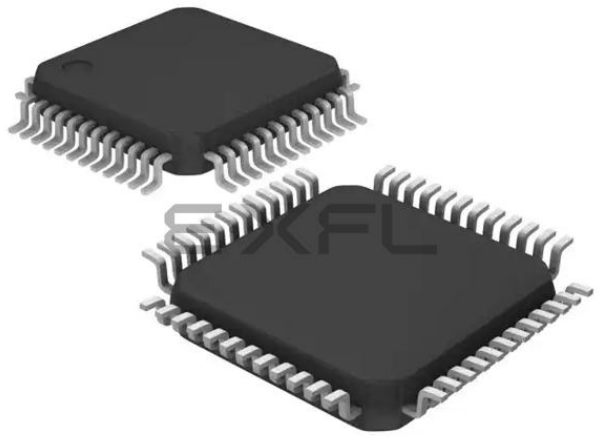

Welcome to E-XFL.COM

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, UART/USART
Peripherals	LCD, POR, PWM
Number of I/O	46
Program Memory Size	18KB (18K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.4V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	48-LQFP
Supplier Device Package	48-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/nuvoton-technology-corporation-america/n76e616al48

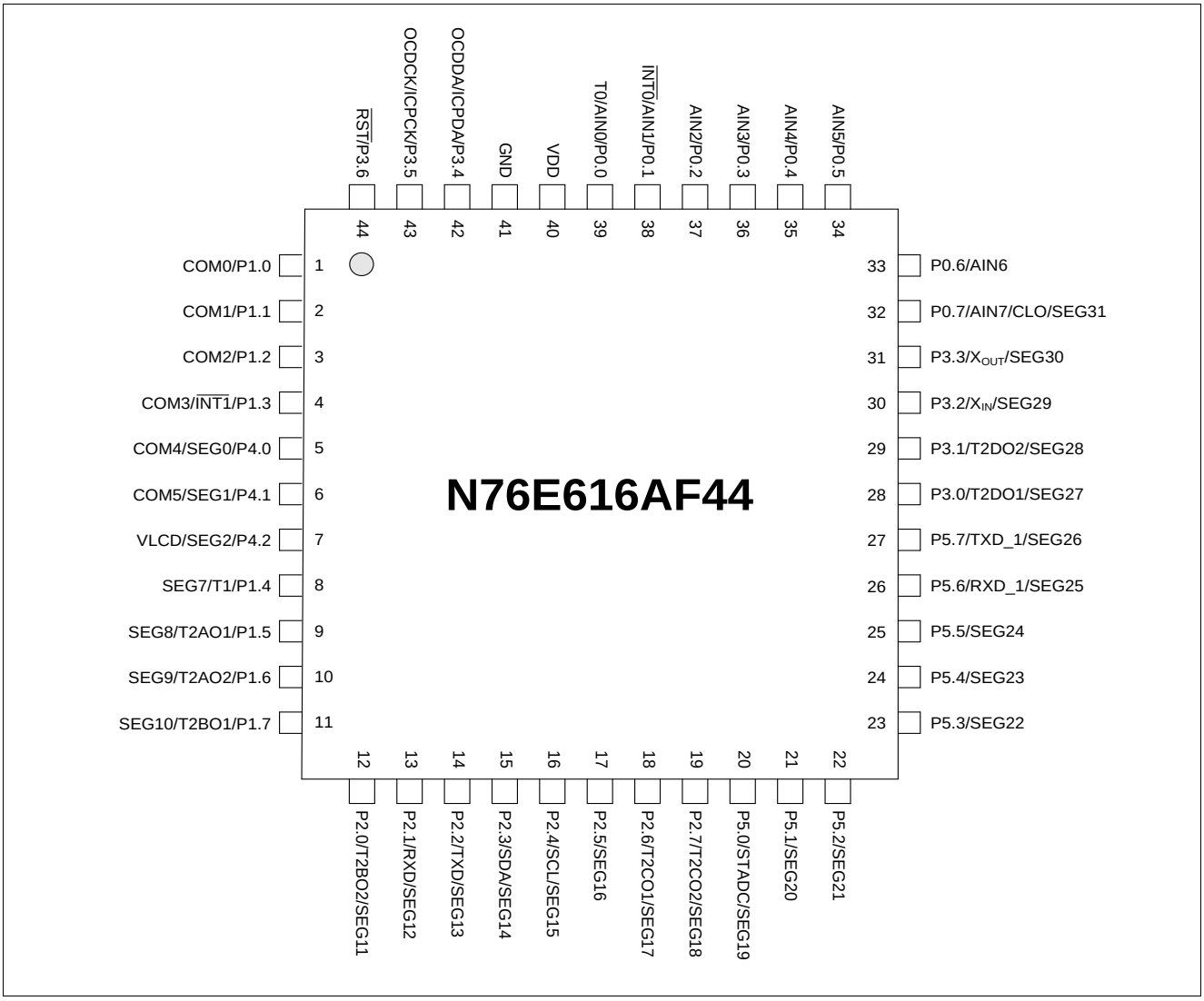

Figure 4.2 Pin Assignment of PQFP-44 Package

Table 4–1. Pin Description

Pin Number		Symbol	Multi-Function Description[1]
PQFP44 LQFP44	LQFP48		
43	47	P3.5/ICPCK/ OCDCK	**P3.5:** Port 3 bit 5. **ICPCK:** ICP clock input. **OCDCK:** OCD clock input.
44	48	P3.6/$\overline{RST}$	**P3.6:** Port 3 bit 6 input pin available when RPD (CONFIG0.2) is programmed as 0. **$\overline{RST}$:** $\overline{RST}$ pin is a Schmitt trigger input pin for hardware device reset. A low on this pin resets the device. $\overline{RST}$ pin has an internal pull-up resistor allowing power-on reset by simply connecting an external capacitor to GND.
P4[6:0]			**P4:** Port 4 is a byte-addressable, maximum 7-bit I/O port. After reset, all pins are in input-only mode.
5	5	P4.0/SEG0/ COM4	**P4.0:** Port 4 bit 0. **SEG0:** LCD segment 0 output. **COM4:** LCD common 4 output.
6	6	P4.1/SEG1/ COM5	**P4.1:** Port 4 bit 1. **SEG1:** LCD segment 1 output. **COM5:** LCD common 5 output.
7	7	P4.2/SEG2	**P4.2:** Port 4 bit 2. **SEG2:** LCD segment 2 output.
-	8	P4.3/SEG3	**P4.3:** Port 4 bit 3. **SEG3:** LCD segment 3 output.
-	9	P4.4/SEG4	**P4.4:** Port 4 bit 4. **SEG4:** LCD segment 4 output.
-	10	P4.5/SEG5	**P4.5:** Port 4 bit 5. **SEG5:** LCD segment 5 output.
-	11	P4.6/SEG6	**P4.6:** Port 4 bit 6. **SEG6:** LCD segment 6 output.
P5[7:0]			**P5:** Port 5 is a bit-addressable, 8-bit I/O port. After reset, all pins are in input-only mode.
20	24	P5.0/STADC/ SEG19	**P5.0:** Port 5 bit 0. **STADC:** External start ADC trigger. **SEG19:** LCD segment 19 output.
21	25	P5.1/SEG20	**P5.1:** Port 5 bit 1. **SEG20:** LCD segment 20 output.
22	26	P5.2/SEG21	**P5.2:** Port 5 bit 2. **SEG21:** LCD segment 21 output.
23	27	P5.3/SEG22	**P5.3:** Port 5 bit 3. **SEG22:** LCD segment 22 output.
24	28	P5.4/SEG23	**P5.4:** Port 5 bit 4. **SEG23:** LCD segment 23 output.
25	29	P5.5/SEG24	**P5.5:** Port 5 bit 5. **SEG24:** LCD segment 24 output.
26	30	P5.6/RXD_1/ SEG25	**P5.6:** Port 5 bit 6. **RXD_1:** Serial port 1 receive input. **SEG25:** LCD segment 25 output.
27	31	P5.7/TXD_1/ SEG26	**P5.7:** Port 5 bit 7. **TXD_1:** Serial port 1 transmit data output. **SEG26:** LCD segment 26 output.

[1] All I/O pins can be configured as an interrupt pin. This feature is not listed in multi-function description. See Section 16. "Pin Interrupt" on page 98.

7. 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 80C51 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 80C51 .It is used mainly for MUL and DIV instructions.

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 16-bit data pointer. DPL combined with DPH serve as a 16-bit data pointer DPTR to access indirect addressed RAM or Program Memory. DPS (AUXR1.0) bit decides which data pointer, DPTR or DPTR1, is activated.

Bit	Name	Description
1	T2AOE2	**Timer 2A output enable 2** 0 = T2AO2 output Disabled. 1 = T2AO2 output Enabled.
0	T2AOE1	**Timer 2A output enable 1** 0 = T2AO1 output Disabled. 1 = T2AO1 output Enabled.

R2AL – Timer 2A Reload Low Byte

7	6	5	4	3	2	1	0
R2AL[7:0]							
R/W							

Address: CCH Reset value: 0000 0000b

Bit	Name	Description
7:0	R2AL[7:0]	**Timer 2A reload low byte** In auto-reload mode, it holds the low byte of the reload value of Timer 2A. In PWM mode, it holds the low duty value.

R2AH – Timer 2A Reload High Byte

7	6	5	4	3	2	1	0
R2AH[7:0]							
R/W							

Address: CDH Reset value: 0000 0000b

Bit	Name	Description
7:0	R2AH[7:0]	**Timer 2A reload high byte** In auto-reload mode, it holds the high byte of the reload value of Timer 2A. In PWM mode, it holds the high duty value.

R2BL – Timer 2B Reload Low Byte

7	6	5	4	3	2	1	0
R2BL[7:0]							
R/W							

Address: CEH Reset value: 0000 0000b

Bit	Name	Description
7:0	R2BL[7:0]	**Timer 2B reload low byte** In auto-reload mode, it holds the low byte of the reload value of Timer 2B. In PWM mode, it holds the low duty value.

Bit	Name	Description
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.
2:0	WKPS[2:0]	**WKT pre-scalar** These bits determine the pre-scale of WKT clock. 000 = 1/1. 001 = 1/4. 010 = 1/16. 011 = 1/64. 100 = 1/256. 101 = 1/512. 110 = 1/1024. 111 = 1/2048.

RWK – Self Wake-up Timer Reload Byte

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

Address: 86H Reset value: 0000 0000b

Bit	Name	Description
7:0	RWK[7:0]	**WKT reload byte** It holds the 8-bit reload value of WKT. Note that RWK should not be FFH if the pre-scale is 1/1 for implement limitation.

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 6–2. SFR Definitions and Reset Values

Bit	Name	Description
7	SMOD	**Serial port 0 double baud rate enable** Setting this bit doubles the serial port baud rate when UART0 is in Mode 2 or when Timer 1 overflow is used as the baud rate source of UART0 Mode 1 or 3. See Table 14–1. Serial Port 0 Mode Description for details.
6	SMOD0	**Serial port 0 framing error flag access enable** 0 = SCON.7 accesses to SM0 bit. 1 = SCON.7 accesses to FE bit.

T3CON – Timer 3 Control

7	6	5	4	3	2	1	0
SMOD_1	SMOD0_1	BRCK	TF3	TR3	T3PS[2:0]		
R/W	R/W	R/W	R/W	R/W	R/W		

Address: C4H Reset value: 0000 0000b

Bit	Name	Description
7	SMOD_1	**Serial port 1 double baud rate enable** Setting this bit doubles the serial port baud rate when UART1 is in Mode 2. See Table 14–2. Serial Port 1 Mode Description for details.
6	SMOD0_1	**Serial port 1 framing error access enable** 0 = SCON_1.7 accesses to SM0_1 bit. 1 = SCON_1.7 accesses to FE_1 bit.

Table 14–1. Serial Port 0 Mode Description

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

Table 14–2. Serial Port 1 Mode Description

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

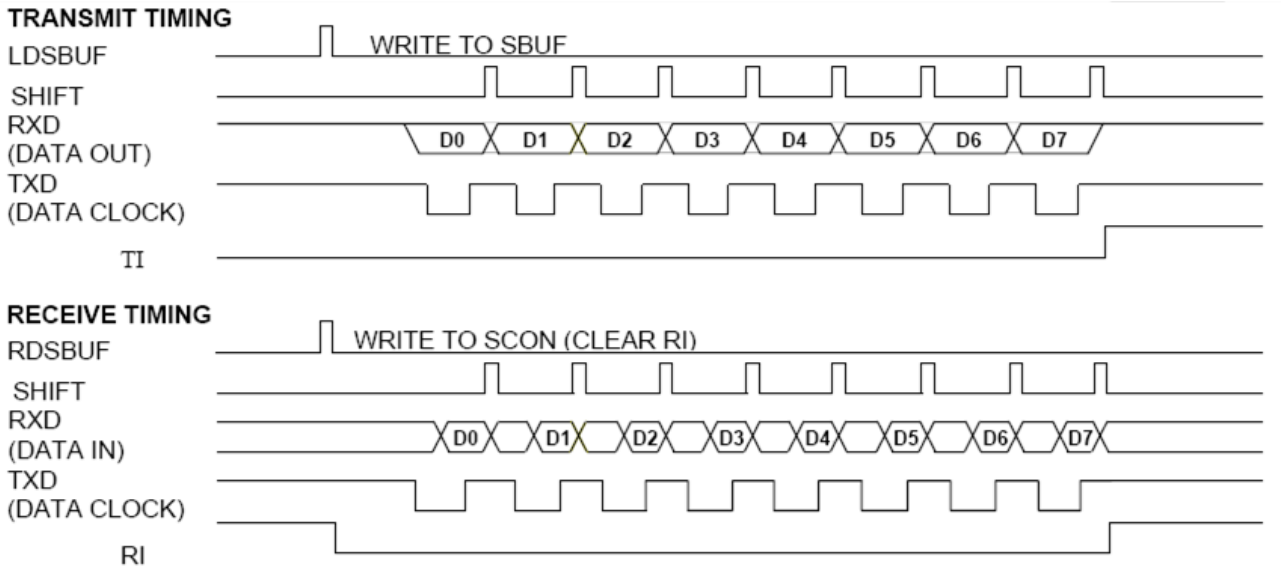

Figure 14.1. Serial Port Mode 0 Timing Diagram

As shown there is one bi-directional data line (RXD) and one shift clock line (TXD). The shift clocks are used to shift data in or out of the serial port controller bit by bit for a serial communication. Data bits enter or emit 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 clocks 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 the condition REN (SCON.4) = 1 and RI (SCON.0) = 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. User can clear RI to triggering the next byte reception.

14.2 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 Timer 1 determines the baud rate. SMOD (PCON.7) setting 1 makes the baud rate double. Figure 14.2 shows the associated timings of the serial port in Mode 1 for transmitting and receiving.

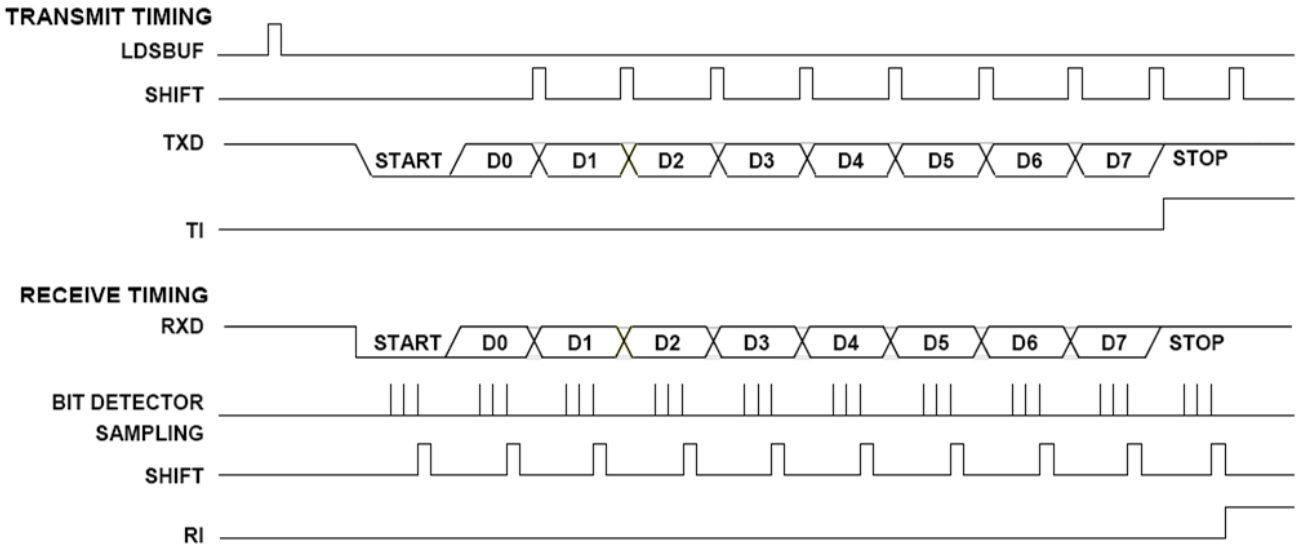

Figure 14.2. Serial Port Mode 1 Timing Diagram

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 should 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 and the received data matches “Given” or “Broadcast” address. (For enhancement function, see 14.7 “Multiprocessor Communication” and 14.8 “Automatic Address Recognition”.)

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-to-0 transition on RXD pin to start next data reception.

14.3 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 or to label address or data frame for multiprocessor communication. The baud rate is fixed as 1/32 or 1/64 the system clock

address, the slave should leave the SDA line high 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 should 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 releases the SDA line to allow the master to generate a STOP or a repeated START condition.

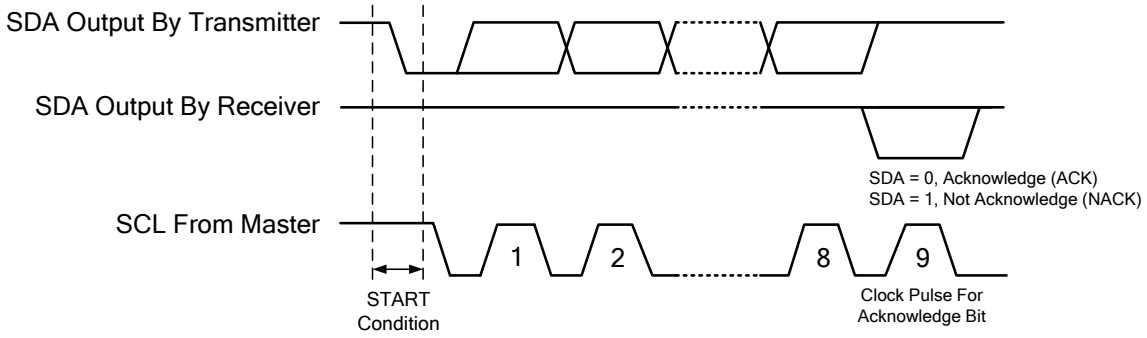

Figure 15.5. Acknowledge Bit

15.1.4 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 continued by the winning master. However, the arbitration lost master continues generating clock pulses on SCL line until the end of the byte in which it loses the arbitration.

All masters continuously monitoring the SDA line after outputting data carry out arbitration. If the value read from the SDA line does not match the value that the master has to 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. 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.

15.3 Operating Modes

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

15.3.1 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. 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 continued by loading I2DAT with the target slave address and the data direction bit “write” (SLA+W). The SI bit should 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 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 also be generated without sending STOP condition to immediately initial another transmission.

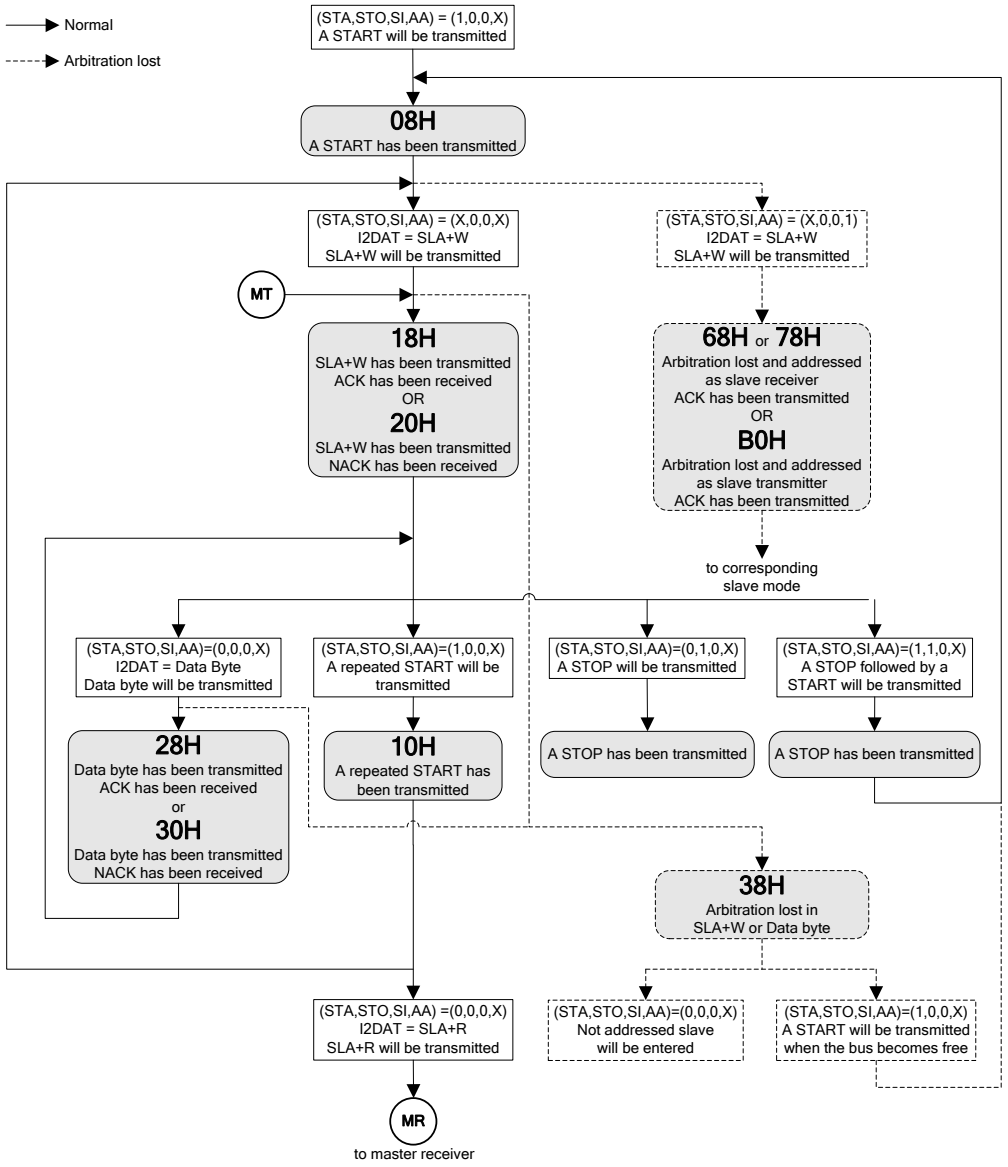

Figure 15.7. Flow and Status of Master Transmitter Mode

15.3.2 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.2) 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.

```
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;
```

16. PIN INTERRUPT

The N76E616 provides pin interrupt input for each I/O pin to detect pin state if button or keypad set is used. A maximum 8-channel pin interrupt detection can be assigned by I/O port sharing. The pin interrupt is generated when any key is pressed on a keyboard or keypad, which produces an edge or level triggering event. Pin interrupt may be used to wake the CPU up from Idle or Power-down mode.

Each channel of pin interrupt can be enabled and polarity controlled independently by PIPEN and PINEN register. PICON selects which port that the pin interrupt is active. PITYP defines which type of pin interrupt is used, level detect or edge detect. Each channel also has its own interrupt flag. There are total eight pin interrupt flags located in PIF register. The respective flags for each pin interrupt channel allow the interrupt service routine to poll on which channel on which the interrupt event occurs. All flags in PIF register are set by hardware and should be cleared by software.

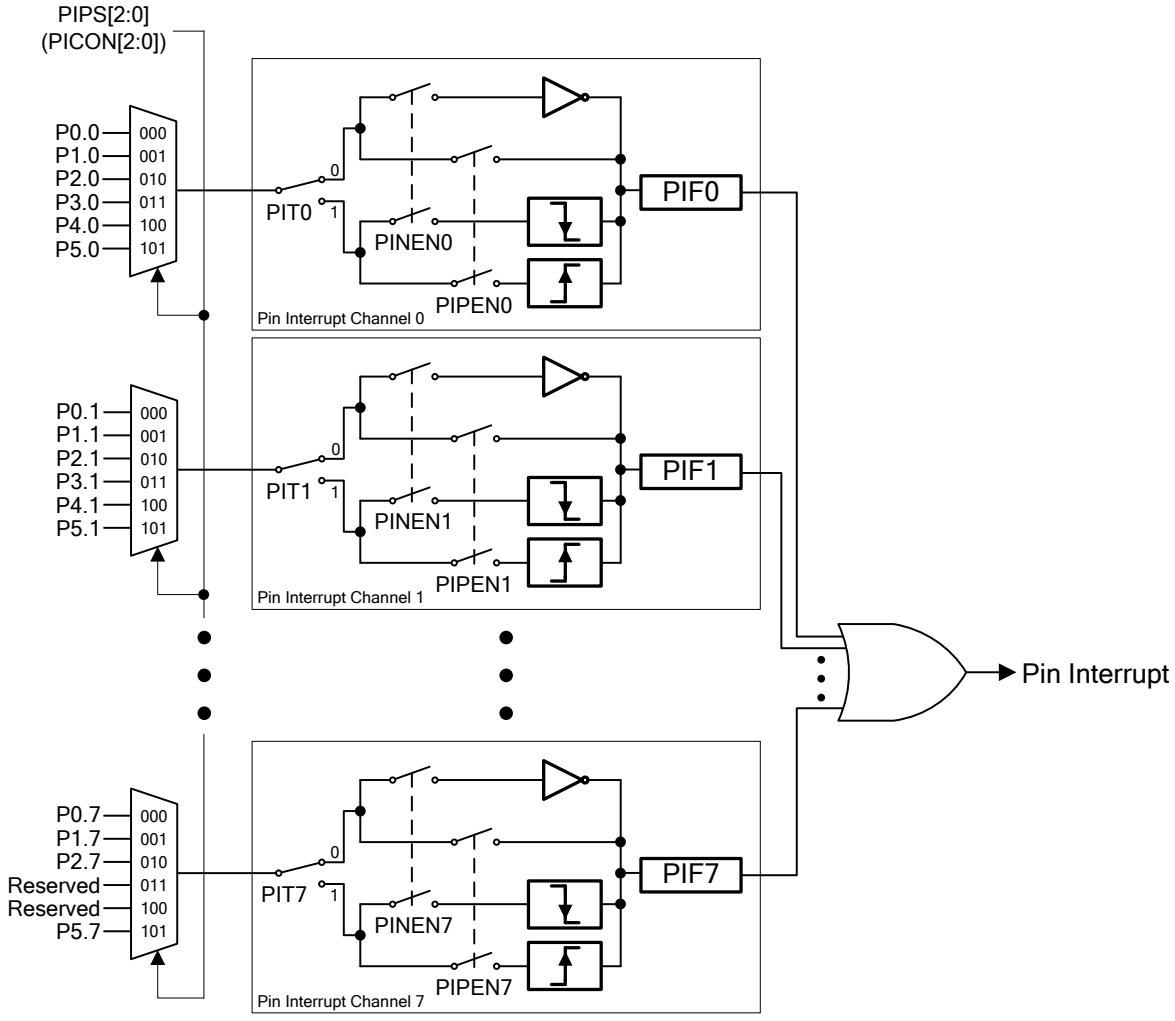

Figure 16.1. Pin Interface Block Diagram

ADCCON1 – ADC Control 1

7	6	5	4	3	2	1	0
-	ADCDIV[2:0]			-	-	ADCEX	ADCEN
-	R/W			-	-	R/W	R/W

Address: E1H Reset value: 0010 0000b

Bit	Name	Description
7	-	**Reserved**
6:4	ADCDIV[2:0]	**ADC clock divider** 000 = F_{ADC} is F_{SYS}/1. 001 = F_{ADC} is F_{SYS}/2. 010 = F_{ADC} is F_{SYS}/4. (By default.) 011 = F_{ADC} is F_{SYS}/8. 100 = F_{ADC} is F_{SYS}/16. 101 = F_{ADC} is F_{SYS}/32. 110 = F_{ADC} is F_{SYS}/64. 111 = F_{ADC} is F_{SYS}/128.
3:2	-	**Reserved**
1	ADCEX	**ADC external conversion trigger select** This bit selects the methods of triggering an A/D conversion. 0 = A/D conversion is started only via setting ADCS bit. 1 = A/D conversion is started via setting ADCS bit or by the falling edge of STADC pin. Note that while ADCS is 1 (busy in converting), the ADC will ignore the following external trigger until ADCS is hardware cleared.
0	ADCEN	**ADC enable** 0 = ADC circuit OFF. 1 = ADC circuit ON.

ADCCON2 – ADC Control 2

7	6	5	4	3	2	1	0
-	ADCMPOP	ADCMPEN	ADCMPO	-	-	-	-
-	R/W	R/W	R	-	-	-	-

Address: E2H Reset value: 0000 0000b

Bit	Name	Description
7	-	**Reserved**
6	ADCMPOP	**ADC comparator output polarity** 0 = ADCMPO is 1 if ADCR[9:0] is greater than or equal to ADCMP[9:0]. 1 = ADCMPO is 1 if ADCR[9:0] is less than ADCMP[9:0].
5	ADCMPEN	**ADC result comparator enable** 0 = ADC result comparator Disabled. 1 = ADC result comparator Enabled.
4	ADCMPO	**ADC comparator output value** This bit is the output value of ADC result comparator based on the setting of ACMPOP. This bit updates after every A/D conversion complete.
3:0	-	**Reserved**

Each COM pin is selected sequentially according to the duty in a frame period. For 1/4 duty, COM0 to COM3 generate LCD driving signals. Whereas, for 1/3 duty, COM0 to COM2 generate signals. COM3 does not generate signals but functions as a general purpose I/O. When 1/6 duty is selected, SEG0 and SEG1 function as COM4 and COM5. The original SEG0 and SEG1 with their control bits are all unavailable.

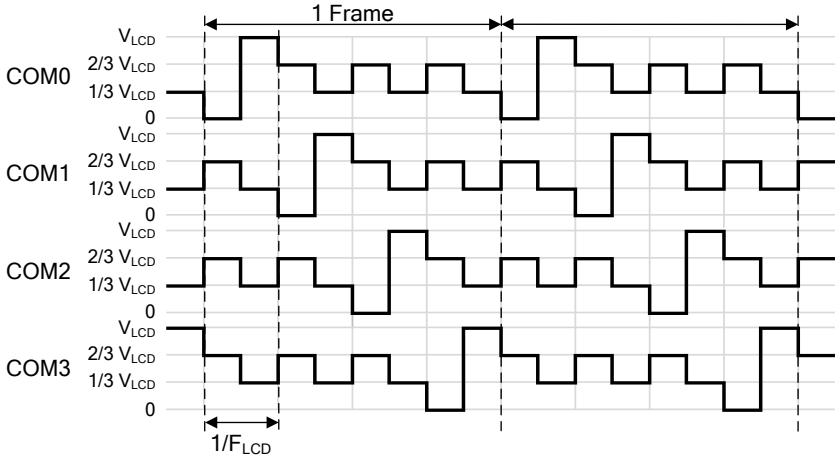

Figure 18.1. COM Driving Signals (1/4 Duty and 1/3 Bias)

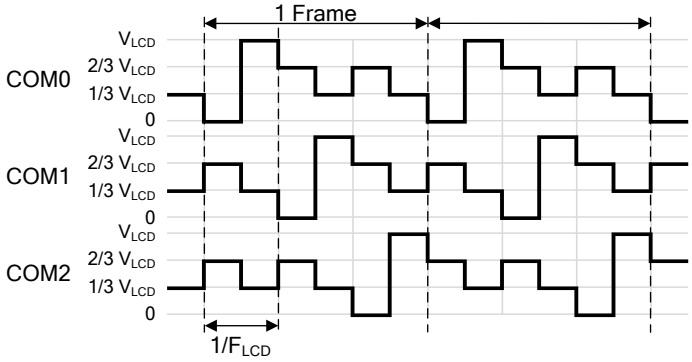

Figure 18.2. COM Driving Signals (1/3 Duty and 1/3 Bias)

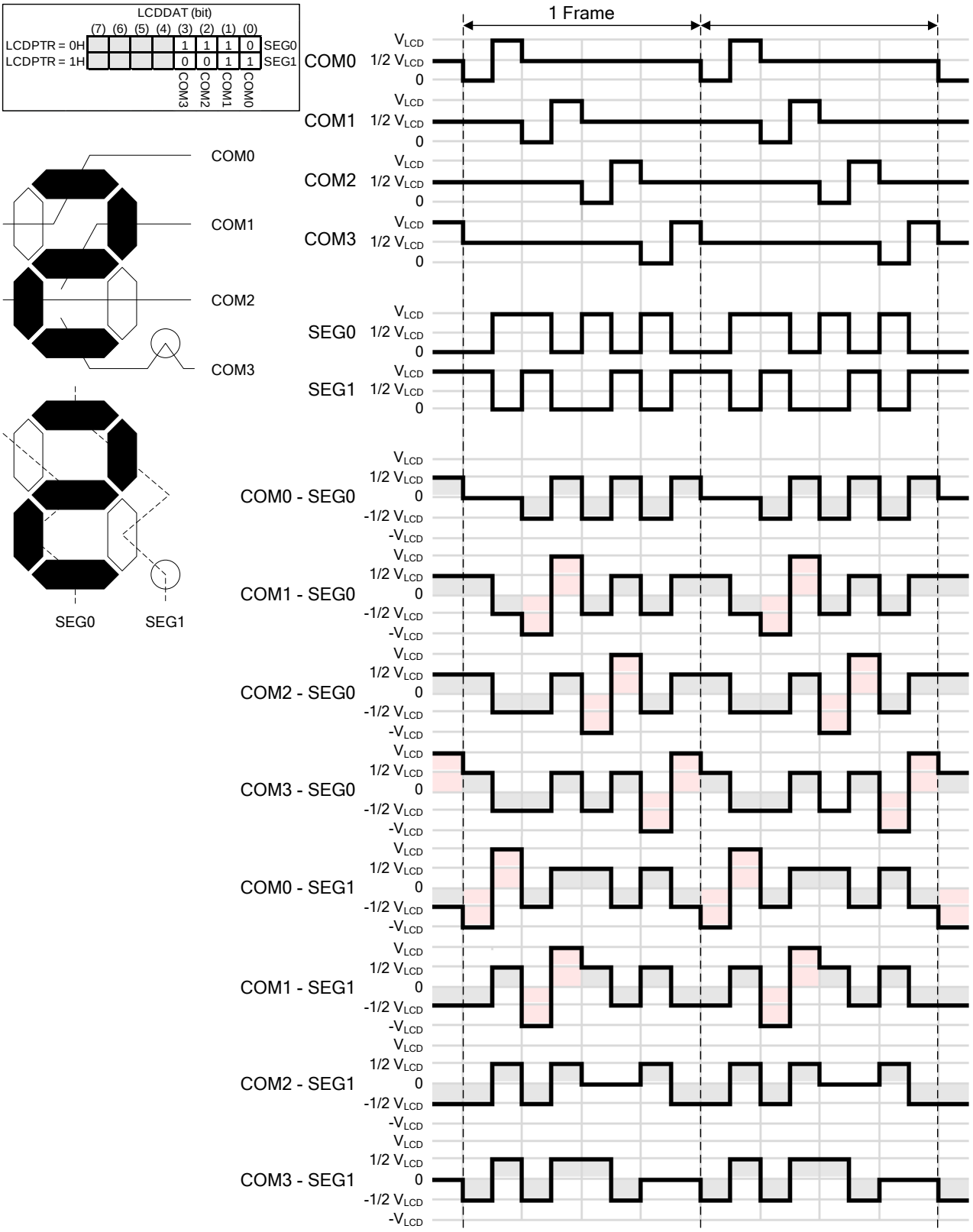

Figure 18.9. Example of COM and SEG Driving Signals (1/2 Bias)

21. IN-APPLICATION-PROGRAMMING (IAP)

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. The N76E616 carried out the flash operation with convenient mechanism to help user re-programming the flash content by In-Application-Programming (IAP). IAP is an in-circuit electrical erasure and programming method through software.

After IAP enabling by setting IAPEN (CHPCON.0 with TA protected) and setting the enable bit in IAPUEN that allows the target block to be updated, user can easily fill the 16-bit target address in IAPAH and IAPAL, data in IAPFD, and command in IAPCN. Then the IAP is ready to begin by setting a triggering bit IAPGO (IAPTRG.0). Note that IAPTRG is also TA protected. At this moment, the CPU holds the Program Counter and the built-in IAP automation takes over to control the internal charge-pump for high voltage and the detail signal timing. The erase and program time is internally controlled disregard of the operating voltage and frequency. Nominally, a page-erase time is 28 ms and a byte-program time is 50 μs. After IAP action completed, the Program Counter continues to run the following instructions. The IAPGO bit will be automatically cleared. An IAP failure flag, IAPFF (CHPCON.6), can be check whether the previous IAP operation was successful or not. Through this progress, user can easily erase, program, and verify the Flash Memory by just taking care of pure software.

The following registers are related to IAP processing.

CONFIG2

7	6	5	4	3	2	1	0
CBODEN	-	CBOV[1:0]		BOIAP	CBORST	-	-
R/W	-	R/W		R/W	R/W	-	-

Factory default value: 1111 1111b

Bit	Name	Description
3	BOIAP	**Brown-out inhibiting IAP** This bit decides whether IAP erasing or programming is inhibited by brown-out status. This bit is valid only when brown-out detection is enabled. 1 = IAP erasing or programming is inhibited if V_{DD} is lower than V_{BOD}. 0 = IAP erasing or programming is allowed under any workable V_{DD}.

CHPCON – Chip Control (TA protected)

7	6	5	4	3	2	1	0
SWRST	IAPFF	-	-	-	-	BS	IAPEN
W	R/W	-	-	-	-	R/W	R/W

Address: 9FH Reset value: see Table 6–2. SFR Definitions and Reset Values

Bit	Name	Description
6	IAPFF	**IAP fault flag** The hardware will set this bit after IAPGO (ISPTRG.0) is set if any of the following condition is met: (1) The accessing address is oversize.

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 follow below.
		`CLR EA`
		`MOV TA,#0AAH`
		`MOV TA,#55H`
		`ORL IAPTRG,#01H`
		`(SETB EA)`

21.1 IAP Commands

The N76E616 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: 2FH
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
CONFIG byte-program	11	1	0	0001	CONFIG0: 0000H CONFIG1: 0001H CONFIG2: 0002H CONFIG4: 0004H	Data in

```
        MOV     IAPCN,#BYTE_PROGRAM_AP
        MOV     IAPAH,#00h
        MOV     IAPAL,#00h
        MOV     DPTR,#AP_code
Program_AP_Loop:
        CLR     A
        MOVC    A,@A+DPTR
        MOV     IAPFD,A
        CALL    Trigger_IAP
        INC     DPTR
        INC     IAPAL
        MOV     A,IAPAL
        CJNE    A,#14,Program_AP_Loop
        RET

Program_AP_Verify:
        MOV     IAPCN,#BYTE_READ_AP
        MOV     IAPAH,#00h
        MOV     IAPAL,#00h
        MOV     DPTR,#AP_code
Program_AP_Verify_Loop:
        CALL    Trigger_IAP
        CLR     A
        MOVC    A,@A+DPTR
        MOV     B,A
        MOV     A,IAPFD
        CJNE    A,B,Program_AP_Verify_Error
        INC     DPTR
        INC     IAPAL
        MOV     A,IAPAL
        CJNE    A,#14,Program_AP_Verify_Loop
        RET

Program_AP_Verify_Error:
        CALL    Disable_IAP
        MOV     P0,#00h
        SJMP    $

;*****************************************************************
;             IAP CONFIG Function
;*****************************************************************
Erase_CONFIG:
        MOV     IAPCN,#ALL_ERASE_CONFIG
        MOV     IAPAH,#00h
        MOV     IAPAL,#00h
        MOV     IAPFD,#0FFh
        CALL    Trigger_IAP
        RET

Read_CONFIG:
        MOV     IAPCN,#BYTE_READ_CONFIG
        MOV     IAPAH,#00h
        MOV     IAPAL,#02h
        CALL    Trigger_IAP
        MOV     R7,IAPFD
        RET

Program_CONFIG:
        MOV     IAPCN,#BYTE_PROGRAM_CONFIG
        MOV     IAPAH,#00h
        MOV     IAPAL,#02h
        MOV     A,R7
        ANL     A,#11111011b
        MOV     IAPFD,A                     ;disable BOD reset
```

30. INSTRUCTION SET

The N76E616 executes all the instructions of the standard 80C51 family fully compatible with MCS-51. However, the timing of each instruction is different for it uses high performance 1T 8051 core. The architecture eliminates redundant bus states and implements parallel execution of fetching, decode, and execution phases. The N76E616 uses one clock per machine-cycle. It leads to performance improvement of rate 8.1 (in terms of MIPS) with respect to traditional 12T 80C81 device working at the same clock frequency. However, the real speed improvement seen in any system will depend on the instruction mix.

All instructions are coded within an 8-bit field called an OPCODE. This single byte should be fetched from Program Memory. The OPCODE is decoded by the CPU. It determines what action the CPU 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, which is two or three byte instructions.

Table 30–1 lists all instructions for details. The note of the instruction set and addressing modes are shown below.

Rn (n = 0~7)	Register R0 to R7 of the currently selected Register Bank.
direct	8-bit internal data location's address. It could be an internal data RAM location (00H to 7FH) or an SFR (80H to FFH).
@Ri (i = 0, 1)	8-bit internal data RAM location (00H to FFH) 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 anywhere within the 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.