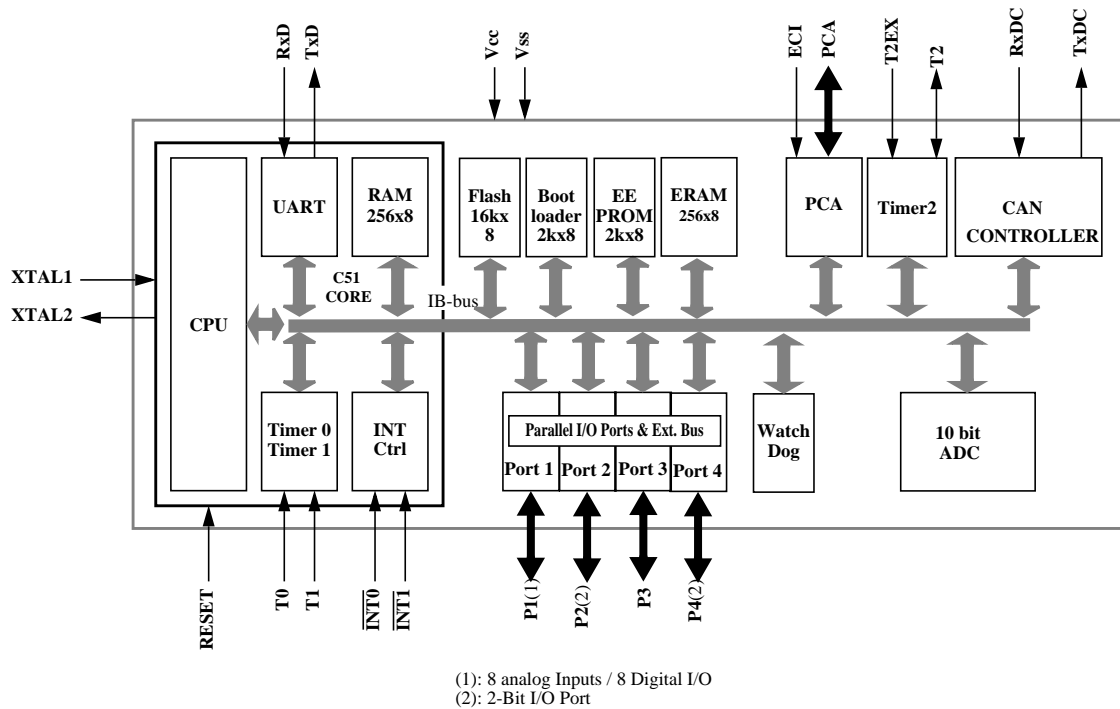
- 80C51 core architecture:
 - 256 bytes of on-chip RAM
 - 256 bytes of on-chip ERAM
 - 16 Kbytes of on-chip Flash memory
Read/Write cycle : 10k
Data Retention 10 years at 85°C
 - 2 Kbytes of on-chip Flash for Bootloader
 - 2 Kbytes of on-chip EEPROM
Read/Write cycle : 100k
 - 14-source 4-level interrupt
 - Three 16-bit timer/counter
 - Full duplex UART compatible 80C51
 - maximum crystal frequency 40 MHz. In X2 mode, 20 MHz (CPU core, 40 MHz)
 - three or four ports: 16 or 20 digital I/O lines
 - two-channel 16-bit PCA with:
 - PWM (8-bit)
 - High-speed output
 - Timer and edge capture
 - Double Data Pointer
 - 21-bit watchdog timer (including 7 programmable bits)
- A 10-bit resolution analog to digital converter (ADC) with 8 multiplexed inputs
 - Separate power supply for analog
- Full CAN controller:
 - Fully compliant with CAN standard rev 2.0 A and 2.0 B
 - Optimized structure for communication management (via SFR)
 - 4 independent message objects:
 - Each message object programmable on transmission or reception
 - individual tag and mask filters up to 29-bit identifier/message object
 - 8-byte cyclic data register (FIFO)/message object
 - 16-bit status & control register/message object
 - 16-bit Time-Stamping register/message object
 - CAN specification 2.0 part A or 2.0 part B programmable message objects
 - Access to message object control and data register via SFR
 - Programmable reception buffer length up to 4 message objects
 - Priority management of reception of hits on several message objects at the same time (Basic CAN Feature)
 - Priority management for transmission
 - message object overrun interrupt
- Supports
 - Time Triggered Communication.
 - Autobaud and Listening mode
 - Automatic reply mode programmable
- 1 Mbit/s maximum transfer rate at 8MHz* Crystal frequency in X2 mode.
- Readable error counters
- Programmable link to on-chip Timer for Time Stamping and Network synchronization
- Independent baud rate prescaler
- Data, Remote, Error and overload frame handling
- Power saving modes:
 - Idle mode
 - Power down mode
- Power supply: 5V +/- 10% ,3V +/- 10%
- Temperature range: Industrial (-40° to +85°C)
- Packages: PLCC28, SOIC28, (TSSOP28, SOIC24)**

T89C51CC02



* At BRP = 1 sampling point will be fixed.
** Ask for availability

3. Block Diagram



4. Pin Configuration

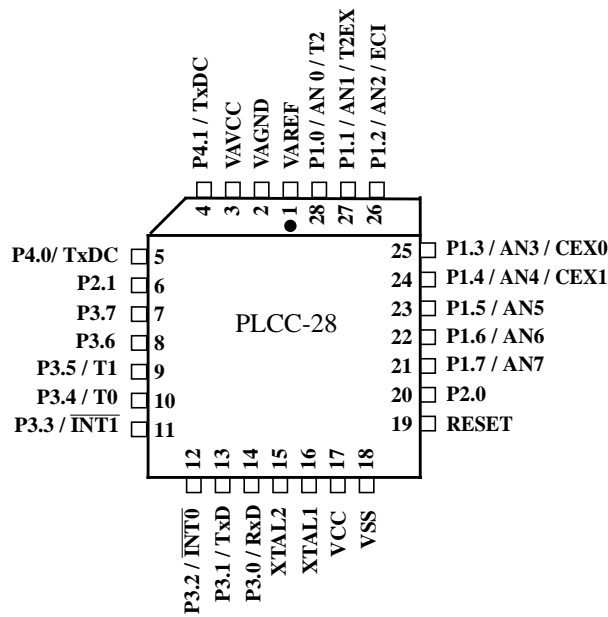
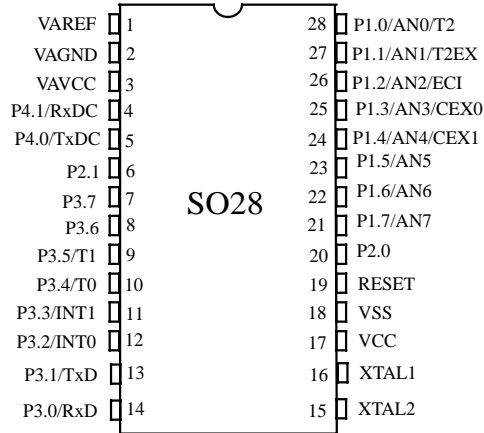


Table 1. Pin Description

Pin Name	Type	Description
VSS	GND	Circuit ground potential.
VCC		Supply voltage during normal, idle, and power-down operation.
VAREF		Reference Voltage for ADC
VAVCC		Supply Voltage for ADC
VAGND		Reference Ground for ADC / Analog Ground
P1.0:7	I/O	<p>Port 1: is an 8-bit bi-directional I/O port with internal pull-ups. Port 1 pins can be used for digital input/output or as analog inputs for the Analog Digital Converter (ADC). Port 1 pins that have 1's written to them are pulled high by the internal pull-up transistors and can be used as inputs in this state. As inputs, Port 1 pins that are being pulled low externally will be the source of current (IIL, on the datasheet) because of the internal pull-ups. Port 1 pins are assigned to be used as analog inputs via the ADCF register. As a secondary digital function, port 1 contains the Timer 2 external trigger and clock input; the PCA external clock input and the PCA module I/O.</p> <p>P1.0 / AN0 / T2 Analog input channel 0, External clock input for Timer/counter2.</p> <p>P1.1 / AN1 / T2EX Analog input channel 1, Trigger input for Timer/counter2.</p> <p>P1.2 / AN2 / ECI Analog input channel 2, PCA external clock input.</p> <p>Pin in the T89C51CC02 Port 1 can sink or source 5mA. It can drive CMOS inputs without external pull-ups.</p>
P2.0:1	I/O	<p>Port 2: Is an 2-bit bi-directional I/O port with internal pull-ups. Port 2 pins that have 1's written to them are pulled high by the internal pull-ups and can be used as inputs in this state. As inputs, Port 2 pins that are being pulled low externally will be a source of current (IIL, on the datasheet) because of the internal pull-ups. In the T89C51CC02 Port 2 can sink or source 5mA. It can drive CMOS inputs without external pull-ups.</p>
P3.0:7	I/O	<p>Port 3: Is an 8-bit bi-directional I/O port with internal pull-ups. Port 3 pins that have 1's written to them are pulled high by the internal pull-up transistors and can be used as inputs in this state. As inputs, Port 3 pins that are being pulled low externally will be a source of current (IIL, on the datasheet) because of the internal pull-ups. The output latch corresponding to a secondary function must be programmed to one for that function to operate. The secondary functions are assigned to the pins of port 3 as follows:</p> <p>P3.0 / RxD: Receiver data input (asynchronous) or data input/output (synchronous) of the serial interface</p> <p>P3.1 / TxD: Transmitter data output (asynchronous) or clock output (synchronous) of the serial interface</p> <p>P3.2 / INT0: External interrupt 0 input / timer 0 gate control input</p> <p>P3.3 / INT1: External interrupt 1 input / timer 1 gate control input</p> <p>P3.4 / T0: Timer 0 counter input</p> <p>P3.5 / T1: Timer 1 counter input</p> <p>P3.6 P3.7</p> <p>In the T89C51CC02 Port 3 can sink or source 5mA. It can drive CMOS inputs without external pull-ups.</p>

Pin Name	Type	Description
P4.0:1	I/O	<p>Port 4: Is an 2-bit bi-directional I/O port with internal pull-ups. Port 4 pins that have 1's written to them are pulled high by the internal pull-ups and can be used as inputs in this state. As inputs, Port 4 pins that are being pulled low externally will be a source of current (IIL, on the datasheet) because of the internal pull-up transistor. The output latch corresponding to a secondary function RxDC must be programmed to one for that function to operate. The secondary functions are assigned to the two pins of port 4 as follows:</p> <p>P4.0 / TxDC: Transmitter output of CAN controller</p> <p>P4.1 / RxDC: Receiver input of CAN controller. In the T89C51CC02 Port 4 can sink or source 5mA. It can drive CMOS inputs without external pull-ups.</p>
RESET	I/O	<p>Reset: A high level on this pin during two machine cycles while the oscillator is running resets the device. An internal pull-down resistor to VSS permits power-on reset using only an external capacitor to VCC.</p>
XTAL1	I	<p>XTAL1: Input of the inverting oscillator amplifier and input of the internal clock generator circuits. To drive the device from an external clock source, XTAL1 should be driven, while XTAL2 is left unconnected. To operate above a frequency of 16 MHz, a duty cycle of 50% should be maintained.</p>
XTAL2	O	<p>XTAL2: Output from the inverting oscillator amplifier.</p>

4.1. I/O Configurations

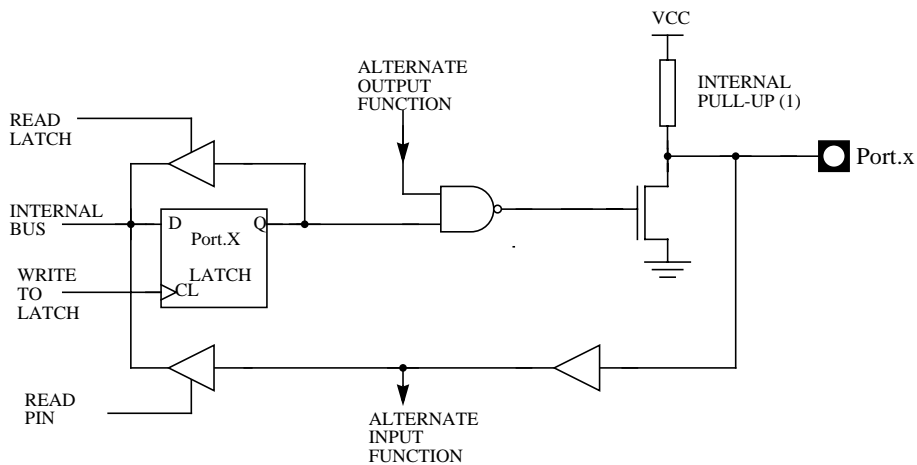
Each Port SFR operates via type-D latches, as illustrated in Figure 1 for Ports 3 and 4. A CPU "write to latch" signal initiates transfer of internal bus data into the type-D latch. A CPU "read latch" signal transfers the latched Q output onto the internal bus. Similarly, a "read pin" signal transfers the logical level of the Port pin. Some Port data instructions activate the "read latch" signal while others activate the "read pin" signal. Latch instructions are referred to as Read-Modify-Write instructions. Each I/O line may be independently programmed as input or output.

4.2. Port Structure

Figure 1 shows the structure of Ports 1 and 3, which have internal pull-ups. An external source can pull the pin low. Each Port pin can be configured either for general-purpose I/O or for its alternate input output function.

To use a pin for general-purpose output, set or clear the corresponding bit in the Px register (x=1,3 or 4). To use a pin for general purpose input, set the bit in the Px register. This turns off the output FET drive.

To configure a pin for its alternate function, set the bit in the Px register. When the latch is set, the "alternate output function" signal controls the output level (see Figure 1). The operation of Ports 1, 3 and 4 is discussed further in "quasi-Bidirectional Port Operation" paragraph.



NOTE:

1. The internal pull-up can be disabled on P1 when analog function is selected.

Figure 1. Port Structure

4.3. Read-Modify-Write Instructions

Some instructions read the latch data rather than the pin data. The latch based instructions read the data, modify the data and then rewrite the latch. These are called "Read-Modify-Write" instructions. Below is a complete list of these special instructions (see Table 2). When the destination operand is a Port or a Port bit, these instructions read the latch rather than the pin:

Table 2. Read-Modify-Write Instructions

Instruction	Description	Example
ANL	logical AND	ANL P1, A
ORL	logical OR	ORL P2, A
XRL	logical EX-OR	XRL P3, A
JBC	jump if bit = 1 and clear bit	JBC P1.1, LABEL
CPL	complement bit	CPL P3.0
INC	increment	INC P2
DEC	decrement	DEC P2
DJNZ	decrement and jump if not zero	DJNZ P3, LABEL
MOV Px.y, C	move carry bit to bit y of Port x	MOV P1.5, C
CLR Px.y	clear bit y of Port x	CLR P2.4
SET Px.y	set bit y of Port x	SET P3.3

It is not obvious the last three instructions in this list are Read-Modify-Write instructions. These instructions read the port (all 8 bits), modify the specifically addressed bit and write the new byte back to the latch. These Read-Modify-Write instructions are directed to the latch rather than the pin in order to avoid possible misinterpretation of voltage (and therefore, logic) levels at the pin. For example, a Port bit used to drive the base of an external bipolar transistor can not rise above the transistor's base-emitter junction voltage (a value lower than V_{IL}). With a logic one written to the bit, attempts by the CPU to read the Port at the pin are misinterpreted as logic zero. A read of the latch rather than the pins returns the correct logic-one value.

4.4. Quasi-Bidirectional Port Operation

Port 1, Port 3 and Port 4 have fixed internal pull-ups and are referred to as "quasi-bidirectional" Ports. When configured as an input, the pin impedance appears as logic one and sources current in response to an external logic zero condition. Resets write logic one to all Port latches. If logical zero is subsequently written to a Port latch, it can be returned to input conditions by a logical one written to the latch.

NOTE:

Port latch values change near the end of Read-Modify-Write instruction cycles. Output buffers (and therefore the pin state) update early in the instruction after Read-Modify-Write instruction cycle.

Logical zero-to-one transitions in Port 1, Port 3 and Port 4 use an additional pull-up (p1) to aid this logic transition see Figure. This increases switch speed. This extra pull-up sources 100 times normal internal circuit current during 2 oscillator clock periods. The internal pull-ups are field-effect transistors rather than linear resistors. Pull-ups consist of three p-channel FET (pFET) devices. A pFET is on when the gate senses logical zero and off when the gate senses logical one. pFET #1 is turned on for two oscillator periods immediately after a zero-to-one transition in the Port latch. A logical one at the Port pin turns on pFET #3 (a weak pull-up) through the inverter. This inverter and pFET pair form a latch to drive logical one. pFET #2 is a very weak pull-up switched on whenever the associated nFET is switched off. This is traditional CMOS switch convention. Current strengths are 1/10 that of pFET #3.

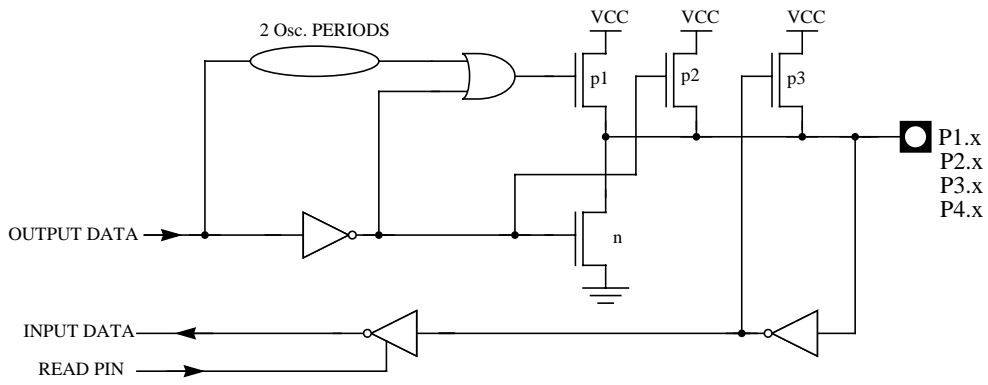


Figure 2. Internal Pull-Up Configurations

5. SFR Mapping

The Special Function Registers (SFRs) of the T89C51CC02 fall into the following categories:

Table 3. C51 Core SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
ACC	E0h	Accumulator								
B	F0h	B Register								
PSW	D0h	Program Status Word								
SP	81h	Stack Pointer LSB of SPX								
DPL	82h	Data Pointer Low byte LSB of DPTR								
DPH	83h	Data Pointer High byte MSB of DPTR								

Table 4. I/O Port SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
P1	90h	Port 1								
P2	A0h	Port 2 (x2)								
P3	B0h	Port 3								
P4	C0h	Port 4 (x2)								

Table 5. Timers SFRs

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
T2CON	C8h	Timer/Counter 2 control	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2#	CP/RL2#
T2MOD	C9h	Timer/Counter 2 Mode	-	-	-	-	-	-	T2OE	DCEN
RCAP2H	CBh	Timer/Counter 2 Reload/Capture High byte								
RCAP2L	CAh	Timer/Counter 2 Reload/Capture Low byte								
WDTRST	A6h	WatchDog Timer Reset								
WDTPRG	A7h	WatchDog Timer Program	-	-	-	-	-	S2	S1	S0

Table 6. Serial I/O Port SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
SCON	98h	Serial Control	FE/SM0	SM1	SM2	REN	TB8	RB8	TI	RI
SBUF	99h	Serial Data Buffer								
SADEN	B9h	Slave Address Mask								
SADDR	A9h	Slave Address								

Table 7. PCA SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
CCON	D8h	PCA Timer/Counter Control	CF	CR	-	CCF4	CCF3	CCF2	CCF1	CCF0
CMOD	D9h	PCA Timer/Counter Mode	CIDL	WDTE	-	-	-	CPS1	CPS0	ECF
CL	E9h	PCA Timer/Counter Low byte								
CH	F9h	PCA Timer/Counter High byte								
CCAPM0	DAh	PCA Timer/Counter Mode 0	-	ECOM0	CAPP0	CAP0	MAT0	TOG0	PWM0	ECCF0
CCAPM1	DBh	PCA Timer/Counter Mode 1	-	ECOM1	CAPP1	CAP1	MAT1	TOG1	PWM1	ECCF1
CCAP0H	FAh	PCA Compare Capture Module 0 H	CCAP0H7	CCAP0H6	CCAP0H5	CCAP0H4	CCAP0H3	CCAP0H2	CCAP0H1	CCAP0H0
CCAP1H	FBh	PCA Compare Capture Module 1 H	CCAP1H7	CCAP1H6	CCAP1H5	CCAP1H4	CCAP1H3	CCAP1H2	CCAP1H1	CCAP1H0
CCAP0L	EAh	PCA Compare Capture Module 0 L	CCAP0L7	CCAP0L6	CCAP0L5	CCAP0L4	CCAP0L3	CCAP0L2	CCAP0L1	CCAP0L0
CCAP1L	EBh	PCA Compare Capture Module 1 L	CCAP1L7	CCAP1L6	CCAP1L5	CCAP1L4	CCAP1L3	CCAP1L2	CCAP1L1	CCAP1L0

Table 8. Interrupt SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
IEN0	A8h	Interrupt Enable Control 0	EA	AC	ET2	ES	ET1	EX1	ET0	EX0
IEN1	E8h	Interrupt Enable Control 1	-	-	-	-	-	ETIM	EADC	ECAN
IPL0	B8h	Interrupt Priority Control Low 0	-	PPC	PT2	PS	PT1	PX1	PT0	PX0
IPH0	B7h	Interrupt Priority Control High 0	-	PPCH	PT2H	PSH	PT1H	PX1H	PT0H	PX0H
IPL1	F8h	Interrupt Priority Control Low 1	-	-	-	-	-	POVRL	PADCL	PCANL
IPH1	F7h	Interrupt Priority Control High1	-	-	-	-	-	POVRH	PADCH	PCANH

Table 9. ADC SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
ADCON	F3h	ADC Control	-	PSIDLE	ADEN	ADEOC	ADSST	SCH2	SCH1	SCH0
ADCF	F6h	ADC Configuration	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
ADCLK	F2h	ADC Clock	-	-	-	PRS4	PRS3	PRS2	PRS1	PRS0
ADDH	F5h	ADC Data High byte	ADAT9	ADAT8	ADAT7	ADAT6	ADAT5	ADAT4	ADAT3	ADAT2
ADDL	F4h	ADC Data Low byte	-	-	-	-	-	-	ADAT1	ADAT0

Table 10. CAN SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
CANGCON	ABh	CAN General Control	ABRQ	OVRQ	TTC	SYNCTTC	AUT-BAUD	TEST	ENA	GRES
CANGSTA	AAh	CAN General Status	-	OVRQ	-	TBSY	RBSY	ENFG	BOFF	ERRP
CANGIT	9Bh	CAN General Interrupt	CANIT	-	OVRTIM	OVRBUF	SERG	CERG	FERG	AERG
CANBT1	B4h	CAN Bit Timing 1	-	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	-
CANBT2	B5h	CAN Bit Timing 2	-	SJW1	SJW2	-	PRS2	PRS1	PRS0	-
CANBT3	B6h	CAN Bit Timing 3	-	PHS22	PHS21	PHS20	PHS12	PHS11	PHS10	SMP
CANEN	CFh	CAN Enable Channel byte	-	-	-	-	ENCH3	ENCH2	ENCH1	ENCH0
CANGIE	C1h	CAN General Interrupt Enable	-	-	ENRX	ENTX	ENER	ENBUF	-	-
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

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
CANTIML	ACh	CAN Timer low	CANTIM 7	CANTIM 6	CANTIM 5	CANTIM 4	CANTIM 3	CANTIM 2	CANTIM 1	CANTIM 0
CANSTMH	AFh	CAN Timer Stamp high	TIMSTMP 15	TIMSTMP 14	TIMSTMP 13	TIMSTMP 12	TIMSTMP 11	TIMSTMP 10	TIMSTMP 9	TIMSTMP 8
CANSTML	A Eh	CAN Timer Stamp low	TIMSTMP 7	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
CANCONH	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	RB0TAF
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
CANIDM2	C5h	CAN Identifier Mask byte 2(PartA) CAN Identifier Mask byte 2(PartB)	IDMSK2 IDMSK20	IDMSK1 IDMSK19	IDMSK0 IDMSK18	- IDMSK17	- IDMSK16	- IDMSK15	- IDMSK14	- IDMSK13
CANIDM3	C6h	CAN Identifier Mask byte 3(PartA) CAN Identifier Mask byte 3(PartB)	- IDMSK12	- IDMSK11	- IDMSK10	- IDMSK9	- IDMSK8	- IDMSK7	- IDMSK6	- IDMSK5
CANIDM4	C7h	CAN Identifier Mask byte 4(PartA) CAN Identifier Mask byte 4(PartB)	- IDMSK4	- IDMSK3	- IDMSK2	- IDMSK1	- IDMSK0	RTRMSK	-	IDEMSK

Table 11. Other SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
PCON	87hh	Power Control	SMOD1	SMOD0	-	POF	GF1	GF0	PD	IDL
AUXR1	A2h	Auxiliary Register 1	-	-	ENBOOT	-	GF3	-	-	DPS
CKCON	8Fh	Clock Control	CANX2	WDX2	PCAX2	SIX2	T2X2	T1X2	T0X2	X2
FCON	D1h	FLASH Control	FPL3	FPL2	FPL1	FPL0	FPS	FMOD1	FMOD0	FBUSY
EECON	D2h	EEPROM Contol	EEPL3	EEPL2	EEPL1	EEPL0	-	-	EEE	EEBUSY

Table 12. SFR's mapping

	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					FFh
F0h	B 0000 0000		ADCLK xx00 0000	ADCON x000 0000	ADDL 0000 0000	ADDH 0000 0000	ADCF 0000 0000	IPH1 xxxx x000	F7h
E8h	IEN1 xxxx x000	CL 0000 0000	CCAP0L 0000 0000	CCAP1L 0000 0000					EFh
E0h	ACC 0000 0000								E7h
D8h	CCON 00xx xx00	CMOD 00xx x000	CCAPM0 x000 0000	CCAPM1 x000 0000					DFh
D0h	PSW 0000 0000	FCON 0000 0000	EECON xxxx xx00						D7h
C8h	T2CON 0000 0000	T2MOD xxxx xx00	RCAP2L 0000 0000	RCAP2H 0000 0000	TL2 0000 0000	TH2 0000 0000		CANEN2 xxxx 0000	CFh
C0h	P4 xxxx xx11	CANGIE 0000 0000		CANIE2 xxx 0000	CANIDM1 xxxx xxxx	CANIDM2 xxxx xxxx	CANIDM3 xxxx xxxx	CANIDM4 xxxx xxxx	C7h
B8h	IPL0 x000 0000	SADEN 0000 0000		CANSIT2 xxxx 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 x0x0 0000	CANGCON 0000 x000	CANTIML 0000 0000	CANTIMH 0000 0000	CANSTMPL 0000 0000	CANSTMPH 0000 0000	AFh
A0h	P2 xxxx xx11	CANTCON 0000 0000	AUXR1 0000 0000	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			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		CKCON 0000 0000	8Fh
80h		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	

Note:

2. These registers are bit-addressable.

Sixteen addresses in the SFR space are both byte-addressable and bit-addressable. The bit-addressable SFR's are those whose address ends in 0 and 8. The bit addresses, in this area, are 0x80 through to 0xFF.

6. Clock

6.1. Introduction

The T89C51CC02 core needs only 6 clock periods per machine cycle. This feature, called "X2", provides the following advantages:

- Divides frequency crystals by 2 (cheaper crystals) while keeping the same CPU power.
- Saves power consumption while keeping the same CPU power (oscillator power saving).
- Saves power consumption by dividing dynamic operating frequency by 2 in operating and idle modes.
- Increases CPU power by 2 while keeping the same crystal frequency.

In order to keep the original C51 compatibility, a divider-by-2 is inserted between the XTAL1 signal and the main clock input of the core (phase generator). This divider may be disabled by the software.

An extra feature is available for selected hardware in the X2 mode. This feature allows starting of the CPU in the X2 mode, without starting in the standard mode.

The hardware CPU X2 mode can be read and write via IAP (SetX2mode, ClearX2mode, ReadX2mode), see In-System Programming section.

These IAPs are detailed in the "In-System Programming" section.

6.2. Description

The clock for the whole circuit and peripheral is first divided by two before being used by the CPU core and peripherals. This allows any cyclic ratio to be accepted on the XTAL1 input. In X2 mode, as this divider is bypassed, the signals on XTAL1 must have a cyclic ratio between 40 to 60%. Figure 3. shows the clock generation block diagram. The X2 bit is validated on the XTAL1÷2 rising edge to avoid glitches when switching from the X2 to the STD mode. Figure 4 shows the mode switching waveforms.

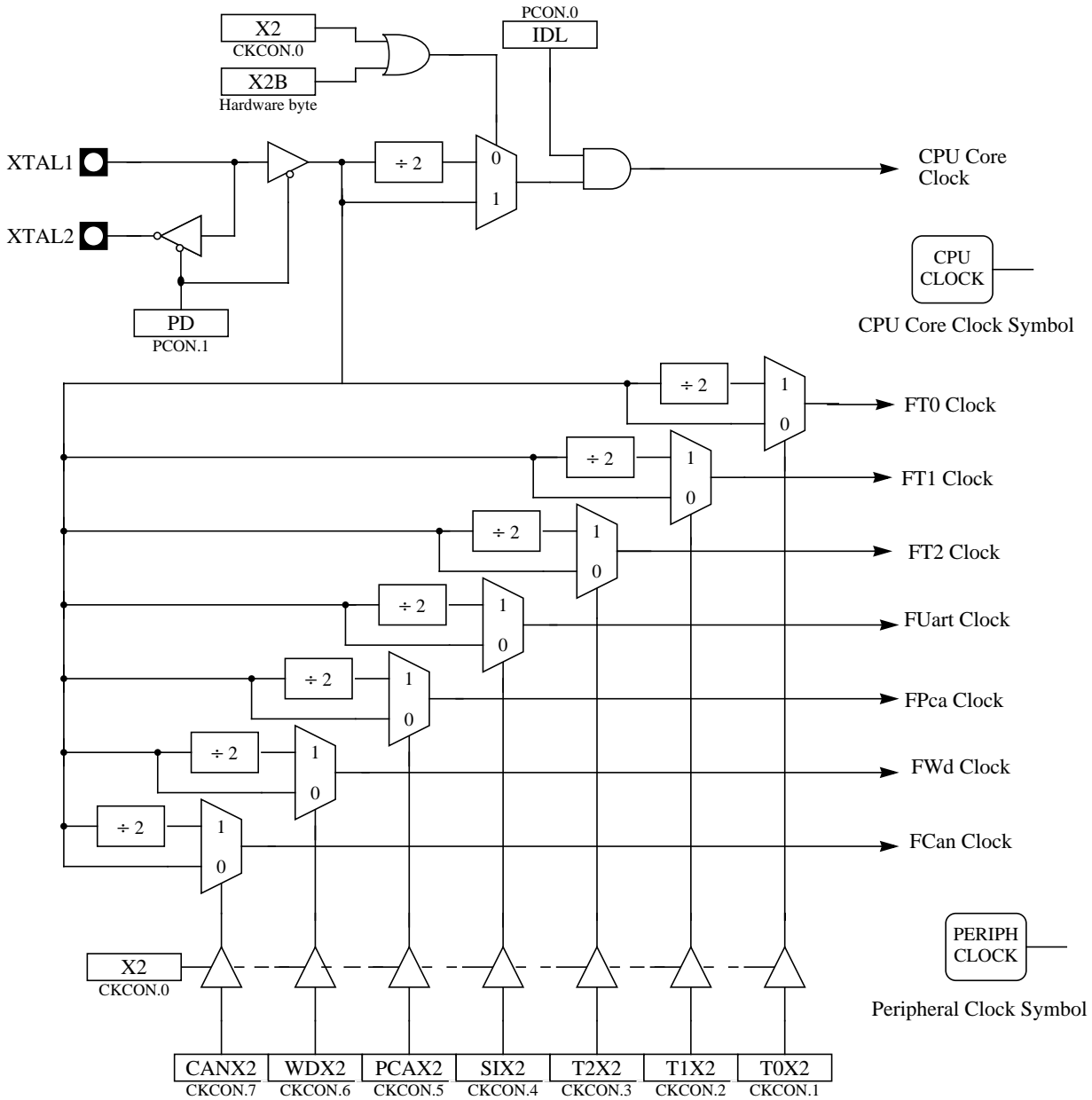


Figure 3. Clock CPU Generation Diagram

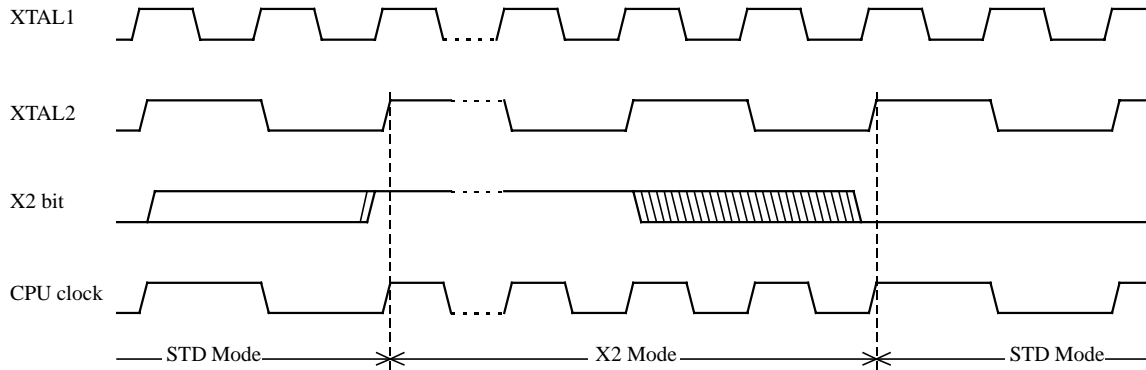


Figure 4. Mode Switching Waveforms

The X2 bit in the CKCON register (See Table 5) allows switching from 12 clock cycles per instruction to 6 clock cycles and vice versa. At reset, the standard speed is activated (STD mode). Setting this bit activates the X2 feature (X2 mode).

CAUTION

In order to prevent any incorrect operation while operating in the X2 mode, users must be aware that all peripherals using the clock frequency as a time reference (UART, timers...) will have their time reference divided by two. For example a free running timer generating an interrupt every 20 ms will then generate an interrupt every 10 ms. A UART with a 4800 baud rate will have a 9600 baud rate.

6.3. Register

CKCON (S:8Fh)

Clock Control Register

7	6	5	4	3	2	1	0
CANX2	WDX2	PCAX2	SIX2	T2X2	T1X2	T0X2	X2

Bit Number	Bit Mnemonic	Description
7	CANX2	CAN clock (1) Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
6	WDX2	Watchdog clock (1) Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
5	PCAX2	Programmable Counter Array clock (1) Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
4	SIX2	Enhanced UART clock (MODE 0 and 2) (1) Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
3	T2X2	Timer2 clock (1) Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
2	T1X2	Timer1 clock (1) Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
1	T0X2	Timer0 clock (1) Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
0	X2	CPU clock Clear to select 12 clock periods per machine cycle (STD mode) for CPU and all the peripherals. Set to select 6 clock periods per machine cycle (X2 mode) and to enable the individual peripherals "X2"bits.

NOTE:

1. This control bit is validated when the CPU clock bit X2 is set; when X2 is low, this bit has no effect.

Reset Value = 0000 0000b

Figure 5. CKCON Register

7. Program/Code Memory

7.1. Introduction

The T89C51CC02 implement 16 Kbytes of on-chip program/code memory. The FLASH memory increases EPROM and ROM functionality by in-circuit electrical erasure and programming. Thanks to the internal charge pump, the high voltage needed for programming or erasing FLASH cells is generated on-chip using the standard VDD voltage. Thus, the FLASH Memory can be programmed using only one voltage and allows in application software programming commonly known as IAP. Hardware programming mode is also available using specific programming tool.

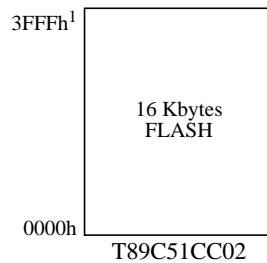


Figure 6. Program/Code Memory Organization

7.2. FLASH Memory Architecture

T89C51CC02 features two on-chip flash memories:

- Flash memory FM0:
containing 16 Kbytes of program memory (user space) organized into 128 byte pages,
- Flash memory FM1:
2 Kbytes for boot loader and Application Programming Interfaces (API).

The FM0 supports both parallel programming and Serial In-System Programming (ISP) whereas FM1 supports only parallel programming by programmers. The ISP mode is detailed in the "In-System Programming" section.

All Read/Write access operations on FLASH Memory by user application are managed by a set of API described in the "In-System Programming" section.

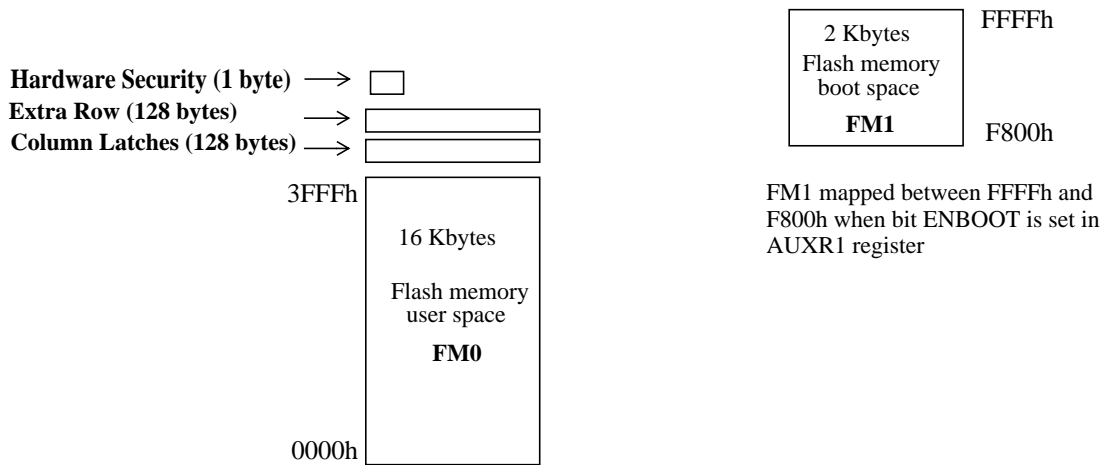


Figure 7. Flash memory architecture

7.2.1. FM0 Memory Architecture

The flash memory is made up of 4 blocks (see Figure 7):

1. The memory array (user space) 16 Kbytes
2. The Extra Row
3. The Hardware security bits
4. The column latch registers

7.2.1.1. User Space

This space is composed of a 16 Kbytes FLASH memory organized in 128 pages of 128 bytes. It contains the user's application code.

7.2.1.2. Extra Row (XRow)

This row is a part of FM0 and has a size of 128 bytes. The extra row may contain information for boot loader usage.

7.2.1.3. Hardware security space

The Hardware security space is a part of FM0 and has a size of 1 byte.

The 4 MSB can be read/written by software, the 4 LSB can only be read by software and written by hardware in parallel mode.

7.2.1.4. Column latches

The column latches, also part of FM0, have a size of full page (128 bytes).

The column latches are the entrance buffers of the three previous memory locations (user array, XROW and Hardware security byte).

7.3. Overview of FM0 operations

The CPU interfaces to 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)
- Select the flash memory FM0/FM1.

7.3.1. 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 this bit takes precedence on the EXTRAM bit in AUXR 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 13. A MOVC instruction is then used for reading these spaces.

Table 13. FM0 blocks select bits

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

7.3.2. 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 5 followed by A. Table 14 summarizes the memory spaces to program according to FMOD1:0 bits.

Table 14. Programming spaces

	Write to FCON				Operation
	FPL3:0	FPS	FMOD1	FMOD0	
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
Security Space	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

The FLASH memory enters a busy state as soon as programming is launched. In this state, the memory is no more available for fetching code. Thus to avoid any erratic execution during programming, the CPU enters Idle mode. Exit is automatically performed at the end of programming.

Caution:

Interrupts that may occur during programming time must be disable to avoid any spurious exit of the idle mode.

7.3.3. Status of the flash memory

The bit FBUSY in FCON register is used to indicate the status of programming.

FBUSY is set when programming is in progress.

7.3.4. Selecting FM1/FM1

The bit ENBOOT in AUXR1 register is used to choose between FM0 and FM1 mapped up to F800h.

7.3.5. Loading the Column Latches

Any number of data from 1 byte to 128 bytes can be loaded in the column latches. This provides the capability to program the whole memory by byte, by page or by any number of bytes in a page.

When programming is launched, an automatic erase of the locations loaded in the column latches is first performed, then programming is effectively done. Thus no page or block erase is needed and only the loaded data are programmed in the corresponding page.

The following procedure is used to load the column latches and is summarized in Figure 8:

- Map the column latch space by setting FPS bit.
- Load the DPTR with the address to load.
- Load Accumulator register with the data to load.
- Execute the MOVX @DPTR, A instruction.
- If needed loop the three last instructions until the page is completely loaded.

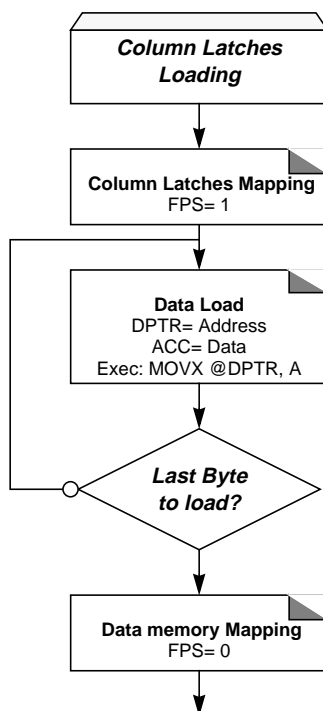


Figure 8. Column Latches Loading Procedure

7.3.6. Programming the FLASH Spaces

User

The following procedure is used to program the User space and is summarized in Figure 9:

- Load data in the column latches from address 0000h to 3FFFh¹.
- Disable the interrupts.
- Launch the programming by writing the data sequence 50h followed by A0h in FCON register. The end of the programming indicated by the FBUSY flag cleared.
- Enable the interrupts.

Note:

1. The last page address used when loading the column latch is the one used to select the page programming address.

Extra Row

The following procedure is used to program the Extra Row space and is summarized in Figure 9:

- Load data in the column latches from address FF80h to FFFFh.
- Disable the interrupts.
- Launch the programming by writing the data sequence 52h followed by A2h in FCON register. The end of the programming indicated by the FBUSY flag cleared.
- Enable the interrupts.

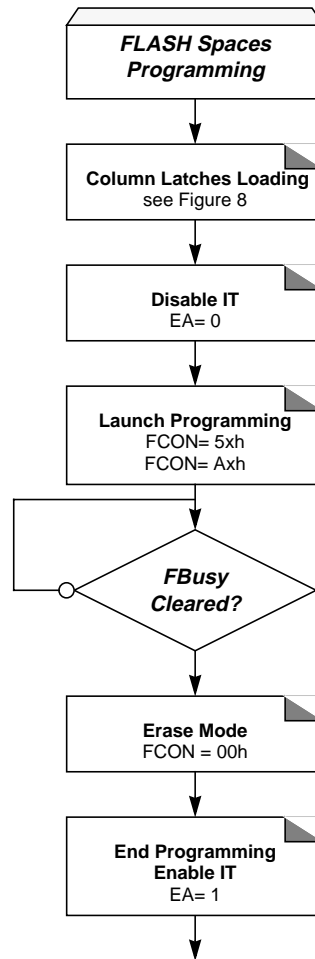


Figure 9. Flash and Extra row Programming Procedure

Hardware Security

The following procedure is used to program the Hardware Security space and is summarized in Figure 10:

- Set FPS and map Hardware byte (FCON = 0x0C)
- Disable the interrupts.
- Load DPTR at address 0000h.
- Load Accumulator register with the data to load.
- Execute the MOVX @DPTR, A instruction.
- Launch the programming by writing the data sequence 54h followed by A4h in FCON register. The end of the programming indicated by the FBusy flag cleared.
- Enable the interrupts.

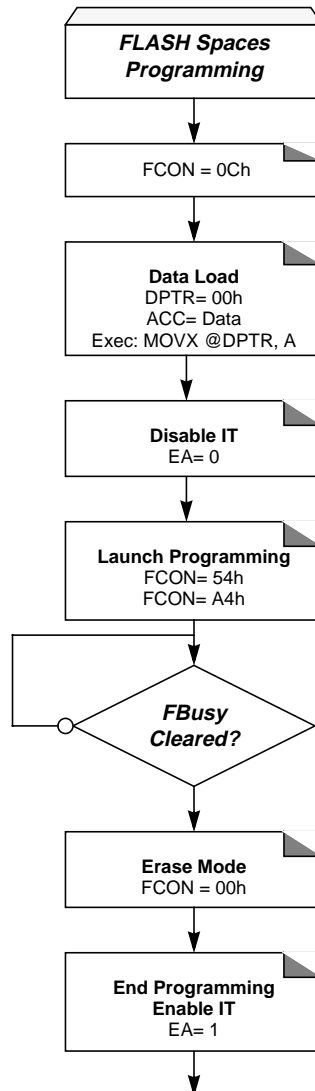


Figure 10. Hardware Programming Procedure

7.3.7. Reading the FLASH Spaces

User

The following procedure is used to read the User space and is summarized in Figure 11:

- Map the User space by writing 00h in FCON register.
- Read one byte in Accumulator by executing `MOVC A,@A+DPTR` with `A= 0` & `DPTR= 0000h` to `FFFFh`.

Extra Row

The following procedure is used to read the Extra Row space and is summarized in Figure 11:

- Map the Extra Row space by writing 02h in FCON register.
- Read one byte in Accumulator by executing `MOVC A,@A+DPTR` with `A= 0` & `DPTR= FF80h` to `FFFFh`.

Hardware Security

The following procedure is used to read the Hardware Security space and is summarized in Figure 11:

- Map the Hardware Security space by writing 04h in FCON register.
- Read the byte in Accumulator by executing `MOVC A,@A+DPTR` with `A= 0` & `DPTR= 0000h`.

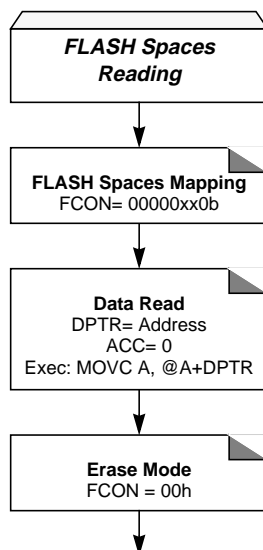


Figure 11. Reading Procedure

7.4. Registers

FCON (S:D1h)

FLASH Control Register

7	6	5	4	3	2	1	0
FPL3	FPL2	FPL1	FPL0	FPS	FMOD1	FMOD0	FBUSY

Bit Number	Bit Mnemonic	Description
7-4	FPL3:0	Programming Launch Command Bits Write 5Xh followed by AXh to launch the programming according to FMOD1:0. (see Table 14.)
3	FPS	FLASH Map Program Space Set to map the column latch space in the data memory space. Clear to re-map the data memory space.
2-1	FMOD1:0	FLASH Mode See Table 13 or Table 14.
0	FBUSY	FLASH Busy Set by hardware when programming is in progress. Clear by hardware when programming is done. Can not be cleared by software.

Reset Value= 0000 0000b

Figure 12. FCON Register

8. Data Memory

8.1. Introduction

The T89C51CC02 provides data memory access in two different spaces:

1. The internal space mapped in three separate segments:
 - the lower 128 bytes RAM segment.
 - the upper 128 bytes RAM segment.
 - the expanded 256 bytes RAM segment (ERAM).

A fourth internal segment is available but dedicated to Special Function Registers, SFRs, (addresses 80h to FFh) accessible by direct addressing mode.

Figure 13 shows the internal data memory spaces organization.

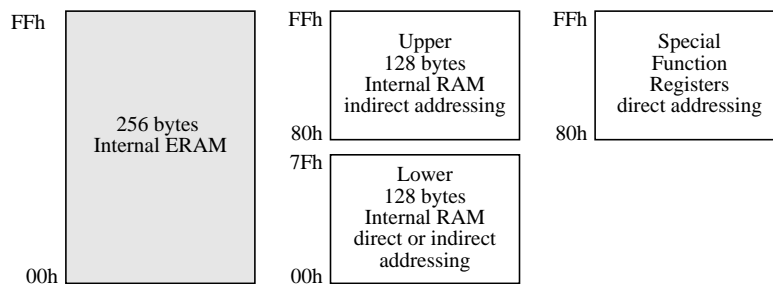


Figure 13. Internal Data Memory Organization

8.2. Internal Space

8.2.1. Lower 128 Bytes RAM

The lower 128 bytes of RAM (see Figure 13) are accessible from address 00h to 7Fh using direct or indirect addressing modes. The lowest 32 bytes are grouped into 4 banks of 8 registers (R0 to R7). Two bits RS0 and RS1 in PSW register (see Figure 16) select which bank is in use according to Table 15. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing, and can be used for context switching in interrupt service routines.

Table 15. Register Bank Selection

RS1	RS0	Description
0	0	Register bank 0 from 00h to 07h
0	1	Register bank 0 from 08h to 0Fh
1	0	Register bank 0 from 10h to 17h
1	1	Register bank 0 from 18h to 1Fh

The next 16 bytes above the register banks form a block of bit-addressable memory space. The C51 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00h to 7Fh.

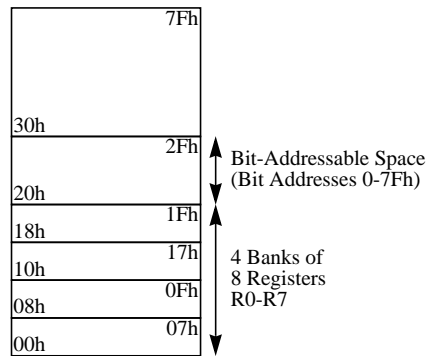


Figure 14. Lower 128 bytes Internal RAM Organization

8.2.2. Upper 128 Bytes RAM

The upper 128 bytes of RAM are accessible from address 80h to FFh using only indirect addressing mode.

8.2.3. Expanded RAM

The on-chip 256 bytes of expanded RAM (ERAM) are accessible from address 0000h to FFh using indirect addressing mode through MOVX instructions.

Caution:

Lower 128 bytes RAM, Upper 128 bytes RAM, and expanded RAM are made of volatile memory cells. This means that the RAM content is indeterminate after power-up and must then be initialized properly.

8.3. Dual Data Pointer

8.3.1. 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 17) is used to select whether DPTR is the data pointer 0 or the data pointer 1 (see Figure 15).

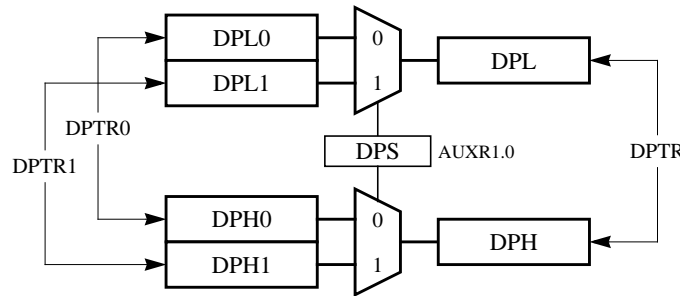


Figure 15. Dual Data Pointer Implementation

8.3.2. Application

Software can take advantage of the additional data pointers to both increase speed and reduce code size, for example, block operations (copy, compare, search ...) 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. Latest C compiler take also advantage of this feature by providing enhanced algorithm libraries.

The INC instruction is a short (2 bytes) and fast (6 CPU clocks) 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 of entry state unless an extra INC AUXR1 is added

```

```

AUXR1    EQU    0A2h

move:    mov     DPTR,#SOURCE          ; address of SOURCE
         inc     AUXR1                ; switch data pointers
         mov     DPTR,#DEST          ; address of DEST
mv_loop: inc     AUXR1                ; switch data pointers
         movx   A,@DPTR              ; get a byte from SOURCE
         inc     DPTR                ; increment SOURCE address
         inc     AUXR1                ; switch data pointers
         movx   @DPTR,A              ; write the byte to DEST
         inc     DPTR                ; increment DEST address
         jnz    mv_loop              ; check for NULL terminator

end_move:

```

8.4. Registers

PSW (S:8Eh)

Program Status Word Register.

7	6	5	4	3	2	1	0
CY	AC	F0	RS1	RS0	OV	F1	P

Bit Number	Bit Mnemonic	Description
7	CY	Carry Flag Carry out from bit 1 of ALU operands.
6	AC	Auxiliary Carry Flag Carry out from bit 1 of addition operands.
5	F0	User Definable Flag 0.
4-3	RS1:0	Register Bank Select Bits Refer to Table 15 for bits description.
2	OV	Overflow Flag Overflow set by arithmetic operations.
1	F1	User Definable Flag 1.
0	P	Parity Bit Set when ACC contains an odd number of 1's. Cleared when ACC contains an even number of 1's.

Reset Value= 0000 0000b

Figure 16. PSW Register

AUXR1 (S:A2h)

Auxiliary Control Register 1.

7	6	5	4	3	2	1	0
-	-	ENBOOT	-	GF3	0	-	DPS

Bit Number	Bit Mnemonic	Description
7-6	-	Reserved The value read from these bits is indeterminate. Do not set these bits.
5	ENBOOT	Enable Boot Flash Set this bit for map the boot flash between F800h -FFFFh Clear this bit for disable boot flash.
4	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
3	GF3	General Purpose Flag 3.
2	0	Always Zero This bit is stuck to logic 0 to allow INC AUXR1 instruction without affecting GF3 flag.
1	-	Reserved for Data Pointer Extension.
0	DPS	Data Pointer Select Bit Set to select second dual data pointer: DPTR1. Clear to select first dual data pointer: DPTR0.

Reset Value= XXXX 00X0b

Figure 17. AUXR1 Register

9. EEPROM data memory

9.1. General description

The 2k byte on-chip EEPROM memory block is located at addresses 0000h to 07FFh of the ERAM memory space and is selected by setting control bits in the EECON register.

A read in the EEPROM memory is done with a MOVX instruction.

A physical write in the EEPROM memory is done in two steps: write data in the column latches and transfer of all data latches into an EEPROM memory row (programming).

The number of data written on the page may vary from 1 to 128 bytes (the page size). When programming, only the data written in the column latch is programmed and a ninth bit is used to obtain this feature. This provides the capability to program the whole memory by bytes, by page or by a number of bytes in a page. Indeed, each ninth bit is set when the writing the corresponding byte in a row and all these ninth bits are reset after the writing of the complete EEPROM row.

9.2. Write Data in the column latches

Data is written by byte to the column latches as for an ERAM memory. Out of the 11 address bits of the data pointer, the 4 MSBs are used for page selection (row) and 7 are used for byte selection. Between two EEPROM programming sessions, all the addresses in the column latches must stay on the same page, meaning that the 4 MSB must no be changed.

The following procedure is used to write to the column latches:

- Set bit EEE of EECON register
- Stretch the MOVX to accommodate the slow access time of the column latch
- Load DPTR with the address to write
- Store A register with the data to be written
- Execute a MOVX @DPTR, A
- If needed loop the three last instructions until the end of a 128 bytes page

9.3. Programming

The EEPROM programming consists on the following actions:

- writing one or more bytes of one page in the column latches. Normally, all bytes must belong to the same page; if not, the first page address will be latched and the others discarded.
- launching programming by writing the control sequence (54h followed by A4h) to the EECON register.
- EEBUSY flag in EECON is then set by hardware to indicate that programming is in progress and that the EEPROM segment is not available for reading.
- The end of programming is indicated by a hardware clear of the EEBUSY flag.

9.4. Read Data

The following procedure is used to read the data stored in the EEPROM memory:

- Set bit EEE of EECON register
- Stretch the MOVX to accommodate the slow access time of the column latch
- Load DPTR with the address to read
- Execute a MOVX A, @DPTR

9.5. Registers

EECON (S:0D2h)

EEPROM Control Register

7	6	5	4	3	2	1	0
EEPL3	EEPL2	EEPL1	EEPL0	-	-	EEE	EEBUSY

Bit Number	Bit Mnemonic	Description
7-4	EEPL3-0	Programming Launch command bits Write 5Xh followed by AXh to EEPL to launch the programming.
3	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
2	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
1	EEE	Enable EEPROM Space bit Set to map the EEPROM space during MOVX instructions (Write in the column latches) Clear to map the ERAM space during MOVX.
0	EEBUSY	Programming Busy flag Set by hardware when programming is in progress. Cleared by hardware when programming is done. Can not be set or cleared by software.

Reset Value= XXXX XX00b

Not bit addressable

Figure 18. EECON Register

10. In-System-Programming (ISP)

10.1. Introduction

With the implementation of the User ROM and the Boot ROM in Flash technology the T89C51CC02 allows the system engineer the development of applications with a very high level of flexibility. This flexibility is based on the possibility to alter the customer programming on all stages of a product's life:

- During the final production phase, the 1st personalization of the product by parallel or serial charging of the code in the User ROM and if wanted also a customized Boot loader in the Boot memory (Atmel will provide also a standard Boot loader by default).
- After assembling of the product in its final, embedded position by serial mode via the CAN bus.

This In-System-Programming (ISP) allows code modification over the total lifetime of the product.

Besides the default Boot loader Atmel will provide to the customer also all the needed Application-Programming-Interfaces (API) which are needed for the ISP. The API will be located also in the Boot memory.

This will allow the customer to have a full use of the 16 Kbyte user memory.

Two blocks flash memories are implemented (see Figure 19):

- Flash memory FM0:
containing 16 Kbytes of program memory organized in page of 128 bytes,
- Flash memory FM1:
2 Kbytes for default boot loader and Application Programming Interfaces (API).

The FM0 supports both, hardware (parallel) and software programming whereas FM1 supports only hardware programming.

The ISP functions are assumed by:

- FCON register & bit ENBOOT in AUXR1 register,
- Software Boot Vector (SBV), which can be read and modified by using an API or the parallel programming mode (see Figure 22)
The SBV is stored in XROW.
- The Fuse bit Boot Loader Jump Bit (BLJB) can be read and modified using an API or the parallel programming mode.
The BLJB is located in the Hardware security byte (see Figure 24).
- The Extra Byte (EB) and Boot Status Byte (BSB) can be modified only by using API (see Figure 24).
EB is stored in XROW

The bit ENBOOT in AUXR1 register allows to map FM1 between address F800h and FFFFh of FM0.

The FM0 can be programmed by:

- The Atmel boot loader, located by default in FM1.
- The user boot loader located in FM0
- The user boot loader located in FM1 in place of Atmel boot loader.

API contained in FM1 can be called by the user boot loader located in FM0 at the address [SBV]00h.

The user program simply calls the common entry point with appropriate parameters in FM1 to accomplish the desired operation (all these methods will describe in Application Notes on api-description).

Boot Flash operations include: erase block, program byte or page, verify byte or page, program security lock bit, etc. Indeed, Atmel provides the binary code of the default Flash boot loader.

10.2. Flash Programming and Erasure

There are three methods of programming the Flash memory:

- The Atmel bootloader located in FM1 is activated by the application. Low level API routines (located in FM1) to program FM0 will be used. The interface used for serial downloading to FM0 is the UART or the CAN. API can be called also by user's bootloader located in FM0 at [SBV]00h.
- A further method exist in activating the Atmel boot loader by hardware activation.
- The FM0 can be prograded also by the parallel mode using a programmer.

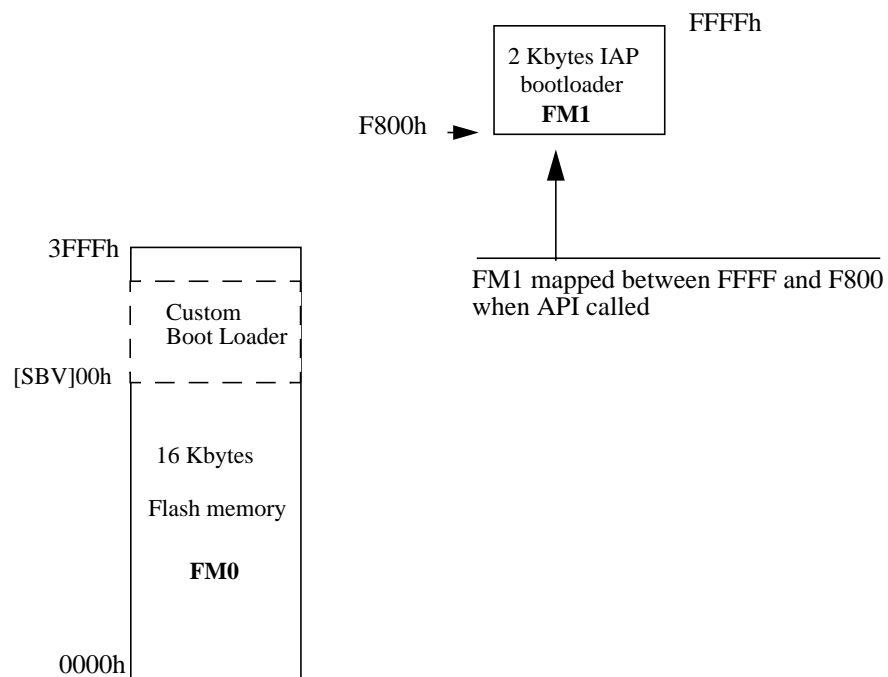


Figure 19. Flash Memory Mapping

10.2.1. Flash Parallel Programming

The three lock bits in Hardware byte are programmed according to Table, will provide different level of protection for the on-chip code and data located in FM0 and FM1.

The only way for write this bits are the parallel mode.

Table 16. Program Lock bit

Program Lock Bits				Protection description
Security level	LB0	LB1	LB2	
1	U	U	U	No program lock features enabled. MOVC instruction executed from external program memory returns non encrypted data.
2	P	U	U	MOVC instruction executed from external program memory are disabled from fetching code bytes from internal memory.
3	U	P	U	Same as 2, also verify through parallel programming interface is disabled.
4	U	U	P	Same as 3, also external execution is disabled.

Program Lock bits

U: unprogrammed

P: programmed

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

Program Lock bits

These security bits protect the code access through the parallel programming interface. They are set by default to level 4.

10.3 Boot Process

10.3.1. Software boot process example

Many algorithms can be used for the software boot process. Before describing them, some explanations are needed for the utility of different flags and bytes available.

Boot Loader Jump Bit (BLJB):

- This bit indicates if on RESET the user wants jump on his application at address @0000h on FM0 or execute the boot loader at address @F800h on FM1.
- BLJB = 0 on parts delivered with bootloader programmed.
- To read or modified this bit, the APIs are used.

Boot Vector Address (SBV):

- This byte contains the msb of the user boot loader address in FM0.
- The default value of SBV is FFh (no user boot loader in FM0).
- To read or modified this byte, the APIs are used.

Extra Byte (EB) & Boot Status Byte (BSB):

- These bytes are reserved for customer use.
- To read or modified this byte, the APIs are used.

Example of software boot process in FM1 (see Figure 21)

In this example the Extra Byte (EB) is a configuration bit which forces the user boot loader execution even on the hardware condition.

10.3.2. Hardware boot process

At the falling edge of RESET, the bit ENBOOT in AUXR1 register is initialized with the value of Boot Loader Jump Bit (BLJB).

FCON register is initialized with the value 00h and the program in FM1 can be executed.

Check of the BLJB value.

- If bit BLJB is cleared (BLJB = 1):
User application in FM0 will be started at @0000h (standard reset).
- If bit BLJB is set (BLJB = 0):
Boot loader will be started at @F800h in FM1.

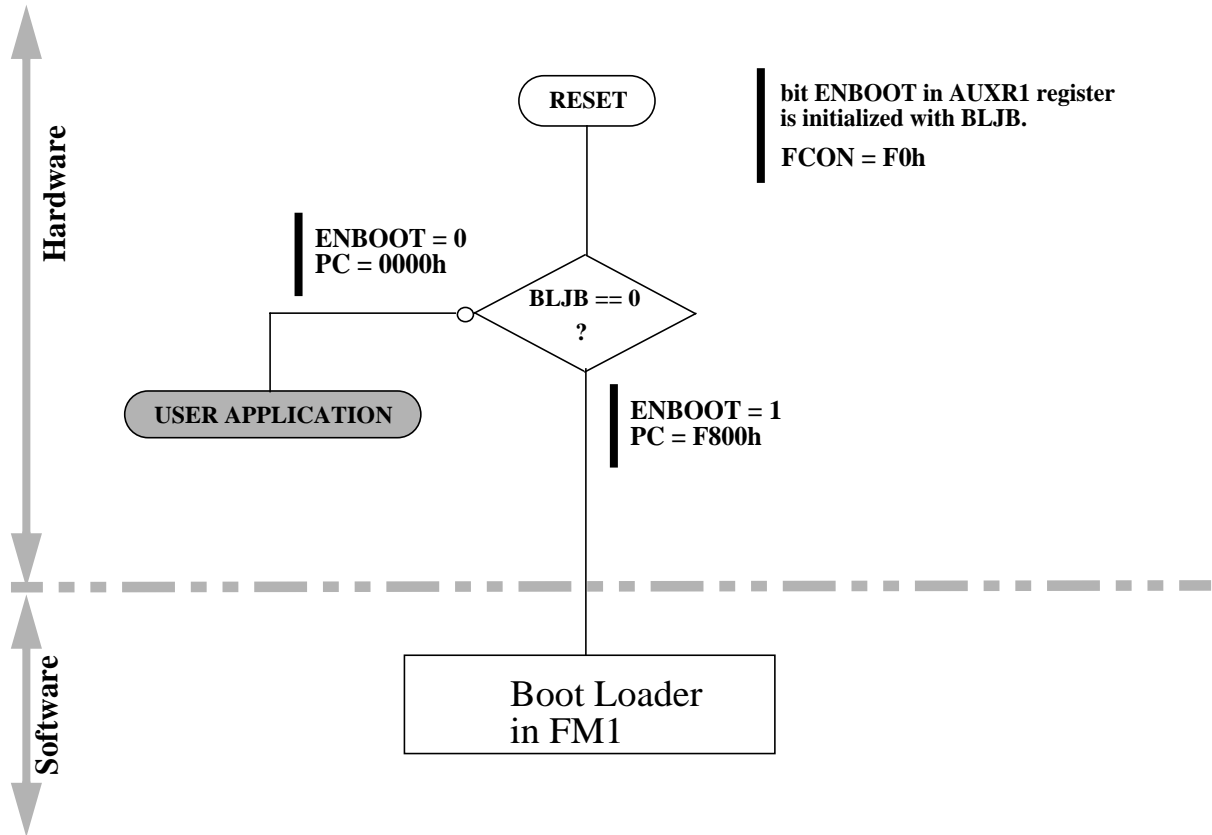


Figure 20. Hardware Boot Process Algorithm

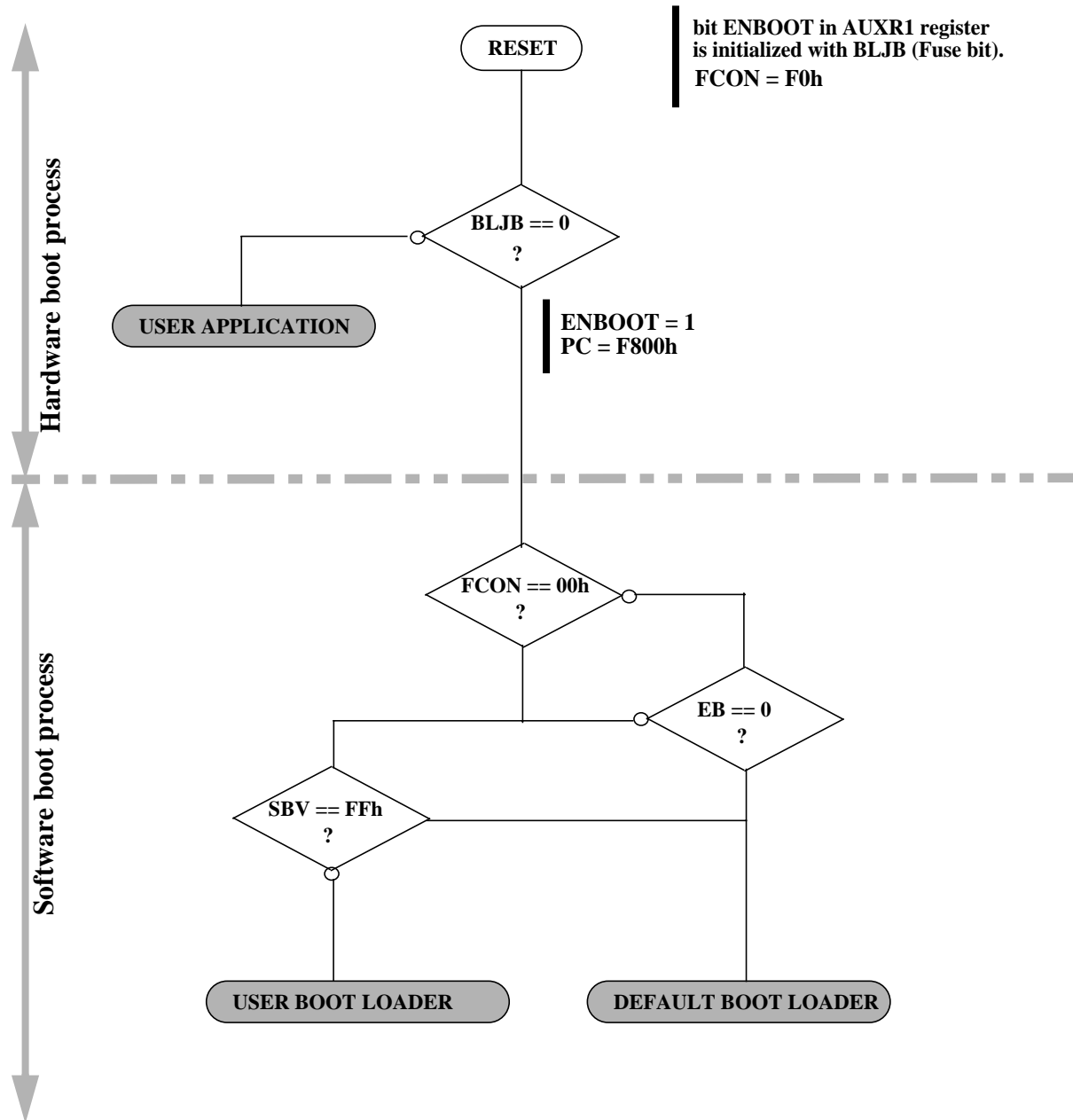


Figure 21. Example of Software Boot process

10.4. 2 Application-Programming-Interface

Several Application Program Interface (API) calls are available for use by an application program to permit selective erasing and programming of FLASH pages. All calls are made by functions.

All these APIs will be described in an application note.

API CALL	Description
PROGRAM DATA BYTE	Write a byte in flash memory
PROGRAM DATA PAGE	Write a page (128 bytes) in flash memory
PROGRAM EEPROM BYTE	Write a byte in Eeprom memory
ERASE BLOCK	Erase all flash memory
ERASE BOOT VECTOR (SBV)	Erase the boot vector
PROGRAM BOOT VECTOR (SBV)	Write the boot vector
PROGRAM EXTRA BYTE (EB)	Write the extra byte
READ DATA BYTE	
READ EEPROM BYTE	
READ FAMILY CODE	
READ MANUFACTURER CODE	
READ PRODUCT NAME	
READ REVISION NUMBER	
READ STATUS BIT (BSB)	Read the status bit
READ BOOT VECTOR (SBV)	Read the boot vector
READ EXTRA BYTE (EB)	Read the extra byte
PROGRAM X2	Write the hardware flag for X2 mode
READ X2	Read the hardware flag for X2 mode
PROGRAM BLJB	Write the hardware flag BLJB
READ BLJB	Read the hardware flag BLJB

10.5. Application remarks

- After loading a new program using by the boot loader, the BLJB bit must be set to allow user application to start at RESET.
- A user bootloader can be mapped at address [SBV]00h. The byte SBV contains the high byte of the boot address, and can be read and written by API.
- The API can be called during user application, without disabling interrupt. The interrupts are disabled by some APIs, for complex operations.

10.6. XROW Bytes

Mnemonic	Description	Default value	Address
SBV	Boot Vector Address	F8h	01h
SSB	Software Security Byte	FFh	05h
EB	Extra Byte	FFh	06h
	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	F7h	60h
	Copy of the Device ID#3:Name and Revision	FFh	61h

Table 17. Xrow mapping

SBV register

Software Boot Vector

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

Bit Number	Bit Mnemonic	Description
7-0	ADD7:0	MSB of user boot loader address location

Default value after erasing chip: FFh

NOTE:

Only accessed by the API or in the parallel programming mode.

Figure 22. SBV Register

EB register

EXTRA BYTE

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

Bit Number	Bit Mnemonic	Description
7-0	-	User definition

Default value after erasing chip: FFh

NOTE:

Only accessed by the API or in the parallel programming mode.

Figure 23. EB Register

10.7. Hardware Byte

7	6	5	4	3	2	1	0
X2B	BLJB	-	-	-	LB2	LB1	LB0

Bit Number	Bit Mnemonic	Description
7	X2B	X2 Bit Set this bit to start in standard mode Clear this bit to start in X2 mode.
6	BLJB	Boot Loader Jump Bitt Clear (=1)this bit to start the user's application on next RESET (@0000h) located in FM0, Set (=0)this bit to start the boot loader(@F800h) located in FM1.
5-3	-	Reserved The value read from these bits are indeterminate.
2-0	LB2:0	Lock Bits

Default value after erasing chip: FFh

NOTE:

Only the 4 MSB bits can be access by software.

The 4 LSB bits can only be access by parallel mode.

Figure 24. Hardware byte

11. Serial I/O Port

The T89C51CC02 I/O serial port is compatible with the I/O serial port in the 80C52.

It provides both synchronous and asynchronous communication modes. It operates as a Universal Asynchronous Receiver and Transmitter (UART) in three full-duplex modes (Modes 1, 2 and 3). Asynchronous transmission and reception can occur simultaneously and at different baud rates

Serial I/O port includes the following enhancements:

- Framing error detection
- Automatic address recognition

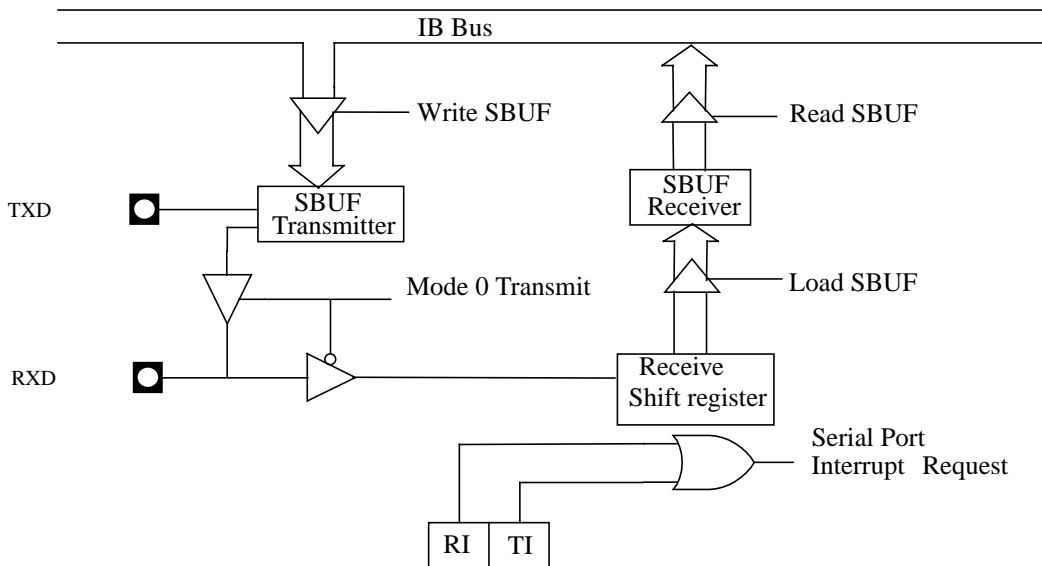


Figure 25. Serial I/O Port Block Diagram

11.1. Framing Error Detection

Framing bit error detection is provided for the three asynchronous modes. To enable the framing bit error detection feature, set SMOD0 bit in PCON register.

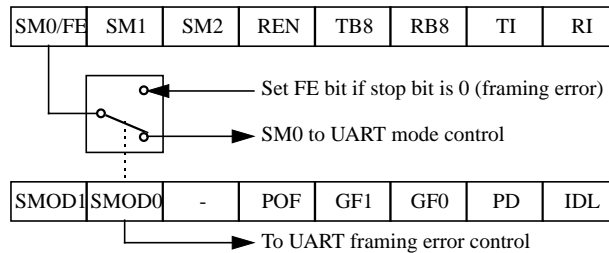


Figure 26. Framing Error Block Diagram

When this feature is enabled, the receiver checks each incoming data frame for a valid stop bit. An invalid stop bit may result from noise on the serial lines or from simultaneous transmission by two CPUs. If a valid stop bit is not found, the Framing Error bit (FE) in SCON register bit is set.

The software may examine the FE bit after each reception to check for data errors. Once set, only software or a reset clears the FE bit. Subsequently received frames with valid stop bits cannot clear the FE bit. When the FE feature is enabled, RI rises on the stop bit instead of the last data bit (See Figure 27. and Figure 28.).

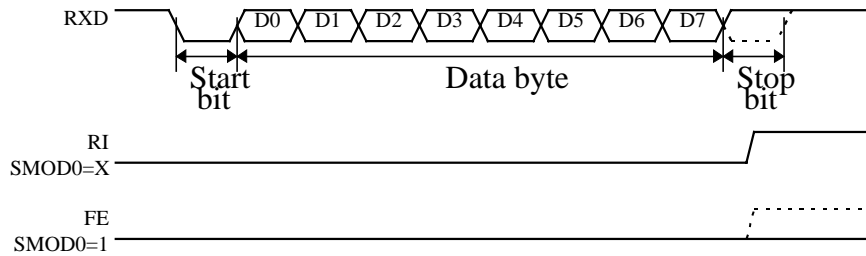


Figure 27. UART Timing in Mode 1

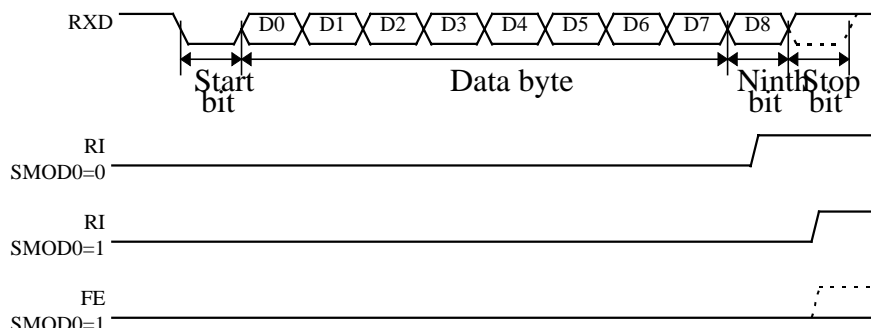


Figure 28. UART Timing in Modes 2 and 3

11.2. Automatic Address Recognition

The automatic address recognition feature is enabled when the multiprocessor communication feature is enabled (SM2 bit in SCON register is set).

Implemented in the hardware, automatic address recognition enhances the multiprocessor communication feature by allowing the serial port to examine the address of each incoming command frame. Only when the serial port recognizes its own address will the receiver set the RI bit in the SCON register to generate an interrupt. This ensures that the CPU is not interrupted by command frames addressed to other devices.

If necessary, you can enable the automatic address recognition feature in mode 1. In this configuration, the stop bit takes the place of the ninth data bit. Bit RI is set only when the received command frame address matches the device's address and is terminated by a valid stop bit.

To support automatic address recognition, a device is identified by a given address and a broadcast address.

NOTE: The multiprocessor communication and automatic address recognition features cannot be enabled in mode 0 (i.e. setting SM2 bit in SCON register in mode 0 has no effect).

11.3. 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:

SADDR	0101 0110b
SADEN	<u>1111 1100b</u>
Given	0101 01XXb

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

Slave A:	SADDR	1111 0001b
	SADEN	<u>1111 1010b</u>
	Given	1111 0X0Xb

Slave B:	SADDR	1111 0011b
	SADEN	<u>1111 1001b</u>
	Given	1111 0XX1b

Slave C:	SADDR	1111 0010b
	SADEN	<u>1111 1101b</u>
	Given	1111 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).

11.4. 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.:

SADDR	0101 0110b
SADEN	1111 1100b
SADDR OR SADEN	1111 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:	SADDR	1111 0001b
	SADEN	<u>1111 1010b</u>
	Given	1111 1X11b,

Slave B:	SADDR	1111 0011b
	SADEN	<u>1111 1001b</u>
	Given	1111 1X11B,

Slave C:	SADDR=	1111 0010b
	SADEN	<u>1111 1101b</u>
	Given	1111 1111b

For slaves A and B, bit 2 is a don't care bit; for slave C, bit 2 is set. To communicate with all of the slaves, the master must send an address FFh. To communicate with slaves A and B, but not slave C, the master can send and address FBh.

11.5. REGISTERS

SCON (S:98h)

Serial Control Register

7	6	5	4	3	2	1	0
FE/SM0	SM1	SM2	REN	TB8	RB8	TI	RI

Bit Number	Bit Mnemonic	Description															
7	FE	Framing Error bit (SMOD0=1) Clear to reset the error state, not cleared by a valid stop bit. Set by hardware when an invalid stop bit is detected.															
	SM0	Serial port Mode bit 0 (SMOD0=0) Refer to SM1 for serial port mode selection.															
6	SM1	Serial port Mode bit 1 <table style="margin-left: 20px; border-collapse: collapse;"> <tr> <td style="text-align: center;"><u>SM0</u></td> <td style="text-align: center;"><u>SM1</u></td> <td style="text-align: center;"><u>ModeBaud Rate</u></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Shift Register$F_{XTAL}/12$</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>8-bit UARTVariable</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>9-bit UART$F_{XTAL}/64$ or $F_{XTAL}/32$</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>9-bit UARTVariable</td> </tr> </table>	<u>SM0</u>	<u>SM1</u>	<u>ModeBaud Rate</u>	0	0	Shift Register $F_{XTAL}/12$	0	1	8-bit UARTVariable	1	0	9-bit UART $F_{XTAL}/64$ or $F_{XTAL}/32$	1	1	9-bit UARTVariable
<u>SM0</u>	<u>SM1</u>	<u>ModeBaud Rate</u>															
0	0	Shift Register $F_{XTAL}/12$															
0	1	8-bit UARTVariable															
1	0	9-bit UART $F_{XTAL}/64$ or $F_{XTAL}/32$															
1	1	9-bit UARTVariable															
5	SM2	Serial port Mode 2 bit / Multiprocessor Communication Enable bit Clear to disable multiprocessor communication feature. Set to enable multiprocessor communication feature in mode 2 and 3.															
4	REN	Reception Enable bit Clear to disable serial reception. Set to enable serial reception.															
3	TB8	Transmitter Bit 8 / Ninth bit to transmit in modes 2 and 3 Clear to transmit a logic 0 in the 9th bit. Set to transmit a logic 1 in the 9th bit.															
2	RB8	Receiver Bit 8 / Ninth bit received in modes 2 and 3 Cleared by hardware if 9th bit received is a logic 0. Set by hardware if 9th bit received is a logic 1.															
1	TI	Transmit Interrupt flag Clear to acknowledge interrupt. Set by hardware at the end of the 8th bit time in mode 0 or at the beginning of the stop bit in the other modes.															
0	RI	Receive Interrupt flag Clear to acknowledge interrupt. Set by hardware at the end of the 8th bit time in mode 0, see Figure 27. and Figure 28. in the other modes.															

Reset Value = 0000 0000b

Bit addressable

Figure 29. SCON Register

SADEN (S:B9h)

Slave Address Mask Register

7	6	5	4	3	2	1	0

Bit Number	Bit Mnemonic	Description
7-0		Mask Data for Slave Individual Address

Reset Value = 0000 0000b

Not bit addressable

Figure 30. SADEN Register

SADDR (S:A9h)

Slave Address Register

7	6	5	4	3	2	1	0

Bit Number	Bit Mnemonic	Description
7-0		Slave Individual Address

Reset Value = 0000 0000b

Not bit addressable

Figure 31. SADDR Register

SBUF (S:99h)

Serial Data Buffer

7	6	5	4	3	2	1	0

Bit Number	Bit Mnemonic	Description
7-0		Data sent/received by Serial I/O Port

Reset Value = 0000 0000b

Not bit addressable

Figure 32. SBUF Register

PCON (S:87h)
Power Control Register

7	6	5	4	3	2	1	0
SMOD1	SMOD0	-	POF	GF1	GF0	PD	IDL

Bit Number	Bit Mnemonic	Description
7	SMOD1	Serial port Mode bit 1 Set to select double baud rate in mode 1, 2 or 3.
6	SMOD0	Serial port Mode bit 0 Clear to select SM0 bit in SCON register. Set to select FE bit in SCON register.
5	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
4	POF	Power-Off Flag Clear to recognize next reset type. Set by hardware when VCC rises from 0 to its nominal voltage. Can also be set by software.
3	GF1	General purpose Flag Cleared by user for general purpose usage. Set by user for general purpose usage.
2	GF0	General purpose Flag Cleared by user for general purpose usage. Set by user for general purpose usage.
1	PD	Power-Down mode bit Cleared by hardware when reset occurs. Set to enter power-down mode.
0	IDL	Idle mode bit Clear by hardware when interrupt or reset occurs. Set to enter idle mode.

Reset Value = 00X1 0000b

Not bit addressable

Figure 33. PCON Register

12. Timers/Counters

12.1. Introduction

The T89C51CC02 implements two general-purpose, 16-bit Timers/Counters. They are identified as Timer 0 and Timer 1, and can be independently configured to operate in a variety of modes as a Timer or as an event Counter. When operating as a Timer, the Timer/Counter runs for a programmed length of time, then issues an interrupt request. When operating as a Counter, the Timer/Counter counts negative transitions on an external pin. After a preset number of counts, the Counter issues an interrupt request.

The various operating modes of each Timer/Counter are described in the following sections.

12.2. Timer/Counter Operations

For instance, a basic operation is Timer registers TH_x and TL_x (x= 0, 1) connected in cascade to form a 16-bit Timer. Setting the run control bit (TR_x) in TCON register (see Figure 39) turns the Timer on by allowing the selected input to increment TL_x. When TL_x overflows it increments TH_x; when TH_x overflows it sets the Timer overflow flag (TF_x) in TCON register. Setting the TR_x does not clear the TH_x and TL_x Timer registers. Timer registers can be accessed to obtain the current count or to enter preset values. They can be read at any time but TR_x bit must be cleared to preset their values, otherwise the behavior of the Timer/Counter is unpredictable.

The C/Tx# control bit selects Timer operation or Counter operation by selecting the divided-down peripheral clock or external pin Tx as the source for the counted signal. TR_x bit must be cleared when changing the mode of operation, otherwise the behavior of the Timer/Counter is unpredictable.

For Timer operation (C/Tx# = 0), the Timer register counts the divided-down peripheral clock. The Timer register is incremented once every peripheral cycle (6 peripheral clock periods). The Timer clock rate is $F_{PER} / 6$, i.e. $F_{OSC} / 12$ in standard mode or $F_{OSC} / 6$ in X2 mode.

For Counter operation (C/Tx# = 1), the Timer register counts the negative transitions on the Tx external input pin. The external input is sampled every peripheral cycles. When the sample is high in one cycle and low in the next one, the Counter is incremented. Since it takes 2 cycles (12 peripheral clock periods) to recognize a negative transition, the maximum count rate is $F_{PER} / 12$, i.e. $F_{OSC} / 24$ in standard mode or $F_{OSC} / 12$ in X2 mode. There are no restrictions on the duty cycle of the external input signal, but to ensure that a given level is sampled at least once before it changes, it should be held for at least one full peripheral cycle.

12.3. Timer 0

Timer 0 functions as either a Timer or event Counter in four modes of operation. Figure 34 to Figure 37 show the logical configuration of each mode.

Timer 0 is controlled by the four lower bits of TMOD register (see Figure 40) and bits 0, 1, 4 and 5 of TCON register (see Figure 39). TMOD register selects the method of Timer gating (GATE0), Timer or Counter operation (T/C0#) and mode of operation (M10 and M00). TCON register provides Timer 0 control functions: overflow flag (TF0), run control bit (TR0), interrupt flag (IE0) and interrupt type control bit (IT0).

For normal Timer operation (GATE0 = 0), setting TR0 allows TL0 to be incremented by the selected input. Setting GATE0 and TR0 allows external pin INT0# to control Timer operation.

Timer 0 overflow (count rolls over from all 1s to all 0s) sets TF0 flag generating an interrupt request.

It is important to stop Timer/Counter before changing mode.

12.3.1. Mode 0 (13-bit Timer)

Mode 0 configures Timer 0 as an 13-bit Timer which is set up as an 8-bit Timer (TH0 register) with a modulo 32 prescaler implemented with the lower five bits of TL0 register (see Figure 34). The upper three bits of TL0 register are indeterminate and should be ignored. Prescaler overflow increments TH0 register.

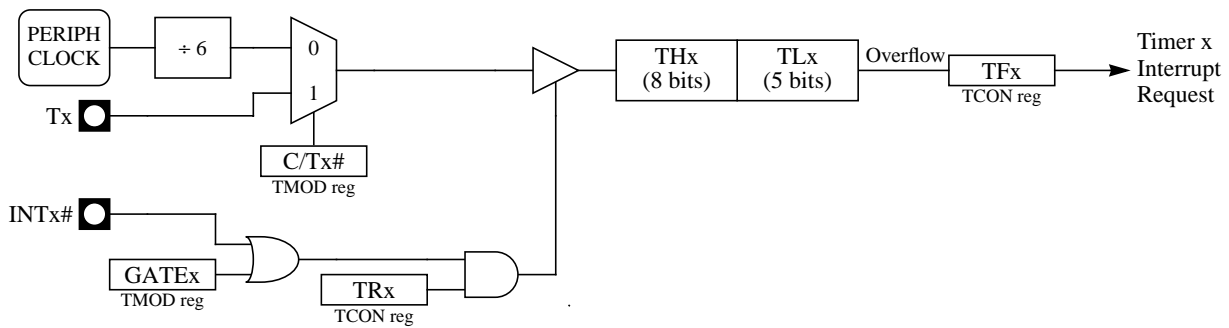


Figure 34. Timer/Counter x (x= 0 or 1) in Mode 0

12.3.2. Mode 1 (16-bit Timer)

Mode 1 configures Timer 0 as a 16-bit Timer with TH0 and TL0 registers connected in cascade (see Figure 35). The selected input increments TL0 register.

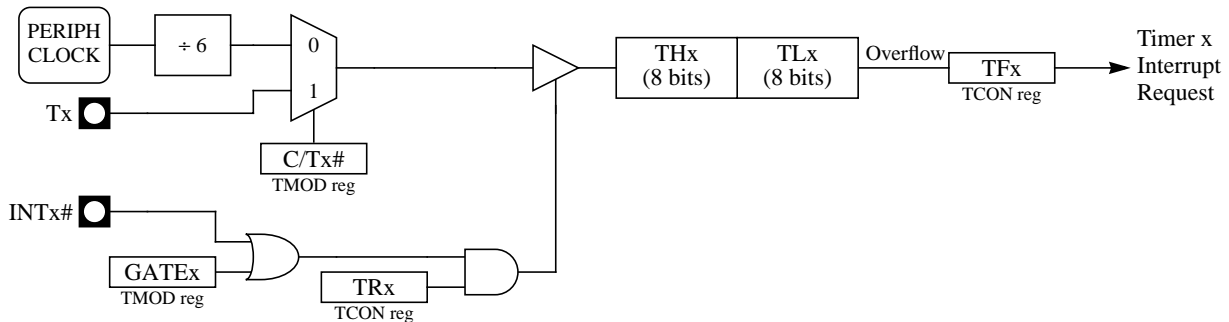


Figure 35. Timer/Counter x (x= 0 or 1) in Mode 1

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

Mode 2 configures Timer 0 as an 8-bit Timer (TL0 register) that automatically reloads from TH0 register (see Figure 36). TL0 overflow sets TF0 flag in TCON register and reloads TL0 with the contents of TH0, which is preset by software. When the interrupt request is serviced, hardware clears TF0. The reload leaves TH0 unchanged. The next reload value may be changed at any time by writing to TH0 register.

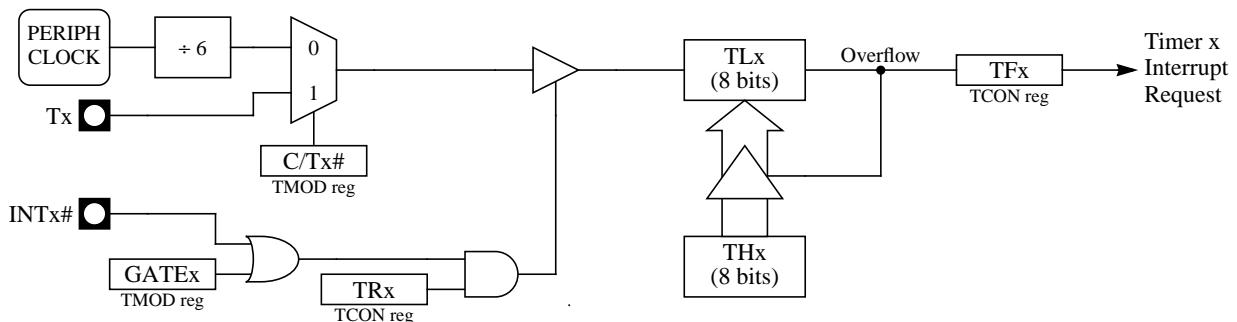


Figure 36. Timer/Counter x (x= 0 or 1) in Mode 2

12.3.4. 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 37). 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.

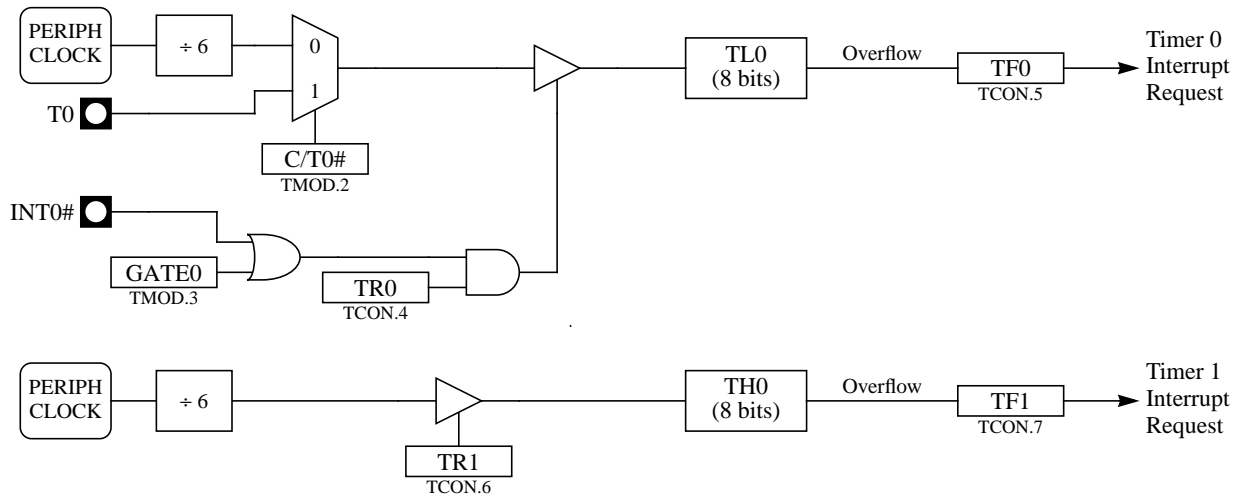


Figure 37. Timer/Counter 0 in Mode 3: Two 8-bit Counters

12.4. 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 34 to Figure 36 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 40) and bits 2, 3, 6 and 7 of TCON register (see Figure 39). 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.

12.4.1. 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 34). The upper 3 bits of TL1 register are ignored. Prescaler overflow increments TH1 register.

12.4.2. 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 35). The selected input increments TL1 register.

12.4.3. 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 36). 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.

12.4.4. 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.

12.5. 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.

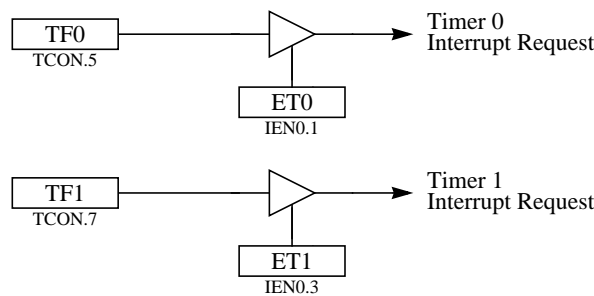


Figure 38. Timer Interrupt System

12.6. Registers

TCON (S:88h)

Timer/Counter Control Register.

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Bit Number	Bit Mnemonic	Description
7	TF1	Timer 1 Overflow Flag Cleared by hardware when processor vectors to interrupt routine. Set by hardware on Timer/Counter overflow, when Timer 1 register overflows.
6	TR1	Timer 1 Run Control Bit Clear to turn off Timer/Counter 1. Set to turn on Timer/Counter 1.
5	TF0	Timer 0 Overflow Flag Cleared by hardware when processor vectors to interrupt routine. Set by hardware on Timer/Counter overflow, when Timer 0 register overflows.
4	TR0	Timer 0 Run Control Bit Clear to turn off Timer/Counter 0. Set to turn on Timer/Counter 0.
3	IE1	Interrupt 1 Edge Flag Cleared by hardware when interrupt is processed if edge-triggered (see IT1). Set by hardware when external interrupt is detected on INT1# pin.
2	IT1	Interrupt 1 Type Control Bit Clear to select low level active (level triggered) for external interrupt 1 (INT1#). Set to select falling edge active (edge triggered) for external interrupt 1.
1	IE0	Interrupt 0 Edge Flag Cleared by hardware when interrupt is processed if edge-triggered (see IT0). Set by hardware when external interrupt is detected on INTO# pin.
0	IT0	Interrupt 0 Type Control Bit Clear to select low level active (level triggered) for external interrupt 0 (INT0#). Set to select falling edge active (edge triggered) for external interrupt 0.

Reset Value= 0000 0000b

Figure 39. TCON Register

T89C51CC02



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 <table border="1"> <thead> <tr> <th>M11</th> <th>M01</th> <th>Operating mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Mode 0: 8-bit Timer/Counter (TH1) with 5-bit prescaler (TL1).</td> </tr> <tr> <td>0</td> <td>1</td> <td>Mode 1: 16-bit Timer/Counter.</td> </tr> <tr> <td>1</td> <td>0</td> <td>Mode 2: 8-bit auto-reload Timer/Counter (TL1). Reloaded from TH1 at overflow.</td> </tr> <tr> <td>1</td> <td>1</td> <td>Mode 3: Timer 1 halted. Retains count.</td> </tr> </tbody> </table>	M11	M01	Operating mode	0	0	Mode 0: 8-bit Timer/Counter (TH1) with 5-bit prescaler (TL1).	0	1	Mode 1: 16-bit Timer/Counter.	1	0	Mode 2: 8-bit auto-reload Timer/Counter (TL1). Reloaded from TH1 at overflow.	1	1	Mode 3: Timer 1 halted. Retains count.
M11	M01		Operating mode														
0	0	Mode 0: 8-bit Timer/Counter (TH1) with 5-bit prescaler (TL1).															
0	1	Mode 1: 16-bit Timer/Counter.															
1	0	Mode 2: 8-bit auto-reload Timer/Counter (TL1). Reloaded from TH1 at overflow.															
1	1	Mode 3: Timer 1 halted. Retains count.															
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 INTO# 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 <table border="1"> <thead> <tr> <th>M10</th> <th>M00</th> <th>Operating mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Mode 0: 8-bit Timer/Counter (TH0) with 5-bit prescaler (TL0).</td> </tr> <tr> <td>0</td> <td>1</td> <td>Mode 1: 16-bit Timer/Counter.</td> </tr> <tr> <td>1</td> <td>0</td> <td>Mode 2: 8-bit auto-reload Timer/Counter (TL0). Reloaded from TH0 at overflow.</td> </tr> <tr> <td>1</td> <td>1</td> <td>Mode 3: TL0 is an 8-bit Timer/Counter. TH0 is an 8-bit Timer using Timer 1's TR0 and TF0 bits.</td> </tr> </tbody> </table>	M10	M00	Operating mode	0	0	Mode 0: 8-bit Timer/Counter (TH0) with 5-bit prescaler (TL0).	0	1	Mode 1: 16-bit Timer/Counter.	1	0	Mode 2: 8-bit auto-reload Timer/Counter (TL0). Reloaded from TH0 at overflow.	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.
M10	M00		Operating mode														
0	0	Mode 0: 8-bit Timer/Counter (TH0) with 5-bit prescaler (TL0).															
0	1	Mode 1: 16-bit Timer/Counter.															
1	0	Mode 2: 8-bit auto-reload Timer/Counter (TL0). Reloaded from TH0 at overflow.															
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

Figure 40. TMOD 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

Figure 41. TH0 Register

TL0 (S:8Ah)

Timer 0 Low Byte Register.

7	6	5	4	3	2	1	0
Bit Number	Bit Mnemonic	Description					
7:0		Low Byte of Timer 0.					

Reset Value= 0000 0000b

Figure 42. TL0 Register

TH1 (S:8Dh)

Timer 1 High Byte Register.

7	6	5	4	3	2	1	0
Bit Number	Bit Mnemonic	Description					
7:0		High Byte of Timer 1.					

Reset Value= 0000 0000b

Figure 43. TH1 Register

TL1 (S:8Bh)

Timer 1 Low Byte Register.

7	6	5	4	3	2	1	0
Bit Number	Bit Mnemonic	Description					
7:0		Low Byte of Timer 1.					

Reset Value= 0000 0000b

Figure 44. TL1 Register

13. Timer 2

13.1. Introduction

The T89C51CC02 timer 2 is compatible with timer 2 in the 80C52.

It is a 16-bit timer/counter: the count is maintained by two eight-bit timer registers, TH2 and TL2 that are cascade-connected. It is controlled by T2CON register (See Table 47) and T2MOD register (See Table 48). Timer 2 operation is similar to Timer 0 and Timer 1. $C/\overline{T2}$ selects $F_{OSC}/6$ (timer operation) or external pin T2 (counter operation) as timer register input. Setting TR2 allows TL2 to be incremented by the selected input.

Timer 2 includes the following enhancements:

- Auto-reload mode (up or down counter)
- Programmable clock-output

13.2. Auto-Reload Mode

The auto-reload mode configures timer 2 as a 16-bit timer or event counter with automatic reload. This feature is controlled by the DCEN bit in T2MOD register (See Table 48). Setting the DCEN bit enables timer 2 to count up or down as shown in Figure 45. In this mode the T2EX pin controls the counting direction.

When T2EX is high, timer 2 up-counts. Timer overflow occurs at FFFFh which sets the TF2 flag and generates an interrupt request. The overflow also causes the 16-bit value in RCAP2H and RCAP2L registers to be loaded into the timer registers TH2 and TL2.

When T2EX is low, timer 2 down-counts. Timer underflow occurs when the count in the timer registers TH2 and TL2 equals the value stored in RCAP2H and RCAP2L registers. The underflow sets TF2 flag and reloads FFFFh into the timer registers.

The EXF2 bit toggles when timer 2 overflow or underflow, depending on the direction of the count. EXF2 does not generate an interrupt. This bit can be used to provide 17-bit resolution.

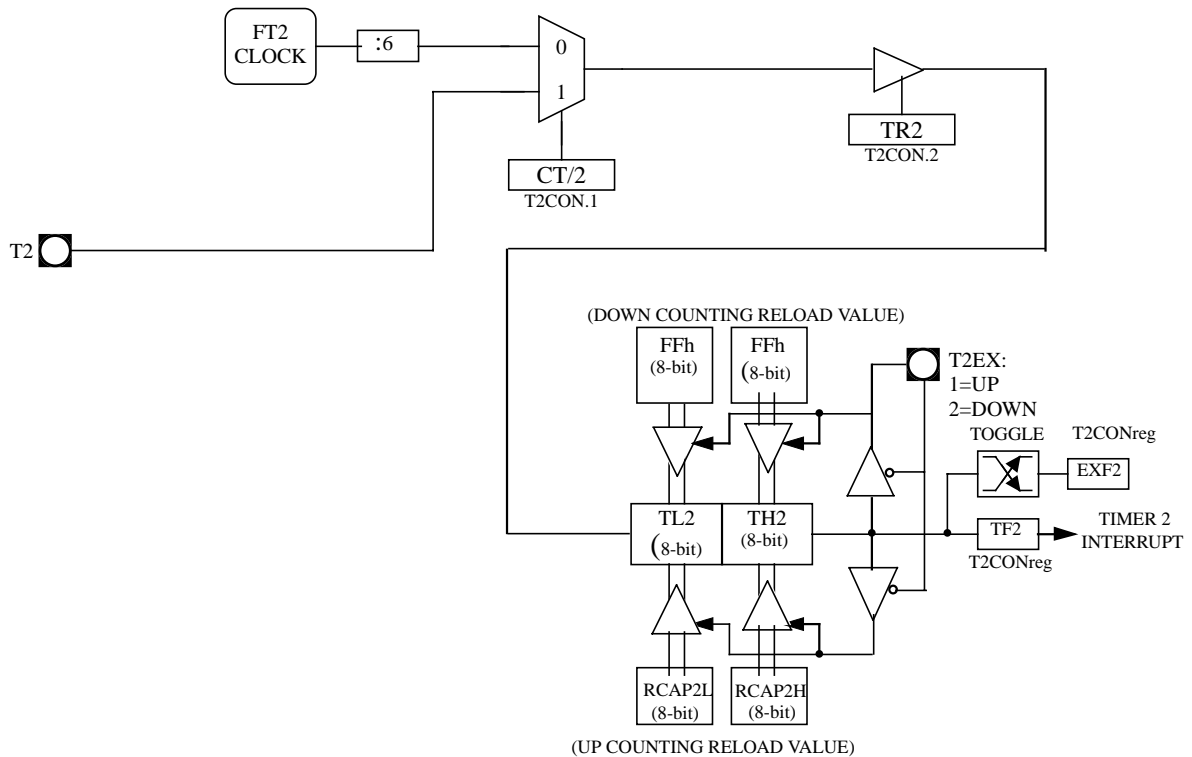


Figure 45. Auto-Reload Mode Up/Down Counter

13.3. Programmable Clock-Output

In clock-out mode, timer 2 operates as a 50%-duty-cycle, programmable clock generator (See Figure 46). The input clock increments TL2 at frequency $F_{OSC}/2$. The timer repeatedly counts to overflow from a loaded value. At overflow, the contents of RCAP2H and RCAP2L registers are loaded into TH2 and TL2. In this mode, timer 2 overflows do not generate interrupts. The formula gives the clock-out frequency depending on the system oscillator frequency and the value in the RCAP2H and RCAP2L registers:

$$Clock - OutFrequency = \frac{F_{osc} \times 2^{x2}}{4 \times (65536 - RCAP2H/RCAP2L)}$$

*NOTE: X2 bit is located in CKCON register.
In X2 mode, $F_{OSC}=F_{XTAL}$. In standard mode, $F_{OSC}=F_{XTAL}/2$.*

For a 16 MHz system clock, timer 2 has a programmable frequency range of 61 Hz ($F_{OSC}/2^{16}$) to 4 MHz ($F_{OSC}/4$). The generated clock signal is brought out to T2 pin (P1.0).

Timer 2 is programmed for the clock-out mode as follows:

- Set T2OE bit in T2MOD register.
- Clear $C/\overline{T2}$ bit in T2CON register.
- Determine the 16-bit reload value from the formula and enter it in RCAP2H/RCAP2L registers.

- Enter a 16-bit initial value in timer registers TH2/TL2. It can be the same as the reload value or different depending on the application.
- To start the timer, set TR2 run control bit in T2CON register.

It is possible to use timer 2 as a baud rate generator and a clock generator simultaneously. For this configuration, the baud rates and clock frequencies are not independent since both functions use the values in the RCAP2H and RCAP2L registers.

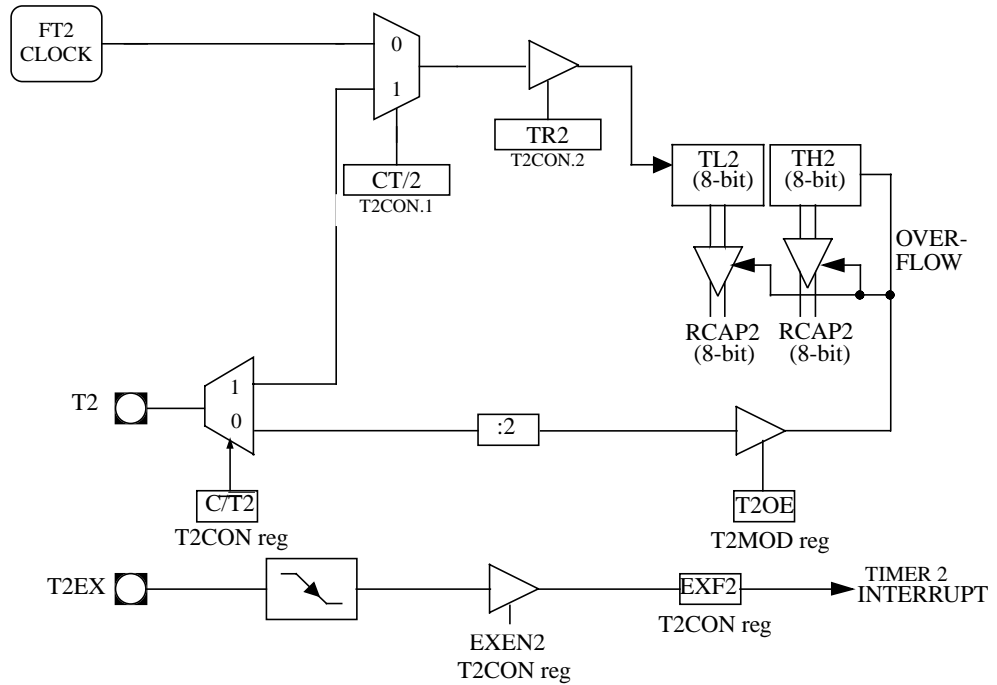


Figure 46. Clock-Out Mode

13.4. Registers

T2CON (S:C8h)

Timer 2 Control Register

7	6	5	4	3	2	1	0
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2#	CP/RL2#

Bit Number	Bit Mnemonic	Description
7	TF2	Timer 2 overflow Flag TF2 is not set if RCLK=1 or TCLK = 1. Must be cleared by software. Set by hardware on timer 2 overflow.
6	EXF2	Timer 2 External Flag Set when a capture or a reload is caused by a negative transition on T2EX pin if EXEN2=1. Set to cause the CPU to vector to timer 2 interrupt routine when timer 2 interrupt is enabled. Must be cleared by software.
5	RCLK	Receive Clock bit Clear to use timer 1 overflow as receive clock for serial port in mode 1 or 3. Set to use timer 2 overflow as receive clock for serial port in mode 1 or 3.
4	TCLK	Transmit Clock bit Clear to use timer 1 overflow as transmit clock for serial port in mode 1 or 3. Set to use timer 2 overflow as transmit clock for serial port in mode 1 or 3.
3	EXEN2	Timer 2 External Enable bit Clear to ignore events on T2EX pin for timer 2 operation. Set to cause a capture or reload when a negative transition on T2EX pin is detected, if timer 2 is not used to clock the serial port.
2	TR2	Timer 2 Run control bit Clear to turn off timer 2. Set to turn on timer 2.
1	C/T2#	Timer/Counter 2 select bit Clear for timer operation (input from internal clock system: F _{OSC}). Set for counter operation (input from T2 input pin).
0	CP/RL2#	Timer 2 Capture/Reload bit If RCLK=1 or TCLK=1, CP/RL2# is ignored and timer is forced to auto-reload on timer 2 overflow. Clear to auto-reload on timer 2 overflows or negative transitions on T2EX pin if EXEN2=1. Set to capture on negative transitions on T2EX pin if EXEN2=1.

Reset Value = 0000 0000b

Bit addressable

Figure 47. T2CON Register

T2MOD (S:C9h)

Timer 2 Mode Control Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	T2OE	DCEN

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	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
1	T2OE	Timer 2 Output Enable bit Clear to program P1.0/T2 as clock input or I/O port. Set to program P1.0/T2 as clock output.
0	DCEN	Down Counter Enable bit Clear to disable timer 2 as up/down counter. Set to enable timer 2 as up/down counter.

Reset Value = XXXX XX00b

Not bit addressable

Figure 48. T2MOD Register

TH2 (S:CDh)

Timer 2 High Byte Register

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

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

Reset Value = 0000 0000b

Not bit addressable

Figure 49. TH2 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

Figure 50. TL2 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

Figure 51. RCAP2H Register

RCAP2L (S:CAh)

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

Figure 52. RCAP2L Register

14. WatchDog Timer

14.1. Introduction

T89C51CC02 contains a powerful programmable hardware WatchDog Timer (WDT) that automatically resets the chip if it software fails to reset the WDT before the selected time interval has elapsed. It permits large Time-Out ranking from 16ms to 2s @Fosc = 12MHz.

This WDT consist of a 14-bit counter plus a 7-bit programmable counter, a WatchDog Timer reset register (WDTRST) and a WatchDog Timer programming (WDTPRG) register. When exiting reset, the WDT is -by default- disable. To enable the WDT, the user has to write the sequence 1EH and E1H into WDRST register. When the WatchDog Timer is enabled, it will increment every machine cycle while the oscillator is running and there is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will generate an output RESET pulse at the RST pin. The RESET pulse duration is $96 \times T_{OSC}$, where $T_{OSC} = 1 / F_{OSC}$. To make the best use of the WDT, it should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

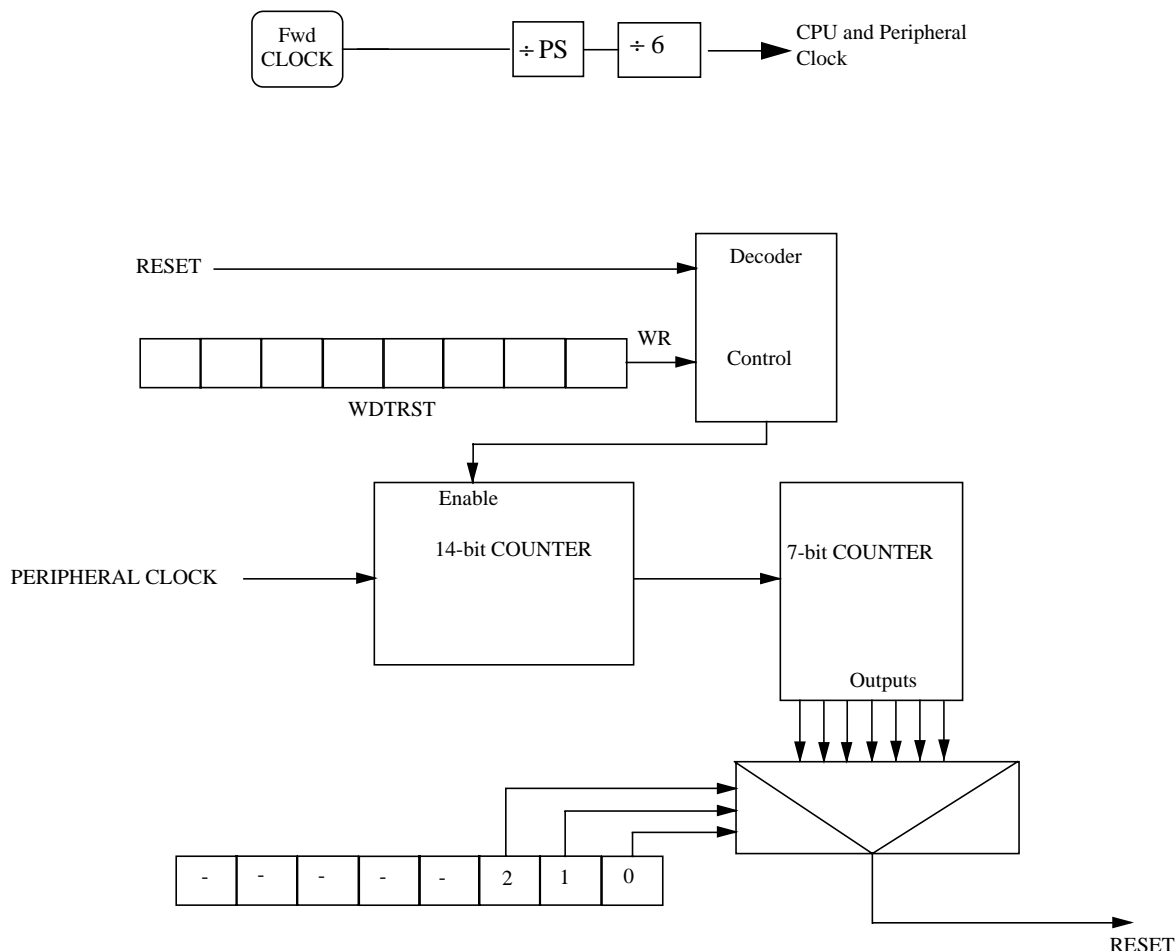


Figure 53. WatchDog Timer

14.2. WatchDog Programming

The three lower bits (S0, S1, S2) located into WDTPRG register permits to program the WDT duration.

Table 18. Machine Cycle Count

S2	S1	S0	Machine Cycle Count
0	0	0	$2^{14} - 1$
0	0	1	$2^{15} - 1$
0	1	0	$2^{16} - 1$
0	1	1	$2^{17} - 1$
1	0	0	$2^{18} - 1$
1	0	1	$2^{19} - 1$
1	1	0	$2^{20} - 1$
1	1	1	$2^{21} - 1$

To compute WD Time-Out, the following formula is applied:

$$FTime - Out = \frac{F_{XTAL}}{12 \times ((2^{14} \times 2^{Svalue}) - 1)}$$

Note: Svalue represents the decimal value of (S2 S1 S0)

Find Hereafter computed Time-Out value for $Fosc_{XTAL} = 12\text{MHz}$

Table 19. Time-Out Computation

S2	S1	S0	Fosc=12MHz	Fosc=16MHz	Fosc=20MHz
0	0	0	16.38 ms	12.28 ms	9.82 ms
0	0	1	32.77 ms	24.57 ms	19.66 ms
0	1	0	65.54 ms	49.14 ms	39.32 ms
0	1	1	131.07 ms	98.28 ms	78.64 ms
1	0	0	262.14 ms	196.56 ms	157.28 ms
1	0	1	524.29 ms	393.12 ms	314.56 ms
1	1	0	1.05 s	786.24 ms	629.12 ms
1	1	1	2.10 s	1.57 s	1.25 ms

14.3. 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, servicing the WDT should occur as it normally does whenever T89C51CC02 is reset. Exiting Power Down with an interrupt is significantly different. The interrupt is 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 of 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 T89C51CC02 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.

14.4. 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

Figure 54. WDTPRG Register

WDTRST (S:A6h Write only)

WatchDog Timer Enable register

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

Bit Number	Bit Mnemonic	Description
7	-	Watchdog Control Value

Reset Value = 1111 1111b

NOTE:

The WDRST register is used to reset/enable the WDT by writing 1EH then E1H in sequence.

Figure 55. WDTRST Register

15. Atmel CAN Controller

15.1. Introduction

The Atmel CAN Controller provides all the features required to implement the serial communication protocol CAN as defined by the BOSCH GmbH. The CAN specifications as referred to in ISO/11898 (2.0A & 2.0B) for high speed and ISO/11519-2 for low speed are applied. The CAN Controller is able to handle all types of frames (Data, Remote, Error and Overload) and achieves a bitrate of 1 Mbit/s at 8MHz¹ Crystal frequency in X2 mode.

NOTE:

1. At BRP = 1 sampling point will be fixed.

15.2. 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.

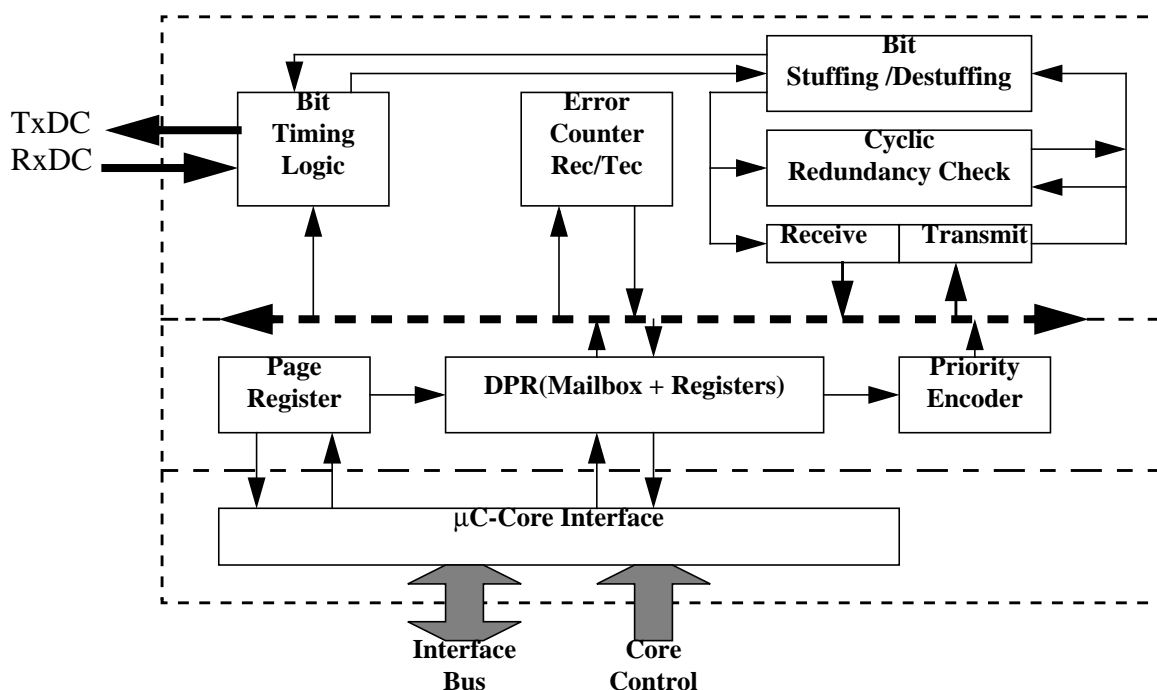


Figure 56. CAN Controller block diagram

15.3. CAN Controller Mailbox and Registers Organization

A pagination allows management of the 48 registers and the 32 (4x8) bytes of the mailbox via 28SFR's. All actions on message object window SFRs are reflected to the corresponding message object registers.

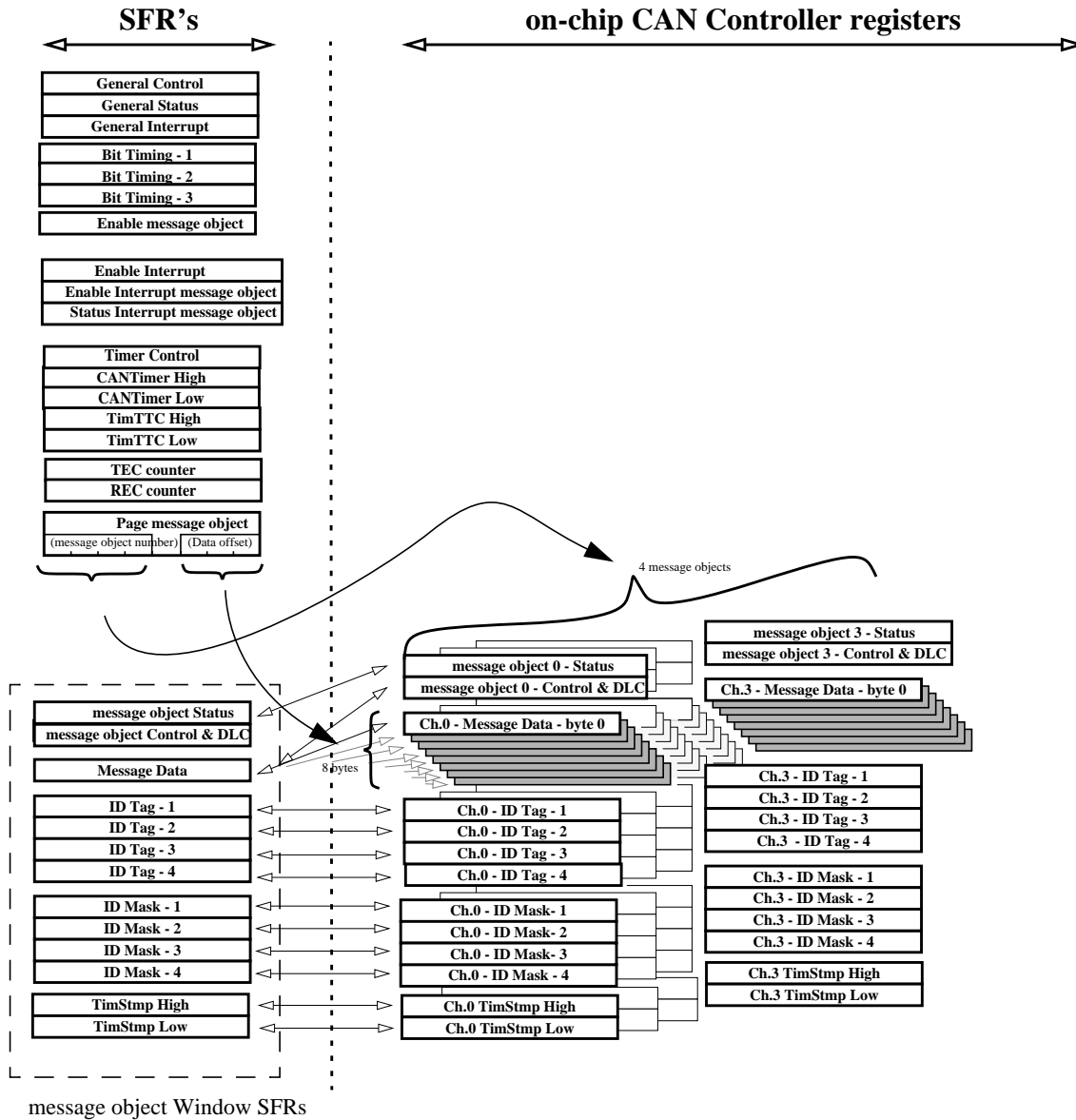


Figure 57. CAN Controller memory organization

15.3.1. Working on message objects

The Page message object register (CANPAGE) is used to select one of the 4 message objects. Then, message object Control (CANCONCH) and message object Status (CANSTCH) are available for this selected message object number in the corresponding SFRs. A single register (CANMSG) is used for the message. The mailbox pointer is managed by the Page message object register with an auto-incrementation at the end of each access. The range of this counter is 8.

Note that the mailbox is a pure RAM, dedicated to one message object, without overlap. In most cases, it is not necessary to transfer the received message into the standard memory. The message to be transmitted can be built directly in the mailbox. Most calculations or tests can be executed in the mailbox area.

15.3.2. CAN Controller management

In order to enable the CAN Controller correctly the following registers have to be initialized:

- General Control (CANGCON),
- Bit Timing (CANBT 1,2&3),
- And for each page
 - message object Control (CANCONCH),
 - message object Status (CANSTCH).

During operation, the CAN Enable message object registers (CANEN) will give a fast overview of the message object availability.

The CAN messages can be handled by interrupt or polling modes.

A message object can be configured as follows:

- Transmit message object,
- Receive message object,
- Receive buffer message object.
- Disable

This configuration is made in the CONCH field of the CANCONCH register (see Table 20).

When a message object is configured, the corresponding ENCH bit of CANEN register is set.

Table 20. Configuration for CONCH1:2

CONCH 1	CONCH 2	Type of message object
0	0	disable
0	1	Transmitter
1	0	Receiver
1	1	Receiver buffer

When a Transmitter or Receiver action of a message object is finished, the corresponding ENCH bit of the CANEN register is cleared. In order to re-enable the message object, it is necessary to re-write the configuration.

Non-consecutive message objects can be used for all three types of message objects (Transmitter, Receiver and Receiver buffer),

15.3.3. Buffer mode

Any message object can be used to define the buffer, including non-consecutive message objects, and with no limitation on length.

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

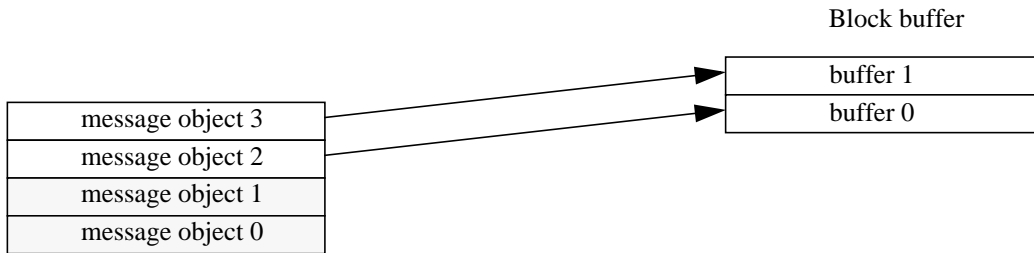


Figure 58. Buffer mode

The same acceptance filter must be defined for each message object 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 for the next reception.

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 be taken into account 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.

15.4. IT CAN management

The different interrupts are:

- Transmission interrupt,
- Reception interrupt,
- Interrupt on error (bit error, stuff error, crc error, form error, acknowledge error),
- Interrupt when Buffer receive is full,
- Interrupt on overrun of CAN Timer.

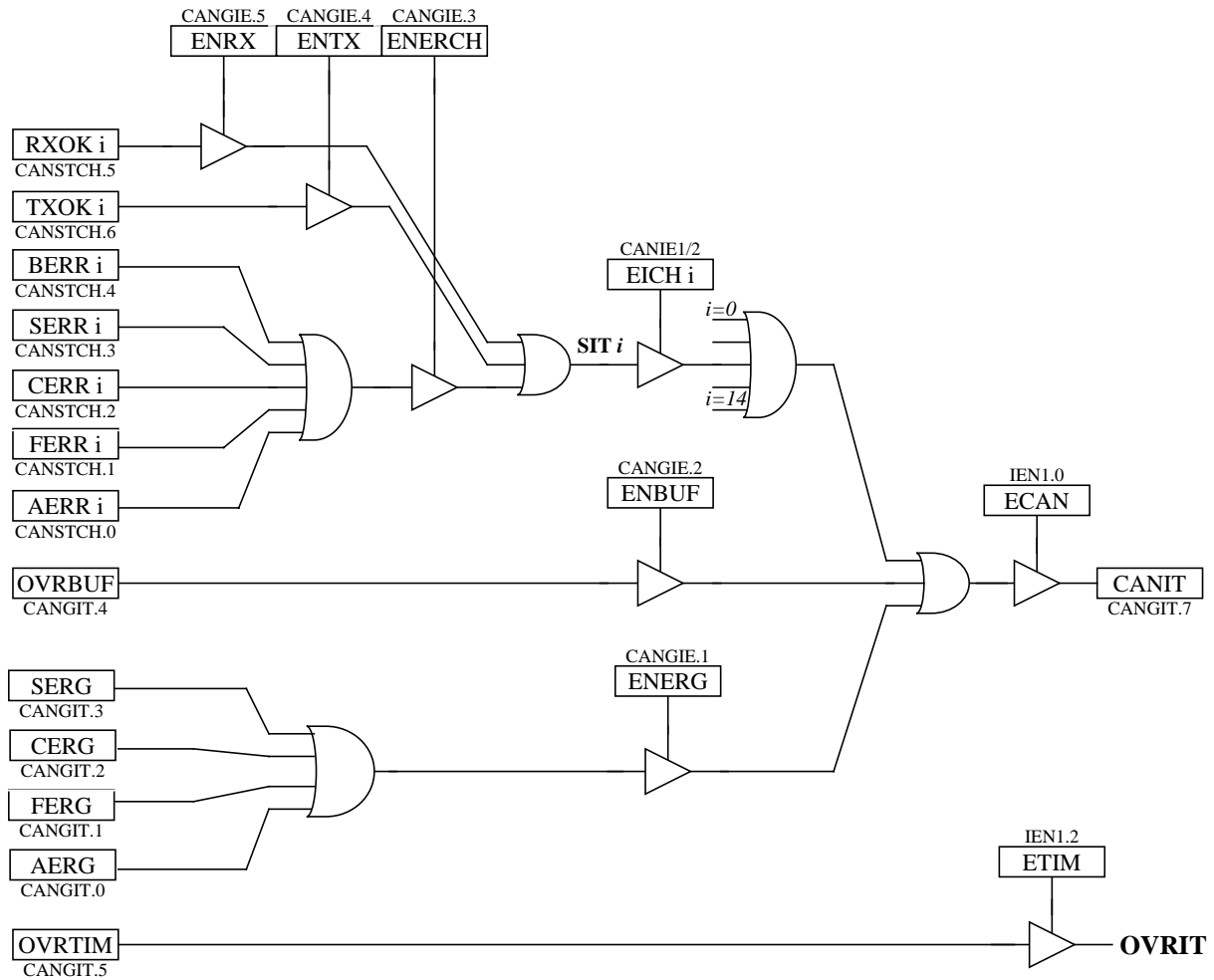


Figure 59. CAN Controller interrupt structure

To enable a transmission interrupt:

- Enable General CAN IT in the interrupt system register,
- Enable interrupt by message object, EICH_i,
- Enable transmission interrupt, ENTX.

To enable a reception interrupt:

- Enable General CAN IT in the interrupt system register,
- Enable interrupt by message object, EICH_i,
- Enable reception interrupt, ENRX.

To enable an interrupt on message object error:

- Enable General CAN IT in the interrupt system register,
- Enable interrupt by message object, EICH_i,
- Enable interrupt on error, ENERCH.

To enable an interrupt on general error:

- Enable General CAN IT in the interrupt system register,
- Enable interrupt on error, ENERG.

To enable an interrupt on Buffer-full condition:

- Enable General CAN IT in the interrupt system register,
- Enable interrupt on Buffer full, ENBUF.

To enable an interrupt when Timer overruns:

- Enable Overrun IT in the interrupt system register.

When an interrupt occurs, the corresponding message object bit is set in the SIT register.

To acknowledge an interrupt, the corresponding CANSTCH bits (RXOK, TXOK,...) or CANGIT bits (OVRTIM, OVRBUF,...), must be cleared by the software application.

When the CAN node is in transmission and detects a Form Error in its frame, a bit Error will also be raised. Consequently, two consecutive interrupts can occur, both due to the same error.

When a message object error occur and set in CANSTCH register, no general error are setting in CANGIE register.

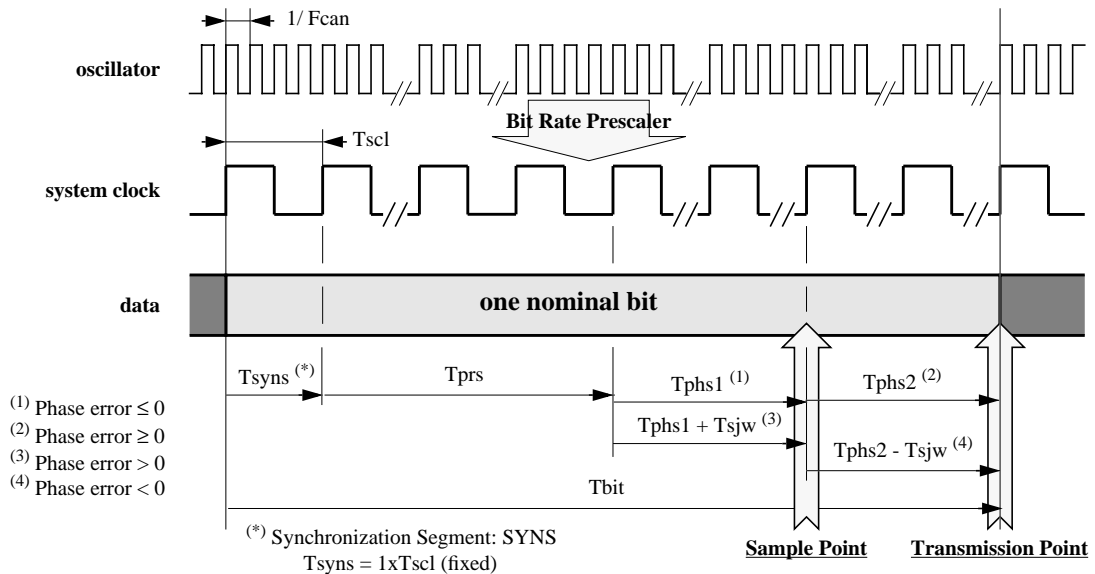
15.5. Bit Timing and BaudRate

The baud rate selection is made by Tbit calculation:

$$T_{bit} = T_{syns} + T_{prs} + T_{phs1} + T_{phs2}$$

1. $T_{syns} = T_{scl} = (BRP[5..0] + 1) / F_{can}$
2. $T_{prs} = (1 \text{ to } 8) * T_{scl} = (PRS[2..0] + 1) * T_{scl}$
3. $T_{phs1} = (1 \text{ to } 8) * T_{scl} = (PHS1[2..0] + 1) * T_{scl}$
4. $T_{phs2} = (1 \text{ to } 8) * T_{scl} = (PHS2[2..0] + 1) * T_{scl}$
5. $T_{sjw} = (1 \text{ to } 4) * T_{scl} = (SJW[1..0] + 1) * T_{scl}$

The total number of T_{scl} (Time Quanta) in a bit time is from 8 to 25.



Tbit calculation: $T_{bit} = T_{syns} + T_{prs} + T_{phs1} + T_{phs2}$

Figure 60. General structure of a bit period

example:

For a Baud Rate of 100 kbit/s and $F_{osc} = 12$ MHz For have 10 TQ:
 BRP = 5
 PRS = 2
 PHS2 = 2
 PHS1 = 2

15.6. Fault Confinement

With respect to fault confinement, a unit may be in one of the three following statuses:

- error active,
- error passive,
- bus off.

An error active unit takes part in bus communication and can send an active error frame when the CAN macro detects an error.

An error passive unit cannot send an active error frame. It takes part in bus communication, but when an error is detected, a passive error frame is sent. Also, after a transmission, an error passive unit will wait before initiating further transmission.

A bus off unit is not allowed to have any influence on the bus.

For fault confinement, two error counters (TEC and REC) are implemented.

See CAN Specification for details on Fault confinement.

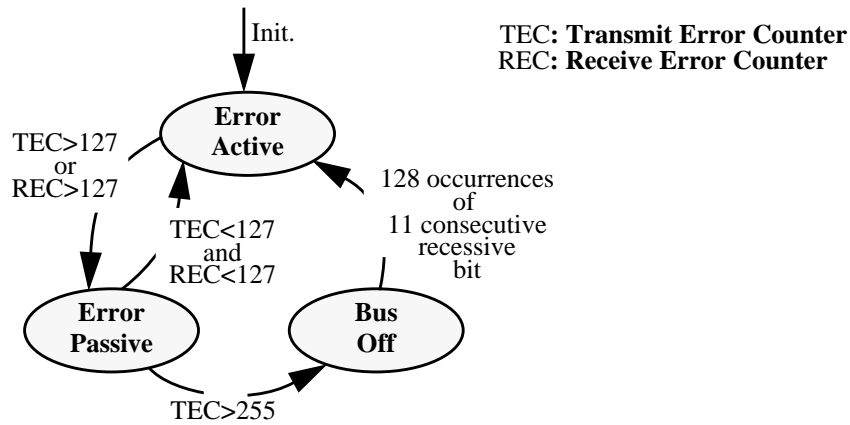


Figure 61. Line error mode

15.7. 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.

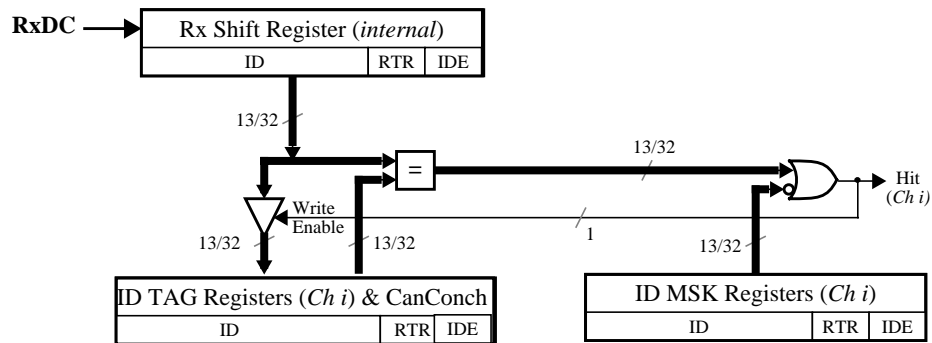


Figure 62. Acceptance filter block diagram

example:

For accept only ID = 318h in part A.

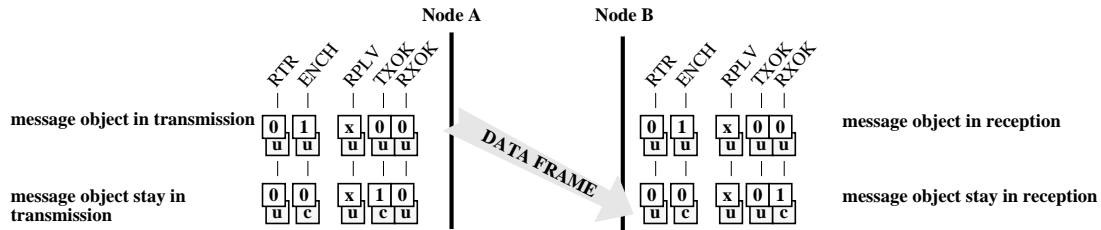
ID MSK = 111 1111 1111 b

ID TAG = 011 0001 1000 b

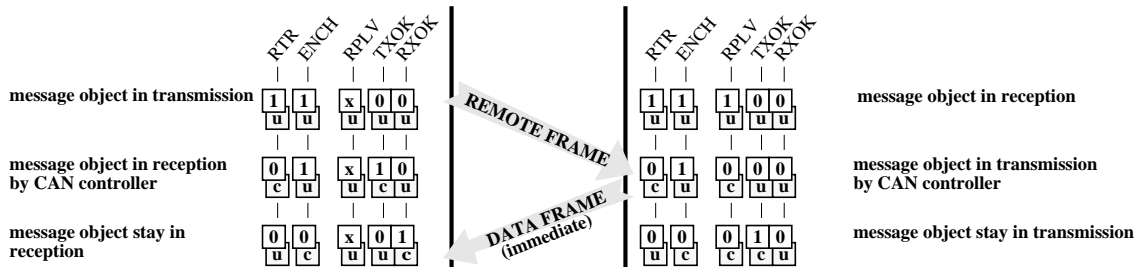
15.8. Data and Remote frame

Description of the different steps for:

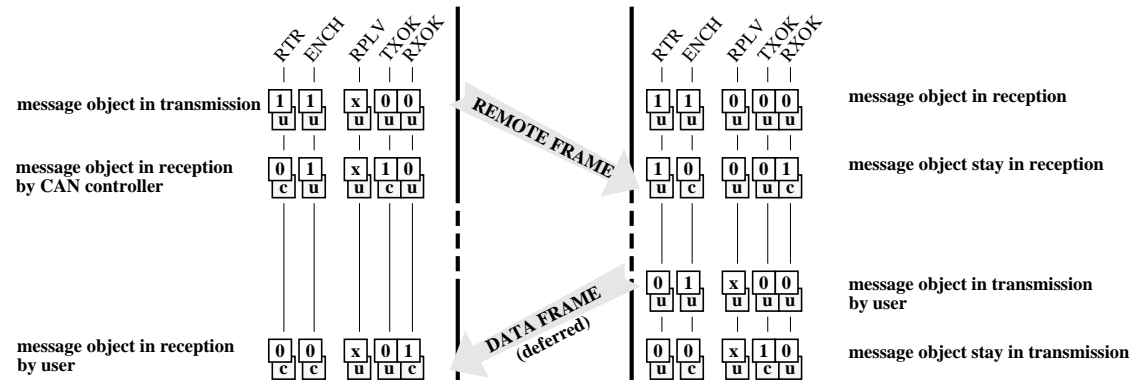
- Data frame,



- Remote frame, with automatic reply,



- Remote frame.



$\begin{matrix} i \\ u \end{matrix}$: modified by user $\begin{matrix} i \\ c \end{matrix}$: modified by CAN

15.9. Time Trigger Communication (TTC) and Message Stamping

The T89C51CC02 has a programmable 16-bit Timer (CANTIMH&CANTIML) for message stamp and TTC.

This CAN Timer starts after the CAN controller is enabled by the ENA bit in the CANGCON register.

Two user modes of the timer are implemented:

- Time Trigger Communication:

Catch of this timer in the CANTTCH & CANTTCL registers on SOF or EOF, depending on the SYNCTTC bit in the CANGCON register, when the network is configured in TTC by the TTC bit in the CANGCON register.

In this mode, CAN *only sends the frame once, even if an error occurs.*
- Message Stamping

Catch of this timer in the CANSTMPH & CANSTMPL registers of the message object which received or sent the frame.

All messages can be stamps.

The stamping of a received frame occurs when the RxOk flag is set.

The stamping of a sent frame occurs when the TxOk flag is set.

The CAN Timer works in a loopback mode (0x0000... 0xFFFF, 0x0000) which serves as a time base to stamp all received or transmitted messages.

When the timer overflows from 0xFFFF to 0x0000, an interrupt is generated if the ETIM bit of the CAN Timer in a micro-controller interrupt system register is set.

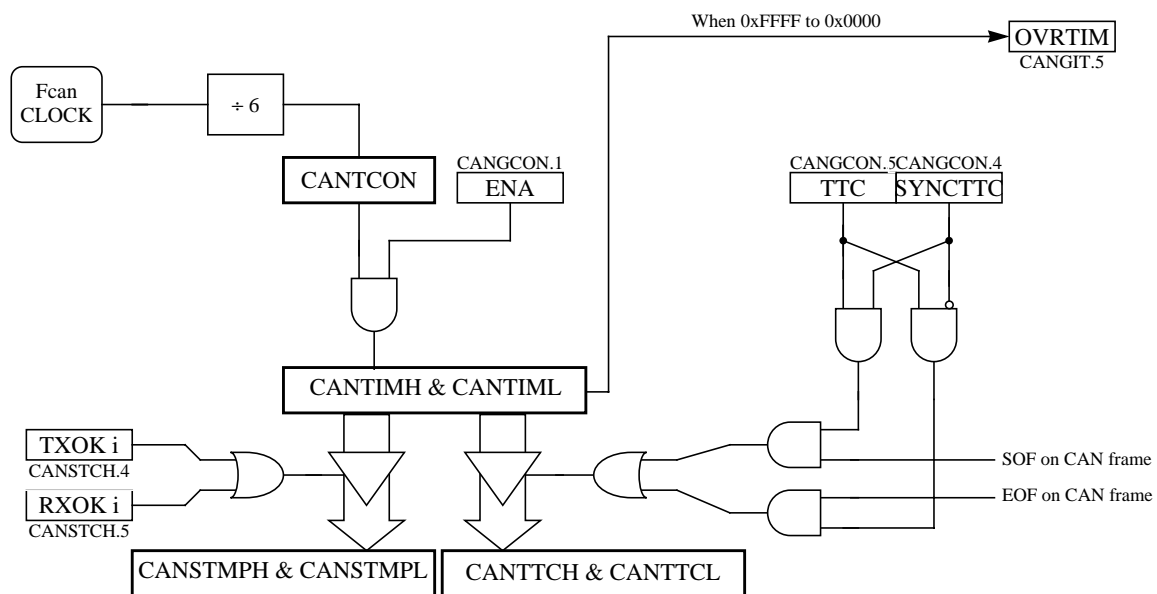


Figure 63. Block diagram of CAN Timer

15.10. CAN Autobaud and Listening mode

To activate the Autobaud feature, the AUTOBAUD bit in the CANGCON register is set. In this mode, the CAN controller is only listening to the line without acknowledging the received messages. It cannot send any message. The error flags are updated. The bit timing can be adjusted until no error occurs (good configuration find).

In this mode, the error counters are frozen.

To go back to the standard mode, the AUTOBAUD bit must be cleared by the software.

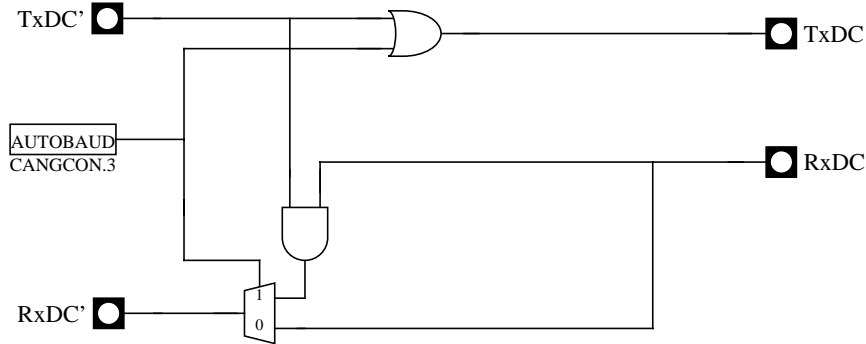


Figure 64. Autobaud Mode

15.11. CAN SFR's

Table 21. 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					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					E7h
E0h	ACC 0000 0000								E7h
D8h	CCON 00xx xx00	CMOD 00xx x000	CCAPM0 x000 0000	CCAPM1 x000 0000					DFh
D0h	PSW 0000 0000	FCON 0000 0000	EECON xxxx xx00						D7h
C8h	T2CON 0000 0000	T2MOD xxxx xx00	RCAP2L 0000 0000	RCAP2H 0000 0000	TL2 0000 0000	TH2 0000 0000		CANEN xxxx 0000	CFh
C0h	P4 xxxx xx11	CANGIE 0000 0000		CANIE xxxx 0000	CANIDM1 xxxx xxxx	CANIDM2 xxxx xxxx	CANIDM3 xxxx xxxx	CANIDM4 xxxx xxxx	C7h
B8h	IPL0 x000 0000	SADEN 0000 0000		CANSIT xxxx 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 xxx xx11	CANTCON 0000 0000	AUXR1 0000 0000	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			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		CKCON 0000 0000	8Fh
80h		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	

15.12. Registers

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-resettable bit. A reset of the ENCH bit (message object control & DLC register) is done for each message object. The pending communications are immediately disabled and the on-going communication will be terminated normally, setting the appropriate status flags, TXOK or RXOK.					
6	OVRQ	Overload frame request (initiator). Auto-resettable 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 0 - no TTC. 1 - node in TTC.					
4	SYNCTTC	Synchronization of TTC When this bit is set to "1" the TTC timer is caught on the last bit of the End Of Frame. When this bit is set to "0" the TTC timer is caught on the Start Of Frame. This bit is only used in the TTC mode.					
3	AUTOBAUD	AUTOBAUD 0 - no autobaud 1 - autobaud mode.					
2	TEST	Test mode. The test mode is intended for factory testing and not for customer use.					
1	ENA/STB	Enable/Standby CAN controller When this bit is set to "1", it enables the CAN controller and its input clock. When this bit is set to "0", 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-resettable 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 0x00b

Figure 65. CANGCON Register

CANGSTA (S:AAh)
CAN General Status Register

7	6	5	4	3	2	1	0
-	OVFG	-	TBSY	RBSY	ENFG	BOFF	ERRP

Bit Number	Bit Mnemonic	Description
7	-	Reserved The values read from this bit is indeterminate. Do not set this bit.
6	OVFG	Overload frame flag (1) This status bit is set by the hardware as long as the produced overload frame is sent. This flag does not generate an interrupt.
5	-	Reserved The values read from this bit is indeterminate. Do not set this bit.
4	TBSY	Transmitter busy (1) This status bit is set by the hardware as long as the CAN transmitter generates a frame (remote, data, overload or error frame) or an ack field. This bit is also active during an InterFrame Spacing if a frame must be sent. This flag does not generate an interrupt.
3	RBSY	Receiver busy (1) This status bit is set by the hardware as long as the CAN receiver acquires or monitors a frame. This flag does not generate an interrupt.
2	ENFG	Enable on-chip CAN controller flag (1) Because an enable/disable command is not effective immediately, this status bit gives the true state of a chosen mode. This flag does not generate an interrupt.
1	BOFF	Bus off mode (1) see Figure 61
0	ERRP	Error passive mode (1) see Figure 61

NOTE:

1. These fields are Read Only.

Reset Value: x0x0 0000b

Figure 66. CANGSTA Register

CANGIT (S:9Bh)
CAN General Interrupt

7	6	5	4	3	2	1	0
CANIT	-	OVRTIM	OVRBUF	SERG	CERG	FERG	AERG

Bit Number	Bit Mnemonic	Description
7	CANIT	General interrupt flag (1) This status bit is the image of all the CAN controller interrupts sent to the interrupt controller. It can be used in the case of the polling method.
6	-	Reserved The values read from this bit is indeterminate. Do not set this bit.
5	OVRTIM	Overflow CAN Timer This status bit is set when the CAN timer switches 0xFFFF to 0x0000. If the ENOVRTIM bit in the IE1 register is set, an interrupt is generated. The user clears this bit in order to reset the interrupt.
4	OVRBUF	Overflow BUFFER 0 - no interrupt. 1 - IT turned on This bit is set when the buffer is full. Bit resettable by user. see Figure 59.
3	SERG	Stuff error General Detection of more than five consecutive bits with the same polarity. This flag can generate an interrupt.
2	CERG	CRC error General The receiver performs a CRC check on each destuffed received message from the start of frame up to the data field. If this checking does not match with the destuffed CRC field, a CRC error is set. This flag can generate an interrupt.
1	FERG	Form error General The form error results from one or more violations of the fixed form in the following bit fields: CRC delimiter acknowledgment delimiter end_of_frame This flag can generate an interrupt.
0	AERG	Acknowledgment error General No detection of the dominant bit in the acknowledge slot. This flag can generate an interrupt.

Reset Value: 0x00 0000b

Figure 67. CANGIT Register

CANTEC (S:9Ch Read Only)
CAN Transmit Error Counter

7	6	5	4	3	2	1	0
TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0

Bit Number	Bit Mnemonic	Description
7-0	TEC7:0	Transmit Error Counter see Figure 61

Reset Value: 00h

Figure 68. CANTEC Register

CANREC (S:9Dh Read Only)
CAN Reception Error Counter

7	6	5	4	3	2	1	0
REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0

Bit Number	Bit Mnemonic	Description
7-0	REC7:0	Reception Error Counter see Figure 61

Reset Value: 00h

Figure 69. CANREC Register

CANGIE (S:C1h)

CAN General Interrupt Enable

7	6	5	4	3	2	1	0
-	-	ENRX	ENTX	ENERCH	ENBUF	ENERG	-

Bit Number	Bit Mnemonic	Description
7-6	-	Reserved The values read from these bits are indeterminate. Do not set these bits.
5	ENRX	Enable receive interrupt 0 - Disable 1 - Enable
4	ENTX	Enable transmit interrupt 0 - Disable 1 - Enable
3	ENERCH	Enable message object error interrupt 0 - Disable 1 - Enable
2	ENBUF	Enable BUF interrupt 0 - Disable 1 - Enable
1	ENERG	Enable general error interrupt 0 - Disable 1 - Enable
0	-	Reserved The value read from this bit is indeterminate. Do not set this bit.

NOTE:

see Figure 59

Reset Value: xx00 000xb

Figure 70. CANGIE Register

CANEN (S:CFh Read Only)

CAN Enable message object Registers

7	6	5	4	3	2	1	0
-	-	-	-	ENCH3	ENCH2	ENCH1	ENCH0

Bit Number	Bit Mnemonic	Description
7-4	-	Reserved The value read from these bit are indeterminate. Do not set these bits.
3-0	ENCH3:0	Enable message object 0 - message object is disabled => the message object is free for a new emission or reception. 1 - message object is enabled. This bit is resettable by re-writing the CANCONCH of the corresponding message object.

Reset Value: xxxx 0000b

Figure 71. CANEN Register

CANSIT (S:BBh Read Only)

CAN Status Interrupt message object Registers

7	6	5	4	3	2	1	0
-	-	-	-	SIT3	SIT2	SIT1	SIT0

Bit Number	Bit Mnemonic	Description
7-4	-	Reserved The value read from these bit are indeterminate. Do not set these bits.
3-0	SIT3:0	Status of interrupt by message object 0 - no interrupt. 1 - IT turned on. Reset when interrupt condition is cleared by user. example: CANSIT = 0b 0000 1001 -> IT's on message objects 3 & 0. see Figure 59.

Reset Value: xxxx 0000b

Figure 72. CANSIT Register

CANIE (S:C3h)

CAN Enable Interrupt message object Registers

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

Bit Number	Bit Mnemonic	Description
7-4	-	Reserved The value read from these bit are indeterminate. Do not set these bits.
3-0	IECH3:0	Enable interrupt by message object 0 - disable IT. 1 - enable IT. example: CANIE= 0b 0000 1100 -> Enable IT's of message objects 3 & 0.

Reset Value: xxxx 0000b

Figure 73. CANIE Register

CANBT1 (S:B4h)

CAN Bit Timing Registers 1

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

Bit Number	Bit Mnemonic	Description
7	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
6-1	BRP5:0	Baud rate prescaler The period of the CAN controller system clock T_{scl} is programmable and determines the individual bit timing. $T_{scl} = \frac{BRP[5...0] + 1}{F_{can}}$
0	-	Reserved The value read from this bit is indeterminate. Do not set this bit.

Note:

The CAN controller bit timing registers must be accessed only if the CAN controller is disabled with the ENA bit of the CANGCON register set to 0. See Figure 60.

No default value after reset.

Figure 74. CANBT1 Register

CANBT2 (S:B5h)
CAN Bit Timing Registers 2

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

Bit Number	Bit Mnemonic	Description
7	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
6-5	SJW1:0	Re-synchronization jump width To compensate for phase shifts between clock oscillators of different bus controllers, the controller must re-synchronize on any relevant signal edge of the current transmission. The synchronization jump width defines the maximum number of clock cycles. A bit period may be shortened or lengthened by a re-synchronization. $T_{sjw} = T_{scl} \times (SJW[1, 0] + 1)$
4	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
3-1	PRS2:0	Programming time segment This part of the bit time is used to compensate for the physical delay times within the network. It is twice the sum of the signal propagation time on the bus line, the input comparator delay and the output driver delay. $T_{prs} = T_{scl} \times (PRS[2...0] + 1)$
0	-	Reserved The value read from this bit is indeterminate. Do not set this bit.

Note:

The CAN controller bit timing registers must be accessed only if the CAN controller is disabled with the ENA bit of the CANGCON register set to 0. See Figure 60.

No default value after reset.

Figure 75. CANBT2 Register

CANBT3 (S:B6h)

CAN Bit Timing Registers 3

7	6	5	4	3	2	1	0
-	PHS2 2	PHS2 1	PHS2 0	PHS1 2	PHS1 1	PHS1 0	SMP

Bit Number	Bit Mnemonic	Description
7	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
6-4	PHS2 2:0	Phase segment 2 This phase is used to compensate for phase edge errors. This segment can be shortened by the re-synchronization jump width. $T_{phs2} = T_{scl} \times (PHS2[2...0] + 1)$
3-1	PHS1 2:0	Phase segment 1 This phase is used to compensate for phase edge errors. This segment can be lengthened by the re-synchronization jump width. $T_{phs1} = T_{scl} \times (PHS1[2...0] + 1)$
0	SMP	Sample type 0 - once, at the sample point. 1 - three times, the threefold sampling of the bus is the sample point and twice over a distance of a 1/2 period of the T _{scl} . The result corresponds to the majority decision of the three values.

Note:

The CAN controller bit timing registers must be accessed only if the CAN controller is disabled with the ENA bit of the CANGCON register set to 0. See Figure 60.

No default value after reset.

Figure 76. CANBT3 Register

CANPAGE (S:B1h)

CAN message object Page Register

7	6	5	4	3	2	1	0
-	-	CHNB 1	CHNB 0	$\overline{\text{AINC}}$	INDX2	INDX1	INDX0

Bit Number	Bit Mnemonic	Description
7-6	-	Reserved The value read from these bit are indeterminate. Do not set these bits.
5-4	CHNB1:0	Selection of message object number The available numbers are: 0 to 3 (see Figure 57).
3	$\overline{\text{AINC}}$	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 57).

Reset Value: 0000 0000b

Figure 77. CANPAGE 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: Transmitter 1 0: Receiver 1 1: Receiver 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 & 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. See Figure 62.

No default value after reset

Figure 78. CANCONCH Register

CANSTCH (S:B2h)

CAN message object Status Register

7	6	5	4	3	2	1	0
DLCW	TXOK	RXOK	BERR	SERR	CERR	FERR	AERR

Bit Number	Bit Mnemonic	Description
7	DLCW	Data length code warning The incoming message does not have the DLC expected. Whatever the frame type, the DLC field of the CANCONCH register is updated by the received DLC.
6	TXOK	Transmit OK The communication enabled by transmission is completed. When the controller is ready to send a frame, if two or more message objects are enabled as producers, the lower index message object (0 to 13) is supplied first. This flag can generate an interrupt.
5	RXOK	Receive OK The communication enabled by reception is completed. In the case of two or more message object reception hits, the lower index message object (0 to 13) is updated first. This flag can generate an interrupt.
4	BERR	Bit error (only in transmission) The bit value monitored is different from the bit value sent. Exceptions: the monitored recessive bit sent as a dominant bit during the arbitration field and the acknowledge slot detecting a dominant bit during the sending of an error frame. This flag can generate an interrupt.
3	SERR	Stuff error Detection of more than five consecutive bits with the same polarity. This flag can generate an interrupt.
2	CERR	CRC error The receiver performs a CRC check on each destuffed received message from the start of frame up to the data field. If this checking does not match with the destuffed CRC field, a CRC error is set. This flag can generate an interrupt.
1	FERR	Form error The form error results from one or more violations of the fixed form in the following bit fields: CRC delimiter acknowledgment delimiter end_of_frame This flag can generate an interrupt.
0	AERR	Acknowledgment error No detection of the dominant bit in the acknowledge slot. This flag can generate an interrupt.

NOTE:

See Figure 59.

No default value after reset.

Figure 79. CANSTCH Register

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 62.

No default value after reset.

Figure 80. CANIDT1 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 62.
4-0	-	Reserved The values read from these bits are indeterminate. Do not set these bits.

No default value after reset.

Figure 81. CANIDT2 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.

Figure 82. CANIDT3 Register for V2.0 part A

CANIDT4 for V2.0 part A (S:BFh)

CAN Identifier Tag Registers 4

7	6	5	4	3	2	1	0
-	-	-	-	-	RTRTAG	-	RB0TAG

Bit Number	Bit Mnemonic	Description
7-3	-	Reserved The values read from these bits are indeterminate. Do not set these bits.
2	RTRTAG	Remote transmission request tag value.
1	-	Reserved The values read from this bit are indeterminate. Do not set these bit.
0	RB0TAG	Reserved bit 0 tag value.

No default value after reset.

Figure 83. CANIDT4 Register for V2.0 part A

CANIDT1 for V2.0 part B (S:BCh)

CAN Identifier Tag Registers 1

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

Bit Number	Bit Mnemonic	Description
7-0	IDT28:21	Identifier tag value See Figure 62.

No default value after reset.

Figure 84. CANIDT1 Register for V2.0 part B

CANIDT2 for V2.0 part B (S:BDh)

CAN Identifier Tag Registers 2

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

Bit Number	Bit Mnemonic	Description
7-0	IDT20:13	Identifier tag value See Figure 62.

No default value after reset.

Figure 85. CANIDT2 Register for V2.0 part B

CANIDT3 for V2.0 part B (S:BEh)

CAN Identifier Tag Registers 3

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

Bit Number	Bit Mnemonic	Description
7-0	IDT12:5	Identifier tag value See Figure 62.

No default value after reset.

Figure 86. CANIDT3 Register for V2.0 part B

CANIDT4 for V2.0 part B (S:BFh)

CAN Identifier Tag Registers 4

7	6	5	4	3	2	1	0
IDT 4	IDT 3	IDT 2	IDT 1	IDT 0	RTRTAG	RB1TAG	RB0TAG

Bit Number	Bit Mnemonic	Description
7-3	IDT4:0	Identifier tag value See Figure 62.
2	RTRTAG	Remote transmission request tag value
1	RB1TAG	Reserved bit 1 tag value.
0	RB0TAG	Reserved bit 0 tag value.

No default value after reset.

Figure 87. CANIDT4 Register for V2.0 part B

CANIDM1 for V2.0 part A (S:C4h)

CAN Identifier Mask Registers 1

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

Bit Number	Bit Mnemonic	Description
7-0	IDTMSK10:3	Identifier mask value 0 - comparison true forced. 1 - bit comparison enabled. See Figure 62.

No default value after reset.

Figure 88. CANIDM1 Register for V2.0 part A

CANIDM2 for V2.0 part A (S:C5h)

CAN Identifier Mask Registers 2

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

Bit Number	Bit Mnemonic	Description
7-5	IDTMSK2:0	Identifier mask value 0 - comparison true forced. 1 - bit comparison enabled. See Figure 62.
4-0	-	Reserved The values read from these bits are indeterminate. Do not set these bits.

No default value after reset.

Figure 89. CANIDM2 Register for V2.0 part A

CANIDM3 for V2.0 part A (S:C6h)

CAN Identifier Mask 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.

No default value after reset.

Figure 90. CANIDM3 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.

Figure 91. CANIDM4 Register for V2.0 part A

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 62.

NOTE:

The ID Mask is only used for reception.

No default value after reset.

Figure 92. CANIDM1 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 62.					

NOTE:

The ID Mask is only used for reception.

No default value after reset.

Figure 93. CANIDM2 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 62.					

NOTE:

The ID Mask is only used for reception.

No default value after reset.

Figure 94. CANIDM3 Register for V2.0 part B

CANIDM4 for V2.0 part B (S:C7h) CAN Identifier Mask Registers 4

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

Bit Number	Bit Mnemonic	Description
7-3	IDMSK4:0	Identifier mask value 0 - comparison true forced. 1 - bit comparison enabled. See Figure 62.
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.

Figure 95. CANIDM4 Register for V2.0 part B

CANMSG (S:A3h)

CAN Message Data Register

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

Bit Number	Bit Mnemonic	Description
7-0	MSG7:0	Message data This register contains the mailbox data byte pointed at the page message object register. After writing in the page message object register, this byte is equal to the specified message location (in the mailbox) of the pre-defined identifier + index. If auto-incrementation is used, at the end of the data register writing or reading cycle, the mailbox pointer is auto-incremented. The dynamic of the counting is 8 with no end loop (0, 1,..., 7, 0,...)

No default value after reset.

Figure 96. CANMSG Register

CANTCON (S:A1h)

CAN Timer ClockControl

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

Bit Number	Bit Mnemonic	Description
7-0	TPRESC7:0	Timer Prescaler of CAN Timer This register is a prescaler for the main timer upper counter range = 0 to 255. See Figure 63.

Reset Value: 00h

Figure 97. CANTCON Register

CANTIMH (S:ADh Read Only)

CAN Timer High

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

Bit Number	Bit Mnemonic	Description
7-0	CANGTIM15:8	High byte of Message Timer See Figure 63.

Reset Value: 0000 0000b

Figure 98. CANTIMH Register

CANTIML (S:ACh Read Only)

CAN Timer Low

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

Bit Number	Bit Mnemonic	Description
7-0	CANGTIM7:0	Low byte of Message Timer See Figure 63.

Reset Value: 0000 0000b

Figure 99. CANTIML 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 63.

No default value after reset

Figure 100. CANSTMPH 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 63.

No default value after reset

Figure 101. CANSTMPL 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 63.

Reset Value: 0000 0000b

Figure 102. CANTTCH Register

CANTTCL (S:A4h Read Only)

CAN TTC Timer Low

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

Bit Number	Bit Mnemonic	Description
7-0	TIMTTC7:0	Low byte of TTC Timer See Figure 63.

Reset Value: 0000 0000b

Figure 103. CANTTCL Register

16. Programmable Counter Array PCA

16.1. Introduction

The PCA provides more timing capabilities with less CPU intervention than the standard timer/counters. Its advantages include reduced software overhead and improved accuracy. The PCA consists of a dedicated timer/counter which serves as the time base for an array of two compare/capture modules. Its clock input can be programmed to count any of the following signals:

- PCA clock frequency / 6
- PCA clock frequency / 2
- Timer 0 overflow
- External input on ECI (P1.2)

Each compare/capture modules can be programmed in any one of the following modes:

- rising and/or trailing edge capture,
- software timer,
- high-speed output,
- pulse width modulator.

When the compare/capture modules are programmed in capture mode, software timer, or high speed output mode, an interrupt can be generated when the module executes its function. All two modules plus the PCA timer overflow share one interrupt vector.

The PCA timer/counter and compare/capture modules share Port 1 for external I/Os. These pins are listed below. If the port is not used for the PCA, it can still be used for standard I/O.

PCA component	External I/O Pin
16-bit Counter	P1.2 / ECI
16-bit Module 0	P1.3 / CEX0
16-bit Module 1	P1.4 / CEX1

The PCA timer is a common time base for all modules (see Figure 9). The timer count source is determined from the CPS1 and CPS0 bits in the **CMOD SFR** (see Table 8) and can be programmed to run at:

- 1/6 the PCA clock frequency.
- 1/2 the PCA clock frequency.
- the Timer 0 overflow.
- the input on the ECI pin (P1.2).

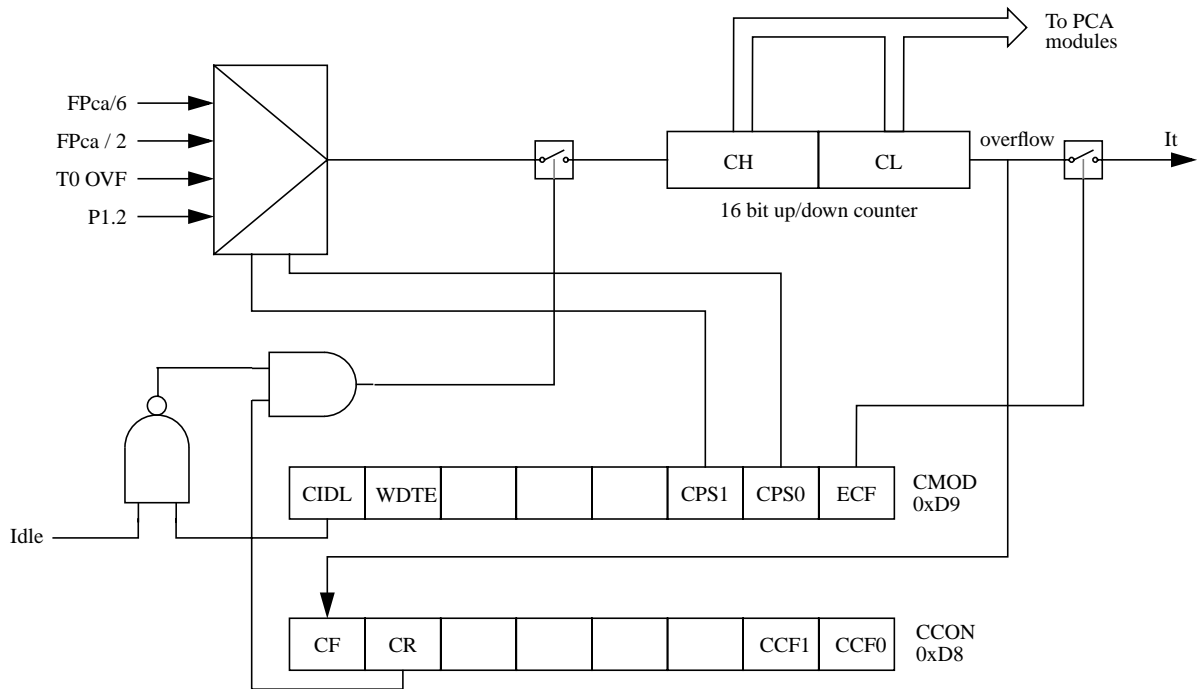


Figure 104. PCA Timer/Counter

16.2. PCA Interrupt

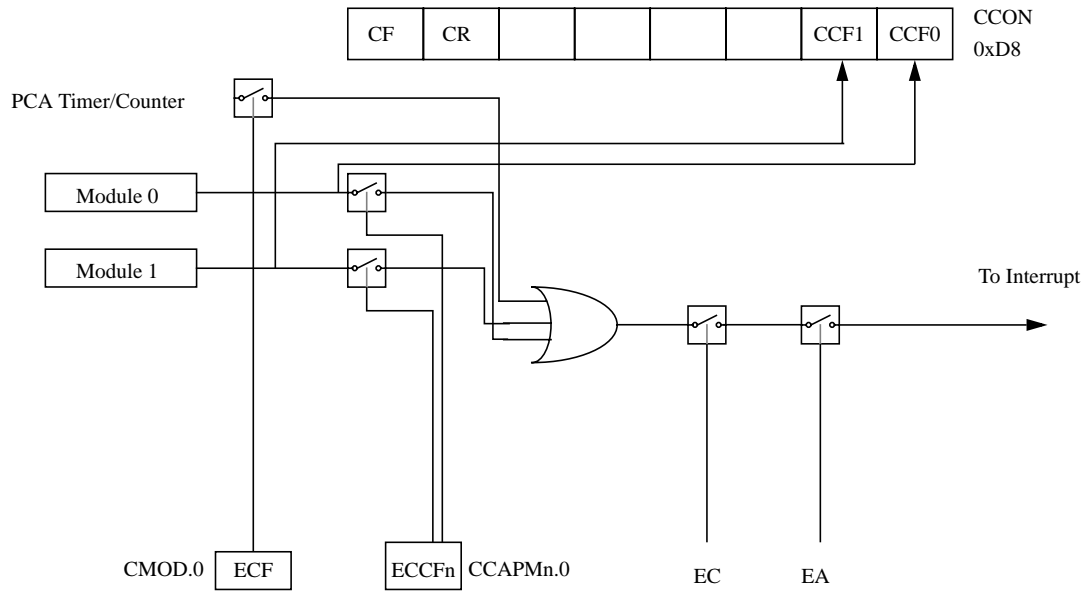


Figure 105. PCA Timer Interrupts

16.3. PCA Capture Mode

To use one of the PCA modules in capture mode either one or both of the CCAPM bits CAPN and CAPP for that module must be set. The external CEX input for the module (on port 1) is sampled for a transition. When a valid transition occurs the PCA hardware loads the value of the PCA counter registers (CH and CL) into the module's capture registers (CCAPnL and CCAPnH). If the CCFn bit for the module in the CCON SFR and the ECCFn bit in the CCAPMn SFR are set then an interrupt will be generated.

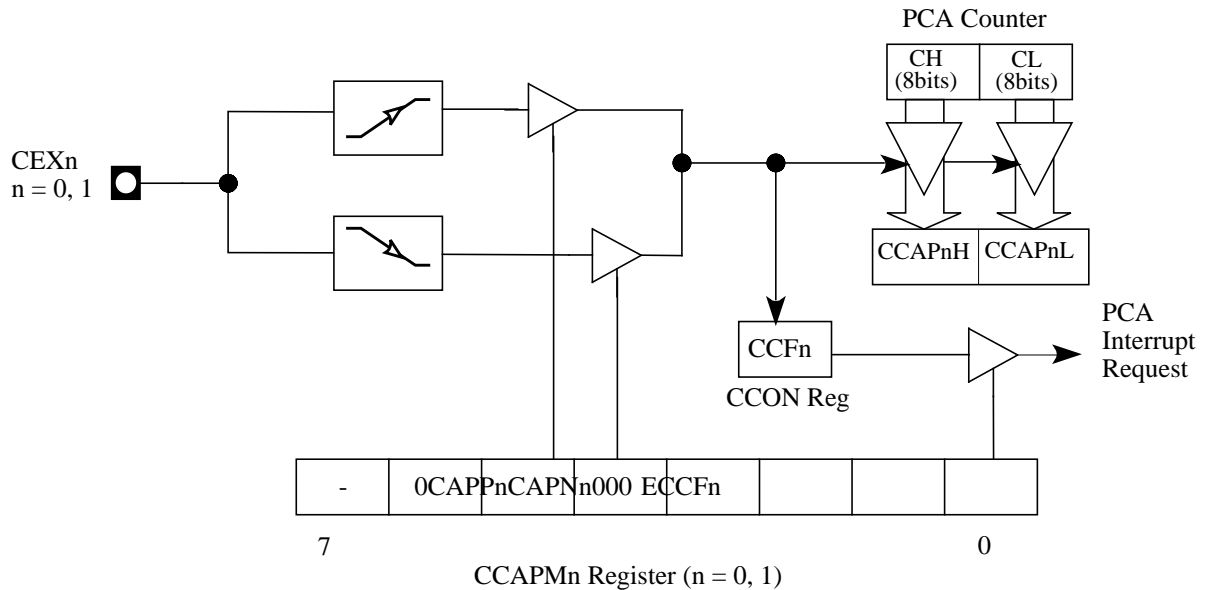


Figure 106. PCA Capture Mode

16.4. 16-bit Software Timer Mode

The PCA modules can be used as software timers by setting both the ECOM and MAT bits in the modules CCAPMn register. The PCA timer will be compared to the module's capture registers and when a match occurs an interrupt will occur if the CCFn (CCON SFR) and the ECCFn (CCAPMn SFR) bits for the module are both set.

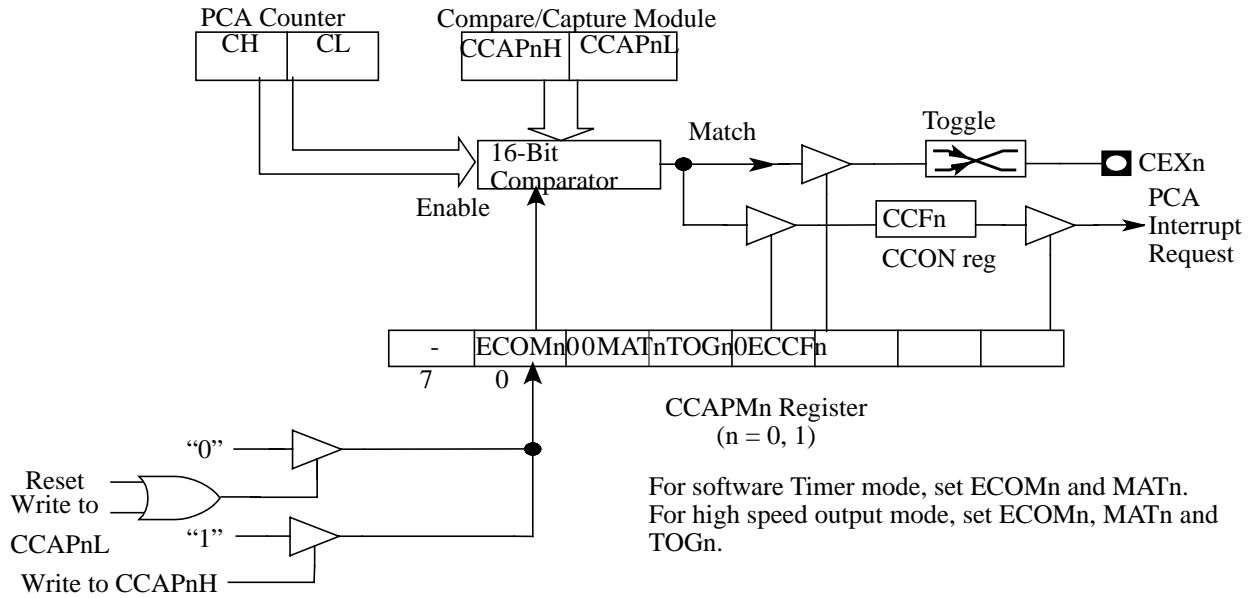


Figure 107. PCA 16-bit Software Timer and High Speed Output Mode

16.5. 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.

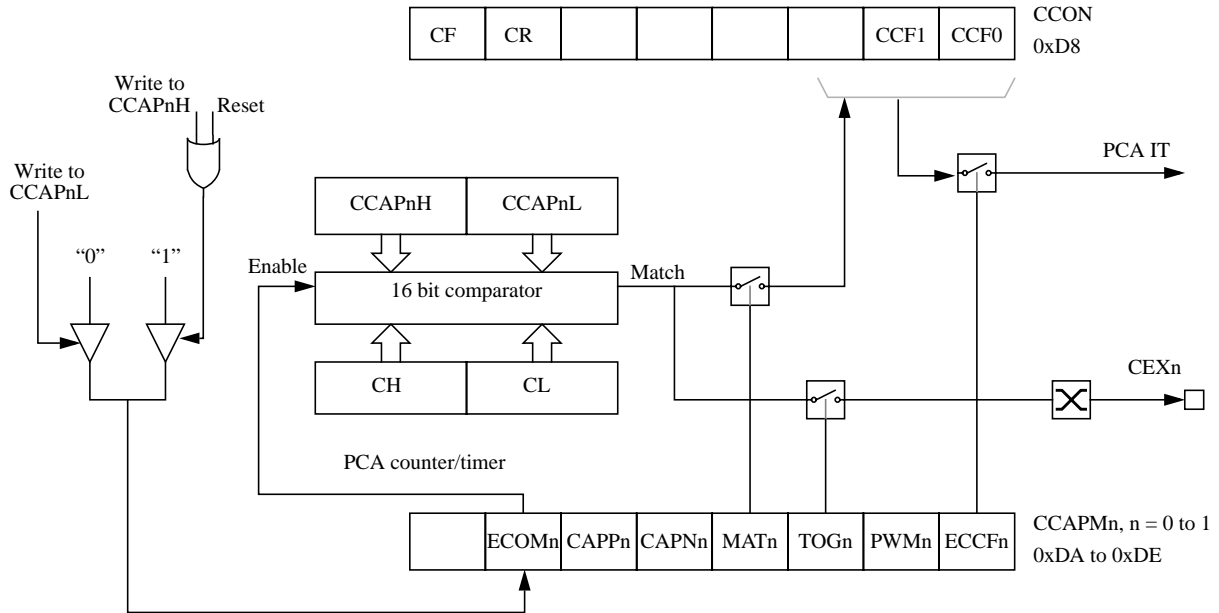


Figure 108. PCA High speed Output Mode

16.6. 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. This 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.

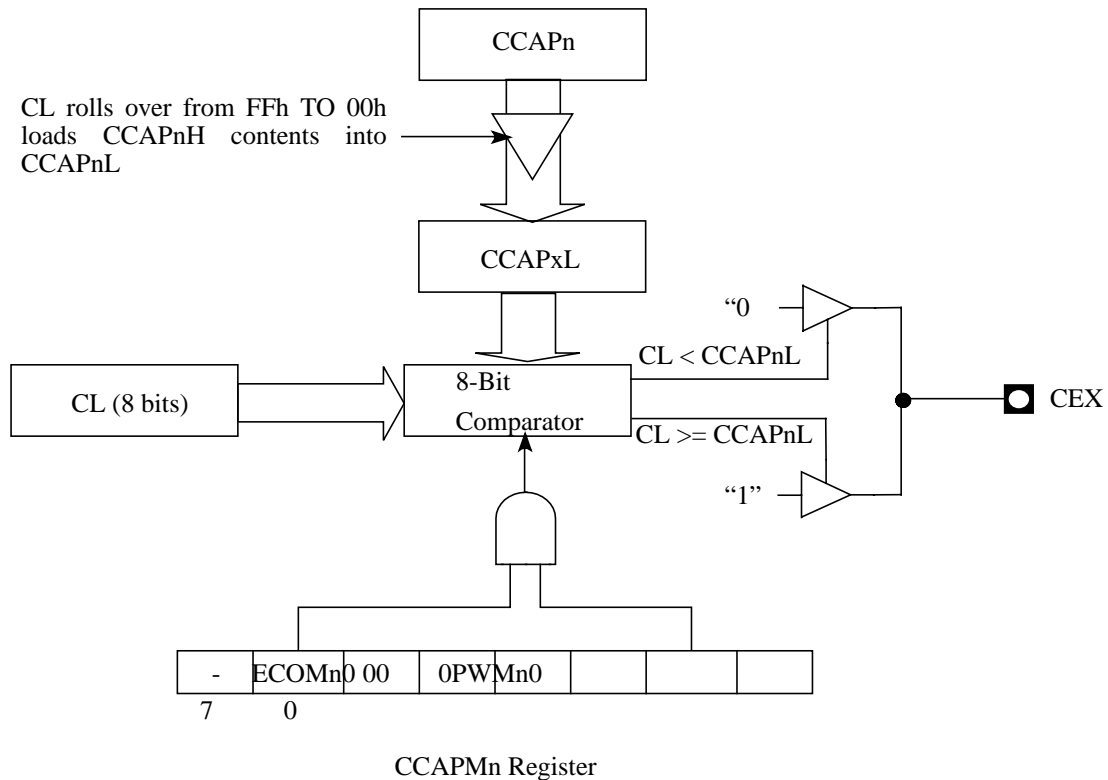


Figure 109. PCA PWM Mode

16.7. PCA Registers

CMOD (S:D8h)

PCA Counter Mode Register

7	6	5	4	3	2	1	0
CIDL	WDTE	-	-	-	CPS1	CPS0	ECF

Bit Number	Bit Mnemonic	Description															
7	CIDL	PCA Counter Idle Control bit Clear to let the PCA run during Idle mode. Set to stop the PCA when Idle mode is invoked.															
6	WDTE	Watchdog Timer Enable Clear to disable Watchdog Timer function on PCA Module 4, Set to enable it.															
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	CPS1	EWC Count Pulse Select bits <table border="1"> <thead> <tr> <th>CPS1</th> <th>CPS0</th> <th>Clock source</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Internal Clock, FPca/6</td> </tr> <tr> <td>0</td> <td>1</td> <td>Internal Clock, FPca/2</td> </tr> <tr> <td>1</td> <td>0</td> <td>Timer 0 overflow</td> </tr> <tr> <td>1</td> <td>1</td> <td>External clock at ECI/P1.2 pin (Max. Rate = FPca/4)</td> </tr> </tbody> </table>	CPS1	CPS0	Clock source	0	0	Internal Clock, FPca/6	0	1	Internal Clock, FPca/2	1	0	Timer 0 overflow	1	1	External clock at ECI/P1.2 pin (Max. Rate = FPca/4)
CPS1	CPS0	Clock source															
0	0	Internal Clock, FPca/6															
0	1	Internal Clock, FPca/2															
1	0	Timer 0 overflow															
1	1	External clock at ECI/P1.2 pin (Max. Rate = FPca/4)															
1	CPS0																
0	ECF	Enable PCA Counter Overflow Interrupt bit Clear to disable CF bit in CCON register to generate an interrupt. Set to enable CF bit in CCON register to generate an interrupt.															

Reset Value = 00XX X000b

Figure 110. CMOD Register

CCON (S:D8h)

PCA Counter Control Register

7	6	5	4	3	2	1	0
CF	CR	-				CCF1	CCF0

Bit Number	Bit Mnemonic	Description
7	CF	PCA Timer/Counter Overflow flag Set by hardware when the PCA Timer/Counter rolls over. This generates a PCA interrupt request if the ECF bit in CMOD register is set. Must be cleared by software.
6	CR	PCA Timer/Counter Run Control bit Clear to turn the PCA Timer/Counter off. Set to turn the PCA Timer/Counter on.
5-2	-	Reserved The value read from these bits are indeterminate. Do not set these bits.
1	CCF1	PCA Module 1 Compare/Capture flag Set by hardware when a match or capture occurs. This generates a PCA interrupt request if the ECCF 1 bit in CCAPM 1 register is set. Must be cleared by software.
0	CCF0	PCA Module 0 Compare/Capture flag Set by hardware when a match or capture occurs. This generates a PCA interrupt request if the ECCF 0 bit in CCAPM 0 register is set. Must be cleared by software.

Reset Value = 00xx xx00b

Figure 111. CCON Register

CCAP0H (S:FAh)

CCAP1H (S:FBh)

PCA High Byte Compare/Capture Module n Register (n=0..1)

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

Bit Number	Bit Mnemonic	Description
7:0	CCAPnH 7:0	High byte of EWC-PCA comparison or capture values

Reset Value = 0000 0000b

Figure 112. CCAPnH Registers

CCAP0L (S:EAh)

CCAP1L (S:EBh)

PCA Low Byte Compare/Capture Module n Register (n=0..1)

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

Bit Number	Bit Mnemonic	Description
7:0	CCAPnL 7:0	Low byte of EWC-PCA comparison or capture values

Reset Value = 0000 0000b

Figure 113. CCAPnL Registers

CCAPM0 (S:DAh)

CCAPM1 (S:DBh)

PCA Compare/Capture Module n Mode registers (n=0..1)

7	6	5	4	3	2	1	0
-	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn

Bit Number	Bit Mnemonic	Description
7	-	Reserved The Value read from this bit is indeterminate. Do not set this bit.
6	ECOMn	Enable Compare Mode Module x bit Clear to disable the Compare function. Set to enable the Compare function. The Compare function is used to implement the software Timer, the high-speed output, the Pulse Width Modulator (PWM) and the Watchdog Timer (WDT).
5	CAPPn	Capture Mode (Positive) Module x bit Clear to disable the Capture function triggered by a positive edge on CEXx pin. Set to enable the Capture function triggered by a positive edge on CEXx pin
4	CAPNn	Capture Mode (Negative) Module x bit Clear to disable the Capture function triggered by a negative edge on CEXx pin. Set to enable the Capture function triggered by a negative edge on CEXx pin.
3	MATn	Match Module x bit Set when a match of the PCA Counter with the Compare/Capture register sets CCFx bit in CCON register, flagging an interrupt. Must be cleared by software.
2	TOGn	Toggle Module x bit The toggle mode is configured by setting ECOMx, MATx and TOGx bits. Set when a match of the PCA Counter with the Compare/Capture register toggles the CEXx pin. Must be cleared by software.
1	PWMn	Pulse Width Modulation Module x Mode bit Set to configure the module x as an 8-bit Pulse Width Modulator with output waveform on CEXx pin. Must be cleared by software.
0	ECCFn	Enable CCFx Interrupt bit Clear to disable CCFx bit in CCON register to generate an interrupt request. Set to enable CCFx bit in CCON register to generate an interrupt request.

Reset Value = X000 0000b

Figure 114. CCAPMn Registers

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 0000b

Figure 115. CH 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 0000b

Figure 116. CL Register

17. Analog-to-Digital Converter (ADC)

17.1. Introduction

This section describes the on-chip 10 bit analog-to-digital converter of the T89C51CC02. Eight ADC channels are available for sampling of the external sources AN0 to AN7. An analog multiplexer allows the single ADC converter to select one from the 8 ADC channels as ADC input voltage (ADCIN). ADCIN is converted by the 10 bit-cascaded potentiometric ADC.

Two kind of conversion are available:

- Standard conversion (8 bits).
- Precision conversion (10 bits).

For the precision conversion, set bit PSIDLE in ADCON register and start conversion. The chip is in a pseudo-idle mode, the CPU doesn't run but the peripherals are always running. This mode allows digital noise to be as low as possible, to ensure high precision conversion.

For this mode it is necessary to work with end of conversion interrupt, which is the only way to wake up the chip. If another interrupt occurs during the precision conversion, it will be treated only after this conversion is ended.

17.2. Features

- 8 channels with multiplexed inputs
- 10-bit cascaded potentiometric ADC
- Conversion time 20 micro-seconds
- Zero Error (offset) +/- 2 LSB max
- Positive Reference Voltage Range 2.4 to 3.0Volt
- VACC Analog supply voltage for ADC
- ADCIN Range 0 to 3Volt
- Integral non-linearity typical 1 LSB, max. 2 LSB
- Differential non-linearity typical 0.5 LSB, max. 1 LSB
- Conversion Complete Flag or Conversion Complete Interrupt
- Selected ADC Clock

17.3. ADC Port1 I/O Functions

Port 1 pins are general I/O that are shared with the ADC channels. The channel select bit in ADCF register define which ADC channel/port1 pin will be used as ADCIN. The remaining ADC channels/port1 pins can be used as general purpose I/O or as the alternate function that is available. Writes to the port register which aren't selected by the ADCF will not have any effect.

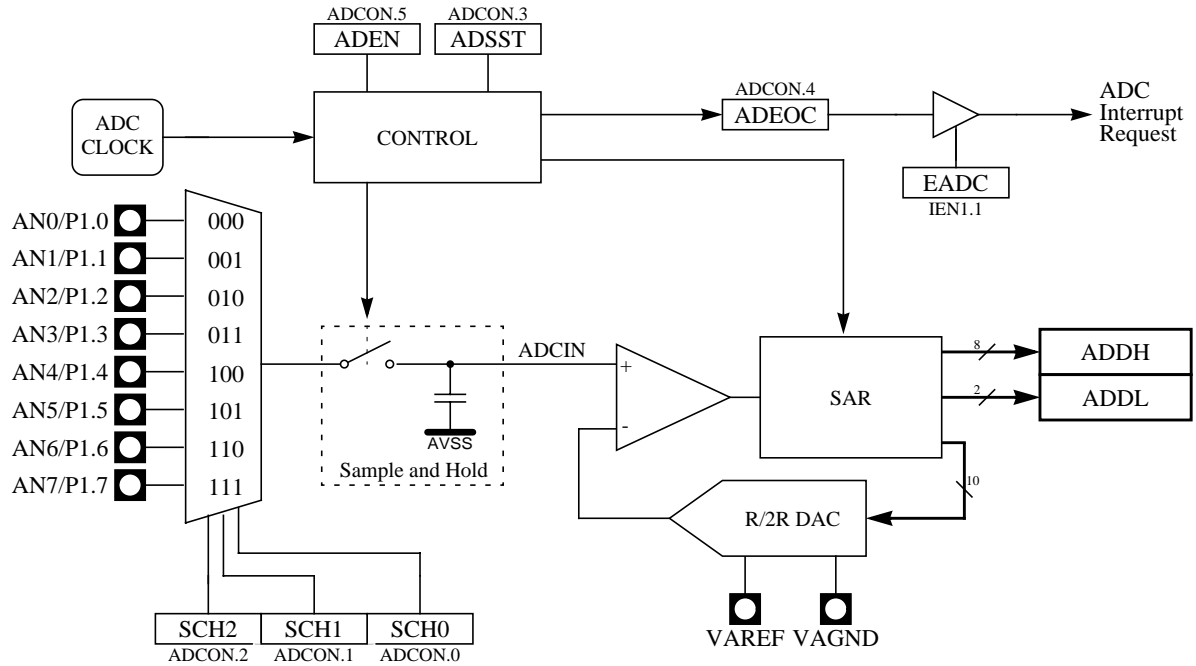


Figure 117. ADC Description

Figure 118 shows the timing diagram of a complete conversion. For simplicity, the figure depicts the waveforms in idealized form and do not provide precise timing information. For ADC characteristics and timing parameters refer to the Section “AC Characteristics” of the T89C51CC02 datasheet.

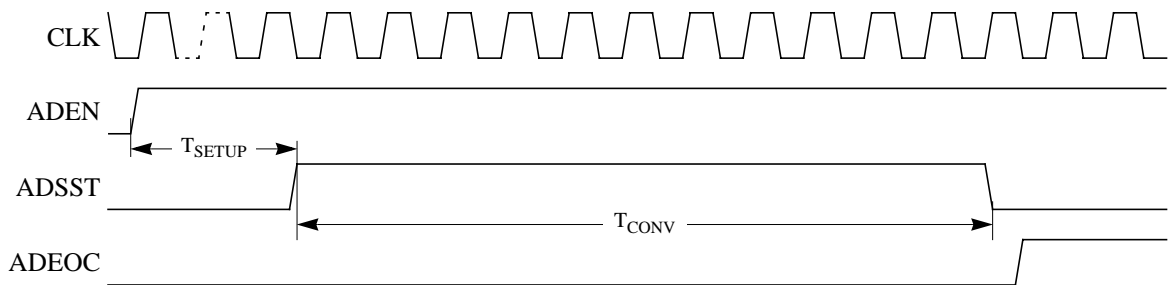


Figure 118. Timing Diagram

NOTE:

$T_{setup} = 4 \mu s$

$T_{conv} = 11 \text{ clock ADC}$

17.4. ADC Converter Operation

A start of single A/D conversion is triggered by setting bit ADSST (ADCON.3).

The busy flag ADSST(ADCON.3) is automatically set when an A/D conversion is running. After completion of the A/D conversion, it is cleared by hardware. This flag can be read only, a write has no effect.

The end-of-conversion flag ADEOC (ADCON.4) is set when the value of conversion is available in ADDH and ADDL, it is cleared by software. If the bit EADC (IEN1.1) is set, an interrupt occur when flag ADEOC is set (see Figure 120). Clear this flag for re-arming the interrupt.

The bits SCH0 to SCH2 in ADCON register are used for the analog input channel selection.

Before Starting Power reduction modes the ADC conversion has to be completed.

Table 22. Selected Analog input

SCH2	SCH1	SCH0	Selected Analog input
0	0	0	AN0
0	0	1	AN1
0	1	0	AN2
0	1	1	AN3
1	0	0	AN4
1	0	1	AN5
1	1	0	AN6
1	1	1	AN7

17.5. Voltage Conversion

When the ADCIN is equals to VAREF the ADC converts the signal to 3FFh (full scale). If the input voltage equals VAGND, the ADC converts it to 000h. Input voltage between VAREF and VAGND are a straight-line linear conversion. All other voltages will result in 3FFh if greater than VAREF and 000h if less than VAGND.

Note that ADCIN should not exceed VAREF absolute maximum range!

17.6. Clock Selection

The maximum clock frequency for ADC is 700KHz. A prescaler is featured (ADCCLK) to generate the ADC clock from the oscillator frequency.

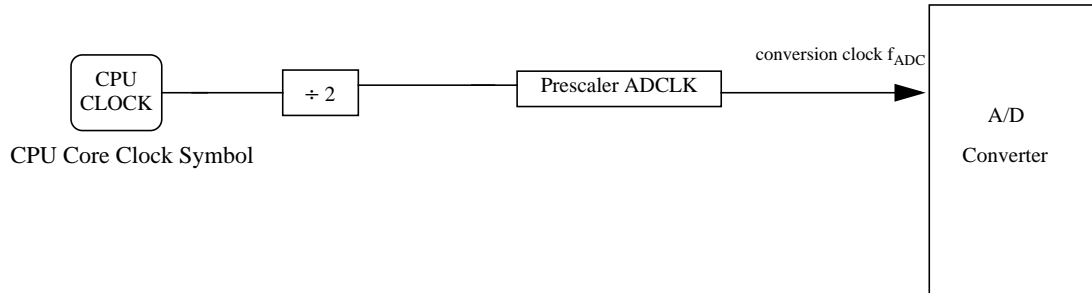


Figure 119. A/D Converter clock

17.7. ADC Standby Mode

When the ADC is not used, it is possible to set it in standby mode by clearing bit ADEN in ADCON register. In this mode the power dissipation is about 1 μ W.

17.8. IT ADC management

An interrupt end-of-conversion will occur when the bit ADEOC is activated and the bit EADC is set. For re-arming the interrupt the bit ADEOC must be cleared by software.

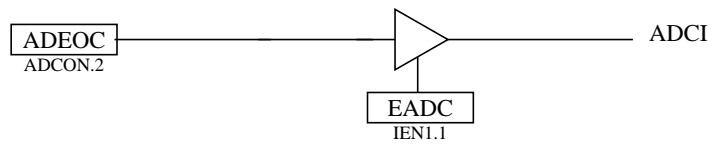


Figure 120. ADC interrupt structure

17.9. Registers

ADCF (S:F6h)

ADC Configuration

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 0:7	Channel Configuration Set to use P1.x as ADC input. Clear tu use P1.x as standart I/O port.

Reset Value=0000 0000b

Figure 121. ADCF Register

ADCON (S:F3h)

ADC Control Register

7	6	5	4	3	2	1	0
-	PSIDLE	ADEN	ADEOC	ADSST	SCH2	SCH1	SCH0

Bit Number	Bit Mnemonic	Description
7	-	
6	PSIDLE	Pseudo Idle mode (best precision) Set to put in idle mode during conversion Clear to converte without idle mode.
5	ADEN	Enable/Standby Mode Set to enable ADC Clear for Standby mode (power dissipation 1 uW).
4	ADEOC	End Of Conversion Set by hardware when ADC result is ready to be read. This flag can generate an interrupt. Must be cleared by software.
3	ADSST	Start and Status Set to start an A/D conversion. Cleared by hardware after completion of the conversion
2-0	SCH2:0	Selection of channel to convert see Table 22

Reset Value=X000 0000b

Figure 122. ADCON Register

ADCLK (S:F2h)

ADC Clock Prescaler

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

Bit Number	Bit Mnemonic	Description
7-5	-	Reserved The value read from these bits are indeterminate. Do not set these bits.
4-0	PRS4:0	Clock Prescaler $f_{ADC} = f_{osc} / (4 \text{ (or 2 in X2 mode)} * PRS)$

Reset Value: XXX0 0000b

Figure 123. ADCLK Register

ADDH (S:F5h Read Only)

ADC Data High byte register

7	6	5	4	3	2	1	0
ADAT 9	ADAT 8	ADAT 7	ADAT 6	ADAT 5	ADAT 4	ADAT 3	ADAT 2

Bit Number	Bit Mnemonic	Description
7-0	ADAT9:2	ADC result bits 9-2

Reset Value: 00h

Figure 124. ADDH Register

ADDL (S:F4h Read Only)

ADC Data Low byte register

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

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

Figure 125. ADDL Register

18. Interrupt System

18.1. Introduction

The CAN Controller has a total of 10 interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), three timer interrupts (timers 0, 1 and 2), a serial port interrupt, a PCA, a CAN interrupt, a timer overrun interrupt and an ADC. These interrupts are shown below.

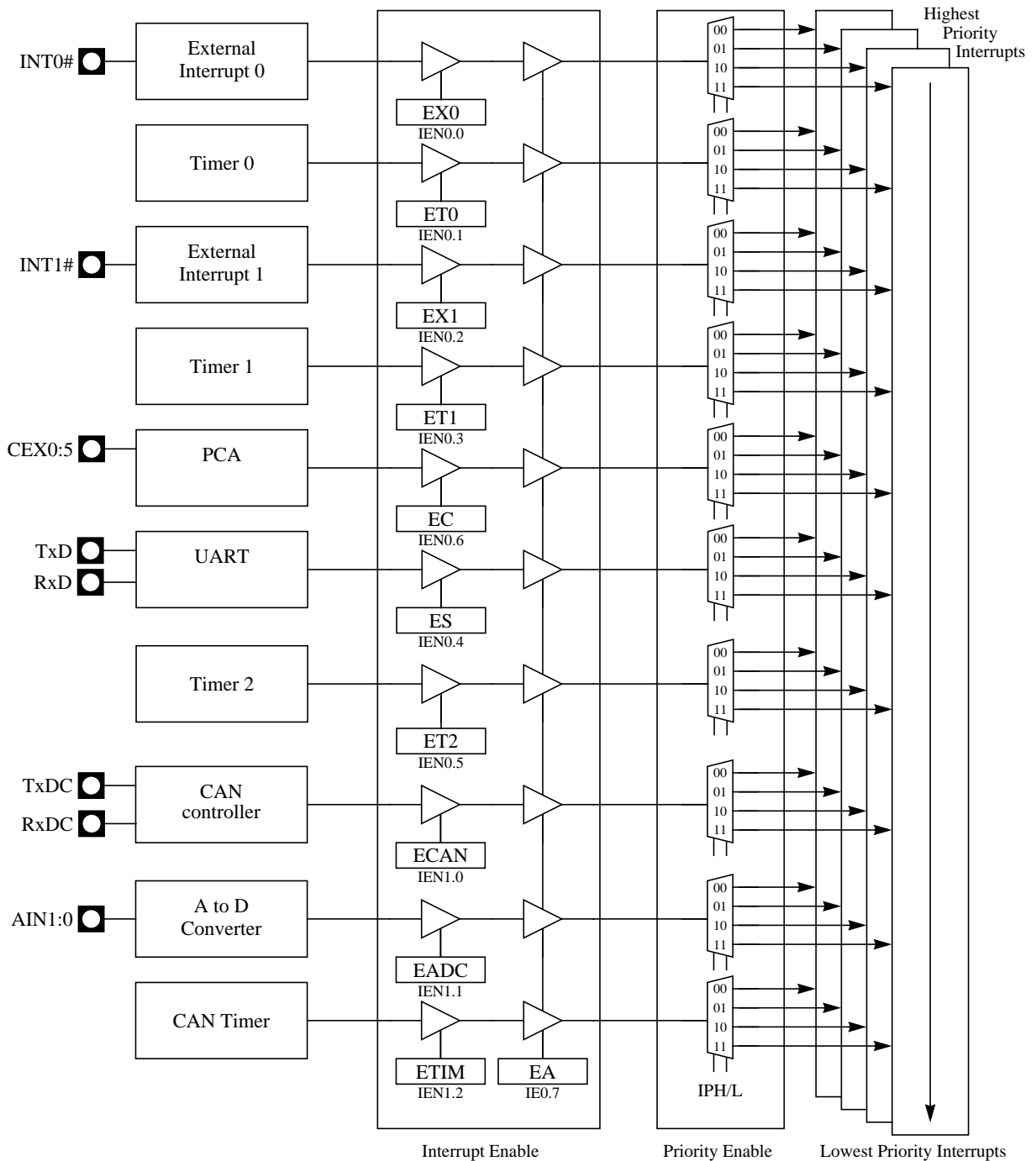


Figure 126. Interrupt Control System

Each of the interrupt sources can be individually enabled or disabled by setting or clearing a bit in the Interrupt Enable register. This register also contains a global disable bit which must be cleared to disable all the interrupts at the same time.

Each interrupt source can also be individually programmed to one of four priority levels by setting or clearing a bit in the Interrupt Priority registers. The Table below shows the bit values and priority levels associated with each combination.

Table 23. Priority Level Bit Values

IPH.x	IPL.x	Interrupt Level Priority
0	0	0 (Lowest)
0	1	1
1	0	2
1	1	3 (Highest)

A low-priority interrupt can be interrupted by a high priority interrupt but not by another low-priority interrupt. A high-priority interrupt cannot be interrupted by any other interrupt source.

If two interrupt requests of different priority levels are received simultaneously, the request of the higher priority level is serviced. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence, see Table 24.

Table 24. Interrupt priority Within level

Interrupt Name	Interrupt Address Vector	Priority Number
external interrupt (INT0)	0003h	1
Timer0 (TF0)	000Bh	2
external interrupt (INT1)	0013h	3
Timer1 (TF1)	001Bh	4
PCA (CF or CCFn)	0033h	5
UART (RI or TI)	0023h	6
Timer2 (TF2)	002Bh	7
CAN (Txok, Rxok, Err or OvrBuf)	003Bh	8
ADC (ADCI)	0043h	9
CAN Timer Overflow (OVRTIM)	004Bh	10

18.2. Registers

IE0 (S:A8h)

Interrupt Enable Register

7	6	5	4	3	2	1	0
EA	EC	ET2	ES	ET1	EX1	ET0	EX0
Bit Number	Bit Mnemonic	Description					
7	EA	Enable All interrupt bit Clear to disable all interrupts. Set to enable all interrupts. If EA=1, each interrupt source is individually enabled or disabled by setting or clearing its interrupt enable bit.					
6	EC	PCA Interrupt Enable Clear to disable the PCA interrupt. Set to enable the PCA interrupt.					
5	ET2	Timer 2 overflow interrupt Enable bit Clear to disable timer 2 overflow interrupt. Set to enable timer 2 overflow interrupt.					
4	ES	Serial port Enable bit Clear to disable serial port interrupt. Set to enable serial port interrupt.					
3	ET1	Timer 1 overflow interrupt Enable bit Clear to disable timer 1 overflow interrupt. Set to enable timer 1 overflow interrupt.					
2	EX1	External interrupt 1 Enable bit Clear to disable external interrupt 1. Set to enable external interrupt 1.					
1	ET0	Timer 0 overflow interrupt Enable bit Clear to disable timer 0 overflow interrupt. Set to enable timer 0 overflow interrupt.					
0	EX0	External interrupt 0 Enable bit Clear to disable external interrupt 0. Set to enable external interrupt 0.					

Reset Value: 0000 0000b
 bit addressable

Figure 127. IE0 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

Figure 128. IE0 Register

IPL0 (S:B8h)
Interrupt Enable Register

7	6	5	4	3	2	1	0
-	PPC	PT2	PS	PT1	PX1	PT0	PX0

Bit Number	Bit Mnemonic	Description
7	-	Reserved The value read from this bit is indeterminate. Do not set this bit.
6	PPC	EWC Counter Interrupt Priority bit Refer to PPCH for priority level.
5	PT2	Timer 2 overflow interrupt Priority bit Refer to PT2H for priority level.
4	PS	Serial port Priority bit Refer to PSH for priority level.
3	PT1	Timer 1 overflow interrupt Priority bit Refer to PT1H for priority level.
2	PX1	External interrupt 1 Priority bit Refer to PX1H for priority level.
1	PT0	Timer 0 overflow interrupt Priority bit Refer to PT0H for priority level.
0	PX0	External interrupt 0 Priority bit Refer to PX0H for priority level.

Reset Value: X000 0000b
bit addressable

Figure 129. IPL0 Register

IPL1 (S:F8h)

Interrupt Priority Low Register 1

7	6	5	4	3	2	1	0
-	-	-	-		POVRL	PADCL	PCANL

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	POVRL	Timer overrun Interrupt Priority level less significant bit. Refer to PI2CH for priority level.
1	PADCL	ADC Interrupt Priority level less significant bit. Refer to PSPIH for priority level.
0	PCANL	CAN Interrupt Priority level less significant bit. Refer to PKBH for priority level.

Reset Value: XXXX X000b

bit addressable

Figure 130. IPL1 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	EWC-PCA Counter Interrupt Priority level most significant bit <table border="0"> <tr> <td><u>PPCH</u></td> <td><u>PPC</u></td> <td><u>Priority level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest priority</td> </tr> </table>	<u>PPCH</u>	<u>PPC</u>	<u>Priority level</u>	0	0	Lowest	0	1		1	0		1	1	Highest priority
<u>PPCH</u>	<u>PPC</u>	<u>Priority level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest priority															
5	PT2H	Timer 2 overflow interrupt High Priority bit <table border="0"> <tr> <td><u>PT2H</u></td> <td><u>PT2</u></td> <td><u>Priority Level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>PT2H</u>	<u>PT2</u>	<u>Priority Level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>PT2H</u>	<u>PT2</u>	<u>Priority Level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															
4	PSH	Serial port High Priority bit <table border="0"> <tr> <td><u>PSH</u></td> <td><u>PS</u></td> <td><u>Priority Level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>PSH</u>	<u>PS</u>	<u>Priority Level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>PSH</u>	<u>PS</u>	<u>Priority Level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															
3	PT1H	Timer 1 overflow interrupt High Priority bit <table border="0"> <tr> <td><u>PT1H</u></td> <td><u>PT1</u></td> <td><u>Priority Level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>PT1H</u>	<u>PT1</u>	<u>Priority Level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>PT1H</u>	<u>PT1</u>	<u>Priority Level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															
2	PX1H	External interrupt 1 High Priority bit <table border="0"> <tr> <td><u>PX1H</u></td> <td><u>PX1</u></td> <td><u>Priority Level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>PX1H</u>	<u>PX1</u>	<u>Priority Level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>PX1H</u>	<u>PX1</u>	<u>Priority Level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															
1	PT0H	Timer 0 overflow interrupt High Priority bit <table border="0"> <tr> <td><u>PT0H</u></td> <td><u>PT0</u></td> <td><u>Priority Level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>PT0H</u>	<u>PT0</u>	<u>Priority Level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>PT0H</u>	<u>PT0</u>	<u>Priority Level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															
0	PX0H	External interrupt 0 high priority bit <table border="0"> <tr> <td><u>PX0H</u></td> <td><u>PX0</u></td> <td><u>Priority Level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>PX0H</u>	<u>PX0</u>	<u>Priority Level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>PX0H</u>	<u>PX0</u>	<u>Priority Level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															

Reset Value: X000 0000b

Figure 131. IPL0 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 <table border="0"> <tr> <td><u>POVRH</u></td> <td><u>POVRL</u></td> <td><u>Priority level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>POVRH</u>	<u>POVRL</u>	<u>Priority level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>POVRH</u>	<u>POVRL</u>	<u>Priority level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															
1	PADCH	ADC Interrupt Priority level most significant bit <table border="0"> <tr> <td><u>PADCH</u></td> <td><u>PADCL</u></td> <td><u>Priority level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>PADCH</u>	<u>PADCL</u>	<u>Priority level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>PADCH</u>	<u>PADCL</u>	<u>Priority level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															
0	PCANH	CAN Interrupt Priority level most significant bit <table border="0"> <tr> <td><u>PCANH</u></td> <td><u>PCANL</u></td> <td><u>Priority level</u></td> </tr> <tr> <td>0</td> <td>0</td> <td>Lowest</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>1</td> <td>Highest</td> </tr> </table>	<u>PCANH</u>	<u>PCANL</u>	<u>Priority level</u>	0	0	Lowest	0	1		1	0		1	1	Highest
<u>PCANH</u>	<u>PCANL</u>	<u>Priority level</u>															
0	0	Lowest															
0	1																
1	0																
1	1	Highest															

Reset Value = XXXX X000b

Figure 132. IPH1 Register

19. Electrical Characteristics

19.1. Absolute Maximum Ratings ⁽¹⁾

Ambiant Temperature Under Bias:

I = industrial -40°C to 85°C

Storage Temperature -65°C to + 150°C

Voltage on V_{CC} to V_{SS} -0.5 V to + 6V

Voltage on Any Pin to V_{SS} -0.5 V to V_{CC} + 0.2 V

Power Dissipation 1 W⁽²⁾

NOTES

1. Stresses at or above those listed under “ Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions may affect device reliability.
2. This value is based on the maximum allowable die temperature and the thermal resistance of the package.

19.2. DC Parameters for Standard Voltage

$T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$; $V_{SS} = 0\text{ V}$; $V_{CC} = 5\text{ V} \pm 10\%$; $F = 0$ to 40 MHz .

Symbol	Parameter	Min	Typ	Max	Unit	Test Conditions
V_{IL}	Input Low Voltage	-0.5		0.7 ⁽⁷⁾	V	
V_{IH}	Input High Voltage except XTAL1, RST	0.2 $V_{CC} + 0.9$		$V_{CC} + 0.5$	V	
V_{IH1}	Input High Voltage, XTAL1, RST	0.7 V_{CC}		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage, ports 1, 2, 3 and 4 ⁽⁶⁾			0.3 0.45 1.0	V V V	$I_{OL} = 100\ \mu\text{A}$ ⁽⁴⁾ $I_{OL} = 1.6\ \text{mA}$ ⁽⁴⁾ $I_{OL} = 3.5\ \text{mA}$ ⁽⁴⁾
V_{OH}	Output High Voltage, ports 1, 2, 3, 4 and 5	$V_{CC} - 0.3$ $V_{CC} - 0.7$ $V_{CC} - 1.5$			V V V	$I_{OH} = -10\ \mu\text{A}$ $I_{OH} = -30\ \mu\text{A}$ $I_{OH} = -60\ \mu\text{A}$ $V_{CC} = 5\text{ V} \pm 10\%$
R_{RST}	RST Pulldown Resistor	20	40 ⁽⁵⁾	200	k Ω	
I_{IL}	Logical 0 Input Current ports 1, 2, 3 and 4			-50	μA	$V_{in} = 0.45\text{ V}$
I_{LI}	Input Leakage Current			± 10	μA	$0.45\text{ V} < V_{in} < V_{CC}$
I_{TL}	Logical 1 to 0 Transition Current, ports 1, 2, 3 and 4			-650	μA	$V_{in} = 2.0\text{ V}$
C_{IO}	Capacitance of I/O Buffer			10	pF	$F_c = 1\text{ MHz}$ $T_A = 25^{\circ}\text{C}$
I_{PD}	Power Down Current		120	350	μA	$4.5\text{ V} < V_{CC} < 5.5\text{ V}$ ⁽³⁾
I_{CC}	Power Supply Current (Typical) $I_{CCOP} = 0.5\text{ Freq (MHz)} + 3\text{ mA}$ $I_{CCIDLE} = 0.3\text{ Freq (MHz)} + 2\text{ mA}$					

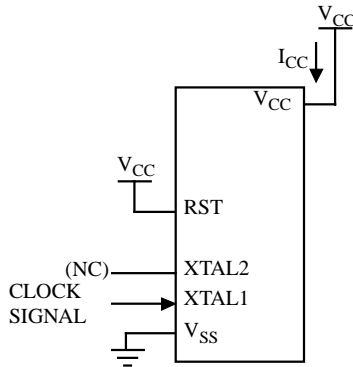
Table 25. DC Parameters in Standard Voltage

NOTES

- Operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with T_{CLCH} , $T_{CHCL} = 5\text{ ns}$ (see Figure 136.). $V_{IL} = V_{SS} + 0.5\text{ V}$, $V_{IH} = V_{CC} - 0.5\text{V}$; XTAL2 N.C.; RST = V_{CC} . I_{CC} would be slightly higher if a crystal oscillator used (see Figure 133.).
- Idle I_{CC} is measured with all output pins disconnected; XTAL1 driven with T_{CLCH} , $T_{CHCL} = 5\text{ ns}$, $V_{IL} = V_{SS} + 0.5\text{ V}$, $V_{IH} = V_{CC} - 0.5\text{ V}$; XTAL2 N.C.; RST = V_{SS} (see Figure 134.).
- Power Down I_{CC} is measured with all output pins disconnected; ; XTAL2 N.C.; RST = V_{SS} (see Figure 135.). In addition, the WDT must be inactive and the POF flag must be set.
- Capacitance loading on Ports 0 and 2 may cause spurious noise pulses to be superimposed on the V_{OLS} of ALE and Ports 1 and 3. The noise is due to external bus capacitance discharging into the Port 0 and Port 2 pins when these pins make 1 to 0 transitions during bus operation. In the worst cases (capacitive loading 100pF), the noise pulse on the ALE line may exceed 0.45V with maxi V_{OL} peak 0.6V. A Schmitt Trigger use is not necessary.
- Typicals are based on a limited number of samples and are not guaranteed. The values listed are at room temperature..
- Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
Maximum I_{OL} per port pin: 10 mA
Maximum I_{OL} per 8-bit port:
Ports 1, 2 and 3: 15 mA
Maximum total I_{OL} for all output pins: 71 mA

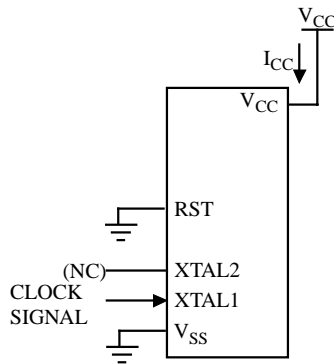
If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.

7. Lower than standart C51 product independant from Vcc supply.



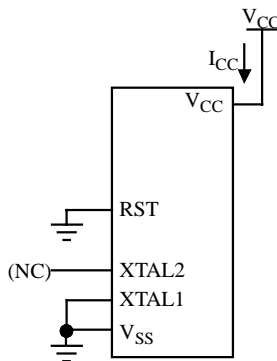
All other pins are disconnected.

Figure 133. I_{CC} Test Condition, Active Mode



All other pins are disconnected.

Figure 134. I_{CC} Test Condition, Idle Mode



All other pins are disconnected.

Figure 135. I_{CC} Test Condition, Power-Down Mode

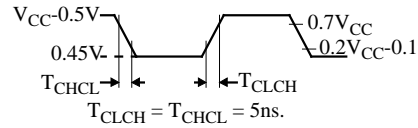


Figure 136. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes

19.3. DC Parameters for A/D Converter

Table 26. DC Parameters for AD Converter

Symbol	Parameter	Min	Typ	Max	Unit	Test Conditions
AVin	Analog input voltage	$V_{ss} - 0.2$		$V_{ref} + 0.2$	V	
Rref	Resistance between Vref and Vss	12	18	24	KOhm	
Vref	Reference voltage	2.40		3.00	V	
Cai	Analog input Capacitance		60		pF	During sampling
INL	Integral non linearity		1	2	lsb	
DNL	Differential non linearity		0.5	1	lsb	
OE	Offset error	-2		2	lsb	

19.4. AC Parameters

19.4.1. Explanation of the AC Symbols

Each timing symbol has 5 characters. The first character is always a “T” (stands for time). The other characters, depending on their positions, stand for the name of a signal or the logical status of that signal. The following is a list of all the characters and what they stand for.

$T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$; $V_{SS} = 0\text{ V}$; $V_{CC} = 5\text{ V} \pm 10\%$; $F = 0$ to 40 MHz .

$T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$; $V_{SS} = 0\text{ V}$; $V_{CC} = 5\text{ V} \pm 10\%$.

(Load Capacitance for all outputs = 60 pF.)

Table 29 give the frequency derating formula of the AC parameter for each speed range description. To calculate each AC symbols. take the x value and use this value in the formula.

19.4.2. Serial Port Timing - Shift Register Mode

Table 27. Symbol Description (F= 40 MHz)

Symbol	Parameter
T_{XLXL}	Serial port clock cycle time
T_{QVHX}	Output data set-up to clock rising edge
T_{XHQX}	Output data hold after clock rising edge
T_{XHDX}	Input data hold after clock rising edge
T_{XHDV}	Clock rising edge to input data valid

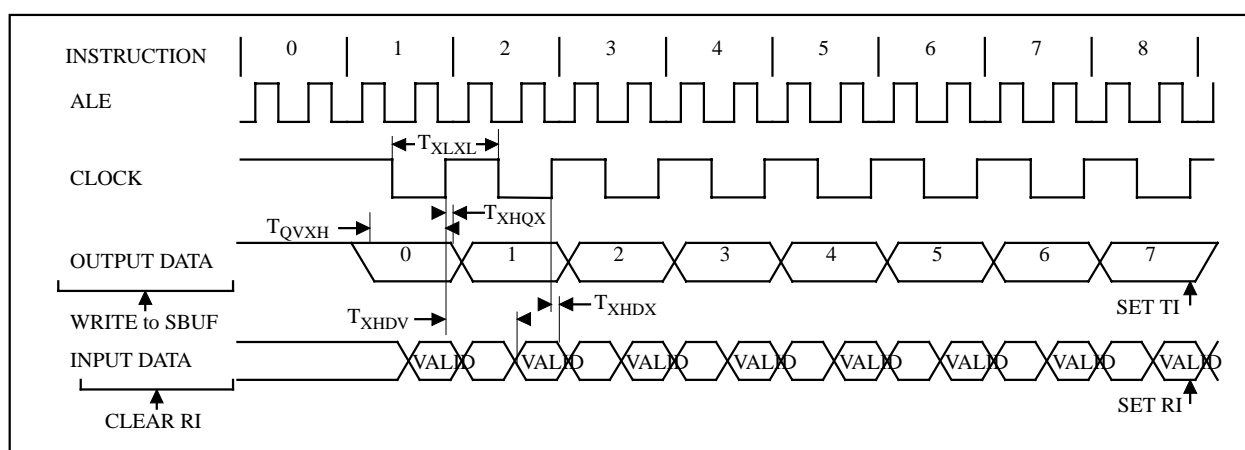
Table 28. AC Parameters for a Fix Clock (F= 40 MHz)

Symbol	Min	Max	Units
T_{XLXL}	300		ns
T_{QVHX}	200		ns
T_{XHQX}	30		ns
T_{XHDX}	0		ns
T_{XHDV}		117	ns

Table 29. AC Parameters for a Variable Clock

Symbol	Type	Standard Clock	X2 Clock	X parameter for -M range	Units
T_{XLXL}	Min	12 T	6 T		ns
T_{QVHX}	Min	10 T - x	5 T - x	50	ns
T_{XHQX}	Min	2 T - x	T - x	20	ns
T_{XHDX}	Min	x	x	0	ns
T_{XHDV}	Max	10 T - x	5 T - x	133	ns

19.4.3. Shift Register Timing Waveforms

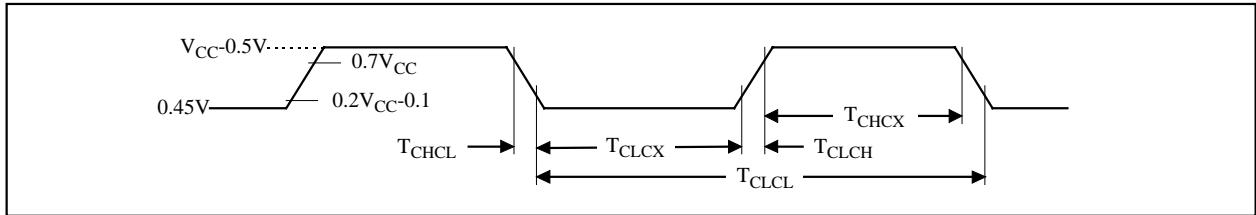


19.4.4. External Clock Drive Characteristics (XTAL1)

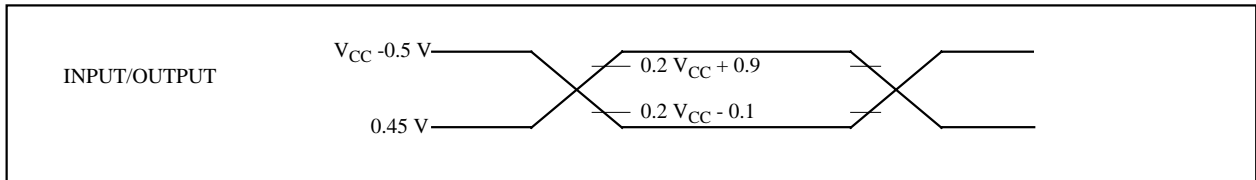
Symbol	Parameter	Min	Max	Units
T_{CLCL}	Oscillator Period	25		ns
T_{CHCX}	High Time	5		ns
T_{CLCX}	Low Time	5		ns
T_{CLCH}	Rise Time		5	ns
T_{CHCL}	Fall Time		5	ns
T_{CHCX}/T_{CLCX}	Cyclic ratio in X2 mode	40	60	%

Table 30. AC Parameters

19.4.5. External Clock Drive Waveforms

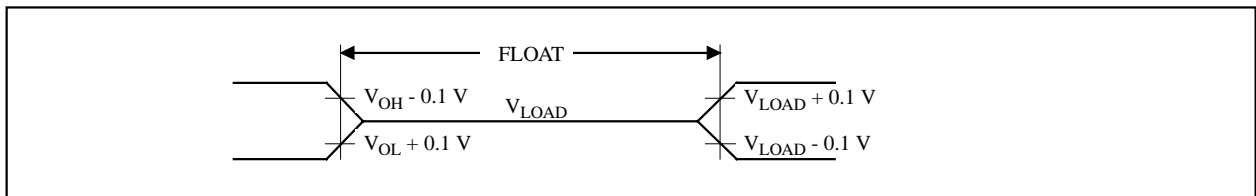


19.4.6. AC Testing Input/Output Waveforms



AC inputs during testing are driven at $V_{CC} - 0.5$ for a logic “1” and 0.45V for a logic “0”. Timing measurement are made at V_{IH} min for a logic “1” and V_{IL} max for a logic “0”.

19.4.7. Float Waveforms



For timing purposes as port pin is no longer floating when a 100 mV change from load voltage occurs and begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs. $I_{OL}/I_{OH} \geq \pm 20\text{mA}$.

20. Ordering Information

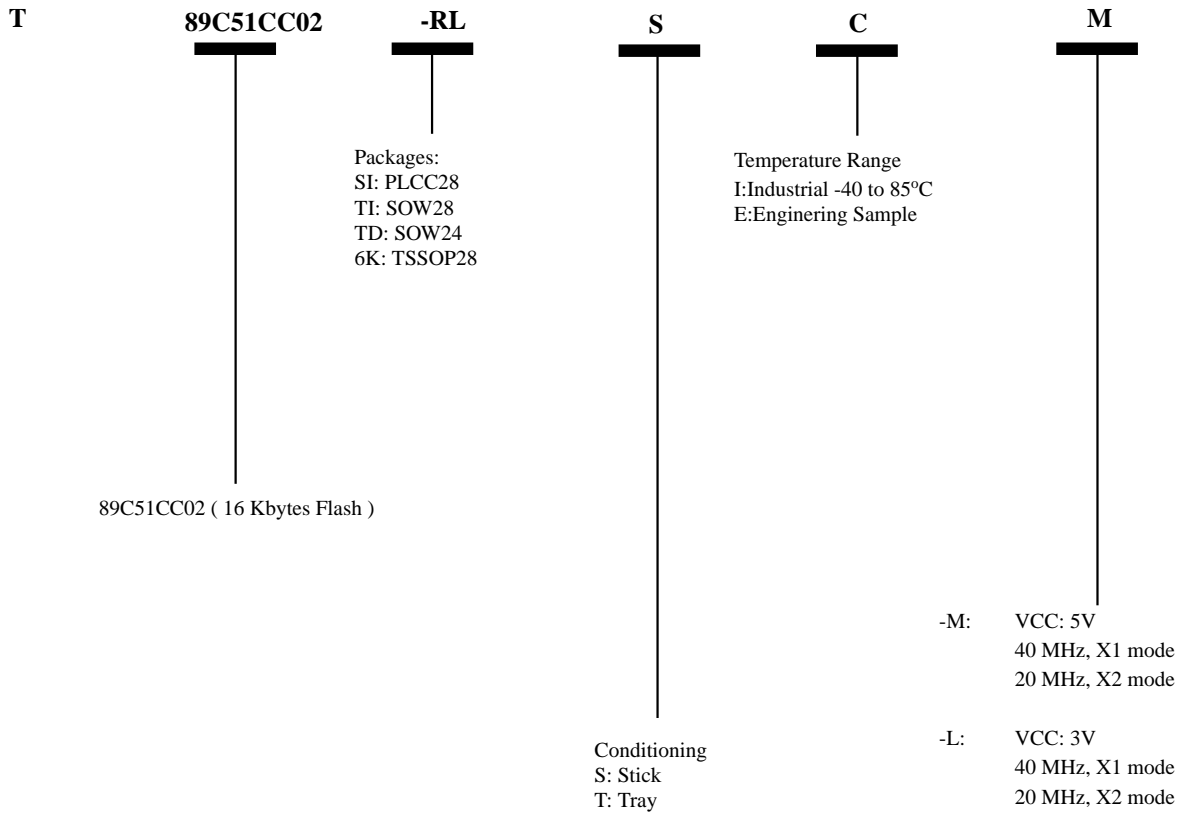


Table 31. Possible order entries

Extension	Type
-SISIM	Stick, PLCC28, Ind, 5V
-TISIM	Stick, SOIC28, Ind, 5V
-TDSIM	Stick, SOIC24, Ind, 5V
-6KSIM	Stick, TSSOP28, Ind, 5V
-SISIL	Stick, PLCC28, Ind, 3V
-TISIL	Stick, SOIC28, Ind, 3V
-TDSIL	Stick, SOIC24, Ind, 3V
-6KSIL	Stick, TSSOP28, Ind, 3V
-SISEM	Stick, PLCC28, Sample
-TISEM	Stick, SOIC28, Sample