Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	8051
Core Size	8-Bit
Speed	24MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LED, LVD, POR, PWM, WDT
Number of I/O	17
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.4V ~ 5.5V
Data Converters	A/D 7x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-TSSOP (0.173", 4.40mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/nuvoton-technology-corporation-america/n79e845awg

Table 5–1 Pin Description

Pin number		Sym-	Alternate Function				Type	Description
SOP16	SOP20 TSSOP20		1	2	3			
12	15	V_{DD}					P	**POWER SUPPLY:** Supply voltage V_{DD} for operation.
5	5	V_{SS}					P	**GROUND:** Ground potential
4	4	/RST					I (ST)	**RESET:** Chip reset pin that is low active. Because reset pin has internal pull-up resistor (about 200 KΩ at V_{DD} = 5V), this pin can-not be floating. Reset pin should be connected to 100Ω pull-up resistor and 10uF pull-low capacitor.
1	1	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 is PWM output channel. T1: Timer 1 External Input SPICLK: SPI-1 clock pin
16	20	P0.1	PWM0	ADC0	KB1		I/O	
15	19	P0.2	BRAKE	ADC1	KB2		I/O	
14	18	P0.3		ADC2	KB3		I/O	
13	17	P0.4		ADC3	KB4		I/O	
-	16	P0.5		ADC4	KB5		I/O	
-	14	P0.6		ADC5	KB6		I/O	
-	13	P0.7	T1	ADC6	KB7	IC1	I/O	
11	12	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 I^2C 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
10	11	P1.1	RXD				I/O	
9	10	P1.2	T0		SCL	IC0	D	
8	9	P1.3	/INT0		SDA		D	

6.1 APROM Flash memory

The N79E845/844/8432 has **16K/8K/4K** Program Memory. All instructions are fetched for execution from this memory area. The MOVC instruction can also read this memory region.

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

6.2 LDROM Flash Memory

Each device of the N79E845/844/8432 is equipped with 2 Kbytes LDROM stored the ISP application program. User may develop the ISP function in LDROM for updating application program or Data Flash. Similarly, APROM can also reprogram LDROM and Data Flash. The start address of LDROM is at 0000H corresponding to the physical address of the Flash memory. However, when CPU runs in LDROM, CPU automatically re-vectors the LDROM start address to 0000H, therefore user program regards the LDROM as an independent program memory, meanwhile, with all interrupt vectors that CPU provides.

6.3 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 LDROM can also erase/program CONFIG-bits.

6.4 On-chip Non-volatile Data Flash

The N79E845/844/8432 additionally has non-volatile Data Flash, which 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, SP mode can erase, written, or read the Data Flash only . Of course, hardware with parallel Programmer/Writer or ICP programmer can also access the Data Flash.

The Data Flash size is software adjustable in 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 is 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 protection while a write to SHBDA is required. The Data Flash size will be 15.75k bytes and there will is 256 bytes APROM.

The Data Flash size is fixed as 4 Kbytes from address 3000H through 3FFFH in N79E844/8432. SHBDA affects nothing.

8 General 80C51 System Control

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.

P3 – Port 3 (Bit-addressable)

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

Address: B0H Reset value: 0000 0011B

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

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	-	P1M1.4	P1M1.3	P1M1.2	P1M1.1	P1M1.0
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	-	P1M2.4	P1M2.3	P1M2.2	P1M2.1	P1M2.0
R/W	R/W	-	R/W	R/W	R/W	R/W	R/W

Address: B4H Reset value: 0000 0000B

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	Enable Schmitt trigger inputs on Port 3.
6	-	Reserved

10 Timers/Counters

The N79E845/844/8432 has three 16-bit programmable timers/counters.

10.1 Timers/Counters 0 and 1

Timer/Counter 0 and 1 in the N79E845/844/8432 is two 16-bit Timers/Counters. Each of them has two 8-bit registers that form the 16 bit counting register. For Timer/Counter 0 they are TH0, the upper 8-bit register, and TL0, the lower 8-bit register. Similar 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 timers/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 clock system or 1/4 of the clock system. 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 N79E845/844/8432 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	-	-	-

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
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 0 overflow rate.

10.1.1 Mode 0 (13-bit Timer)

In Mode 0, the timers/counters act as a 8-bit counter with a 5-bit, divide by 32 pre-scale. In this mode we have 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 rollover to 0000h. The timer overflow flag TFx of the relevant timer is set and if enabled an interrupts will occur.

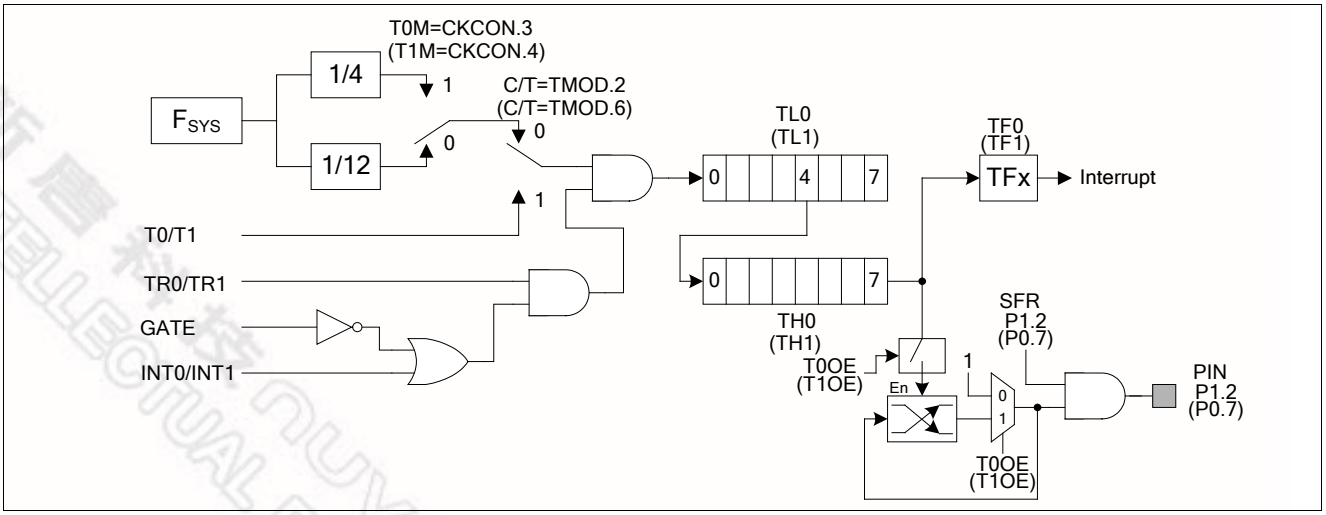

Figure 10–1 Timers/Counters 0 and 1 in Mode 0

12 Serial Port (UART)

The N79E845/844/8432 includes 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 Selection**
6	SM1	SMOD0 (PCON.6) = 0: See Table 12–1 Serial Port Mode Description for 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.
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.

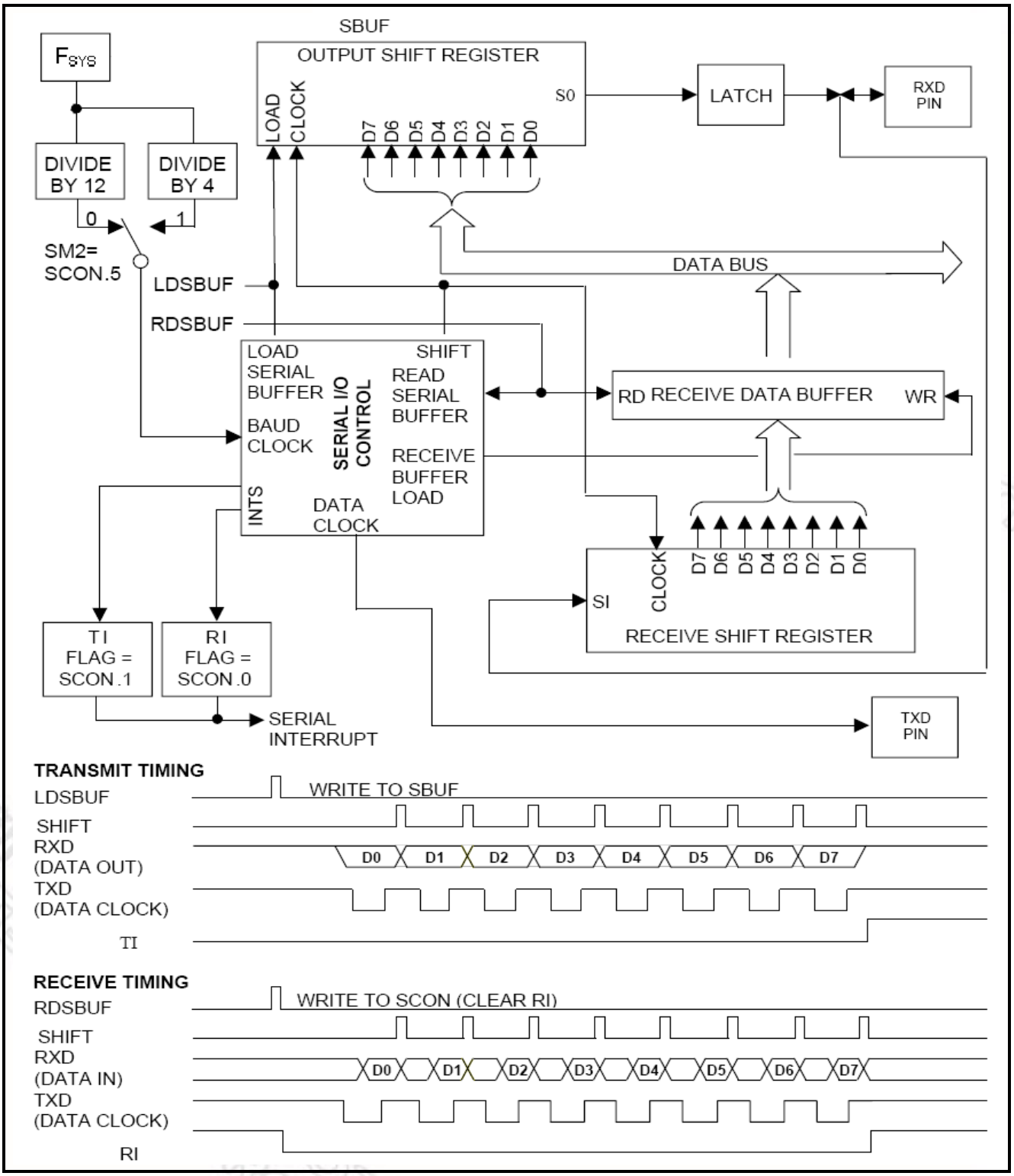

Figure 12–1 Serial Port Mode 0 Function Block

```
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 care”. In most cases, interpreting the “don’t care” 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 care” 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.

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 be 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 should 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 should read SPDR out and the first SPIF should 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 Format 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. Communicating in different data formats with one another will result in undetermined results.

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 entering Idle or Power-down mode. The result of a completed conversion (ADCI = logic 1) remains unaffected when entering Idle mode.

When ADCCON0.5 (ADCEX) is set by external pin to start ADC conversion, after the N79E845/844/8432 entry Idle mode, P1.4 can start ADC conversion at least ONE machine-cycle.

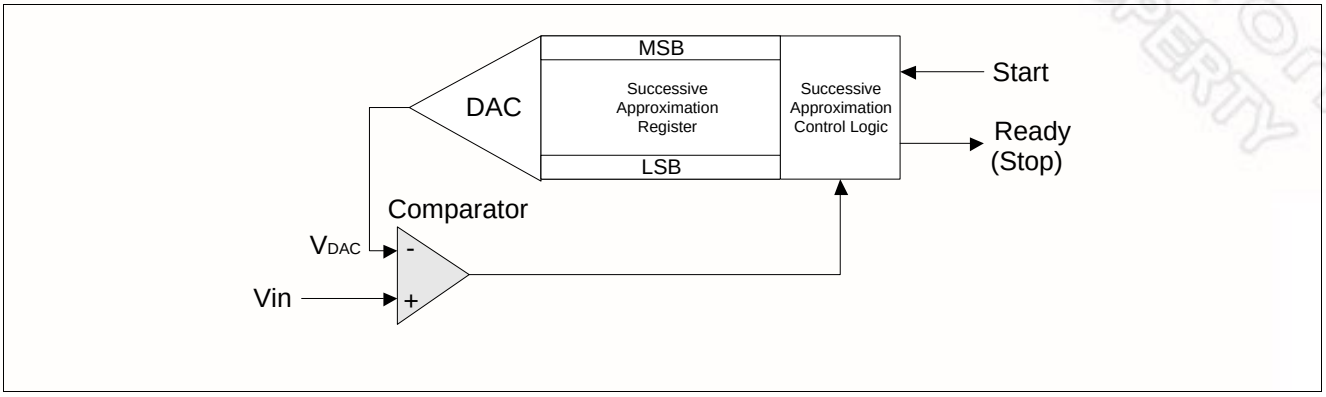

Figure 15-1 Successive Approximation ADC

The ADC circuit has its own supply pins (AV_{DD} and AVSS) and one pins (Vref+) connected to each end of the DAC's resistance-ladder that the AV_{DD} and Vref+ are connected to V_{DD} 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 AV_{SS}, 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 AV_{SS} 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 AV_{SS} may be between AV_{DD} + 0.2V and AVSS – 0.2 V. Avref+ should be positive with respect to AV_{SS}, and the input voltage (Vin) should be between Avref+ and AV_{SS}.

The result can always be calculated according to the following formula:

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

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.

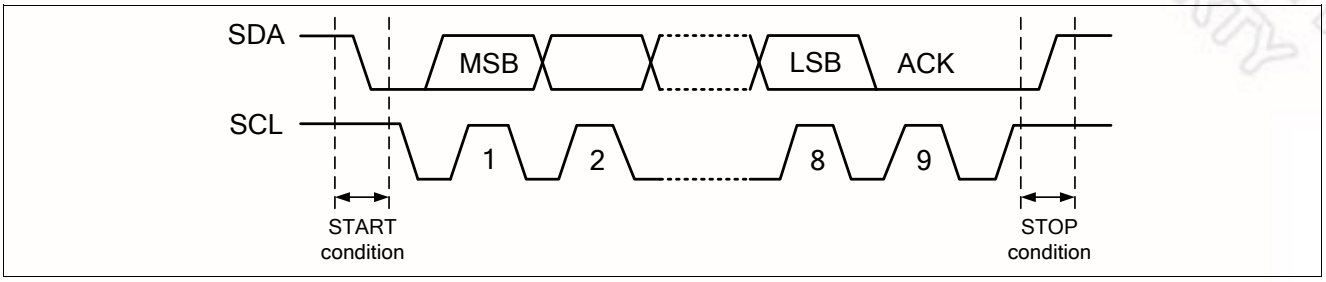

Figure 16–2 I²C Bus Protocol

16.2.1 START and STOP Conditions

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.

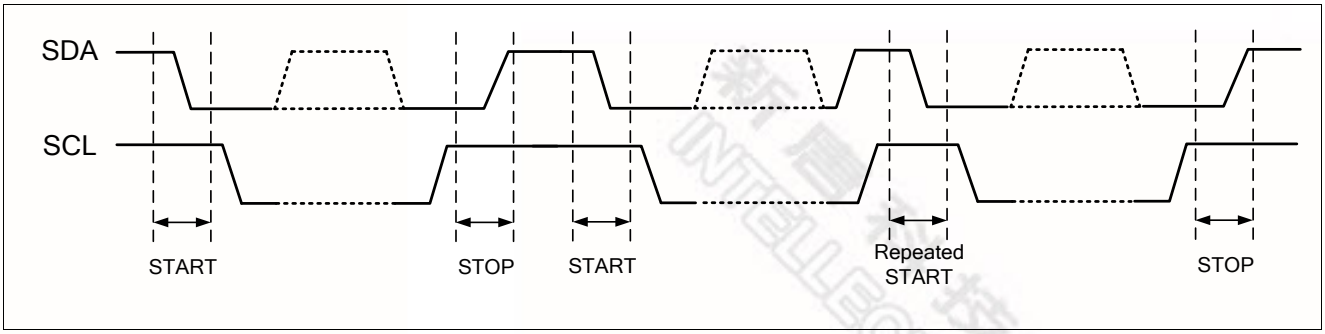

Figure 16–3 START, Repeated START, and STOP Conditions

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.

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 I2STA 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.

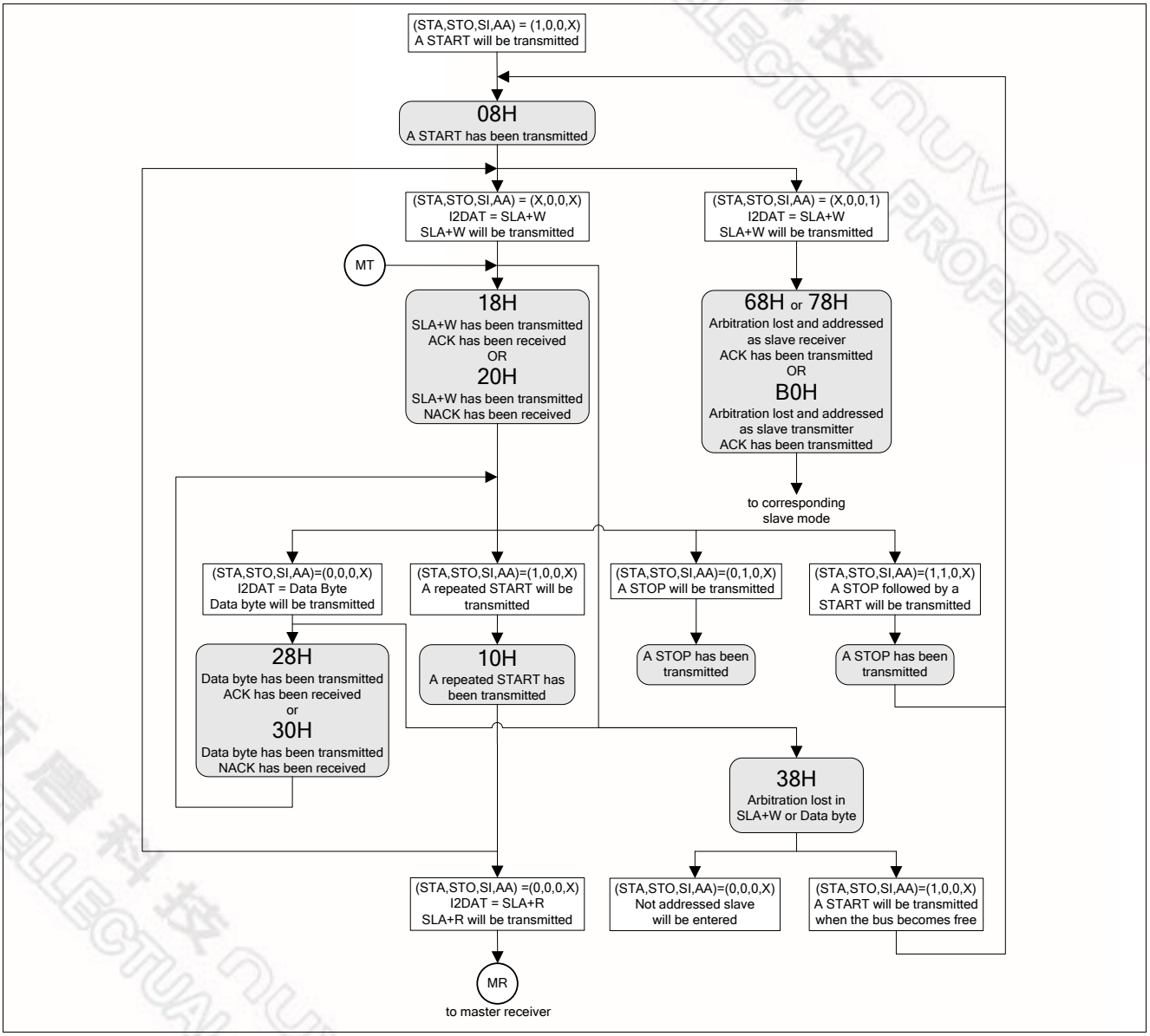

Figure 16–7 Flow and Status of Master Transmitter Mode

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

should be cleared. After that, STO is cleared by hardware and release the I²C 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 I²C 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 I²C 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 I²C 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 I²C Interrupt Service Routine

The following software example in C language for KEIL C51 compiler shows the typical structure of the I²C interrupt service routine including the 26 state service routines and may be used as a base for user applications. User 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 (I2STA)
      {
            //================================================
            //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
```

```
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;
```

Bit	Name	Description
		0 = Enable ISP function. 1 = Disable ISP function. To enable ISP function will start the internal 22.1184 MHz RC oscillator for timing control. To clear ISPEN should always be the last instruction after ISP operation 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: 0011 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.

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

(4) **ISP works under V_{DD} = 3.0V ~ 5.5V.**

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

Note: 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
```

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. WDTEN is initialized by inverted CWDTEN (CONFIG3, bit-7) at any other resets.
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 Selection** 1 = Select 22.1184 MHz as the clock system if internal RC oscillator mode is used. It bypasses the divided-by-2 path of internal oscillator to select 22.1184MHz output as the clock system source. 0 = Select 11.0592 MHz as the clock system if internal RC oscillator mode is used. The internal RC divided-by-2 path is selected. The internal oscillator is equivalent to 11.0592 MHz output used as the clock system.
2	-	**Reserved**
1	FOSC1	**Oscillator Select Bit**
0	FOSC0	**Chip Clock Source Selection (See the Following Table)**

(FOSC1, FOSC0)	Chip Clock Source
(1, 1)	Internal RC oscillator
(1, 0)	Reserved
(0, 0) (0, 1)	External crystal, 4 MHz ~ 24 MHz

Table 26–1 Instruction Set for the N79E845/844/8432

Instruction		OPCODE	Bytes	Clock Cycles	N79E845/844/8432 vs. 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
MOV	C, bit	A2	2	8	1.5