What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	8051
Core Size	8-Bit
Speed	24MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, KPI, LCD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.4V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-TSSOP (0.173", 4.40mm Width)
Supplier Device Package	20-TSSOP
Purchase URL	https://www.e-xfl.com/product-detail/nuvoton-technology-corporation-america/n79e715at20

6.8 Bit-addressable Locations

The Scratch-pad RAM area from location 20h to 2Fh is byte as well as bit-addressable. This means that a bit in this area can be individually addressed. In addition, some of the SFRs are also bit-addressable. The instruction decoder is able to distinguish a bit access from a byte access by the type of the instruction itself. In the SFR area, any existing SFR whose address ends in a 0 or 8 is bit-addressable.

6.9 Stack

The scratch-pad RAM can be used for the stack. This area is selected by the Stack Pointer (SP), which stores the address of the top of the stack. Whenever a jump, call or interrupt is invoked, the return address is placed on the stack. There is no restriction as to where the stack can begin in the RAM. By default, however, the Stack Pointer contains 07h at reset. The user can then change this to any value desired. The SP will point to the last used value. Therefore, the SP will be incremented and then address saved onto the stack. Conversely, while popping from the stack the contents will be read first, and then the SP is decreased.

P1 – Port 1 (Bit-addressable)

7	6	5	4	3	2	1	0
P17	P16	-	P14	P13	P12	P11	P10
R/W	R/W	-	R/W	R/W	R/W	R/W	R/W

Address: 90H Reset value: 1111 1111B

Bit	Name	Description
7:0	P1[7:0]	**Port 1** These pins are in quasi-bidirectional mode except P1.2 and P1.3 pins. The P1.2 and P1.3 are dedicating open-drain pins for I^2C interface after reset.

P2 – Port 2 (Bit-addressable)

7	6	5	4	3	2	1	0
P27	P26	P25	P24	P23	P22	P21	P20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: A0H Reset value: 1111 1111B

Bit	Name	Description
7:0	P2[7:0]	**Port 2** Port 2 is an 8-bit quasi bidirectional I/O port.

P3 – Port 3 (Bit-addressable)

7	6	5	4	3	2	1	0
-	-	-	-	-	-	P31	P30
-	-	-	-	-	-	R/W	R/W

Address: B0H Reset value: 0000 0011B

Bit	Name	Description
7:2	-	**Reserved**
1	P3.1	X1 or I/O pin by alternative.
0	P3.0	X2 or CLKOUT or I/O pin by alternative.

Bit	Name	Description
4	P0S	Enable Schmitt trigger inputs on Port 0.
1	P3M1.1	Control the output configuration of P3.1.
0	P3M1.0	Control the output configuration of P3.0.

P3M2 – Port3 Output Mode2

7	6	5	4	3	2	1	0
-	-	-	-	-	ENCLK	P3M2.1	P3M2.0
-	-	-	-	-	R/W	R/W	R/W

Address: 97H Reset value: 0000 0000B

Bit	Name	Description
7:3	-	**Reserved**
0	ENCLK	Clock Output to XTAL2 Pin (P3.0) Enable If the clock is from HIRC, the frequency of P3.0 is $F_{HIRC}/4$.
1	P3M2.1	Refer to Table 9-1 Setting Table for I/O Port Structure
0	P3M2.0	

CKCON – Clock Control

7	6	5	4	3	2	1	0
-	-	-	T1M	T0M	-	-	-
-	-	-	R/W	R/W	-	-	-

Address: 8EH Reset value: 0000 0000B

Bit	Name	Description
7:5	-	**Reserved**
4	T1M	**Timer 1 Clock Selection** 0 = Timer 1 uses a divide by 12 clocks. 1 = Timer 1 uses a divide by 4 clocks.
3	T0M	**Timer 0 Clock Selection** 0 =Timer 0 uses a divide by 12 clocks. 1 = Timer 0 uses a divide by 4 clocks.
2:0	-	**Reserved**

TMOD – Timer 0 and 1 Mode

7	6	5	4	3	2	1	0
GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: 89H Reset value: 0000 0000B

Bit	Name	Description
7	GATE	**Timer 1 Gate Control** 0 = Timer 1 will clock when TR1 = 1 regardless of $\overline{\text{INT1}}$ logic level. 1 = Timer 1 will clock only when TR1 = 1 and $\overline{\text{INT1}}$ is logic 1.
6	C/$\overline{T}$	**Timer 1 Counter/Timer Selection** 0 = Timer 1 is incremented by internal peripheral clocks. 1 = Timer 1 is incremented by the falling edge of the external pin T1.
5	M1	**Timer 1 Mode Selection**
4	M0	

M1	M0	Timer 1 Mode
0	0	Mode 0: 8-bit Timer/Counter with 5-bit pre-scalar (TL1[4:0])
0	1	Mode 1: 16-bit Timer/Counter
1	0	Mode 2: 8-bit Timer/Counter with auto-reload from TH1
1	1	Mode 3: Timer 1 halted

Bit	Name	Description
3	GATE	**Timer 0 Gate Control** 0 = Timer 0 will clock when TR0 = 1 regardless of $\overline{\text{INT0}}$ logic level. 1 = Timer 0 will clock only when TR0 = 0 and $\overline{\text{INT0}}$ is logic 1.

Bit	Name	Description
2	$C/\overline{T}$	**Timer 0 Counter/Timer Selection** 0 = Timer 0 is incremented by internal peripheral clocks. 1 = Timer 0 is incremented by the falling edge of the external pin T0.
1	M1	**Timer 0 Mode Selection**
0	M0	See mode table below.

M1	M0	Timer 0 Mode
0	0	Mode 0: 8-bit Timer/Counter with 5-bit pre-scalar (TL0[4:0])
0	1	Mode 1: 16-bit Timer/Counter
1	0	Mode 2: 8-bit Timer/Counter with auto-reload from TH0
1	1	Mode 3: TL0 as a 8-bit Timer/Counter and TH0 as a 8-bit Timer

TCON – Timer 0 and 1 Control (Bit-addressable)

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

Address: 88H Reset value: 0000 0000B

Bit	Name	Description
7	TF1	**Timer 1 Overflow Flag** This bit is set when Timer 1 overflows. It is automatically cleared by hardware when the program executes the Timer 1 interrupt service routine. Software can also set or clear this bit.
6	TR1	**Timer 1 Run Control** 0 = Timer 1 is halted. Clearing this bit will halt Timer 1 and the current count will be preserved in TH1 and TL1. 1 = Timer 1 is enabled.
5	TF0	**Timer 0 Overflow Flag** This bit is set when Timer 0 overflows. It is automatically cleared via hardware when the program executes the Timer 0 interrupt service routine. Software can also set or clear this bit.
4	TR0	**Timer 0 Run Control** 0 = Timer 0 is halted. Clearing this bit will halt Timer 0 and the current count will be preserved in TH0 and TL0. 1 = Timer 0 is enabled.

10.1.2 Mode 1 (16-bit Timer)

Mode 1 is similar to Mode 0 except that the counting registers are fully used as a 16-bit counter. Rollover occurs when a count moves FFFFH to 0000H. The Timer overflow flag TFx of the relevant Timer/Counter is set and an interrupt will occurs if enabled.

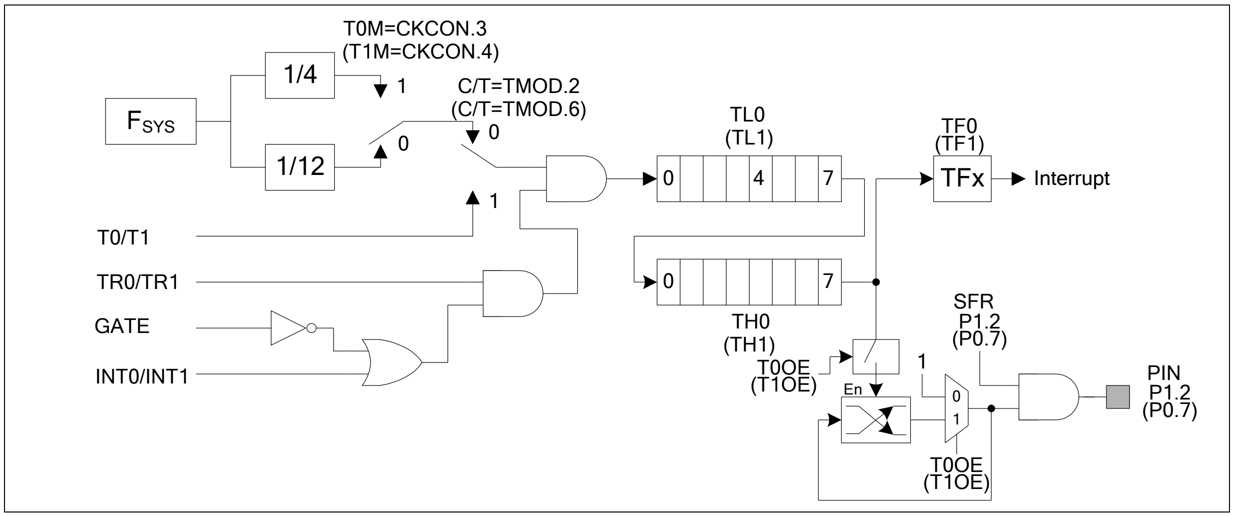
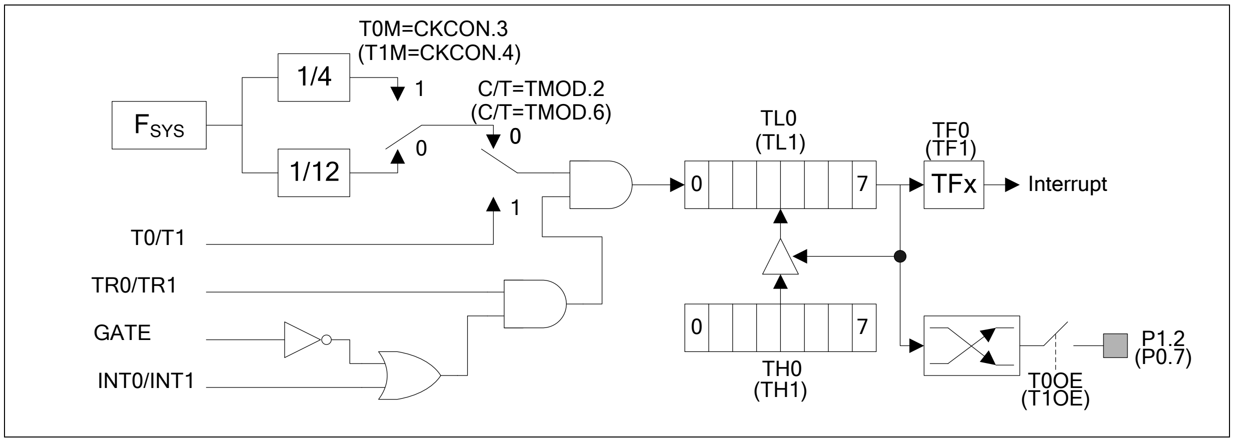


Figure 10-2 Timers/Counters 0 and 1 in Mode 1

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

In Mode 2, the Timer/Counter is in auto-reload mode. In this mode, TLx acts as an 8-bit count register whereas THx holds the reload value. When the TLx register overflows from FFH to 00H, the TFx bit in TCON is set and TLx is reloaded with the contents of THx and the counting process continues from here. The reload operation leaves the contents of the THx register unchanged. This feature is best suitable for UART baud rate generator for it runs without continuous software intervention. Note that only Timer1 can be the baud rate source for UART. Counting is enabled by the TRx bit and proper setting of GATE and $\overline{\text{INTx}}$ pins. The functions of GATE and $\overline{\text{INTx}}$ pins are just the same as Mode 0 and 1.



While any input capture channel is enabled and the selected edge trigger occurs, the content of the free running Timer 2 counter, TH2 and TL2, will be captured, transferred, and stores into the capture registers CnH and CnL. The edge triggering also causes CAPFn (CAPCON0.n) is set by hardware. The interrupt will also be generated if ECPTF (EIE.2) and EA bit are both set. For three input capture flags shares the same interrupt vector, the user should check CAPFn to confirm which channel comes the input capture edge. These flags should be cleared by software.

The bit CAPCR (T2MOD.3) benefits the implement of period calculation. Setting CAPCR makes the hardware clear Timer 2 as 0000H automatically after the value of TH2 and TL2 have been captured after an input capture edge event occurs. It eliminates the routine software overhead of writing 16-bit counter or an arithmetic subtraction.

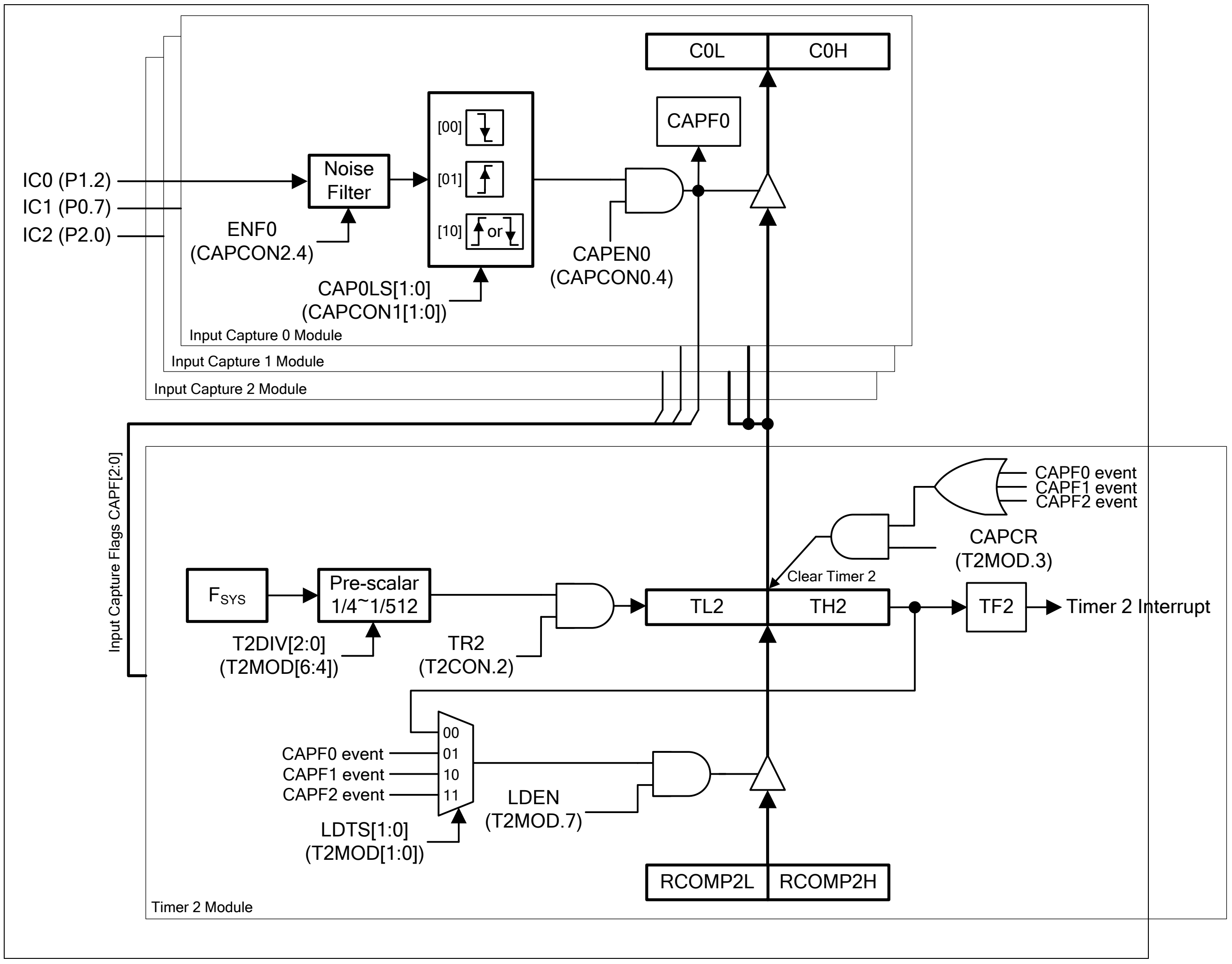


Figure 10-5 Timer 2 Input Capture and Auto-reload Mode Function Block

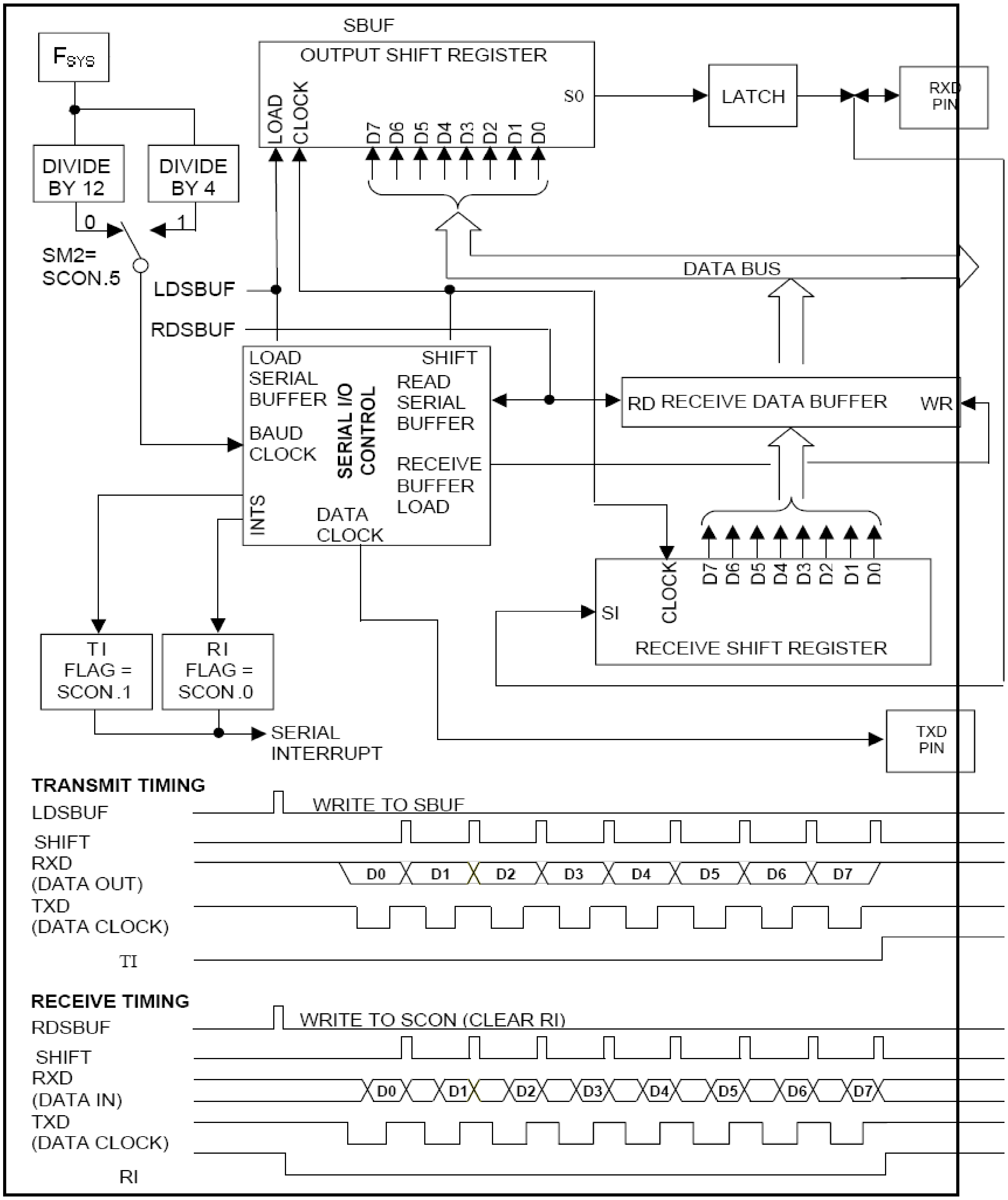


Figure 12-1 Serial Port Mode 0 Function Block

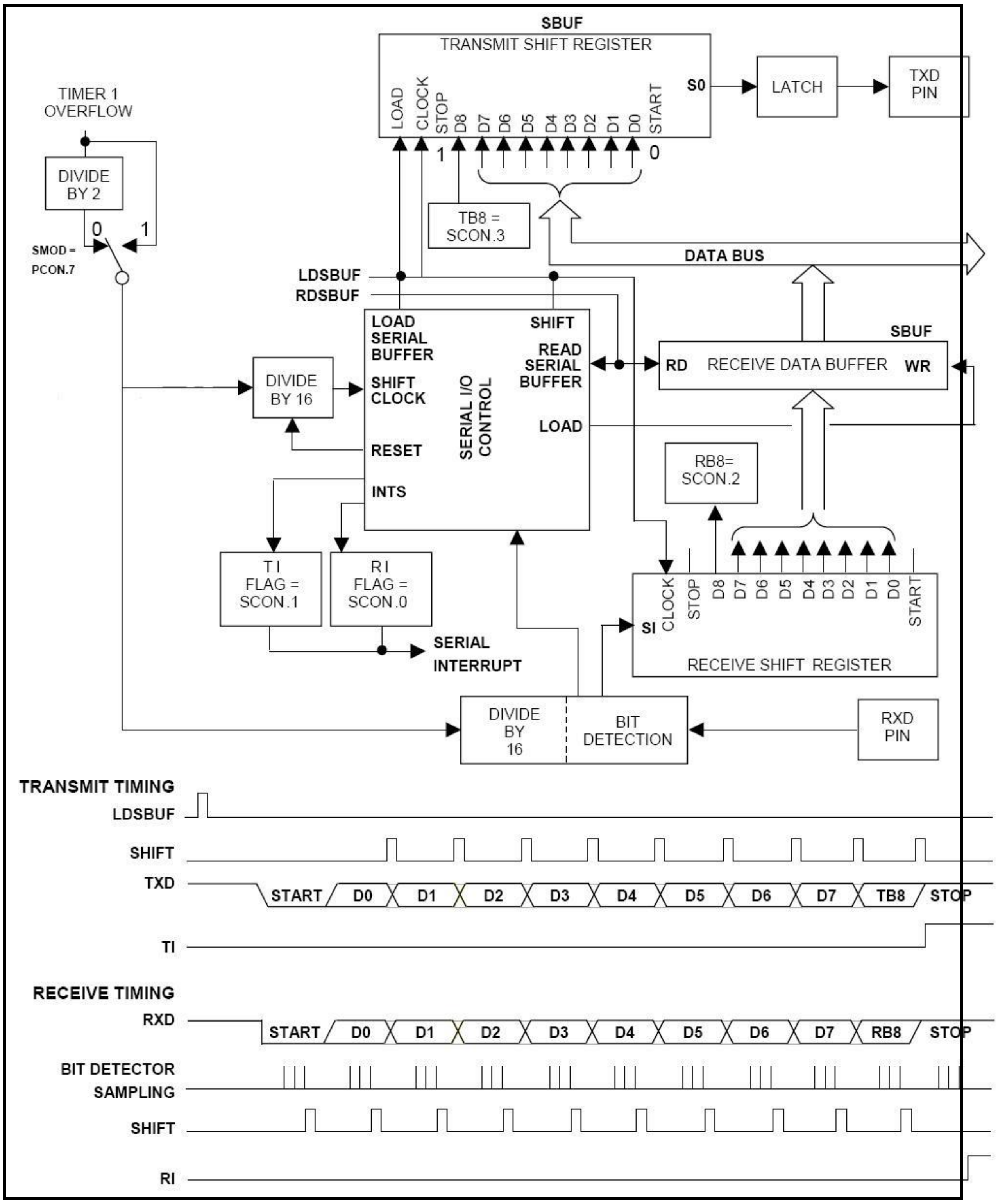


Figure 12-4 Serial Port Mode 3 Function Block

SADDR – Slave Address

7	6	5	4	3	2	1	0
SADDR[7:0]							
R/W							

Address: A9H Reset value: 0000 0000B

Bit	Name	Description
7:0	SADDR[7:0]	**Slave Address** This byte specifies the microcontroller's own slave address for UART multiprocessor communication.

SADEN – Slave Address Mask

7	6	5	4	3	2	1	0
SADEN[7:0]							
R/W							

Address: B9H Reset value: 0000 0000B

Bit	Name	Description
7:0	SADEN[7:0]	**Slave Address Mask** This byte 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 examples will help to show the versatility of this scheme.

Example 1, slave 0:

```
SADDR = 11000000b
SADEN = 11111101b
Given = 110000X0b
```

Example 2, slave 1:

```
SADDR = 11000000b
SADEN = 11111110b
Given = 1100000Xb
```

In the above example SADDR is the same and the SADEN data is used to differentiate between the two slaves. Slave 0 requires a 0 in bit 0 and it ignores bit 1. Slave 1 requires a 0 in bit 1 and bit 0 is ignored. A unique address for Slave 0 would be 1100 0010B since slave 1 requires a 0 in bit 1. A unique address for slave 1 would be 11000001b since a 1 in bit 0 will exclude slave 0. Both slaves can be selected at the same time by an address which has bit 0 = 0 (for slave 0) and bit 1 = 0 (for slave 1). Thus, both could be addressed with 11000000b.

In a more complex system the following could be used to select slaves 1 and 2 while excluding slave 0:

KBLS1 – Keyboard Level Select 1[1]

7	6	5	4	3	2	1	0
KBLS1[7:0]							
R/W							

Address: ECH   Reset value: 0000 0000B

Bit	Name	Description
7:0	KBLS1[7:0]	**Keyboard Level Select 1**

[1] **KBLS1 and KBLS0 are used in combination to determine the input type of each channel of KBI (on P0). Refer to Table 14–1 Configuration for Different KBI Level Select.**

ADCCON0 – ADC Control Register 0

7	6	5	4	3	2	1	0
ADC.1	ADC.0	ADCEX	ADCI	ADCS	AADR2	AADR1	AADR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: F8H Reset value: 0000 0000B

Bit	Name	Description
7	ADC.1	ADC conversion result.
6	ADC.0	ADC conversion result.
5	ADCEX	0 = Disable external start of conversion by P1.4. 1 = Enable external start of conversion by P1.4. The STADC signal at least 1 machine-cycle.
4	ADCI	0 = The ADC is not busy. 1 = The ADC conversion result is ready to be read. An interrupt is invoked if it is enabled. It can not set by software.
3	ADCS	ADC Start and Status: Set this bit to start an A/D conversion. It may also be set by STADC if ADCEX is 1. This signal remains high while the ADC is busy and is reset right after ADCI is set. Notes: It is recommended to clear ADCI ***before*** ADCS is set. However, if ADCI is cleared and ADCS is set at the same time, a new A/D conversion may start on the same channel. Software clearing of ADCS will abort conversion in progress. ADC cannot start a new conversion while ADCS or ADCI is high.
2	AADR2	ADC input select.
1	AADR1	ADC input select.
0	AADR0	ADC input select.

ADCI	ADCS	ADC Status
0	0	ADC not busy; A conversion can be started.
0	1	ADC busy; Start of a new conversion is blocked
1	0	Conversion completed; the start of a new conversion requires ADCI = 0
1	1	Conversion completed; the start of a new conversion requires ADCI = 0

If ADC is cleared by software while ADCS is set at the same time, a new A/D conversion with the same channel number may be started. However, it is recommended to reset ADCI before ADCS is set.

ADDR2, AADR1, AADR0: ADC Analog Input Channel select bits:

These bits can only be changed when ADCI and ADCS are both zero.

AADR2	AADR1	AADR0	Selected Analog Channel
0	0	0	ADC0 (P0.1)

16 Inter-Integrated Circuit (I^2C)

16.1 Features

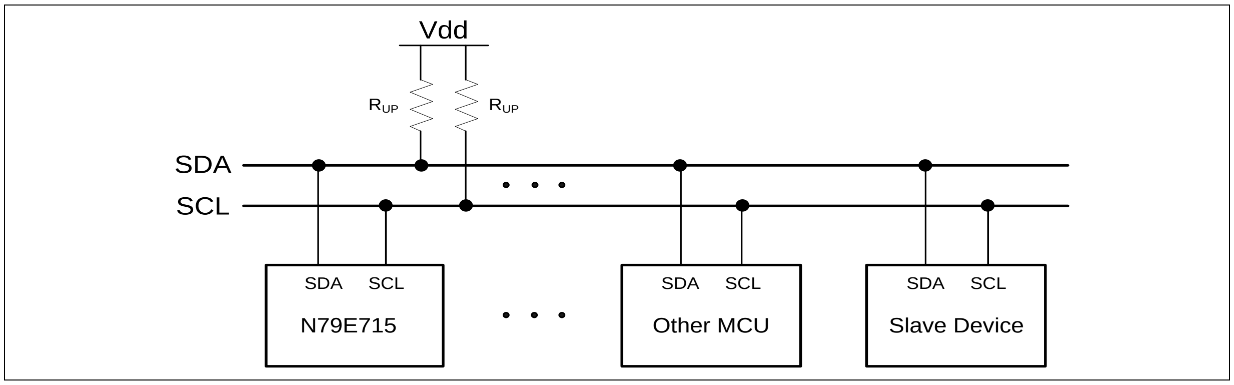
The Inter-Integrated Circuit (I^2C) bus serves as a serial interface between the microcontroller and the I^2C devices such as EEPROM, LCD module, and so on. The I^2C bus used two wires design (a serial data line SDA and a serial clock line SCL) to transfer information between devices.

The I^2C bus uses bidirectional data transfer between masters and slaves. There is no central master and the multi-master system is allowed by arbitration between simultaneously transmitting masters. The serial clock synchronization allows devices with different bit rates to communicate via one serial bus. The I^2C bus supports four transfer modes including master transmitter mode, master receiver mode, slave receiver mode, and slave transmitter mode. The I^2C interface only supports 7-bit addressing mode and General Call can be accepted. The I^2C can meet both standard (up to 100kbps) and fast (up to 400kbps) speeds.

16.2 Functional Description

For the bidirectional transfer operation, the SDA and SCL pins should be connected to open-drain pads. This implements a wired-AND function which is essential to the operation of the interface. A low level on a I^2C bus line is generated when one or more I^2C devices output a “0”. A high level is generated when all I^2C devices output “1”, allowing the pull-up resistors to pull the line high.

In N79E715, the user should set output latches of P1.2 and P1.3. as logic 1 before enabling the I^2C function by setting I2CEN (I2CON.6). The P1.2 and P1.3 are configured as the open-drain I/O once the I^2C function is enabled. The P1M2 and P1M1 will also be re-configured. It is strongly recommended that the Schmitt trigger input buffer be enabled by setting P1S for improved glitch suppression.



I2STA – I^{2}C Status

7	6	5	4	3	2	1	0
I2STA[7:3]					0	0	0
R					R	R	R

Address: BDH Reset value: 1111 1000B

Bit	Name	Description
7:3	I2STA[7:3]	**I^{2}C Status Code** The most five bits of I2STA contains the status code. There are 26 possible status codes. When I2STA is F8H, no relevant state information is available and SI flag keeps 0. All other 25 status codes correspond to the I^{2}C states. When each of the status is entered, SI will be set as logic 1 and a interrupt is requested.
2:0	-	**Reserved** The least three bits of I2STA are always read as 0.

I2DAT – I^{2}C Data

7	6	5	4	3	2	1	0
I2DAT[7:0]							
R/W							

Address: BCH Reset value: 0000 0000B

Bit	Name	Description
7:0	I2DAT[7:0]	**I^{2}C Data** I2DAT contains a byte of the I^{2}C data is going to transmit or a byte has just received. Data in I2DAT remains as long as SI is logic 1. The result of reading or writing I2DAT during I^{2}C transmit progress is unpredicted. While data in I2DAT shift out, data on the bus is simultaneously being shifted into update I2DAT. I2DAT always shows the last byte that presented on the I^{2}C bus. Thus the event of lost arbitration, the original value of I2DAT changes after the transaction.

I2ADDR – I^{2}C Own Slave Address

7	6	5	4	3	2	1	0
I2ADDR[7:1]							GC
R/W							R/W

Address: C1H Reset value: 0000 0000B

Bit	Name	Description
7:1	I2ADDR[7:1]	**I^{2}C device's own Slave Address** In Master mode: These bits have no effect. In Slave mode: The 7 bits define the slave address of this I^{2}C device by the user. The master should address this I^{2}C device by sending the same address in the first byte data after a START or a repeated START condition. If the AA flag is set, this I^{2}C device will acknowledge the master after receiving its own address and become an addressed slave. Otherwise, the addressing from the master will be ignored.

Bit	Name	Description
0	GC	**General Call Bit** In Master mode: This bit has no effect. In Slave mode: 0 = General Call is always ignored. 1 = General Call is recognized if AA flag is 1; otherwise, it is ignored if AA is 0.

I2CLK – I^{2}C Clock

7	6	5	4	3	2	1	0
I2CLK[7:0]							
R/W							

Address: BEH Reset value: 0000 1110B

Bit	Name	Description
7:0	I2CLK[7:0]	**I^{2}C Clock Setting** In Master mode: This register determines the clock rate of I^{2}C bus when the device is in Master mode. The clock rate follows the formula below. $F_{I^2C} = \frac{F_{SYS}}{4 \times (I2CLK + 1)}$ The default value will make the clock rate of I^{2}C bus 400kbps if the clock system 24 MHz with DIVM 1/4 mode is used. Note that the I2CLK value of 00H and 01H are not valid. This is an implement limitation. In Slave mode: This byte has no effect. In slave mode, the I^{2}C device will automatically synchronize with any given clock rate up to 400kps.

16.4 Operation Modes

In I^{2}C protocol definitions, there are four operating modes including master transmitter, master receiver, slave receive, and slave transmitter. There is also a special mode called General Call. Its operation is similar to master transmitter mode.

16.4.1 Master Transmitter Mode

In Master Transmitter mode, several bytes of data are transmitted to a slave receiver. The master should prepare by setting desired clock rate in I2CLK and enabling I^{2}C bus by writing I2CEN (I2CON.6) as logic 1. The master transmitter mode may now be entered by setting STA (I2CON.5) bit as 1. The hardware will test the bus and generate a START condition as soon as the bus becomes free. After a START condition is successfully produced, the SI flag (I2CON.3) will be set and the status code in I2STA show 08H. The progress is continued by loading I2DAT with the target slave address

```
             I2DAT = SLA_ADDR1;          //LOAD SLA+W/R
             break;
     case 0x10:                          /*10H, a repeated START transmitted*/
             STA = 0;
             I2DAT = SLA_ADDR2;
             break;
     //=======================
     //Master Transmitter Mode
     //=======================
     case 0x18:                          /*18H,    SLA+W    transmitted,    ACK
received*/
             I2DAT = NEXT_SEND_DATA1;    //LOAD DATA
             break;
     case 0x20:                          /*20H,    SLA+W    transmitted,    NACK
received*/
             STO = 1;                    //transmit STOP
             AA = 1;                     //ready for ACK own SLA+W/R
             break;
     case 0x28:                          /*28H,    DATA    transmitted,    ACK
received*/
             if (Conti_TX_Data)          //if continuing to send DATA
                     I2DAT = NEXT_SEND_DATA2;
             else                        //if no DATA to be sent
             {
                     STO = 1;
                     AA = 1;
             }
             break;
     case 0x30:                          /*30H,    DATA    transmitted,    NACK
received*/
             STO = 1;
             AA = 1;
             break;
     //===========
     //Master Mode
     //===========
     case 0x38:                          /*38H, arbitration lost*/
             STA = 1;                    //retry to transmit START if bus free
             break;
     //====================
     //Master Receiver Mode
     //====================
     case 0x40:                          /*40H,    SLA+R    transmitted,    ACK
received*/
             AA = 1;                     //ACK next received DATA
             break;
     case 0x48:                          /*48H,    SLA+R    transmitted,    NACK
received*/
             STO = 1;
             AA = 1;
             break;
     case 0x50:                          /*50H,    DATA    received,    ACK
transmitted*/
             DATA_RECEIVED1 = I2DAT;     //store received DATA
             if (To_RX_Last_Data1)       //if last DATA will be received
                     AA = 0;             //not ACK next received DATA
             else                        //if continuing receiving DATA
                     AA = 1;
             break;
     case 0x58:                          /*58H,    DATA    received,    NACK
transmitted*/
```

Bit	Name	Description
3:2	FP[1:0]	Select PWM frequency pre-scalar select bits. The clock source of pre-scalar, Fpwm is in phase with F_{SYS} if PWMRUN=1. FP[1:0] / Fpwm: 00 — F_{SYS} (Default) 01 — F_{SYS} /2 10 — F_{SYS} /4 11 — F_{SYS} /16
1	-	**Reserved**
0	BKF	**External Brake Pin Flag** 0 = PWM is not brake. 1 = PWM is brake by external brake pin. It will be cleared by software.

FP[1:0]	Fpwm
00	F_{SYS} (Default)
01	F_{SYS} /2
10	F_{SYS} /4
11	F_{SYS} /16

The Brake function, which is controlled by the contents of the PWMCON1 register, is somewhat unique. In general when Brake is asserted the four PWM outputs are forced to a user selected state, namely the state selected by PWMCON1 bits 0 to 3. As shown in the description of the operation of the PWMCON1 register if PWMCON1.4 is a “1” brake is asserted under the control PWMCON1.7, BKCH, and PWMCON1.5, BPEN. As shown if both are a “0” Brake is asserted. If PWMCON1.7 is a “1” brake is asserted when the run bit, PWMCON0.7, is a “0.” If PWMCON1.6 is a “1” brake is asserted when the Brake Pin, P0.2, has the same polarity as PWMCON1.6. When brake is asserted in response to this pin the RUN bit, PWMCON0.7, is automatically cleared and BKF(PWMCON2.0) flag will be set. The combination of both PWMCON1.7 and PWMCON1.5 being a “1” is not allowed.

Since the Brake Pin being asserted will automatically clear the Run bit of PWMCON0.7and BKF(PWMCON2.0) flag will be set, the user program can poll this bit or enable PWM’s brake interrupt to determine when the Brake Pin causes a brake to occur. The other method for detecting a brake caused by the Brake Pin would be to tie the Brake Pin to one of the external interrupt pins. This latter approach is needed if the Brake signal can be of insufficient length to ensure that it can be captured by a polling routine. When, after being asserted, the condition causing the brake is removed, the PWM outputs go to whatever state that had immediately prior to the brake. This means that to go from brake being asserted to having the PWM run without going through an indeterminate state care should be taken. If the Brake Pin causes brake to be asserted the following prototype code will allow the PWM to go from brake to run smoothly by software polling BKF flag or enable PWM’s interrupt.

Note that if a narrow pulse on the Brake Pin causes brake to be asserted, it may not be possible to go through the above code before the end of the pulse. In this case, in addition to the code shown, an external latch on the Brake Pin may be required to ensure that there is a smooth transition in going from brake to run.

18 Timed Access Protection (TA)

The N79E715 has several features like the Watchdog Timer, the ISP function, Boot select control, etc. are crucial to proper operation of the system. If leaving these control registers unprotected, errant code may write undetermined value into them, it results in incorrect operation and loss of control. To prevent this risk, the N79E715 has a protection scheme which limits the write access to critical SFRs. This protection scheme is done using a timed access. The following registers are related to TA process.

TA – Timed Access

7	6	5	4	3	2	1	0
TA[7:0]							
W							

Address: C7H Reset value: 1111 1111B

Bit	Name	Description
7:0	TA[7:0]	**Timed Access** The timed access register controls the access to protected SFRs. To access protected bits, the user should first write AAH to the TA and immediately followed by a write of 55H to TA. After the two steps, a writing permission window is opened for three machine-cycles during which the user may write to protected SFRs.

In timed access method, the bits, which are protected, have a timed write enable window. A write is successful only if this window is active, otherwise the write will be discarded. When the software writes AAH to TA, a counter is started. This counter waits for three machine-cycles looking for a write of 55H to TA. If the second write of 55H occurs within three machine-cycles of the first write of AAH, then the timed access window is opened. It remains open for three machine-cycles during which the user may write to the protected bits. After three machine-cycles, this window automatically closes. Once the window closes, the procedure should be repeated to access the other protected bits. Not that the TA protected SFRs are required timed access for writing. However, the reading is not protected. The user may read TA protected SFR without giving AAH and 55H to TA. The suggestion code for opening the timed access window is shown below.

```
(CLR   EA)                ;if any interrupt is enabled, disable temporarily
MOV    TA, #0AAH
MOV    TA, #55H
(Instruction that writes a TA protected register)
(SETB  EA)                ;resume interrupts enabled
```

The writes of AAH and 55H should occur within 3 machine-cycles of each other. Interrupts should be disabled during this procedure to avoid delay between the two writes. If there is no interrupt enabled,

R/W	R/W	R/W	R/W	-	R/W	R/W	R/W

Address: E8H Reset value: 0000 0000B

Bit	Name	Description
7	ET2	0 = Disable Timer 2 Interrupt. 1 = Enable Timer 2 Interrupt.
6	ESPI	SPI interrupt enable: 0 = Disable SPI Interrupt. 1 = Enable SPI Interrupt.
5	EPWM	0 = Disable PWM Interrupt when external brake pin was braked. 1 = Enable PWM Interrupt when external brake pin was braked.
4	EWDI	0 = Disable Watchdog Timer Interrupt. 1 = Enable Watchdog Timer Interrupt.
3	-	Reserved
2	ECPTF	0 = Disable capture interrupts. 1 = Enable capture interrupts.
1	EKB	0 = Disable Keypad Interrupt. 1 = Enable Keypad Interrupt.
0	EI2C	0 = Disable I^{2}C Interrupt. 1 = Enable I^{2}C Interrupt.

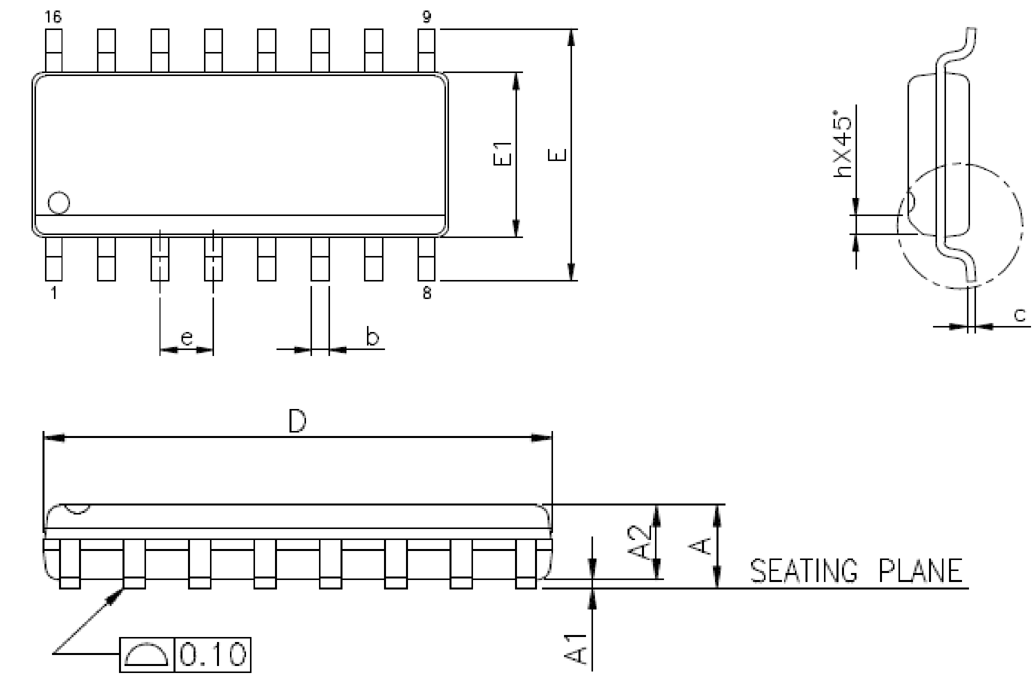
IP – Interrupt Priority-0 Register

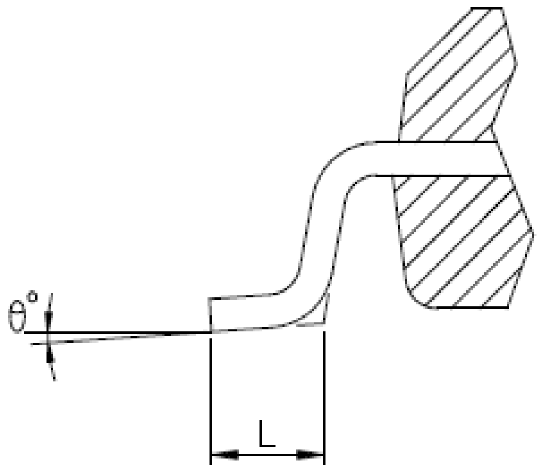
7	6	5	4	3	2	1	0
PCAP	PADC	PBOD	PS	PT1	PX1	PT0	PX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: B8H Reset value: 0000 0000B

Bit	Name	Description
7	PCAP	1 = Set interrupt high priority of Capture 0/1/2 as highest priority level.
6	PADC	1 = Set interrupt priority of ADC as higher priority level.
5	PBOD	1 = Set interrupt priority of BOD Detector as higher priority level.
4	PS	1 = Set interrupt priority of Serial port 0 as higher priority level.

30.5 16-pin SOP - 150 mil





SYMBOLS	MIN.	NOM.	MAX.
A	1.51	1.63	1.75
A1	0.10	0.18	0.25
A2	1.25	1.45	1.65
b	0.31	0.41	0.51
c	0.10	0.18	0.25
D	9.70	9.90	10.10
E	5.80	6.00	6.20
E1	3.70	3.90	4.10
e	1.17	1.27	1.37
L	0.40	0.84	1.27
h	0.25	0.38	0.50
θ°	0	4	8

UNIT : mm