Nuvoton 8-bit Microcontroller

N79E845A
N79E844A
N79E843A

Datasheet

Version: A1.0

Contents

1 Description

The **N79E845** series are the 8-bit Turbo 51 (4T Mode) microcontrollers embedded with 16K[1]/8K/4K Flash EPROM which can be programmed through universal hardware writer, serial ICP (In Circuit Program) programmer, software ISP function. The instruction sets of the **N79E845** series are fully compatible with the standard 8052. The **N79E845** series contain a 16K/8K/4K bytes of Application Flash EPROM (APROM) memory, 4K bytes Data Flash memory and 2K bytes Load Flash EPROM (LDROM) memory; a 256 bytes of direct and indirect RAM, 256 bytes MOVX RAM;25 I/O with bit-addressable I/O ports; two 16-bit timer/counters; 8-channel multiplexed 10-bit A/D convert; 4-channel 10-bit PWM; three serial ports that includes a SPI, an I2C and an enhanced full duplex serial port; 2-level BOD voltage detection/reset, and power-on reset(POR). The **N79E845** series also supports internal RC oscillator at the nominal frequency of 22.1184MHz with ±1% accuracy by factory trimming mechanism. These peripherals are supported by 16 sources of four-level interrupt capability. To facilitate programming and verification, the Flash EPROM inside the **N79E845** series allow the program memory to be programmed and read electronically. Once the code is confirmed, the user can protect the code for security.

N79E845 microcontroller series, feature wide operating voltage range, built-in rich analog and digital peripherals and nonvolatile Flash memory, are widely suit for general control applications, DC/BLDC motor driving system and home appliance.

[1] For **N79E845**, Data Flash and APROM share 16k-byte space.

2 Features

- Core
  - Fully static design 8-bit Turbo 51 (4T) CMOS microcontroller
  - Instruction sets are fully compatible with the MCS-51
- Operating voltage range
  - **V_{DD} = 2.4V to 5.5V @ F_{SYS} = 4MHz to 24MHz**
- Operating temperature range
  - **-40°C ~85°C**
- Clock Source
  - High speed external oscillator:
    - Up to 24 MHz Crystal and resonator (enabled by config-bits)
  - Internal RC oscillator: 22.1184MHz/11.0592MHz (selectable by config-bit)
    - ±1% at V_{DD} = 2.4V ~ 5.5V and 25°C condition
    - ±5% at V_{DD} = 2.4V ~ 5.5V and -40°C ~ 85°C condition
  - Flexible CPU clock source configurable by config-bits and software
  - 8-bit Programmable CPU clock divider(DIVM)
- On-chip Memory
  - 2K-byte LD Flash bock for ISP function (LDROM)
  - 16K/8K/4K bytes of on-chip Flash EPROM with page erase and byte program
    - 16K/8K/4K Application Program Flash Block (APROM)
  - Data Flash 4K bytes or Full usage of rest Flash (DATAFlash)
    - Configurable APROM and DataFlash size in 16K AP Flash mode
  - N79E845A: Total 16K-byte separated to AP Flash and Data Flash by config-bits definition
  - N79E844A: 8K-byte AP Flash and 4K-byte Data Flash
  - N79E843A: 4K-byte AP Flash and 4K-byte Data Flash
  - APROM/LDROM and DATAFlash security protection.
  - Flash page size is 128 bytes
  - 256 bytes of on-chip direct/indirect RAM
  - 256 bytes of MOVX-RAM, accessed by MOVX instruction
  - On-chip Flash programmed through
    - Parallel H/W writer mode
    - Serial In-Circuit-Program mode (ICP)
    - Software implemented ISP (In-System-Program)
- I/O Ports
  - Maximum 17 I/O pins
  - Each pin supports 4 software configurable output modes
  - Software selectable TTL or Schmitt trigger input type per port
  - 14 interrupts source with four levels of priority
  - LED drive capability 40mA on P10, P11, P14, P16, P17

- LED drive capability 20mA on port 0, 3 pins

- Timer/Counter
  - 2 sets 16-bit timer/counters
  - One Timer with **two** inputs captures capability
- Watch Dog Timer
  - Programmable Watchdog Timer
  - Clock source supported by internal 10KHz 50% accuracy RC oscillator
- Serial ports (UART, SPI, I2C)
  - One set enhanced full duplex UART port with framing error detection and automatic address recognition.
  - One set SPI with master/slave capability.
  - One set I2C with master/slave capability
- PWM
  - 4 channels 10-bit PWM outputs with one brake/fault input
- KBI
  - 8-keypad interrupt inputs(KBI) with 8 falling/both-edge detection pins selected by SW
- ADC
  - 10-bit A/D converter
  - Up to 150 Ksps.(sample/second)
  - **7** analog input channels
- Brownout Detector
  - 2-level (3.8V/2.7V) BOD detector
  - Supports interrupt and reset options
- POR (Power on Reset)
  - Threshold voltage levels is 2.0V
- Built-in power management.
  - Idle mode
  - Power down mode with optional enabled low power BOD and WDT functions
- Development Tools
  - ICE(In Circuit Emulation) tool
  - H/W writer
  - ICP programmer
  - ISP programmer
- Packages
  - N79E845ADG --- PDIP20
  - N79E845ASG --- SOP20
  - N79E845AWG --- TSSOP20
  - N79E844ADG --- PDIP20
  - N79E844ASG --- SOP20
  - N79E844AWG --- TSSOP20
  - N79E843ADG --- PDIP20

- N79E843ASG --- SOP20
- N79E843AWG --- TSSOP20

3 PARTS INFORMATION LIST

Lead Free (RoHS) Parts information list

PART NO.	APROM	LDROM	RAM	DATA FLASH	PACKAGE	REMARK
N79E845ASG	16KB	2KB	512B	-	SOP-20 Pin	The DATA shares APROM
N79E845AWG	16KB	2KB	512B	-	TSSOP-20 Pin	The DATA shares APROM
N79E845ADG	16KB	2KB	512B	-	PDIP-20 Pin	The DATA shares APROM
N79E844ASG	8KB	2KB	512B	4KB	SOP-20 Pin	
N79E844AWG	8KB	2KB	512B	4KB	TSSOP-20 Pin	
N79E844ADG	8KB	2KB	512B	4KB	PDIP-20 Pin	
N79E843ASG	4KB	2KB	512B	4KB	SOP-20 Pin	
N79E843AWG	4KB	2KB	512B	4KB	TSSOP-20 Pin	
N79E843ADG	4KB	2KB	512B	4KB	PDIP-20 Pin	

Table 3-1: Lead Free (RoHS) Parts information list

4 Block Diagram

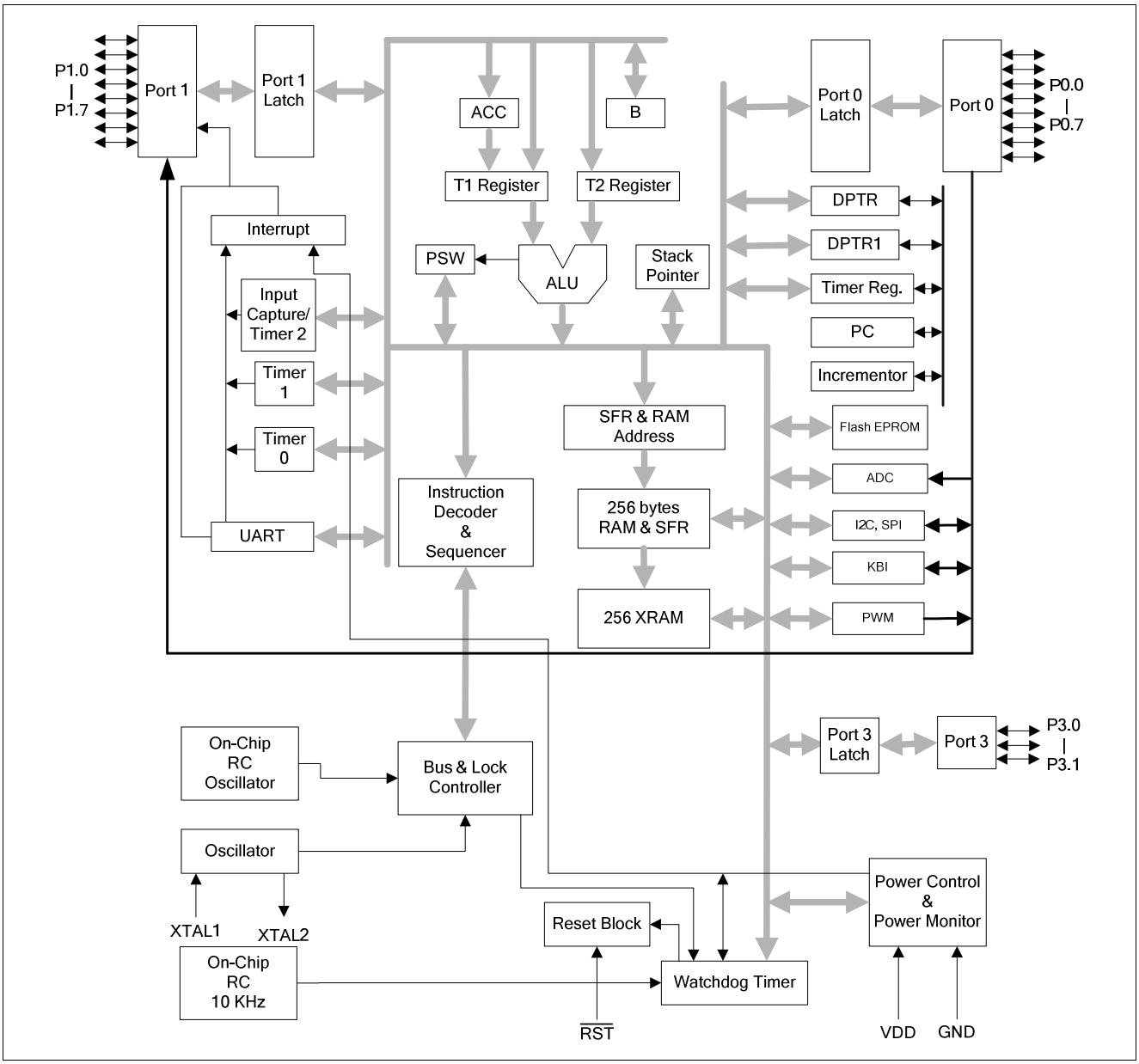


Figure 4–1. N79E845 Series Function Block Diagram

5 Pin Configuration

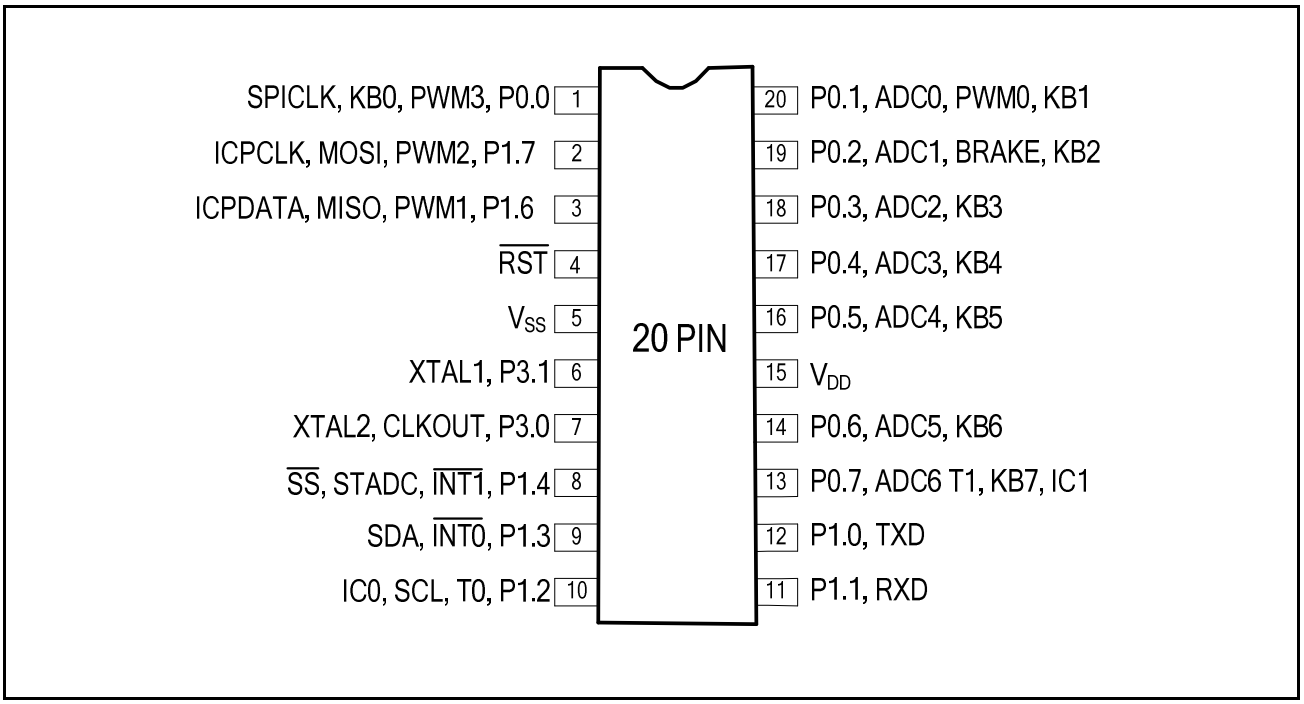


Figure 5–1. Pin Assignment of TSSOP/PDIP/SOP 20 pin

Table 5–1. Pin Description

Pin number	Symbol	Alternate Function				Type[1]	Description
DIP20 SOP20 TSSOP20		1	2	3			
21	V_{DD}					P	**POWER SUPPLY:** Supply voltage V_{DD} for operation.
7	V_{SS}					P	**GROUND:** Ground potential.
6	/RST					I (ST)	**RESET:** Chip reset pin, it is Low active.
3	P0.0	PWM3		KB0	SPICLK	I/O	**PORT0:** Port 0 has 4-type I/O port. Its multifunction pins are for PWM0, PWM3, T1, BRAKE, SPICLK, ADC0~ADC6 and KB0~KB7. ADC0 ~ADC6: ADC channel input. KB0 ~ KB7: Key Board Input The PWM0 and PWM3 are PWM output channel T1: Timer 1 External Input SPICLK: SPI-1 clock pin
26	P0.1	PWM0	ADC0	KB1		I/O	
25	P0.2	BRAKE	ADC1	KB2		I/O	
24	P0.3		ADC2	KB3		I/O	
23	P0.4		ADC3	KB4		I/O	
22	P0.5		ADC4	KB5		I/O	
20	P0.6		ADC5	KB6		I/O	
19	P0.7	T1	ADC6	KB7	IC1	I/O	
18	P1.0	TXD				I/O	**PORT1:** Port 1 has 4-type I/O port. Its multifunction pins are for TXD, RXD, T0, /INT0, /INT1, SCL, SDA, STADC, ICPDAT, ICPCLK and /SS, MISO, MOSI. The TXD and RXD are UART port The SCL and SDA are I2C function with open-drain port. The ICPDAT and ICPCLK are ICP (In Circuit Programming) function pin. The /SS, MISO, MOSI are SPI-1 function pins. The PWM1 and PWM2 are PWM output channel T0: Timer 0 External Input IC0/1: Input Capture pin STADC: ADC trigger by external pin
17	P1.1	RXD				I/O	
12	P1.2	T0		SCL	IC0	D	
11	P1.3	/INT0		SDA		D	
10	P1.4	/INT1	STADC		/SS	I/O	
5	P1.6	PWM1		ICPDAT	MISO	I/O	
4	P1.7	PWM2		ICPCLK	MOSI	I/O	

9	P3.0	XTAL2	CLKOUT			I/O	**PORT3:** Port 3 has 4-type I/O port. Its multifunction pins are for XTAL1, XTAL2 and CLKOUT, **CLKOUT:** Internal RC OSC/4 output pin. **XTAL2:** This is the output pin from the internal inverting amplifier. It emits the inverted signal of XTAL2. **XTAL1:** This is the output pin from the internal inverting amplifier. It emits the inverted signal of XTAL1.
8	P3.1	XTAL1				I/O	

[1] I/O type description I: input, O: output, I/O: quasi bi-direction, D: open-drain, P: power pins, ST: Schmitt trigger.

6 Memory Organization

The **N79E845** series have embedded Flash EEPROM includes 16K/8K/4K bytes Application Program Flash Memory(AP Flash), fixed 4K bytes Data Flash(except the device with 16K AP Flash), fixed 2K bytes Load ROM Flash memory(LD Flash) and Config-bits Flash. **N79E845** series also provide 256 bytes of on-chip direct/indirect RAM and 256 bytes of MOVX-RAM accessed by MOVX instruction.

For the device of 16K-byte AP Flash, the AP Flash block and Data Flash block comprise the 16K bytes embedded Flash. The block size is config-bit/software configurable.

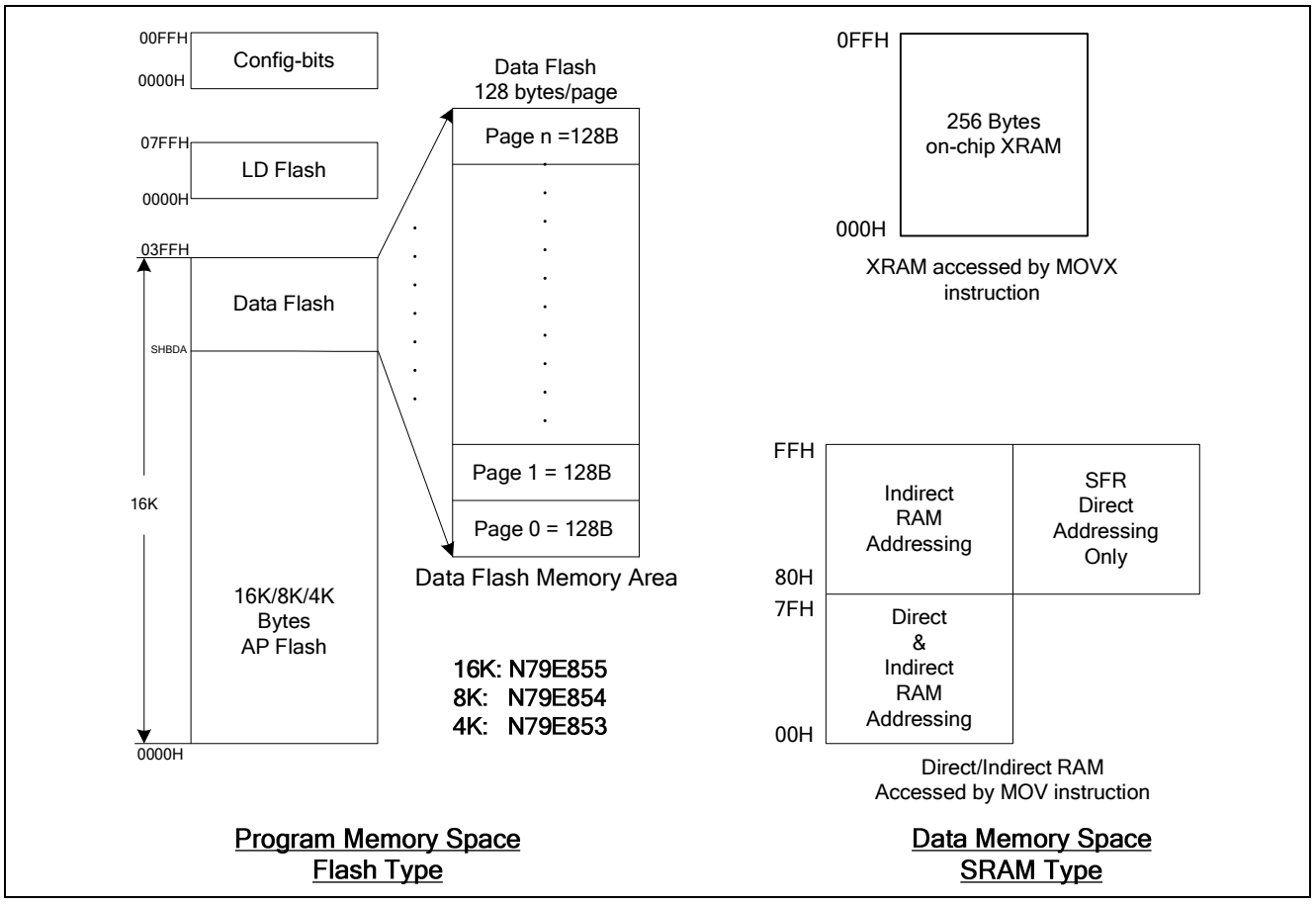


Figure 6-1 N79E845 Series Memory Map

6.1 General Description for Flash IP

The **N79E845** series are built-in a CMOS page-erase and byte-program Embedded Flash Memory which is partitioned into two memory blocks. The page erase operation erases all bytes within a page of 128 bytes.

The embedded flash memory of **N79E845** series are divided into 4 blocks of AP Flash, Data Flash, LD Flash and Config-bits. The detailed configuration is specified in the following sections.

6.1.1 APROM Flash memory

The Program Memory on **N79E845** series can be up to **16K/8K/4K** bytes long. All instructions are fetched for execution from this memory area. The MOVC instruction can also access this memory region.

The user application program is located in AP Flash. When CPU boots from AP Flash (CHPCON.BS=0), CPU starts executing the program from address 0000H. If the value of program counter (PC) is over the space of AP Flash, CPU will execute NOP operand and program counter increases one by one until PC reaches 3FFFH then it wraparounds to address 0000H of AP Flash, the CPU executes the application program again.

6.1.2 LDROM Flash memory

Each device of **N79E845** series is equipped 2K-byte LD Flash stored the ISP application program. User may develop the ISP function in LD Flash for updating application program or Data Flash. Similarly, AP Flash also can re-program LD Flash and Data Flash. The start address of LD Flash is at 0000H corresponding to the physical address of the Flash memory. However, when CPU runs in LD Flash, CPU automatically re-vectors the LD Flash start address to 0000H, therefore user program regards the LD Flash as an independent program memory, meanwhile, with all interrupt vectors that CPU provides.

6.1.3 Data Flash memory

For 16K-byte AP Flash device, the AP and Data block are located in the 16K Flash. The Data Flash start address (DF_Start_Addr) is determined by SFR SHBDA(9CH). The value of SHBDA is loaded with config-bits CHBDA at power-on reset. SFR SHBDA[7:0] has time-access protect at a write to SHBDA. Caution that if SHBDA is set to an unreasonable value, for example over 16K-byte addressed space, CPU will force the Data Flash to be unavailable to software.

For example, if user wants to configure a 16K embedded Flash with 10K AP Flash and 4K Data Flash. The start address of Data Flash is 2800H, so the CHBDA/SHBDA must be programmed with 28H.

6.1.4 Config-bits

There are several bytes of config-bits located Config-bits block. The config-bits define the CPU initial setting after power up or reset. Only hardware parallel writer or hardware ICP writer can erase/program Config-bits. ISP program in LD Flash also can software erase/program Config-bits.

6.2 On-chip XRAM

N79E845 series provide additional on-chip 256 bytes auxiliary RAM called XRAM to enlarge the RAM space. It occupies the address space from 00H through FFH. The 256 bytes of XRAM are indirectly accessed by move external instruction MOVX @DPTR or MOVX @Ri. (See the demo code below.) Note that the stack pointer may not be located in any part of XRAM. Figure 6-1 shows the memory map for this product series.

XRAM demo code:

```
MOV    R0,#23H             ;write #5AH to XRAM with address @23H
MOV    A,#5AH
MOVX   @R0,A

MOV    R1,#23H             ;read from XRAM with address @23H
MOVX   A,@R1

MOV    DPTR,#0023H         ;write #5BH to XRAM with address @0023H
MOV    A,#5BH
MOVX   @DPTR,A

MOV    DPTR,#0023H         ;read from XRAM with address @0023H
MOVX   A,@DPTR
```

6.3 On-chip scratch-pad RAM and SFR

N79E845 series provide the on-chip 256 bytes scratch pad RAM and Special Function Registers (SFRs) which be accessed by software. The SFRs be accessed only by direct addressing, while the on-chip RAM be accessed by either direct or indirect addressing.

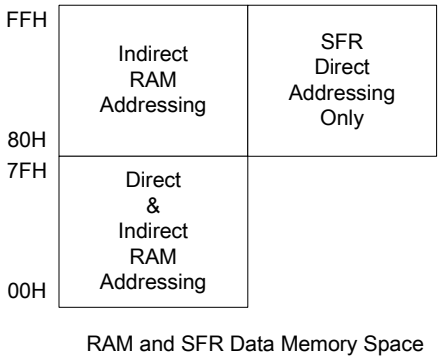
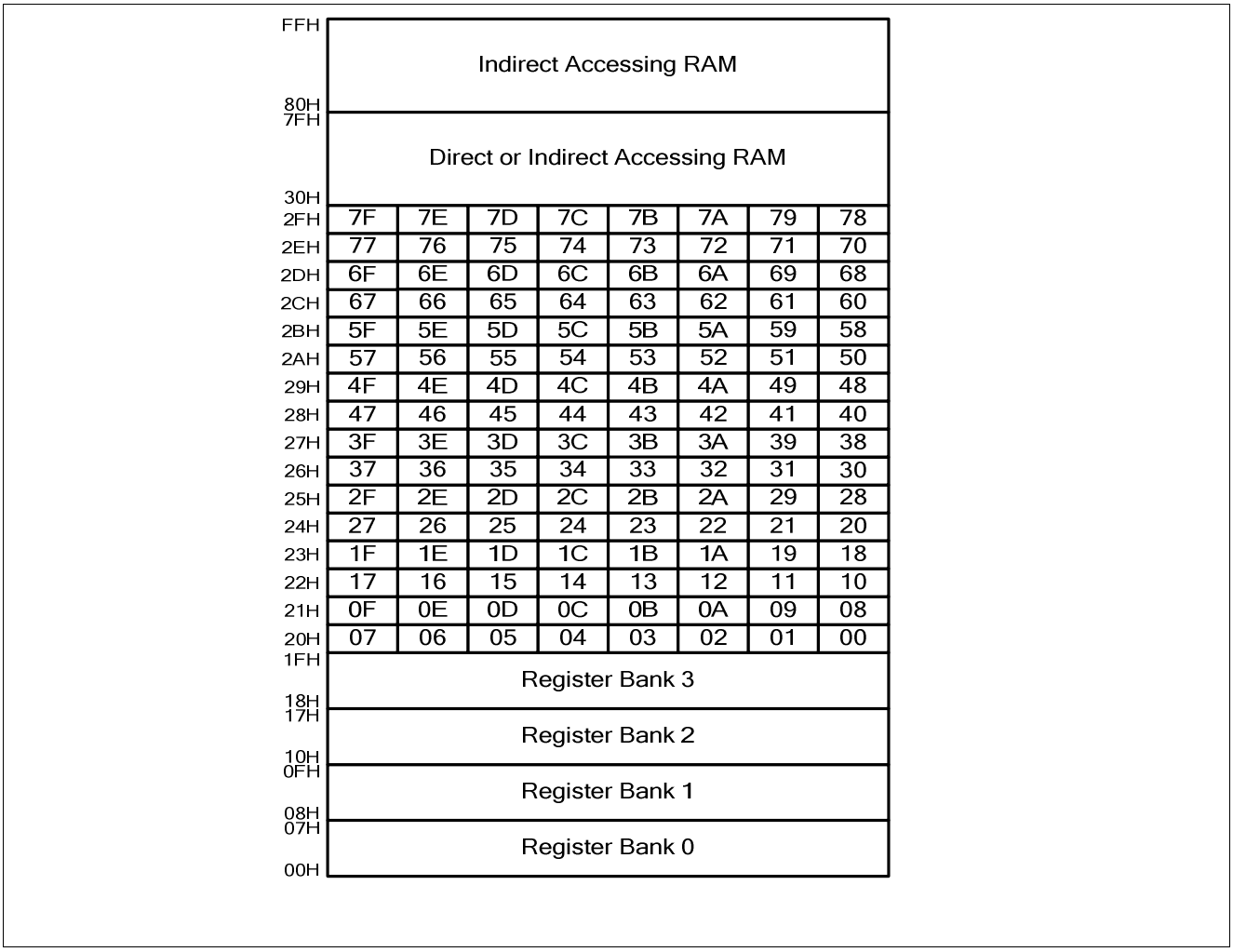


Figure 6-2 256 bytes RAM and SFR

Since the scratch-pad RAM is only 256 byte it can be used only when data contents are small. There are several other special purpose areas within the scratch-pad RAM. These are described as follows.



6.4 Working Registers

There are four sets of working registers, each consisting of eight 8-bit registers. These are termed ads Banks 0, 1, 2, and 3. Individual registers within these banks can be directly accessed by separate instructions. These individual registers are named as R0, R1, R2, R3, R4, R5, R6 and R7. However, at one time the **N79E845** series can work with only one particular bank. The bank selection is done by setting RS1-RS0 bits in the PSW. The R0 and R1 registers are used to store the address for indirect accessing.

6.5 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.

6.6 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.

6.7 On-chip Non-volatile Data Flash

The **N79E845** series additionally has non-volatile Data Flash. The Data Flash is non-volatile so that it remains its content even after the power is off. Therefore, in general application the user can write or read data which rules as parameters or constants. By the software path, the Data Flash can be erased or written only through ISP mode and read through MOVC instruction or ISP read. By the software path, the Data Flash can be erased, written, or read only through ISP mode. Of course, the Data Flash can be accessed via hardware with parallel Programmer/Writer.

The Data Flash size is software adjustable on **N79E845** (16KB) by updating the content of SHBDA. SHBDA[7:0] represents the high byte of 16-bit Data Flash start address and the low byte are hardware set to 00H. The value of SHBDA is loaded from the content of CONFIG1 (CHBDA) after all resets. The application program can dynamically adjust the Data Flash size by resetting SHBDA value. Once the Data Flash size is changed the APROM size is changed accordingly. SHBDA has time access protect while a write to SHBDA is required. Be aware that if CHBDA is 00H, the Data Flash size will be 16k bytes and there will be no APROM.

The Data Flash size is fixed 4k bytes from address 3000H through 3FFFH on **N79E844/N79E843**. SHBDA affects nothing.

The CONFIG bit DFEN (CONFIG0.0) should be programmed as a 0 before access the Data Flash block. If DFEN remains its un-programmed value 1, APROM will occupy whole 16k-byte block in **N79E845** and the 4k-byte block from 3000H through 3FFFH will keep in empty in **N79E844/N79E843**.

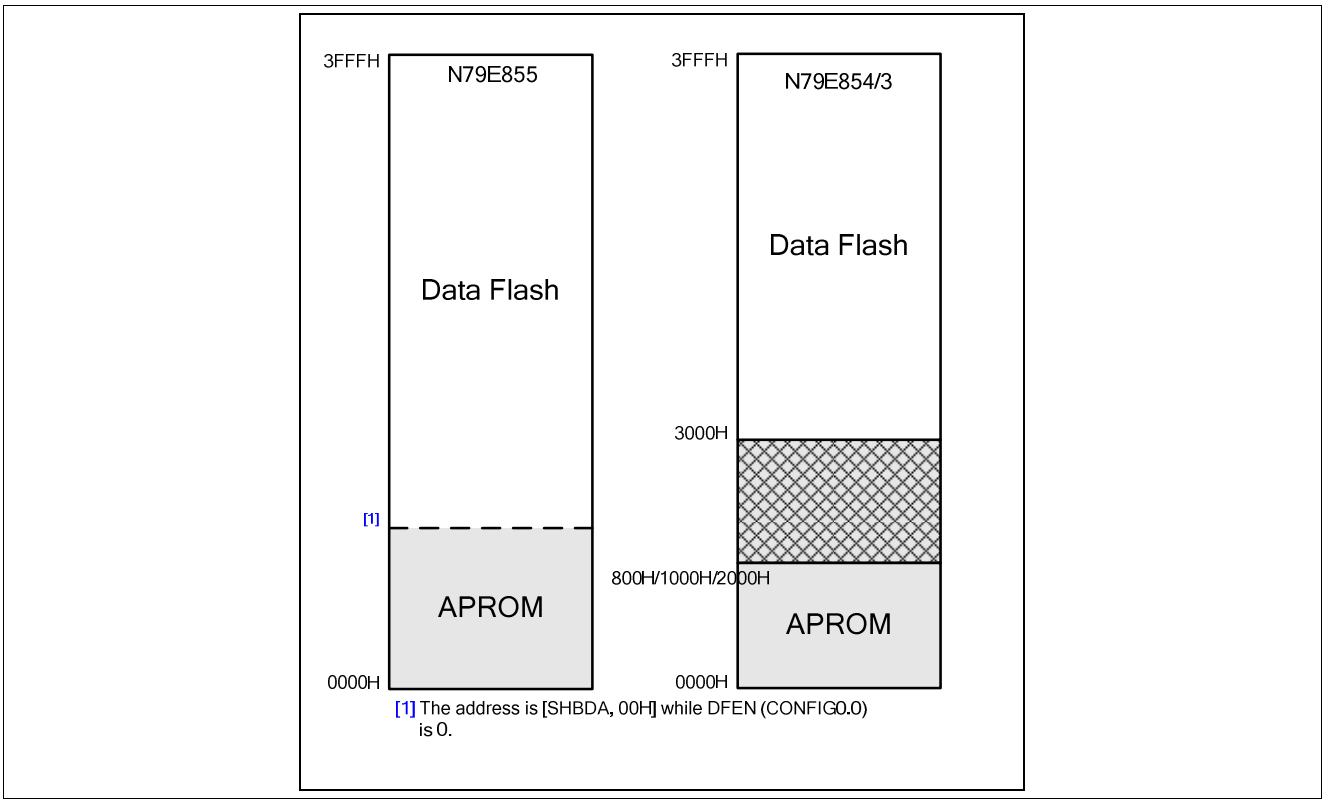


Figure 6–3. N79E845 Data Flash

SHBDA – SFR High Byte of Data Flash Starting Address (TA protected, N79E845 Only)

7	6	5	4	3	2	1	0
SHBDA[7:0] [1]							
r/w							

Address: 9CH

Bit	Name	Description
7:0	SHBDA[7:0]	**SFR high byte of Data Flash starting address.** This byte is valid only when DFEN (CONFIG0.0) being 0 condition. It is used to dynamic adjust the starting address of the Data Flash when the application program is executing.

[1] SHBDA is loaded from CONFIG1 after all resets.

7 Special Function Register (SFR)

The **N79E845** series use Special Function Registers (SFRs) to control and monitor peripherals and their modes. The SFRs reside in the register locations 80~FFH and are accessed by direct addressing only. Some of the SFRs are bit-addressable. This is very useful in cases where users would like to modify a particular bit directly without changing other bits. Those which are bit-addressable SFRs end their addresses as 0H or 8H. **N79E845** series contain all the SFRs presenting in the standard 8051. However some additional SFRs are built in. Therefore, some of unused bytes in the original 8051 have been given new functions. The SFRs is listed as below.

Table 7–1. N79E845 Series Special Function Registers (SFR) Mapping

F8	**ADCCON0**	-	-	-	-	-	-	EIP	FF
F0	**B**	-	-	SPCR	SPSR	SPDR	P0DIDS	EIPH	F7
E8	**EIE**	KBIE	KBIF	KBLS0	KBLS1	C2L	C2H	-	EF
E0	**ACC**	ADCCON1	ADCH	-	C0L	C0H	C1L	C1H	E7
D8	**WDCON0**[3]	PWMPL	PWM0L	PWM1L	PWMCON0	PWM2L	PWM3L	PWMCON1	DF
D0	**PSW**	PWMPH	PWM0H	PWM1H	-	PWM2H	PWM3H	PWMCON2	D7
C8	**T2CON**	T2MOD	RCOMP2L	RCOM2H	TL2	TH2	-	-	CF
C0	**I2CON**	I2ADDR	-	-	-	-	-	TA	C7
B8	**IP**	SADEN	-	-	I2DATA	I2STAT	I2CLK	I2TOC	BF
B0	**P3**	P0M1	P0M2	P1M1	P1M2	-	-	IPH	B7
A8	**IE**	SADDR	-	WDCON1[3]	-	-	ISPFD	ISPCN	AF
A0		-	AUXR1	PMCR[3]	ISPTRG[3]	-	ISPAL	ISPAH	A7
98	**SCON**	SBUF	-	-	SHBDA[3]	-	-	CHPCON[3]	9F
90	**P1**	-	CAPCON0	CAPCON1	CAPCON2	DIVM	P3M1	P3M2	97
88	**TCON**	TMOD	TL0	TL1	TH0	TH1	CKCON	-	8F
80	**P0**	SP	DPL	DPH	-	-	-	PCON	87

In Bold	bit-addressable
-	reserved

[1]. The reserved SFR addresses must be kept in their own initial states. Users should never change their values.

[2]. The SFRs in the column with dark borders are bit-addressable

[3]. With TA-Protection. (Time Access Protection)

Table 7–2. N79E845 Series SFR Descriptions and Reset Values

Symbol	Definition	Address	MSB							LSB	Reset Value[1]
EIP	Interrupt Priority 1	FFH	PT2	PSPI	PPWM	PWDI	-	-	PKB	PI2	0000 0000B
ADCCON0	ADC control register	F8H	(FF) ADC.1	(FE) ADC.0	(FD) ADCEX	(FC) ADCI	(FB) ADCS	(FA) AADR2	(F9) AADR1	(F8) AADR0	XX00 0X00B
EIPH	Interrupt High Priority 1	F7H	PT2H	PSPIH	PPWMH	PWDIH	-	-	PKBH	PI2H	XX00 0000B
P0DIDS	Port 0 Digital Input Disable	F6H	P0DIDS[7:0]								0000 0000B
SPDR	Serial Peripheral Data Register	F5H	SPDR[7:0]								XXXX XXXXB
SPSR	Serial Peripheral Status Register	F4H	SPIF	WCOL	SPIOVF	MODF	DISMODF	-	-	-	0000 0XXXB
SPCR	Serial Peripheral Control Register	F3H	SSOE	SPIEN	LSBFE	MSTR	CPOL	CPHA	SPR1	SPR0	0000 0100B
B	B register	F0H	(F7) B.7	(F6) B.6	(F5) B.5	(F4) B.4	(F3) B.3	(F2) B.2	(F1) B.1	(F0) B.0	0000 0000B
C2H	Input Capture 2 High	EEH	C2H[7:0]								0000 0000B
C2L	Input Capture 2 Low	EDH	C2L[7:0]								0000 0000B
KBLS1	Keyboard level select 1	ECH	KBLS1[7:0]								0000 0000B
KBLS0	Keyboard level select 0	EBH	KBLS0[7:0]								0000 0000B
KBIF	KBI Interrupt Flag	EAH	KBIF[7:0]								0000 0000B
KBIE	Keyboard Interrupt Enable	E9H	KBIE[7:0]								0000 0000B
EIE	Interrupt enable 1	E8H	(EF) ET2	(EE) ESPI	(ED) EPWM	(EC) EWDI	(E7)	(E8) ECPTF	(E9) EKB	(E8) EI2C	0000 0000B
C1H	Input Capture 1 High	E7H	C1H[7:0]								0000 0000B
C1L	Input Capture 1 Low	E6H	C1L[7:0]								0000 0000B
C0H	Input Capture 0 High	E5H	C0H[7:0]								0000 0000B
C0L	Input Capture 0 Low	E4H	C0L[7:0]								0000 0000B
ADCH	ADC converter result	E2H	ADC.9	ADC.8	ADC.7	ADC.6	ADC.5	ADC.4	ADC.3	ADC.2	XXXX XXXXB
ADCCON1	ADC control register1	E1H	ADCEN	-	-	-	-	-	RCCLK	ADC0SEL	0000 0000B
ACC	Accumulator	E0H	(E7) ACC.7	(E6) ACC.6	(E5) ACC.5	(E4) ACC.4	(E3) ACC.3	(E2) ACC.2	(E1) ACC.1	(E0) ACC.0	0000 0000B
PWMCON1	PWM control register 1	DFH	BKCH	BKPS	BPEN	BKEN	PWM3B	PWM2B	PWM1B	PWM0B	0000 0000B
PWM3L	PWM 3 low bits register	DEH	PWM3.7	PWM3.6	PWM3.5	PWM3.4	PWM3.3	PWM3.2	PWM3.1	PWM3.0	0000 0000B
PWM2L	PWM 2 low bits register	DDH	PWM2.7	PWM2.6	PWM2.5	PWM2.4	PWM2.3	PWM2.2	PWM2.1	PWM2.0	0000 0000B
PWMCON0	PWM control register 0	DCH	PWMRUN	load	CF	CLRPWM	PWM3I	PWM2I	PWM1I	PWM0I	0000 0000B
PWM1L	PWM 1 low bits register	DBH	PWM1.7	PWM1.6	PWM1.5	PWM1.4	PWM1.3	PWM1.2	PWM1.1	PWM1.0	0000 0000B
PWM0L	PWM 0 low bits register	DAH	PWM0.7	PWM0.6	PWM0.5	PWM0.4	PWM0.3	PWM0.2	PWM0.1	PWM0.0	0000 0000B
PWMPL	PWM counter low register	D9H	PWMP0.7	PWMP0.6	PWMP0.5	PWMP0.4	PWMP0.3	PWMP0.2	PWMP0.1	PWMP0.0	0000 0000B
WDCON0[4]	Watch-Dog control 0	D8H	(DF) WDTEN	(DE) WDCLR	(DD) WDTF	(DC) WIDPD	(DB) WDTRF	(DA) WPS2	(D9) WPS1	(D8) WPS0	Power-ON C000 0000B Watch reset C0UU 1UUUB Other reset C0UU UUUUB
PWMCON2	PWM control register 2	D7H	-	-	-	-	FP1	FP0	-	BKF	XXXX 00X0B
PWM3H	PWM 3 high bits register	D6H	-	-	-	-	-	-	PWM3.9	PWM3.8	XXXX XX00B
PWM2H	PWM 2 high bits register	D5H	-	-	-	-	-	-	PWM2.9	PWM2.8	XXXX XX00B
PWM1H	PWM 1 high bits register	D3H	-	-	-	-	-	-	PWM1.9	PWM1.8	XXXX XX00B
PWM0H	PWM 0 high bits register	D2H	-	-	-	-	-	-	PWM0.9	PWM0.8	XXXX XX00B

Table 7–2. N79E845 Series SFR Descriptions and Reset Values

Symbol	Definition	Address	MSB							LSB	Reset Value[1]
PWMPH	PWM counter high register	D1H	-	-	-	-	-	-	PWMP0.9	PWMP0.8	0000 0000B
PSW	Program status word	D0H	(D7) CY	(D6) AC	(D5) F0	(D4) RS1	(D3) RS0	(D2) OV	(D1) F1	(D0) P	0000 0000B
TH2	Timer 2 MSB	CDH	TH2[7:0]								0000 0000B
TL2	Timer 2 LSB	CCH	TL2[7:0]								0000 0000B
RCOMP2H	Timer 2 Reload MSB	CBH	RCOMP2H[7:0]								0000 0000B
RCOMP2L	Timer 2 Reload LSB	CAH	RCOMPL2[7:0]								0000 0000B
T2MOD	Timer 2 Mode	C9H	LDEN	T2DIV2	T2DIV1	T2DIV0	CAPCR	COMPCR	LDTS1	LDTS0	0000 0000B
T2CON	Timer 2 Control	C8H	(CF) TF2	-	-	-	-	(CA) TR2	-	(C8) CMP/RL2	0XXX X0X0B
TA	Timed Access Protection	C7H									0000 0000B
I2ADDR	I2C address	C1H	ADDR.6	ADDR.5	ADDR.4	ADDR.3	ADDR.2	ADDR.1	ADDR.0	GC	XXXX XXX0B
I2CON	I2C Control register	C0H	(C7) -	(C6) I2CEN	(C5) STA	(C4) STO	(C3) SI	(C2) AA	(C1) -	(C0) -	X000 00XXB
I2TOC	I2C Time-out Counter register	BFH	-	-	-	-	-	I2TOCEN	DIV	I2TOF	0000 0000B
I2CLK	I2C Clock Rate	BEH	I2CLK[7:0]								0000 0000B
I2STAT	I2C Status Register	BDH	I2STAT.4	I2STAT.3	I2STAT.2	I2STAT.1	I2STAT.0	0	0	0	1111 1000B
I2DAT	I2C Data Register	BCH	I2DAT[7:0]								XXXX XXXXB
SADEN	Slave address mask	B9H	SADEN[7:0]								0000 0000B
IP	Interrupt priority	B8H	(BF) PCAP	(BE) PADC	(BD) PBOD	(BC) PS	(BB) PT1	(BA) PX1	(B9) PT0	(B8) PX0	X000 0000B
IPH	Interrupt high priority	B7H	PCAPH	PADCH	PBODH	PSH	PT1H	PX1H	PT0H	PX0H	X000 0000B
P1M2	Port 1 output mode 2	B4H	P1M2[7:0]								0000 0000B
P1M1	Port 1 output mode 1	B3H	P1M1[7:0]								0000 0000B
P0M2	Port 0 output mode 2	B2H	P0M2[7:0]								0000 0000B
P0M1	Port 0 output mode 1	B1H	P0M1[7:0]								0000 0000b
P3	Port3	B0H	-	-	-	-	-	-	(B1) X1	(B0) X2 CLKOUT	XXXX XX00B
ISPCN	ISP Control Register	AFH	ISPA17	ISPA16	FOEN	FCEN	FCTRL3	FCTRL2	FCTRL1	FCTRL0	0011 0000B
ISPFD	ISP Flash Data Register	AEH	ISPFD[7:0], ISP Flash Data								0000 0000B
WDCON1[4]	Watch-Dog control1	ABH	-	-	-	-	-	-	-	EWRST	0000 0000B
SADDR	Slave address	A9H	SADDR[7:0]								00000000B
IE	Interrupt enable	A8H	(AF) EA	(AE) EADC	(AD) EBOD	(AC) ES	(AB) ET1	(AA) EX1	(A9) ET0	(A8) EX0	0000 0000B
ISPAH	ISP Flash Address High-byte	A7H	ISPAH[7:0]								0000 0000B
ISPAL	ISP Flash Address Low-byte	A6H	ISPAL[7:0]								0000 0000B
ISPTRG[4]	ISP Trigger Register	A4H	-	-	-	-	-	-	-	ISPGO	XXXX XXX0B
PMCR[2][4]	Power Monitor Control Register	A3H	BODEN	BOV	-	BORST	BOF	LPBOD	-	-	Power-on CCXC 10XXB BOR reset UUXU 10XXB Other reset UUXU 00XXB
AUXR1	AUX function register	A2H	SPI_Sel	UART_Sel	-	-	-	-	-	DPS	0000 0000B

Table 7–2. N79E845 Series SFR Descriptions and Reset Values

Symbol	Definition	Address	MSB							LSB	Reset Value[1]
CHPCON[4]	Chip Control	9FH	SWRST	ISPF (Read only)	LDUE	-	-	-	BS[3]	ISPEN	Power-ON 0000 00C0B Other reset 000X XUU0B
SHBDA[4]	High-byte Data Flash Start Address	9CH	SHBDA[7:0], SHBDA Initial by CHBDA								Power ON CCCC CCCCB Other Reset UUUU UUUUB
SBUF	Serial buffer	99H	SBUF.7	SBUF.6	SBUF.5	SBUF.4	SBUF.3	SBUF.2	SBUF.1	SBUF.0	0000 0000B
SCON	Serial control	98H	(9F) SM0/FE	(9E) SM1	(9D) SM2	(9C) REN	(9B) TB8	(9A) RB8	(99) TI	(98) RI	0000 0000B
P3M2	Port 3 output mode 2	97H	-	-	-	-	-	ENCLK	P3M2.1	P3M2.0	XXXXX000B
P3M1	Port 3 output mode 1	96H	P3S	-	P1S	P0S	T1OE	T0OE	P3M1.1	P3M1.0	00000000B
DIVM	CPU Clock Divide Register	95H	DIVM[7:0]								0000 0000B
CAPCON2	Input capture control 2	94H	-	-	ENF1	ENF0	-	-	-	-	0000 0000b
CAPCON1	Input capture control 1	93H	-	-	-	-	CAP1LS1	CAP1LS0	CAP0LS1	CAP0LS0	0000 0000b
CAPCON0	Input capture control 0	92H	-	-	CAPEN1	CAPEN0	-	-	CAPF1	CAPF0	0000 0000b
P1	Port 1	90H	(97) P17	(96) P16	-	(94) P14	(93) P13	(92) P12	(91) P11	(90) P10	1111 1111B
CKCON	Clock control	8EH	-	-	-	T1M	T0M	-	-	-	XXX0 0XXXB
TH1	Timer high 1	8DH	TH1[7:0]								0000 0000B
TH0	Timer high 0	8CH	TH0[7:0]								0000 0000B
TL1	Timer low 1	8BH	TL1[7:0]								0000 0000B
TL0	Timer low 0	8AH	TL0[7:0]								0000 0000B
TMOD	Timer mode	89H	GATE	C/T	M1	M0	GATE	C/T	M1	M0	0000 0000B
TCON	Timer control	88H	(8F) TF1	(8E) TR1	(8D) TF0	(8C) TR0	(8B) IE1	(8A) IT1	(89) IE0	(88) IT0	0000 0000B
PCON	Power control	87H	SMOD	SMOD0	-	POF	GF1	GF0	PD	IDL	Power-on 0001 0000B BOD Reset 00xu 0000B Other reset 000u 0000B
DPH	Data pointer high	83H	DPH[7:0]								0000 0000B
DPL	Data pointer low	82H	DPL[7:0]								0000 0000B
SP	Stack pointer	81H	SP[7:0]								0000 0111B
P0	Port 0	80H	(87) P07	(86) P06	(85) P05	(84) P04	(83) P03	(82) P02	(81) P01	(80) P00	1111 1111B

Note that bits marked in "-" must be kept in their own initial states. Users should never change their values.

Note:

[1.] ( ) item means the bit address in bit-addressable SFRs.

[2.] BODEN, BOD and BORST are initialized by CONFIG2 at power-on reset, and keep unchanged at any other resets. If BODEN=1, BOF will be automatically set by hardware at power-on reset, and keeps unchanged at any other resets.

[3.] Initialized by power-on reset. WDTEN=/CWDTEN; BS=/CBS;

[4.] With TA-Protection. (Time Access Protection)

[5.] Notation “C” means the bit is defined by config-bits; “U” means the bit is unchanged after any reset except power-on reset.

[6.] Reset value symbol description. 0: logic 0, 1: logic 1, U: unchanged, X:, C: initial by Config. T: initial by Trim bits.

8 General 8051 Core

A or ACC – Accumulator (bit-addressable)

7	6	5	4	3	2	1	0
ACC.7	ACC.6	ACC.5	ACC.4	ACC.3	ACC.2	ACC.1	ACC.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: E0H reset value: 0000 0000B

Bit	Name	Description
7:0	**ACC[7:0]**	**Accumulator.** The A or ACC register is the standard 8051 accumulator for arithmetic operation.

B – B Register (bit-addressable)

7	6	5	4	3	2	1	0
B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: F0H reset value: 0000 0000B

Bit	Name	Description
7:0	**B[7:0]**	**B register.** The B register is the other accumulator of the standard 8051. It is used mainly for MUL and DIV operations.

SP – Stack Pointer

7	6	5	4	3	2	1	0
SP[7:0]							
r/w							

Address: 81H reset value: 0000 0111B

Bit	Name	Description
7:0	**SP[7:0]**	**Stack pointer.** The Stack Pointer stores the scratch-pad RAM address where the stack begins. It is incremented before data is stored during PUSH or CALL instructions. Note that the default value of SP is 07H. It causes the stack to begin at location 08H.

DPL – Data Pointer Low Byte

7	6	5	4	3	2	1	0
DPL[7:0]							
r/w							

Address: 82H reset value: 0000 0000B

Bit	Name	Description
7:0	**DPL[7:0]**	**Data pointer low byte.** This is the low byte of the standard 8051 16-bit data pointer. DPL combined with DPH serve as a 16-bit data pointer DPTR to address non-scratch-pad memory or Program Memory.

DPH – Data Pointer High Byte

7	6	5	4	3	2	1	0
DPH[7:0]							
r/w							

Address: 83H reset value: 0000 0000B

Bit	Name	Description
7:0	**DPH[7:0]**	**Data pointer high byte.** This is the high byte of the standard 8051 16-bit data pointer. DPH combined with DPL serve as a 16-bit data pointer DPTR to address non-scratch-pad memory or Program Memory.

PSW – Program Status Word (bit-addressable)

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

Address: D0H reset value: 0000 0000B

Bit	Name	Description
7	**CY**	**Carry flag.** For a adding or subtracting operation, CY will be set when the previous operation resulted in a carry-out from or a borrow-in to the Most Significant bit, otherwise cleared. If the previous operation is MUL or DIV, CY is always 0. CY is affected by DA A instruction which indicates that if the original BCD sum is greater than 100. For a CJNE branch, CY will be set if the first unsigned integer value is less than the second one. Otherwise, CY will be cleared.
6	**AC**	**Auxiliary carry.** Set when the previous operation resulted in a carry-out from or a borrow-in to the 4th bit of the low order nibble, otherwise cleared.

Bit	Name	Description
5	**F0**	**User flag 0.** The general purpose flag that can be set or cleared by the user.
4	**RS1**	**Register Bank selecting bits.** These two bits select one of four banks in which R0~R7 locate. RS1 RS0 Register Bank RAM Address 0 0 0 00~07H 0 1 1 08~0FH 1 0 2 10~17H 1 1 3 18~1FH
3	**RS0**	
2	**OV**	**Overflow flag.** OV is used for a signed character operands. For a ADD or ADDC instruction, OV will be set if there is a carry out of bit 6 but not out of bit 7, or a carry out of bit 7 but not bit 6. Otherwise, OV is cleared. OV indicates a negative number produced as the sum of two positive operands or a positive sum from two negative operands. For a SUBB, OV is set if a borrow is needed into bit6 but not into bit 7, or into bit7 but not bit 6. Otherwise, OV is cleared. OV indicates a negative number produced when a negative value is subtracted from a positive value, or a positive result when a positive number is subtracted from a negative number. For a MUL, if the product is greater than 255 (00FFH), OV will be set. Otherwise, it is cleared. For a DIV, it is normally 0. However, if B had originally contained 00H, the values returned in A and B will be undefined. Meanwhile, the OV will be set.
1	**F1**	**User flag 1.** The general purpose flag that can be set or cleared by the user via software.
0	**P**	**Parity flag.** Set to 1 to indicate an odd number of ones in the accumulator. Cleared for an even number of ones. It performs even parity check.

Table 8–1. Instructions that affect flag settings

Instruction	CY	OV	AC	Instruction	CY	OV	AC
ADD	X[1]	X	X	CLR C	0		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C, bit	X		
MUL	0	X		ANL C, /bit	X		
DIV	0	X		ORL C, bit	X		
DA A	X			ORL C, /bit	X		

Instruction	CY	OV	AC	Instruction	CY	OV	AC
RRC A	X			MOV C, bit	X		
RLC A	X			CJNE	X		
SETB C	1						

[1] X indicates the modification depends on the result of the instruction.

PCON – Power Control

7	6	5	4	3	2	1	0
SMOD	SMOD0	-	POF	GF1	GF0	PD	IDL
r/w	r/w	-	r/w	r/w	r/w	r/w	r/w

Address: 87H reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
7	**SMOD**	**Serial Port Baud Rate Double Enable** 1: This bit doubles the serial port baud rate in mode 1, 2, and 3 when set to 1.
6	**SMOD0**	**Framing Error Detection Enable** 0: Framing Error Detection Disable. SCON.7 acts as per the standard 8052 function. 1: Framing Error Detection Enable, then and SCON.7 indicates a Frame Error and acts as the FE flag.
3	**GF1**	**General purpose flag 1.** The general purpose flag that can be set or cleared by the user.
2	**GF0**	**General purpose flag 0.** The general purpose flag that can be set or cleared by the user.

9 GENERAL PURPOSE IO (GPIO)

There are **four** I/O ports of **N79E845** series, which are port 0, port 1, port2 and port 3. If used on-chip RC oscillator and reset pin configurations, **N79E845** series can be supported up to **25** pins. All pins of I/O ports may be configured to one of four types by software as below table.

Table 9–1. Setting table for I/O Ports structure

PxM1.y	PxM2.y	Port Input/Output Mode
0	0	Quasi-bidirectional
0	1	Push-Pull
1	0	Input Only (High Impedance)
1	1	Open Drain

Note: P1.2 and P1.3 are no-effect with this table.

After reset, these pins are in quasi-bidirectional mode except P1.2 and P1.3 pins.

The P1.2 and P1.3 are dedicating open-drain pin for I2C interface after reset.

Each I/O port of **N79E845** series may be selected to use TTL level inputs or Schmitt inputs by P(n)S bit on P3M1 register; where n is 0, 1 or 2. When P(n)S is set to 1, Ports are selected Schmitt trigger inputs on Port(n).

The P3.0 (XTAL2) can be configured as clock output when used on-chip RC or external Oscillator is clock source, and the frequency of clock output is divided by 4 on on-chip RC clock or external Oscillator.

9.1 Quasi-Bidirectional Output Configuration

The default port configuration for standard **N79E845** series I/O ports is the “quasi-bidirectional” mode that is common on the 80C51 and most of its derivatives. This type rules as both input and output. When the port outputs a logic high, it is weakly driven, allowing an external device to pull the pin low. When the pin is pulled low, it is driven strongly and able to sink a large current. In the quasi bi-directional I/O structure, there are three pull-up transistors. Each of them serves different purposes. One of these pull-ups, called the “very weak” pull-up, is turned on whenever the port latch contains a logic 1. The “very weak” pull-up sources a very small current that will pull the pin high if it is left floating.

A second pull-up, called the “weak” pull-up, is turned on when the outside port pin itself is at a logic 1 level. This pull-up provides the primary source current for a quasi bi-directional pin that is outputting a 1. If a pin that has a logic 1 on it is pulled low by an external device, the “weak” pull-up turns off, and only the “very weak” pull-up remains on. In order to pull the pin low under these conditions, the external device has to sink enough

current (larger than I_{TL}) to overcome the “weak” pull-up and make the voltage on the port pin below its input threshold (lower than V_{IL}).

The third pull-up is the “strong” pull-up. This pull-up is used to speed up low-to-high transitions on a quasi bi-directional port pin when the port latch changes from a logic 0 to a logic 1. When this occurs, the strong pull-up turns on for two-peripheral-clock time in order to pull the port pin high quickly. Then it turns off and “weak: pull-up continues remaining the port pin high. The quasi bi-directional port structure is shown as below.

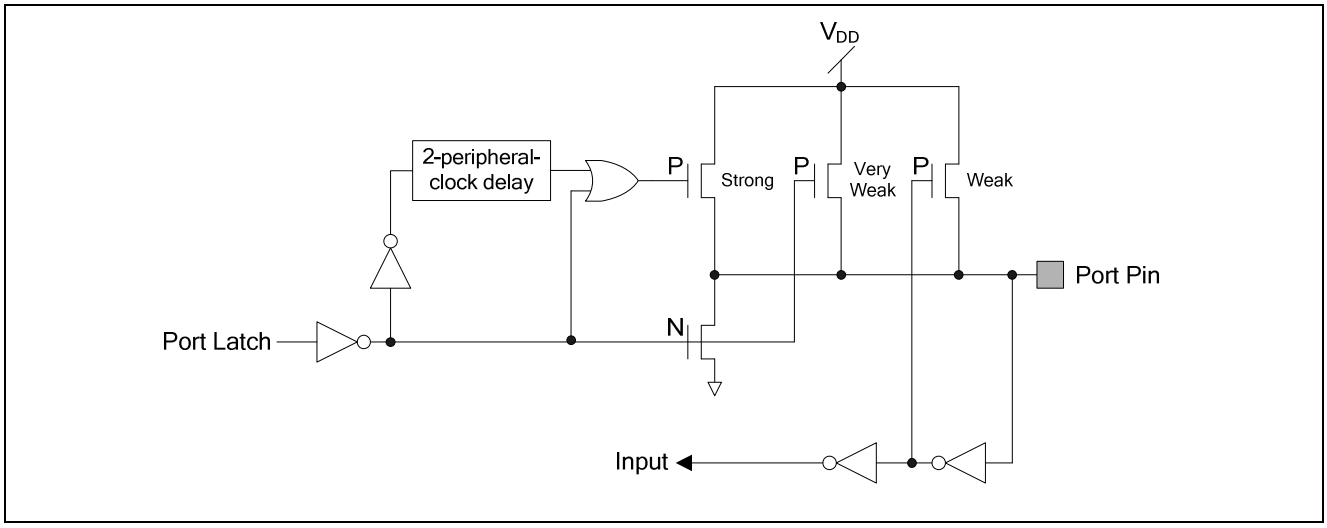


Figure 9–1. Quasi Bi-direction I/O Structure

9.1.1 Read-Modify-Write

In standard 8051 instruction set, one kind of instructions, read-modify-write instructions, should be specially taken care of. Instead of the normal instructions, the read-modify-write instructions read the internal port latch (Px in SFRs) rather than the external port pin state. This kind of instructions read the port SFR value, modify it and write back to the port SFR. Read-modify-write instructions are listed as follows.

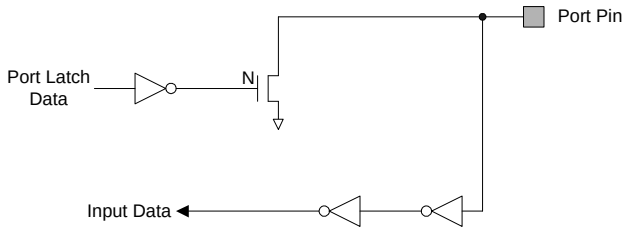
Instruction	Description
ANL	Logical AND. (ANL Px,A and ANL Px,direct)
ORL	Logical OR. (ORL Px,A and ORL Px,direct)
XRL	Logical exclusive OR. (XRL Px,A and XRL Px,direct)
JBC	Jump if bit = 1 and clear it. (JBC Px.y,LABEL)
CPL	Complement bit. (CPL Px.y)
INC	Increment. (INC Px)
DEC	Decrement. (DEC Px)
DJNZ	Decrement and jump if not zero. (DJNZ Px,LABEL)

MOV	Px.y,C	Move carry bit to Px.y.
CLR	Px.y	Clear bit Px.y.
SETB	Px.y	Set bit Px.y.

The last three seems not obviously read-modify-write instructions but actually they are. They read the entire port latch value, modify the changed bit, then write the new value back to the port latch.

9.2 Open Drain Output Configuration

The open drain output configuration turns off all pull-ups and only drives the pull-down transistor of the port driver when the port latch contains a logic 0. To be used as a logic output, a port configured in this manner must have an external pull-up, typically a resistor tied to V_{DD}. The pull-down for this mode is the same as for the quasi-bidirectional mode. The open drain port configuration is shown as below.



Open Drain Output

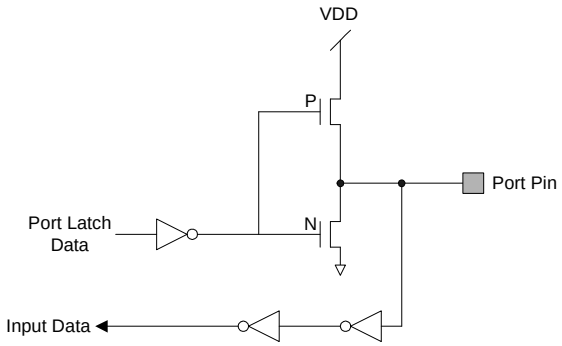
Figure 9-2 Open Drain Output

9.3 Push-Pull Output Configuration

The push-pull output configuration has the same pull-down structure as both the open drain and the quasi-bidirectional output modes, but provides a continuous strong pull-up when the port latch contains a logic 1. The push-pull mode may be used when more source current is needed from a port output. The push-pull port configuration is shown in Figure 0-3. The two port pins that cannot be configured are P1.2 (SCL) and P1.3 (SDA). The port pins P1.2 and P1.3 are permanently configured as open drain outputs. They may be used as inputs by writing ones to their respective port latches. Additionally, port pins P3.0 and P3.1 are disabled for both input and output if one of the crystal oscillator options is chosen. Those options are described in the Oscillator section.

When port pins are driven high at reset, they are in quasi-bidirectional mode and therefore do not source large amounts of current. Every output on the **N79E845** series may potentially be used as a 40 mA sink LED drive output. However, there is a maximum total output current for all ports which must not be exceeded.

All ports pins of the **N79E845** series have slew rate controlled outputs. This is to limit noise generated by quickly switching output signals. The slew rate is factory set to approximately 10 ns rise and fall times. The bits in the P3M1 register that are not used to control configuration of P3.1 and P3.0 are used for other purposes. These bits can enable Schmitt trigger inputs on each I/O port, enable toggle outputs from Timer 0 and Timer 1, and enable a clock output if either the internal RC oscillator or external clock input is being used. The last two functions are described in the Timer/Counters and Oscillator sections respectively. Each I/O port of the **N79E845** series may be selected to use TTL level inputs or Schmitt inputs with hysteresis. A single configuration bit determines this selection for the entire port. The P1.2 and P1.3 can be selected TTL level input or Schmitt trigger input.



Push-Pull Output

Figure 9-3 Push-Pull Output

9.4 Input Only Configuration

By configure this mode; the ports are only input mode. After config this mode, the pin will be Hi-Impendence.

P0 – Port 0 (bit-addressable)

7	6	5	4	3	2	1	0
P07	P06	P05	P04	P03	P02	P01	P00
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: 80H reset value: 1111 1111B

Bit	Name	Description
7:0	**P0[7:0]**	**Port 0.** Port 1 is an 8-bit quasi bi-directional I/O port.

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 pin for I2C interface after reset.**

P3 – Port 3 (bit-addressable)

7	6	5	4	3	2	1	0
-	-	-	-	-	-	P31	P30
-	-	-	-	-	-	r/w	r/w

Address: B0H reset value: 1111 1111B

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.

P0M1 – Port 0 Output Mode1

7	6	5	4	3	2	1	0
P0M1.7	P0M1.6	P0M1.5	P0M1.4	P0M1.3	P0M1.2	P0M1.1	P0M1.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: B1H reset value: 0000 0000B

P0M2 – Port 0 Output Mode2

7	6	5	4	3	2	1	0
P0M2.7	P0M2.6	P0M2.5	P0M2.4	P0M2.3	P0M2.2	P0M2.1	P0M2.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: B2H reset value: 0000 0000B

P1M1 – Port 1 Output Mode1

7	6	5	4	3	2	1	0
P1M1.7	P1M1.6	P1M1.5	P1M1.4	P1M1.3	P1M1.2	P1M1.1	P1M1.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: B3H reset value: 0000 0000B

P1M2 – Port 1 Output Mode2

7	6	5	4	3	2	1	0
P1M2.7	P1M2.6	P1M2.5	P1M2.4	P1M2.3	P1M2.2	P1M2.1	P1M2.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: B4H reset value: 0000 0000B

Port Output Configuration Settings:

PxM1.y	PxM2.y	Port Input/Output Mode
0	0	Quasi-bidirectional
0	1	Push-Pull
1	0	Input Only (High Impedance)
1	1	Open Drain

P3M1 – Port3 Output Mode1

7	6	5	4	3	2	1	0
P3S	-	P1S	P0S	T1OE	T0OE	P3M1.1	P3M1.0
r/w	-	r/w	r/w	r/w	r/w	r/w	r/w

Address: 96H reset value: 0000 0000B

Bit	Name	Description
7	**P3S**	Enables Schmitt trigger inputs on Port 3.
6	**-**	Reserved.
5	**P1S**	Enables Schmitt trigger inputs on Port 1.

Bit	Name	Description
4	**P0S**	Enables Schmitt trigger inputs on Port 0.
3	**T1OE**	P0.7 pin is toggled whenever Timer 1 overflows. The output frequency is therefore one half of the Timer 1 overflow rate.
2	**T0OE**	P1.2 pin is toggled whenever Timer 0 overflows. The output frequency is therefore one half of the Timer 1 overflow rate.
1	**P3M1.1**	To control the output configuration of P3.1.
0	**P3M1.0**	To 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**	Enabled clock output to XTAL2 pin (P3.0) If The clock from internal RC, the frequency of P3.0 is internal RC/4 (22.1184MHz/4).
1	**P3M2.1**	Reference P1M2 SFR.
0	**P3M2.0**	

P0DIDS – Port0 Digital Input Disable

7	6	5	4	3	2	1	0
P0DIDS.7	P0DIDS.6	P0DIDS.5	P0DIDS.4	P0DIDS.3	P0DIDS.2	P0DIDS.1	P0DIDS.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: F6H reset value: 0000 0000B

Bit	Name	Description
7:0	**P0DIDS.x**	1: Disable digital function for each Port0. 0: Enable digital function for each Port0.

10 Timers/Counters (Timer)

There are three 16-bit programmable timers/counters of **N79E845** series.

10.1 Timer/Counters 0 and 1

Timer/Counter 0 and 1 on **N79E845** series are two 16-bit Timer/Counters. Each of them has two 8 bit registers which form the 16 bit counting register. For Timer/Counter 0 they are TH0, the upper 8 bits register, and TL0, the lower 8 bit register. Similarly Timer/Counter 1 has two 8 bit registers, TH1 and TL1. TCON and TMOD can configure modes of Timer/Counter 0 and 1.

They have additional timer 0 or timer 1 overflow toggle output enable feature as compare to conventional timer/counters. This timer overflow toggle output can be configured to automatically toggle T0 or T1 pin output whenever a timer overflow occurs.

When configured as a “Timer”, the timer counts clock cycles. The timer clock can be programmed to be thought of as 1/12 of the system clock or 1/4 of the system clock. In the “Counter” mode, the register is incremented on the falling edge of the external input pin, T0 in case of Timer 0, and T1 for Timer 1. The T0 and T1 inputs are sampled in every machine-cycle at C4. If the sampled value is high in one machine-cycle and low in the next, then a valid high to low transition on the pin is recognized and the count register is incremented. Since it takes two machine-cycles to recognize a negative transition on the pin, the maximum rate at which counting will take place is 1/24 of the master clock frequency. In either the “Timer” or “Counter” mode, the count register will be updated at C3. Therefore, in the “Timer” mode, the recognized negative transition on pin T0 and T1 can cause the count register value to be updated only in the machine-cycle following the one in which the negative edge was detected.

The “Timer” or “Counter” function is selected by the “$C/\overline{T}$” bit in the TMOD Special Function Register. Each Timer/Counter has one selection bit for its own; bit 2 of TMOD selects the function for Timer/Counter 0 and bit 6 of TMOD selects the function for Timer/Counter 1. In addition each Timer/Counter can be set to operate in any one of four possible modes. The mode selection is done by bits M0 and M1 in the TMOD SFR.

The **N79E845** series can operate like the standard 8051/52 family, counting at the rate of 1/12 of the clock speed, or in turbo mode, counting at the rate of 1/4 clock speed. The speed is controlled by the T0M and T1M bits in CKCON, and the default value is zero, which uses the standard 8051/52 speed.

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 select:** 0: Timer 1 uses a divide by 12 clocks. 1: Timer 1 uses a divide by 4 clocks.
3	**T0M**	**Timer 0 clock select:** 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 $\overline{INT1}$ logic level. 1 = Timer 1 will clock only when TR1 = 1 and $\overline{INT1}$ is logic 1.
6	C/$\overline{T}$	**Timer 1 Counter/Timer select.** 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 select.** **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
4	**M0**	
3	**GATE**	**Timer 0 gate control.** 0 = Timer 0 will clock when TR0 = 1 regardless of $\overline{INT0}$ logic level. 1 = Timer 0 will clock only when TR0 = 0 and $\overline{INT0}$ is logic 1.
2	C/$\overline{T}$	**Timer 0 Counter/Timer select.** 0 = Timer 0 is incremented by internal peripheral clocks. 1 = Timer 0 is incremented by the falling edge of the external pin T0.

Bit	Name	Description
1	**M1**	**Timer 0 mode select.**
0	**M0**	**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.

TL0 – Timer 0 Low Byte

7	6	5	4	3	2	1	0
TL0[7:0]							
r/w							

Address: 8AH reset value: 0000 0000B

Bit	Name	Description
7:0	**TL0[7:0]**	**Timer 0 low byte.** The TL0 register is the low byte of the 16-bit Timer 0.

TH0 – Timer 0 High Byte

7	6	5	4	3	2	1	0
TH0[7:0]							
r/w							

Address: 8CH reset value: 0000 0000B

Bit	Name	Description
7:0	**TH0[7:0]**	**Timer 0 high byte.** The TH0 register is the high byte of the 16-bit Timer 0.

TL1 – Timer 1 Low Byte

7	6	5	4	3	2	1	0
TL1[7:0]							
r/w							

Address: 8BH reset value: 0000 0000B

Bit	Name	Description
7:0	**TL1[7:0]**	**Timer 1 low byte.** The TL1 register is the low byte of the 16-bit Timer 1.

TH1 – Timer 1 High Byte

7	6	5	4	3	2	1	0
TH1[7:0]							
r/w							

Address: 8DH reset value: 0000 0000B

Bit	Name	Description
7:0	**TH1[7:0]**	**Timer 1 high byte.** The TH1 register is the high byte of the 16-bit Timer 1.

10.1.1 Mode 0 (13-bit Timer)

In Mode 0, the timer/counters act as a 8-bit counter with a 5-bit, divide by 32 pre-scale. In this mode we have a 13-bit timer/counter. The 13-bit counter consists of 8 bits of THx and 5 lower bits of TLx. The upper 3 bits of TLx are ignored.

The negative edge of the clock is increments count in the TLx register. When the fifth bit in TLx moves from 1 to 0, then the count in the THx register is incremented. When the count in THx moves from FFh to 00h, then the overflow flag TFx in TCON SFR is set. The counted input is enabled only if TRx is set and either GATE = 0 or $\overline{INTx}$ = 1. When $C/\overline{T}$ is set to 0, then it will count clock cycles, and if $C/\overline{T}$ is set to 1, then it will count 1 to 0 transitions on T0 (P1.2) for timer 0 and T1 (P0.7) for timer 1. When the 13-bit count reaches 1FFFh the next count will cause it to roll-over to 0000h. The timer overflow flag TFx of the relevant timer is set and if enabled an interrupts will occur.

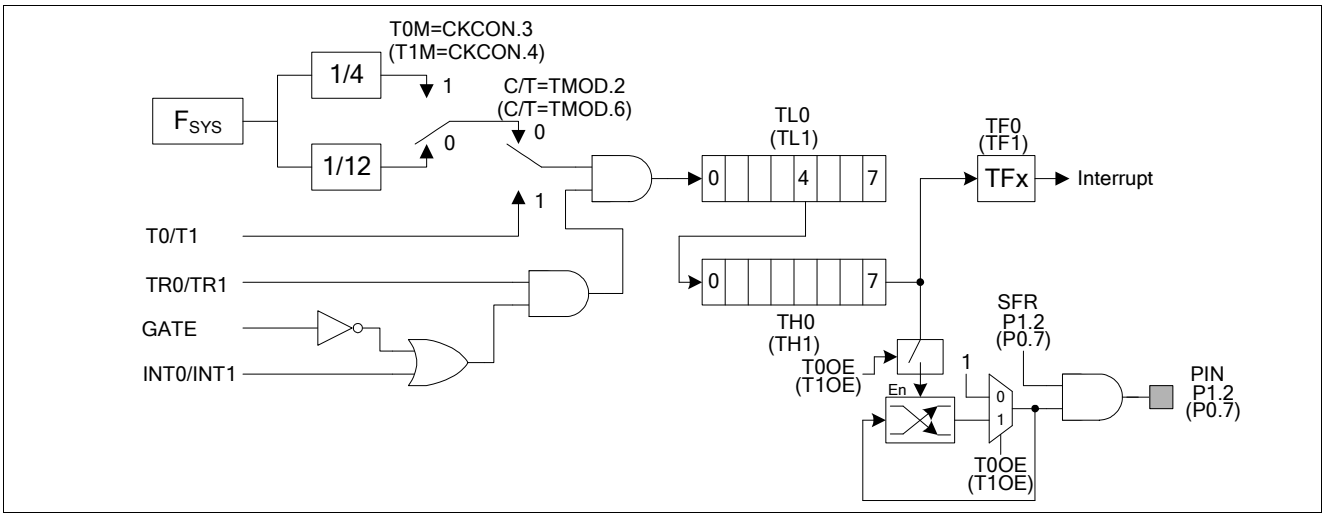


Figure 10–1. Timer/Counters 0 and 1 in Mode 0

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. Roll-over 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.

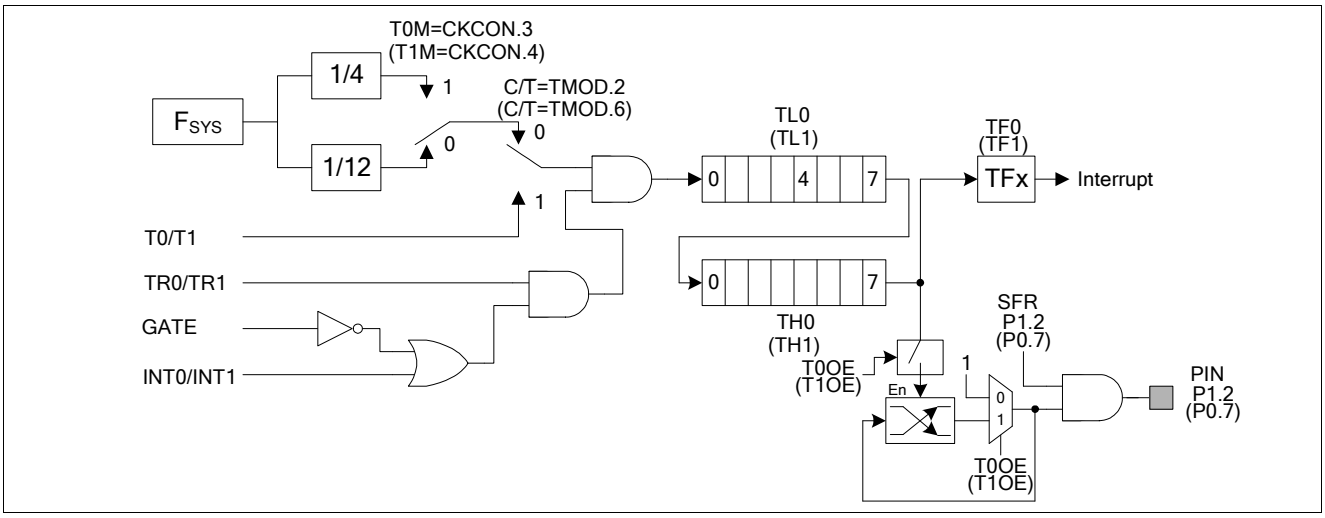


Figure 10–2. Timer/Counters 0 and 1 in Mode 1

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.

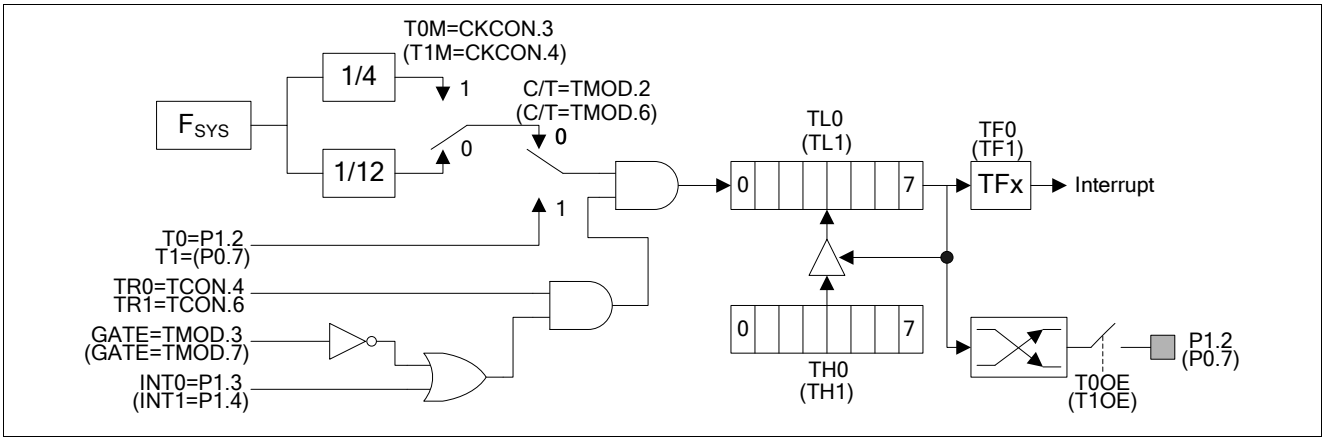


Figure 10–3. Timer/Counter 0 and 1 in Mode 2

10.1.4 Mode 3 (Two Separate 8-bit Timers)

Mode 3 has different operating methods for the two timer/counters. For timer/counter 1, mode 3 simply freezes the counter. Timer/Counter 0, however, configures TL0 and TH0 as two separate 8 bit count registers in this mode. The logic for this mode is shown in the figure. TL0 uses the Timer/Counter 0 control bits $C/\overline{T}$, GATE, TR0, $\overline{INT0}$ and TF0. The TL0 can be used to count clock cycles (clock/12 or clock/4) or 1-to-0 transitions on pin T0 as determined by C/T (TMOD.2). TH0 is forced as a clock cycle counter (clock/12 or clock/4) and takes over the use of TR1 and TF1 from Timer/Counter 1. Mode 3 is used in cases where an extra 8 bit timer is needed. With Timer 0 in Mode 3, Timer 1 can still be used in Modes 0, 1 and 2, but its flexibility is somewhat limited. While its basic functionality is maintained, it no longer has control over its overflow flag TF1 and the enable bit TR1. Timer 1 can still be used as a timer/counter and retains the use of GATE and INT1 pin. In this condition it can be turned on and off by switching it out of and into its own Mode 3. It can also be used as a baud rate generator for the serial port.

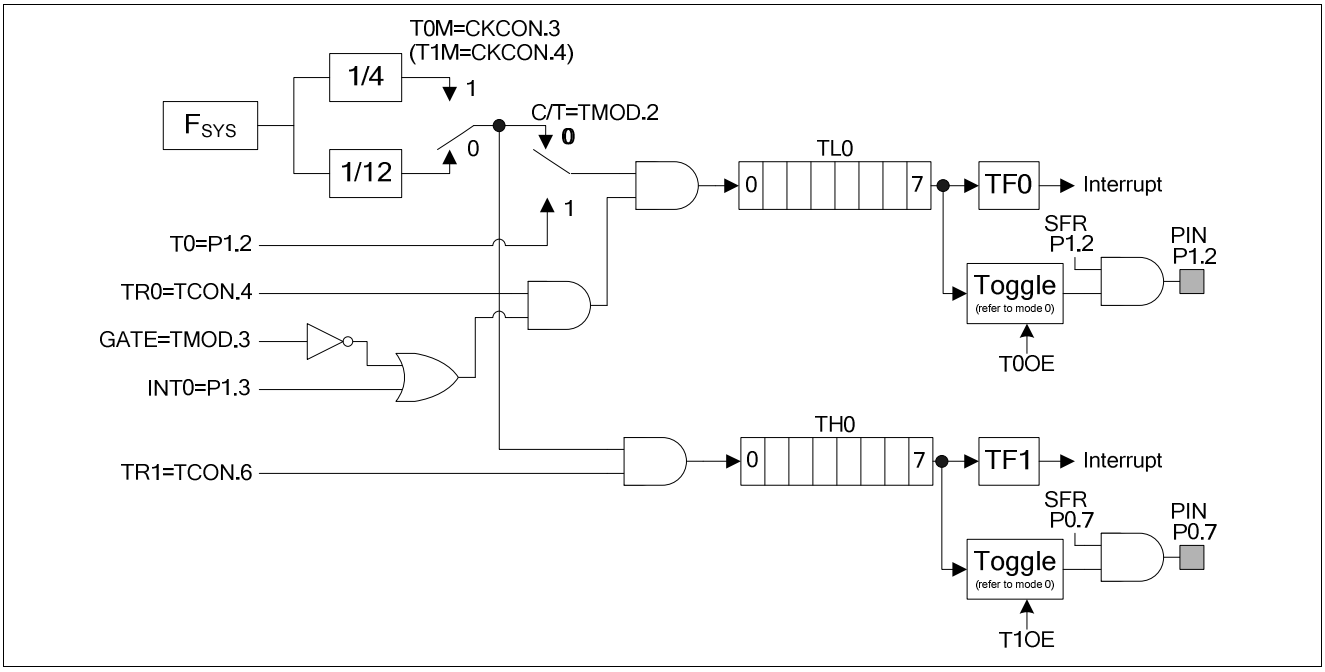


Figure 10–4. Timer/Counter 0 in Mode 3

10.2 Timer/Counter 2

Timer 2 is a 16-bit up counter cascaded with TH2, the upper 8 bits register, and TL2, the lower 8 bit register. Equipped with RCOMP2H and RCOMP2L, Timer 2 can operate under compare mode and auto-reload mode. The additional 3-channel input capture module makes Timer 2 detect and measure the width or period of input pulses. The results of 3 input captures are stores in C0H and C0L, C1H and C1L, C2H and C2L individually. The clock source of Timer 2 is from the system clock pre-scaled by a clock divider with 8 different scales for wide field application. The clock is enabled when TR2 (T2CON.2) is a 1, and disabled when TR2 is a 0. The following registers are related to Timer 2 function.

T2CON – Timer 2 Control (bit-addressable)

7	6	5	4	3	2	1	0
TF2	-	-	-	-	TR2	-	$CP/\overline{RL2}$
r/w	-	-	-	-	r/w	-	r/w

Address: C8H reset value: 0000 0000b

Bit	Name	Description
7	TF2	**Timer 2 overflow flag.** This bit is set when Timer 2 overflows or a compare match occurs. If the Timer 2 interrupt and the global interrupt are enable, setting this bit will make CPU execute Timer 2 interrupt service routine. This bit is not automatically cleared via hardware and must be cleared via software.
6:3	-	**Reserved.**
2	TR2	**Timer 2 run control.** 0 = Timer 2 is halted. Clearing this bit will halt Timer 2 and the current count will be preserved in TH2 and TL2. 1 = Timer 2 is enabled.
1	-	**Reserved.**
0	$CP/\overline{RL2}$	**Timer 2 Capture or Reload select.** This bit selects whether Timer 2 functions in compare or auto-reload mode. 0 = Auto-reload on Timer 2 overflow or any input capture event. 1 = Compare mode of Timer 2.

T2MOD – Timer 2 Mode

7	6	5	4	3	2	1	0
LDEN	T2DIV2	T2DIV1	T2DIV0	CAPCR	COMPCR	LDTS[1:0]	
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: C9H reset value: 0000 0000b

Bit	Name	Description
7	LDEN	**Enable Auto-reload.** 0 = Disable reloading RCOMP2H and RCOMP2L to TH2 and TL2 on Timer 2 overflow or any input capture event. 1 = Enable reloading RCOMP2H and RCOMP2L to TH2 and TL2 on Timer 2 overflow or any input capture event.

Bit	Name	Description
6:4	T2DIV[2:0]	**Timer 2 clock divider.** 000 = Timer 2 clock divider is 1/4. 001 = Timer 2 clock divider is 1/8. 010 = Timer 2 clock divider is 1/16. 011 = Timer 2 clock divider is 1/32. 100 = Timer 2 clock divider is 1/64. 101 = Timer 2 clock divider is 1/128. 110 = Timer 2 clock divider is 1/256. 111 = Timer 2 clock divider is 1/512.
3	CAPCR	**Capture auto-clear** This bit enables auto-clear Timer 2 value in TH2 and TL2 when a determined input capture event occurs. 0 = Timer 2 continues counting when a capture event occurs. 1 = Timer 2 value is auto-cleared as 0000H when a capture event occurs.
2	COMPCR	**Compare match auto-clear.** This bit enables auto-clear Timer 2 value in TH2 and TL2 when a compare match occurs. 0 = Timer 2 continues counting when a compare match occurs. 1 = Timer 2 value is auto-cleared as 0000H when a compare match occurs.
1:0	LDTS[1:0]	**Auto-reload trigger select.** These bits select the reload trigger event. 00 = Reload when Timer 2 overflows. 01 = Reload when input capture 0 event occurs. 10 = Reload when input capture 1 event occurs.

RCOMP2L – Timer 2 Reload/Compare Low Byte

7	6	5	4	3	2	1	0
RCOMP2L[7:0]							
r/w							

Address: CAH reset value: 0000 0000b

Bit	Name	Description
7:0	RCOMP2L[7:0]	**Timer 2 reload/compare low byte.** This register stores the low byte of compare value when Timer 2 is configured in compare mode. And it holds the low byte of the reload value when auto-reload mode.

RCOMP2H – Timer 2 Reload/Compare High Byte

7	6	5	4	3	2	1	0
RCOMP2H[7:0]							
r/w							

Address: CBH reset value: 0000 0000b

Bit	Name	Description
7:0	RCOMP2H[7:0]	**Timer 2 reload/compare high byte.** This register stores the high byte of compare value when Timer 2 is configured in compare mode. And it holds the high byte of the reload value when auto-reload mode.

TL2 – Timer 2 Low Byte

7	6	5	4	3	2	1	0
TL2[7:0]							
r/w							

Address: CCH reset value: 0000 0000b

Bit	Name	Description
7:0	TL2[7:0]	**Timer 2 low byte.** The TL2 register is the low byte of the 16-bit Timer 2.

TH2 – Timer 2 High Byte

7	6	5	4	3	2	1	0
TH2[7:0]							
r/w							

Address: CDH reset value: 0000 0000b

Bit	Name	Description
7:0	TH2[7:0]	**Timer 2 high byte.** The TH2 register is the high byte of the 16-bit Timer 2.

Timer/Counter 2 provides three operating mode which can be selected by control bits in T2CON and T2MOD as shown in table below. Note that the TH2 and TL2 are accessed separately. It is strongly recommended that the user stop Timer 2 temporally for a reading from or writing to TH2 and TL2. The free-running reading or writing may cause unpredictable situation.

Table 10–1. Timer 2 Operating Modes

Timer 2 Mode	CP/$\overline{RL2}$ (T2CON.0)	LDEN (T2MOD.7)
Input capture	0	0
Auto-reload	0	1
Compare	1	X

10.2.1 Input Capture Mode

The input capture module with Timer 2 implements the input capture mode. Timer 2 should be configured by clearing $CP/\overline{RL2}$ and LDEN bit to enter into input capture mode. The input capture module is configured through CAPCON0~2 registers. The input capture module supports 3-channel inputs (IC0, and IC1 pins) that share I/O pin P1.2, P0.7. Each input channel consists its own Schmitt trigger input. The noise filter for each channel is enabled via setting ENF0~1 (CAPCON2[5:4]). It filters input glitches smaller than 4 CPU clocks. Input capture 0~1 have independent edge detector but share with unique Timer 2. The trigger edge is also configured individually by setting CAPCON1. It supports positive edge capture, negative edge capture, or both edge capture. Each input capture channel has its own enabling bit CAPEN0~1 (CAPCON0[5:4]).

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 be also generated if ECPTF (EIE.7) 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 must 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.

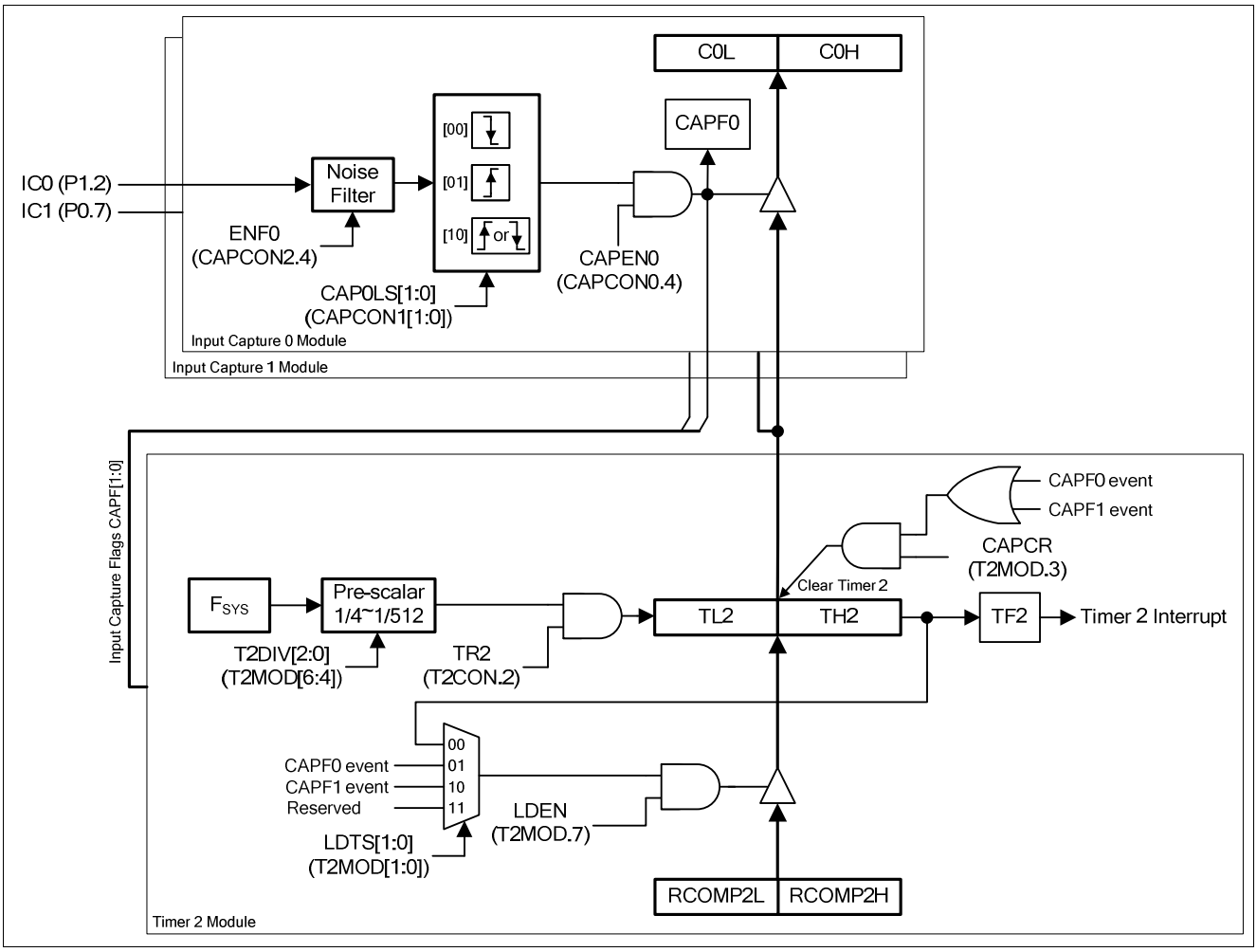


Figure 10–5. Timer 2 Input Capture and Auto-reload Mode Function Block

CAPCON0 – Input Capture Control 0

7	6	5	4	3	2	1	0
-	-	CAPEN1	CAPEN0	-	-	CAPF1	CAPF0
-	-	r/w	r/w	-	-	r/w	r/w

Address: 92H reset value: 0000 0000b

Bit	Name	Description
7	-	**Reserved.**
6	-	**Reserved.**
5	CAPEN1	**Input capture 1 enable.** 0 = Disable input capture channel 1. 1 = Enable input capture channel 1.
4	CAPEN0	**Input capture 0 enable.** 0 = Disable input capture channel 0. 1 = Enable input capture channel 0.

Bit	Name	Description
3	-	**Reserved.**
2	-	**Reserved.**
1	CAPF1	**Input capture 1 flag.** This bit is set by hardware if the determined edge of input capture 1 occurs. This bit must cleared by software.
0	CAPF0	**Input capture 0 flag.** This bit is set by hardware if the determined edge of input capture 0 occurs. This bit must cleared by software.

CAPCON1 – Input Capture Control 1

7	6	5	4	3	2	1	0
-	-	-	-	CAP1LS[1:0]		CAP0LS[1:0]	
-	-	-	-	r/w	r/w	r/w	r/w

Address: 92H reset value: 0000 0000b

Bit	Name	Description
7:6	-	**Reserved.**
5:4	-	**Reserved.**
3:2	CAP1LS[1:0]	**Input capture 1 level select.** 00 = Falling edge. 01 = Rising edge. 10 = Either Rising or falling edge. 11 = Reserved.
1:0	CAP0LS[1:0]	**Input capture 0 level select.** 00 = Falling edge. 01 = Rising edge. 10 = Either Rising or falling edge. 11 = Reserved.

CAPCON2 – Input Capture Control 2

7	6	5	4	3	2	1	0
-	-	ENF1	ENF0	-	-	-	-
-	-	r/w	r/w	-	-	-	-

Address: 92H reset value: 0000 0000b

Bit	Name	Description
7	-	**Reserved.**
6	-	**Reserved.**
5	ENF1	**Enable noise filer on input capture 1.** 0 = Disable noise filter on input capture channel 1. 1 = Enable noise filter on input capture channel 1.
4	ENF0	**Enable noise filer on input capture 0.** 0 = Disable noise filter on input capture channel 0. 1 = Enable noise filter on input capture channel 0.
3:0	-	**Reserved.**

10.2.2 Auto-reload Mode

Timer 2 can be configured as auto-reload mode by clearing $CP/\overline{RL2}$ and setting LDEN bit. In this mode RCOMP2H and RCOMP2L registers stores the reload value. The contents in RCOMP2H and RCOM3L transfer into TH2 and TL2 once the auto-reload event occurs. The event can be the Timer 2 overflow or one of the triggering event on any of enabled input capture channel depending on the LDTS[1:0] (T2MOD[1:0]) selection.

Note that once CAPCR (T2MOD.3) is set, an input capture event only clears TH2 and TL2 without reloading RCOMP2H and RCOMP2L contents.

10.2.3 Compare Mode

Timer 2 can also be configured simply as the compare mode by setting $CP/\overline{RL2}$. In this mode RCOMP2H and RCOMP2L registers serve as the compare value registers. As Timer 2 up counting, TH2 and TL2 match RCOMP2H and RCOMP2L, TF3 (T2CON.7) will be set by hardware to indicate a compare match event.

Setting COMPCR (T2MOD.2) makes the hardware to clear Timer 2 counter as 0000H automatically after a compare match has occurred.

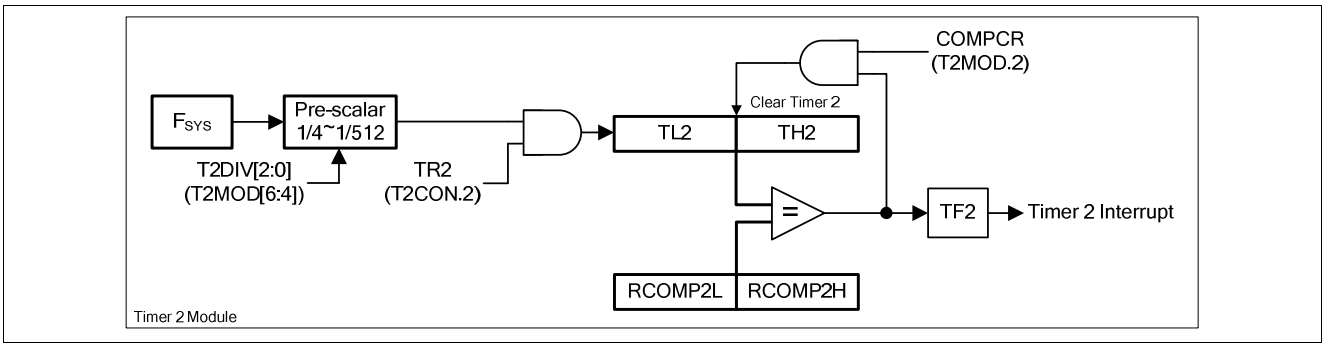


Figure 10–6. Timer 2 Compare Mode Function Block

11 Watchdog Timer (WDT)

N79E845 series provide one Watchdog Timer to serve as a system monitor, which improve the reliability of the system. Watchdog Timer is useful for systems that are susceptible to noise, power glitches, or electrostatic discharge. The periodic interrupt of Watchdog Timer can also serve as an event timer or a durational system supervisor in a monitoring system which generally operates in Idle or Power Down mode. The Watchdog Timer is basic a setting of divider that divides a internal low speed clock source. The divider output is selectable and determines the time-out interval. When the time-out interval is fulfilled, it will wake the system up from Idle or Power Down mode and an interrupt event will occur. If Watchdog Timer reset is enabled, a system reset will occur after a period of delay if without any software response.

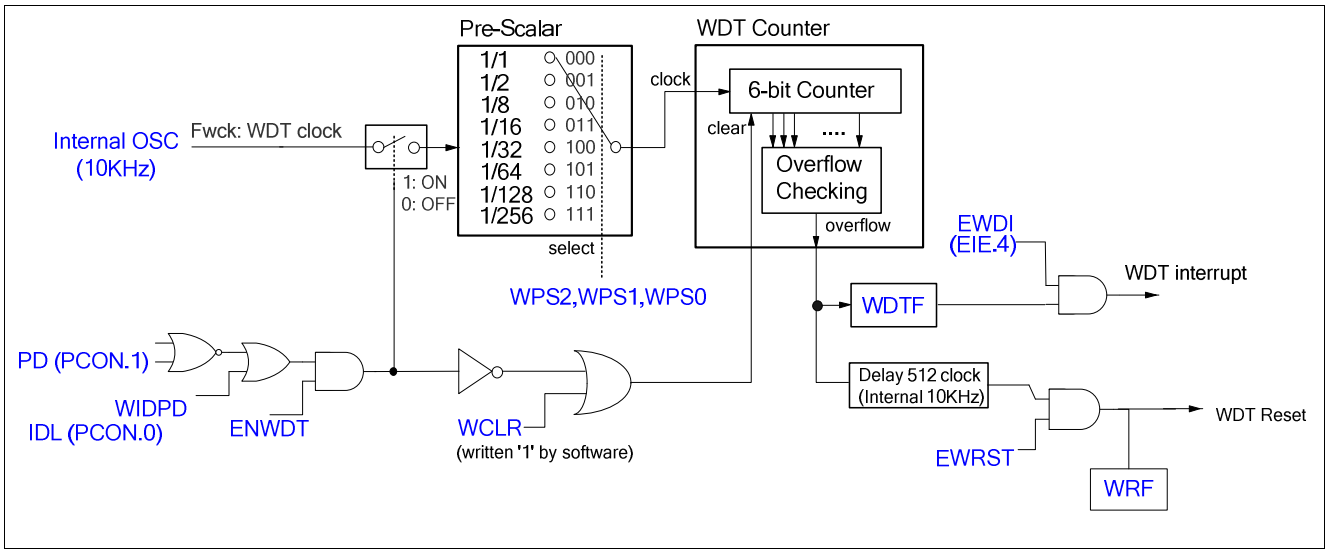


Figure 11-1 Watchdog Timer

11.1 Function Description

The Watchdog Timer should first be reset 00H by using WDCLR(WDCON0.6) to ensure that the timer starts from a known state. After disable Watchdog Timer through clearing WDTEN (WDCON.7) will also clear this counter. The WDCLR bit is used to reset the Watchdog Timer. This bit is self-cleared thus the user doesn't need to clear it. After writing a 1 to WDCLR, the hardware will automatically clear it. After WDTEN set as 1, the Watchdog Timer starts counting. The time-out interval is selected by the three bits WPS2, WPS1, and WPS0 (WDCON0[2:0]). When the selected time-out occurs, the Watchdog Timer will set the interrupt flag WDTF (WDCON0.5). The Watchdog Timer interrupt enable bit locates at EIE.4 register. If Watchdog Timer reset is enabled by writing a logic 1 to EWRST (WDCON1.0) bit. An additional 512 clocks of the low speed internal RC delays to expect a counter clearing by setting WDCLR. If no WDCLR setting during this 512-clock period, a reset will happen. Once a reset due to Watchdog Timer occurs, the Watchdog Timer reset flag WDTRF

(WDCON0.3) will be set. This bit keeps unchanged after any reset other than a power-on reset. The user may clear WDTRF via software. In general, software should restart the counter to put it into a known state by setting WDCLR. The Watchdog Timer also provides an WIDPD bit (WDCON.4) to allow the Watchdog Timer continuing running after the system enters into Idle or Power Down operating mode.

WDT counter should be specially taken care. The hardware automatically clears WDT counter after entering into or being woken-up from Idle or Power Down mode. It prevents unconscious system reset.

WDCON0 – Watchdog Timer Control (TA protected)

7	6	5	4	3	2	1	0
WDTEN	WDCLR	WDTF	WIDPD	WDTRF	WPS2	WPS1	WPS0
r/w	w	r/w	r/w	r/w	r/w	r/w	r/w

Address: D8H reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
7	**WDTEN**	**WDTEN is initialized by inverted CWDTEN (CONFIG3, bit-7) at any other resets.** **WDTEN: WDT enable bit** 0: Disable WDT at power-on reset. 1: Enable WDT at power-on reset.
6	**WDCLR**	**WDT counter clear bit. Writing "1" to clear the WDT counter to 0000H. Note this bit is written-only and has no need to be cleared by being written "0".**
5	**WDTF**	**WDT interrupt flag. This bit will be set by hardware when WDT counter overflows.**.
4	**WIDPD**	**Watchdog Timer running in Idle and Power Down mode.** This bit decides whether Watchdog Timer runs in Idle or Power Down mode. 0 = WDT counter is halted while CPU is in Idle or Power Down mode. 1 = WDT keeps running while CPU is in Idle or Power Down mode.

Bit	Name	Description
3	**WDTRF**	**WDT reset flag.** When the MCU resets itself, this bit is set by hardware. The bit should be cleared by software. If EWRST=0, the interrupt flag WDTF won't be set by hardware, and the MCU will reset itself right away. If EWRST=1, the interrupt flag WDTF will be set by hardware and the MCU will jump into WDT's interrupt service routine if WDT interrupt is enabled, and the MCU won't reset itself until 512 CPU clocks elapse. In other words, in this condition, the user needs also to clear the WDT counter (by writing '1' to WDCLR bit) during this period of 512 CPU clocks, or the MCU will also reset itself when 512 CPU clocks elapse.
2:0	**WPS[2:0]**	WDT prescaler select. Use these bits to select WDT time-out period. The WDT time-out period is determined by the formula $= \frac{64}{(F_{wck} \times \mathrm{Pr}eScalar)}$, where Fwck is the frequency of the WDT clock source. The following table shows an example of WDT timeout period for different Fwck.

[1] WDTEN is initialized by reloading the inversed value of CWDTEN (CONFIG3.7) after all resets.
[2] WIDPD and WPS[2:0] are cleared after power-on reset and keep unchanged after any other resets.
[3] WDTRF will be cleared after power-on reset, be set after Watchdog Timer reset, and remains unchanged after any other resets.

WDCON1 – Watchdog Timer Control (TA protected)

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	EWRST
-	-	-	-	-	-	-	r/w

Address: ABH reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
0	**EWRST**	0: Disable WDT Reset function. 1: Enable WDT Reset function.

[1] EWRST is cleared after power-on reset and keeps unchanged after any other resets.

The Watchdog time-out interval is determined by the formula $= \frac{64}{(F_{wck} \times \Pr eScalar)}$. where F_{wck} is the frequency of internal 10kHz RC. The following table shows an example of the Watchdog time-out interval under different F_{WCK} and pre-scalars.

EIE – Extensive Interrupt Enable

7	6	5	4	3	2	1	0
ET2	ESPI	EPWM	EWDI	-	ECPTF	EKB	EI2
r/w	r/w	r/w	r/w	-	r/w	r/w	r/w

Address: E8H reset value: 0000 0000B

Bit	Name	Description
4	**EWDI**	0: Disable Watchdog Timer Interrupt. 1: Enable Watchdog Timer Interrupt.

The Watchdog Timer time-out selection will result in different time-out values depending on the clock speed. The reset, when enabled, will occur when 512 clocks after time-out has occurred.

Table 11-1: Time-out values for the Watchdog timer

(WPS2,WPS1,WPS0)	Pre-Scalar	WDT Interrupt time-out		Reset time-out	
		Number of Clocks	Time	Number of Clocks	Time
(0,0,0)	1/1	2^6	6.4ms	2^6+512	57.6ms
(0,0,1)	1/2	2x2^6	12.8ms	2x2^6+512	64ms
(0,1,0)	1/8	8x2^6	51.2ms	8x2^6+512	102.4ms
(0,1,1)	1/16	16x2^6	102.40ms	16x2^6+512	153.6ms
(1,0,0)	1/32	32x2^6	204.80ms	32x2^6+512	256ms
(1,0,1)	1/64	64x2^6	409.60ms	64x2^6+512	460.8ms
(1,1,0)	1/128	128x2^6	819.20ms	128x2^6+512	870.4ms
(1,1,1)	1/256	256x2^6	1.638s	256x2^6+512	1.6892s

11.2 Applications of Watchdog Timer Reset

The main application of the Watchdog Timer with time-out reset enabling is for the system monitor. This is important in real-time control applications. In case of some power glitches or electro-magnetic interference, the processor may begin to execute erroneous codes and operate in an unpredictable state. If this is left unchecked the entire system may crash. Using the Watchdog Timer during software development will require the user to select ideal watchdog reset locations for inserting instructions to reset the Watchdog Timer. By inserting the instruction setting WDCLR, it will allow the code to run without any Watchdog Timer reset. However If any erroneous code executes by any power of other interference, the instructions to clear the Watchdog Timer counter will not be executed at the required instants. Thus the Watchdog Timer reset will occur to reset the system start from an erroneously executing condition. The user should remember that WDCON requires a timed access writing.

11.3 Applications of Watchdog Timer Interrupt

There is another application of the Watchdog Timer, which is used as a simple timer. The WDTF flag will be set while the Watchdog Timer completes the selected time interval. The software polls the WDTF flag to detect a time-out and the WDCLR allows software to restart the timer. The Watchdog Timer can also be used as a very long timer. Every time the time-out occurs, an interrupt will occur if the individual interrupt EWDT (EIE.4) and global interrupt enable EA is set.

In some application of low power consumption, the CPU usually stays in Idle mode when nothing needs to be served to save power consumption. After a while the CPU will be woken up to check if anything needs to be served at an interval of programmed period implemented by Timer 0, 1 or 2. However, the current consumption of Idle mode still keeps at a “mA” level. To further reducing the current consumption to “μA” level, the CPU should stay in Power Down mode when nothing needs to be served, and has the ability of waking up at a programmable interval. **N79E845** series are equipped with this useful function. It provides a very low power internal RC 10kHz. Along with the low power consumption application, the Watchdog Timer needs to count under Idle and Power Down mode and wake CPU up from Idle or Power Down mode. The demo code to accomplish this feature is shown below.

The demo code of Watchdog Timer waking up CPU from Power Down.

```
ORG    0000H
LJMP   START

ORG    005BH
LJMP   WDT_ISR

ORG    0100H
```

```
WDT_ISR:
       CLR    EA
       MOV    TA,#0AAH
       MOV    TA,#55H
       ORL    WDCON0,#01000000B          ;clear Watchdog Timer counter
       SETB   EA
Check_clear1:
       MOV    A,WDCON0
       JB     A.6,Check_clear1

       CLR    EA
       MOV    TA,#0AAH
       MOV    TA,#55H
       ANL    WDCON0,#11011111B          ;clear Watchdog Timer interrupt flag
       SETB   EA
       RETI

START:
       MOV    TA,#0AAH
       MOV    TA,#55H
       ORL    WDCON0,#01000000B          ;clear Watchdog Timer counter
Check_clear:
       MOV    A,WDCON0
       JB     A.6,Check_clear

       MOV    TA,#0AAH
       MOV    TA,#55H
       ORL    WDCON0,#00000111B          ;choose interval length
       MOV    TA,#0AAH
       MOV    TA,#55H
       ANL    WDCON1,#11111110B          ;disable Watchdog Timer reset
       SETB   EWDT                       ;enable Watchdog Timer interrupt

       MOV    TA,#0AAH
       MOV    TA,#55H
       ORL    WDCON0,#10000000B          ;enable Watchdog Timer to run
       SETB   EA

;*********************************************************************
;Enter into Power Down mode
;*********************************************************************
LOOP:
       ORL    PCON,#02H
       LJMP   LOOP
```

12 Serial Port (UART)

The **N79E845** series include one enhanced full duplex serial port with automatic address recognition and framing error detection. The serial port supports three modes of full duplex UART (Universal Asynchronous Receiver and Transmitter) in Mode 1, 2, and 3. This means it can transmit and receive simultaneously. The serial port is also receiving-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the register. The receiving and transmitting registers are both accessed at SBUF. Writing to SBUF loads the transmitting register, and reading SBUF accesses a physically separate receiving register. There are four operation modes in serial port. In all four modes, transmission initiates by any instruction that uses SBUF as a destination register. Note that before serial port function works, the port latch bits of RXD and TXD pins have to be set to 1.

SCON – Serial Port Control (bit-addressable)

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

Address: 98H reset value: 0000 0000B

Bit	Name	Description
7	SM0/FE	**Serial port mode select.** SMOD0 (PCON.6) = 0: See Table 12–1. Serial Port Mode Description in details. SMOD0 (PCON.6) = 1: SM0/FE bit is used as frame error (FE) status flag. 0 = Frame error (FE) does not occur. 1 = Frame error (FE) occurs and is detected.
6	SM1	
5	SM2	**Multiprocessor communication mode enable.** The function of this bit is dependent on the serial port mode. Mode 0: This bit select the baud rate between F_{SYS}/12 and F_{SYS}/4. 0 = The clock runs at F_{SYS}/12 baud rate. It maintains standard 8051 compatibility. 1 = The clock runs at F_{SYS}/4 baud rate for faster serial communication. Mode 1: This bit checks valid stop bit. 0 = Reception is always valid no matter the logic level of stop bit. 1 = Reception is valid only when the received stop bit is logic 1 and the received data matches GIVEN or BROADCAST address. Mode 2 or 3: For multiprocessor communication. 0 = Reception is always valid no matter the logic level of the 9th bit. 1 = Reception is valid only when the received 9th bit is logic 1 and the received data matches GIVEN or BROADCAST address.

Bit	Name	Description
4	REN	**Receiving enable.** 0 = Serial port reception is disabled. 1 = Serial port reception is enabled in Mode 1,2, and 3. In Mode 0, clearing and then setting REN initiates one-byte reception. After reception is complete, this bit will not be cleared via hardware. The user should clear and set REN again via software to triggering the next byte reception.
3	TB8	**9th transmitted bit.** This bit defines the state of the 9th transmission bit in serial port Mode 2 and 3. It is not used in Mode0 and 1.
2	RB8	**9th received bit.** The bit identifies the logic level of the 9th received bit in Modes 2 and 3. In Mode 1, if SM2 0, RB8 is the logic level of the received stop bit. RB8 is not used in Mode 0.
1	TI	**Transmission interrupt flag.** This flag is set via hardware when a byte of data has been transmitted by the UART after the 8th bit in Mode 0 or the last bit of data in other modes. When the UART interrupt is enabled, setting this bit causes the CPU to execute the UART interrupt service routine. This bit must be cleared manually via software.
0	RI	**Receiving interrupt flag.** This flag is set via hardware when a 8-bit or 9-bit data has been received by the UART after the 8th bit in Mode 0, after sampling the stop bit in Mode 1, or after sampling the 9th bit in Mode 2 and 3. SM2 bit has restriction for exception. When the UART interrupt is enabled, setting this bit causes the CPU to execute to the UART interrupt service routine. This bit must be cleared manually via software.

Table 12–1. Serial Port Mode Description

Mode	SM0	SM1	Description	Frame Bits	Baud Rate
0	0	0	Synchronous	8	F_{SYS} divided by 12 or by 4[1]
1	0	1	Asynchronous	10	Timer 1 overflow rate divided by 32 or divided by 16[2]
2	1	0	Asynchronous	11	F_{SYS} divided by 64 or 32 [2]
3	1	1	Asynchronous	11	Timer 1 overflow rate divided by 32 or divided by 16[2]

[1] While SM2 (SCON.5) is logic 1.
[2] While SMOD (PCON.7) is logic 1.

PCON – Power Control

7	6	5	4	3	2	1	0
SMOD	SMOD0	-	POF	GF1	GF0	PD	IDL
r/w	r/w	-	r/w	r/w	r/w	r/w	r/w

Address: 87H reset value: see Table 7–2. N79E845 Series SFR Descriptions and Reset Values

Bit	Name	Description
7	SMOD	**Serial port double baud rate enable.** Setting this bit doubles the serial port baud rate in UART mode 2 and mode 1 or 3 only if Timer 1 overflow is used as the baud rate source. See Table 12–1. Serial Port Mode Description in details.

Bit	Name	Description
6	SMOD0	**Framing error detection enable.** 0 = Framing error detection is disabled. SM0/FE (SCON.7) bit is used as SM0 as standard 80C51 function. 1 = Framing error detection is enabled. SM0/FE bit is used as frame error (FE) status flag.

SBUF – Serial Data Buffer

7	6	5	4	3	2	1	0
SBUF[7:0]							
r/w							

Address: 99H reset value: 0000 0000B

Bit	Name	Description
7:0	SBUF[7:0]	**Serial data buffer.** This byte actually consists two separate registers. One is the receiving resister, and the other is the transmitting buffer. When data is moved to SBUF, it goes to the transmitting buffer and is shifted for serial transmission. When data is moved from SBUF, it comes from the receiving buffer. The transmission is initiated through moving a byte to SBUF.

12.1 Mode 0

Mode 0 provides synchronous communication with external devices. Serial data enters and exits through RXD pin. TXD outputs the shift clock. 8 bits are transmitted or received. Mode 0 therefore provides half-duplex communication because the transmitting or receiving data is via the same data line RXD. The baud rate is enhanced to be selected as $F_{SYS}/12$ if SM2 (SCON.5) is 0 or as $F_{SYS}/4$ if SM2 is 1. Note that whenever transmitting or receiving, the serial clock is always generated by the microcontroller. Thus any device on the serial port in Mode 0 must accept the microcontroller as the Master. Figure 12–1 shows a simplified functional diagram of the serial port in Mode 0 and associated timing.

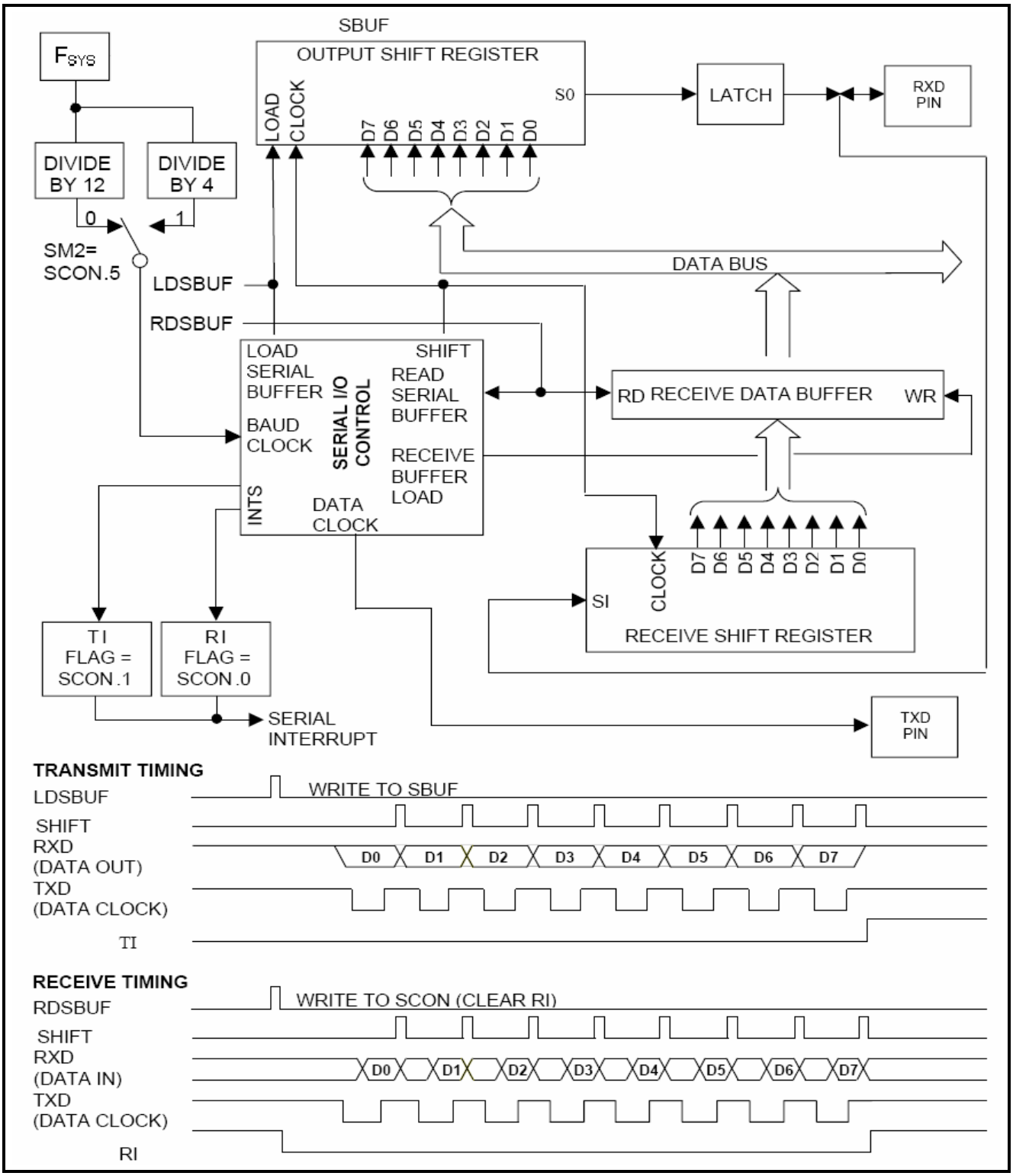


Figure 12–1. Serial Port Mode 0 Function Block

As shown there is one bi-directional data line (RXD) and one shift clock line (TXD). The shift clock is used to shift data in or out of the serial port controller bit by bit for a serial communication. Data bits enter or exit LSB first. The band rate is equal to the shift clock frequency.

Transmission is initiated by any instruction writes to SBUF. The control block will then shift out the clock and begin to transfer data until all 8 bits are complete. Then the transmitted flag TI (SCON.1) will be set 1 to indicate one byte transmitting complete.

Reception is initiated by clearing and then setting REN (SCON.4) while RI (SCON..0) is 0. This condition tells the serial port controller that there is data to be shifted in. This process will continue until 8 bits have been received. Then the received flag RI will be set as 1. Note that REN will not be cleared via hardware. The user should first clear RI, clear REN and then set REN again via software to triggering the next byte reception.

12.2 Mode 1

Mode 1 supports asynchronous, full duplex serial communication. The asynchronous mode is commonly used for communication with PCs, modems or other similar interfaces. In Mode 1, 10 bits are transmitted (through TXD) or received (through RXD) including a start bit (logic 0), 8 data bits (LSB first) and a stop bit (logic 1). The baud rate is determined by the Timer 1. SMOD (PCON.7) setting 1 makes the baud rate double while Timer 1 is selected as the clock source. Figure 12–2 shows a simplified functional diagram of the serial port in Mode 1 and associated timings for transmitting and receiving.

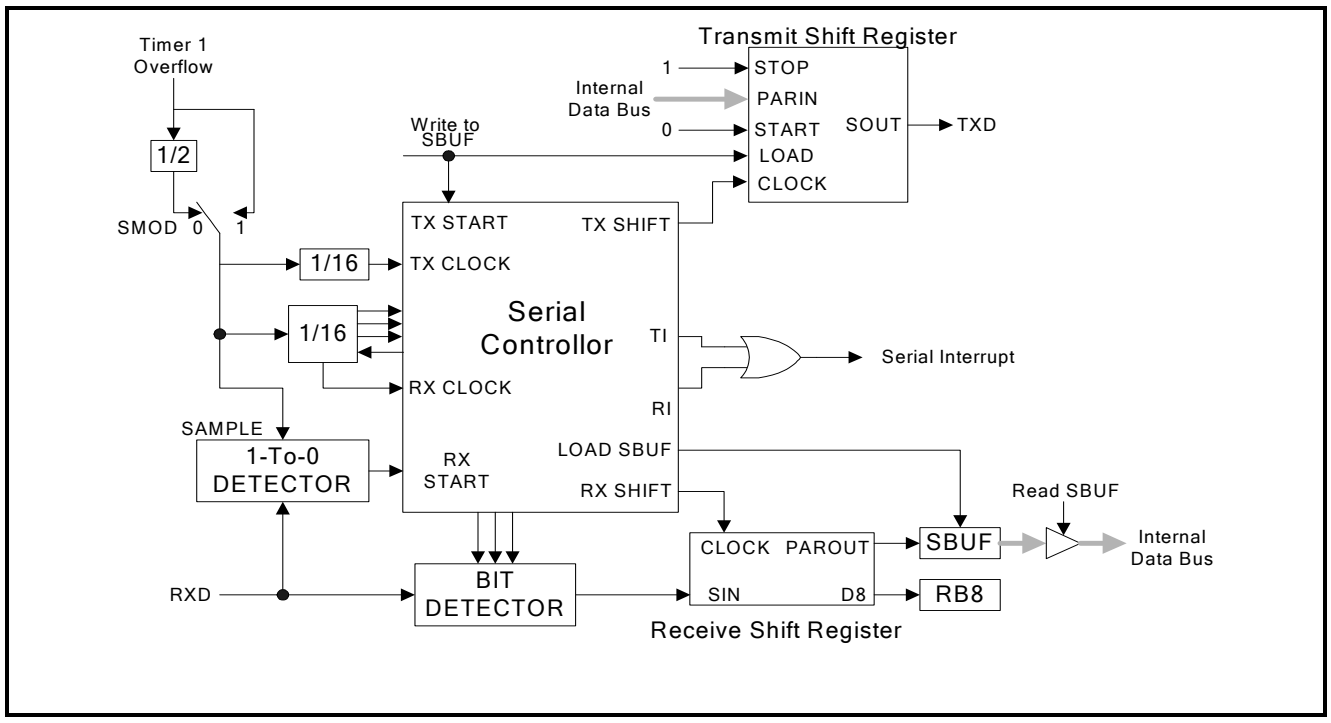


Figure 12–2. Serial Port Mode 1 Function Block

Transmission is initiated by any writing instructions to SBUF. Transmission takes place on TXD pin. First the start bit comes out, the 8-bit data follows to be shifted out and then ends with a stop bit. After the stop bit appears, TI (SCON.1) will be set to indicate one byte transmission complete. All bits are shifted out depending on the rate determined by the baud rate generator.

Once the baud rate generator is activated and REN (SCON.4) is 1, the reception can begin at any time. Reception is initiated by a detected 1-to-0 transition at RXD. Data will be sampled and shifted in at the selected baud rate. In the midst of the stop bit, certain conditions must be met to load SBUF with the received data:

1. RI (SCON.0) = 0, and

2. Either SM2 (SCON.5) = 0, or the received stop bit = 1 while SM2 = 1.

If these conditions are met, then the SBUF will be loaded with the received data, the RB8 (SCON.2) with stop bit, and RI will be set. If these conditions fail, there will be no data loaded and RI will remain 0. After above receiving progress, the serial control will look forward another 1-0 transition on RXD pin in order to start next data reception.

12.3 Mode 2

Mode 2 supports asynchronous, full duplex serial communication. Different from Mode1, there are 11 bits to be transmitted or received. They are a start bit (logic 0), 8 data bits (LSB first), a programmable 9th bit TB8 or RB8 bit and a stop bit (logic 1). The most common use of 9th bit is to put the parity bit in it. The baud rate is fixed as 1/32 or 1/64 the system clock frequency depending on SMOD bit. Figure 12–3 shows a simplified functional diagram of the serial port in Mode 2 and associated timings for transmitting and receiving.

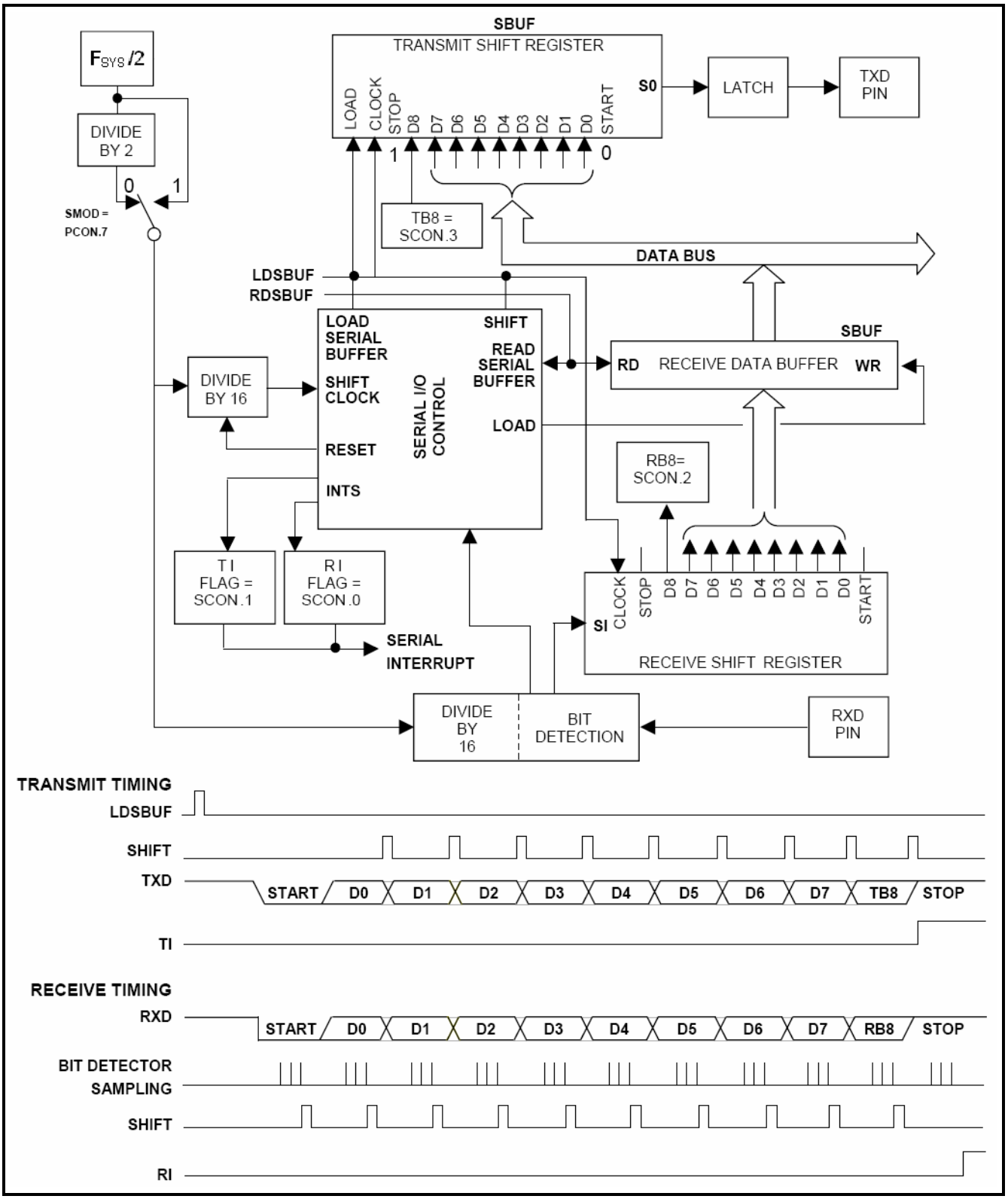


Figure 12–3. Serial Port Mode 2 Function Block and

Transmission is initiated by any writing instructions to SBUF. Transmission takes place on TXD pin. First the start bit comes out, the 8-bit data and bit TB8 (SCON.3) follows to be shifted out and then ends with a stop bit. After the stop bit appears, TI will be set to indicate the transmission complete.

While REN is set, the reception is allowed at any time. A falling edge of a start bit on RXD will initiate the reception progress. Data will be sampled and shifted in at the selected baud rate. In the midst of the 9th bit, certain conditions must be met to load SBUF with the received data:

1. RI (SCON.0) = 0, and

2. Either SM2(SCON.5) = 0, or the received 9th bit = 1 while SM2 = 1.

If these conditions are met, then the SBUF will be loaded with the received data, the RB8(SCON.2) with TB8 bit and RI will be set. If these conditions fail, there will be no data loaded and RI will remain 0. After above receiving progress, the serial control will look forward another 1-0 transition on RXD pin in order to start next data reception.

12.4 Mode 3

Mode 3 has the same operation as Mode 2, except its baud rate clock source. As shown is Figure 12–4, Mode 3 uses Timer 1 overflow as its baud rate clock.

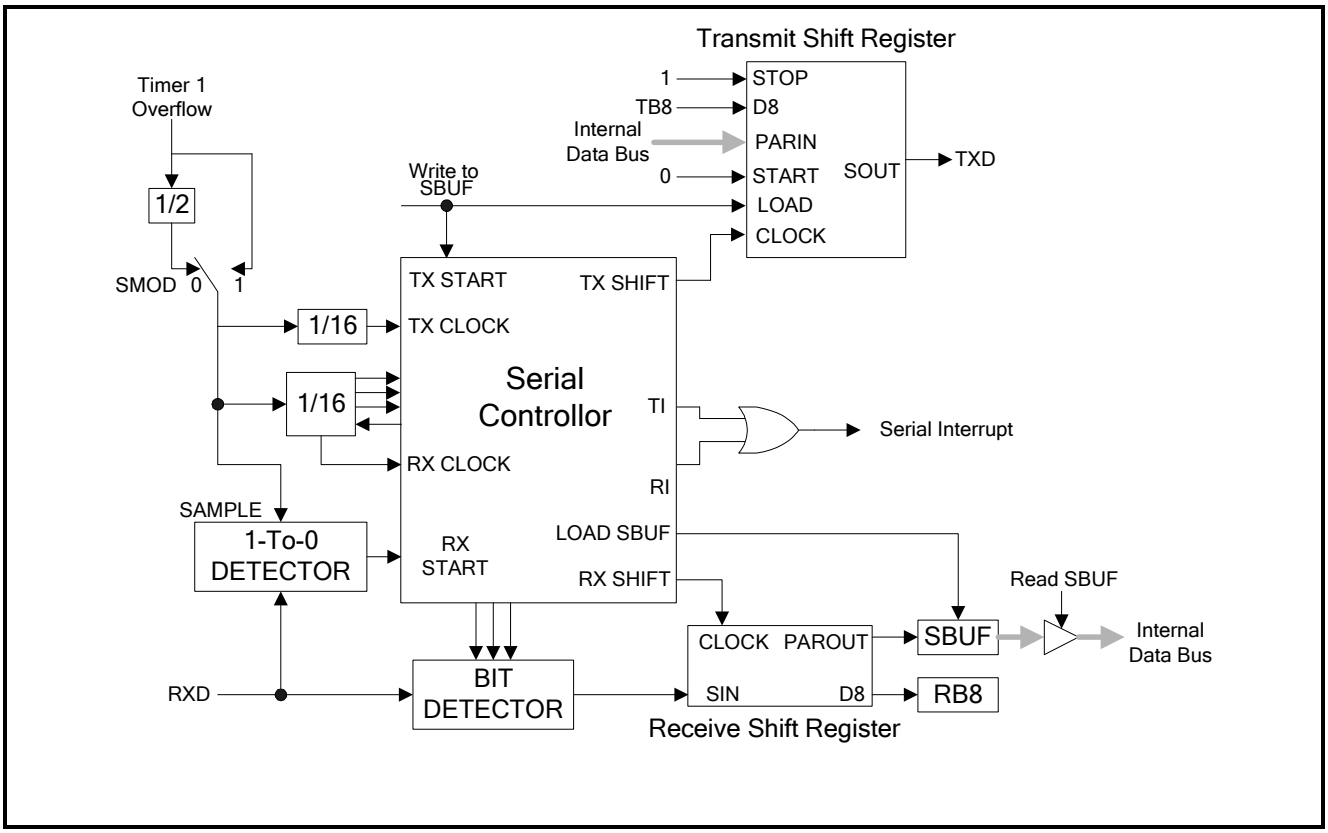


Figure 12–4. Serial Port Mode 3 Function Block

12.5 Baud Rates

Table 12–2. UART Baud Rate Formulas

UART mode	Baud rate clock source	Baud rate
0	Oscillator	$F_{SYS}/12$ or $F_{SYS}/4$ [1]
2	Oscillator	$\frac{2^{SMOD}}{64} \times F_{SYS}$
1 or 3	Timer/Counter 1 overflow[2]	$\frac{2^{SMOD}}{32} \times \frac{F_{SYS}}{12 \times (256 - TH1)}$ or $\frac{2^{SMOD}}{32} \times \frac{F_{SYS}}{4 \times (256 - TH1)}$ [3]

[1] While SM2 (SCON.5) is set as a logic 1.
[2] Timer 1 is configured as a timer in auto-reload mode (Mode 2).
[3] While T1M (CKCON.4) is set as a logic 1.

Note that in using Timer 1 as the baud rate generator, the interrupt should be disabled. The Timer itself can be configured for either “Timer” or “Counter” operation. And Timer 1 can be in any of its 3 running modes. In the

most typical applications, it is configured for “Timer” operation, in the auto-reload mode (Mode2). If Timer 1 is used as the baud rate generator, the reloaded value is stored in TH1. Therefore the baud rate is determined by TH1 value.

Table 12–3 lists various commonly used baud rates and how they can be obtained from Timer 1. In this mode, Timer 1 operates with divided-by-12 pre-scale, as an auto-reload Timer with SMOD (PCON.7) is 0. If SMOD is 1, the baud rate will be doubled.

Table 12–3. Timer 1 Generated Commonly Used Baud Rates

TH1 reload value	Oscillator Frequency (MHz)			
	11.0592	14.7456	18.432	22.1184
Baud Rate				
57600				FFh
38400		FFh		
19200		FEh		FDh
9600	FDh	FCh	FBh	FAh
4800	FAh	F8h	F6h	F4h
2400	F4h	F0h	ECh	E8h
1200	E8h	E0h	D8h	D0h
300	A0h	80h	60h	40h

12.6 Framing Error Detection

Framing error detection is provided for asynchronous modes. (Mode 1, 2 and 3.) The framing error occurs when a valid stop bit is not detected due to the bus noise or contention. The UART can detect a framing error and notify the software.

The framing error bit, FE, is located in SCON.7. This bit normally serves as SM0. While the framing error detection enable bit SMOD0 (PCON.6) is set 1, it serves as FE flag. Actually SM0 and FE locate in different registers.

The FE bit will be set 1 via hardware while a framing error occurs. It must be cleared via software. Note that SMOD0 must be 1 while reading or writing to FE. If FE is set, then any following frames received without any error will not clear the FE flag. The clearing has to be done via software.

12.7 Multiprocessor Communication

The **N79E845** series multiprocessor communication features let a Master device send a multiple frame serial message to a Slave device in a multi-slave configuration. It does this without interrupting other slave devices that may be on the same serial line. This feature can be used only in UART mode 2 or 3 mode. After 9 data bits are received. The 9th bit value is written to RB8 (SCON.2). The user can enable this function by setting SM2 (SCON.5) as a logic 1 so that when the stop bit is received, the serial interrupt will be generated only if RB8 is 1. When the SM2 bit is 1, serial data frames that are received with the 9th bit as 0 do not generate an interrupt. In this case, the 9th bit simply separates the address from the serial data.

When the Master device wants to transmit a block of data to one of several slaves on a serial line, it first sends out an address byte to identify the target slave. Note that in this case, an address byte differs from a data byte: In an address byte, the 9th bit is 1 and in a data byte, it is 0. The address byte interrupts all slaves so that each slave can examine the received byte and see if it is being addressed. The addressed slave then clears its SM2 bit and prepares to receive incoming data bytes. The SM2 bits of slaves that were not addressed remain set, and they continue operating normally while ignoring the incoming data bytes.

Follow these steps to configure multiprocessor communications:

1. Set all devices (Masters and Slaves) to UART mode 2 or 3.

2. Write the SM2 bit of all the Slave devices to 1.

3. The Master device's transmission protocol is:

   – First byte: the address, identifying the target slave device, (9th bit = 1).

   – Next bytes: data, (9th bit = 0).

4. When the target Slave receives the first byte, all of the Slaves are interrupted because the 9th data bit is 1. The targeted Slave compares the address byte to its own address and then clears its SM2 bit in order to receiving incoming data. The other slaves continue operating normally.

5. After all data bytes have been received, set SM2 back to 1 to wait for next address.

SM2 has no effect in Mode 0, and in Mode 1 can be used to check the validity of the stop bit. For mode 1 reception, if SM2 is 1, the receiving interrupt will not be issue unless a valid stop bit is received.

12.8 Automatic Address Recognition

The automatic address recognition is a feature which enhances the multiprocessor communication feature by allowing the UART to recognize certain addresses in the serial bit stream by using hardware to make the comparisons. This feature saves a great deal of software overhead by eliminating the need for the software to examine every serial address which passes by the serial port. Only when the serial port recognizes its own address, the receiver sets RI bit to request an interrupt. The automatic address recognition feature is enabled when the multiprocessor communication feature is enabled. (SM2 is set.)

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

Using the automatic address recognition feature allows a master to selectively communicate with one or more slaves by invoking the "Given" slave address or addresses. All of the slaves may be contacted by using the "Broadcast" address. Two SFRs are used to define the slave address, SADDR, and the slave address mask, SADEN. SADEN is used to define which bits in the SADDR are to be used and which bits are "don't care". The SADEN mask can be logically ANDed with the SADDR to create the "Given" address which the master will use for addressing each of the slaves. Use of the "Given" address allows multiple slaves to be recognized while excluding others.

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 0010 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:

Example 1, slave 0:

```
SADDR = 11000000b
SADEN = 11111001b
Given = 11000XX0b
```

Example 2, slave 1:

```
SADDR = 11100000b
SADEN = 11111010b
Given = 11100X0Xb
```

Example 3, slave 2:

```
SADDR = 11000000b
SADEN = 11111100b
Given = 110000XXb
```

In the above example the differentiation among the 3 slaves is in the lower 3 address bits. Slave 0 requires that bit 0 = 0 and it can be uniquely addressed by 11100110b. Slave 1 requires that bit 1 = 0 and it can be uniquely addressed by 11100101b. Slave 2 requires that bit 2 = 0 and its unique address is 11100011b. To select Slaves 0 and 1 and exclude Slave 2 use address 11100100b, since it is necessary to make bit 2 = 1 to exclude slave 2. The “Broadcast” address for each slave is created by taking the logical OR of SADDR and SADEN. Zeros in this result are treated as “don’t-cares”. In most cases, interpreting the “don’t-cares” as ones, the broadcast address will be FFH.

On reset, SADDR and SADEN are initialized to 00H. This produces a “Given” address of all “don’t cares” as well as a “Broadcast” address of all XXXXXXXXb (all “don’t care” bits). This effectively disables the automatic addressing mode and allows the microcontroller to use standard UART drivers which do not make use of this feature.

13 Serial Peripheral Interface (SPI)

13.1 Features

N79E845 series exist a Serial Peripheral Interface (SPI) block to support high speed serial communication. SPI is a full-duplex, high speed, synchronous communication bus between MCUs or other peripheral devices such as serial EEPROM, LCD driver, or D/A converter. It provides either Master or Slave mode, high speed rate up to F_{SYS}/16 for Master mode and F_{SYS}/4 for Slave mode, transfer complete and write collision flag. For a multi-master system, SPI supports Master Mode Fault to protect a multi-master conflict.

13.2 Function Description

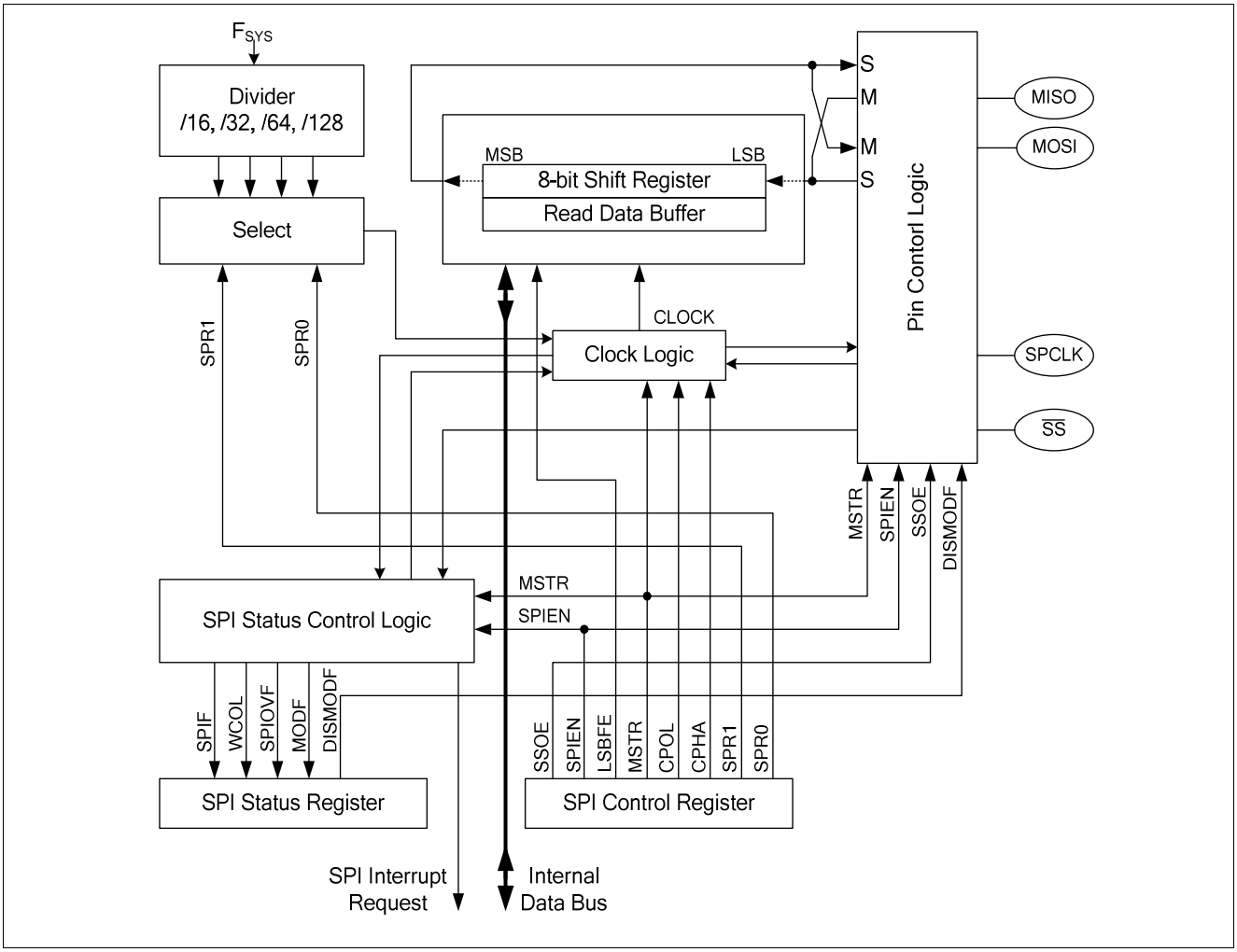


Figure 13–1. SPI Block Diagram

Figure 13–1 shows SPI block diagram. It provides an overview of SPI architecture in this device. The main blocks of SPI are the SPI control register logic, SPI status logic, clock rate control logic, and pin control logic. For a serial data transfer or receiving, The SPI block exists a shift register and a read data buffer. It is single buffered in the transmit direction and double buffered in the receiving direction. Transmit data cannot be written to the shifter until the previous transfer is complete. Receiving logic consists of parallel read data buffer so the shift register is free to accept a second data, as the first received data will be transferred to the read data buffer.

The four pins of SPI interface are Master-In/Slave-Out (MISO), Master-Out/Slave-In (MOSI), Shift Clock (SPCLK), and Slave Select ($\overline{SS}$). The MOSI pin is used to transfer a 8-bit data in series from the Master to the Slave. Therefore, MOSI is an output pin for Master device and a input for Slave. Respectively, the MISO is used to receive a serial data from the Slave to the Master.

The SPCLK pin is the clock output in Master mode, but is the clock input in Slave mode. The shift clock is used to synchronize the data movement both in and out of the devices through their MOSI and MISO pins. The shift clock is driven by the Master mode device for eight clock cycles which exchanges one byte data on the serial lines. For the shift clock is always produced out of the Master device, the system should never exist more than one device in Master mode for avoiding device conflict. The Schmitt trigger input buffer is strongly recommended to be enabled by setting P1S and P0S for improved glitch suppression.

Each Slave peripheral is selected by one Slave Select pin ($\overline{SS}$). The signal must stay low for any Slave access. When $\overline{SS}$ is driven high, the Slave device will be inactivated. If the system is multi-slave, there should be only one Slave device selected at the same time. In the Master mode MCU, the $\overline{SS}$ pin does not function and it can be configured as a general purpose I/O. However, $\overline{SS}$ can be used as Master Mode Fault detection (see Section 13.7“Mode Fault Detection” on page 84) via software setting if multi-master environment exists. **N79E845** series also provide auto-activating function to toggle $\overline{SS}$ between each byte-transfer.

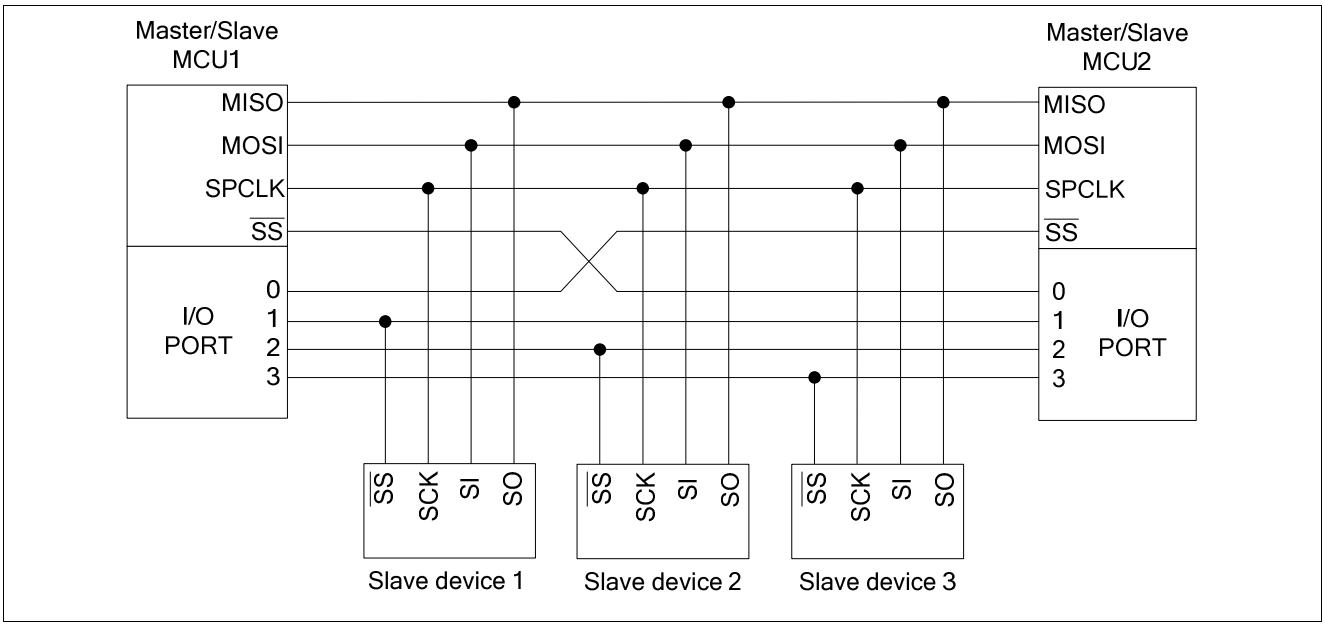


Figure 13–2. SPI Multi-master, Multi-slave Interconnection

Figure 13–2 shows a typical interconnection of SPI devices. The bus generally connects devices together through three signal wires, MOSI to MOSI, MISO to MISO, and SPCLK to SPCLK. The Master devices select the individual Slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins. MCU1 and MCU2 play either Master or Slave mode. The $\overline{SS}$ should be configured as Master Mode Fault detection to avoid multi-master conflict.

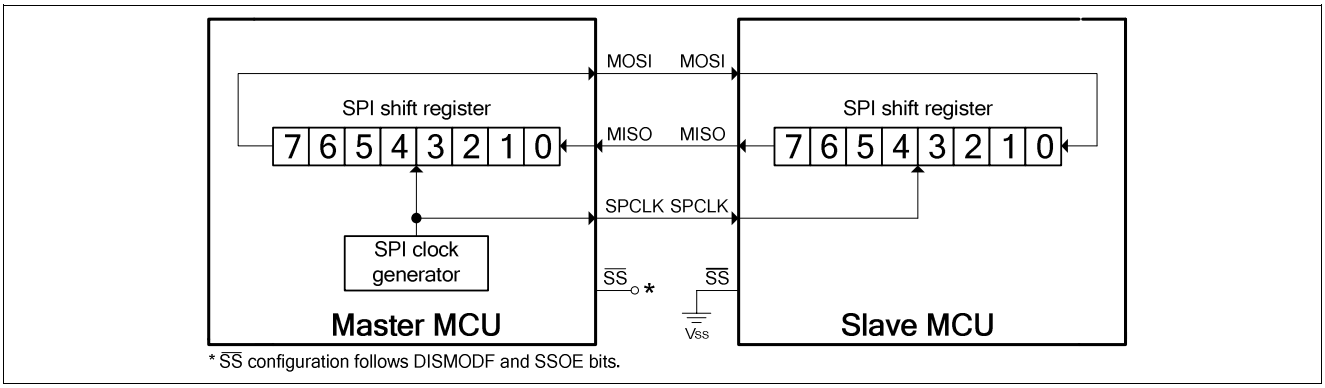


Figure 13–3. SPI Single-master, Single-slave Interconnection

Figure 13–3 shows the simplest SPI system interconnection, single-master and signal-slave. During a transfer, the Master shifts data out to the Slave via MOSI line. While simultaneously, the Master shifts data in from the Slave via MISO line. The two shift registers in the Master MCU and the Slave MCU can be considered as one 16-bit circular shift register. Therefore, while a transfer data pushed from Master into Slave, the data in Slave will also be pulled in Master device respectively. The transfer effectively exchanges the data which was in the SPI shift registers of the two MCUs.

By default, SPI data is transferred MSB first. If the LSBFE (SPCR.5) is set, SPI data shifts LSB first. This bit does not affect the position of the MSB and LSB in the data register. Note that all following descriptions and figures are under the condition of LSBFE logic 0. MSB is transmitted and received first.

13.3 Control Registers of SPI

There are three SPI registers to support its operations, they are SPI control register (SPCR), SPI status register (SPSR), and SPI data register (SPDR). These registers provide control, status, data storage functions, and clock rate selection. The following registers relate to SPI function.

SPCR – Serial Peripheral Control Register

7	6	5	4	3	2	1	0
SSOE	SPIEN	LSBFE	MSTR	CPOL	CPHA	SPR1	SPR0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: F3H reset value: 0000 0000B

Bit	Name	Description
7	**SSOE**	**Slave select output enable.** This bit is used in combination with the DISMODF (SPSR.3) bit to determine the feature of $\overline{SS}$ pin. This bit takes effect only under MSTR = 1 and DISMODF = 1 condition. 0 = $\overline{SS}$ functions as a general purpose I/O pin. 1 = $\overline{SS}$ automatically goes low for each transmission when selecting external Slave device and goes high during each idle state to de-select the Slave device.
6	**SPIEN**	**SPI enable.** 0 = Disable SPI function. 1 = Enable SPI function.
5	**LSBFE**	**LSB first enable.** 0 = The SPI data is transferred MSB first. 1 = The SPI data is transferred LSB first.

Bit	Name	Description
4	**MSTR**	**Master mode enable.** This bit switches the SPI operating between Master and Slave modes. 0 = The SPI is configured as Slave mode. 1 = The SPI is configured as Master mode.
3	**CPOL**	**SPI clock polarity select.** CPOL bit determines the idle state level of the SPI clock. See Figure 13–4. SPI Clock Formats. 0 = The SPI clock is low in idle state. 1 = The SPI clock is high in idle state.
2	**CPHA**	**SPI clock phase select.** CPHA bit determines the data sampling edge of the SPI clock. See Figure 13–4. SPI Clock Formats. 0 = The data is sampled on the first edge of the SPI clock. 1 = The data is sampled on the second edge of the SPI clock.
1	**SPR1**	**SPI clock rate select.**
0	**SPR0**	These two bits select four grades of SPI clock divider. SPR1 SPR0 Divider SPI clock rate 0 0 16 1.25M bit/s 0 1 32 625k bit/s 1 0 64 312k bit/s 1 1 128 156k bit/s The clock rates above are illustrated under F_{SYS} = 20MHz condition.

Table 13–1. Slave Select Pin Configurations

DISMODF	SSOE	Master Mode (MSTR = 1)	Slave Mode (MSTR = 0)
0	x	$\overline{SS}$ input for Mode Fault	$\overline{SS}$ Input for Slave select
1	0	General purpose I/O	
1	1	Automatic $\overline{SS}$ output	

SPSR – Serial Peripheral Status Register

7	6	5	4	3	2	1	0
SPIF	WCOL	SPIOVF	MODF	DISMODF	-	-	-
r/w	r/w	r/w	r/w	r/w	-	-	-

Address: F4H reset value: 0000 0000B

Bit	Name	Description
7	**SPIF**	**SPI complete flag.** This bit is set to logic 1 via hardware while an SPI data transfer is complete or an receiving data has been moved into the SPI read buffer. If ESPI (EIE .6) and EA are enabled, an SPI interrupt will be required. This bit must be cleared via software. Attempting to write to SPDR is inhibited if SPIF is set.
6	**WCOL**	**Write collision error flag.** This bit indicates a write collision event. Once a write collision event occurs, this bit will be set. It must be cleared via software.
5	**SPIOVF**	**SPI overrun error flag.** This bit indicates an overrun event. Once an overrun event occurs, this bit will be set. If ESPI and EA are enabled, an SPI interrupt will be required. This bit must be cleared via software.
4	**MODF**	**Mode Fault error flag.** This bit indicates a Mode Fault error event. If $\overline{SS}$ pin is configured as Mode Fault input (MSTR = 1 and DISMODF = 0) and $\overline{SS}$ is pulled low by external devices, a Mode Fault error occurs. Instantly MODF will be set as logic 1. If ESPI and EA are enabled, an SPI interrupt will be required. This bit must be cleared via software.
3	**DISMODF**	**Disable Mode Fault error detection.** This bit is used in combination with the SSOE (SPCR.7) bit to determine the feature of $\overline{SS}$ pin. DISMODF affects only in Master mode (MSTR = 1). 0 = Mode Fault detection is not disabled. $\overline{SS}$ serves as input pin for Mode Fault detection disregard of SSOE. 1 = Mode Fault detection is disabled. The feature of $\overline{SS}$ follows SSOE bit.
2:0	**-**	**Reserved**.

SPDR – Serial Peripheral Data Register

7	6	5	4	3	2	1	0
SPDR[7:0]							
r/w							

Address: F5H reset value: 0000 0000B

Bit	Name	Description
7:0	**SPDR[7:0]**	**Serial peripheral data.** This byte is used of transmitting or receiving data on SPI bus. A write of this byte is a write to the shift register. A read of this byte is actually a read of the read data buffer. In Master mode, a write to this register initiates transmission and reception of a byte simultaneously.

13.4 Operating Modes

13.4.1 Master mode

The SPI can operate in Master mode while MSTR (SPCR.4) is set as 1. Only one Master SPI device can initiate transmissions. A transmission always begins by Master through writing to SPDR. The byte written to SPDR begins shifting out on MOSI pin under the control of SPCLK. Simultaneously, another byte shifts in from the Slave on the MISO pin. After 8-bit data transfer complete, SPIF (SPSR.7) will automatically set via hardware to indicate one byte data transfer complete. At the same time, the data received from the Slave is also transferred in SPDR. The user can clear SPIF and read data out of SPDR.

13.4.2 Slave Mode

When MSTR is 0, the SPI operates in Slave mode. The SPCLK pin becomes input and it will be clocked by another Master SPI device. The $\overline{SS}$ pin also becomes input. The Master device cannot exchange data with the Slave device until the $\overline{SS}$ pin of the Slave device is externally pulled low. Before data transmissions occurs, the $\overline{SS}$ of the Slave device must be pulled and remain low until the transmission is complete. If $\overline{SS}$ goes high, the SPI is forced into idle state. If the $\overline{SS}$ is force to high at the middle of transmission, the transmission will be aborted and the rest bits of the receiving shifter buffer will be high and goes into idle state.

In Slave mode, data flows from the Master to the Slave on MOSI pin and flows from the Slave to the Master on MISO pin. The data enters the shift register under the control of the SPCLK from the Master device. After one byte is received in the shift register, it is immediately moved into the read data buffer and the SPIF bit is set. A read of the SPDR is actually a read of the read data buffer. To prevent an overrun and the loss of the byte that

caused by the overrun, the Slave must read SPDR out and the first SPIF must be cleared before a second transfer of data from the Master device comes in the read data buffer.

13.5 Clock Formats and Data Transfer

To accommodate a wide variety of synchronous serial peripherals, the SPI has a clock polarity bit CPOL (SPCR.3) and a clock phase bit CPHA (SPCR.2). Figure 13–4. SPI Clock Formats shows that CPOL and CPHA compose four different clock formats. The CPOL bit denotes the SPCLK line level in ISP idle state. The CPHA bit defines the edge on which the MOSI and MISO lines are sampled. The CPOL and CPHA should be identical for the Master and Slave devices on the same system. To Communicate in different data formats with one another will result undetermined result.

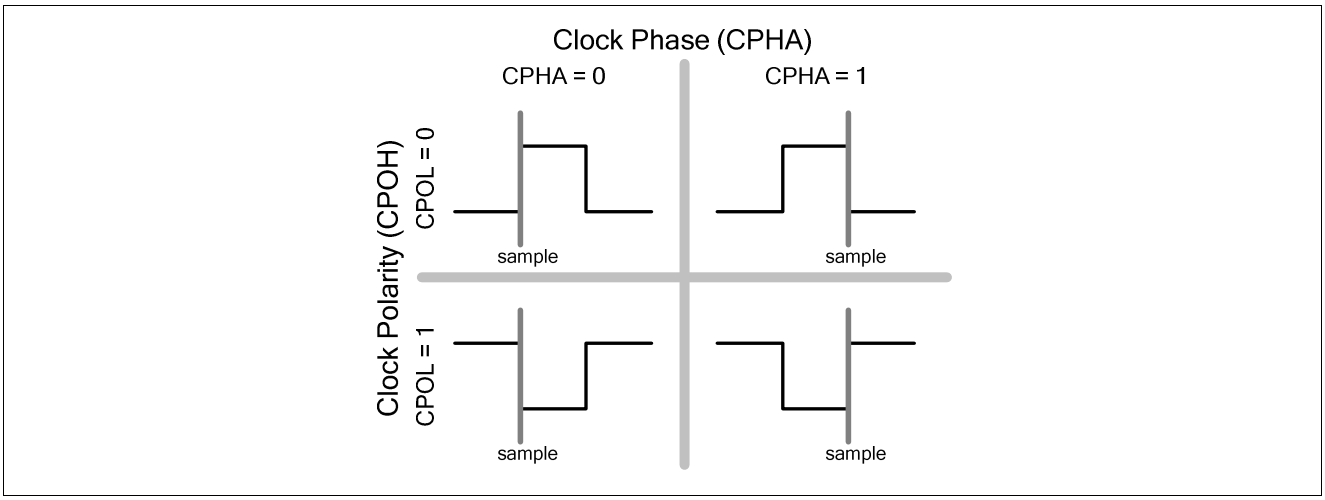


Figure 13–4. SPI Clock Formats

In SPI, a Master device always initiates the transfer. If SPI is selected as Master mode (MSTR = 1) and enabled (SPIEN = 1), writing to the SPI data register (SPDR) by the Master device starts the SPI clock and data transfer. After shifting one byte out and receiving one byte in, the SPI clock stops and SPIF (SPSR.7) in both Master and Slave are set. If SPI interrupt enable bit ESPI (EIE.6) is set 1 and global interrupt is enabled (EA = 1), the interrupt service routine (ISR) of SPI will be executed.

Concerning the Slave mode, the $\overline{SS}$ signal needs to be taken care. As shown in Figure 13–4. SPI Clock Formats, when CPHA = 0, the first SPCLK edge is the sampling strobe of MSB (for an example of LSBFE = 0, MSB first). Therefore, the Slave must shift its MSB data before the first SPCLK edge. The falling edge of $\overline{SS}$ is used for preparing the MSB on MISO line. The $\overline{SS}$ pin therefore must toggle high and then low between each successive serial byte. Furthermore, if the slave writes data to the SPI data register (SPDR) while $\overline{SS}$ is low, a write collision error occurs.

When CPHA = 1, the sampling edge thus locates on the second edge of SPCLK clock. The Slave uses the first SPCLK clock to shift MSB out rather than the $\overline{SS}$ falling edge. Therefore, the $\overline{SS}$ line can remain low between successive transfers. This format may be preferred in systems having single fixed Master and single fixed Slave. The $\overline{SS}$ line of the unique Slave device can be tied to V_{SS} as long as only CPHA = 1 clock mode is used.

Note: The SPI should be configured before it is enabled (SPIEN = 1), or a change of LSBFE, MSTR, CPOL, CPHA and SPR[1:0] will abort a transmission in progress and force the SPI system into idle state. Prior to any configuration bit changed, SPIEN must be disabled first.

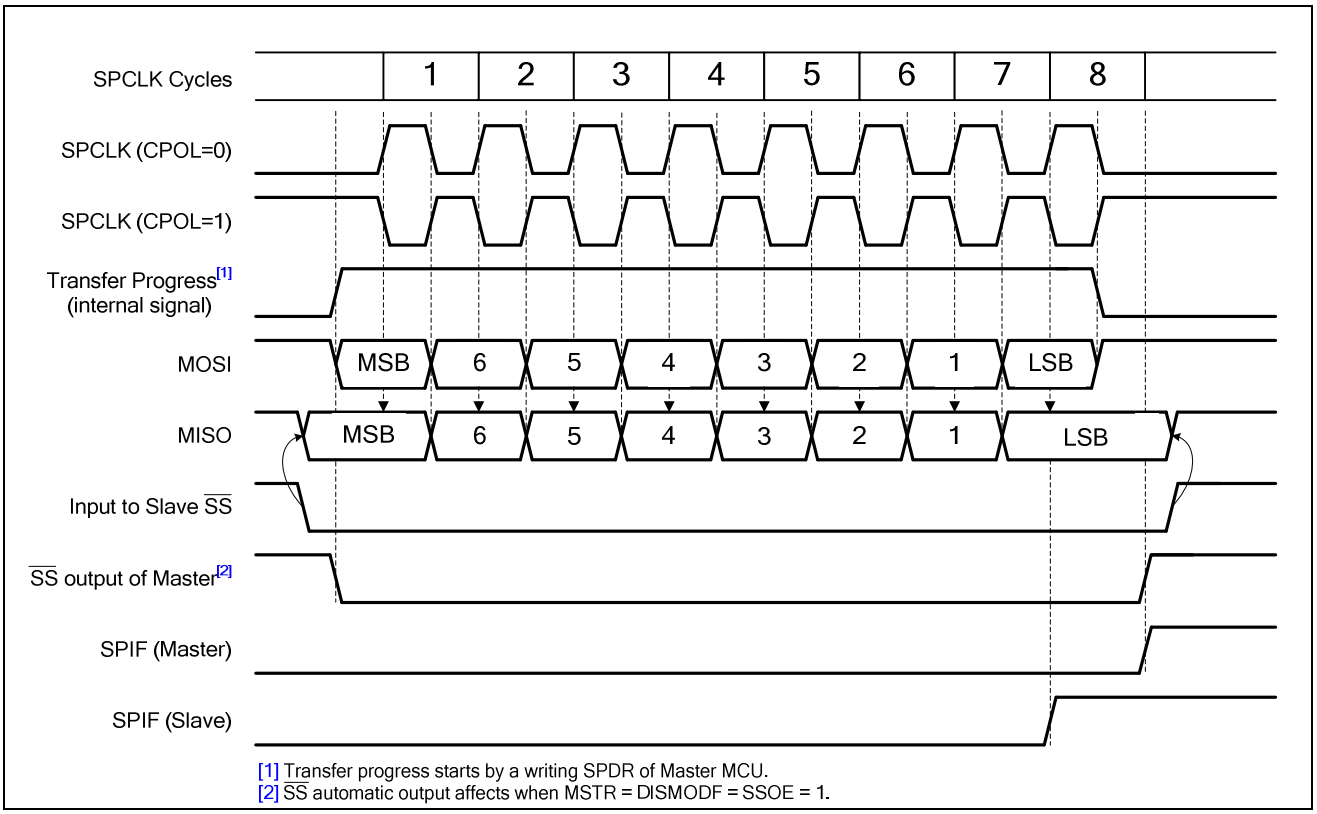


Figure 13–5. SPI Clock and Data Format with CPHA = 0

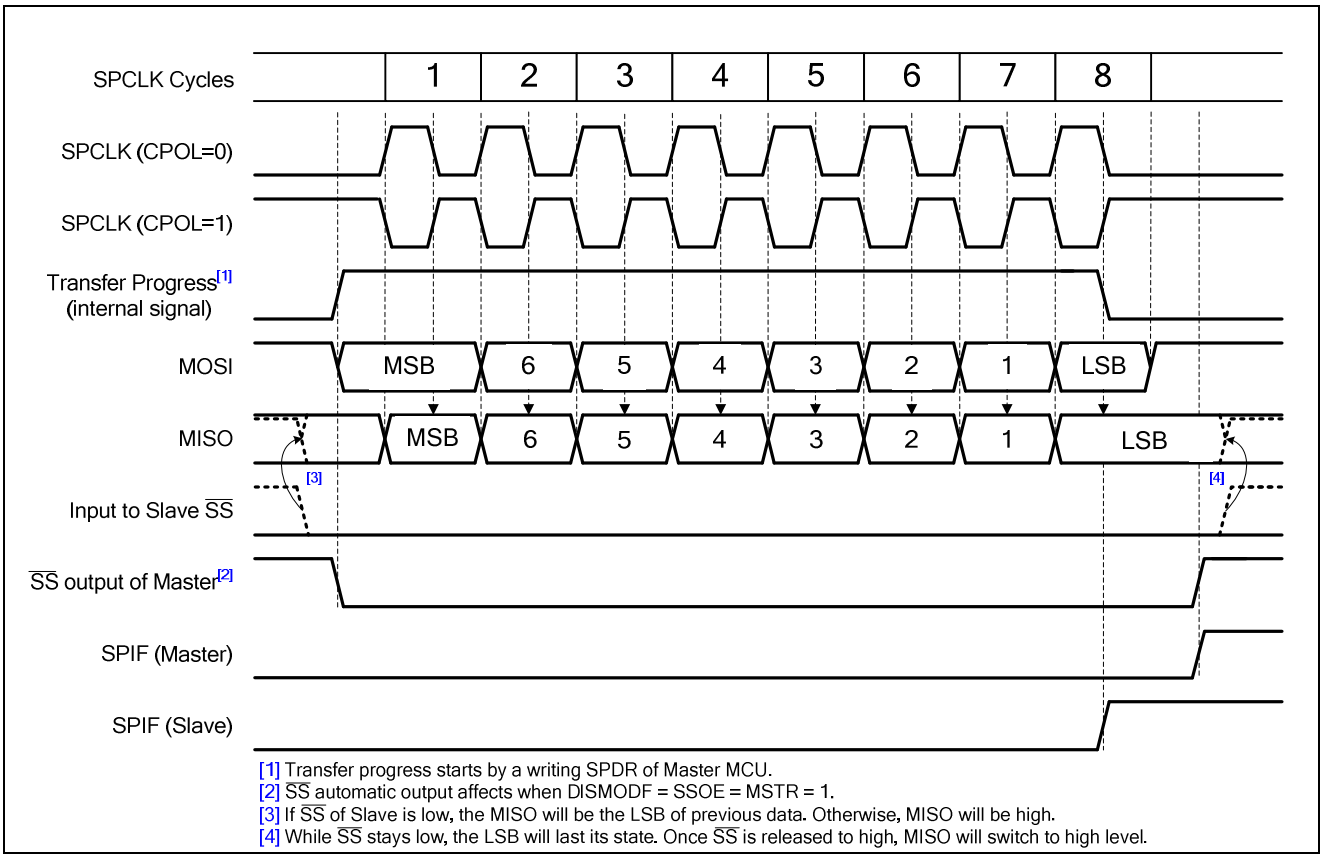


Figure 13–6. SPI Clock and Data Format with CPHA = 1

13.6 Slave Select Pin Configuration

N79E845 series SPI give a flexible $\overline{SS}$ pin feature for different system requirements. When the SPI operates as a Slave, $\overline{SS}$ pin always rules as Slave select input. When the Master mode is enabled, $\overline{SS}$ has three different functions according to DISMODF (SPSR.3) and SSOE (SPCR.7). By default, DISMODF is 0. It means that the Mode Fault detection activates. $\overline{SS}$ is configured as a input pin to check if the Mode Fault appears. On the contrary, if DISMODF is 1, Mode Fault is inactivated and the SSOE bit takes over to control the function of the $\overline{SS}$ pin. While SSOE is 1, it means the Slave select signal will generate automatically to select a Slave device. The $\overline{SS}$ as output pin of the Master usually connects with the $\overline{SS}$ input pin of the Slave device. The $\overline{SS}$ output automatically goes low for each transmission when selecting external Slave device and goes high during each idle state to de-select the Slave device. While SSOE is 0 and DISMODF is 1, $\overline{SS}$ is no more used by the SPI and reverts to be a general purpose I/O pin.

13.7 Mode Fault Detection

The Mode Fault detection is useful in a system where more than one SPI devices might become Masters at the same time. It may induce data contention. A Mode Fault error occurs once the $\overline{SS}$ is pulled low by others. It indicates that some other SPI device is trying to address this Master as if it is a Slave. Instantly the MSTR and SPIEN control bits in the SPCR are cleared via hardware to disable SPI, Mode Fault flag MODF (SPSR.4) is set and an interrupt is generated if ESPI (EIE .6) and EA are enabled.

13.8 Write Collision Error

The SPI is signal buffered in the transfer direction and double buffered in the receiving direction. New data for transmission cannot be written to the shift register until the previous transaction is complete. Write collision occurs while an attempt was made to write data to the SPDR while a transfer was in progress. SPDR is not double buffered in the transmit direction. Any writing to SPDR cause data to be written directly into the SPI shift register. Once a write collision error is generated, WCOL (SPSR.6) will be set as a 1 via hardware to indicate a write collision. In this case, the current transferring data continues its transmission. However the new data that caused the collision will be lost. Although the SPI logic can detect write collisions in both Master and Slave modes, a write collision is normally a Slave error because a Slave has no indicator when a Master initiates a transfer. During the receive of Slave, a write to SPDAT causes a write collision under Slave mode. WCOL flag needs to be cleared via software.

13.9 Overrun Error

For receiving data, the SPI is double buffered in the receiving direction. The received data is transferred into a parallel read data buffer so the shifter is free to accept a second serial byte. However, the received data must be read from SPDR before the next data has been completely shifted in. As long as the first byte is read out of the read data buffer and SPIF is cleared before the next byte is ready to be transferred, no overrun error condition occurs. Otherwise the overrun error occurs. In this condition, the second byte data will not be successfully received into the read data register and the previous data will remains. If overrun occur, SPIOVF (SPSR.5) will be set via hardware. This will also require an interrupt if enabled. Figure 13–7. SPI Overrun Waveform shows the relationship between the data receiving and the overrun error.

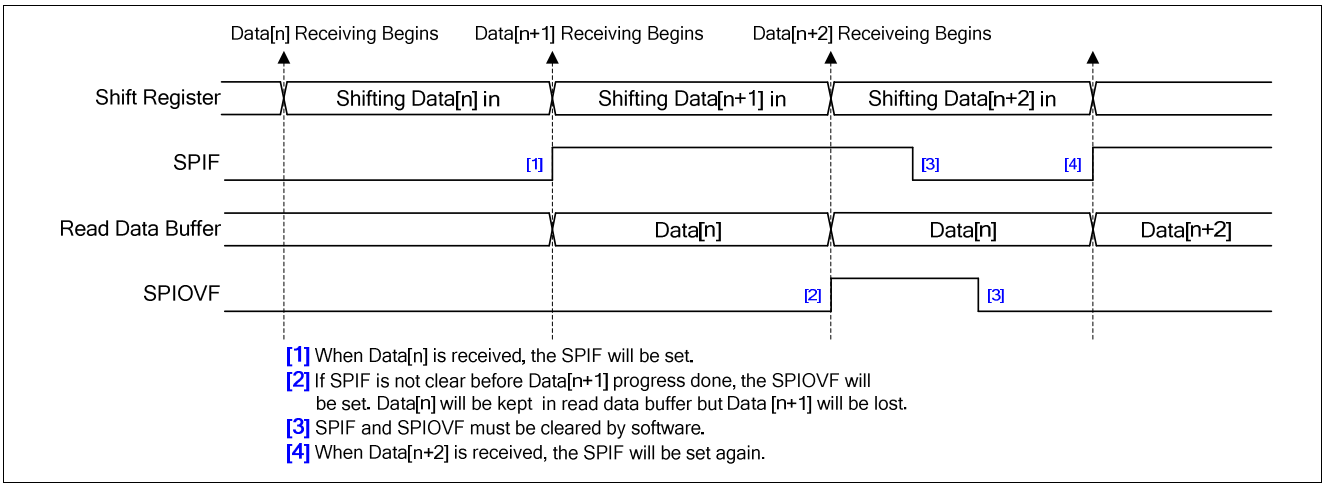


Figure 13–7. SPI Overrun Waveform

13.10 SPI Interrupts

Three SPI status flags, SPIF, MODF, and SPIOVF, can generate an SPI event interrupt requests. All of them locate in SPSR. SPIF will be set after completion of data transfer with external device or a new data have been received and copied to SPDR. MODF becomes set to indicate a low level on $\overline{SS}$ causing the Mode Fault state. SPIOVF denotes a receiving overrun error. If SPI interrupt mask is enabled via setting ESPI (EIE.6) and EA is 1, CPU will executes the SPI interrupt service routine once any of these three flags is set. The user needs to check flags to determine what event caused the interrupt. These three flags are software cleared.

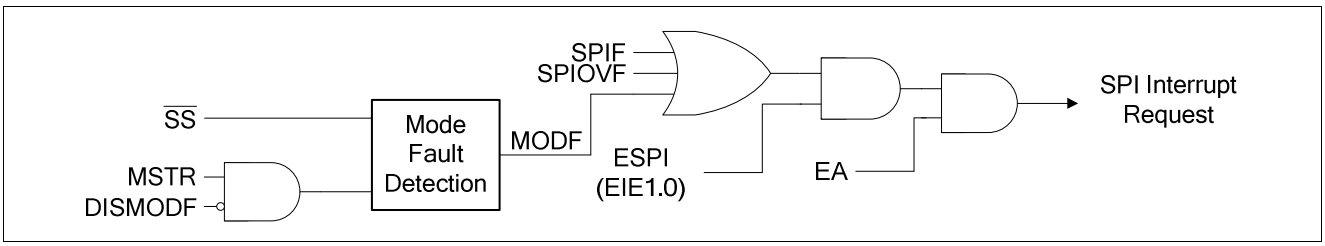


Figure 13–8. SPI Interrupt Request

14 Keyboard Interrupt (KBI)

The **N79E845** series provide 8 keyboard interrupt function to detect keypad status which key is acted, and allow a single interrupt to be generated when any key is pressed on a keyboard or keypad connected to specific pins of the **N79E845** series, as shown below figure. This interrupt may be used to wake up the CPU from Idle or Power Down modes, after chip is in Power Down or Idle Mode.

Keyboard function is supported through by Port 0. It can allow any or all pins of Port 0 to be enabled to cause this interrupt. Port pins are enabled by the setting of bits of KBI0 ~ KBI7 in the KBI register, as shown below figure. The Keyboard Interrupt Flag, KBIF[7:0] in the KBIF(EAH), is set when any enabled pin is triggered while the KBI interrupt function is active, an interrupt will be generated if it has been enabled. The KBIF[7:0] bit is set by hardware and must be cleared by software. In order to determine which key was pressed, the KBI will allow the interrupt service routine to poll port 0.

KBI supports four triggered conditions which are low level, falling edge, rising edge and either rising or falling edge detection. The triggered condition of each port pin is individually controlled by two bits KBIS1(ECH).x and KBIS0(EBH).x where x is 0 to 7.

After Trigger occur and two machines pass, KBIF assert.

KBI is generally used to detect an edge transient from peripheral devices like keyboard or keypad. During idle state, the system prefers to enter Power Down mode to minimize power consumption and waits for event trigger. The **N79E845** series support KBI interrupt waking up MCU from Power Down. Note that if KBI is selected as any of edge trigger mode, restrictions must be followed to make Power Down woken up valid. For a falling edge waking up, pin state should be high at the moment of entering Power Down mode. Respectively, pin state should be low for a rising edge waking up.

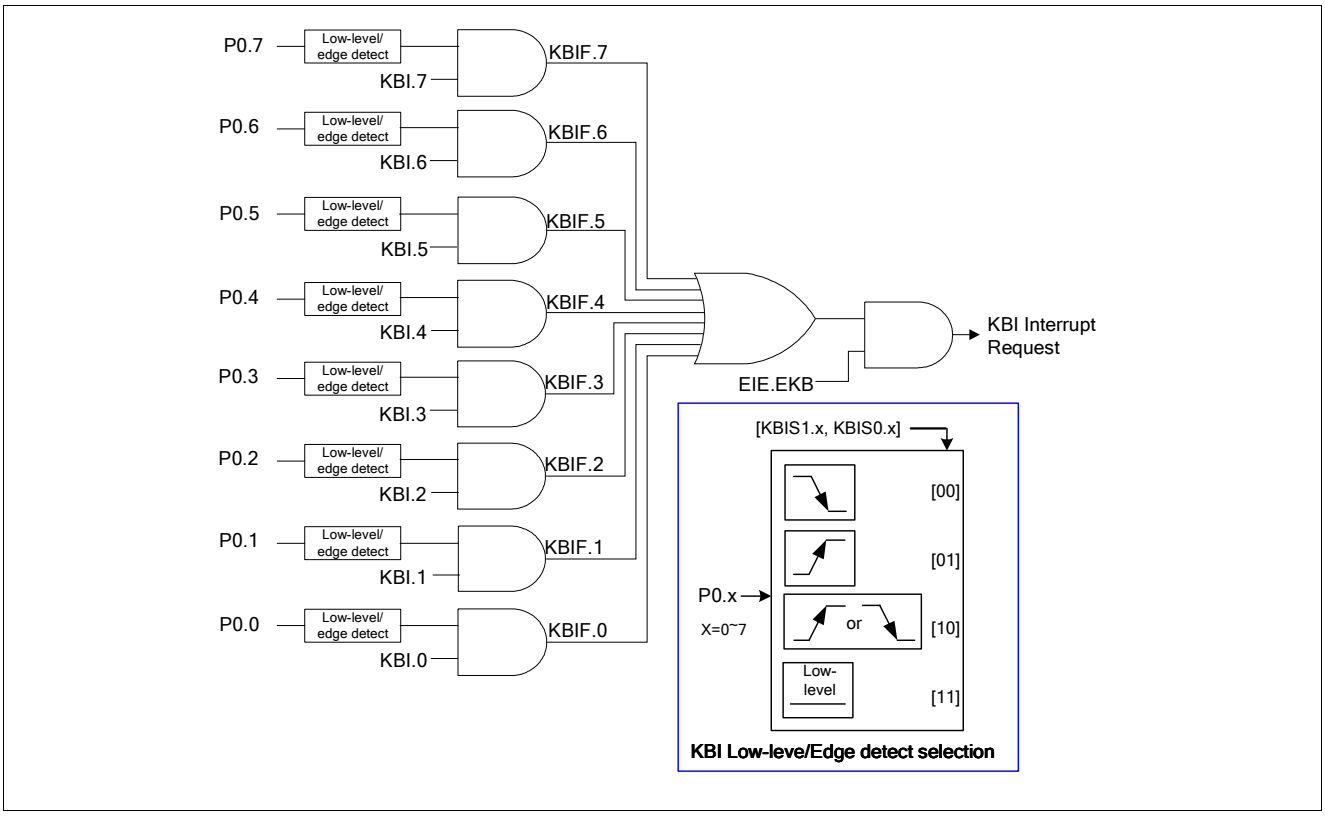


Figure 14-1 Keyboard Interrupt Detect

Table 14–1. Configuration for Different KBI Level Select

KBLS1.n	KBLS0.n	KBI Channel n Type
0	0	Falling edge
0	1	Rising edge
1	0	Either falling or rising edge
1	1	Low level

KBIE – Key Board Interrupt Enable Register

7	6	5	4	3	2	1	0
KBIE.7	KBIE.6	KBIE.5	KBIE.4	KBIE.3	KBIE.3	KBIE.1	KBIE.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: E9H reset value: 0000 0000B

Bit	Name	Description
7:0	**KBIE**	**Key Board Interrupt** Enable P0[7:0] as a cause of a Keyboard interrupt.

KBIF – Keyboard Interface Flags

7	6	5	4	3	2	1	0
KBIF7	KBIF6	KBIF5	KBIF4	KBIF3	KBIF2	KBIF1	KBIF0
r (level) r/w (edge)	r (level) r/w (edge)	r (level) r/w (edge)	r (level) r/w (edge)	r (level) r/w (edge)	r (level) r/w (edge)	r (level) r/w (edge)r	r (level) r/w (edge)

Address: EAH reset value: 0000 0000B

Bit	Name	Description
7:0	KBIFn	**Keyboard interface channel n flag.** If any edge trigger mode of KBI is selected, this flag will be set by hardware if KBI channel n (P0.n) detects a type defined edge. This flag should be cleared by soft-ware. If the low level trigger mode of KBI is selected, this flag follows the inverse of the input signal's logic level on KBI channel n (P0.n)l. Software cannot control it.

KBLS0 – Keyboard Level Select 0[1]

7	6	5	4	3	2	1	0
KBLS0.7	KBLS0.6	KBLS0.5	KBLS0.4	KBLS0.3	KBLS0.2	KBLS0.1	KBLS0.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: EBH reset value: 0000 0000B

Bit	Name	Description
7:0	KBLS0[7:0]	**Keyboard level select 0.**

KBLS1 – Keyboard Level Select 1[1]

7	6	5	4	3	2	1	0
KBLS1.7	KBLS1.6	KBLS1.5	KBLS1.4	KBLS1.3	KBLS1.2	KBLS1.1	KBLS1.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: ECH reset value: 0000 0000B

Bit	Name	Description
7:0	KBLS1[7:0]	**Keyboard level select 1.**

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

15 Analog-To-Digital Converter (ADC)

The ADC contains a DAC which converts the contents of a successive approximation register to a voltage (V_{DAC}) which is compared to the analog input voltage (Vin). The output of the comparator is fed to the successive approximation control logic which controls the successive approximation register. A conversion is initiated by setting ADCS in the ADCCON0 register. ADCS can be set by software only or by either hardware or software.

Be noticed that when the ADC function is disabled, all ADC related SFR bits will be unavailable and will not effect any other CPU functions.
The power of ADC block is approached to zero.

The software only start mode is selected when control bit ADCCON0.5 (ADCEX) =0. A conversion is then started by setting control bit ADCCON0.3 (ADCS) The hardware or software start mode is selected when ADCCON0.5 (ADCEX) =1, and a conversion may be started by setting ADCCON0.3 as above or by applying a rising edge to external pin STADC. When a conversion is started by applying a rising edge, a low level must be applied to STADC for at least one machine-cycle followed by a high level for at least one machine-cycle.

The low-to-high transition of STADC is recognized at the end of a machine-cycle, and the conversion commences at the beginning of the next cycle. When a conversion is initiated by software, the conversion starts at the beginning of the machine-cycle which follows the instruction that sets ADCS. ADCS is actually implemented with tpw flip-flops: a command flip-flop which is affected by set operations, and a status flag which is accessed during read operations.

The next two machine-cycles are used to initiate the converter. At the end of the first cycle, the ADCS status flag is set end a value of “1” will be returned if the ADCS flag is read while the conversion is in progress. Sampling of the analog input commences at the end of the second cycle.

During the next eight machine-cycles, the voltage at the previously selected pin of port 5 is sampled, and this input voltage should be stable in order to obtain a useful sample. In any event, the input voltage slew rate must be less than 10V/ms in order to prevent an undefined result.

The successive approximation control logic first sets the most significant bit and clears all other bits in the successive approximation register (10 0000 0000b). The output of the DAC (50% full scale) is compared to the input voltage Vin. If the input voltage is greater than V_{DAC}, then the bit remains set; otherwise if is cleared.

The successive approximation control logic now sets the next most significant bit (11 0000 0000b or 01 0000 0000b, depending on the previous result), and the V_{DAC} is compared to Vin again. If the input voltage is greater then V_{DAC}, then the bit remains set; otherwise it is cleared. This process is repeated until all ten bits have been tested, at which stage the result of the conversion is held in the successive approximation register. The conversion takes four machine-cycles per bit.

The end of the 10-bit conversion is flagged by control bit ADCCON0.4 (ADCI). The upper 8 bits of the result are held in special function register ADCH, and the two remaining bits are held in ADCCON0.7 (ADC.1) and ADCCON0.6 (ADC.0). The user may ignore the two least significant bits in ADCCON0 and use the ADC as an 8-bit converter (8 upper bits in ADCH). In any event, the total actual conversion time is 35 machine-cycles. ADC will be set and the ADCS status flag will be reset 35 cycles after the ADCS is set.
Control bits ADCCON0.0 ~ ADCCON0.2 are used to control an analog multiplexer which selects one of 8 analog channels. An ADC conversion in progress is unaffected by an external or software ADC start. The result of a completed conversion remains unaffected provided ADCI = logic 1; a new ADC conversion already in progress is aborted when the idle or power down mode is entered. The result of a completed conversion (ADCI = logic 1) remains unaffected when entering the idle mode.

When ADCCON0.5 (ADCEX) is set by external pin to start ADC conversion, after **N79E845** series entry idle mode, P1.4 can start ADC conversion at least ONE machine-cycle.

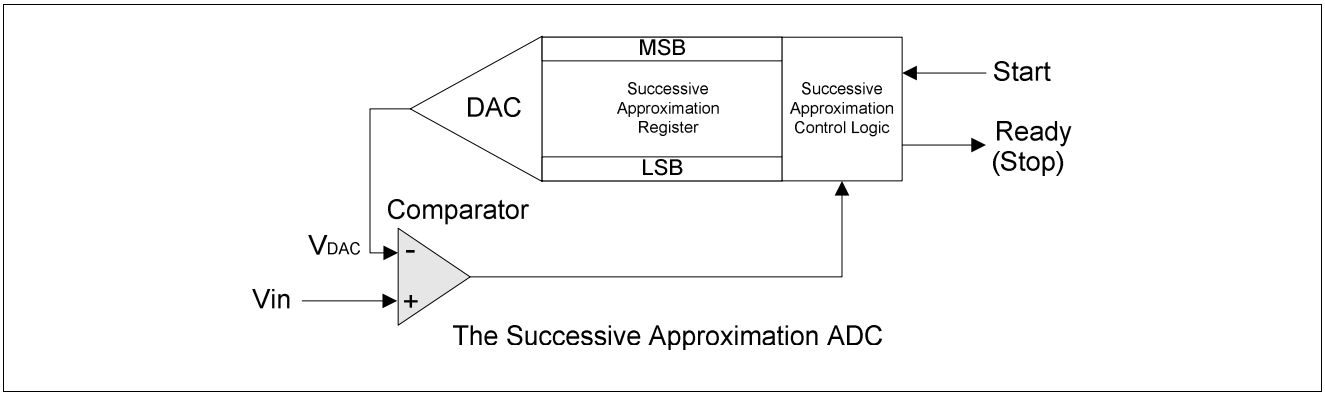


Figure 15-1 Successive Approximation ADC

15.1 ADC Resolution and Analog Supply

The ADC circuit has its own supply pins (AVDD and AVSS) and one pins (Vref+) connected to each end of the DAC's resistance-ladder that the AVDD and Vref+ are connected to VDD and AVSS is connected to V_{SS}. The ladder has 1023 equally spaced taps, separated by a resistance of "R". The first tap is located 0.5×R above Avss, and the last tap is located 0.5×R below Vref+. This gives a total ladder resistance of 1024×R. This structure ensures that the DAC is monotonic and results in a symmetrical quantization error.

For input voltages between Avss and [(Vref+) + ½ LSB], the 10-bit result of an A/D conversion will be 0000000000B = 000H. For input voltages between [(Vref+) – 3/2 LSB] and Vref+, the result of a conversion will be 1111111111B = 3FFH. Avref+ and AVSS may be between AVDD + 0.2V and AVSS – 0.2 V. Avref+ should be positive with respect to AVSS, and the input voltage (Vin) should be between Avref+ and AVSS.

The result can always be calculated from the following formula:

$$\text{Result} = 1024 \times \frac{\text{Vin}}{\text{AVref}+} \quad \text{or Result} = 1024 \times \frac{\text{Vin}}{\text{VDD}}$$

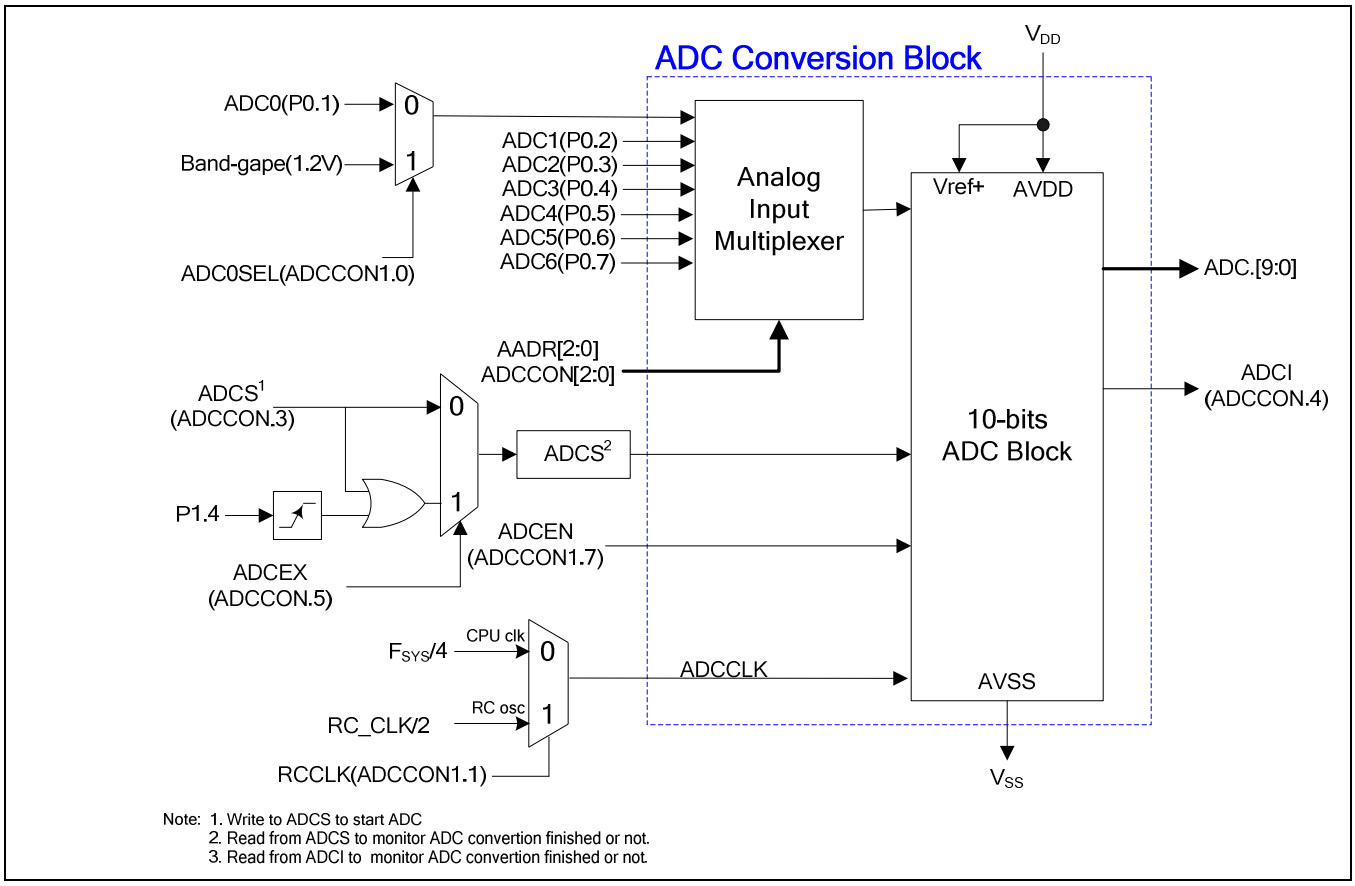


Figure 15-2 ADC Block Diagram

ADCCON0 – ADC Control Register

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: XX00 0X00B

Bit	Name	Description
7	**ADC.1**	The ADC conversion result.
6	**ADC.0**	The 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 al 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**	The ADC input select.
1	**AADR1**	The ADC input select.
0	**AADR0**	The 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; Start of a new conversion requires ADCI = 0
1	1	Conversion completed; 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. But 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)
0	0	1	ADC1 (P0.2)

0	1	0	ADC2 (P0.3)
0	1	1	ADC3 (P0.4)
1	0	0	ADC4 (P0.5)
1	0	1	ADC5 (P0.6)
1	1	0	ADC6 (P0.7)

ADCH – ADC Converter Result Register

7	6	5	4	3	2	1	0
ADC.9	ADC.8	ADC.7	ADC.6	ADC.5	ADC.4	ADC.3	ADC.2
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: E2H reset value: 0000 0000B

Bit	Name	Description
7:0	**ADCH**	The ADC conversion result bits [9:2].

ADCCON1 – ADC Control Register

7	6	5	4	3	2	1	0
ADCEN	-	-	-	-	-	RCCLK	ADC0SEL
r/w	-	-	-	-	-	r/w	r/w

Address: E1H reset value: 0000 0000B

Bit	Name	Description
7	**ADCEN**	0: Disable ADC circuit. 1: Enable ADC circuit.
6:2	**-**	**Reserved**
1	**RCCLK**	0: The F_{SYS}/4 clock is used as ADC clock. 1: The internal RC/2 clock is used as ADC clock.
0	**ADC0SEL**	0: Select ADC channel 0 as input. 1: Select Band-gape (~1.3V) as input.

16 Inter-Integrated Circuit (I^2C)

16.1 Features

The Inter-Integrated Circuit (I^2C) bus serves as an 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 bi-directional 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 Function Description

For a bi-directional transfer operation, the SDA and SCL pins must 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 **N79E845** series, 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. The Schmitt trigger input buffer is strongly recommended to be enabled by setting P1S for improved glitch suppression.

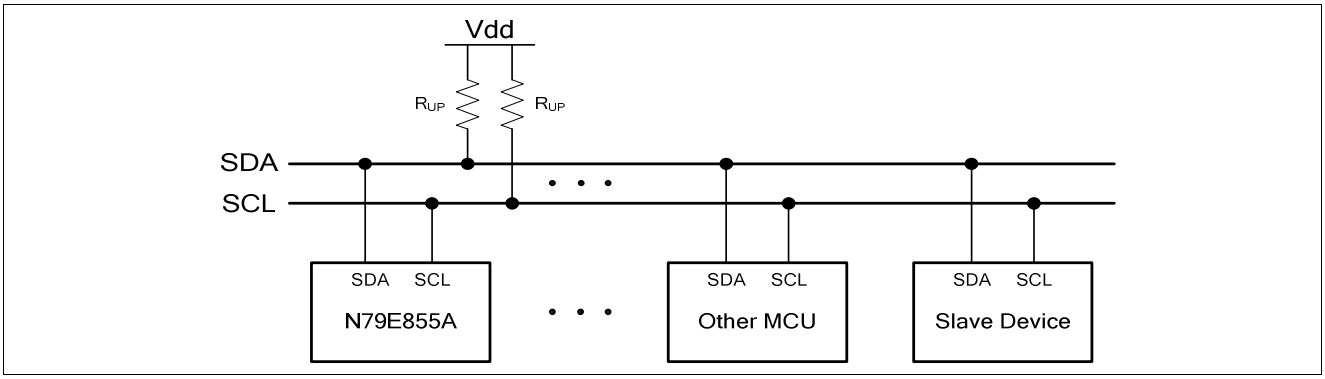


Figure 16–1. I²C bus Interconnection

The I²C is considered free when both lines are high. Meanwhile, any device which can operate as a master can occupy the bus and generate one transfer after generating a START condition. The bus now is considered busy before the transfer ends by sending a STOP condition. The master generates all of the serial clock pulses and the START and STOP condition. However if there is no START condition on the bus, all devices serve as not addressed slave. The hardware looks for its own slave address or a General Call address. (The General Call address detection may be enabled or disabled by GC (I2ADDR.0).) If the matched address is received, an interrupt is requested.

Every transaction on the I²C bus is 9 bits long, consisting of 8 data bits (MSB first) and a single acknowledge bit. The number of bytes per transfer (defined as the time between a valid START and STOP condition) is unrestricted but each byte has to be followed by an acknowledge bit. The master device generates 8 clock pulse to send the 8-bit data. After the 8th falling edge of the SCL line, the device outputting data on the SDA changes that pin to an input and reads in an acknowledge value on the 9th clock pulse. After 9th clock pulse, the data receiving device can hold SCL line stretched low if next receiving is not prepared ready. It forces the next byte transaction suspended. The data transaction continues when the receiver releases the SCL line.

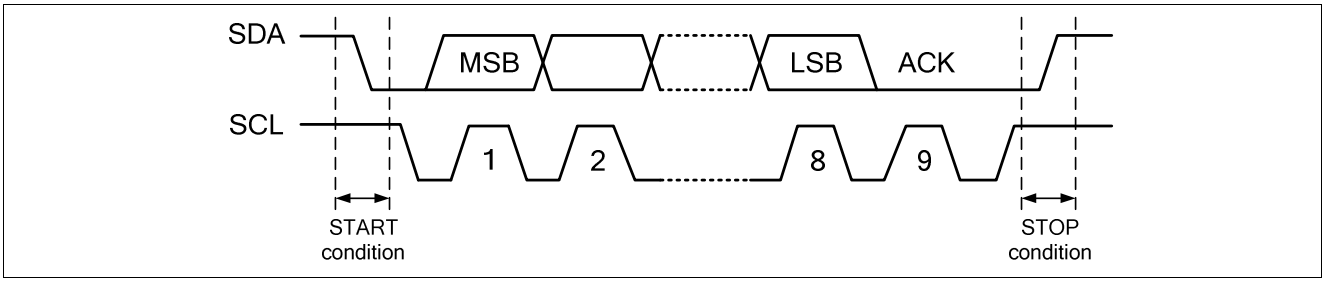


Figure 16–2. I²C Bus Protocol

16.2.1 START and STOP Condition

The protocol of the I²C bus defines two states to begin and end a transfer, START (S) and STOP (P) conditions. A START condition is defined as a high-to-low transition on the SDA line while SCL line is high. The STOP condition is defined as a low-to-high transition on the SDA line while SCL line is high. A START or a STOP condition is always generated by the master and I²C bus is considered busy after a START condition and free after a STOP condition. After issuing the STOP condition successful, the original master device will release the control authority and turn back as a not addressed slave. Consequently, the original addressed slave will become a not addressed slave. The I²C bus is free and listens to next START condition of next transfer.

A data transfer is always terminated by a STOP condition generated by the master. However, if a master still wishes to communicate on the bus, it can generate a repeated START (Sr) condition and address the pervious or another slave without first generating a STOP condition. Various combinations of read/write formats are then possible within such a transfer.

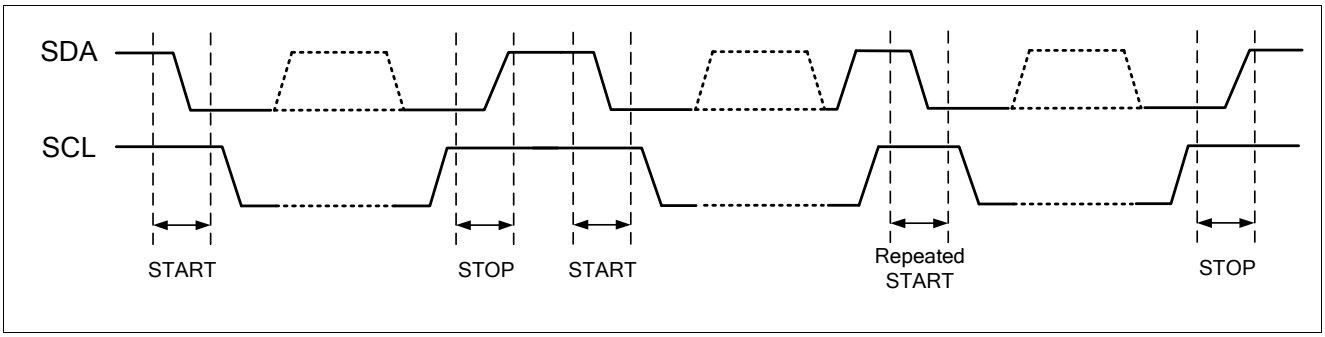


Figure 16–3. START, Repeated START, and STOP Conditions

16.2.2 7-bit Address with Data Format

Following the START condition is generated, one byte of special data should be transmitted by the master. It includes a 7-bit long slave address (SLA) following by an 8th bit, which is a data direction bit (R/W), to address the target slave device and determine the direction of data flow. If R/W bit is 0, it indicates that the master will write information to a selected slave, and if this bit is 1, it indicates that the master will read information from the slave. An address packet consisting of a slave address and a read (R) or a write (W) bit is called SLA+R or SLA+W, respectively. A transmission basically consists of a START condition, a SLA+R/W, one or more data packets and a STOP condition. After the specified slave is addressed by SLA+R/W, the second and following 8-bit data bytes issue by the master or the slave devices according to the R/W bit configuration.

There is an exception called “General Call” address which can address all devices by giving the first byte of data all 0. A General Call is used when a master wishes to transmit the same message to several slaves in the system. When this address is used, other devices may respond with an acknowledge or ignore it according to individual software configuration. If a device response the General Call, it operates as like in the slave-receiver mode.

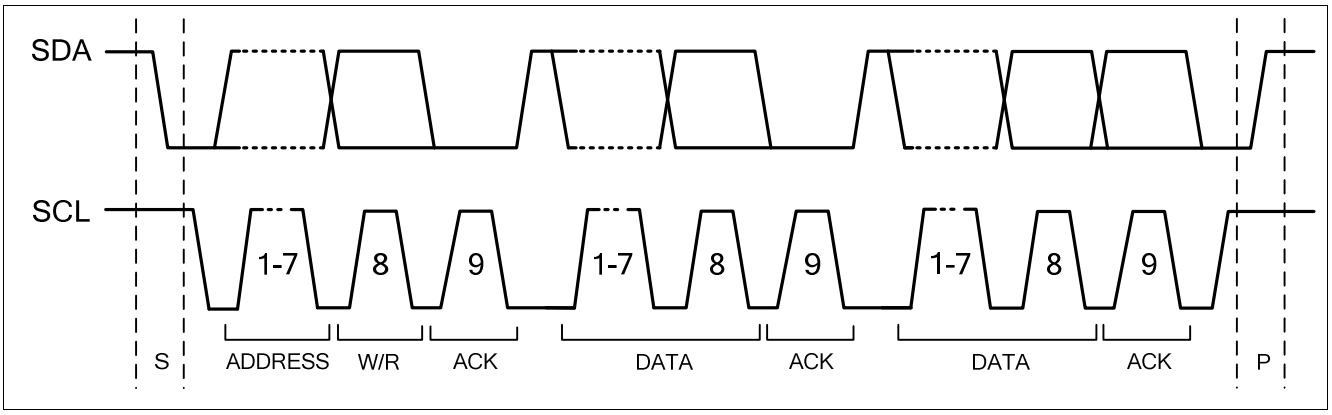


Figure 16–4. Data Format of an I²C Transfer

During the data transaction period, the data on the SDA line must be stable during the high period of the clock, and the data line can only change when SCL is low.

16.2.3 Acknowledge

The 9th SCL pulse for any transferred byte is dedicated as an Acknowledge (ACK). It allows receiving devices (which can be the master or slave) to respond back to the transmitter (which also can be the master or slave) by pulling the SDA line low. The acknowledge-related clock pulse is generated by the master. The transmitter must release control of SDA line during the acknowledge clock pulse. The ACK is an active-low signal, pulling the SDA line low during the clock pulse high duty, indicates to the transmitter that the device has received the transmitted data. Commonly, a receiver which has been addressed is requested to generate an ACK after each byte has been received. When a slave receiver does not acknowledge (NACK) the slave address, the SDA line must be left high by the slave so that the mater can generate a STOP or a repeated START condition.

If a slave-receiver does acknowledge the slave address, it switches itself to not addressed slave mode and cannot receive any more data bytes. This slave leaves the SDA line high. The master should generate a STOP or a repeated START condition.

If a master-receiver is involved in a transfer, because the master controls the number of bytes in the transfer, it must signal the end of data to the slave-transmitter by not generating an acknowledge on the last byte. The slave-transmitter then switches to not addressed mode and release the SDA line to allow the master to generate a STOP or a repeated START condition.

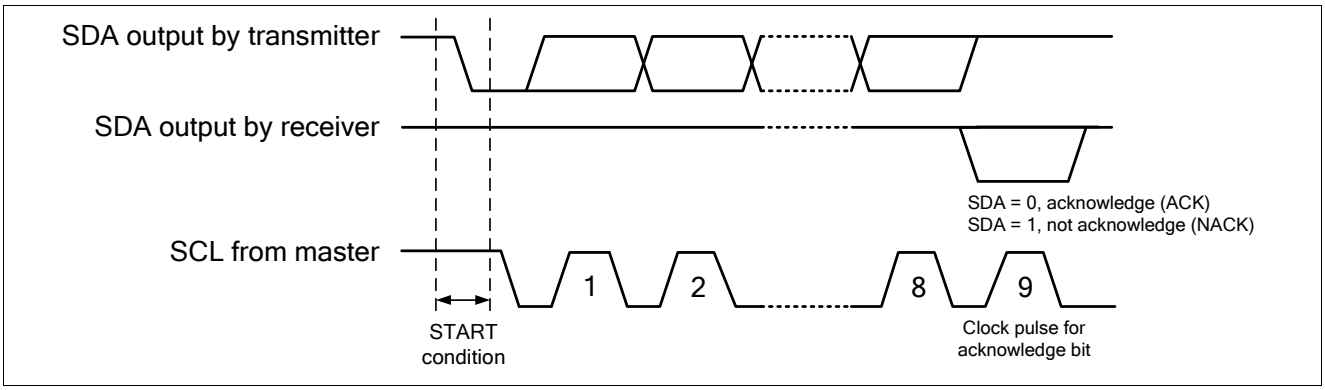


Figure 16–5. Acknowledge Bit

16.2.4 Arbitration

A master may start a transfer only if the bus is free. It is possible for two or more masters to generate a START condition. In these situations, an arbitration scheme takes place on the SDA line, while SCL is high. During arbitration, the first of the competing master devices to place a '1' (high) on SDA while another master transmits a '0' (low) switches off its data output stage because the level on the bus does not match its own level. The arbitration lost master switches to the not addressed slave immediately to detect its own slave address in the same serial transfer whether it is being addressed by the winning master. It also releases SDA line to high level for not affecting the data transfer initiated by the winning master. However, the arbitration lost master continues SCL line to generate the clock pulses until the end of the byte in which it loses the arbitration. If the address matches the losing master's own slave address, it switches to the addressed-slave mode.

Arbitration is carried out by all masters continuously monitoring the SDA line after outputting data. If the value read from the SDA line does not match the value the master had output, it has lost the arbitration. Note that a master can only lose arbitration when it outputs a high SDA value while another master outputs a low value. Arbitration will continue until only one master remains, and this may take many bits. If several masters are trying to address the same slave, arbitration will continue into the data packet.

Arbitration can take place over several bits. Its first stage is a comparison of address bits, and if both masters are trying to address the same device, arbitration continues on to the comparison of data bits or acknowledge bit.

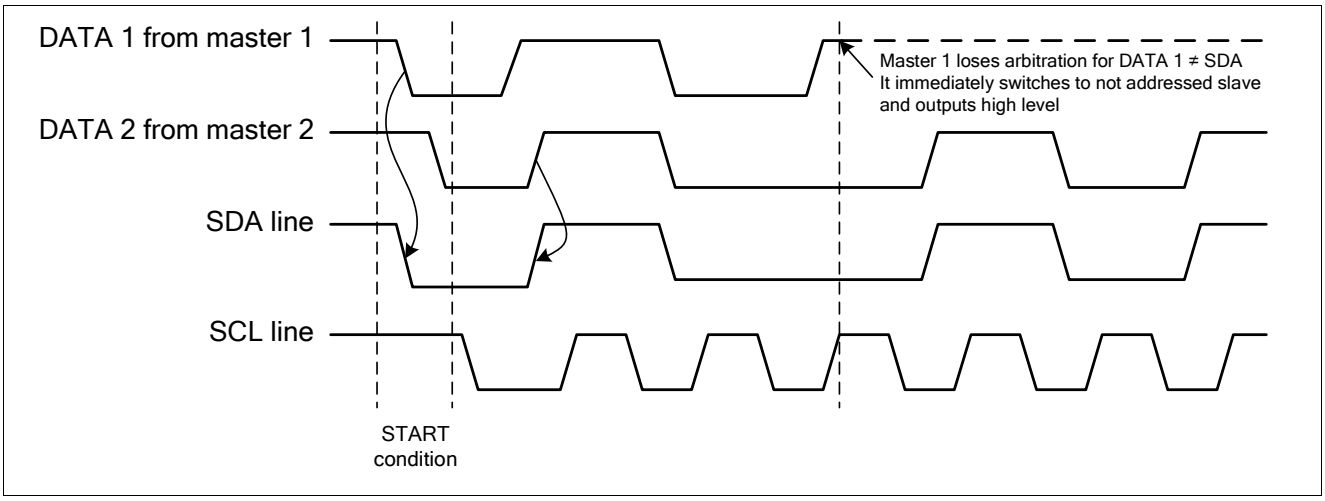


Figure 16–6. Arbitration Procedure of Two Masters

Since control of the I²C bus is decided solely on the address or master code and data sent by competing masters, there is no central master, nor any order of priority on the bus.

Slaves are not involved in the arbitration procedure.

16.3 Control Registers of I²C

There are five control registers to interface the I²C bus. They are I2CON, I2STAT, I2DAT, I2ADDR, I2CLK, and I2TMR. These registers provide protocol control, status, data transmit and receive functions, clock rate configuration, and timeout notification. The following registers relate to I²C function.

I2CON – I²C Control

7	6	5	4	3	2	1	0
-	I2CEN	STA	STO	SI	AA	-	-
-	r/w	r/w	r/w	r/w	r/w	-	-

Address: C0H reset value: 0000 0000B

Bit	Name	Description
7	-	**Reserved.**
6	I2CEN	**I²C bus enable.** 0 = The I²C bus is disabled. 1 = The I²C bus is enabled. Before enabling the I²C, Px.x and Px.x port latches must be set to logic 1. Once the I²C bus is enabled, SDA pin (Px.x) and SCL pin (Px.x) will be automatically switched to the open-drain mode. PxM2 and PxM1 registers will also be re-configured accordingly.

Bit	Name	Description
5	STA	**START flag.** When STA is set, the I²C generates a START condition if the bus is free. If the bus is busy, the I²C waits for a STOP condition and generates a START condition following. If STA is set while the I²C is already in the master mode and one or more bytes have been transmitted or received, the I²C generates a repeated START condition. Note that STA can be set anytime even in a slave mode, but STA is not hardware automatically cleared after START or repeated START condition has been detected. The user should take care of it by clearing STA manually.
4	STO	**STOP flag.** When STO is set if the I²C is in the master mode, a STOP condition is transmitted to the bus. STO is automatically cleared by hardware once the STOP condition has been detected on the bus. The STO flag setting is also used to recover the I²C device from the bus error state (I2STAT as 00H). In this case, no STOP condition is transmitted to the I²C bus. If the STA and STO bits are both set and the device is original in the master mode, the I²C bus will generate a STOP condition and immediately follow a START condition. If the device is in slave mode, STA and STO simultaneous setting should be avoid from issuing illegal I²C frames.
3	SI	**Serial interrupt flag.** SI flag is set by hardware when one of 25 possible I²C status (besides F8H status) is entered. After SI is set, the software should read I2STAT register to determine which step has been passed and take actions for next step. SI is cleared by software. Before the SI is cleared, the low period of SCL line is stretched. The transaction is suspended. It is useful for the slave device to deal with previous data bytes until ready for receiving the next byte. The serial transaction is suspended until SI is cleared by software. After SI is cleared, I²C bus will continue to generate START or repeated START condition, STOP condition, 8-bit data, or so on depending on the software configuration of controlling byte or bits. Therefore the user should take care of it by preparing suitable setting of registers before SI is software cleared.
2	AA	**Acknowledge assert flag.** If the AA flag is set, an ACK (low level on SDA) will be returned during the acknowledge clock pulse of the SCL line while the I²C device is a receiver which can be a master, an addressed slave, an own-address-matching slave, or a Genera-Call acceptable slave. If the AA flag is cleared, a NACK (high level on SDA) will be returned during the acknowledge clock pulse of the SCL line while the I²C device is a receiver which can be a master, an addressed slave. A device with its own AA flag cleared will ignore its own salve address and the General Call. Consequently, SI will note be asserted and no interrupt is requested. Note that if an addressed slave does not return an ACK under slave receiver mode or not receive an ACK under slave transmitter mode, the slave device will become a not addressed slave. It cannot receive any data until its AA flag is set and a master addresses it again. There is a special case of I2STAT value C8H occurs under slave transmitter mode. Before the slave device transmit the last data byte to the master, AA flag can be cleared as 0. Then after the last data byte transmitted, the slave device will actively switch to not addressed slave mode of disconnecting with the master. The further reading of the master will be all FFH.
1:0	-	**Reserved.**

I2STAT – I²C Status

7	6	5	4	3	2	1	0
I2STAT[4:0]					0	0	0
r					r	r	r

Address: BDH reset value: 1111 1000B

Bit	Name	Description
7:3	I2STAT[4:0]	**I²C status code.** The most five bits of I2STAT contains the status code. There are 26 possible status codes. When I2STAT is F8H, no relevant state information is available and SI flag keeps 0. All other 25 status codes correspond to the I²C states. When each of these status is entered, SI will be set as logic 1 and a interrupt is requested.
2:0	-	**Reserved.** The least three bits of I2STAT are always read as 0.

I2DAT – I²C 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]	**I²C data.** I2DAT contains a byte of the I²C data to be transmitted or a byte which has just received. Data in I2DAT remains as long as SI is logic 1. The result of reading or writing I2DAT during I²C transceiving progress is unpredicted. While data in I2DAT is shifted out, data on the bus is simultaneously being shifted in to update I2DAT. I2DAT always shows the last byte that presented on the I²C bus. Thus the event of lost arbitration, the original value of I2DAT changes after the transaction.

I2ADDR – I²C Own Slave Address

7	6	5	4	3	2	1	0
I2ADDR[6:0]							GC
r/w							r/w

Address: C1H reset value: 0000 0000B

Bit	Name	Description
7:1	I2ADDR[6:0]	**I²C device's own slave address.** In master mode: These bits have no effect. In slave mode: These 7 bits define the slave address of this I²C device by the user. The master should address this I²C 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 I²C 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
6	GC	**General Call bit.** In master mode: This bit has no effect. In slave mode: 0 = The General Call is always ignored. 1 = The General Call is recognized if AA flag is 1; otherwise, it is ignored if AA is 0.

I2CLK – I²C 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]	**I²C clock setting.** In master mode: This register determines the clock rate of I²C bus when the device is in a master mode. The clock rate follows the formula below. $F_{I^2C} = \frac{F_{PHERI}}{1+I2CLK}$ The default value will make the clock rate of I²C bus 400kbps if the system clock 24MHz 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 I²C device will automatically synchronize with any given clock rate up to 400kps.

16.4 Modes of Operation

In I²C protocol definition, 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 operating is similar to master transmitter mode.

16.4.1 Master Transmitter Mode

In the 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 I²C 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 I2STAT show 08H. The progress is contin-

ued by loading I2DAT with the target slave address and the data direction bit “write” (SLA+W). The SI bit must then be cleared to commence SLA+W transaction.

After the SLA+W byte has been transmitted and an acknowledge (ACK) has been returned by the addressed slave device, the SI flag is set again and I2STAT is read as 18H. The appropriate action to be taken follows the user defined communication protocol by sending data continuously. After all data is transmitted, the master can send a STOP condition by setting STO (I2CON.4) and then clearing SI to terminate the transmission. A repeated START condition can be also generated without sending STOP condition to immediately initial another transmission.

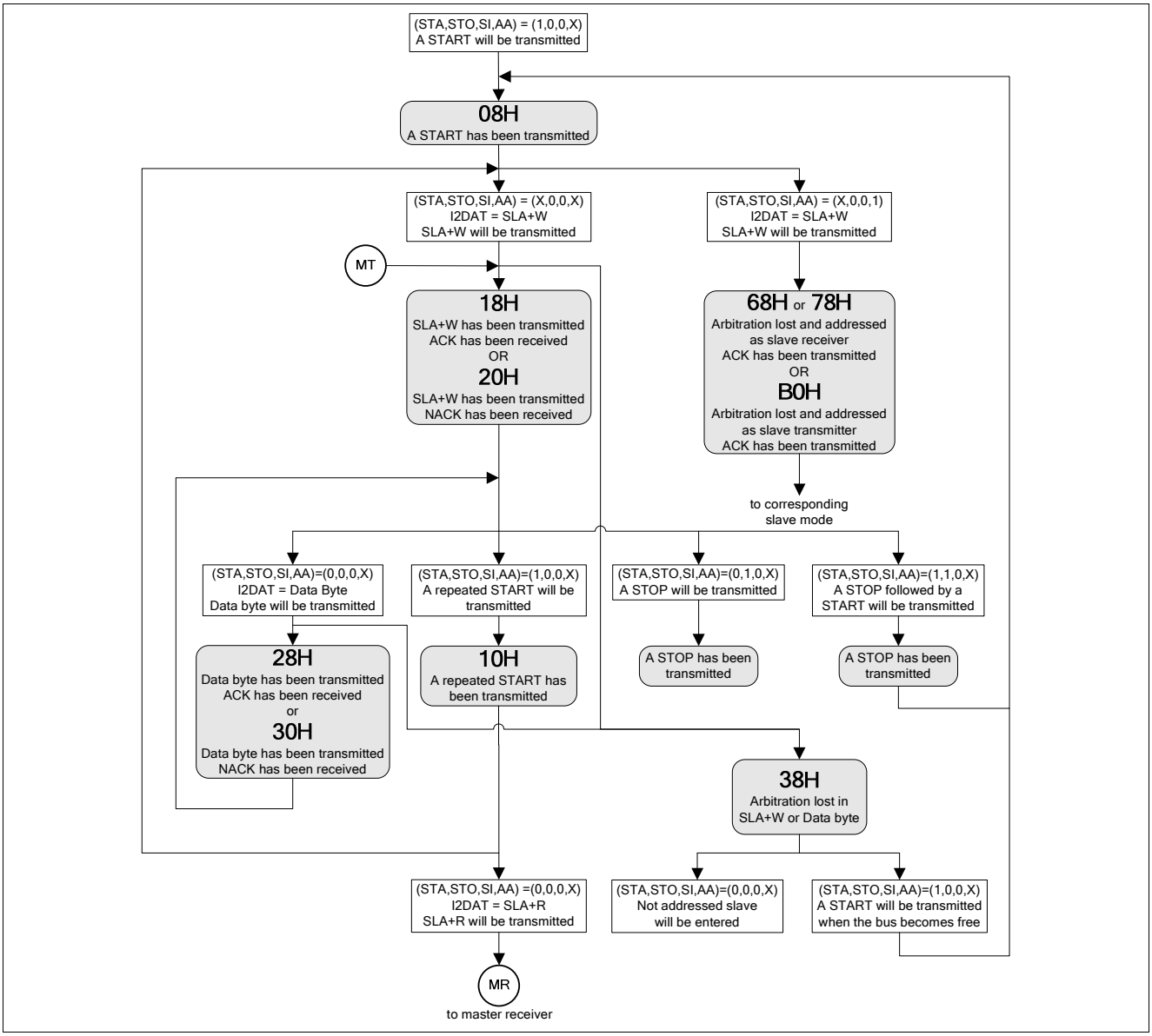


Figure 16–7. Flow and Status of Master Transmitter Mode

16.4.2 Master Receiver Mode

In the master receiver mode, several bytes of data are received from a slave transmitter. The transaction is initialized just as the master transmitter mode. Following the START condition, I2DAT should be loaded with the target slave address and the data direction bit “read” (SLA+R). After the SLA+R byte is transmitted and an acknowledge bit has been returned, the SI flag is set again and I2STAT is read as 40H. SI flag then should be cleared to receive data from the slave transmitter. If AA flag (I2CON.3) is set, the master receiver will acknowledge the slave transmitter. If AA is cleared, the master receiver will not acknowledge the slave and release the slave transmitter as a not addressed slave. After that, the master can generate a STOP condition or a repeated START condition to terminate the transmission or initial another one.

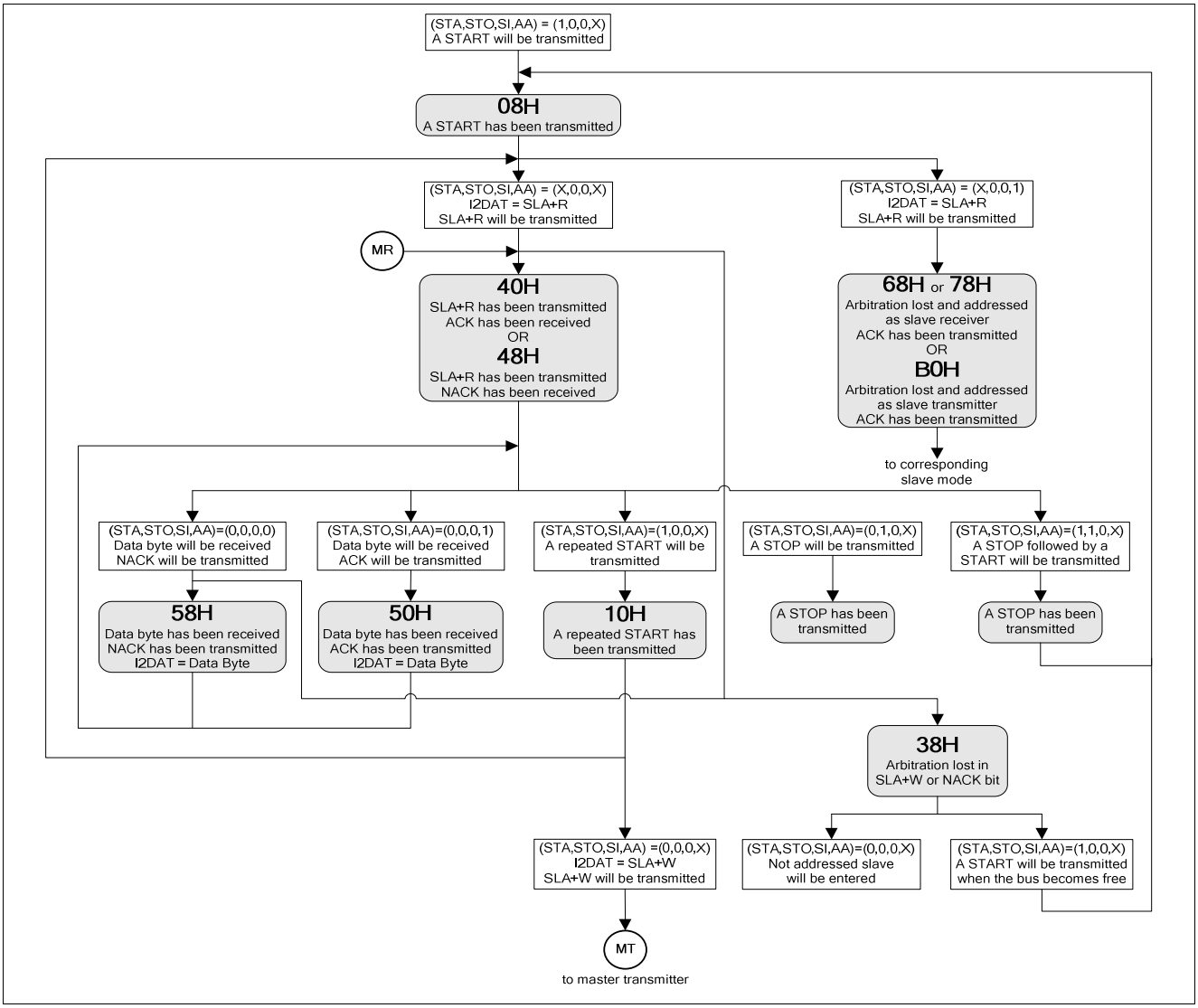


Figure 16–8. Flow and Status of Master Receiver Mode

16.4.3 Slave Receiver Mode

In the slave receiver mode, several bytes of data are received form a master transmitter. Before a transmission is commenced, I2ADDR must be loaded with the address to which the device will respond when addressed by a master. I2CLK does not affect in slave mode. The AA bit must be set to enable acknowledging its own slave address or General Call. After the initialization above, the I^2C wait until it is addressed by its own address with the data direction bit “write” (SLA+W) or by General Call addressing. The slave receiver mode may also be entered if arbitration is lost.

After the slave is addressed by SLA+W, it should clear its SI flag to receive the data from the master transmitter. If the AA bit is 0 during a transaction, the slave will return a non-acknowledge after the next received data byte. The slave will also become not addressed and isolate with the master. It cannot receive any byte of data with I2DAT remaining the previous byte of data which is just received.

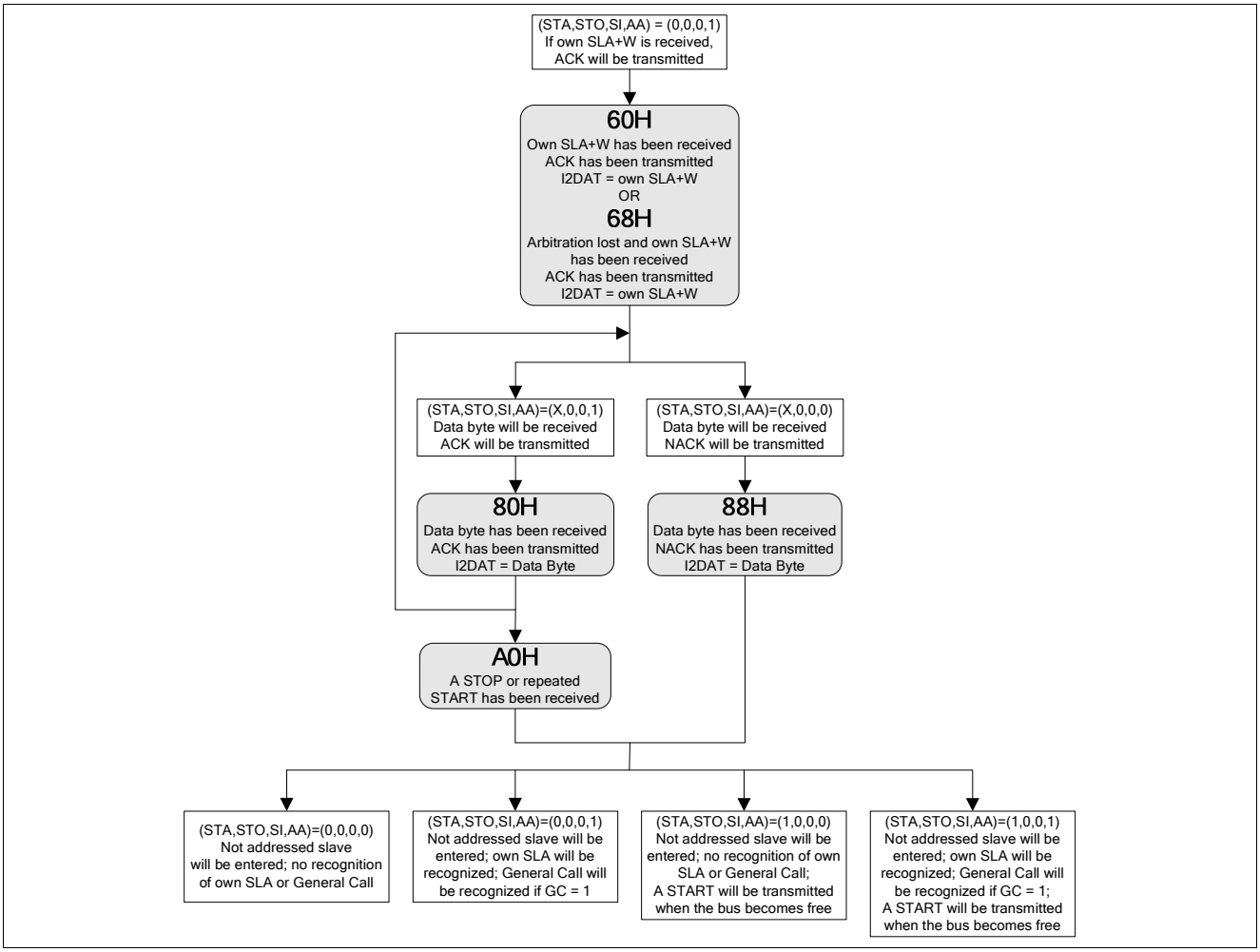


Figure 16–9. Flow and Status of Slave Receiver Mode

16.4.4 Slave Transmitter Mode

In the slave transmitter mode, several bytes of data are transmitted to a master receiver. After I2ADDR and I2CON values are given, the I^2C wait until it is addressed by its own address with the data direction bit “read” (SLA+R). The slave transmitter mode may also be entered if arbitration is lost.

After the slave is addressed by SLA+W, it should clear its SI flag to transmit the data to the master transmitter. Normally the master receiver will return an acknowledge after every byte of data is transmitted by the slave. If the acknowledge is not received, it will transmit all “1” data if it continues the transaction. It becomes a not addressed slave. If the AA flag is cleared during a transaction, the slave transmit the last byte of data. The next transmitting data will be all “1” and the slave becomes not addressed.

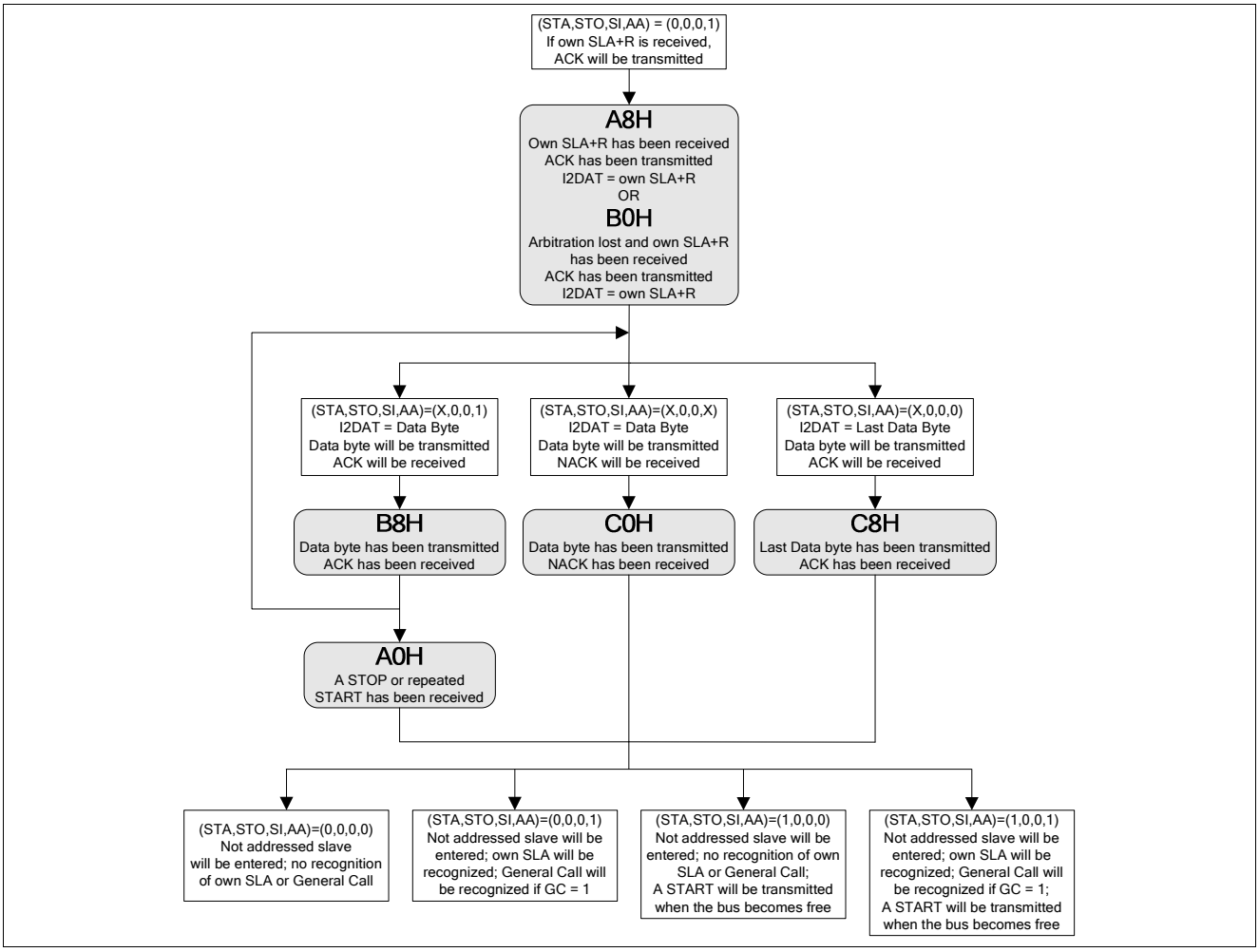


Figure 16–10. Flow and Status of Slave Transmitter Mode

16.4.5 General Call

The General Call is a special condition of slave receiver mode by sending all “0” data in slave address with data direction bit. The slave addressed by a General Call has different status codes in I2STAT with normal slave receiver mode. The General Call may also be produced if arbitration is lost.

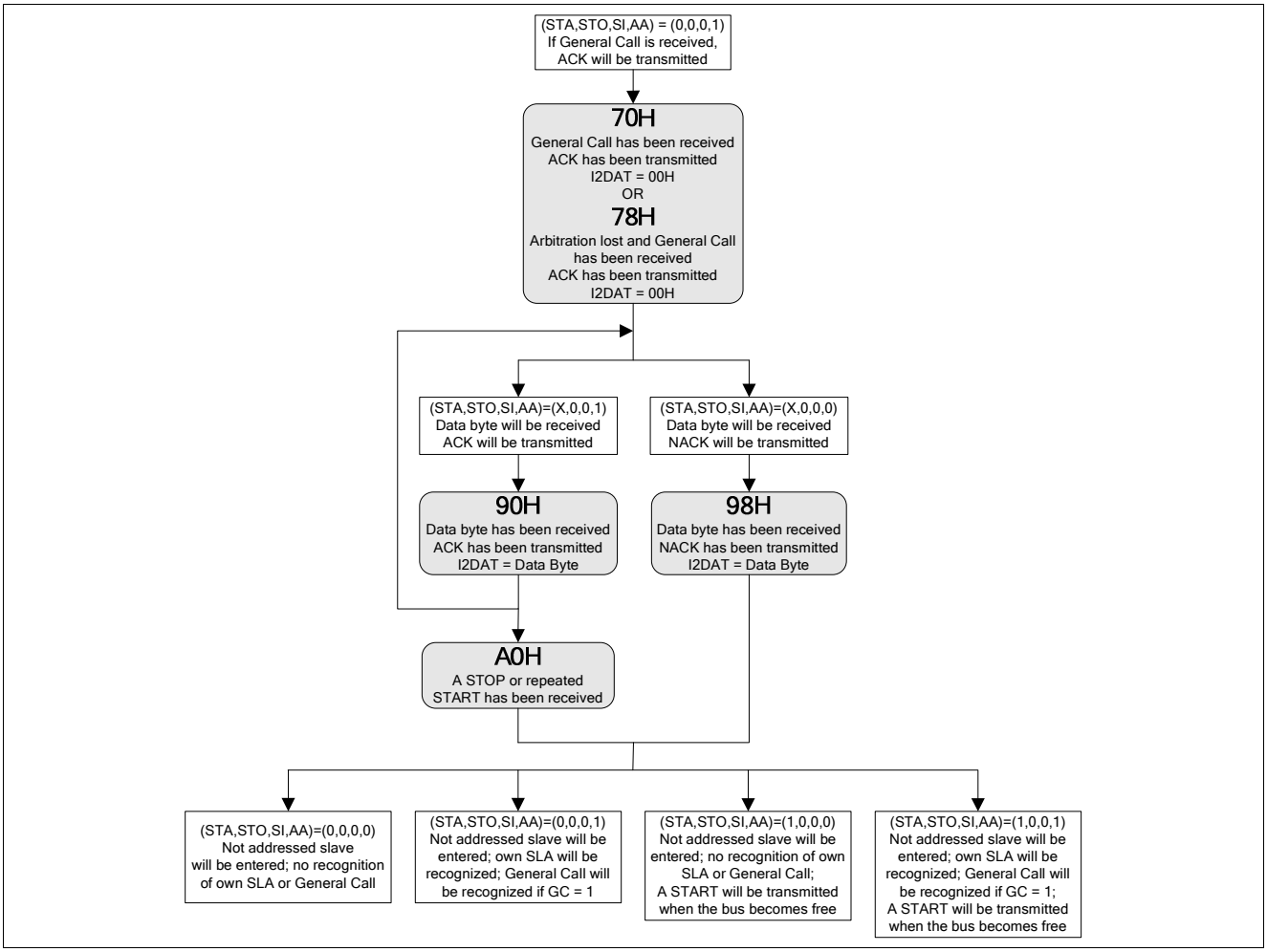


Figure 16–11. Flow and Status of General Call Mode

16.4.6 Miscellaneous States

There are two I2STAT status codes that do not correspond to the 24 defined states, which are mentioned in previous sections. These are F8H and 00H states.

The first status code F8H indicates that no relevant information is available during each transaction. Meanwhile, the SI flag is 0 and no I^2C interrupt is required.

The other status code 00H means a bus error has occurred during a transaction. A bus error is caused by a START or STOP condition appearing temporarily at an illegal position such as the second through eighth bits in an address byte or a data byte including the acknowledge bit. When a bus error occurs, the SI flag is set immediately. When a bus error is detected on the I^2C bus, the operating device immediately switches to the not addressed salve mode, release SDA and SCL lines, sets the SI flag, and loads I2STAT 00H. To recover from a bus error, the STO bit must be set as a logic 1 and SI must be cleared. After that, STO is cleared by hardware and release the I^2C bus without issuing a real STOP condition waveform.

There is a special case if a START or a repeated START condition is not successfully generated for I^2C bus is obstructed by a low level on SDA line e.g. a slave device out of bit synchronization, the problem can be solved by transmitting additional clock pulses on the SCL line. The I^2C hardware transmits additional clock pulses when the STA bit is set, but no START condition can be generated because the SDA line is pulled low. When the SDA line is eventually released, a normal START condition is transmitted, state 08H is entered, and the serial transaction continues. If a repeated START condition is transmitted while SDA is obstructed low, the I^2C hardware also performs the same action as above. In this case, state 08H is entered instead of 10H after a successful START condition is transmitted. Note that the software is not involved in solving these bus problems.

16.5 Typical Structure of I^2C Interrupt Service Routine

The following software example in C language for KEIL C51 compiler shows the typical structure of the I^2C interrupt service routine including the 26 state service routines and may be used as a base for user applications. Users can follow or modify it for their own application. If one or more of the five modes are not used, the associated state service routines may be removed, but care should be taken that a deleted routine can never be invoked.

```
void I2C_ISR (void) interrupt 6
{
      switch (I2STAT)
      {
            //============================================
            //Bus Error, always put in ISR for noise handling
            //============================================
            case 0x00:                        /*00H, bus error occurs*/
                  STO = 1;                    //recover from bus error
                  break;
```

```
//===========
//Master Mode
//===========
case 0x08:                          /*08H, a START transmitted*/
      STA = 0;                      //STA bit should be cleared by software
      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*/
      DATA_RECEIVED_LAST1 = I2DAT;
      STO = 1;
      AA = 1;
      break;
```

```
//====================================
//Slave Receiver and General Call Mode
//====================================
case 0x60:                          /*60H, own SLA+W received, ACK returned*/
      AA = 1;
      break;
case 0x68:                          /*68H, arbitration lost in SLA+W/R
                                      own SLA+W received, ACK returned */
      AA = 0;                       //not ACK next received DATA after
                                    //arbitration lost
      STA = 1;                      //retry to transmit START if bus free
      break;
case 0x70:                          /*70H, General Call received, ACK returned */
      AA = 1;
      break;
case 0x78:                          /*78H, arbitration lost in SLA+W/R
                                      General Call received, ACK returned*/
      AA = 0;
      STA = 1;
      break;
case 0x80:                          /*80H, previous own SLA+W, DATA received,
                                      ACK returned*/
      DATA_RECEIVED2 = I2DAT;
      if (To_RX_Last_Data2)
            AA = 0;
      else
            AA = 1;
      break;
case 0x88:                          /*88H, previous own SLA+W, DATA received,
                                      NACK returned, not addressed SLAVE mode
                                      entered*/
      DATA_RECEIVED_LAST2 = I2DAT;
      AA = 1;                       //wait for ACK next Master addressing
      break;
case 0x90:                          /*90H, previous General Call, DATA received,
                                      ACK returned*/
      DATA_RECEIVED3 = I2DAT;
      if (To_RX_Last_Data3)
            AA = 0;
      else
            AA = 1;
      break;
case 0x98:                          /*98H, previous General Call, DATA received,
                                      NACK returned, not addressed SLAVE mode
                                      entered*/
      DATA_RECEIVED_LAST3 = I2DAT;
      AA = 1;
      break;
//==========
//Slave Mode
//==========
case 0xA0:                          /*A0H, STOP or repeated START received while
                                      still addressed SLAVE mode*/
      AA = 1;
      break;
```

```
            //======================
            //Slave Transmitter Mode
            //======================
            case 0xA8:                          /*A8H, own SLA+R received, ACK returned*/
                  I2DAT = NEXT_SEND_DATA3;
                  AA = 1;                       //when AA is "1", not last data to be
                                                //transmitted
                  break;
            case 0xB0:                          /*B0H, arbitration lost in SLA+W/R
                                                  own SLA+R received, ACK returned */
                  I2DAT = DUMMY_DATA;
                  AA = 0;                       //when AA is "0", last data to be
                                                //transmitted
                  STA = 1;                      //retry to transmit START if bus free
                  break;
            case 0xB8:                          /*B8H, previous own SLA+R, DATA transmitted,
                                                  ACK received*/
                  I2DAT = NEXT_SEND_DATA4;
                  if (To_TX_Last_Data)//if last DATA will be transmitted
                        AA = 0;
                  else
                        AA = 1;
                  break;
            case 0xC0:                          /*C0H, previous own SLA+R, DATA transmitted,
                                                  NACK received, not addressed SLAVE mode
                                                  entered*/
                  AA = 1;
                  break;
            case 0xC8:                          /*C8H, previous own SLA+R, last DATA trans-
                                                  mitted, ACK received, not addressed SLAVE
                                                  mode entered*/
                  AA = 1;
                  break;
      }//end of switch (I2STAT)

      SI = 0;                                   //SI should be the last step of I2C ISR
      while(STO);                               //wait for STOP transmitted or bus error
                                                //free, STO is cleared by hardware
}//end of I2C_ISR
```

16.6 I²C Time-out

There is a 14-bit time-out counter which can be used to deal with the I2C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows. Meanwhile TIF will be set by hardware and requests I²C interrupt. When time-out counter is enabled, setting flag SI to high will reset counter and restart counting up after SI is cleared. If the I²C bus hangs up, it causes the SI flag not set for a period. The 14-bit time-out counter will overflow and require the interrupt service.

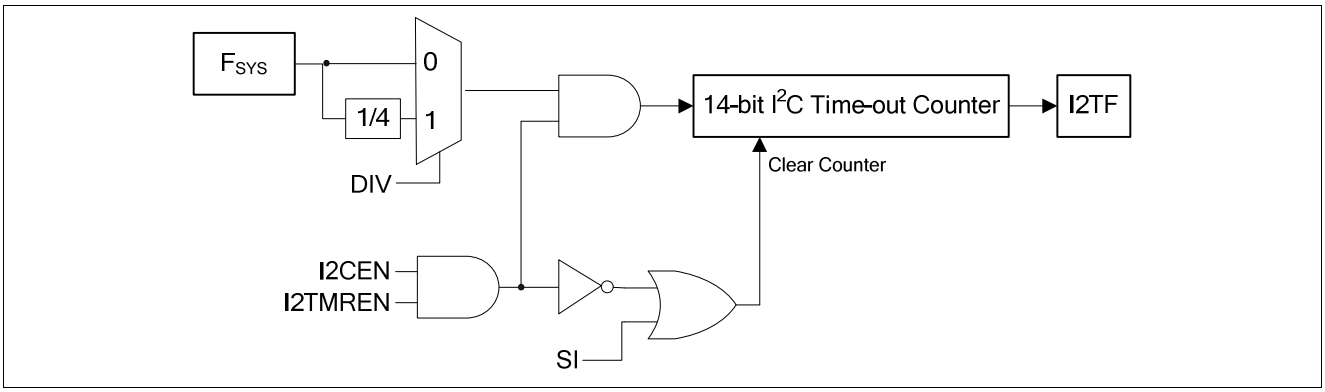


Figure 16–12. I²C Time-out Count

I2TOC – I²C Time-out Counter

7	6	5	4	3	2	1	0
-	-	-	-	-	I2TOCEN	DIV	I2TOF
-	-	-	-	-	r/w	r/w	r/w

Address: BFH reset value: 0000 0000B

Bit	Name	Description
7:3	-	**Reserved.**
2	I2TOCEN	**I²C time-out counter enable.** 0 = The I²C time-out counter is disabled. 1 = The I²C time-out counter is enabled.
1	DIV	**I²C time-out counter clock divider.** 0 = The divider of I²C time-out counter is 1/1 of F_{SYS}. 1 = The divider of I²C time-out counter is 1/4 of F_{SYS}.
0	I2TOF	**I²C time-out counter overflow flag.** I2TOF flag is set by hardware if 14-bit I²C time-out counter overflows. I2TOF flag is cleared by software.

16.7 I²C Interrupts

There are two I²C flags, SI and I2TOF. Both of them can generate an I²C event interrupt requests. If I²C interrupt mask is enabled via setting EI2C (EIE.0) and EA is 1, CPU will executes the I²C interrupt service routine once any of these two flags is set. The user needs to check flags to determine what event caused the interrupt. Both of I²C flags are cleared by software.

17 Pulse Width Modulated (PWM)

17.1 Features

PWM (Pulse Width Modulation) signal is a useful control solution in wide application field. It can used on motor driving, fan control, backlight brightness tuning, LED light dimming, or simulating as a simple digital to analog converter output through a low pass filter circuit. **N79E845** series provide four channels, maximum 10-bit PWM output.

17.2 Function Description

The **N79E845** series contain four Pulse Width Modulated (PWM) channels which generate pulses of programmable length and interval. The output for PWM0 is on P0.1, PWM1 on P1.6, PWM2 on P1.7 and PWM3 on P0.0. After chip reset the internal output of the each PWM channel is a “1”. In this case before the pin will reflect the state of the internal PWM output a “1” must be written to each port bit that serves as a PWM output. A block diagram is shown as below Figure 17-1. The interval between successive outputs is controlled by a 10–bit down counter which uses configurable internal clock prescaler as its input. The PWM counter clock has the frequency as the clock source $F_{PWM} = F_{SYS}$/Prescaler. When the counter reaches underflow it is reloaded with a user selectable value. This mechanism allows the user to set the PWM frequency at any integer sub–multiple of the microcontroller clock frequency. The repetition frequency of the PWM is given by:

PWM frequency = $\frac{F_{SYS}}{1+PWMP}$, PWM active level duty = $\frac{\text{PWMn}}{1+\text{PWMP}}$.

where PWMP is contained in PWMPH and PWMPL as described in the following.

PWMPL – PWM Counter Low Bits Register

7	6	5	4	3	2	1	0
PWMP.7	PWMP.6	PWMP.5	PWMP.4	PWMP.3	PWMP.2	PWMP.1	PWMP.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: D9H reset value: 0000 0000B

Bit	Name	Description
7:0	PWMPL	The PWM Counter Bits Register bit[7:0].

PWMPH – PWM Counter High Bits Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	PWMP.9	PWMP.8

-	-	-	-	-	-	r/w	r/w

Address: D1H reset value: 0000 0000B

Bit	Name	Description
7:2	-	Reserved.
1:0	PWMPH	The PWM Counter Bits Register bit[9:8].

The user should follow the initialization steps below to start generating the PWM signal output. In the first step by setting CLRPWM (PWMCON0.4), it ensures the 10-bit down counter a determined value. After setting all period and duty registers, PWMRUN (PWMCON0.7) can be set as a logic 1 to trigger the 10-bit down counter running. In the beginning the PWM output remains high until the counter value is less than the value in duty control registers of PWMnH and PWMnL. At this point the PWM output goes low until the next underflow. When the 10-bit down counter underflows, PWMP buffer register will be reloaded in 10-bit down counter. It continues PWM signal output by repeating this routine.

The hardware for all period and duty control registers is double buffered designed. Therefore the PWMP and PWMn registers can be written to at any time, but the period and duty cycle of PWM will not updated immediately until the LOAD (PWMCON0.6) is set and previous period is complete. This allows updating the PWM period and duty glitchless operation.

PWM0L – PWM 0 Low Register

7	6	5	4	3	2	1	0
PWM0.7	PWM0.6	PWM0.5	PWM0.4	PWM0.3	PWM0.2	PWM0.1	PWM0.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: DAH reset value: 0000 0000B

Bit	Name	Description
7:0	PWM0L	The PWM 0 Low Bits Register bit[7:0].

PWM0H – PWM 0 High Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	PWM0.9	PWM0.8
-	-	-	-	-	-	r/w	r/w

Address: D2H reset value: 0000 0000B

Bit	Name	Description
7:2	-	Reserved.
1:0	PWM0H	The PWM 0 High Bits Register bit[9:8].

PWM1L – PWM 1 Low Register

7	6	5	4	3	2	1	0
PWM1.7	PWM1.6	PWM1.5	PWM1.4	PWM1.3	PWM1.2	PWM1.1	PWM1.0

r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: DBH reset value: 0000 0000B

Bit	Name	Description
7:0	PWM1L	The PWM 0 Low Bits Register bit[7:0].

PWM1H – PWM 1 High Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	PWM1.9	PWM1.8
-	-	-	-	-	-	r/w	r/w

Address: D3H reset value: 0000 0000B

Bit	Name	Description
7:2	-	Reserved.
1:0	PWM1H	The PWM 1 High Bits Register bit[9:8].

PWM2L – PWM 2 Low Register

7	6	5	4	3	2	1	0
PWM2.7	PWM2.6	PWM2.5	PWM2.4	PWM2.3	PWM2.2	PWM2.1	PWM2.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: DDH reset value: 0000 0000B

Bit	Name	Description
7:0	PWM2L	The PWM 2 Low Bits Register bit[7:0].

PWM2H – PWM 2 High Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	PWM2.9	PWM2.8
-	-	-	-	-	-	r/w	r/w

Address: D5H reset value: 0000 0000B

Bit	Name	Description
7:2	-	Reserved.
1:0	PWM2H	The PWM 2 High Bits Register bit[9:8].

PWM3L – PWM 3 Low Register

7	6	5	4	3	2	1	0
PWM3.7	PWM3.6	PWM3.5	PWM3.4	PWM3.3	PWM3.2	PWM3.1	PWM3.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: DEH reset value: 0000 0000B

Bit	Name	Description
7:0	PWM3L	The PWM 0 Low Bits Register bit[7:0].

PWM3H – PWM 3 High Register

7	6	5	4	3	2	1	0
-	-	-	-	-	-	PWM3.9	PWM3.8
-	-	-	-	-	-	r/w	r/w

Address: D6H reset value: 0000 0000B

Bit	Name	Description
7:2	-	Reserved.
1:0	PWM3H	The PWM 3 High Bits Register bit[9:8].

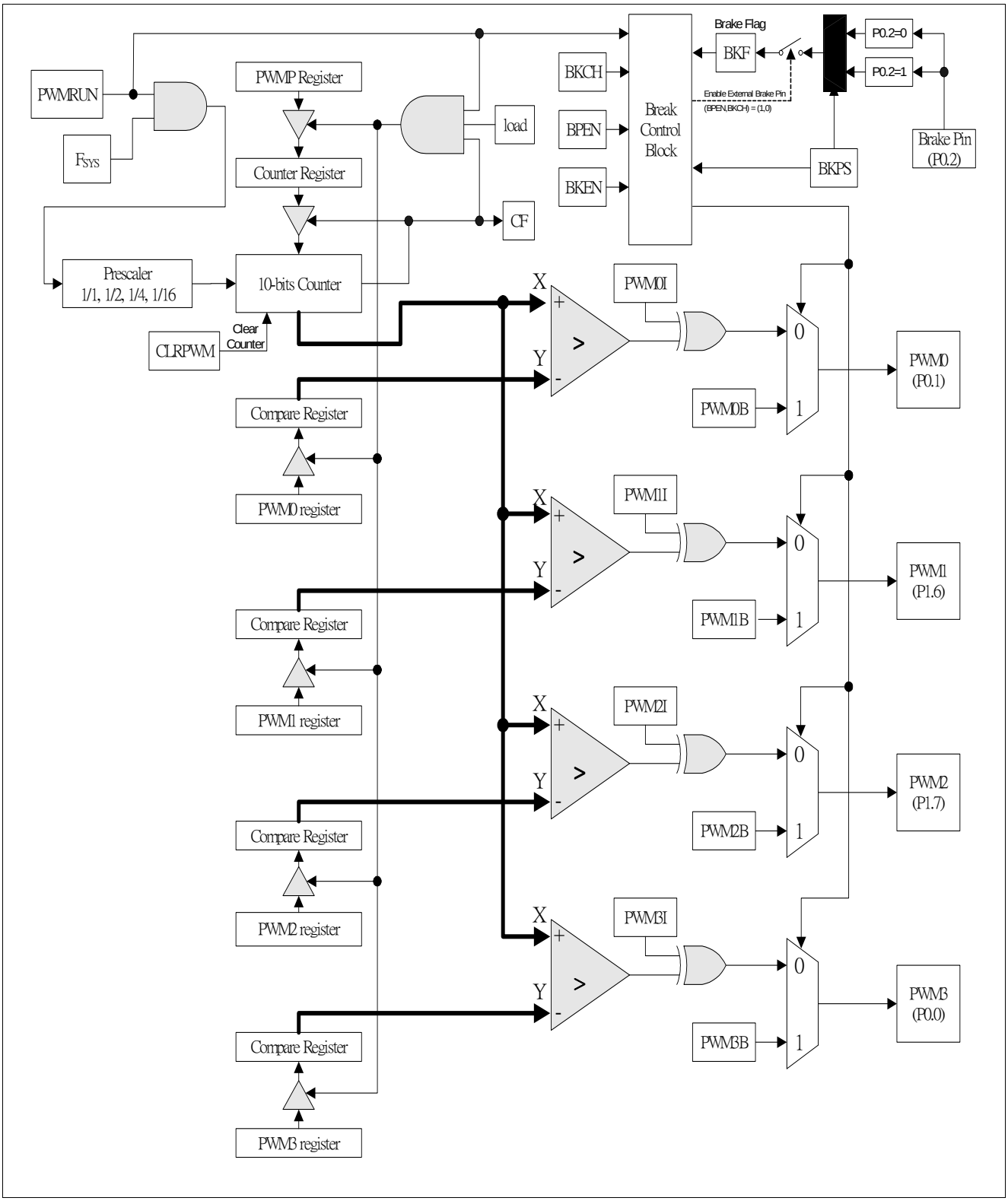


Figure 17-1 PWM Block Diagram

A compare value greater than the counter reloaded value is in the PWM output being permanently low. In addition there are two special cases. A compare value of all zeroes, 000H, causes the output to remain permanently high. A compare value of all ones, 3FFH, results in the PWM output remaining permanently low. Again the compare value is loaded into a Compare register. The transfer from this holding register to the actual Compare register is under program control. The register assignments are shown below where the number immediately following "PWMn" identifies the PWM output. Therefore, the PWM0 controls the width of PWM0, PWM1 the width of PWM1 etc.

The overall functioning of the PWM module is controlled by the contents of the PWMCON0 register. The operation of most of the control bits is straightforward. For example, there is an invert bit for each output which causes results in the output to have the opposite value compared to its non-inverted output. The transfer of the data from the Counter and Compare registers to the control registers is controlled by the PWMCON0.6 (load) while PWMCON0.7 (PWMRUN) allows the PWM to be either in the run or idle state. The user can monitor when underflow causes the transfer to occur by monitoring the Transfer bit PWCON1.6 (load) or PWMCON0.5 (CF flag). When the transfer takes place the PWM logic automatically resets those bits by next clock cycle.

A loading of new period and duty by setting LOAD should be ensured complete by monitoring it and waiting for a hardware automatic clearing LOAD bit. Any updating of PWM control registers during LOAD bit as logic 1 will cause unpredictable output.

PWMCON0 – PWM Control Register 0

7	6	5	4	3	2	1	0
PWMRUN	Load	CF	CLRPWM	PWM3I	PWM2I	PWM1I	PWM0I
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: DCH reset value: 0000 0000B

Bit	Name	Description
7	**PWMRUN**	0: The PWM is not running. 1: The PWM counter is running.
6	**Load**	0: The registers value of PWMP and Comparators are never loaded to counter and Comparator registers. 1: The PWMP register will be load value to counter register after counter underflow, and hardware will clear by next clock cycle.
5	**CF**	10-bit counter overflow flag: 0: The 10-bit counter down count is not underflow. 1: The 10-bit counter down count is underflow.
4	**CLRPWM**	1: Clear 10-bit PWM counter to 000H.

Bit	Name	Description
3	**PWM3I**	0: PWM3 output is non-inverted. 1: PWM3 output is inverted.
2	**PWM2I**	0: PWM2 output is non-inverted. 1: PWM2 output is inverted.
1	**PWM1I**	0: PWM1 output is non-inverted. 1: PWM1 output is inverted.
0	**PWM0I**	0: PWM0 output is non-inverted. 1: PWM0 output is inverted.

The fact that the transfer from the Counter and PWMn register to the working registers(10-bit Counter and Compare register) only occurs when there is an underflow in the counter results in the need for the user's program to observe the following precautions. If PWMCON0 is written with Load set without Run being enabled the transfer will never take place. Thus if a subsequent write sets Run without Load the compare and counter values will not be those expected. If Load and Run are set, and prior to underflow there is a subsequent load of PWMCON0 which sets Run but not Load, the load will never take place. Again the compare and counter values that existed prior to the update attempt will be used.

As outlined above the Load bit can be polled to determine when the load occurs. Unless there is a compelling reason to do otherwise, it is recommended that both PWMRUN (PWMCON0.7), and Load (PWMCON0.6) be set when PWMCON0 is written.

When the PWMRUN bit, PWMCON0.7 is cleared the PWM outputs take on the state they had just prior to the bit being cleared. In general this state is not known. In order to place the outputs in a known state when PWMRUN is cleared the Compare registers can be written to either the "always 1" or "always 0" so the output will have the output desired when the counter is halted. After this PWMCON0 should be written with the Load and Run bits are enabled. After this is done PWMCON0 to is polled to find that the Load or CF flag has taken place. Once the load has occurred the Run bit in PWMCON0 can be cleared. The outputs will retain the state they had just prior to the Run being cleared. If the Brake pin (see discussion below in section concerning the operation of PWMCON1) is not used to control the brake function, the "Brake when not running" function can be used to cause the outputs to have a given state when the PWM is halted. This approach should be used only in time critical situations when there is not sufficient time to use the approach outlined above since going from the Brake state to run without causing an undefined state on the outputs is not straightforward. A discussion on this topic is included in the section on PWMCON1.

PWMCON1 – PWM Control Register 1

7	6	5	4	3	2	1	0
BKCH	BKPS	BPEN	BKEN	PWM3B	PWM2B	PWM1B	PWM0B

r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: DFH reset value: 0000 0000B

Bit	Name	Description
7	BKCH	See the below table, when BKEN is set.
6	BKPS	0: Brake is asserted if P0.2 is low. 1: Brake is asserted if P0.2 is high
5	BPEN	See the below table, when BKEN is set.
4	BKEN	0: The Brake is never asserted. 1: The Brake is enabled, and see the below table.
3	PWM3B	0: The PWM3 output is low, when Brake is asserted. 1: The PWM3 output is high, when Brake is asserted.
2	PWM2B	0: The PWM2 output is low, when Brake is asserted. 1: The PWM2 output is high, when Brake is asserted.
1	PWM1B	0: The PWM1 output is low, when Brake is asserted. 1: The PWM1 output is high, when Brake is asserted.
0	PWM0B	0: The PWM0 output is low, when Brake is asserted. 1: The PWM0 output is high, when Brake is asserted.

Brake Condition table

BPEN	BKCH	BREAK CONDITION
0	0	Brake on, (software brake and keeping brake)
0	1	On, when PWM is not running (PWMRUN=0), the PWM output condition is follow PWMNB setting. Off, when PWM is running (PWMRUN=1).
1	0	Brake on, when break pin asserted, no PWM output, the bit of PWMRUN will be cleared and BKF flag will be set.
1	1	No any active.

PWMCON2 – PWM Control Register 2

7	6	5	4	3	2	1	0
-	-	-	-	FP1	FP0	-	BKF
-	-	-	-	r/w	r/w	-	r/w

Address: D7H reset value: 0000 0000B

Bit	Name	Description
7:4	-	**Reserved.**

Bit	Name	Description
3:2	FP[1:0]	Select PWM frequency prescaler select bits. The clock source of prescaler, 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	**The external brake pin flag.** 0: The PWM is not brake. 1: The PWM is brake by external brake pin. It will be cleared by software.

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 in order to go from brake being asserted to having the PWM run without going through an indeterminate state care must 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. The details for PWMCON1 are shown in the following table.

18 Timed Access Protection(TA)

N79E845 series have 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. In order to prevent this risk, the **N79E845** series have 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: 0000 0000B

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 must first write AAH to the TA and immediately followed by a write of 55H to TA. After these 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 must be repeated to access the other protected bits. Not that the TA protected SFRs are required timed access for writing. But 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 must occur within 3 machine-cycles of each other. Interrupts should be disabled during this procedure to avoid delay between these two writes. If there is no interrupt enabled, the `CLR EA` and

SETB EA instructions can be left out. Once the timed access window closes, the procedure must be repeated to access the other protected bits.

Examples of timed assessing are shown to illustrate correct or incorrect writing processes.

Example 1,

```
      (CLR   EA)                ;if any interrupt is enabled, disable temporarily
      MOV    TA,#0AAH           ;2 machine-cycles.
      MOV    TA,#55H            ;2 machine-cycles.
      ORL    CHPCON,#data       ;2 machine-cycles.
     (SETB   EA)                ;resume interrupts enabled
```

Example 2,

```
      (CLR   EA)                ;if any interrupt is enabled, disable temporarily
      MOV    TA,#0AAH           ;2 machine-cycles.
      MOV    TA,#55H            ;2 machine-cycles.
      NOP                       ;1 machine-cycle.
      NOP                       ;1 machine-cycle.
      ANL    ISPTRG,#data       ;2 machine-cycles.
      (SETB  EA)                ;resume interrupts enabled
```

Example 3,

```
      (CLR   EA)                ;if any interrupt is enabled, disable temporarily
      MOV    TA,#0AAH           ;2 machine-cycles.
      NOP                       ;1 machine-cycle.
      MOV    TA,#55H            ;2 machine-cycles.
      MOV    WDCON0,#data1      ;2 machine-cycles.
      ORL    PMCR,#data2        ;2 machine-cycles.
      (SETB  EA)                ;resume interrupts enabled
```

Example 4,

```
      (CLR   EA)                ;if any interrupt is enabled, disable temporarily
      MOV    TA,#0AAH           ;2 machine-cycles.
      NOP                       ;1 machine-cycle.
      NOP                       ;1 machine-cycle.
      MOV    TA,#55H            ;2 machine-cycles.
      ANL    WDCON0,#data       ;2 machine-cycles.
      (SETB  EA)                ;resume interrupts enabled
```

In the first examples, the writing to the protected bits is done before the three-machine-cycle window closes. In example 2, however, the writing to ISPTRG does not complete during the window opening, there will be no change of the value of ISPTRG. In example 3, the WDCON0 is successful written but the PMCR access is out of the three-machine-cycle window. Therefore PMCR value will not change either. In Example 4, the second write 55H to TA completes after three machine-cycles of the first write TA of AAH, therefore the timed access window in not opened at all, and the write to the protected bit fails.

In **N79E845** series, the TA protected SFRs includes PMCR, CHPCON (9FH), ISPTRG (A4H), SHBDA (9CH), WDCON0 (D8H), and WDCON1 (ABH).

19 Interrupt System (Interrupt)

The **N79E845** series have four priority level interrupts structure with 14 interrupt sources. Each of the interrupt sources has an individual priority bit, flag, interrupt vector and enable bit. In addition, the interrupts can be globally enabled or disabled.

19.1 Interrupt Sources

The External Interrupts $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ can be either edge triggered or level triggered, depending on bits IT0 and IT1. The bits IE0 and IE1 in the TCON register are the flags which are checked to generate the interrupt. In the edge triggered mode, the INTx inputs are sampled in every machine-cycle. If the sample is high in one cycle and low in the next, then a high to low transition is detected and the interrupts request flag IEx in TCON is set. The flag bit requests the interrupt. Since the external interrupts are sampled every machine-cycle, they have to be held high or low for at least one complete machine-cycle. The IEx flag is automatically cleared when the service routine is called. If the level triggered mode is selected, then the requesting source has to hold the pin low till the interrupt is serviced. The IEx flag will not be cleared by the hardware on entering the service routine. If the interrupt continues to be held low even after the service routine is completed, then the processor may acknowledge another interrupt request from the same source.

The Timer 0 and 1 Interrupts are generated by the TF0 and TF1 flags. These flags are set by the overflow in the Timer 0 and Timer 1. The TF0 and TF1 flags are automatically cleared by the hardware when the timer interrupt is serviced.

The Watchdog timer can be used as a system monitor or a simple timer. In either case, when the timeout count is reached, the Watchdog Timer interrupt flag WDTRF (WDCON0.3) is set. If the interrupt is enabled by the enable bit EIE.4, then an interrupt will occur.

The Serial block can generate interrupt on reception or transmission. There are two interrupt sources from the Serial block, which are obtained by the RI and TI bits in the SCON SFR. These bits are not automatically cleared by the hardware, and the user will have to clear these bits using software.

I2C will generate an interrupt due to a new SIO state present in I2STAT register, if both EA and ES bits (in IE register) are both enabled.

SPI asserts interrupt flag, SPIF, upon completion of data transfer with an external device. If SPI interrupt is enabled (ESPI at EIE.6), a serial peripheral interrupt is generated. SPIF flag is software clear, by writing a 0. MODF and SPIOVF also will generate interrupt if occur. They share the same vector address as SPIF.

The ADC can generate interrupt after finished ADC converter. There is one interrupt source, which is obtained by the ADCI bit in the ADCCON SFR. This bit is not automatically cleared by the hardware, and the user will have to clear this bit using software.

PWM brake interrupt flag BKF is generated if P0.2 (Brake pin) detects a high (BKPS=1) or low (BKPS=0) at port pin. At this moment, BKF (PWMCON2.0) is set by hardware and it must be cleared by software. PWM period interrupt flag CF is set by hardware when its' 10-bit down counter underflow and is only cleared by software. BKF is set the PWM interrupt is requested If PWM interrupt is enabled (EPWM=1).

Keyboard interrupt is generated when any of the keypad connected to P0 pins detects a low-level or edge changed at port pin. Each keypad interrupt can be individually enabled or disabled. The KBI flag (KBIF[7:0]) must be cleared by software.

POR detect can cause POF flag, BOF, to be asserted if power voltage drop below BOD voltage level. Interrupt will occur if EBO (IE.5) and global interrupt enable (EA) are set.

All the bits that generate interrupts can be set or reset by software, and thereby software initiated interrupts can be generated. Each of the individual interrupts can be enabled or disabled by setting or clearing a bit in the IE SFR. IE also has a global enable/disable bit EA, in which can be cleared to disable all the interrupts.

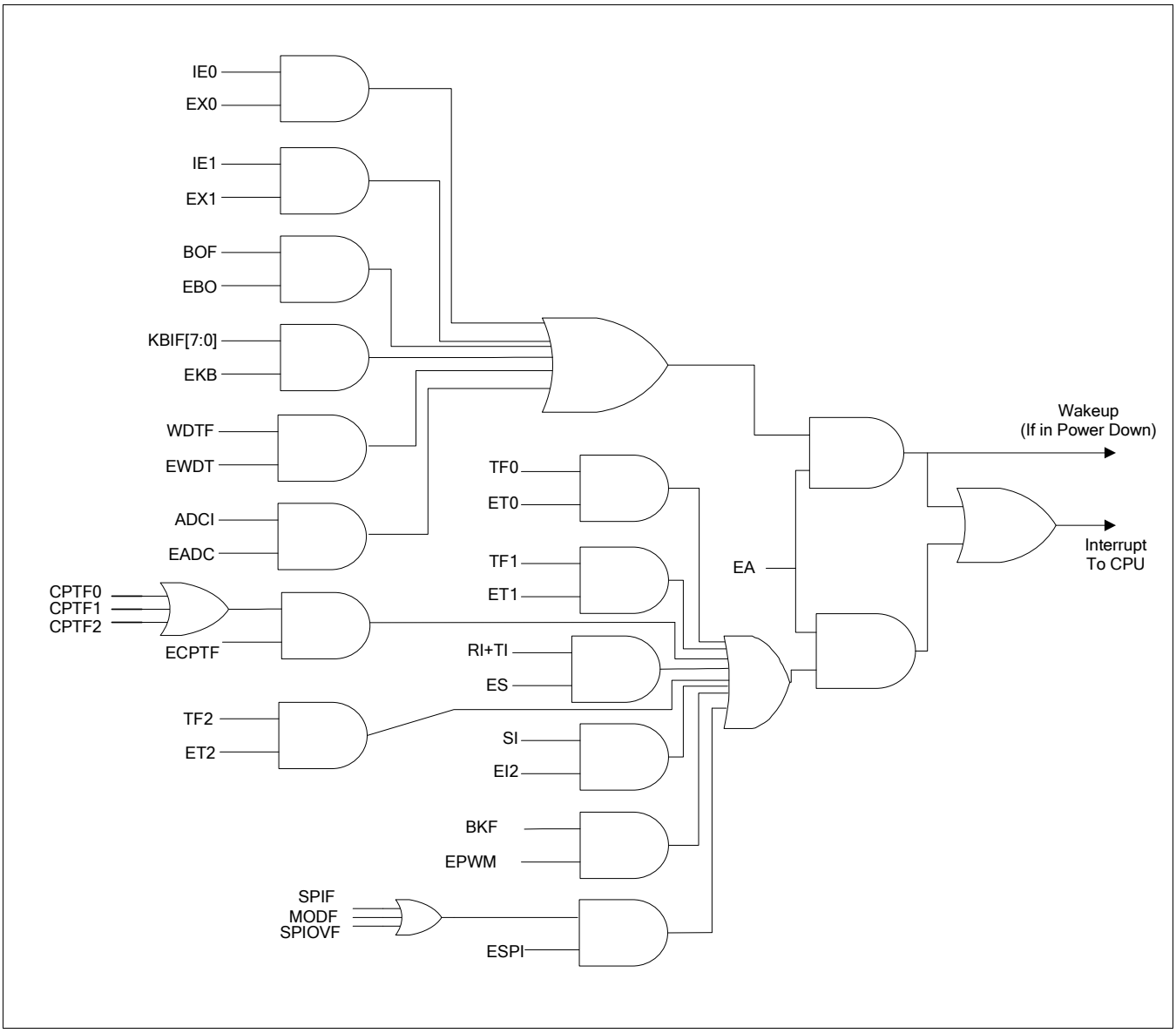


Figure 19-1 Block Diagram for Interrupt Flag

19.2 Priority Level Structure

There are four priority levels for the interrupts, highest, high, low and lowest. The interrupt sources can be individually set to either high or low levels. Naturally, a higher priority interrupt cannot be interrupted by a lower priority interrupt. However there exists a pre-defined hierarchy amongst the interrupts themselves. This hierarchy comes into play when the interrupt controller has to resolve simultaneous requests having the same priority level. This hierarchy is defined as shown in **Table 19-3**, the interrupts are numbered starting from the highest priority to the lowest.

The interrupt flags are sampled every machine-cycle. In the same machine-cycle, the sampled interrupts are polled and their priority is resolved. If certain conditions are met then the hardware will execute an internally generated LCALL instruction which will vector the process to the appropriate interrupt vector address. The conditions for generating the LCALL are

1. An interrupt of equal or higher priority is not currently being serviced.
2. The current polling cycle is the last machine-cycle of the instruction currently being executed.
3. The current instruction does not involve a write to IE, EIE, IP, IPH, EIP or IPH1 registers and is not a RETI.

If any of these conditions are not met, then the LCALL will not be generated. The polling cycle is repeated every machine-cycle, with the interrupts sampled in the same machine-cycle. If an interrupt flag is active in one cycle but not responded to, and is not active when the above conditions are met, the denied interrupt will not be serviced. This means that active interrupts are not remembered; every polling cycle is new.

The processor responds to a valid interrupt by executing an LCALL instruction to the appropriate service routine. This may or may not clear the flag which caused the interrupt. In case of Timer interrupts, the TF0 or TF1 flags are cleared by hardware whenever the processor vectors to the appropriate timer service routine. In case of external interrupt, INT0 and INT1, the flags are cleared only if they are edge triggered. In case of Serial interrupts, the flags are not cleared by hardware. In the case of Timer 2 interrupt, the flags are not cleared by hardware. The hardware LCALL behaves exactly like the software LCALL instruction. This instruction saves the Program Counter contents onto the Stack, but does not save the Program Status Word PSW. The PC is reloaded with the vector address of that interrupt which caused the LCALL. These address of vector for the different sources are as follows

Table 19-1 Vector locations for interrupt sources

Source	Vector Address	Source	Vector Address
External Interrupt 0	0003h	Timer 0 Overflow	000Bh
External Interrupt 1	0013h	Timer 1 Overflow	001Bh
Serial Port	0023h	Timer 2 Overflow/Match	002Bh
I2C Interrupt	0033h	KBI Interrupt	003Bh
BOD Interrupt	0043h	SPI Interrupt	004Bh
Watchdog Timer	0053h	ADC Interrupt	005Bh
Capture	0063h		
PWM brake Interrupt	0073h		

The vector table is not evenly spaced; this is to accommodate future expansions to the device family.

Table 19-2 Four-level interrupt priority

Priority bits		Interrupt Priority Level
IPXH	IPX	
0	0	Level 0 (Lowest priority)
0	1	Level 1
1	0	Level 2
1	1	Level 3 (highest priority)

Execution continues from the vectored address till an RETI instruction is executed. On execution of the RETI instruction the processor pops the Stack and loads the PC with the contents at the top of the stack. The user must take care that the status of the stack is restored to whatever after the hardware LCALL, if the execution is to return to the interrupted program. The processor does not notice anything if the stack contents are modified and will proceed with execution from the address put back into PC. Note that a RET instruction would perform exactly the same process as a RETI instruction, but it would not inform the Interrupt Controller that the interrupt service routine is completed, and would leave the controller still thinking that the service routine is underway.

The **N79E845** series use a four-priority level interrupt structure. This allows great flexibility in controlling the handling of the **N79E845** series many interrupt sources. The **N79E845** series support up to 14 interrupt sources.

Each interrupt source can be individually enabled or disabled by setting or clearing a bit in registers IE or EIE. The IE register also contains a global disable bit, EA, which disables all interrupts at once.

Each interrupt source can be individually programmed to one of four priority levels by setting or clearing bits in the IP, IPH, EIP, and EIPH registers. An interrupt service routine in progress can be interrupted by a higher priority interrupt, but not by another interrupt of the same or lower priority. The highest priority interrupt service cannot be interrupted by any other interrupt source. So, if two requests of different priority levels are received simultaneously, the request of higher priority level is serviced.

If requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. This is called the arbitration ranking. Note that the arbitration ranking is only used to resolve simultaneous requests of the same priority level.

As below Table summarizes the interrupt sources, flag bits, vector addresses, enable bits, priority bits, arbitration ranking, and whether each interrupt may wake up the CPU from Power Down mode.

Table 19-3 Summary of interrupt sources

Description	Interrupt Flag Bit(s)	Vector Address	Interrupt Enable Bit(s)	Flag cleared by	Interrupt Priority	Arbitration Ranking	Power Down Wakeup
External Interrupt 0	IE0	0003H	EX0 (IE0.0)	Hardware, Software	IPH.0, IP.0	1 (highest)	Yes
BOD Detect	BOF	0043H	EBO (IE.5)	Hardware	IPH.5, IP.5	2	Yes
Watchdog Timer	WDTF	0053H	EWDI (EIE.4)	Software	EIPH.4, EIP.4	3	Yes
Timer 0 Interrupt	TF0	000BH	ET0 (IE.1)	Hardware, Software	IPH.1, IP.1	4	No
I2C Interrupt	SI	0033H	EI2 (EIE.0)	Software	EIPH.0, EIP.0	5	No
ADC Converter	ADCI	005BH	EADC (IE.6)	Software	IPH.6, IP.6	6	Yes(1)
External Interrupt 1	IE1	0013H	EX1 (IE.2)	Hardware, Software	IPH.2, IP.2	7	Yes
KBI Interrupt	KBIF[7:0]	003BH	EKB (EIE.1)	Software	EIPH.1, EIP.1	8	Yes
Timer 1 Interrupt	TF1	001BH	ET1 (IE.3)	Hardware, Software	IPH.3, IP.3	9	No
Serial Port Tx and Rx	TI & RI	0023H	ES (IE.4)	Software	IPH.4, IP.4	10	No
PWM Interrupt	BKF	0073H	EPWM (EIE.5)	Software	EIPH.5, EIP.5	11	No
SPI	SPIF + MODF + SPIOVF	004BH	ESPI (EIE.6)	Software	EIPH.6, EIP.6	12	No
Timer 2 Overflow/Match	TF2	002Bh	ET2 (EIE.7)	Software	EIPH.7, EIP.7	13	No
Capture	CAPF0-2	0063H	ECPTF (EIE.2)	Software	IPH.7, IP.7	14 (lowest)	No

Note: 1. The ADC Converter can wake up “Power Down Mode” when its clock source is from internal RC.

19.3 Interrupt Response Time

The response time for each interrupt source depends on several factors, such as the nature of the interrupt and the instruction underway. In the case of external interrupts $\overline{\text{INT0}}$ to RI+TI, they are sampled at C3 of every machine-cycle and then their corresponding interrupt flags IEx will be set or reset. The Timer 0 and 1 overflow flags are set at C3 of the machine-cycle in which overflow has occurred. These flag values are polled only in the next machine-cycle. If a request is active and all three conditions are met, then the hardware generated LCALL is executed. This LCALL itself takes four machine-cycles to be completed. Thus there is a minimum time of five machine-cycles between the interrupt flag being set and the interrupt service routine being executed.

A longer response time should be anticipated if any of the three conditions are not met. If a higher or equal priority is being serviced, then the interrupt latency time obviously depends on the nature of the service routine currently being executed. If the polling cycle is not the last machine-cycle of the instruction being executed, then an additional delay is introduced. The maximum response time (if no other interrupt is in service) occurs if the **N79E845** series are performing a write to IE, EIE, IP, IPH, EIP or EIPH and then executes a MUL or DIV instruction. From the time an interrupt source is activated, the longest reaction time is 12 machine-cycles. This includes 1 machine-cycle to detect the interrupt, 3 machine-cycles to complete the IE, EIE, IP, IPH, EIP or EIPH access, 5 machine-cycles to complete the MUL or DIV instruction and 4 machine-cycles to complete the hardware LCALL to the interrupt vector location.

Thus in a single-interrupt system the interrupt response time will always be more than 5 machine-cycles and not more than 12 machine-cycles. The maximum latency of 12 machine-cycle is 48 clock cycles. Note that in the standard 8051 the maximum latency is 8 machine-cycles which equals 96 machine-cycles. This is a 50% reduction in terms of clock periods.

19.4 SFR of Interrupt

The SFRs associated with these interrupts are listed below.

IE – Interrupt Enable (bit-addressable)

7	6	5	4	3	2	1	0
EA	EADC	EBOD	ES	ET1	EX1	ET0	EX0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: A8H reset value: 0000 0000B

Bit	Name	Description
7	**EA**	**Enable all interrupt.** This bit globally enables/disables all interrupts. It overrides the individual interrupt mask settings. 0 = Disable all interrupt sources. 1 = Enable each interrupt depending on its individual mask setting. Individual interrupts will occur if enabled.
6	**EADC**	**Enable ADC interrupt.**
5	**EBOD**	**Enable BOD interrupt.**
4	**ES**	**Enable serial port (UART) interrupt.** 0 = Disable all UART interrupts. 1 = Enable interrupt generated by TI (SCON.1) or RI (SCON.0).
3	**ET1**	**Enable Timer 1 interrupt.** 0 = Disable Timer 1 interrupt 1 = Enable interrupt generated by TF1 (TCON.7).
2	**EX1**	**Enable external interrupt 1.** 0 = Disable external interrupt 1. 1 = Enable interrupt generated by $\overline{INT1}$ pin (P3.3).
1	**ET0**	**Enable Timer 0 interrupt.** 0 = Disable Timer 0 interrupt 1 = Enable interrupt generated by TF0 (TCON.5).
0	**EX0**	**Enable external interrupt 0.** 0 = Disable external interrupt 0. 1 = Enable interrupt generated by $\overline{INT0}$ pin (P3.2).

EIE – Extensive Interrupt Enable

7	6	5	4	3	2	1	0
ET2	ESPI	EPWM	EWDI	-	ECPTF	EKB	EI2C

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 I2C Interrupt. 1: Enable I2C 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: To set interrupt high priority of Capture 0/1/2 as highest priority level.
6	**PADC**	1: To set interrupt priority of ADC is higher priority level.
5	**PBOD**	1: To set interrupt priority of BOD Detector is higher priority level.
4	**PS**	1: To set interrupt priority of Serial port 0 is higher priority level.

Bit	Name	Description
3	**PT1**	1: To set interrupt priority of Timer 1 is higher priority level.
2	**PX1**	1: To set interrupt priority of External interrupt 1 is higher priority level.
1	**PT0**	1: To set interrupt priority of Timer 0 is higher priority level.
0	**PX0**	1: To set interrupt priority of External interrupt 0 is higher priority level.

IPH – Interrupt High Priority Register

7	6	5	4	3	2	1	0
PCAPH	PADCH	PBODH	PSH	PT1H	PX1H	PT0H	PX0H
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: B7H reset value: 0000 0000B

Bit	Name	Description
7	**PCAPH**	1: To set interrupt high priority of Capture 0/1/2 as highest priority level.
6	**PADCH**	1: To set interrupt high priority of ADC is highest priority level.
5	**PBODH**	1: To set interrupt high priority of BOD Detector is highest priority level.
4	**PSH**	1: To set interrupt high priority of Serial port 0 is highest priority level.
3	**PT1H**	1: Ro set interrupt high priority of Timer 1 is highest priority level.
2	**PX1H**	1: To set interrupt high priority of External interrupt 1 is highest priority level.
1	**PT0H**	1: To set interrupt high priority of Timer 0 is highest priority level.
0	**PX0H**	1: To set interrupt high priority of External interrupt 0 is highest priority level.

EIP – Interrupt Priority-1 Register

7	6	5	4	3	2	1	0
PT2	PSPI	PPWM	PWDI	-	-	PKB	PI2
r/w	r/w	r/w	r/w	-	-	r/w	r/w

Address: FFH reset value: 0000 0000B

Bit	Name	Description
7	**PT2**	1: To set interrupt priority of Timer 2 is higher priority level.
6	**PSPI**	1: To set interrupt priority of SPI is higher priority level.
5	**PPWM**	1: To set interrupt priority of PWM's brake is higher priority level.
4	**PWDI**	1: To set interrupt priority of Watchdog is higher priority level.

Bit	Name	Description
3:2	-	**Reserve**
1	PKB	1: To set interrupt priority of Keypad is higher priority level.
0	PI2	1: To set interrupt priority of I2C is higher priority level.

EIPH – Interrupt High Priority-1 Register

7	6	5	4	3	2	1	0
PT2H	PSPIH	PPWMH	PWDIH	-	-	PKBH	PI2H
r/w	r/w	r/w	r/w	-	-	r/w	r/w

Address: F7H reset value: 0000 0000B

Bit	Name	Description
7	PT2H	1: To set interrupt high priority of Timer 2 is highest priority level.
6	PSPIH	1: To set interrupt high priority of SPI is highest priority level.
5	PPWMH	1: To set interrupt high priority of PWM's external brake pin is highest priority level.
4	PWDIH	1: To set interrupt high priority of Watchdog is highest priority level.
3:2	-	**Reserve**
1	PKBH	1: To set interrupt high priority of Keypad is highest priority level.
0	PI2H	1: To set interrupt high priority of I2C is highest priority level.

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
3	IE1	**External interrupt 1 edge flag.** This flag is set via hardware when an edge/level of type defined by IT1 is detected. If IT1 = 1, this bit will remain set until cleared via software or at the beginning of the External Interrupt 1 service routine. If IT1 = 0, this flag is the inverse of the $\overline{INT1}$ input signal's logic level.

Bit	Name	Description
2	**IT1**	**External interrupt 1 type select.** This bit selects whether the $\overline{INT1}$ pin will detect falling edge or low level triggered interrupts. 0 = $\overline{INT1}$ is low level triggered. 1 = $\overline{INT1}$ is falling edge triggered.
1	**IE0**	**External interrupt 0 edge flag.** This flag is set via hardware when an edge/level of type defined by IT0 is detected. If IT0 = 1, this bit will remain set until cleared via software or at the beginning of the External Interrupt 0 service routine. If IT0 = 0, this flag is the inverse of the $\overline{INT0}$ input signal's logic level.
0	**IT0**	**External interrupt 0 type select.** This bit selects whether the $\overline{INT0}$ pin will detect falling edge or low level triggered interrupts. 0 = $\overline{INT0}$ is low level triggered. 1 = $\overline{INT0}$ is falling edge triggered.

20 In System Programming (ISP)

The internal Program Memory and on-chip Data Flash support both hardware programming and in system programming (ISP). Hardware programming mode uses gang-writers to reduce programming costs and time to market while the products enter into the mass production state. However, if the product is just under development or the end product needs firmware updating in the hand of an end user, the hardware programming mode will make repeated programming difficult and inconvenient. ISP method makes it easy and possible. **N79E845** series support ISP mode allowing a device to be reprogrammed under software control. Furthermore, the capability to update the application firmware makes wide range of applications possible.

ISP is performed without removing the microcontroller from the system. The most common method to perform ISP is via UART along with the firmware in LDROM. General speaking, PC transfers the new APROM code through serial port. Then LDROM firmware receives it and re-programs into APROM through ISP commands. Nuvoton provides ISP firmware, USB ISP writer and PC application program for **N79E845** series. It makes users quite easy perform ISP through Nuvoton standard ISP tool. Please explore Nuvoton 8-bit Microcontroller website:
www.nuvoton.com/hq/enu/ProductAndSales/ProductLines/ConsumerElectronicsIC/Microcontroller/TechnicalSupportingFile.htm.

20.1 ISP Procedure

Unlike RAM's real-time operation, to update flash data often takes long time. Furthermore, it is a quite complex timing procedure to erase, program, or read flash data. Fortunately, **N79E845** series carried out the flash operation with convenient mechanism to help the user update the flash content. After ISP enabled by setting ISPEN (CHPCON.0 with TA protected), the user can easily fill the 16-bit target address in ISPAH and ISPAL, data in ISPFD and command in ISPCN. Then the ISP is ready to begin by setting a triggering bit ISPGO (ISPTRG.0). Note that ISPTRG is also TA protected. At this moment, the CPU holds the Program Counter and the built-in ISP automation takes over to control the internal charge-pump for high voltage and the detail signal timing. After ISP action completed, the Program Counter continues to run the following instructions. The ISPGO bit will be automatically cleared. The user may repeat steps above for next ISP action if necessary. Through this progress, the user can easily erase, program, and verify the embedded flash by just taking care of the pure software.

The following registers relate to ISP processing.

CHPCON – Chip Control (TA protected)

7	6	5	4	3	2	1	0
SWRST	ISPF	LDUEN	-	-	-	BS	ISPEN
w	r	r/w	-	-	-	r/w	r/w

Address: 9FH reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
6	**ISPF**	**ISP fault flag. (Read Only)** The hardware will set this bit when any of the following condition is met: 1. The accessing area is illegal, such as, (a) Erasing or programming APROM itself when APROM code runs. (b) Erasing or programming LDROM when APROM code runs but LDUEN is 0. (c) Erasing, programming, or reading CONFIG bytes when APROM code runs. (d) Erasing or programming LDROM itself when LDROM code runs. (e) Accessing oversize. 2. The ISP operating runs from internal Program Memory into external one. This bit should be cleared via software.
5	**LDUEN**	**Updating LDROM enable.** 0 = The LDROM is inhibited to be erased or programmed when APROM code runs. LDROM remains read-only. 1 = The LDROM is allowed to be fully accessed when APROM code runs.
2	**-**	**Reserved**
1	**BS**	**Boot select.** **There are different meanings of writing to or reading from this bit.** **Writing:** **It defines from which block MCU boots after all resets.** **0 = The next rebooting will be from APROM.** **1 = The next rebooting will be from LDROM.** **Reading:** **It indicates from which block MCU booted after previous reset.**

Bit	Name	Description
		0 = The previous rebooting is from APROM. **1 = The previous rebooting is from LDROM.**
0	**ISPEN**	**ISP enable.** 0 = Enable ISP function. 1 = Disable ISP function. To enable ISP function will start the internal 22.1184MHz RC oscillator for timing control. To clear ISPEN should always be the last instruction after ISP operation in order to stop internal RC for reducing power consumption.

ISPCN – ISP Control

7	6	5	4	3	2	1	0
ISPA17	ISPA16	FOEN	FCEN	FCTRL.3	FCTRL.2	FCTRL.1	FCTRL.0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Address: AFH reset value: 0000 0000B

Bit	Name	Description
7:6	**ISPA[17:16]**	**ISP control.** This byte is for ISP controlling command to decide ISP destinations and actions.
5	**FOEN**	
4	**FCEN**	
3:0	**FCTRL[3:0]**	

ISPAH – ISP Address High Byte

7	6	5	4	3	2	1	0
ISPA[15:8]							
r/w							

Address: A7H reset value: 0000 0000B

Bit	Name	Description
7:0	**ISPA[15:8]**	**ISP address high byte.** ISPAH contains address ISPA[15:8] for ISP operations.

ISPAL – ISP Address Low Byte

7	6	5	4	3	2	1	0
ISPA[7:0]							
r/w							

Address: A6H reset value: 0000 0000B

Bit	Name	Description
7:0	**ISPA[7:0]**	**ISP address low byte.** ISPAL contains address ISPA[7:0] for ISP operations.

ISPFD – ISP Flash Data

7	6	5	4	3	2	1	0
ISPFD[7:0]							
r/w							

Address: AEH reset value: 0000 0000B

Bit	Name	Description
7:0	**ISPFD[7:0]**	**ISP flash data.** This byte contains flash data which is read from or is going to be written to the flash memory. The user should write data into ISPFD for program mode before triggering ISP processing and read data from ISPFD for read/verify mode after ISP processing is finished.

ISPTRG – ISP Trigger (TA protected)

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

Address: A4H reset value: 0000 0000B

Bit	Name	Description
0	**ISPGO**	**ISP go.** ISP begins by setting this bit as a logic 1. After this instruction, the CPU holds the Program Counter (PC) and the ISP hardware automation takes over to control the progress. After ISP action completed, the Program Counter continues to run the following instructions. The ISPGO bit will be automatically cleared and always read as logic 0.

20.2 ISP Mode Table

ISP Mode		ISPCN				ISPAH, ISPAL	ISPFD
		A17, A16	FOEN	FCEN	FCTRL[3:0]	A[15:0]	D[7:0]
Read Company ID		x, x[Note(0)]	0	0	1011	x[Note(0)]	Data out D[7:0]=DAH
Read Device ID		x, x[Note(0)]	0	0	1100	A[15:0]=0000H for low-byte ID A[15:0]=0001H for high-byte ID	Data out D[7:0]=Dev. ID
AP & Data Flash	FLASH Page Erase	0, 0	1	0	0010	Address in A[15:0]	x[Note(0)]
	FLASH Program	0, 0	1	0	0001	Address in A[15:0]	Data in D[7:0]
	FLASH Read	0, 0	0	0	0000	Address in A[15:0]	Data out D[7:0]
LD	FLASH Page Erase	0, 1	1	0	0010	Address in A[15:0]	x[Note(0)]
	FLASH Program	0, 1	1	0	0001	Address in A[15:0]	Data in D[7:0]
	FLASH Read	0, 1	0	0	0000	Address in A[15:0]	Data out D[7:0]
CONFIG[Note(1)] Page Erase		1, 1	1	0	0010	Address in A[15:0]=0000H	x[Note(0)]
CONFIG[Note(1)] Program		1, 1	1	0	0001	Address in A[15:0]	Data in D[7:0]
CONFIG[Note(1)] Read		1, 1	0	0	0000	Address in A[15:0]	Data out D[7:0]

Note:

(0) 'x' means '*don't care*'.

(1) The 'CONFIG' means the MCU hardware configuration.

(3) Each page has 128 bytes. So, the address for Page Erase should be 0000, 0080H, 0100H, 0180H, 0200H, .., which is incremented by 0080H.

20.3 Access table of N79E845 Series ISP Programming

Destination	UNLOCK		LOCK	
	ISP code residence		ISP code residence	
	APROM	LDROM	APROM	LDROM
APROM				
LDROM	[1]		[1]	
Data Flash				
CONFIGs		[2]		[2]
ID (read)				
Block color	Comment			
	Fully accessing			
	Read only			
	Accessing inhibit			
[1]	LDUE should be 1, or it will be read only			
[2]	New CONFIG functions after POR, WDT, Reset pin or S/W reset			

Note 1: CONFIG full accessing by LD while LOCK.
Note 2: Inhibit AP jump to LD or LD jump to AP.
Note 3: MCU run in APROM can't read CONFIGs.

20.4 User Guide of ISP

ISP facilitates the updating flash contents in a convenient way; however, the user should follow some restricted laws in order that the ISP operates correctly. Without noticing warnings will possible cause undetermined results even serious damages of devices. Be attention of these notices. Furthermore, this paragraph will also support useful suggestions during ISP procedures.

(1) If no more ISP operation needs, the user must clear ISPEN (CHPCON.0) to zero. It will make the system void to trigger ISP unaware. Furthermore, ISP requires internal 22.1184MHZ RC oscillator running. If the external clock source is chosen, disabling ISP will stop internal 22.1184MHz RC for saving power consumption. Note that a write to ISPEN is TA protected.

(2) CONFIG bytes can be ISP fully accessed only when loader code executing in LDROM. New CONFIG bytes other than CBS bit activate after all resets. New CBS bit activates after resets other than software reset.

(3) When the LOCK bit (CONFIG0.1) is activated, ISP reading, writing, or erasing can still be valid.

(4) ISP works under V_{DD} 2.7V ~ 5.5V.

(5) APROM and LDROM can read itself through ISP method.

Note that If the user would like to develop ISP program, always erase and program CONFIG bytes at the last step for data security.

20.5 ISP Demo Code

Common Subroutine for ISP

```
Enable_ISP:
MOV     ISPCN,#00110000b      ;select “Standby” mode
CLR     EA                    ;if any interrupt is enabled, disable temporarily
MOV     TA,#0AAH              ;CHPCON is TA-Protection
MOV     TA,#55H               ;
ORL     CHPCON,#00000001b     ;ISPEN=1, enable ISP function
SETB    EA
CALL    Trigger_ISP           ;
RET

Disable_ISP:
MOV     ISPCN,#00110000b      ;select “Standby” mode
CALL    Trigger_ISP           ;
CLR     EA                    ;if any interrupt is enabled, disable temporarily
MOV     TA,#0AAH              ;CHPCON is TA-Protection
MOV     TA,#55H               ;
ANL     CHPCON,#11111110b     ;ISPEN=0, disable ISP function
SETB    EA
RET

Trigger_ISP:
CLR     EA                    ;if any interrupt is enabled, disable temporarily
MOV     TA,#0AAH              ;ISPTRG is TA-Protection
MOV     TA,#55H               ;
MOV     ISPTRG,#00000001b     ;write ‘1’ to bit ISPGO to trigger an ISP processing
SETB    EA
RET
```

Read Company ID

```
CALL    Enable_ISP

MOV     ISPCN,#00001011b      ;select “Read Company ID” mode

CALL    Trigger_ISP
MOV     A,ISPFD               ;now, ISPFD contains Company ID (should be DAH), move to ACC for further
use

CALL    Disable_ISP
```

Read Device ID

```
CALL    Enable_ISP

MOV     ISPCN,#00001100b    ;select “Read Device ID” mode

MOV     ISPAH,#00H          ;fill address with 0000H for low-byte DID
MOV     ISPAL,#00H          ;
CALL    Trigger_ISP
MOV     A,ISPFD             ;now, ISPFD contains low-byte DID, move to ACC for further use

MOV     ISPAH,#00H          ;fill address with 0001H for high-byte DID
MOV     ISPAL,#01H          ;
CALL    Trigger_ISP
MOV     A,ISPFD             ;now, ISPFD contains high-byte DID, move to ACC for further use

CALL    Disable_ISP
```

FLASH Page Erase (target address in APROM/DataFlash/LDROM area)

```
CALL    Enable_ISP

MOV     ISPCN,#00100010b    ;select "FLASH Page Erase" mode, (A17,A16)=(0,0) for APROM/DataFlash/LDROM

MOV     ISPAH,#??H          ;fill page address
MOV     ISPAL,#??H          ;
CALL    Trigger_ISP

CALL    Disable_ISP
```

FLASH Program (target address in APROM/DataFlash/LDROM area)

```
CALL    Enable_ISP

MOV     ISPCN,#00100001b    ;select "FLASH Program" mode, (A17,A16)=(0,0) for APROM/DataFlash/LDROM

MOV     ISPAH,#??H          ;fill byte address
MOV     ISPAL,#??H          ;
MOV     ISPFD,#??H          ;fill data to be programmed
CALL    Trigger_ISP

CALL    Disable_ISP
```

FLASH Read (target address in APROM/DataFlash/LDROM area)

```
CALL    Enable_ISP

MOV     ISPCN,#00000000b    ;select "FLASH Read" mode, (A17,A16)=(0,0) for APROM/DataFlash/LDROM

MOV     ISPAH,#??H          ;fill byte address
MOV     ISPAL,#??H          ;
CALL    Trigger_ISP
MOV     A,ISPFD             ;now, ISPFD contains the Flash data, move to ACC for further use

CALL    Disable_ISP
```

Config Page Erase (target address in Config area)

```
CALL    Enable_ISP

MOV     ISPCN,#11100010b      ;select “CONFIG Page Erase” mode, (A17,A16)=(1,1) for CONFIG

MOV     ISPAH,#00H            ;fill page address #0000H, because there is only one page
MOV     ISPAL,#00H            ;
CALL    Trigger_ISP

CALL    Disable_ISP
```

Config Program (target address in Config area)

```
CALL    Enable_ISP

MOV     ISPCN,#11100001b      ;select “CONFIG Program” mode, (A17,A16)=(1,1) for CONFIG

MOV     ISPAH,#00H            ;fill byte address, 0000H/0001H/0002H/0003H for CONFIG0/1/2/3, respec-
tively
MOV     ISPAL,#??H            ;
MOV     ISPFD,#??H            ;fill data to be programmed
CALL    Trigger_ISP

CALL    Disable_ISP
```

Config Read (target address in Config area)

```
CALL    Enable_ISP

MOV     ISPCN,#11000000b      ;select “CONFIG Read” mode, (A17,A16)=(1,1) for CONFIG

MOV     ISPAH,#00H            ; fill byte address, 0000H/0001H/0002H/0003H for CONFIG0/1/2/3, respec-
tively
MOV     ISPAL,#??H            ;
CALL    Trigger_ISP
MOV     A,ISPFD               ;now, ISPFD contains the CONFIG data, move to ACC for further use

CALL    Disable_ISP
```

21 Power Management

N79E845 series have several features that help the user to control the power consumption of the device. The power saved features have the Power Down mode and the Idle mode of operation. For a stable current consumption, states of P0 pins should be taken care of.

In system power saving modes, the Watchdog Timer should be specially taken care. The hardware will clear WDT counter automatically after entering into or being woken-up from Idle or Power Down mode. It prevents unconscious system reset.

PCON – Power Control

7	6	5	4	3	2	1	0
SMOD	SMOD0	-	POF	GF1	GF0	PD	IDL
r/w	r/w	-	r/w	r/w	r/w	r/w	r/w

Address: 87H reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
1	**PD**	**Power Down mode.** Setting this bit puts MCU into Power Down mode. Under this mode, both CPU and peripheral clocks stop and Program Counter (PC) suspends. It provides the lowest power consumption. After CPU is woken up from Power Down, this bit will be automatically cleared via hardware and the program continue executing the interrupt service routine (ISR) of the very interrupt source that woke the system up before. After return from the ISR, the device continues execution at the instruction which follows the instruction that put the system into Power Down mode. Note that If IDL bit and PD bit are set simultaneously, the MCU will enter into Power Down mode. Then it does not go to Idle mode after exiting Power Down.
0	**IDL**	**Idle mode.** Setting this bit puts MCU into Idle mode. Under this mode, the CPU clock stops and Program Counter (PC) suspends. After CPU is woken up from Idle, this bit will be automatically cleared via hardware and the program continue executing the ISR of the very interrupt source that woke the system up before. After return from the ISR, the device continues execution at the instruction which follows the instruction that put the system into Idle mode.

21.1 Idle Mode

Idle mode suspends CPU processing by holding the Program Counter. No program code are fetched and run in Idle mode. This forces the CPU state to be frozen. The Program Counter (PC), the Stack Pointer (SP), the Program Status Word (PSW), the Accumulator (ACC), and the other registers hold their contents during Idle mode. The port pins hold the logical states they had at the time Idle was activated. Generally it saves considerable power of typical half of the full operating power.

Since the clock provided for peripheral function logic circuit like timer or serial port still remain in Idle mode, the CPU can be released from the Idle mode using any of the interrupt sources if enabled. The user can put the device into Idle mode by writing 1 to the bit IDL (PCON.0). The instruction that sets the IDL bit is the last instruction that will be executed before the device goes into Idle mode.

The Idle mode can be terminated in two ways. First, any interrupt if enabled will cause an exit. This will automatically clear the IDL bit, terminate the Idle mode, and the interrupt service routine (ISR) will be executed. After using the RETI instruction to jump out of the ISR, execution of the program will be the one following the instruction which put the CPU into Idle mode. The second way to terminate the Idle mode is with any reset other than software reset.

21.2 Power Down Mode

Power Down mode is the lowest power state that **N79E845** series can enter. It remain the power consumption as a ”μA” level. This is achieved by stopping the system clock no matter internal RC clock or external crystal. Both of CPU and peripheral functions like Timers or UART are frozen. Flash memory stops. All activity is completely stopped and the power consumption is reduced to the lowest possible value. The device can be put into Power Down mode by writing 1 to bit PD (PCON.1). The instruction that does this action will be the last instruction to be executed before the device goes into Power Down mode. In the Power Down mode, RAM maintains its content. The port pins output the values held by their respective.

There are two ways to exit **N79E845** series from the Power Down mode. First is with all resets except software reset. BOD reset will also wake up CPU from Power Down mode. Be sure that BOD detection is enabled before the system enters into Power Down. But for a principle of least power consumption, it is uncommon to enable BOD detection in Power Down mode. It is not a recommended application. Of course the RST pin reset and power-on reset will remove the Power Down status. After RST pin reset or power-on reset. The CPU is initialized and start executing program code from the beginning.

N79E845 series can be woken up from the Power Down mode by forcing an external interrupt pin activated, providing the corresponding interrupt enabled and the global enable EA bit (IE.7) is set. If these conditions are met, then the trigger on the external pin will asynchronously restart the system clock. Then device executes the

interrupt service routine (ISR) for the corresponding external interrupt. After the ISR is completed, the program execution returns to the instruction after the one which put the device into Power Down mode and continues.

BOD interrupt is another source to wake up CPU from Power Down. As mentioned before the user will endure the large current of BOD detection circuit. It is not a typical application.

22 Clock System

N79E845 series provide three options of the system clock source. It is configured by FOSC (CONFIG3.1~0). It switches the system clock from crystal/resonator, on-chip RC oscillator, or external clock from XTAL1 pin. **N79E845** series embed an on-chip RC oscillator of 22.1184MHz/11.0592MHz selected by CONFIG setting, factory trimmed to ± 1% at room temperature. If the external clock source is from the crystal, the frequency supports from 4MHz to 24MHz.

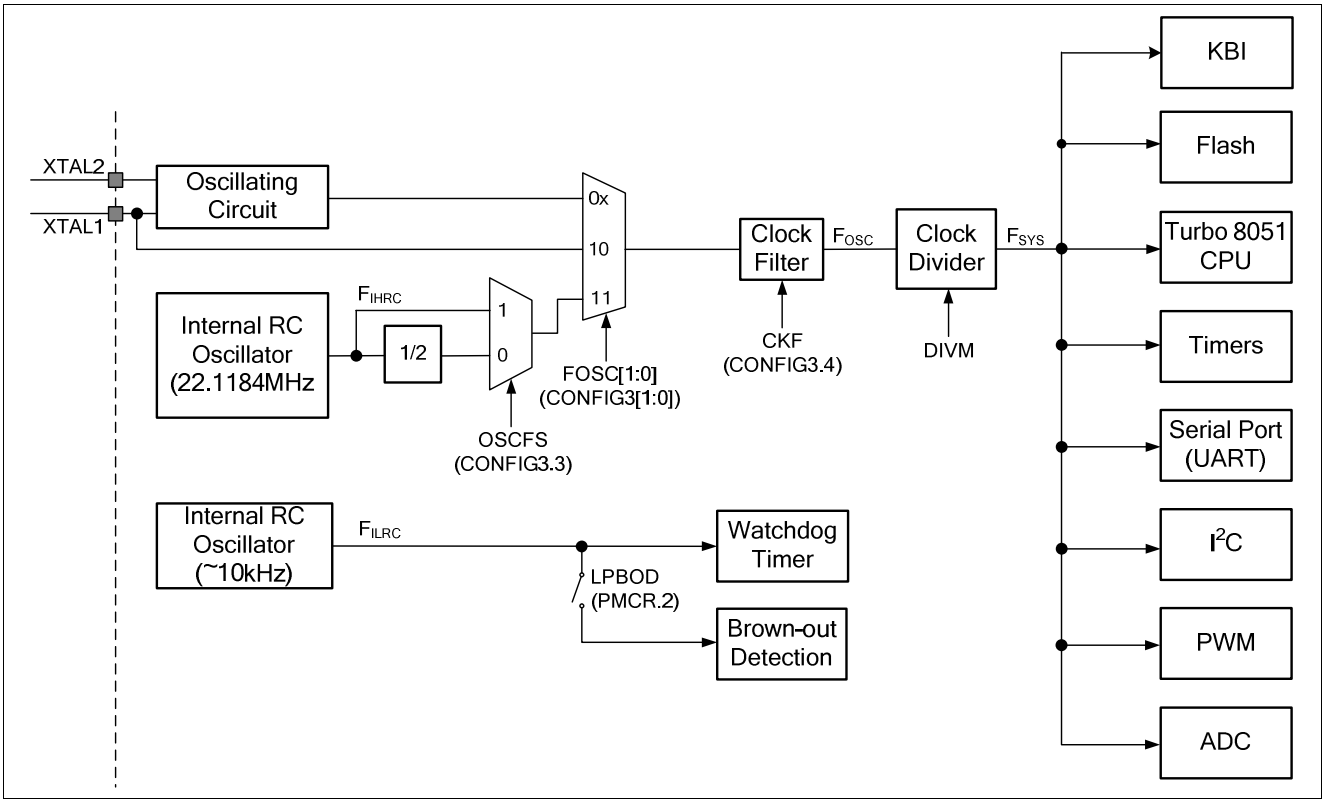


Figure 22–1. Clock System Block Diagram

7	6	5	4	3	2	1	0
CWDTEN	-	-	CKF	OSCFS	-	FOSC1	FOSC0
r/w	-	-	r/w	r/w	-	r/w	r/w

unprogrammed value: 1111 1111B

Bit	Name	Description
7	**CWDTEN**	**CONFIG Watchdog Timer enable.** 1 = Disable Watchdog Timer after all resets. 0 = Enable Watchdog Timer after all resets. WDCON.WDTEN. initial by inverse CWDTEN.

Bit	Name	Description
6	**-**	**Reserved**.
5	**-**	**Reserved**.
4	**CKF**	**Clock filter enable.** 1 = Enable clock filter. It increases noise immunity and EMC capacity. 0 = Disable clock filter.
3	**OSCFS**	**Internal RC oscillator frequency select.** 1 = Select 22.1184MHz as the system clock if internal RC oscillator mode is used. It bypasses the divided-by-2 path of internal oscillator to select 22.1184MHz output as the system clock source. 0 = Select 11.0592MHz as the system clock if internal RC oscillator mode is used. The internal RC divided-by-2 path is selected. The internal oscillator is equivalent to 11.0592MHz output used as the system clock.
2	**-**	**Reserved**.
1:0	**FOSC1** **FOSC0**	**Oscillator selection bit.** Chip clock source select. See the following table.

(FOSC1, FOSC0)	Chip clock source
(1, 1)	Internal RC oscillator
(1, 0)	External oscillator coming from XTAL1-pin
(0, x)	External crystal, 4MHz ~ 24MHz

DIVM – Clock Divider Register

7	6	5	4	3	2	1	0
DIVM[7:0]							
r/w							

Address: 95H reset value: 0000 0000B

Bit	Name	Description
7:0	DIVM[7:0]	**Clock divider.** The system clock frequency F_{SYS} follows the equation below according to DIVM value. $F_{SYS} = F_{OSC}$, while DIVM = 00H. $F_{SYS} = \frac{1}{2(DIVM+1)} \times F_{OSC}$, while DIVM = 01H ~ FFH.

22.1 External Clock Source

The system clock source can be from external XTAL1 pin. When XTAL1 pin is driven by an external clock source, XTAL2 should be left floating. XTAL1 and XTAL2 are the input and output, respectively, of an internal inverting amplifier. A crystal or resonator can be used by connecting between XTAL1 and XTAL2 pins. The crystal or resonator frequency from 4MHz up to 24MHz is allowed. CKF (CONFIG3.4) is the control bit of clock filter circuit of XTAL1 input pin.

22.2 On-Chip RC Oscillator

The on-chip RC oscillator is enabled while FOSC (CONFIG3.1~0) is 1. Setting OSCFS (CONFIG3.3) logic 1 will switch to a divided-by-2 path. Note that XTAL1 pin should be connected to V_{DD} while on-chip RC oscillator is used as the system clock source.

23 Power Monitoring

In order to prevent incorrect execution during power up and power drop, **N79E845** series provide three power monitor functions, power-on detection, BOD detection, and low power detection.

23.1 Power-on Detection

The power-on detection function is designed for detecting power up after power voltage reaches to a level where system can work. After power-on detected, the POF (PCON.4) will be set 1 to indicate a cold reset, a power-on reset complete. The POF flag can be cleared via software.

23.2 BOD Detection

The other power monitoring function, BOD detection circuit is for monitoring the V_{DD} level during execution. There are two programmable BOD trigger levels available for wide voltage applications. The two nominal levels are 2.7V and 3.8V selected via setting CBOV in CONFIG2. When V_{DD} drops to the selected BOD trigger level (V_{BOD}), the BOD detection logic will either reset the CPU or request a BOD interrupt. The user may determine BOD reset or interrupt enable according to different application systems.

The BOD detection will request the interrupt while V_{DD} drops below V_{BOD} while BORST (PMCR.4) is 0. In this case, BOF (PMCR.3) will set as a 1. After the user cleared this flag whereas V_{DD} remains below V_{BOD}, BOF will not set again. BOF just acknowledge the user a power drop occurs. The BOF will set 1 after V_{DD} goes higher than V_{BOD} to indicate a power resuming. V_{BOD} has a hysteresis of 20~200mV.

The BOD detection circuit also provides a low power BOD detection mode for power saving. When LPBOD (PMCR.2) is set 1, the BOD detection repeatedly senses the power voltage about every 12.8ms. For the interval counting, the internal 10kHz RC oscillator will turn on in BOD low power mode. Note that the hysteresis feature will disappear in low power BOD detection mode.

CONFIG2

7	6	5	4	3	2	1	0
CBODEN	CBOV	-	CBORST	-	-	-	-
r/w	r/w	-	r/w	-	-	-	-

unprogrammed value: 1111 1111B

Bit	Name	Description
7	**CBODEN**	**CONFIG BOD detect enable.** 1 = Disable BOD detection. 0 = Enable BOD detection.
6	**CBOV**	**CONFIG BOD voltage select.** This bit select one of two BOD voltage level.
5	**-**	Reserved
4	**CBORST**	**CONFIG BOD reset enable.** This bit decides if a BOD reset is caused after a BOD event. 1 = Enable BOD reset when V_{DD} drops below V_{BOD}. 0 = Disable BOD reset when V_{DD} drops below $V_{BOD.}$ Chip will assert BOF when V_{DD} drops below V_{BOD}.

Config-bits CBOV	SFR BOD	BOD voltage
1	0	Enable BOD= 2.7V
0	1	Enable BOD= 3.8V

PMCR – Power Monitoring Control (TA protected)

7	6	5	4	3	2	1	0
BODEN [1]	BOV	-	BORST[1]	BOF	LPBOD	-	-
r/w	r/w	-	r/w	r/w	r/w	-	-

Address: A3H reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
7	**BODEN**	**BOD-detect function control.** **BODEN is initialized by inverted CBODEN (CONFIG2, bit-7) at any resets.** 1 = Enable BOD detection. 0 = Disable BOD detection.

Bit	Name	Description
6	BOV	**BOD voltage selection bits:** BOD are initialized at reset with the value of bits CBOV in config3-bits BOD Voltage Select bits: **CBOV (Config-bits)** / **BOD voltage** 0 / Enable BOD= 2.7V, 1 / Enable BOD=3.8V
5	-	**Reserved**
4	BORST	**BOD reset enable.** This bit decides if a BOD reset is caused after a BOD event. 0 = Disable BOD reset when V_{DD} drops below V_{BOD}. Chip will assert BOF when V_{DD} drops below V_{BOD}. 1 = Enable BOD reset when V_{DD} drops below V_{BOD}.
3	BOF	**BOD flag.** This flag will be set as a logic 1 via hardware after a V_{DD} dropping below or rising above V_{BOD} event occurs. If both EBOD (IE.52) and EA (IE.7) are set, a BOD interrupt requirement will be generated. This bit must be cleared via software.
2	LPBOD	**Low power BOD detection enable.** This bit switches the BOD detection into a power saving mode. This bit is only effective while CBODEN = 1. 0 = Disable BOD power saving mode. BOD detection operates in normal mode if enabled. The detection is always on. 1 = Enable BOD power saving mode. BOD detection operates in power saving mode if enabled. Enable this bit will switch on internal 10kHz RC to be a timer for about 12.8ms interval of detection. The discrete detection will save much power but the hysteresis feature disappears.
1	-	**Reserved**
0	-	**Reserved**

[1] BODEN and BORST will be directly loaded from CONFIG2[7:4] after all resets.

Note that if BOF is 1 after chip reset, it is strongly recommended to initialize the user's program by clearing BOF.

PCON – Power Control

7	6	5	4	3	2	1	0
SMOD	SMOD0	-	POF	GF1	GF0	PD	IDL
r/w	r/w	-	r/w	r/w	r/w	r/w	r/w

Address: 87H reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
4	**POF**	**Power-on reset flag.** This bit will be set as 1 after a power-on reset. It indicates a cold reset, a power-on reset complete. This bit remains its value after any other resets. This flag is recommended to be cleared via software.

24 Reset Conditions

N79E845 series have several options to place device in reset condition. In general, most SFRs go to their reset value irrespective of the reset condition, but there are several reset source indicating flags whose state depends on the source of reset. There are 5 ways of putting the device into reset state. They are power-on reset, RST pin reset, software reset, Watchdog Timer reset, and BOD reset.

AUXR1 – AUX Function Resgister-1

7	6	5	4	3	2	1	0
-	-	-	-	-	-	0	DPS
-	-	-	-	-	-	r	r/w

Address: A2H reset value: 0000 0000B

Bit	Name	Description
7	**-**	**Reserved**
6	**-**	**Reserved**
5:1	**-**	**Reserved**
0	**DPS**	Dual Data Pointer Select 0: To select DPTR of standard 8051. 1: To select DPTR1

WDCON0 – Watchdog Timer Control (TA protected)

7	6	5	4	3	2	1	0
WDTEN	WDCLR	WDTF	WIDPD	WDTRF	WPS2	WPS1	WPS0
r/w	w	-	r/w	r/w	r/w	r/w	r/w

Address: D8H reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values.

Bit	Name	Description
3	**WDTF**	WDT reset flag. When the MCU resets itself, this bit is set by hardware. The bit should be cleared by software. If EWRST=0, the interrupt flag WDTF won't be set by hardware, and the MCU will reset itself right away. If EWRST=1, the interrupt flag WDTF will be set by hardware and the MCU will jump into WDT's interrupt service routine if WDT interrupt is enabled, and the MCU won't reset itself until 512 CPU clocks elapse. In other words, in this condition, the user needs also to clear the WDT counter (by writing '1' to WDCLR bit) during this period of 512 CPU clocks, or the MCU will also reset itself when 512 CPU clocks elapse.

24.1 Power-on Reset

N79E845 series incorporate an internal voltage reference. During a power-on process of rising power supply voltage V_{DD}, this voltage reference will hold the CPU in power-on reset mode when V_{DD} is lower than the voltage reference threshold. This design makes CPU not access program flash while the V_{DD} is not adequate performing the flash reading. If an undetermined operating code is read from the program flash and executed, this will put CPU and even the whole system in to a erroneous state. After a while, V_{DD} rises above the reference threshold where the system can work, the selected oscillator will start and then program code will be executed from 0000H. At the same time, a power-on flag POF (PCON.4) will be set 1 to indicate a cold reset, a power-on reset complete. Note that the contents of internal RAM will be undetermined after a power-on. The user is recommended to give initial values for the RAM block. P1.6, P1.7, P1.0 and P1.1 are forced to Quasi-bi-direction type during chip in reset state.

The POF is recommended to be cleared to 0 via software in order to check if a cold reset or warm reset performed after the next reset occurs. If a cold reset caused by power off and on, POF will be set 1 again. If the reset is a warm reset caused by other reset sources, POF will remain 0. The user may take a different course to check other reset flags and deal with the warm reset event.

24.2 BOD Reset

BOD detection circuit is for monitoring the V_{DD} level during execution. When V_{DD} drops to the selected BOD trigger level (V_{BOD}), the BOD detection logic will reset the CPU if BORST (PMCR.4) setting 1.

24.3 RST pin Reset

The hardware reset input is RST pin which is the input with a Schmitt trigger. A hardware reset is accomplished by holding the RST pin low for at least two machine-cycles to ensure detection of a valid hardware reset signal. The reset circuitry then synchronously applies the internal reset signal. Thus the reset is a synchronous operation and requires the clock to be running to cause an external reset.

Once the device is in reset condition, it will remain so as long as RST pin is 1. After the RST low is removed, the CPU will exit the reset state with in two machine-cycles and begin code executing from address 0000H. There is no flag associated with the RST pin reset condition. However since the other reset sources have flags, the external reset can be considered as the default reset if those reset flags are cleared.

If a RST pin reset applies while CPU is in Power Down mode, the way to trigger a hardware reset is slightly different. Since the Power Down mode stops system clock, the reset signal will asynchronously cause the system clock resuming. After the system clock is stable, CPU will enter into the reset state, then exit and start to execute program code from address 0000H.

24.4 Watchdog Timer Reset

The Watchdog Timer is a free running timer with programmable time-out intervals. The user can clear the Watchdog Timer at any time, causing it to restart the count. When the selected time-out occurs, the Watchdog Timer will reset the system directly. The reset condition is maintained via hardware for two machine-cycles. After the reset is removed the device will begin execution from 0000H.

Once a reset due to Watchdog Timer occurs the Watchdog Timer reset flag WDTRF (WDCON0.3) will be set. This bit keeps unchanged after any reset other than a power-on reset. The user may clear WDTRF via software.

24.5 Software Reset

N79E845 series are enhanced with a software reset. This allows the program code to reset the whole system in software approach. It is quite useful in the end of an ISP progress. For example, if an LDROM updating APROM ISP finishes and the code in APROM is correctly updated, a software reset can be asserted to reboot

CPU from the APROM in order to check the result of the updated APROM program code immediately. Writing 1 to SWRST (CHPCON.7) will trigger a software reset. Note that this bit is timed access protection. See demo code below. After a software reset the SWDTRF (RSR.0) will be automatically set via hardware. This bit will be preserved its value after all resets except power-on reset. SWDTRF can also be cleared via software.

CHPCON – Chip Control (TA protected)

7	6	5	4	3	2	1	0
SWRST	ISPF	LDUEN	-	-	-	BS	ISPEN
w	r/w	r/w	-	-	-	r/w	r/w

Address: 9FH reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
7	**SWRST**	**Software reset.** To set this bit as a logic 1 will cause a software reset. It will automatically be cleared via hardware after reset in finished.

The software demo code are listed below.

```
CLR EA                  ;if any interrupt is enabled, disable temporarily
MOV TA,#0AAh            ;TA protection.
MOV TA,#55h             ;
ANL CHPCON,#0FDh        ;BS = 0, reset to APROM.
MOV TA,#0AAh
MOV TA,#55h
ORL CHPCON,#80h         ;Software reset
```

24.6 Boot Select

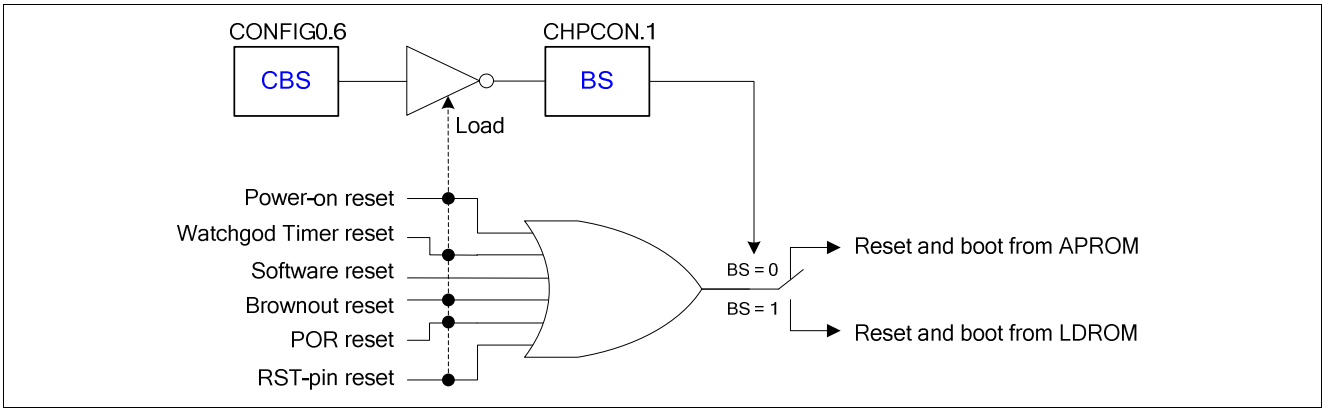


Figure 24–1. Boot Selecting Diagram

N79E845 series provide users a flexible boot selection for variant application. The SFR bit BS in CHPCON.1 determines CPU booting from APROM or LDROM after any source of reset. If reset occurs and BS is 0, CPU will reboot from APPROM. Else, the CPU will reboot from LDROM.

CONFIG0

7	6	5	4	3	2	1	0
CBS	-	-	-	-	-	LOCK	DFEN
r/w	-	-	-	-	-	r/w	r/w

unprogrammed value: 1111 1111B

Bit	Name	Description
7	**CBS**	**CONFIG boot select.** This bit defines from which block MCU boots after all resets except software reset. 1 = MCU will boot from APROM after all resets except software reset. 0 = MCU will boot from LDROM after all resets except software reset.

CHPCON – Chip Control (TA protected)

7	6	5	4	3	2	1	0
SWRST	ISPF	LDUEN	-	-	-	BS[1]	ISPEN
w	r/w	r/w	-	-	-	r/w	r/w

Address: 9FH reset value: see Table 7–2. **N79E845** Series SFR Descriptions and Reset Values

Bit	Name	Description
1	**BS**	**Boot select.** There are different meanings of writing to or reading from this bit. Writing: It defines from which block MCU boots after all resets. 0 = The next rebooting will be from APROM. 1 = The next rebooting will be from LDROM. Reading: It indicates from which block MCU booted after previous reset. 0 = The previous rebooting is from APROM. 1 = The previous rebooting is from LDROM.

[1] Note that this bit is initialized by being loaded from the inverted value of CBS bit in CONFIG0.7 at all resets except software reset. It keeps unchanged after software reset.

Note that after the CPU is released from all reset state, the hardware will always check the BS bit instead of the CBS bit to determine from APROM or LDROM that the device reboots.

24.7 Reset State

The reset state does not affect the on-chip RAM. The data in the RAM will be preserved during the reset. Note that the RAM contents may be lost if the V_{DD} falls below approximately 1.2V. This is the minimum voltage level required for RAM data retention. Therefore, after the power-on reset the RAM contents will be indeterminate. During a power fail condition. If the power falls below the data retention minimum voltage, the RAM contents will also lose.

After a reset, most of SFRs go to their initial values except bits which are affected by different reset events. See the notes of Table 7–2. **N79E845** Series SFR Descriptions and Reset Values. for the initial state of all SFRs. Some special function registers initial value depends on different reset sources. Refer to Table 24- for the detail. The Program Counter is forced to 0000H and held as long as the reset condition is applied. Note that the Stack Pointer is also reset to 07H, therefore the stack contents may be effectively lost during the reset event even though the RAM contents are not altered.

After a reset, interrupts and Timers are disabled. The Watchdog Timer is disabled if the reset source was a power-on reset. The I/O port SFRs have FFH written into them which puts the port pins in a high state.

Table 24-2 Initial State of SFR Caused by Different Resets

SFR	Power-on Reset	Watchdog Reset	Software/ External Reset	BOD Reset	With Time Access Protection
WDCON0 (D8H)	C000 0000B b7(ENWDT)= /CENWDT(config3.7)	C0Uu 1UUUB	C0UU UUUUB		Y
WDCON1 (ABH)	0000 0000B				Y
ISPTRG (A4H)	XXXX XXX0B				Y
PMCR (A3H)	CXCC 10XXB b[7:4]=config2	UXUU U0XXB		UXUU 10XXB	Y

CHPCON (9FH)	0000 00C0B b1(BS)=/CBS	000X XUU0B			Y
SHBDA (9CH)	Config1	Unchanged			Y
PCON (87H)	0001 000b	00uu 0000b	00uu 0000b (SW/External reset)	00uu 0000b	N

Note: The writes of AAH and 55H must occur within 3 machine-cycles of each other. Interrupts should be disabled during this procedure to avoid delay between these two writes.

25 Config Bits (CONFIG)

N79E845 series have several hardware configuration bytes, called CONFIG bits, those are used to configure the hardware options such as the security bits, system clock source, and so on. These hardware options can be re-configured through the Programmer/Writer or ISP modes. **N79E845** series have four CONFIG bits those are CONFIG0~3. Several functions which are defined by certain CONFIG bits are also available to be re-configured by certain SFR bits. Therefore, there is a need to load such CONFIG bits into respective SFR bits. Such loading will occurs after resets. (Software reset will reload all CONFIG bits except CBS bit in CONFIG0.7) These SFR bits can be continuously controlled via user's software. Other resets will remain the values in these SFR bits unchanged.

Note that CONFIG bits marked as "-" should always keep unprogrammed.

25.1 Config0

7	6	5	4	3	2	1	0
CBS	-	-	-	-	-	LOCK	DFEN
r/w	-	-	-	-	-	r/w	r/w

unprogrammed value: 1111 1111B

Bit	Name	Description
7	**CBS**	**CONFIG boot select.** This bit defines from which block MCU boots after all resets except software reset. 1 = MCU will boot from APROM after all resets except software reset. 0 = MCU will boot from LDROM after all resets except software reset.
6:2	**-**	**Reserved.**

Bit	Name	Description
1	**LOCK**	**Chip lock enable.** 1 = Chip is unlocked. All of APROM, LDROM, and Data Flash are not locked. Their contents can be read out through a parallel Programmer/Writer. 0 = Chip is locked. APROM, LDROM, and Data Flash are locked. Their contents read through parallel Programmer/Writer will become FFH. Note that CONFIG bytes are always unlocked and can be read. Hence, once the chip is locked, the CONFIG bytes cannot be erased or programmed individually. The only way to disable chip lock is to use the whole chip erase mode. However, all data within APROM, LDROM, Data Flash, and other CONFIG bits will be erased when this procedure is executed. If the chip is locked, it does not alter the ISP function.
0	**DFEN**	**Data Flash enable. (N79E845 only)** 1 = There is no Data Flash space. The APROM size is 16k-byte. 0 = Data Flash exists. The Data Flash and APROM share 16k bytes depending on SHBDA setting.

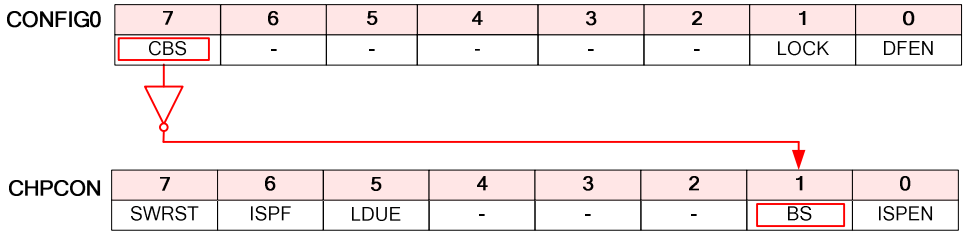


Figure 25–1. CONFIG0 Reset Reloading Except Software Reset

25.2 Config1 (N79E845 Only)

7	6	5	4	3	2	1	0
CHBDA[7:0][1]							
r/w							

unprogrammed value: 1111 1111B

Bit	Name	Description
7:0	**CHBDA[7:0]**	**CONFIG high byte of Data Flash starting address.** This byte is valid only when DFEN (CONFIG0.0) being 0 condition. It is used

Bit	Name	Description
		to determine the starting address of the Data Flash.

[1] Note that there will be no APROM if setting CHBDA 00H. CPU will execute codes in minimum size(256B) of internal Program Memory.

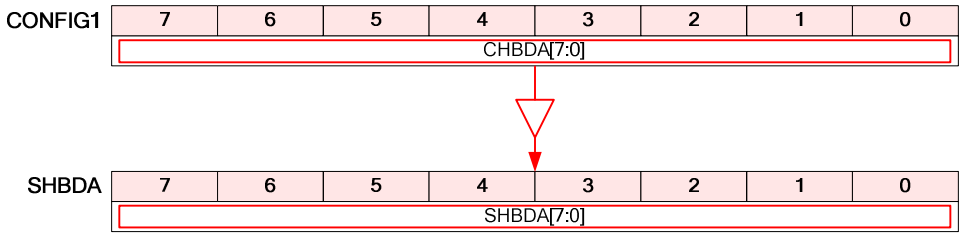


Figure 25–2. CONFIG1 Reset Reloading

25.3 Config2

7	6	5	4	3	2	1	0
CBODEN	CBOV	-	CBORST	-	-	-	-
r/w	r/w	-	r/w	-	-	-	-

unprogrammed value: 1111 1111B

Bit	Name	Description
7	**CBODEN**	**CONFIG BOD detect enable.** 1 = Enable BOD detection. 0 = Disable BOD detection.
6	**CBOV**	**CONFIG BOD voltage select.** This bit selects one of two BOD voltage level. **CBOV** / **BOD voltage** 1 / Enable V_{BOD}= 2.7V 0 / Enable V_{BOD}=3.8V
5	**-**	**Reserved**.
4	**CBORST**	**CONFIG BOD reset enable.** This bit decides if a BOD reset is caused after a BOD event. 1 = Enable BOD reset when V_{DD} drops below V_{BOD}. 0 = Disable BOD reset when V_{DD} drops below V_{BOD}.
3:0	**-**	**Reserved.**

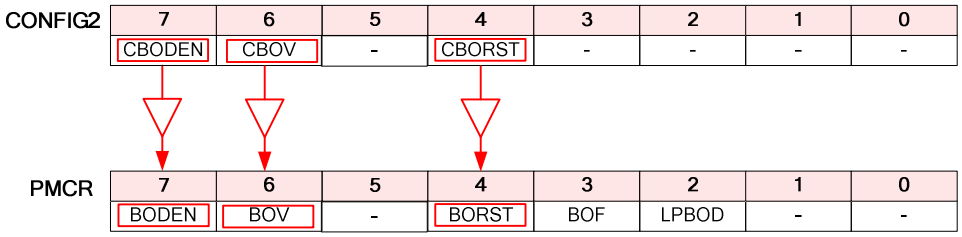


Figure 25–3. CONFIG2 Reset Reloading

25.4 Config3

7	6	5	4	3	2	1	0
CWDTEN	-	-	CKF	OSCFS	-	FOSC1	FOSC0
r/w	-	-	r/w	r/w	-	r/w	r/w

unprogrammed value: 1111 1111B

Bit	Name	Description
7	**CWDTEN**	**CONFIG Watchdog Timer enable.** 1 = Disable Watchdog Timer after all resets. 0 = Enable Watchdog Timer after all resets. WDCON.WDTEN. initial by inverse CWDTEN.
6	**-**	**Reserved**.
5	**-**	**Reserved**.
4	**CKF**	**Clock filter enable.** 1 = Enable clock filter. It increases noise immunity and EMC capacity. 0 = Disable clock filter.
3	**OSCFS**	**Internal RC oscillator frequency select.** 1 = Select 22.1184MHz as the system clock if internal RC oscillator mode is used. It bypasses the divided-by-2 path of internal oscillator to select 22.1184MHz output as the system clock source. 0 = Select 11.0592MHz as the system clock if internal RC oscillator mode is used. The internal RC divided-by-2 path is selected. The internal oscillator is equivalent to 11.0592MHz output used as the system clock.
2	**-**	**Reserved**.
1	**FOSC1**	**Oscillator selection bit.**

Bit	Name	Description
0	**FOSC0**	Chip clock source select. See the following table.

(FOSC1, FOSC0)	Chip clock source
(1, 1)	Internal RC oscillator
(1, 0)	External oscillator coming from XTAL1-pin
(0, x)	External crystal, 4MHz ~ 24MHz

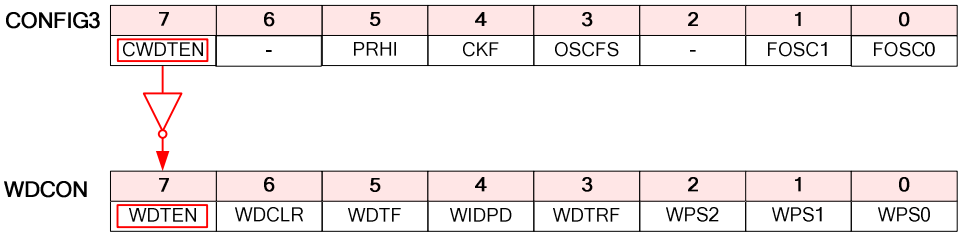


Figure 25–4. CONFIG3 Reset Reloading

26 Instruction Sets

N79E845 series execute all the instructions of the standard 8051 family. All instructions are coded within an 8-bit field called an OPCODE. This single byte must be fetched from Program Memory. The OPCODE is decoded by the CPU. It determines what action the microcontroller will take and whether more operation data is needed from memory. If no other data is needed, then only one byte was required. Thus the instruction is called a one byte instruction. In some cases, more data is needed. These will be two or three byte instructions.

Table 26–1 lists all instructions in details. Note of the instruction set and addressing modes are shown below.

Rn (n = 0~7)	Register R0~R7 of the currently selected Register Bank.
	Direct 8-bit internal data location's address. This could be an internal data RAM location (0~127) or a SFR (e.g. I/O port, control register, status register, etc.) (128~255).
@Ri (i = 0, 1)	8-bit internal data RAM location (0~255) addressed indirectly through register R0 or R1.
#data	8-bit constant included in the instruction.
#data16	16-bit constant included in the instruction.
addr16	16-bit destination address. Used by LCALL and LJMP. A branch can be any where within the 64k-byte Program Memory address space.
addr11	11-bit destination address. Used by ACALL and AJMP. The branch will be within the same 2k-byte page of Program Memory as the first byte of the following instruction.
rel	Signed (2's complement) 8-bit offset byte. Used by SJMP and all conditional branches. The range is -128 to +127 bytes relative to first byte of the following instruction.
bit	Direct addressed bit in internal data RAM or SFR.

Table 26–1. Instruction Set for N79E845 series

Instruction	OPCODE	Bytes	Clock cycles	N79E845 V.S. tradition 80C51 speed ratio
NOP	00	1	4	3.0
ADD A, Rn	28~2F	1	4	3.0
ADD A, @Ri	26, 27	1	4	3.0
ADD A, direct	25	2	8	1.5
ADD A, #data	24	2	8	1.5
ADDC A, Rn	38~3F	1	4	3.0
ADDC A, @Ri	36, 37	1	4	3.0
ADDC A, direct	35	2	8	1.5
ADDC A, #data	34	2	8	1.5
SUBB A, Rn	98~9F	1	4	3.0

Table 26–1. Instruction Set for N79E845 series

Instruction		OPCODE	Bytes	Clock cycles	N79E845 V.S. tradition 80C51 speed ratio
SUBB	A, @Ri	96, 97	1	4	3.0
SUBB	A, direct	95	2	8	1.5
SUBB	A, #data	94	2	8	1.5
INC	A	04	1	4	3.0
INC	Rn	08~0F	1	4	3.0
INC	@Ri	06, 07	1	4	3.0
INC	direct	05	2	8	1.5
INC	DPTR	A3	1	8	3.0
DEC	A	14	1	4	3.0
DEC	Rn	18~1F	1	4	3.0
DEC	@Ri	16, 17	1	4	3.0
DEC	direct	15	2	8	1.5
DEC	DPTR	A5	1	8	-
MUL	AB	A4	1	20	2.4
DIV	AB	84	1	20	2.4
DA	A	D4	1	4	3.0
ANL	A, Rn	58~5F	1	4	3.0
ANL	A, @Ri	56, 57	1	4	3.0
ANL	A, direct	55	2	8	1.5
ANL	A, #data	54	2	8	1.5
ANL	direct, A	52	2	8	1.5
ANL	direct, #data	53	3	12	2.0
ORL	A, Rn	48~4F	1	4	3.0
ORL	A, @Ri	46, 47	1	4	3.0
ORL	A, direct	45	2	8	1.5
ORL	A, #data	44	2	8	1.5
ORL	direct, A	42	2	8	1.5
ORL	direct, #data	43	3	12	2.0
XRL	A, Rn	68~6F	1	4	3.0
XRL	A, @Ri	66, 67	1	4	3.0
XRL	A, direct	65	2	8	1.5
XRL	A, #data	64	2	8	1.5
XRL	direct, A	62	2	8	1.5
XRL	direct, #data	63	3	12	2.0
CLR	A	E4	1	4	3.0
CPL	A	F4	1	4	3.0
RL	A	23	1	4	3.0
RLC	A	33	1	4	3.0
RR	A	03	1	4	3.0
RRC	A	13	1	4	3.0

Table 26–1. Instruction Set for N79E845 series

Instruction	OPCODE	Bytes	Clock cycles	N79E845 V.S. tradition 80C51 speed ratio
SWAP A	C4	1	4	3.0
MOV A, Rn	E8~EF	1	4	3.0
MOV A, @Ri	E6, E7	1	4	3.0
MOV A, direct	E5	2	8	1.5
MOV A, #data	74	2	8	1.5
MOV Rn, A	F8~FF	1	4	3.0
MOV Rn, direct	A8~AF	2	8	3.0
MOV Rn, #data	78~7F	2	8	1.5
MOV @Ri, A	F6, F7	1	4	3.0
MOV @Ri, direct	A6, A7	2	8	3.0
MOV @Ri, #data	76, 77	2	8	1.5
MOV direct, A	F5	2	8	1.5
MOV direct, Rn	88~8F	2	8	3.0
MOV direct, @Ri	86, 87	2	8	3.0
MOV direct, direct	85	3	12	2.0
MOV direct, #data	75	3	12	2.0
MOV DPTR, #data16	90	3	12	2.0
MOVC A, @A+DPTR	93	1	8	3.0
MOVC A, @A+PC	83	1	8	3.0
MOVX A, @Ri[1]	E2, E3	1	8	3.0
MOVX A, @DPTR[1]	E0	1	8	3.0
MOVX @Ri, A[1]	F2, F3	1	8	3.0
MOVX @DPTR, A[1]	F0	1	8	3.0
PUSH direct	C0	2	8	3.0
POP direct	D0	2	8	3.0
XCH A, Rn	C8~CF	1	4	3.0
XCH A, @Ri	C6, C7	1	4	3.0
XCH A, direct	C5	2	8	1.5
XCHD A, @Ri	D6, D7	1	4	3.0
CLR C	C3	1	4	3.0
CLR bit	C2	2	8	1.5
SETB C	D3	1	4	3.0
SETB bit	D2	2	8	1.5
CPL C	B3	1	4	3.0
CPL bit	B2	2	8	1.5
ANL C, bit	82	2	8	3.0
ANL C, /bit	B0	2	8	3.0
ORL C, bit	72	2	8	3.0
ORL C, /bit	A0	2	8	3.0

Table 26–1. Instruction Set for N79E845 series

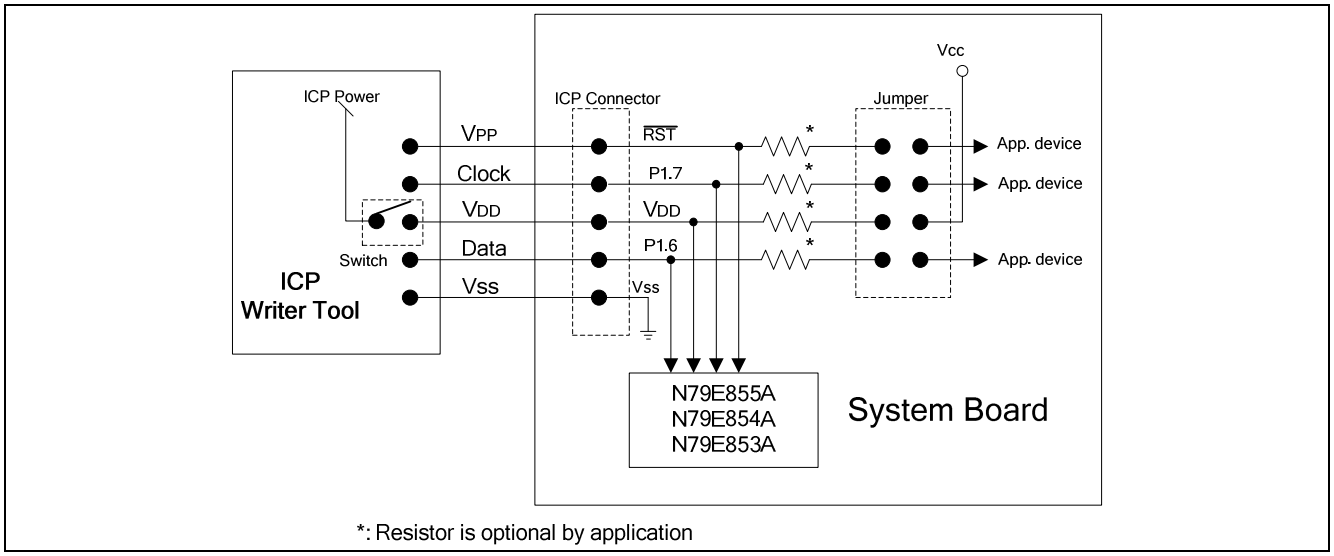
Instruction		OPCODE	Bytes	Clock cycles	N79E845 V.S. tradition 80C51 speed ratio
MOV	C, bit	A2	2	8	1.5
MOV	bit, C	92	2	8	3.0
ACALL	addr11	11, 31, 51, 71, 91, B1, D1, F1[2]	2	12	2.0
LCALL	addr16	12	3	16	1.5
RET		22	1	8	3.0
RETI		32	1	8	3.0
AJMP	addr11	01, 21, 41, 61, 81, A1, C1, E1[3]	2	12	2.0
LJMP	addr16	02	3	16	1.5
JMP	@A+DPTR	73	1	8	3.0
SJMP	rel	80	2	12	2.0
JZ	rel	60	2	12	2.0
JNZ	rel	70	2	12	2.0
JC	rel	40	2	12	2.0
JNC	rel	50	2	12	2.0
JB	bit, rel	20	3	16	1.5
JNB	bit, rel	30	3	16	1.5
JBC	bit, rel	10	3	16	1.5
CJNE	A, direct, rel	B5	3	16	1.5
CJNE	A, #data, rel	B4	3	16	1.5
CJNE	@Ri, #data, rel	B6, B7	3	16	1.5
CJNE	Rn, #data, rel	B8~BF	3	16	1.5
DJNZ	Rn, rel	D8~DF	2	12	2.0
DJNZ	direct, rel	D5	3	16	1.5

[1] The most three significant bits in the 11-bit address [A10:A8] decide the ACALL hex code. The code will be [A10,A9,A8,1,0,0,0,1].

[2] The most three significant bits in the 11-bit address [A10:A8] decide the AJMP hex code. The code will be [A10,A9,A8,0,0,0,0,1].

27 In-Circuit Program (ICP)

The ICP (In-Circuit-Program) mode is another approach to access the Flash EPROM. There are only 3 pins needed to perform the ICP function. One is input /RST pin, which must be fed to GND in the ICP working period. One is clock input, shared with P1.7, which accepts serial clock from external device. Another is data I/O pin, shared with P1.6, that an external ICP program tool shifts in/out data via P1.6 synchronized with clock(P1.7) to access the Flash EPROM of **N79E845** series. User may refer to http://www.manley.com.cn/english/index.asp for ICP Program Tool.
Upon entry into ICP program mode, all pin will be set to quasi-bidirectional mode, and output to level "1". The **N79E845** series support programming of Flash EPROM (16K/8K/4K bytes AP Flash EPROM) and DataFlash memory (**128** bytes per page). User has the option to program the AP flash and DataFlash either individually or both.



Note:
1. When use ICP to upgrade code, the /RST, P1.6 and P1.7 must be taken within design system board.
2. After program finished by ICP, to suggest system power must power off and remove ICP connector then power on.
3. It is recommended that user performs erase function and programming configure bits continuously without any interruption.

28 Electrical Characteristics

28.1 Absolute Maximum Ratings

Parameter	Rating	Unit
Operating temperature under bias	-40 to +85	°C
Storage temperature range	-55 to +150	°C
Voltage on V_{DD} pin to V_{SS}	-0.3 to +6.5	V
Voltage on any other pin to V_{SS}	-0.3 to (V_{DD}+0.3)	V

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.

28.2 DC Electrical Characteristics

Temperature = 25°C; V_{SS} = 0V;

V_{DD} = 2.4V to 5.5V @ Freq = 4MHz to 24MHz

Table 25–1. DC Characteristics

(V_{DD}–V_{SS} = 2.4~5.5V, TA = -40~85°C, unless otherwise specified.)

Sym	Parameter	Test Condition	MIN	TYP	MAX	Unit
V_{IL}	Input Low Voltage (GPIO with TTL input)	2.4 < V_{DD} < 5.5V	-0.5		0.2V_{DD}-0.1	V
V_{IL1}	Input Low Voltage (GPIO with Schmitt trigger input)	2.4 < V_{DD} < 5.5V	-0.5		0.3V_{DD}	V
V_{IL2}	Input Low Voltage (/RST, XTAL1)	2.4 < V_{DD} < 5.5V	-0.5		0.2V_{DD}-0.1	V
V_{IH}	Input High Voltage (GPIO with TTL input)	2.4 < V_{DD} < 5.5V	0.2V_{DD}+0.9		V_{DD}+0.5	V

Table 25–1. DC Characteristics

(V_{DD}–V_{SS} = 2.4~5.5V, TA = -40~85°C, unless otherwise specified.)

Sym	Parameter	Test Condition	MIN	TYP	MAX	Unit
V_{IH1}	Input High Voltage (GPIO with Schmitt trigger input)	2.4 < V_{DD} < 5.5V	0.7V_{DD}		V_{DD}+0.5	V
V_{IH2}	Input High Voltage (/RST, XTAL1)	2.4 < V_{DD} < 5.5V	0.7V_{DD}		V_{DD}+0.5	V
V_{OL}	Output Low Voltage (GPIO of P0,,P3, all modes except input only)	V_{DD}=4.5V, I_{OL}= **20**mA[3], [4]			0.45	V
		V_{DD}=3.0V, I_{OL}= **14**mA[3], [4]			0.45	V
		V_{DD}=2.4V, I_{OL}= **10**mA[3], [4]			0.45	V
V_{OL1}	Output Low Voltage (P10, P11, P14, P16, P17) (All modes except input only)	V_{DD}=4.5V, I_{OL}= **40**mA[3], [4]			0.45	V
		V_{DD} =3.0V, I_{OL}= **27**mA[3], [4]			0.45	V
		V_{DD} =2.4V, I_{OL}= **20**mA[3], [4]			0.45	V
V_{OH}	Output High Voltage (GPIO, quasi bi-directional)	V_{DD}=4.5V I_{OH}= **-380**µA[4]	2.4			V
		V_{DD}=3.0V I_{OH}= **-90**µA[4]	2.4			V
		V_{DD}=2.4V I_{OH}= **-48**µA[4]	2.0			V
V_{OH1}	Output High Voltage (GPIO, push-pull)	V_{DD}=4.5V I_{OH}= **-28.0**mA[3], [4]	2.4			V
		V_{DD}=3.0V I_{OH}= **-7**mA[3], [4]	2.4			V
		V_{DD}=2.4V I_{OH}= **-3.5**mA[3], [4]	2.0			V

Table 25–1. DC Characteristics

(V_{DD}–V_{SS} = 2.4~5.5V, TA = -40~85°C, unless otherwise specified.)

Sym	Parameter	Test Condition	MIN	TYP	MAX	Unit
I_{IL}	Logical 0 Input Current (GPIO, quasi bi-direction)	V_{DD}=5.5V, V_{IN}=0.4V		**-40 @5.5V**	-50	µA
I_{TL}	Logical 1 to 0 Transition Current (GPIO, quasi bi-direction)	V_{DD}=5.5V, V_{IN}=2.0V[2]		**-550 @5.5V**	-650	µA
I_{LI}	Input Leakage Current (GPIO, open-drain or input only)	0 < V_{IN} < V_{DD}+0.5		**<1**	±10	µA
I_{OP}	OP Current (Active mode[5])	XTAL 12MHz, V_{DD}=5.0V		**3.1**		mA
		XTAL 24MHz, V_{DD}=5.5V		**6.4**		mA
		XTAL 12MHz, V_{DD}=3.3V		**1.7**		mA
		XTAL 24MHz, V_{DD}=3.3V		**3.2**		mA
		Internal 22.1184MHz,V_{DD}=5V		**2.3**		mA
		Internal 22.1184MHz,V_{DD}=3.3V		**2.2**		mA
I_{IDLE}	IDLE Current	XTAL 12MHz, V_{DD}=5.0V		**2.7**		mA
		XTAL 24MHz, V_{DD}=5.5V		**4.4**		mA
		XTAL 12MHz, V_{DD}=3.3V		**1.3**		mA
		XTAL 24MHz, V_{DD}=3.3V		**2.3**		mA
		Internal 22.1184MHz,V_{DD}=5V		**1.6**		mA

Table 25–1. DC Characteristics

(V_{DD}–V_{SS} = 2.4~5.5V, TA = -40~85°C, unless otherwise specified.)

Sym	Parameter	Test Condition	MIN	TYP	MAX	Unit
		Internal 22.1184MHz,V_{DD}=3.3V		1.5		mA
I_{PD}	Power-down mode			<5	30	μA
R_{RST}	RST-pin Internal Pull-High Resistor	2.4 < V_{DD} < 5.5V	100		250	KΩ
V_{BOD42}	BOD38 Detect Voltage	2.4 < V_{DD} < 5.5V		3.8		V
V_{BOD27}	BOD27 Detect Voltage	2.4 < V_{DD} < 5.5V		2.7		V

[1]: Typical values are not guaranteed. The values listed are tested at room temperature and based on a limited number of samples.

[2]: Pins of ports 0~3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.

[3]: Under steady state (non-transient) conditions, I_{OL}/I_{OH} must be externally limited as follows:
Maximum I_{OL}/I_{OH} of P0, P3 per port pin: 20 mA
Maximum I_{OL}/I_{OH} of P10, P11, P14, P16, P17: 40 mA
Maximum total I_{OL}/I_{OH} for all outputs: 100mA (Through V_{DD} total current)
Maximum total I_{OL}/I_{OH} for all outputs: 150mA (Through V_{SS} total current)

[4]: If I_{OH} exceeds the test condition, V_{OH} will be lower than the listed specification.
If I_{OL} exceeds the test condition, V_{OL} will be higher than the listed specification.

[5]: Tested while CPU is kept in reset state.

[6]: GPIO mean the general purpose I/O, such as P0, P1, P3.

Other: P1.2 and P1.3 are open drain structure. They have not quasi or push pull modes.

29 Analog Electrical Characteristics

29.1 Specification of LDO Regulator

Parameter	Unit	MIN	TYP	MAX	Note
Input Voltage	V_{DD}	2.4	5	5.5	
Output Voltage	V_{LDO}	-10%	1.8	+10%	

29.2 Specification of 10-bits SAR-ADC

	Symbol	Test Condition	MIN	TYP	MAX	Unit
Operation voltage	VDD	V_{DD}	2.7		5.5	V
Resolution					10	bit
Conversion time				$35t_{ADC}$[1]		us
Sampling rate					150K	Hz
Integral Non-Linearity Error	INL		-1		1	LSB
Differential Non-Linearity	DNL		-1		1	LSB
Gain error	Ge		-1		1	LSB
Offset error	Ofe		-3		3	LSB
Clock frequency	ADCCLK				5.25	MHz
Absolute error			-2		2	LSB
Band-gape	V_{BG}		1.15	1.3	1.45	V

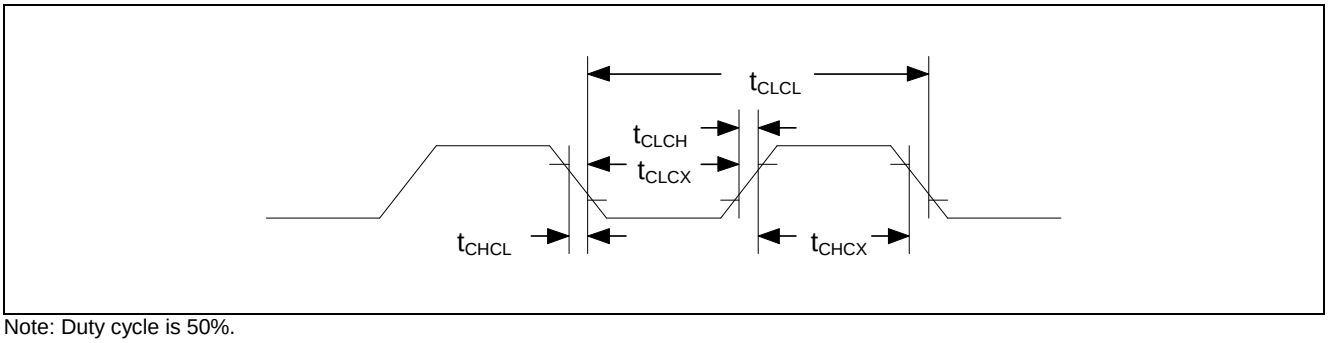
Notes: 1. t_{ADC}: The period time of ADC input clock

30 External Crystal and Internal RC Specification

30.1 Specification of 4~24MHz XTAL Oscillator

Parameter	Condition	MIN.	TYP.	MAX.	Unit
Input clock frequency	External crystal	4		24	MHz

Parameter	Symbol	MIN.	TYP.	MAX.	Units	Notes
Oscillator Frequency	$1/t_{CLCL}$	0	24	MHz		
Clock High Time	t_{CHCX}	20.8	-	-	nS	
Clock Low Time	t_{CLCX}	20.8	-	-	nS	
Clock Rise Time	t_{CLCH}	-	-	10	nS	
Clock Fall Time	t_{CHCL}	-	-	10	nS	



Note: Duty cycle is 50%.

30.2 Specification of Internal RC Oscillator-22.1184MHz

Parameter	Condition	MIN.	TYP.	MAX.	Unit
Center Frequency			22.1184		MHz
Internal Oscillator Frequency	$+25^0$C	-1		+1	%
	-40^0C~$+85^0$C;	-5		+5	%

30.3 Specification of Internal RC Oscillator-10KHz

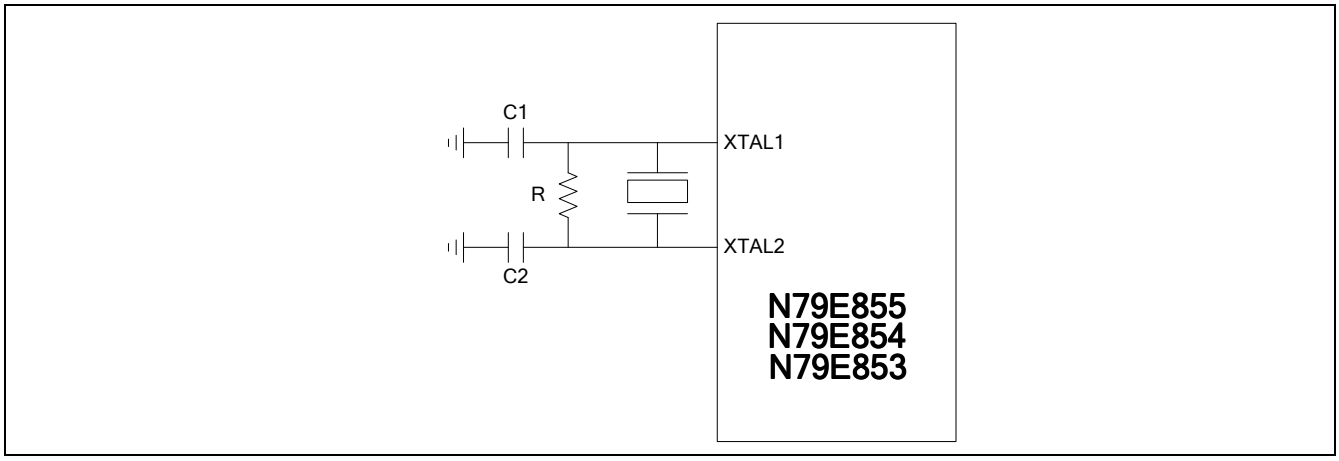
Parameter	Condition	MIN.	TYP.	MAX.	Unit

Center Frequency		5	10	15	KHz

31 Typical Application Circuits

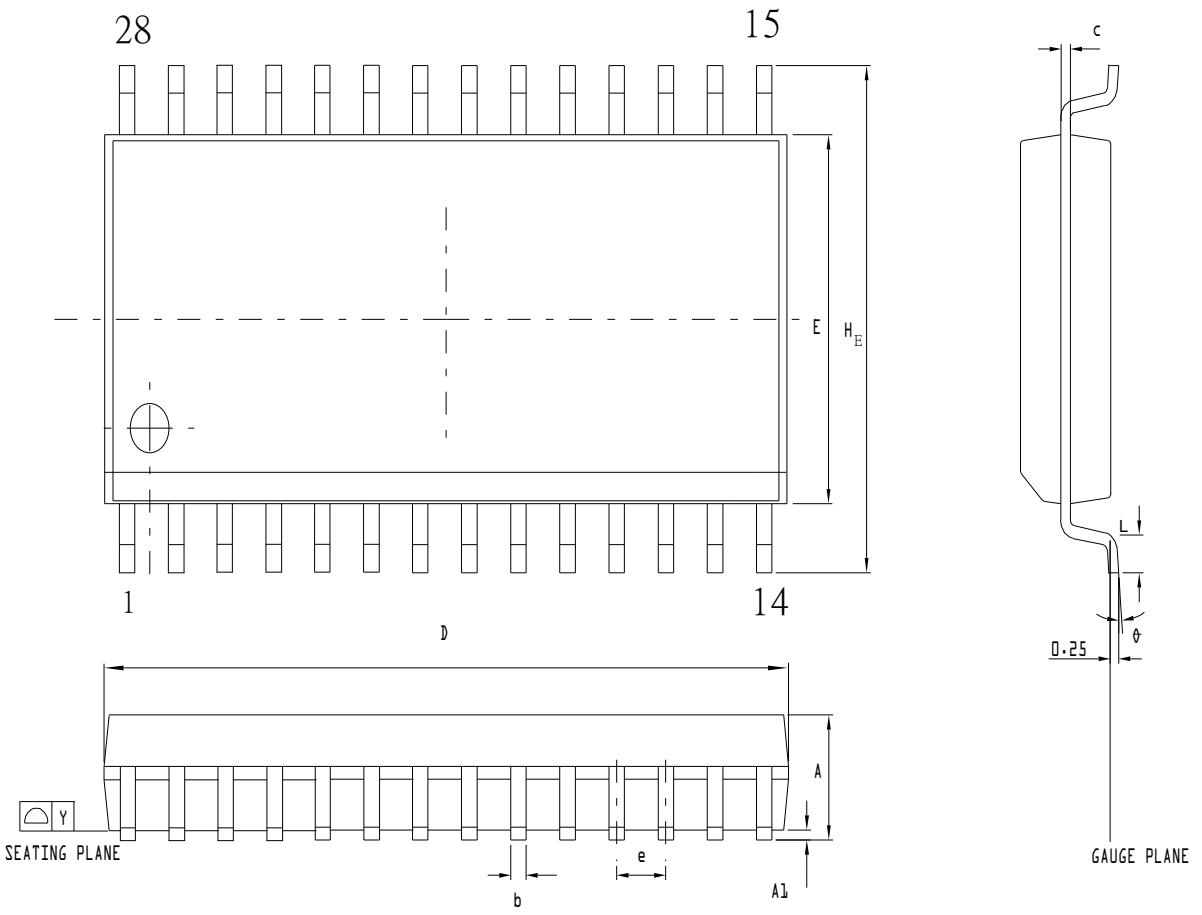
CRYSTAL	C1	C2	R
4MHz ~ 24 MHz	without	without	without

The above table shows the reference values for crystal applications.



32 Package Dimensions

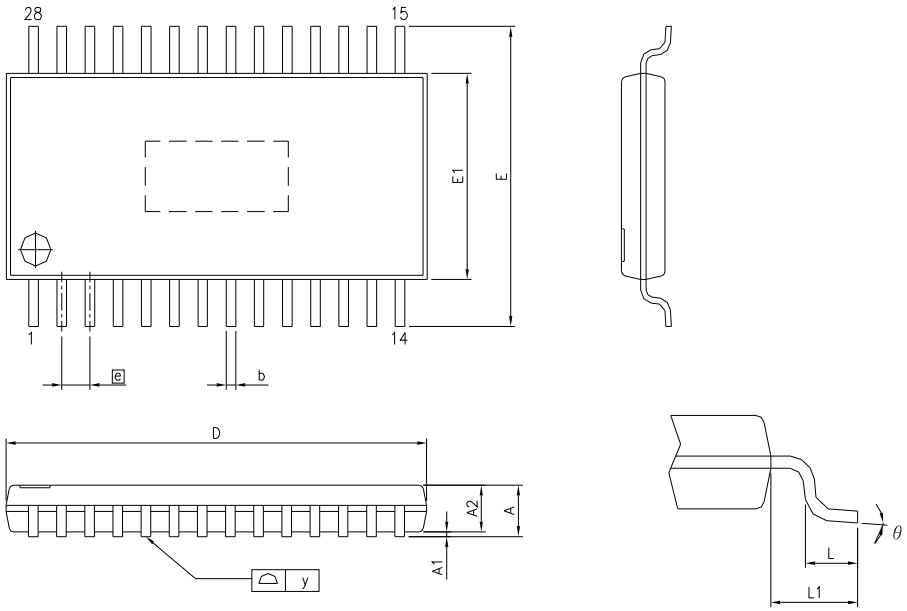
32.1 28-pin SOP - 300 mil



Control demensions are in milmeter

SYMBOL	DIMENSION IN MM		DIMENSION IN INCH	
	MIN.	MAX.	MIN.	MAX.
A	2.35	2.65	0.093	0.104
A1	0.10	0.30	0.004	0.012
b	0.33	0.51	0.013	0.020
c	0.23	0.32	0.009	0.013
E	7.40	7.60	0.291	0.299
D	17.70	18.10	0.697	0.713
e	1.27 BSC		0.050 BSC	
H_E	10.00	10.65	0.394	0.419
Y	———	0.10	———	0.004
L	0.40	1.27	0.016	0.050
θ	0	8	0	8

32.2 28-pin TSSOP - 4.4X9.7mm



Controlling Dimension :Millimeters

	Dimension in mm			Dimension in inch		
SYMBOLS	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	–	–	1.20	–	–	0.0472
A1	0.00	–	0.15	0.00	–	0.0059
A2	0.80	1.00	1.05	0.0314	0.0393	0.4133
b	0.19	–	0.30	0.0074	–	0.0118
D	9.60	9.70	9.80	0.3779	0.3818	0.3858
E1	4.30	4.40	4.50	0.1692	0.1732	0.1771
E	6.40 BSC			0.2519 BSC		
e	0.65 BSC			0.0255 BSC		
L1	1.00 REF			0.0393 REF		
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
θ	0°	–	8°	0°	–	8°
y		0.05			0.00196	

33 Revision History

version	Date	Section	Description
A1.0	2011.05.25		Initial version

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.