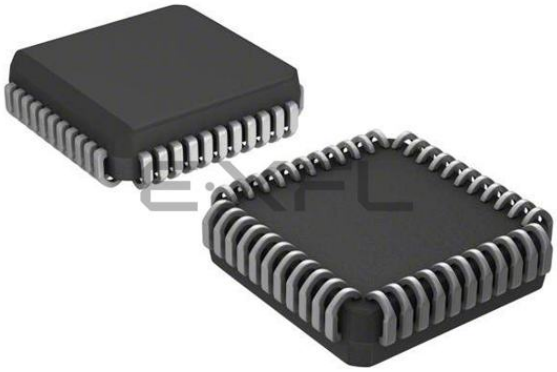


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

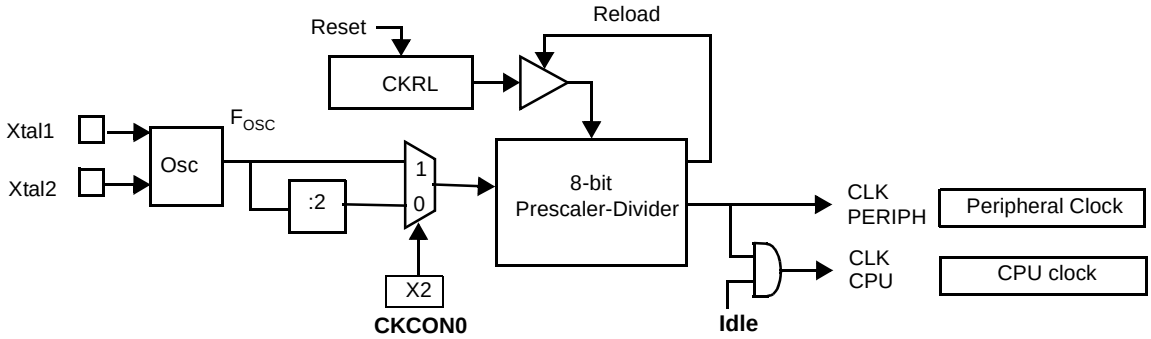
Details	
Product Status	Obsolete
Core Processor	80C51
Core Size	8-Bit
Speed	40MHz
Connectivity	SPI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	32
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.2K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LCC (J-Lead)
Supplier Device Package	44-PLCC (16.6x16.6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/t89c51rc2-slsim

Table 3. Pin Description for 40 - 44 Pin Packages (Continued)

Mnemonic	Pin Number			Type	Name and Function
	DIL	LCC	VQFP44 1.4		
PSEN	29	32	26	O	**Program Strobe ENable:** The read strobe to external program memory. When executing code from the external program memory, $\overline{\text{PSEN}}$ is activated twice each machine cycle, except that two $\overline{\text{PSEN}}$ activations are skipped during each access to external data memory. $\overline{\text{PSEN}}$ is not activated during fetches from internal program memory.
EA	31	35	29	I	**External Access Enable:** $\overline{\text{EA}}$ must be externally held low to enable the device to fetch code from external program memory locations 0000H to FFFFH (RD). If security level 1 is programmed, $\overline{\text{EA}}$ will be internally latched on Reset.

Functional Block Diagram

Figure 3. Functional Oscillator Block Diagram



Prescaler Divider

- A hardware RESET puts the prescaler divider in the following state:
  - CKRL = FFh: $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}/2$ (Standard C51 feature)
- Any value between FFh down to 00h can be written by software into CKRL register in order to divide frequency of the selected oscillator:
  - CKRL = 00h: minimum frequency
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}/1020$ (Standard Mode)
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}/510$ (X2 Mode)
  - CKRL = FFh: maximum frequency
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}/2$ (Standard Mode)
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}$ (X2 Mode)

$F_{CLK\ CPU}$ and $F_{CLK\ PERIPH}$

In X2 Mode:

$$F_{CPU} = F_{CLKPERIPH} = \frac{F_{OSC}}{2 \times (255 - CKRL)}$$

In X1 Mode:

$$F_{CPU} = F_{CLKPERIPH} = \frac{F_{OSC}}{4 \times (255 - CKRL)}$$

Table 7. CKCON1 Register

CKCON1 - Clock Control Register (AFh)

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	SPIX2

Bit Number	Bit Mnemonic	Description
7	-	**Reserved**
6	-	**Reserved**
5	-	**Reserved**
4	-	**Reserved**
3	-	**Reserved**
2	-	**Reserved**
1	-	**Reserved**
0	SPIX2	**SPI** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect). Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.

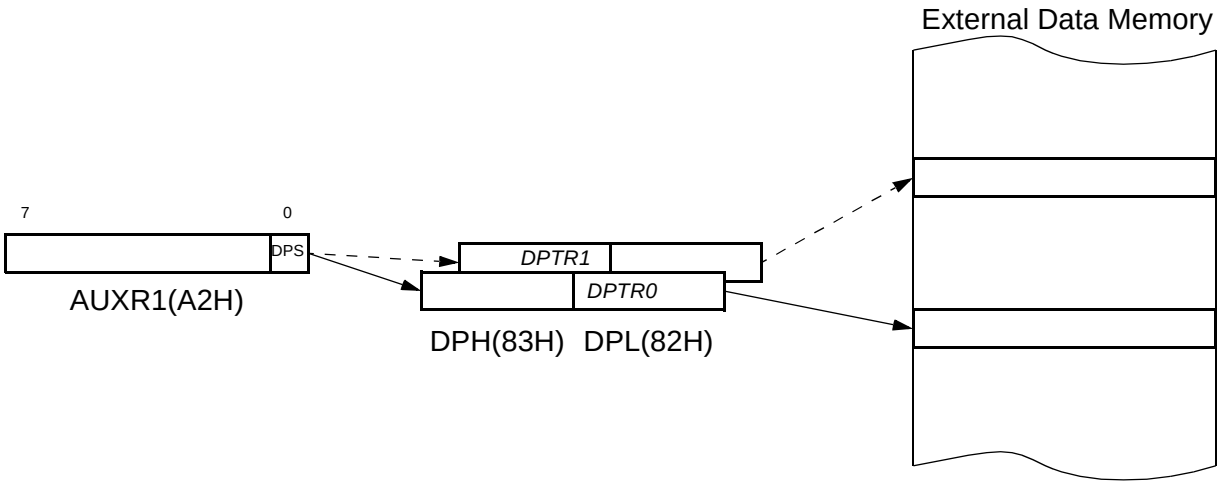
Reset Value = XXXX XXX0b
Not bit addressable

Dual Data Pointer Register DPTR

The additional data pointer can be used to speed up code execution and reduce code size.

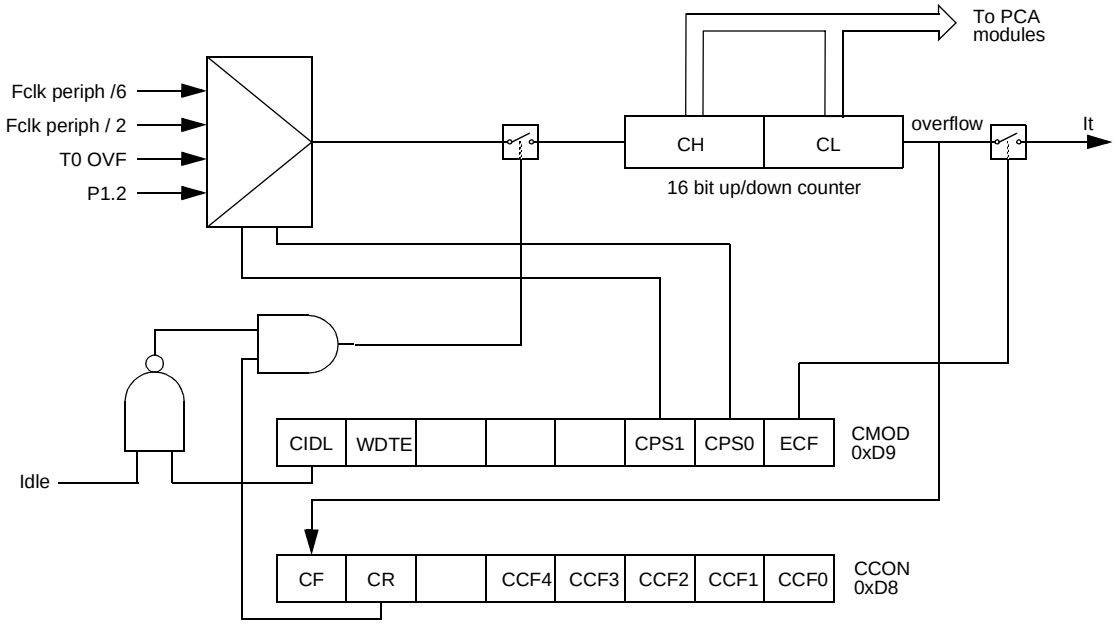
The dual DPTR structure is a way by which the chip will specify the address of an external data memory location. There are two 16-bit DPTR registers that address the external memory, and a single bit called DPS = AUXR1.0 (see Table 8) that allows the program code to switch between them (Refer to Figure 6).

Figure 6. Use of Dual Pointer



INC is a short (2 bytes) and fast (12 clocks) way to manipulate the DPS bit in the AUXR1 SFR. 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. Observe that without the last instruction (INC AUXR1), the routine will exit with DPS in the opposite state.

Figure 10. PCA Timer/Counter



Registers

Table 13. CMOD Register

CMOD - PCA Counter Mode Register (D9h)

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

Bit Number	Bit Mnemonic	Description
7	CIDL	**Counter Idle Control** Cleared to program the PCA Counter to continue functioning during idle Mode. Set to program PCA to be gated off during idle.
6	WDTE	**Watchdog Timer Enable** Cleared to disable Watchdog Timer function on PCA Module 4. Set to enable Watchdog Timer function on PCA Module 4.
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	**PCA Count Pulse Select** CPS1 CPS0 Selected PCA input 0 0 Internal clock fCLK PERIPH/6 0 1 Internal clock fCLK PERIPH/2 1 0 Timer 0 Overflow 1 1 External clock at ECI/P1.2 pin (max rate = fCLK PERIPH/ 4)
1	CPS0	
0	ECF	**PCA Enable Counter Overflow Interrupt** Cleared to disable CF bit in CCON to inhibit an interrupt. Set to enable CF bit in CCON to generate an interrupt.

Reset Value = 00XX X000b
Not bit addressable

The CMOD register includes three additional bits associated with the PCA (See Figure 10 and Table 13).

- The CIDL bit which allows the PCA to stop during idle mode.
- The WDTE bit which enables or disables the watchdog function on module 4.
- The ECF bit which when set causes an interrupt and the PCA overflow flag CF (in the CCON SFR) to be set when the PCA timer overflows.

The CCON register contains the run control bit for the PCA and the flags for the PCA timer (CF) and each module (Refer to Table 14).

- Bit CR (CCON. 6) must be set by software to run the PCA. The PCA is shut off by clearing this bit.
- Bit CF: The CF bit (CCON. 7) is set when the PCA counter overflows and an interrupt will be generated if the ECF bit in the CMOD register is set. The CF bit can only be cleared by software.
- Bits 0 through 4 are the flags for the modules (bit 0 for module 0, bit 1 for module 1, etc.) and are set by hardware when either a match or a capture occurs. These flags also can only be cleared by software.

Table 14. CCON Register

CCON - PCA Counter Control Register (D8h)

7	6	5	4	3	2	1	0
CF	CR	-	CCF4	CCF3	CCF2	CCF1	CCF0

Bit Number	Bit Mnemonic	Description
7	CF	**PCA Counter Overflow flag** Set by hardware when the counter rolls over. CF flags an interrupt if bit ECF in CMOD is set. CF may be set by either hardware or software but can only be cleared by software.
6	CR	**PCA Counter Run control bit** Must be cleared by software to turn the PCA counter off. Set by software to turn the PCA counter on.
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	CCF4	**PCA Module 4 interrupt flag** Must be cleared by software. Set by hardware when a match or capture occurs.
3	CCF3	**PCA Module 3 interrupt flag** Must be cleared by software. Set by hardware when a match or capture occurs.
2	CCF2	**PCA Module 2 interrupt flag** Must be cleared by software. Set by hardware when a match or capture occurs.
1	CCF1	**PCA Module 1 interrupt flag** Must be cleared by software. Set by hardware when a match or capture occurs.
0	CCF0	**PCA Module 0 interrupt flag** Must be cleared by software. Set by hardware when a match or capture occurs.

Reset Value = 000X 0000b
Not bit addressable

The watchdog timer function is implemented in module 4 (See Figure 13).

The PCA interrupt system is shown in Figure 11.

cause an internal reset. The second option is also not recommended if other PCA modules are being used. Remember, the PCA timer is the time base for all modules; changing the time base for other modules would not be a good idea. Thus, in most applications the first solution is the best option.

This watchdog timer won't generate a reset out on the reset pin.

Table 31. T2CON Register

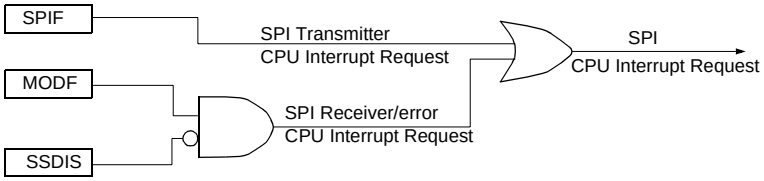
T2CON - Timer 2 Control Register (C8h)

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** Must be cleared by software. Set by hardware on timer 2 overflow, if RCLK = 0 and TCLK = 0.
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. When set, causes the CPU to vector to timer 2 interrupt routine when timer 2 interrupt is enabled. Must be cleared by software. EXF2 doesn't cause an interrupt in Up/down counter mode (DCEN = 1)
5	RCLK	**Receive Clock bit for UART** Cleared 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 for UART** Cleared 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** Cleared 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** Cleared to turn off timer 2. Set to turn on timer 2.
1	C/T2#	**Timer/Counter 2 select bit** Cleared for timer operation (input from internal clock system: $F_{CLK\ PERIPH}$). Set for counter operation (input from T2 input pin, falling edge trigger). Must be 0 for clock out mode.
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. Cleared 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 30. SPI Interrupt Requests Generation



Registers

There are three registers in the module that provide control, status and data storage functions. These registers are describes in the following paragraphs.

Serial Peripheral Control register (SPCON)

- The Serial Peripheral Control Register does the following:
- Selects one of the Master clock rates,
- Configure the SPI module as Master or Slave,
- Selects serial clock polarity and phase,
- Enables the SPI module,
- Frees the SS pin for a general purpose

Table 47 describes this register and explains the use of each bit.

Table 47. SPCON Register

SPCON - Serial Peripheral Control Register (0C3H)

7	6	5	4	3	2	1	0
SPR2	**SPEN**	**SSDIS**	**MSTR**	**CPOL**	**CPHA**	**SPR1**	**SPR0**

Bit Number	Bit Mnemonic	Description
7	SPR2	**Serial Peripheral Rate 2** Bit with SPR1 and SPR0 define the clock rate.
6	SPEN	**Serial Peripheral Enable** Cleared to disable the SPI interface. Set to enable the SPI interface.
5	SSDIS	**$\overline{SS}$ Disable** Cleared to enable SS# in both Master and Slave modes. Set to disable SS# in both Master and Slave modes. In Slave mode, this bit has no effect if CPHA ='0'.
5	MSTR	**Serial Peripheral Master** Cleared to configure the SPI as a Slave. Set to configure the SPI as a Master.
4	CPOL	**Clock Polarity** Cleared to have the SCK set to'0' in idle state. Set to have the SCK set to'1' in idle low.
3	CPHA	**Clock Phase** Cleared to have the data sampled when the SCK leaves the idle state (see CPOL). Set to have the data sampled when the SCK returns to idle state (see CPOL).

Bit Number	Bit Mnemonic	Description			
2	SPR1	**SPR2**	**SPR1**	**SPR0**	**Serial Peripheral Rate**
		0	0	0	$F_{CLK\ PERIPH}$/2
		0	0	1	$F_{CLK\ PERIPH}$/4
		0	1	0	$F_{CLK\ PERIPH}$/8
1	SPR0	0	1	1	$F_{CLK\ PERIPH}$/16
		1	0	0	$F_{CLK\ PERIPH}$/32
		1	0	1	$F_{CLK\ PERIPH}$/64
		1	1	0	$F_{CLK\ PERIPH}$/128
		1	1	1	Invalid

Reset Value= 0001 0100b

Not bit addressable

Serial Peripheral Status Register (SPSTA)

The Serial Peripheral Status Register contains flags to signal the following conditions:

- Data transfer complete
- Write collision
- Inconsistent logic level on $\overline{SS}$ pin (mode fault error)

Table 48 describes the SPSTA register and explains the use of every bit in the register.

Table 48. SPSTA Register

SPSTA - Serial Peripheral Status and Control register (0C4H)

7	6	5	4	3	2	1	0
SPIF	**WCOL**	**-**	**MODF**	**-**	**-**	**-**	**-**

Bit Number	Bit Mnemonic	Description
7	SPIF	**Serial Peripheral data transfer flag** Cleared by hardware to indicate data transfer is in progress or has been approved by a clearing sequence. Set by hardware to indicate that the data transfer has been completed.
6	WCOL	**Write Collision flag** Cleared by hardware to indicate that no collision has occurred or has been approved by a clearing sequence. Set by hardware to indicate that a collision has been detected.
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	MODF	**Mode Fault** Cleared by hardware to indicate that the $\overline{SS}$ pin is at appropriate logic level, or has been approved by a clearing sequence. Set by hardware to indicate that the $\overline{SS}$ pin is at inappropriate logic level.
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

Bit Number	Bit Mnemonic	Description
1	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
0	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.

Reset Value= 00X0 XXXXb

Not Bit addressable

Serial Peripheral DATa register (SPDAT)

The Serial Peripheral Data Register (Table 49) is a read/write buffer for the receive data register. A write to SPDAT places data directly into the shift register. No transmit buffer is available in this model.

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

Table 49. SPDAT Register

SPDAT - Serial Peripheral Data Register (0C5H)

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

Reset Value= Indeterminate

R7:R0: Receive data bits

SPCON, SPSTA and SPDAT registers may be read and written at any time while there is no on-going exchange. However, special care should be taken when writing to them while a transmission is on-going:

- Do not change SPR2, SPR1 and SPR0
- Do not change CPHA and CPOL
- Do not change MSTR
- Clearing SPEN would immediately disable the peripheral
- Writing to the SPDAT will cause an overflow

Hardware Watchdog Timer

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upset. The WDT consists of a 14-bit counter and the WatchDog Timer ReSeT (WDTRST) SFR. The WDT is by default disabled from exiting reset. To enable the WDT, user must write 01EH and 0E1H in sequence to the WDTRST, SFR location 0A6H. When WDT 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 drive an output RESET HIGH pulse at the RST-pin.

Using the WDT

To enable the WDT, user must write 01EH and 0E1H in sequence to the WDTRST, SFR location 0A6H. When WDT is enabled, the user needs to service it by writing to 01EH and 0E1H to WDTRST to avoid WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH) and this will reset the device. When WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least every 16383 machine cycle. To reset the WDT the user must write 01EH and 0E1H to WDTRST. WDTRST is a write only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST-pin. The RESET pulse duration is 96 x $T_{CLK\ PERIPH}$, where $T_{CLK\ PERIPH}= 1/F_{CLK\ PERIPH}$. 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.

To have a more powerful WDT, a 2^7 counter has been added to extend the Time-out capability, ranking from 16ms to 2s @ F_{OSCA} = 12MHz. To manage this feature, refer to WDTPRG register description, Table 50.

Table 50. WDTRST Register

WDTRST - Watchdog Reset Register (0A6h)

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

Reset Value = XXXX XXXXb

Write only, this SFR is used to reset/enable the WDT by writing 01EH then 0E1H in sequence.

Idle Mode

An instruction that sets PCON. 0 indicates that it is the last instruction to be executed before going into Idle mode. In Idle mode, the internal clock signal is gated off to the CPU, but not to the interrupt, Timer, and Serial Port functions. The CPU status is preserved in its entirety: the Stack Pointer, Program Counter, Program Status Word, Accumulator and all other registers maintain their data during idle. The port pins hold the logical states they had at the time Idle was activated. ALE and PSEN hold at logic high level.

There are two ways to terminate the Idle mode. Activation of any enabled interrupt will cause PCON. 0 to be cleared by hardware, terminating the Idle mode. The interrupt will be serviced, and following RETI the next instruction to be executed will be the one following the instruction that put the device into idle.

The flag bits GF0 and GF1 can be used to give an indication if an interrupt occurred during normal operation or during idle. For example, an instruction that activates idle can also set one or both flag bits. When idle is terminated by an interrupt, the interrupt service routine can examine the flag bits.

The other way of terminating the Idle mode is with a hardware reset. Since the clock oscillator is still running, the hardware reset needs to be held active for only two machine cycles (24 oscillator periods) to complete the reset.

Power-down Mode

To save maximum power, a power-down mode can be invoked by software (refer to Table 5, PCON register).

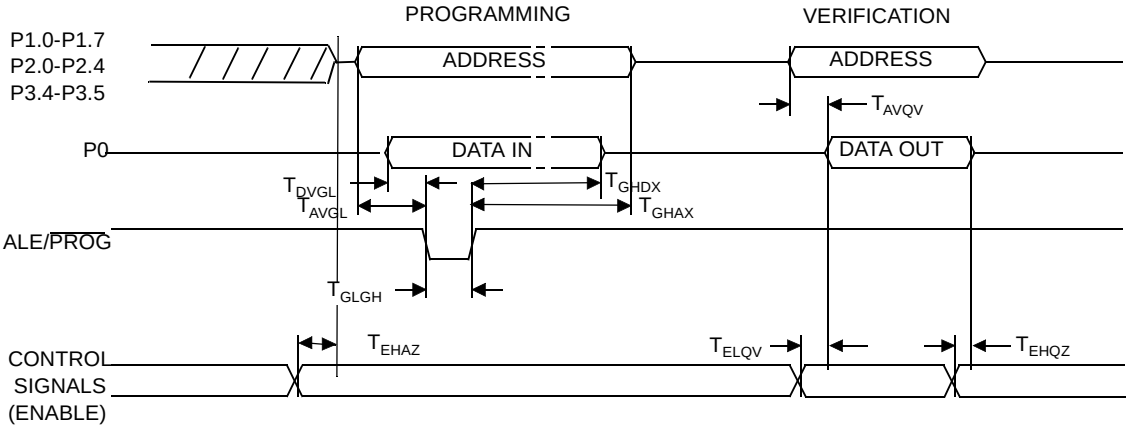
In power-down mode, the oscillator is stopped and the instruction that invoked power-down mode is the last instruction executed. The internal RAM and SFRs retain their value until the power-down mode is terminated. V_{CC} can be lowered to save further power. Either a hardware reset or an external interrupt can cause an exit from power-down. To properly terminate power-down, the reset or external interrupt should not be executed before V_{CC} is restored to its normal operating level and must be held active long enough for the oscillator to restart and stabilize.

Only external interrupts $\overline{\text{INT0}}$, $\overline{\text{INT1}}$ and Keyboard Interrupts are useful to exit from power-down. For that, interrupt must be enabled and configured as level or edge sensitive interrupt input. When Keyboard Interrupt occurs after a power down mode, 1024 clocks are necessary to exit to power down mode and enter in operating mode.

Holding the pin low restarts the oscillator but bringing the pin high completes the exit as detailed in Figure 32. When both interrupts are enabled, the oscillator restarts as soon as one of the two inputs is held low and power down exit will be completed when the first input will be released. In this case the higher priority interrupt service routine is executed. Once the interrupt is serviced, the next instruction to be executed after RETI will be the one following the instruction that put T89C51RB2/RC2 into power-down mode.

Flash EEPROM Programming and Verification Waveforms

Figure 40. Flash EEPROM Programming and Verification Waveforms



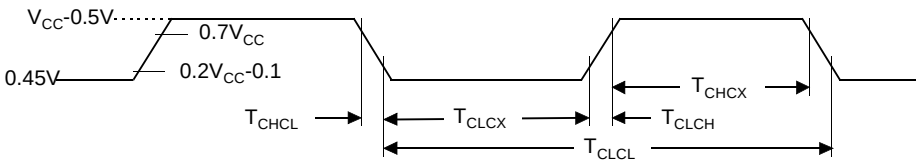
External Clock Drive Characteristics (XTAL1)

Table 68. External Clock Drive Characteristics (XTAL1)

Symbol	Parameter	Min	Max	Units
T_{CLCL}	Oscillator Period	25		ns
T_{CHCX}	High Time	3		ns
T_{CLCX}	Low Time	3		ns
T_{CLCH}	Rise Time		3	ns
T_{CHCL}	Fall Time		3	ns
T_{CHCX}/T_{CLCX}	Cyclic ratio in X2 mode	40	60	%

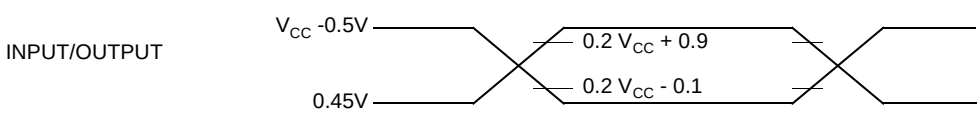
External Clock Drive Waveforms

Figure 41. External Clock Drive Waveforms



AC Testing Input/Output Waveforms

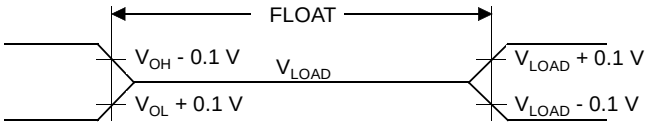
Figure 42. AC Testing Input/Output Waveforms



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”.

Float Waveforms

Figure 43. 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. $I_{OL}/I_{OH} \geq \pm 20$ mA.

Clock Waveforms

Valid in normal clock mode. In X2 mode XTAL2 must be changed to XTAL2/2.

Table 75. API Calls (Continued)

API Call	Parameter
PROGRAM SOFTWARE SECURITY BIT	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 05h DPH = 00h DPL = 00h – security bit # 1 (inhibit writing to Flash) 01h – security bit # 2 (inhibit Flash verify) 10h - allows ISP writing to Flash (see Note 1) 11h - allows ISP Flash verify (see Note 1) Return Parameter none
PROGRAM BOOT STATUS BYTE	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 06h DPH = 00h DPL = 00h ACC = status byte Return Parameter ACC = status byte
PROGRAM BOOT VECTOR	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 06h DPH = 00h DPL = 01h ACC = boot vector Return Parameter ACC = boot vector
PROGRAM X2 MODE	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 0Ah DPH = 00h DPL = 08h ACC = value (00 or 01h) Return Parameter ACC = boot vector
PROGRAM BLJB	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 0Ah DPH = 00h DPL = 04h ACC = value (00 or 01h) Return Parameter ACC = boot vector
READ DEVICE DATA	Input Parameters: R1 = 03h DPTR = address of byte to read Return Parameter ACC = value of byte read

Table 75. API Calls (Continued)

API Call	Parameter
READ copy of the MANUFACTURER ID	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 00h DPH = 00h DPL = 00h (manufacturer ID) Return Parameter ACC = value of byte read
READ copy of the device ID # 1	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 00h DPH = 00h DPL = 01h (device ID # 1) Return Parameter ACC = value of byte read
READ copy of the device ID # 2	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 00h DPH = 00h DPL = 02h (device ID # 2) Return Parameter ACC = value of byte read
READ copy of the device ID # 3	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 00h DPH = 00h DPL = 03h (device ID # 2) Return Parameter ACC = value of byte read
READ SOFTWARE SECURITY BITS	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 07h DPH = 00h DPL = 00h (Software security bits) Return Parameter ACC = value of byte read
READ HARDWARE SECURITY BITS	Input Parameters: R0 = osc freq (integer Not required, left for Philips compatibility) R1 = 07h -> OBh DPH = 00h DPL = 04h (Hardware security bits) Return Parameter ACC = value of byte read

Table of Contents