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	16MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, I²S, POR, PWM, WDT
Number of I/O	18
Program Memory Size	18KB (18K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.4V ~ 5.5V
Data Converters	A/D 8x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	20-WFQFN Exposed Pad
Supplier Device Package	20-QFN (3x3)
Purchase URL	https://www.e-xfl.com/product-detail/nuvoton-technology-corporation-america/n76e003aq20

Bit	Name	Description
6	SMOD0	**Serial port 0 framing error flag access enable** 0 = SCON.7 accesses to SM0 bit. 1 = SCON.7 accesses to FE bit.
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.
3	GF1	**General purpose flag 1** The general purpose flag that can be set or cleared by user via software.
2	GF0	**General purpose flag 0** The general purpose flag that can be set or cleared by user via software.
1	PD	**Power-down mode** Setting this bit puts CPU 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, CPU will enter Power-down mode. Then it does not go to Idle mode after exiting Power-down.
0	IDL	**Idle mode** Setting this bit puts CPU into Idle mode. Under this mode, the CPU clock stops and Program Counter (PC) suspends but all peripherals keep activated. 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.

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 (level) R/W (edge)	R/W	R (level) R/W (edge)	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. This bit can be set or cleared by software.
6	TR1	**Timer 1 run control** 0 = Timer 1 Disabled. Clearing this bit will halt Timer 1 and the current count will be preserved in TH1 and TL1. 1 = Timer 1 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. This bit can be set or cleared by software.

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 counting register of Timer 1.

CKCON – Clock Control

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

Address: 8EH Reset value: 0000 0000b

Bit	Name	Description
6	PWMCKS	**PWM clock source select** 0 = The clock source of PWM is the system clock F_{SYS}. 1 = The clock source of PWM is the overflow of Timer 1.
4	T1M	**Timer 1 clock mode select** 0 = The clock source of Timer 1 is the system clock divided by 12. It maintains standard 8051 compatibility. 1 = The clock source of Timer 1 is direct the system clock.
3	T0M	**Timer 0 clock mode select** 0 = The clock source of Timer 0 is the system clock divided by 12. It maintains standard 8051 compatibility. 1 = The clock source of Timer 0 is direct the system clock.
1	CLOEN	**System clock output enable** 0 = System clock output Disabled. 1 = System clock output Enabled from CLO pin (P1.1).

WKCON – Self Wake-up Timer Control

7	6	5	4	3	2	1	0
-	-	-	WKTF	WKTR	WKPS[2:0]		
-	-	-	R/W	R/W	R/W		

Address: 8FH Reset value: 0000 0000b

Bit	Name	Description
4	WKTF	**WKT overflow flag** This bit is set when WKT overflows. If the WKT interrupt and the global interrupt are enabled, setting this bit will make CPU execute WKT interrupt service routine. This bit is not automatically cleared via hardware and should be cleared via software.
3	WKTR	**WKT run control** 0 = WKT is halted. 1 = WKT starts running. Note that the reload register RWK can only be written when WKT is halted (WKTR bit is 0). If WKT is written while WKTR is 1, result is unpredictable.

I2CON – I²C Control (Bit-addressable)

7	6	5	4	3	2	1	0
-	I2CEN	STA	STO	SI	AA	-	I2CPX
-	R/W	R/W	R/W	R/W	R/W	-	R/W

Address: C0H Reset value: 0000 0000b

Bit	Name	Description
6	I2CEN	**I²C bus enable** 0 = I²C bus Disabled. 1 = I²C bus Enabled. Before enabling the I²C, SCL and SDA port latches should be set to logic 1.
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. 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	**I²C interrupt flag** SI flag is set by hardware when one of 26 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, user should take care of it by preparing suitable setting of registers before SI is software cleared.

Bit	Name	Description
4	CLRPWM	**Clear PWM counter** Setting this bit clears the value of PWM 16-bit counter for resetting to 0000H. After the counter value is cleared, CLRPWM will be automatically cleared via hardware. The meaning of writing and reading CLRPWM bit is different. Writing: 0 = No effect. 1 = Clearing PWM 16-bit counter. Reading: 0 = PWM 16-bit counter is completely cleared. 1 = PWM 16-bit counter is not yet cleared.

PWMPL – PWM Period Low Byte

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

Address: D9H reset value: 0000 0000b

Bit	Name	Description
7:0	PWMP[7:0]	**PWM period low byte** This byte with PWMPH controls the period of the PWM generator signal.

PWM0L – PWM0 Duty Low Byte

7	6	5	4	3	2	1	0
PWM0[7:0]							
R/W							

Address: DAH reset value: 0000 0000b

Bit	Name	Description
7:0	PWM0[7:0]	**PWM0 duty low byte** This byte with PWM0H controls the duty of the output signal PG0 from PWM generator.

PWM1L – PWM/1 Duty Low Byte

7	6	5	4	3	2	1	0
PWM1[7:0]							
R/W							

Address: DBH reset value: 0000 0000b

Bit	Name	Description
7:0	PWM1[7:0]	**PWM1 duty low byte** This byte with PWM1H controls the duty of the output signal PG1 from PWM generator.

Bit	Name	Description
3:0	ADCHS[3:0]	**A/D converting channel select** This filed selects the activating analog input source of ADC. If ADCEN is 0, all inputs are disconnected. 0000 = AIN0. 0001 = AIN1. 0010 = AIN2. 0011 = AIN3. 0100 = AIN4. 0101 = AIN5. 0110 = AIN6. 0111 = AIN7 1000 = Internal band-gap voltage. Others = Reserved.

PICON – Pin Interrupt Control

7	6	5	4	3	2	1	0
PIT67	PIT45	PIT3	PIT2	PIT1	PIT0	PIPS[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Address: E9H Reset value: 0000 0000b

Bit	Name	Description
7	PIT67	**Pin interrupt channel 6 and 7 type select** This bit selects which type that pin interrupt channel 6 and 7 is triggered. 0 = Level triggered. 1 = Edge triggered.
6	PIT45	**Pin interrupt channel 4 and 5 type select** This bit selects which type that pin interrupt channel 4 and 5 is triggered. 0 = Level triggered. 1 = Edge triggered.
5	PIT3	**Pin interrupt channel 3 type select** This bit selects which type that pin interrupt channel 3 is triggered. 0 = Level triggered. 1 = Edge triggered.
4	PIT2	**Pin interrupt channel 2 type select** This bit selects which type that pin interrupt channel 2 is triggered. 0 = Level triggered. 1 = Edge triggered.
3	PIT1	**Pin interrupt channel 1 type select** This bit selects which type that pin interrupt channel 1 is triggered. 0 = Level triggered. 1 = Edge triggered.
2	PIT0	**Pin interrupt channel 0 type select** This bit selects which type that pin interrupt channel 0 is triggered. 0 = Level triggered. 1 = Edge triggered.
1:0	PIPS[:0]	**Pin interrupt port select** This field selects which port is active as the 8-channel of pin interrupt. 00 = Port 0. 01 = Port 1. 10 = Port 2. 11 = Port 3.

Bit	Name	Description
2	CPHA	**SPI clock phase select** CPHA bit determines the data sampling edge of the SPI clock. See Figure 14.4. 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.

SPCR2 – Serial Peripheral Control Register 2

7	6	5	4	3	2	1	0
-	-	-	-	-	-	SPIS1	SPIS0
-	-	-	-	-	-	R/W	R/W

Address: F3H, page 1 Reset value: 0000 0000b

Bit	Name	Description
7:2	-	**Reserved**
1:0	SPIS[1:0]	**SPI Interval time selection between adjacent bytes** SPIS[1:0] and CPHA select eight grades of SPI interval time selection between adjacent bytes. As below table: CPHA SPIS1 SPIS0 SPI clock 0 0 0 0.5 0 0 1 1.0 0 1 0 1.5 0 1 1 2.0 1 0 0 1.0 1 0 1 1.5 1 1 0 2.0 1 1 1 2.5 SPIS[1:0] are valid only under Master mode (MSTR = 1).

SPSR – Serial Peripheral Status Register

7	6	5	4	3	2	1	0
SPIF	WCOL	SPIOVF	MODF	DISMODF	TXBUF	-	-
R/W	R/W	R/W	R/W	R/W	R	-	-

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 .0) and EA are enabled, an SPI interrupt will be required. This bit should 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 should 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 should be cleared via software.

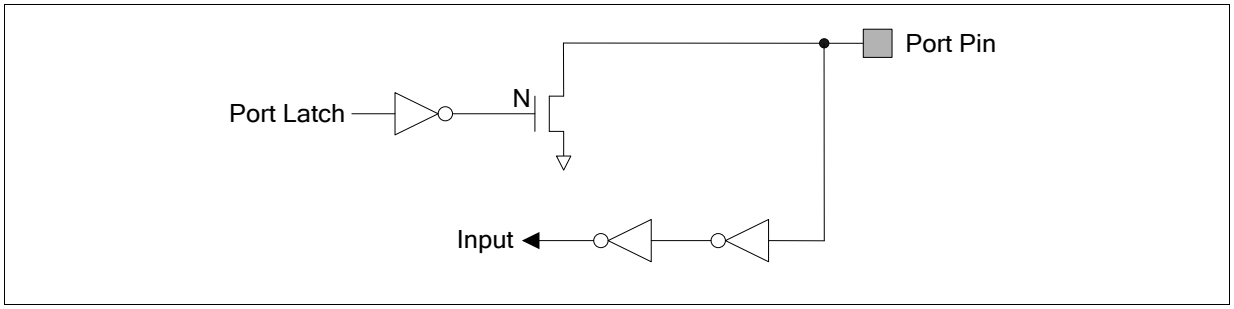


Figure 7.4. Open-Drain Mode Structure

7.5 Read-Modify-Write Instructions

Instructions that read a byte from SFR or internal RAM, modify it, and rewrite it back, are called “Read-Modify-Write” instructions. When the destination is an I/O port or a port bit, these instructions read the internal output latch rather than the external pin state. This kind of instructions read the port SFR value, modify it and write back to the port SFR. All “Read-Modify-Write” instructions are listed as follows.

Instruction		Description
ANL		Logical AND. (ANL direct, A and ANL direct, #data)
ORL		Logical OR. (ORL direct, A and ORL direct, #data)
XRL		Logical exclusive OR. (XRL direct, A and XRL direct, #data)
JBC		Jump if bit = 1 and clear it. (JBC bit, rel)
CPL		Complement bit. (CPL bit)
INC		Increment. (INC direct)
DEC		Decrement. (DEC direct)
DJNZ		Decrement and jump if not zero. (DJNZ direct, rel)
MOV	bit, C	Move carry to bit. (MOV bit, C)
CLR	bit	Clear bit. (CLR bit)
SETB	bit	Set bit. (SETB bit)

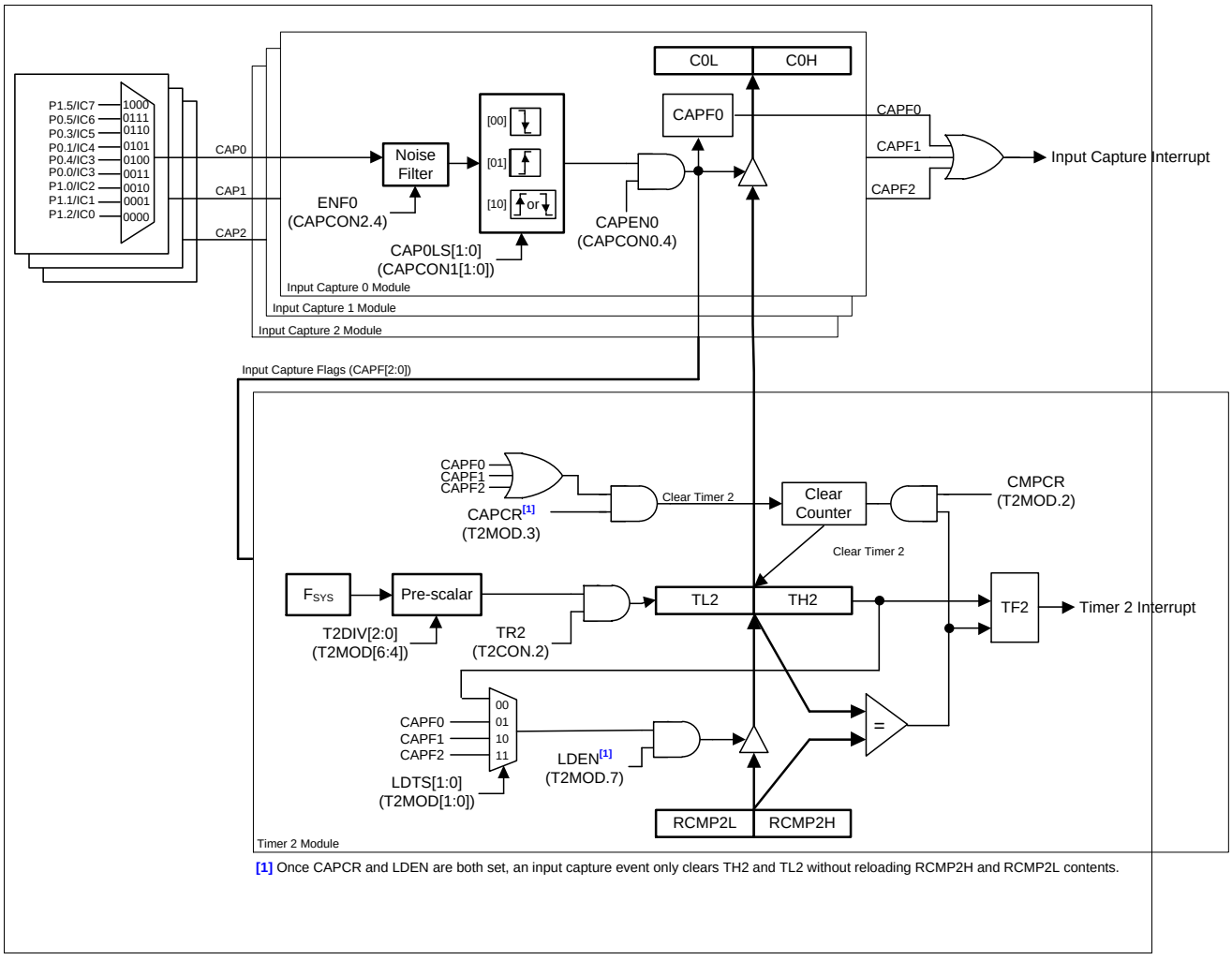
The last three seem not obviously “Read-Modify-Write” instructions but actually they are. They read the entire port latch value, modify the changed bit, and then write the new value back to the port latch.

7.6 Control Registers of I/O Ports

The N76E003 has a lot of I/O control registers to provide flexibility in all kinds of applications. The SFRs related with I/O ports can be categorized into four groups: input and output control, output mode control, input type and sink current control, and output slew rate control. All of SFRs are listed as follows.

9. TIMER 2 AND INPUT CAPTURE

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 RCMP2H and RCMP2L, Timer 2 can operate under compare mode and auto-reload mode selected by CM/$\overline{RL2}$ (T2CON.0). An 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 1, and disabled when TR2 is 0. The following registers are related to Timer 2 function.



[1] Once CAPCR and LDEN are both set, an input capture event only clears TH2 and TL2 without reloading RCMP2H and RCMP2L contents.

Figure 9.1. Timer 2 Block Diagram

TH2 – Timer 2 High Byte

7	6	5	4	3	2	1	0
TH2[7:0]							
R/W							

Address: CDH, Page:0 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 counting register of Timer 2.

Note that the TH2 and TL2 are accessed separately. It is strongly recommended that user stops Timer 2 temporally by clearing TR2 bit before reading from or writing to TH2 and TL2. The free-running reading or writing may cause unpredictable result.

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 for 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.

14.2 Operating Modes

14.2.1 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. User can clear SPIF and read data out of SPDR.

14.2.2 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 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 forced 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.

15. INTER-INTEGRATED CIRCUIT (I^2C)

The Inter-Integrated Circuit (I^2C) bus serves as an serial interface between the microcontrollers and the I^2C devices such as EEPROM, LCD module, temperature sensor, 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, master receiver, slave receiver, and slave transmitter. The I^2C interface only supports 7-bit addressing mode. A special mode General Call is also available. The I^2C can meet both standard (up to 100kbps) and fast (up to 400k bps) speeds.

15.1 Functional Description

For a bi-directional transfer operation, the SDA and SCL pins should be 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 N76E003, user should set output latches of SCL and SDA. As logic 1 before enabling the I^2C function by setting I2CEN (I2CON.6).

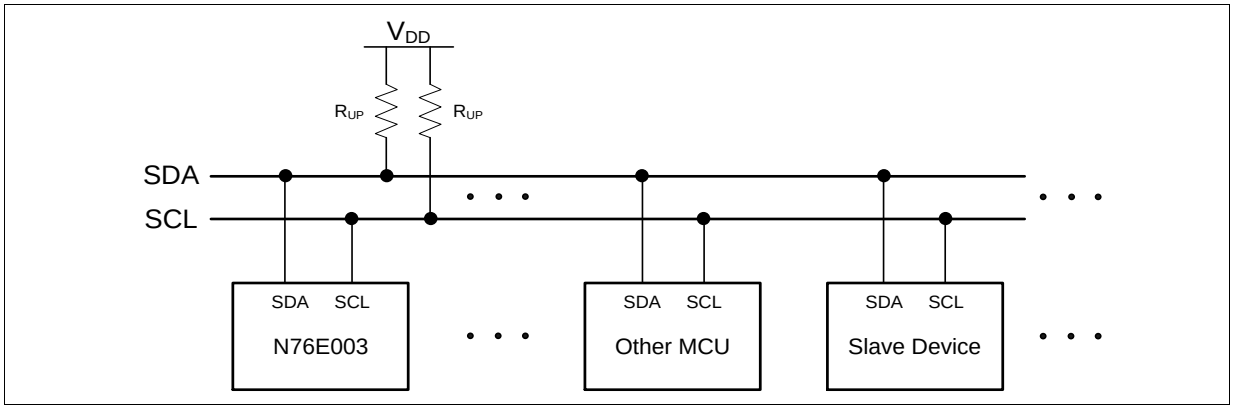


Figure 15.1. I^2C Bus Interconnection

The I^2C 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

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[7:1]							GC
R/W							R/W

Address: C1H Reset value: 0000 0000b

Bit	Name	Description
7:1	I2ADDR[7:1]	**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 user. The master should address 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. Note that I2ADDR[7:1] should not remain its default value of all 0, because address 0x00 is reserved for General Call.
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.

```
            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 or
General Call
                  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,
```

PMD – PWM Mask Data

7	6	5	4	3	2	1	0
-	-	PMD5	PMD4	PMD3	PMD2	PMD1	PMD0
-	-	R/W	R/W	R/W	R/W	R/W	R/W

Address: FCH Reset value: 0000 0000b

Bit	Name	Description
n	PMDn	**PWMn mask data** The PWMn signal outputs mask data once its corresponding PMENn is set. 0 = PWMn signal is masked by 0. 1 = PWMn signal is masked by 1.

17.1.5 Fault Brake

The Fault Brake function is usually implemented in conjunction with an enhanced PWM circuit. It rules as a fault detection input to protect the motor system from damage. Fault Brake pin input (FB) is valid when FBINEN (PWMCON1.3) is set. When Fault Brake is asserted PWM signals will be individually overwritten by FBD corresponding bits. PWMRUN (PWMCON0.7) will also be automatically cleared by hardware to stop PWM generating. The PWM 16-bit counter will also be reset as 0000H. A indicating flag FBF will be set by hardware to assert a Fault Brake interrupt if enabled. FBD data output remains even after the FBF is cleared by software. User should resume the PWM output only by setting PWMRUN again. Meanwhile the Fault Brake state will be released and PWM waveform outputs on pins as usual. Fault Brake input has a polarity selection by FBINLS (FBD.6) bit. Note that the Fault Brake signal feed in FB pin should be longer than eight-system-clock time for FB pin input has a permanent $8/F_{SYS}$ de-bouncing, which avoids fake Fault Brake event by input noise. The other path to trigger a Fault Brake event is the ADC compare event. It asserts the Fault Brake behavior just the same as FB pin input. See Sector 18.1.3 “ADC Conversion Result Comparator” on page 198.

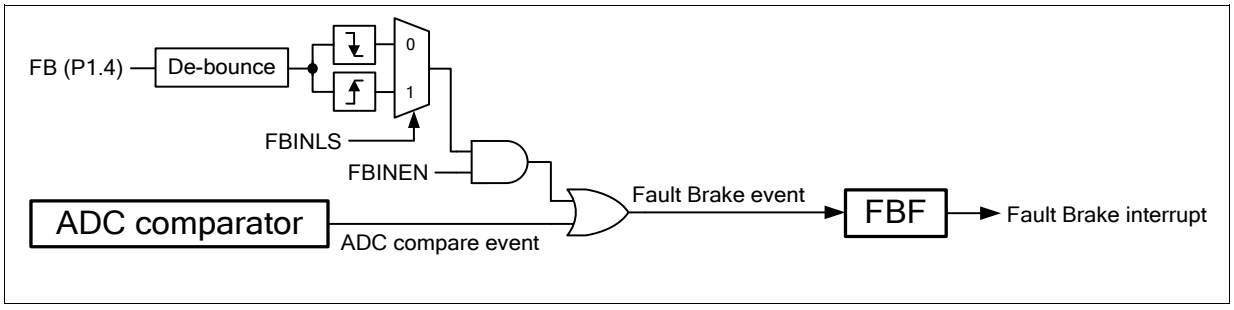


Figure 17.6. Fault Brake Function Block Diagram

IP – Interrupt Priority (Bit-addressable)[1]

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

Address: B8H Reset value: 0000 0000b

Bit	Name	Description
6	PADC	**ADC interrupt priority low bit**
5	PBOD	**Brown-out detection interrupt priority low bit**
4	PS	**Serial port 0 interrupt priority low bit**
3	PT1	**Timer 1 interrupt priority low bit**
2	PX1	**External interrupt 1 priority low bit**
1	PT0	**Timer 0 interrupt priority low bit**
0	PX0	**External interrupt 0 priority low bit**

[1] IP is used in combination with the IPH to determine the priority of each interrupt source. See Table 20-2. Interrupt Priority Level Setting for correct interrupt priority configuration.

IPH – Interrupt Priority High[2]

7	6	5	4	3	2	1	0
-	PADCH	PBODH	PSH	PT1H	PX1H	PT0H	PX0H
-	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: B7H, Page0 Reset value: 0000 0000b

Bit	Name	Description
6	PADC	**ADC interrupt priority high bit**
5	PBOD	**Brown-out detection interrupt priority high bit**
4	PSH	**Serial port 0 interrupt priority high bit**
3	PT1H	**Timer 1 interrupt priority high bit**
2	PX1H	**External interrupt 1 priority high bit**
1	PT0H	**Timer 0 interrupt priority high bit**
0	PX0H	**External interrupt 0 priority high bit**

[2] IPH is used in combination with the IP respectively to determine the priority of each interrupt source. See Table 20-2. Interrupt Priority Level Setting for correct interrupt priority configuration.

EIP – Extensive Interrupt Priority[3]

7	6	5	4	3	2	1	0
PT2	PSPI	PFB	PWDT	PPWM	PCAP	PPI	PI2C
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: EFH Reset value: 0000 0000b

Bit	Name	Description
7	PT2	**Timer 2 interrupt priority low bit**
6	PSPI	**SPI interrupt priority low bit**

IAPTRG – IAP Trigger (TA protected)

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

Address: A4H Reset value: 0000 0000b

Bit	Name	Description

0	IAPGO	**IAP go**
		IAP begins by setting this bit as logic 1. After this instruction, the CPU holds the Program Counter (PC) and the IAP hardware automation takes over to control the progress. After IAP action completed, the Program Counter continues to run the following instruction. The IAPGO bit will be automatically cleared and always read as logic 0.
		Before triggering an IAP action, interrupts (if enabled) should be temporary disabled for hardware limitation. The program process should follows below.
		`CLR EA`
		`MOV TA,#0AAH`
		`MOV TA,#55H`
		`ORL IAPTRG,#01H`
		`(SETB EA)`

21.1 IAP Commands

The N76E003 provides a wide range of applications to perform IAP to APROM, LDROM, or CONFIG bytes. The IAP action mode and the destination of the flash block are defined by IAP control register IAPCN.

Table 21-1. IAP Modes and Command Codes

IAP Mode	IAPCN				IAPA[15:0] {IAPAH, IAPAL}	IAPFD[7:0]
	IAPB[1:0]	FOEN	FCEN	FCTRL[3:0]		
Company ID read	XX[1]	0	0	1011	X	DAH
Device ID read	XX	0	0	1100	Low-byte DID: 0000H High-byte DID: 0001H	Low-byte DID: 50H High-byte DID: 36H
96-bit Unique Code read	XX	0	0	0100	0000H to 000BH	Data out
APROM page-erase	00	1	0	0010	Address in[2]	FFH
LDROM page-erase	01	1	0	0010	Address in[2]	FFH
APROM byte-program	00	1	0	0001	Address in	Data in
LDROM byte-program	01	1	0	0001	Address in	Data in
APROM byte-read	00	0	0	0000	Address in	Data out
LDROM byte-read	01	0	0	0000	Address in	Data out
All CONFIG bytes erase	11	1	0	0010	0000H	FFH

```
        MOV     TA,#55h
        ORL     CHPCON,#80h                 ;software reset and reboot from APROM

        SJMP    $

;*****************************************************************
;             IAP Subroutine
;*****************************************************************
Enable_IAP:
        MOV     TA,#0Aah                    ;CHPCON is TA protected
        MOV     TA,#55h
        ORL     CHPCON,#00000001b           ;IAPEN = 1, enable IAP mode
        RET

Disable_IAP:
        MOV     TA,#0Aah
        MOV     TA,#55h
        ANL     CHPCON,#11111110b           ;IAPEN = 0, disable IAP mode
        RET

Enable_AP_Update:
        MOV     TA,#0Aah                    ;IAPUEN is TA protected
        MOV     TA,#55h
        ORL     IAPUEN,#00000001b           ;APUEN = 1, enable APROM update
        RET

Disable_AP_Update:
        MOV     TA,#0Aah
        MOV     TA,#55h
        ANL     IAPUEN,#11111110b           ;APUEN = 0, disable APROM update
        RET

Enable_CONFIG_Update:
        MOV     TA,#0Aah
        MOV     TA,#55h
        ORL     IAPUEN,#00000100b           ;CFUEN = 1, enable CONFIG update
        RET

Disable_CONFIG_Update:
        MOV     TA,#0Aah
        MOV     TA,#55h
        ANL     IAPUEN,#11111011b           ;CFUEN = 0, disable CONFIG update
        RET

Trigger_IAP:
        MOV     TA,#0Aah                    ;IAPTRG is TA protected
        MOV     TA,#55h
        ORL     IAPTRG,#00000001b           ;write '1' to IAPGO to trigger IAP process
        RET

;*****************************************************************
;             IAP APROM Function
;*****************************************************************
Erase_AP:
        MOV     IAPCN,#PAGE_ERASE_AP
        MOV     IAPFD,#0FFh
        MOV     R0,#00h
Erase_AP_Loop:
        MOV     IAPAH,R0
        MOV     IAPAL,#00h
        CALL    Trigger_IAP
        MOV     IAPAL,#80h
```

BODCON0 – Brown-out Detection Control 0 (TA protected)

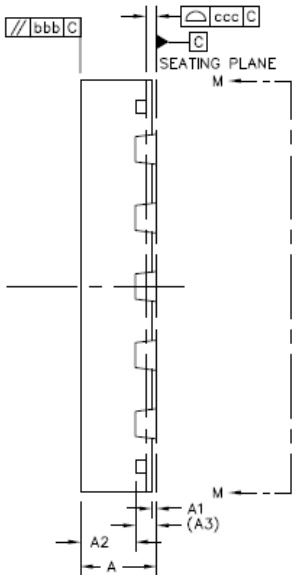
7	6	5	4	3	2	1	0
BODEN[1]		BOV[1:0][1]		BOF[2]	BORST[1]	BORF	BOS
R/W		R/W		R/W	R/W	R/W	R

Address: A3H Reset value: see Table 6-2. SFR Definitions and Reset Values

Bit	Name	Description
7	BODEN	**Brown-out detection enable** 0 = Brown-out detection circuit off. 1 = Brown-out detection circuit on. Note that BOD output is not available until 2~3 LIRC clocks after enabling.
6:4	BOV[1:0]	**Brown-out voltage select** 11 = V_{BOD} is 2.2V. 10 = V_{BOD} is 2.7V. 01 = V_{BOD} is 3.7V. 00 = V_{BOD} is 4.4V.
3	BOF	**Brown-out interrupt flag** This flag will be set as logic 1 via hardware after a V_{DD} dropping below or rising above V_{BOD} event occurs. If both EBOD (EIE.2) and EA (IE.7) are set, a brown-out interrupt requirement will be generated. This bit should be cleared via software.
2	BORST	**Brown-out reset enable** This bit decides whether a brown-out reset is caused by a power drop below V_{BOD}. 0 = Brown-out reset when V_{DD} drops below V_{BOD} Disabled. 1 = Brown-out reset when V_{DD} drops below V_{BOD} Enabled.
1	BORF	**Brown-out reset flag** When the MCU is reset by brown-out event, this bit will be set via hardware. This flag is recommended to be cleared via software.
0	BOS	**Brown-out status** This bit indicates the V_{DD} voltage level comparing with V_{BOD} while BOD circuit is enabled. It keeps 0 if BOD is not enabled. 0 = V_{DD} voltage level is higher than V_{BOD} or BOD is disabled. 1 = V_{DD} voltage level is lower than V_{BOD}. Note that this bit is read-only.

[1] BODEN, BOV[1:0], and BORST are initialized by being directly loaded from CONFIG2 bit 7, [6:4], and 2 after all resets.

[2] BOF reset value depends on different setting of CONFIG2 and V_{DD} voltage level. Please check Table 24-1.



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.5	0.55	0.6
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.40	---
L/F THICKNESS		A3		0.152 REF	
LEAD WIDTH		b	0.17	0.22	0.27
BODY SIZE	X	D		3 BSC	
	Y	E		3 BSC	
LEAD PITCH		e		0.5 BSC	
EP SIZE	X	J	1.4	1.50	1.6
	Y	K	1.4	1.50	1.6
LEAD LENGTH		L	0.25	0.30	0.35
PACKAGE EDGE TOLERANCE		aaa		0.1	
MOLD FLATNESS		bbb		0.1	
COPLANARITY		ccc		0.08	
LEAD OFFSET		ddd		0.1	
EXPOSED PAD OFFSET		eee		0.1	

Figure 32.4. QFN-20 Package Dimension of N76E003BQ20