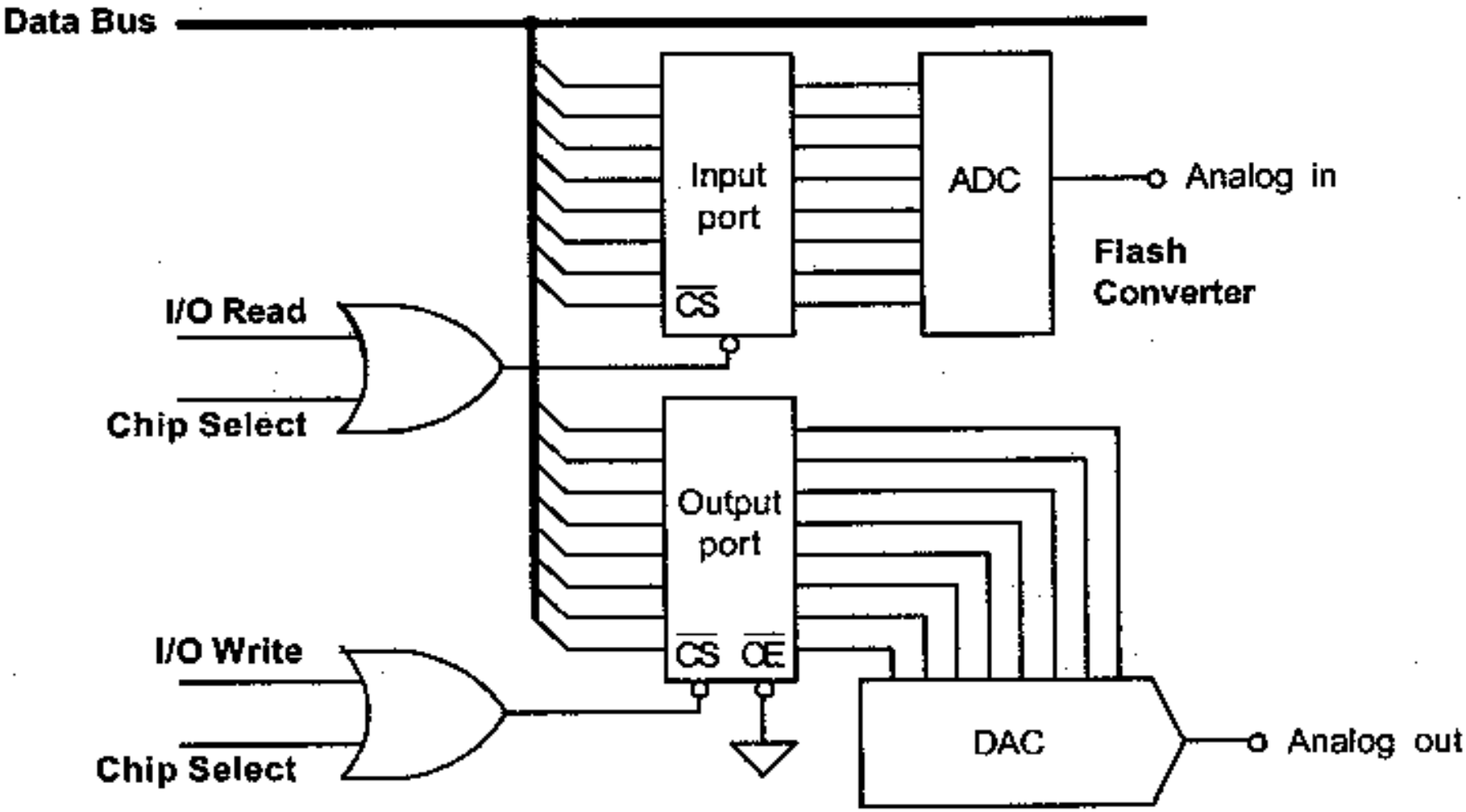
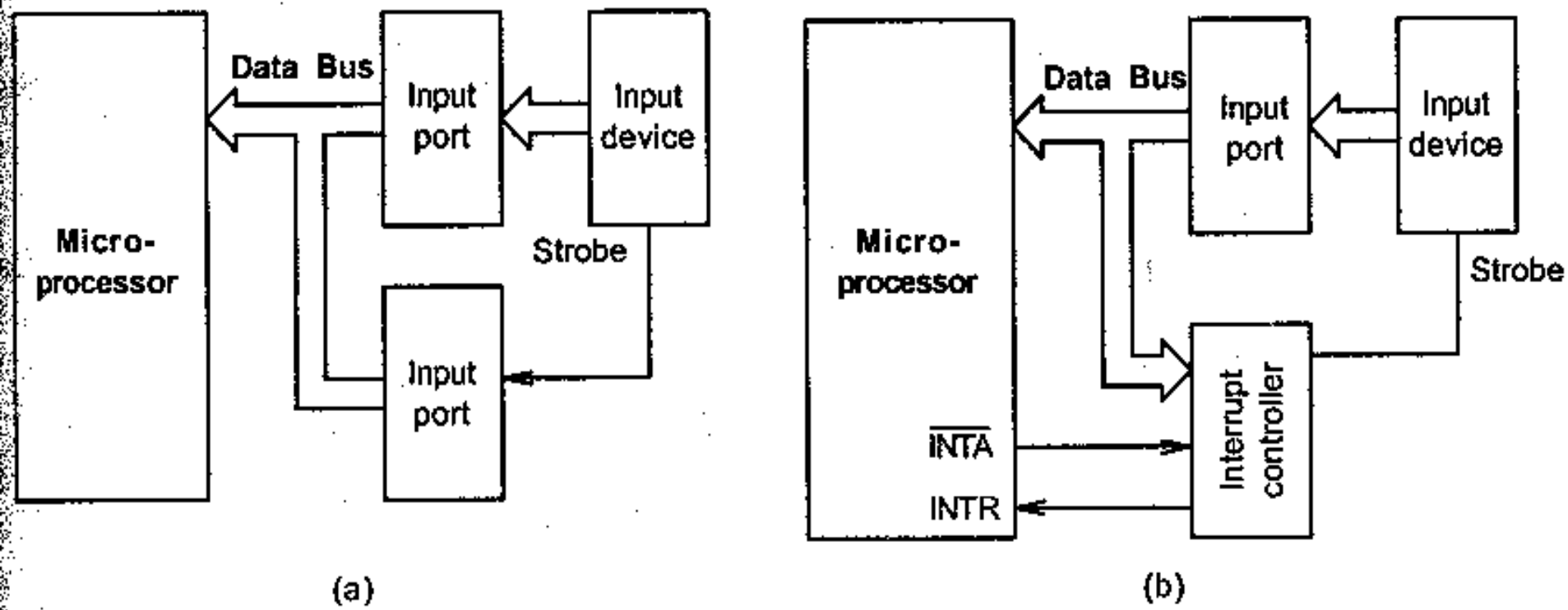
Applications

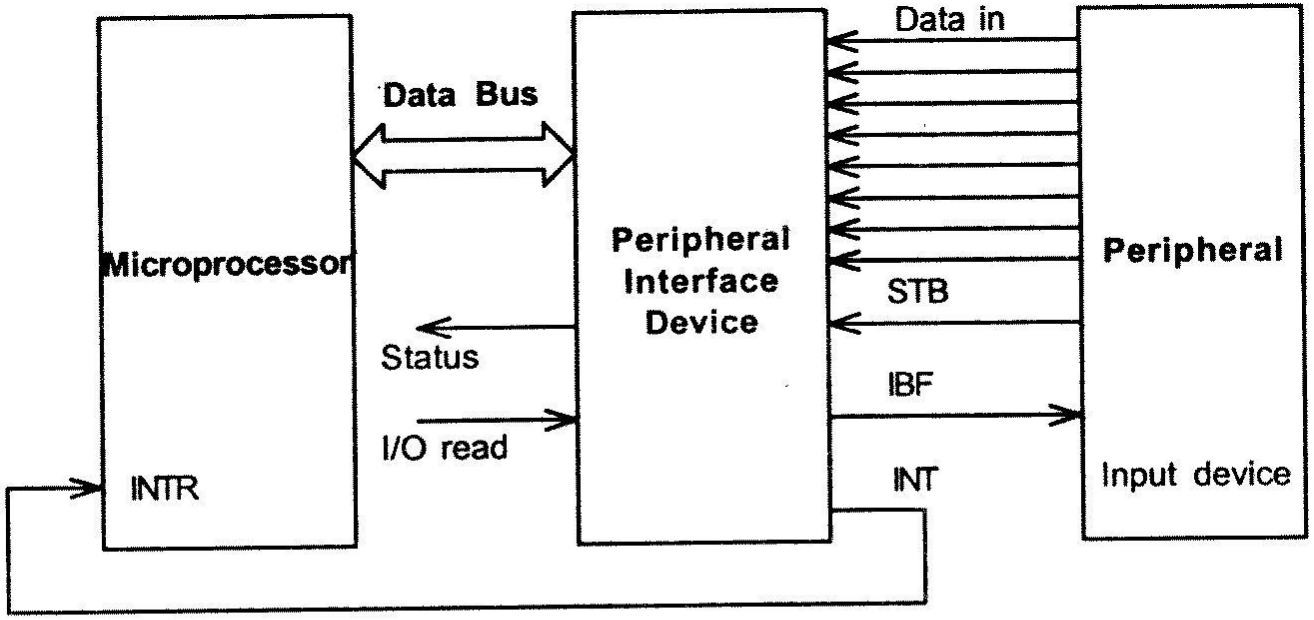
ADC interfacing to Microprocessor



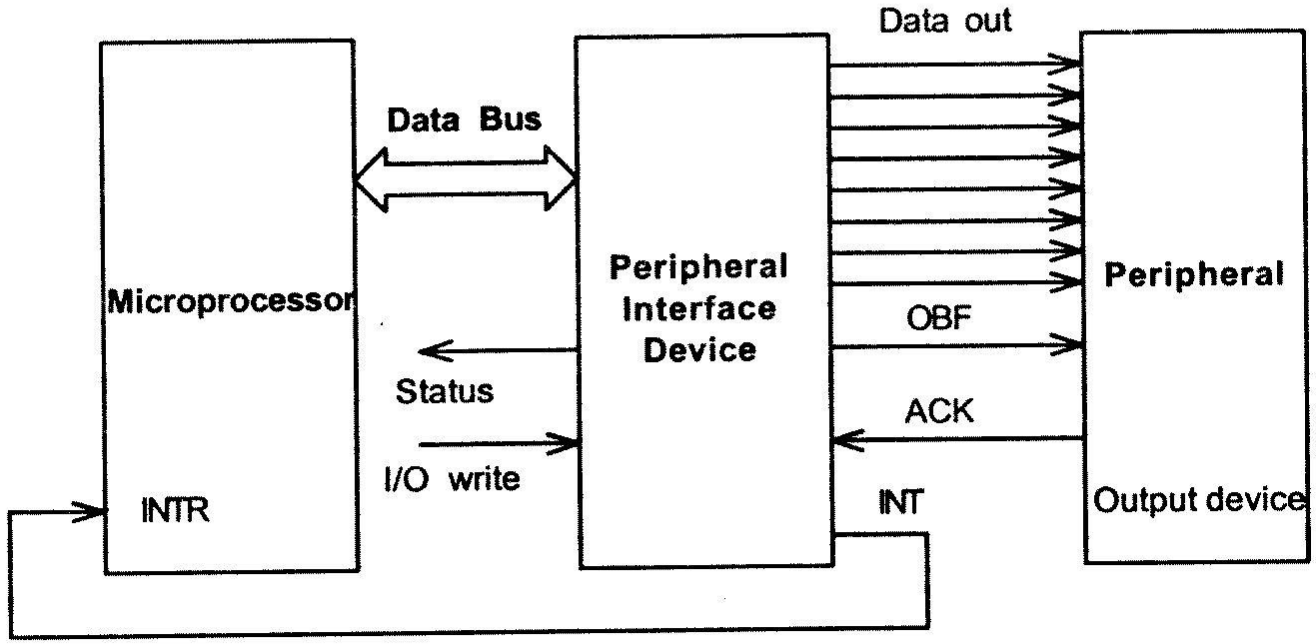
Data transfer by simple I/O.



Strobe I/O—(a) polling, (b) interrupt.



(a)

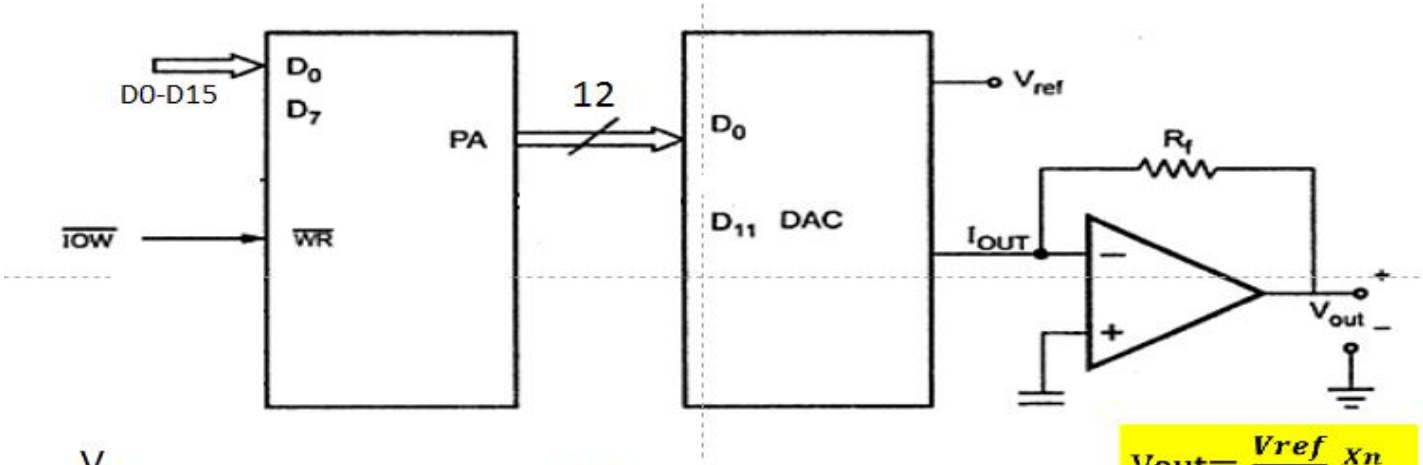


(b)

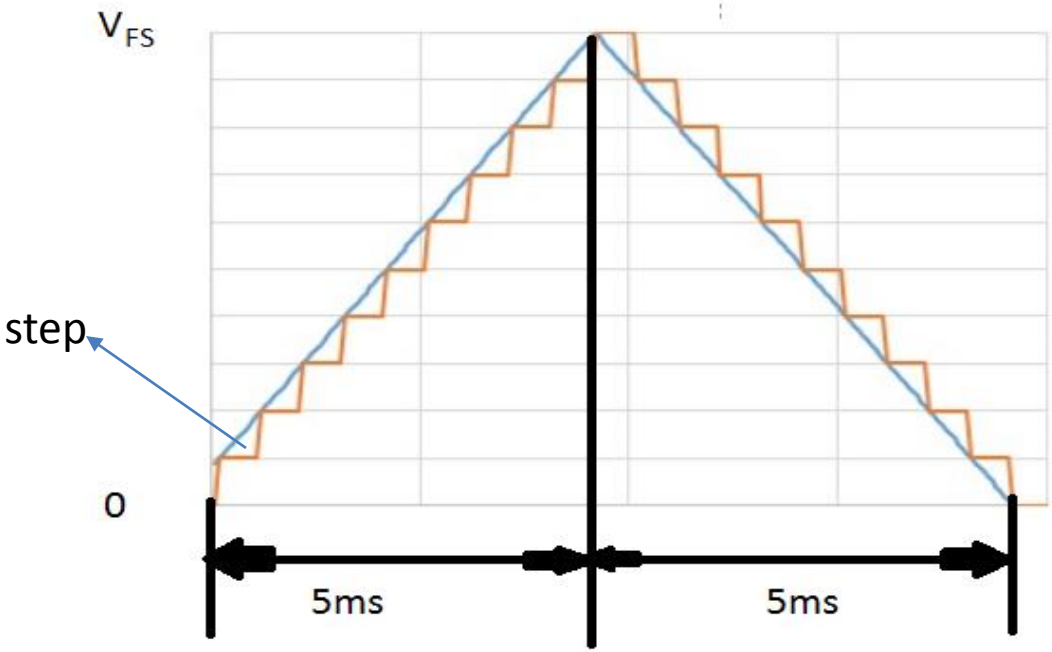
Handshake I/O—(a) input operation, (b) output operation.

EX.

Interface a 12bits to an 8086 microprocessor. (CLK=5MHz). The convertor is connected to a 16 bits port with address 80h. Write the application in AL to generate to the DAC output a triangular waveshape with the frequency of 100Hz and amplitude V_{FS}. The time to change the output to the DAC output is of ~ 5us. Vref=10V



$$Vout = \frac{Vref}{2^n} \times n$$



$$n_{\substack{pasi \\ 0-4095}} = \frac{5\,ms}{5\,\mu s} = 1000\ pasi$$ (steps)

$$n_{LSB/pas} = \frac{4095}{1000} = 4{,}095 \sim 5\ LSB/pas$$

$$4095 = 5 \cdot 819 \Rightarrow 819\ pasi\ (0 \ldots 4095)$$

$$t_{pas} = \frac{5\,ms}{819} \simeq 6{,}1\,\mu s > 5\,\mu s$$

$$\frac{t_{pas}}{t_{cicl.}} = \frac{6{,}1\,\mu s}{0{,}2\,\mu s} = 30{,}05\ \underset{procesor}{cicli} \sim 30\ cicli/pas$$ Cycles/step

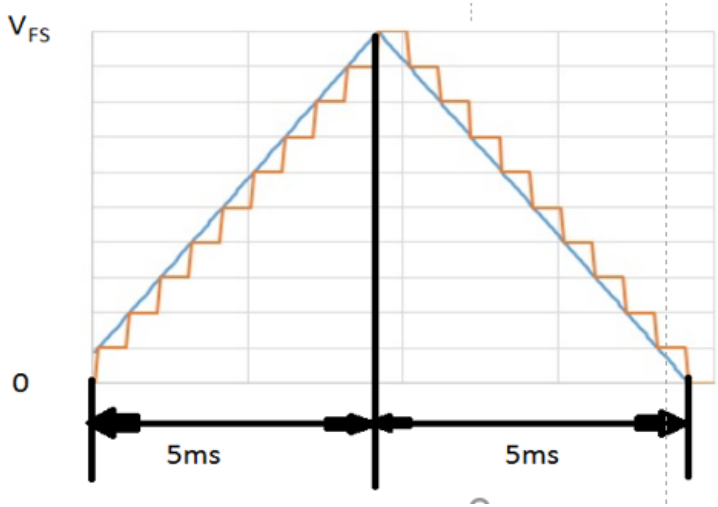
Cx=contor pasi; BX=val. DAC; AX

```
ST:     MOV CX,819    ;4CLK
        MOV BX,0      ;4CLK
BK0:    MOV AX,BX     ;2CLK
        OUT  80h,AX   ;10CLK
        ADD  BX,5     ;4CLK
        MOV  DX,DX    ,2CLK
        MOV  DX,DX    ;2CLK
        NOP           ;3CLK
        DEC CX        ;3CLK
        JNZ BK0       ;4/2CLK
```

30CLK

```
ST:     MOV CX,819    ;4CLK
        SUB BX,5      ;4CLK
BK1:    MOV AX,BX     ;2CLK
        OUT  80h,AX   ;10CLK
        MOV  BX,BX    ;2CLK
        MOV  BX,BX    ;2CLK
        NOP           ;3CLK
        DEC CX        ;3CLK
        JNZ  BK1      ;4CLK
        JMP ST        ;15CLK
```

30CLK



ADConverter - AD 574A

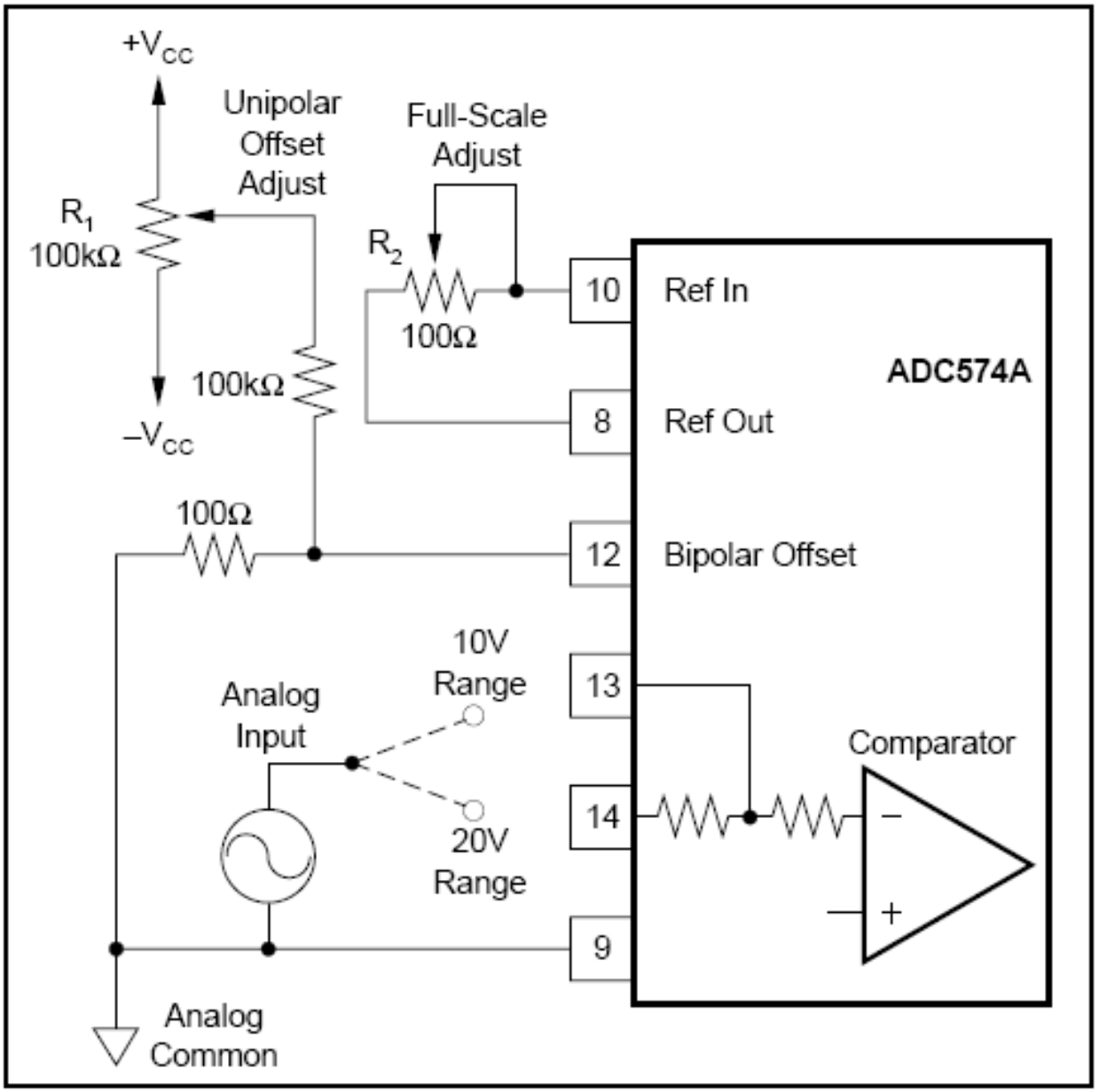


FIGURE 2. Unipolar Configuration.

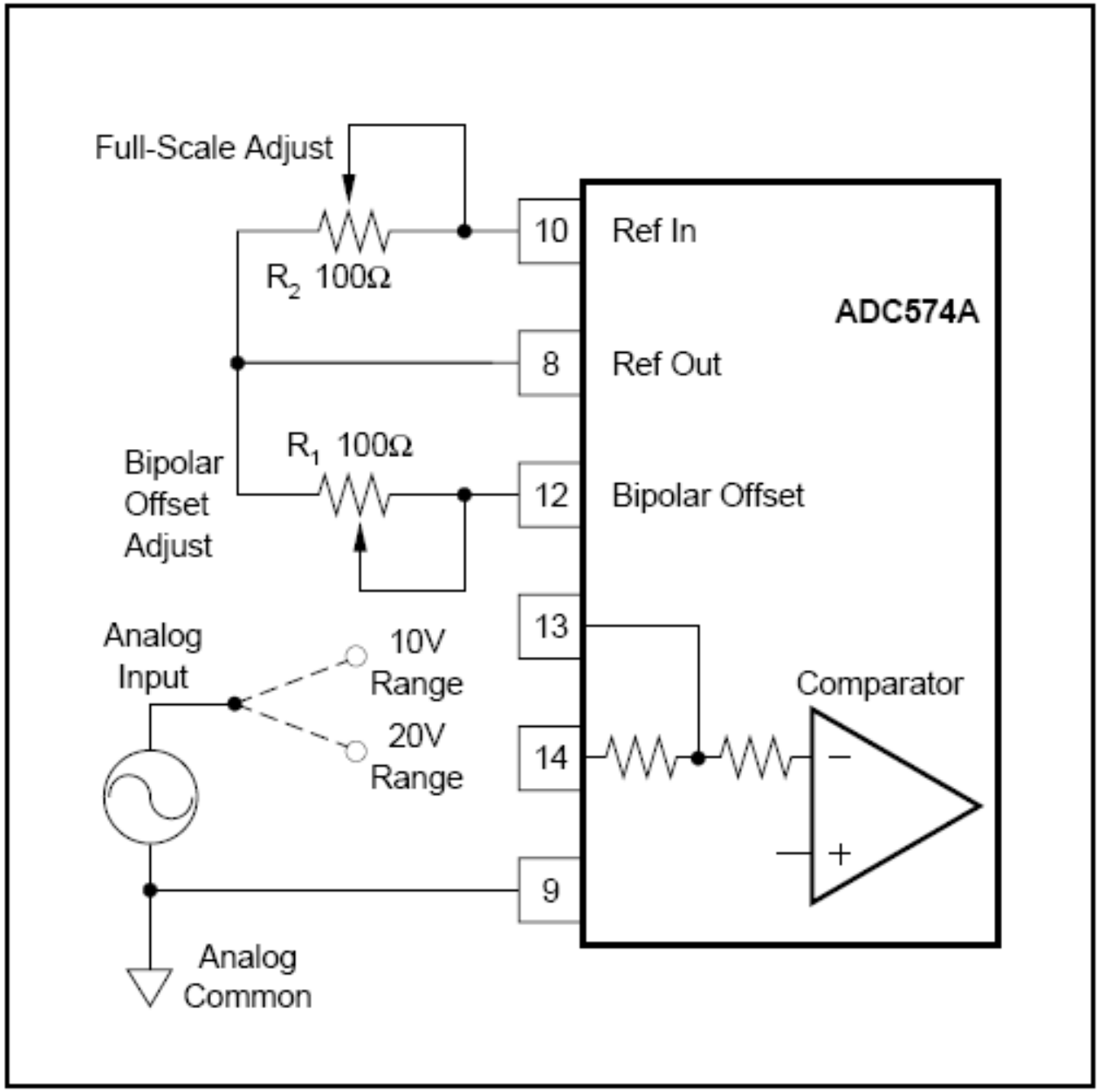


FIGURE 3. Bipolar Configuration.

CE	$\overline{CS}$	R/$\overline{C}$	12/$\overline{8}$	A_0	OPERATION
0	X	X	X	X	None
X	1	X	X	X	None
↑	0	0	X	0	Initiate 12-bit conversion
↑	0	0	X	1	Initiate 8-bit conversion
1	↓	0	X	0	Initiate 12-bit conversion
1	↓	0	X	1	Initiate 8-bit conversion
1	0	↓	X	0	Initiate 12-bit conversion
1	0	↓	X	1	Initiate 8-bit conversion
1	0	1	1	X	Enable 12-bit output
1	0	1	0	0	Enable 8 MSBs only
1	0	1	0	1	Enable 4 LSBs plus 4 trailing zeros

2

1

2

TABLE III. Control Input Truth Table.

PIN DESIGNATION	DEFINITION	FUNCTION
CE (Pin 6)	Chip Enable (active high)	Must be high ("1") to either initiate a conversion or read output data. 0-1 edge may be used to initiate a conversion.
$\overline{CS}$ (Pin 3)	Chip Select (active low)	Must be low ("0") to either initiate a conversion or read output data. 1-0 edge may be used to initiate a conversion.
R/$\overline{C}$(Pin 5)	Read/Convert ("1" = read) ("0" = convert)	Must be low ("0") to initiate either 8- or 12-bit conversions. 1-0 edge may be used to initiate a conversion. Must be high ("1") to read output data. 0-1 edge may be used to initiate a read operation.
A_0 (Pin 4)	Byte Address Short Cycle	In the start-convert mode, A_0 selects 8-bit (A_0 = "1") or 12-bit (A_0 = "0") conversion mode. When reading output data in two 8-bit bytes, A_0 = "0" accesses 8 MSBs (high byte) and A_0 = "1" accesses 4 LSBs and trailing "0s" (low byte).
12/$\overline{8}$ (Pin 2)	Data Mode Select ("1" = 12 bits) ("0" = 8 bits)	When reading output data, 12/$\overline{8}$ = "1" enables all 12 output bits simultaneously. 12/$\overline{8}$ = "0" will enable the MSBs or LSBs as determined by the A_0 line.

TABLE II. ADC574A Control Line Functions.

CONVERSION TIME (4)							
8-Bit Cycle	10	13	17	*	*	*	µs
12-Bit Cycle	15	20	25	*	•	•	µs

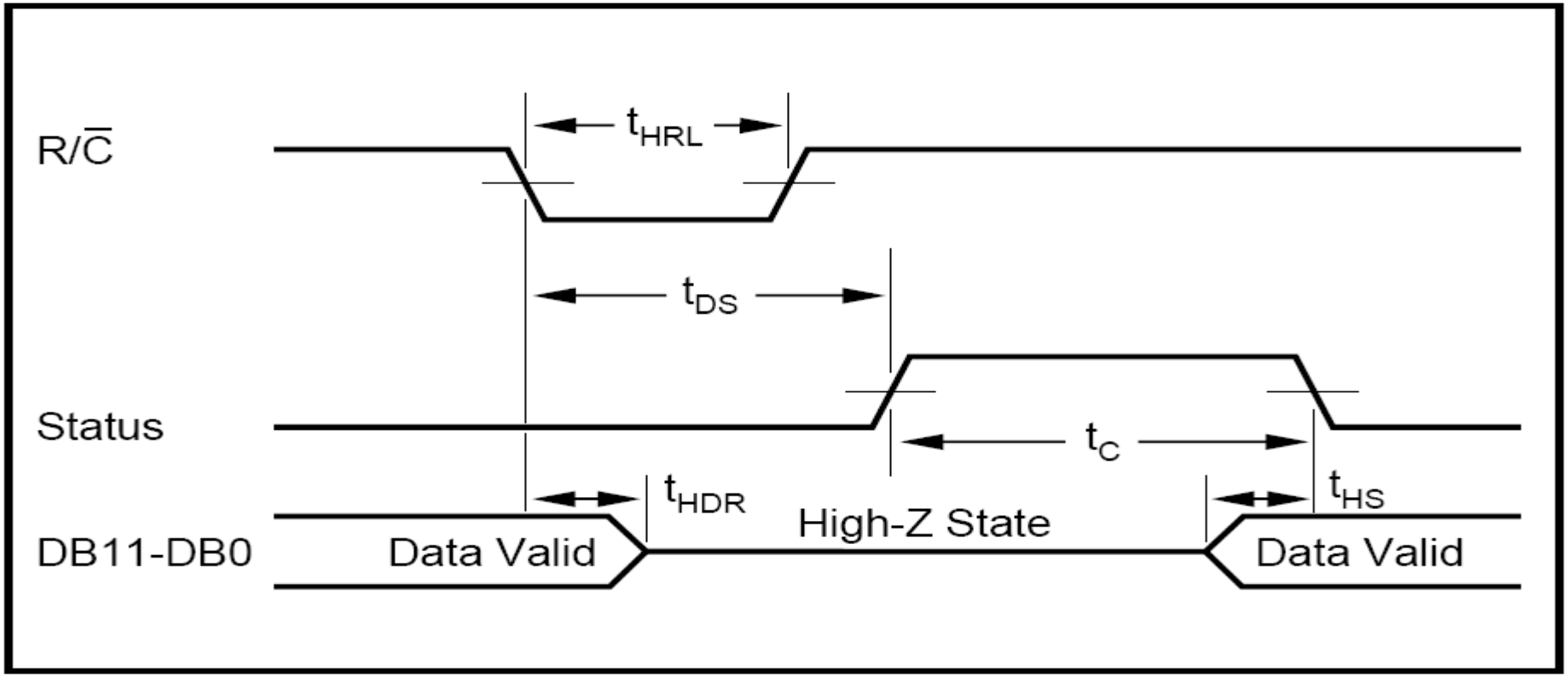


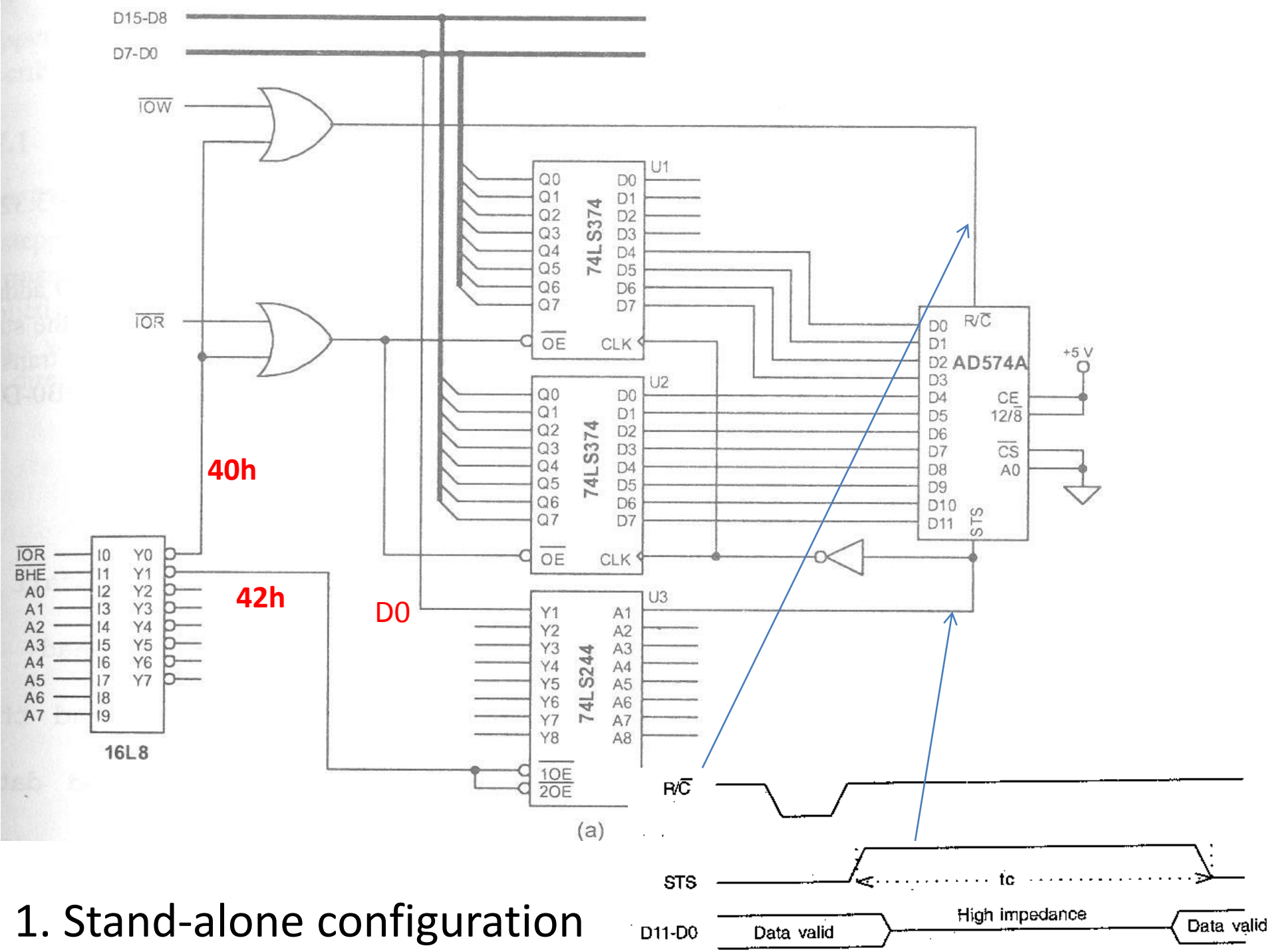
FIGURE 4. R/$\overline{C}$ Pulse Low—Outputs Enabled After Conversion.

LS374

D_n	LE	$\overline{OE}$	O_n
H	⎍	L	H
L	⎍	L	L
X	X	H	Z*

SN74LS244

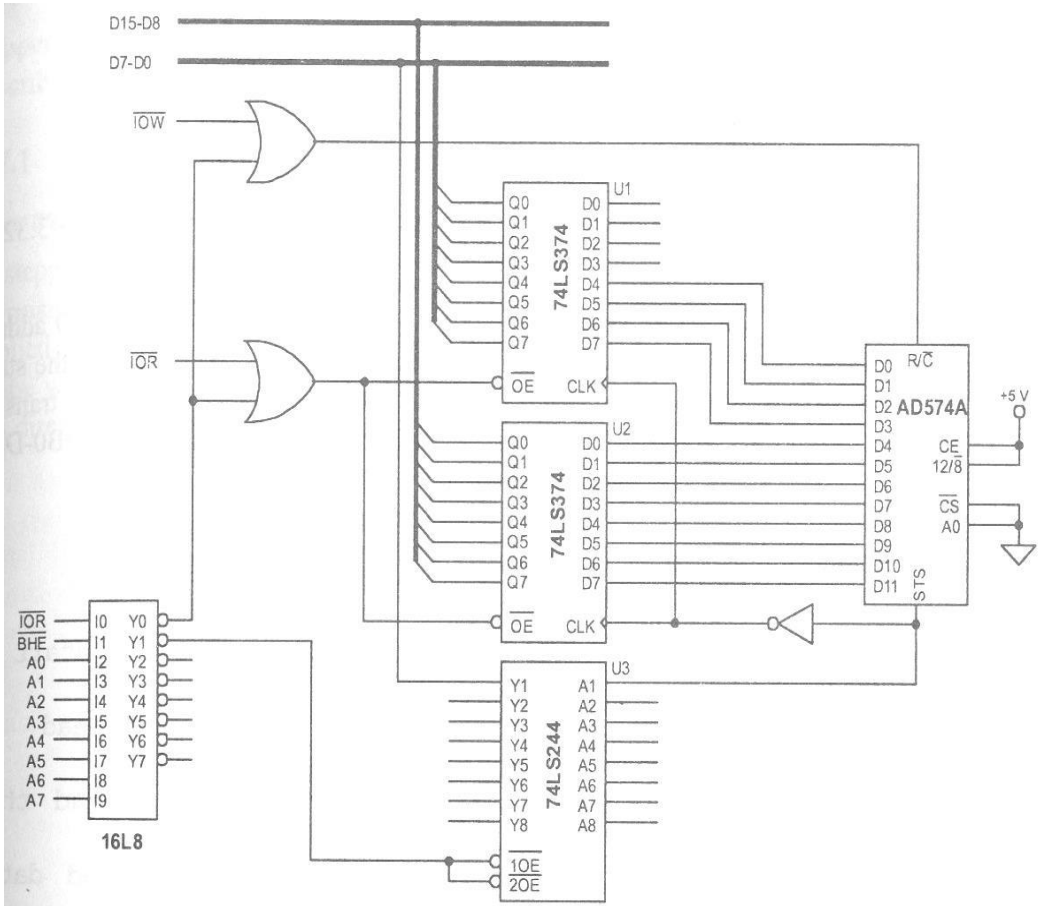
INPUTS		OUTPUT
$1\overline{G}$, $2\overline{G}$	D	
L	L	L
L	H	H
H	X	(Z)



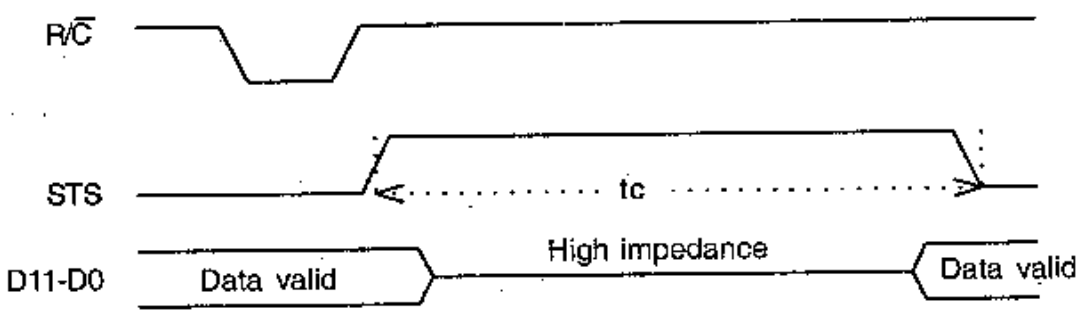
1. Stand-alone configuration

```
ST_C        EQU     40h
DATA_P      equ     40h
STATUS_P    EQU     42H
SAMPLES     DW      4000   DUP(0)
……................
            mov     cx,4000    ; 4000
            mov     si,offset SAMPLES
next:       out     ST_C,ax
Et1:        in      al,STATUS_P
            ror     al,1       ;D0>>C
            jc      et1        ;STS=1, et1
            in      ax,DATA_P  ;STS=0
            push    cx
            mov     cl,4
            shr     ax,cl
            mov     [si],ax
            inc     si
            inc     si
            pop     cx
            loop    next
            ……
```

AH		AL	
Nibble H	NibbleM	NibbleL	X



(a)

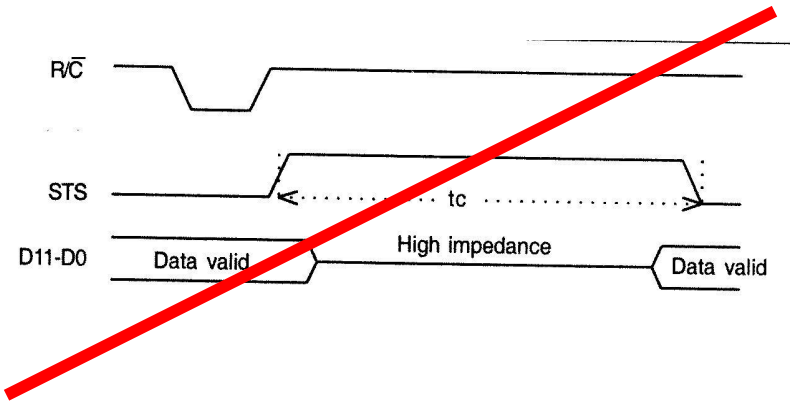


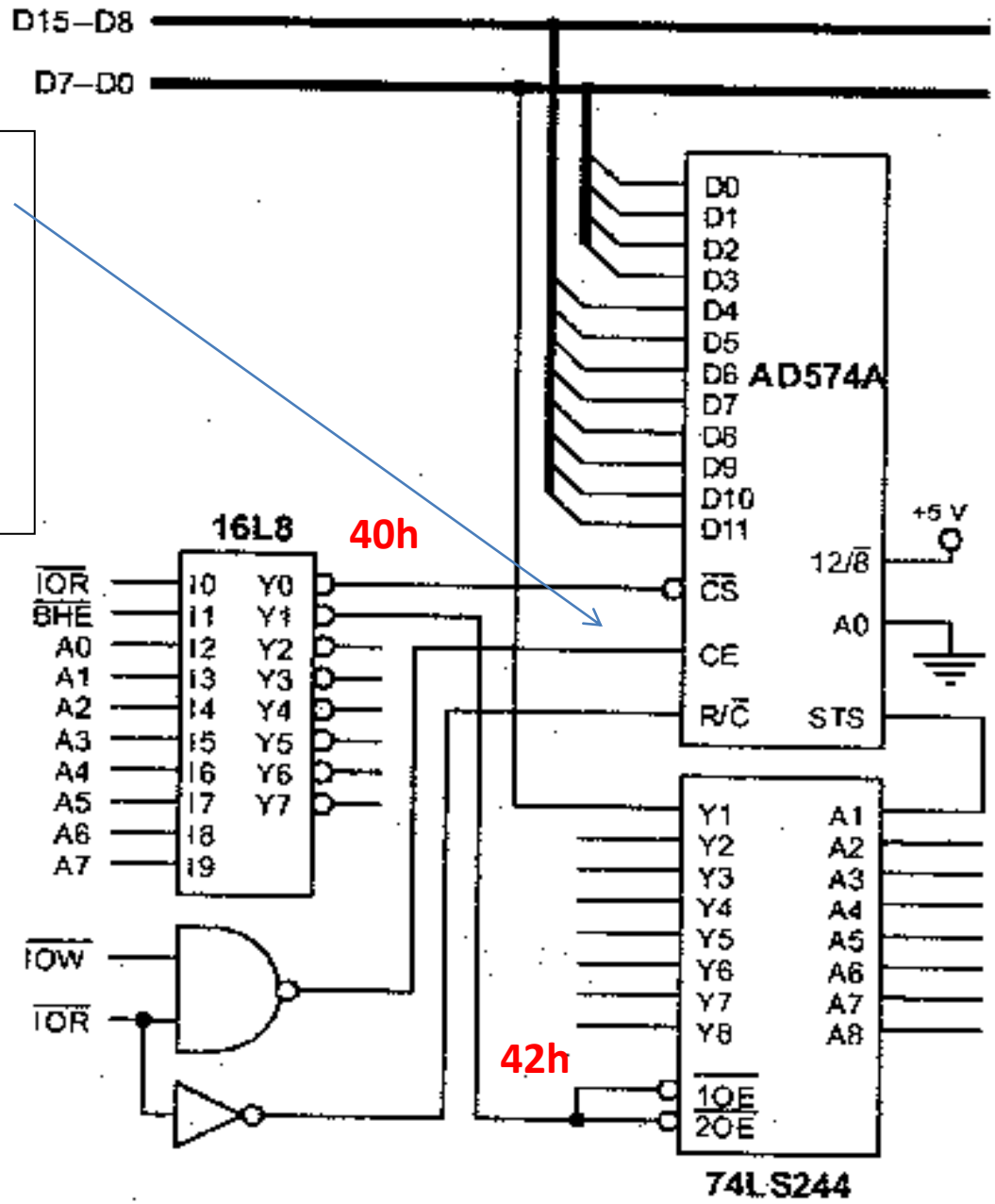
CE	$\overline{CS}$	R/$\overline{C}$	12/$\overline{8}$	A_0	OPERATION
↑	0	0	X	0	Initiate 12-bit conversion

START_Conversie - OUT 40h,AX
/CS=0, A0=0, CE=IOW=L^H, R/C=//IOR=0

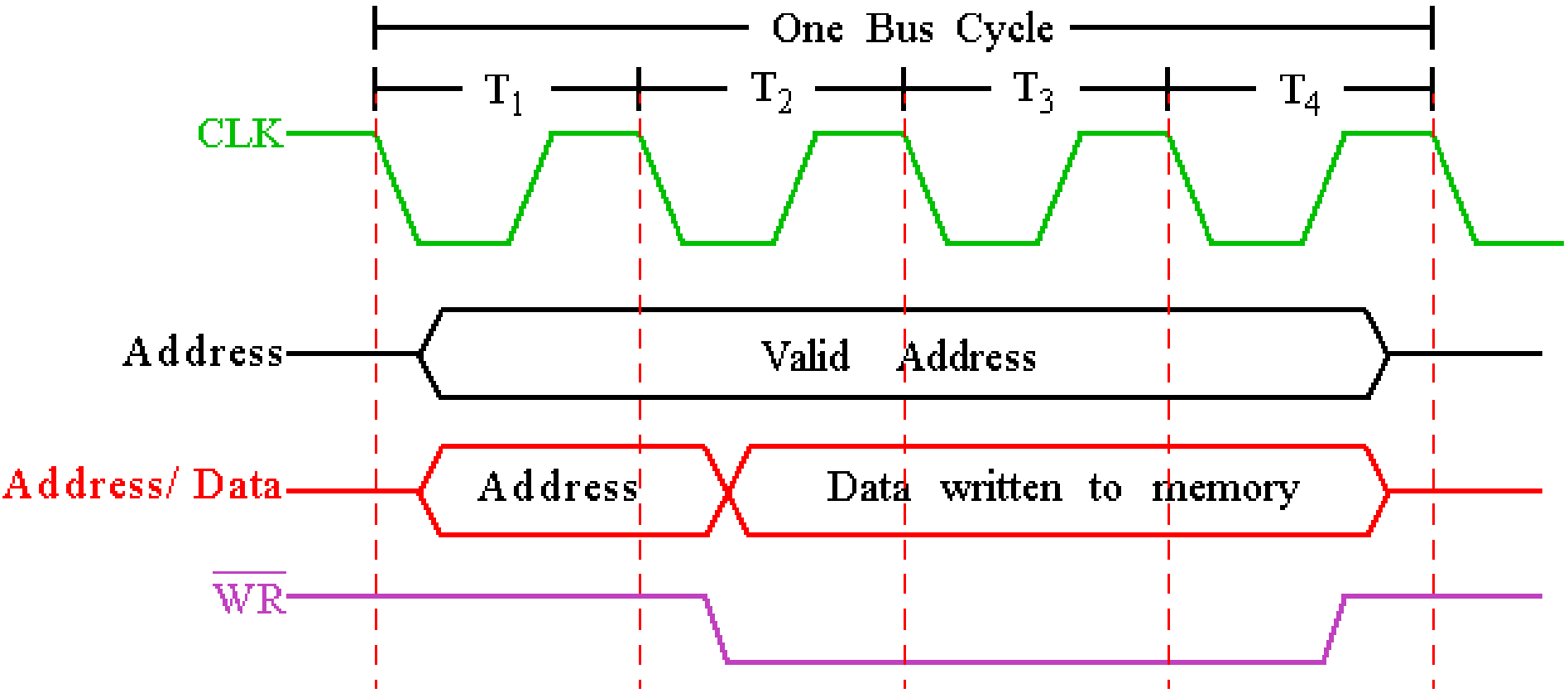
Read IN AX, 40h
/CS=0, R/C=//IOR=1, CE=1, A0=0, IOR=0

see slide 8.

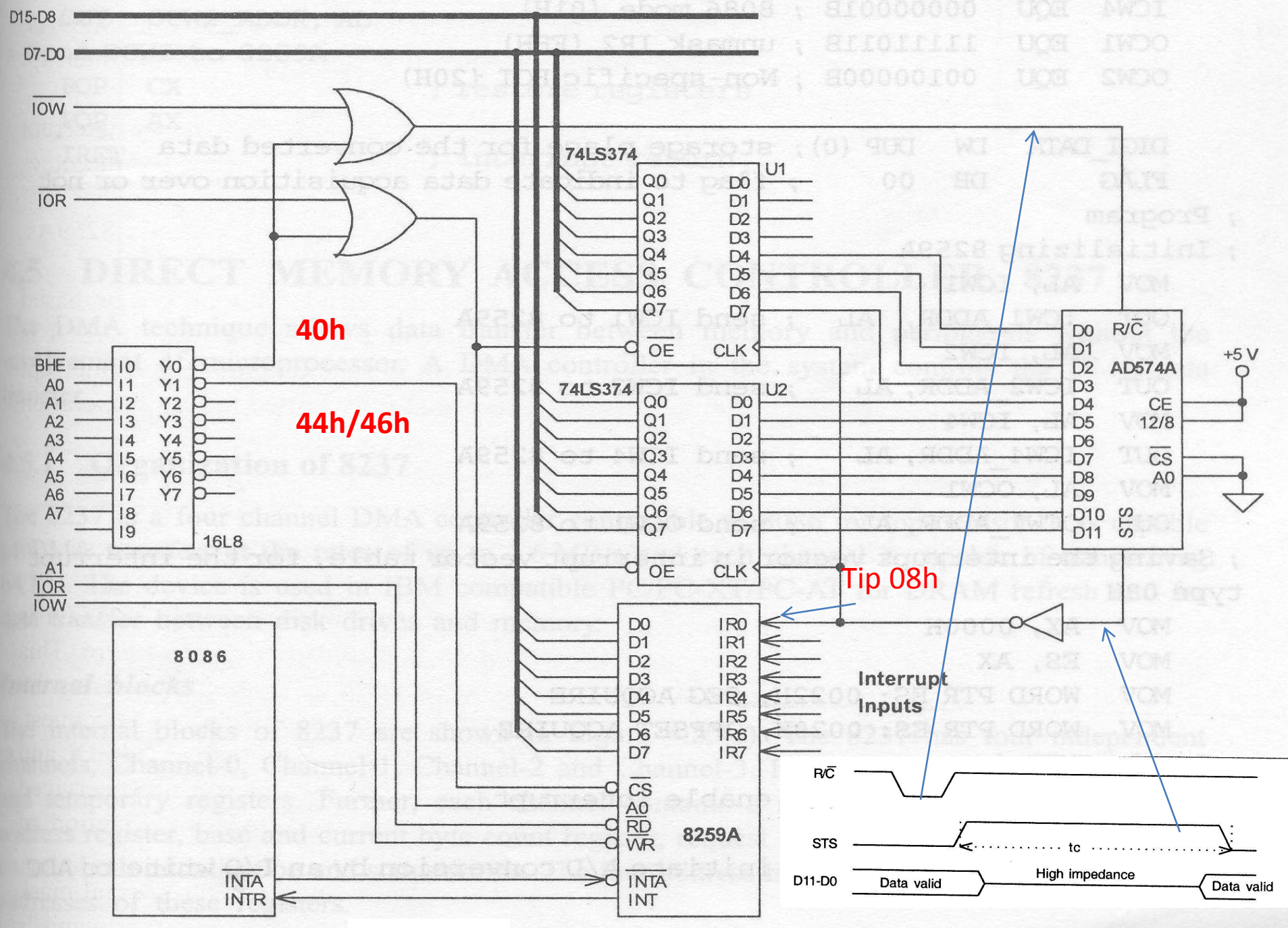




2. Direct connection



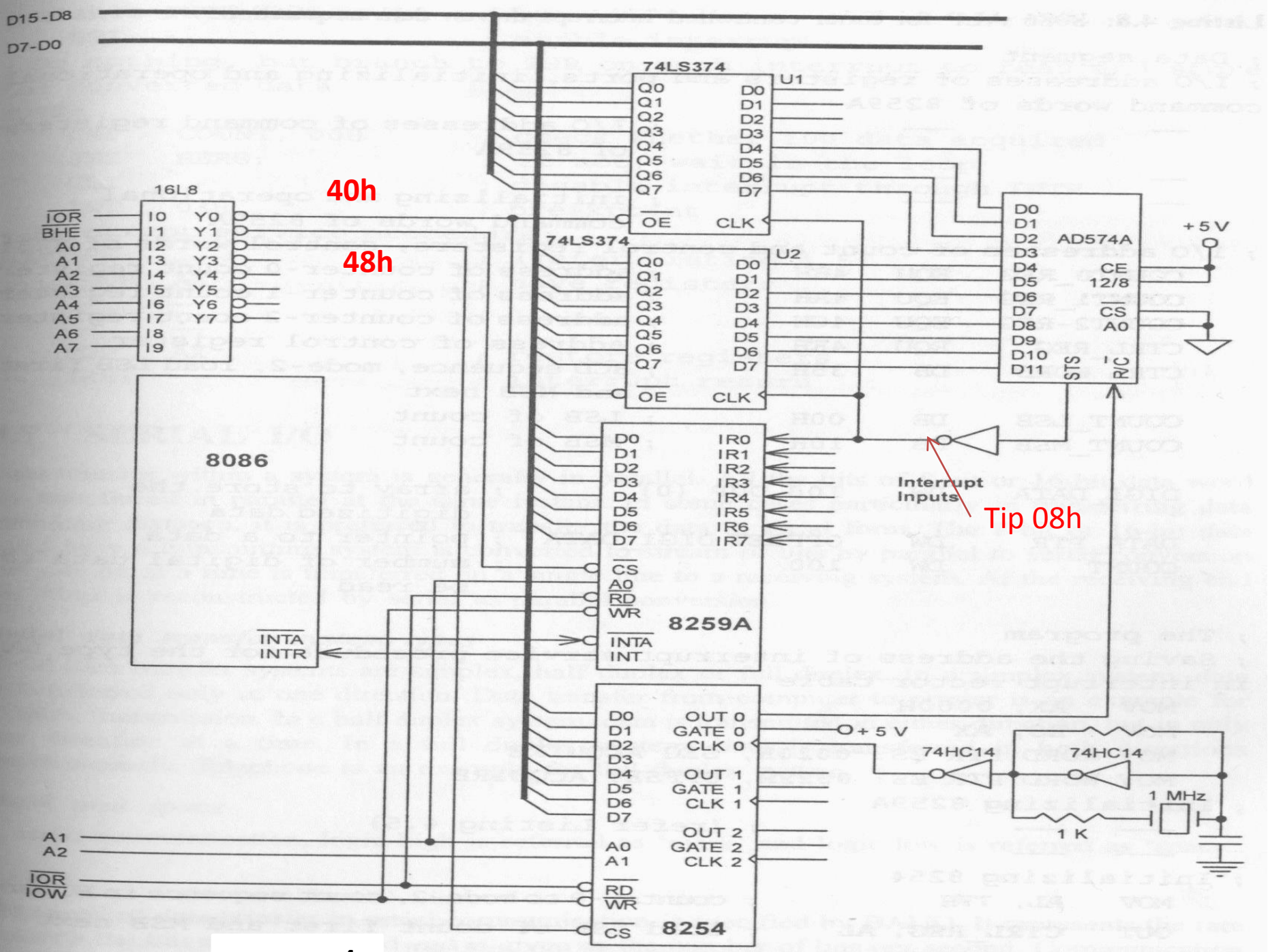
Simplified 8086 Write Bus Cycle



3. Interrupt driven data acquisition.

```
ST_C        EQU        40h
DATA_P      equ        40h
STATUS_P EQU           42H
SAMPLES DW             4000   DUP(0)
FLAG        DB         0
……................
;initializing 8259A
……................
;saving interrupt vector in IVT type 08h
            mov ax,0
            mov es,ax
            mov wordptr es:22h ; seg ACQ
            mov wordptr es:20h ; offset ACQ
            sti        ;IF=1
            mov cx,4000
            mov si,offset SAMPLES
Et0:        out ST_C,ax   ;start conversion
Et1:        cmp  FLAG,0
            jz  et1
            mov FLAG,0
            add  si,2
            Loop et0
……................
```

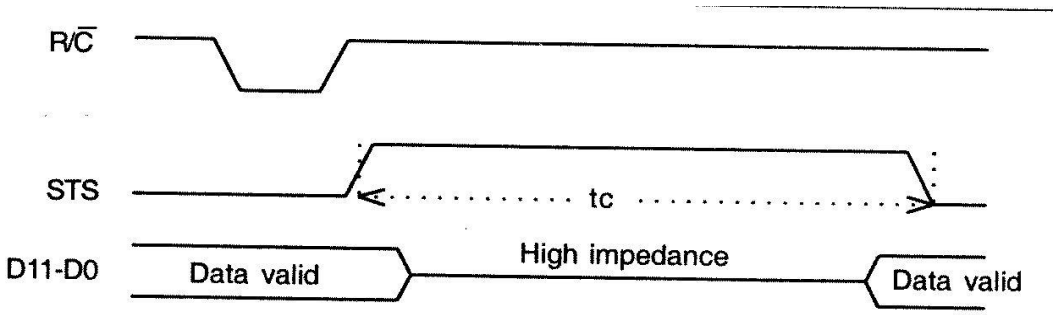
```
ACQ PROC FAR
      Push ax
      Push cx
      In ax,DATA_P
      Mov cx,4
      Shr ax,cl
      Mov [si],ax
      Inc FLAG
      Sti    ;IF=1
      ;EOI if necesary
      Pop cx
      Pop ax
      IRET
ACQ   ENDP
```



4. Timer controlled interrupt driven data acquisition.

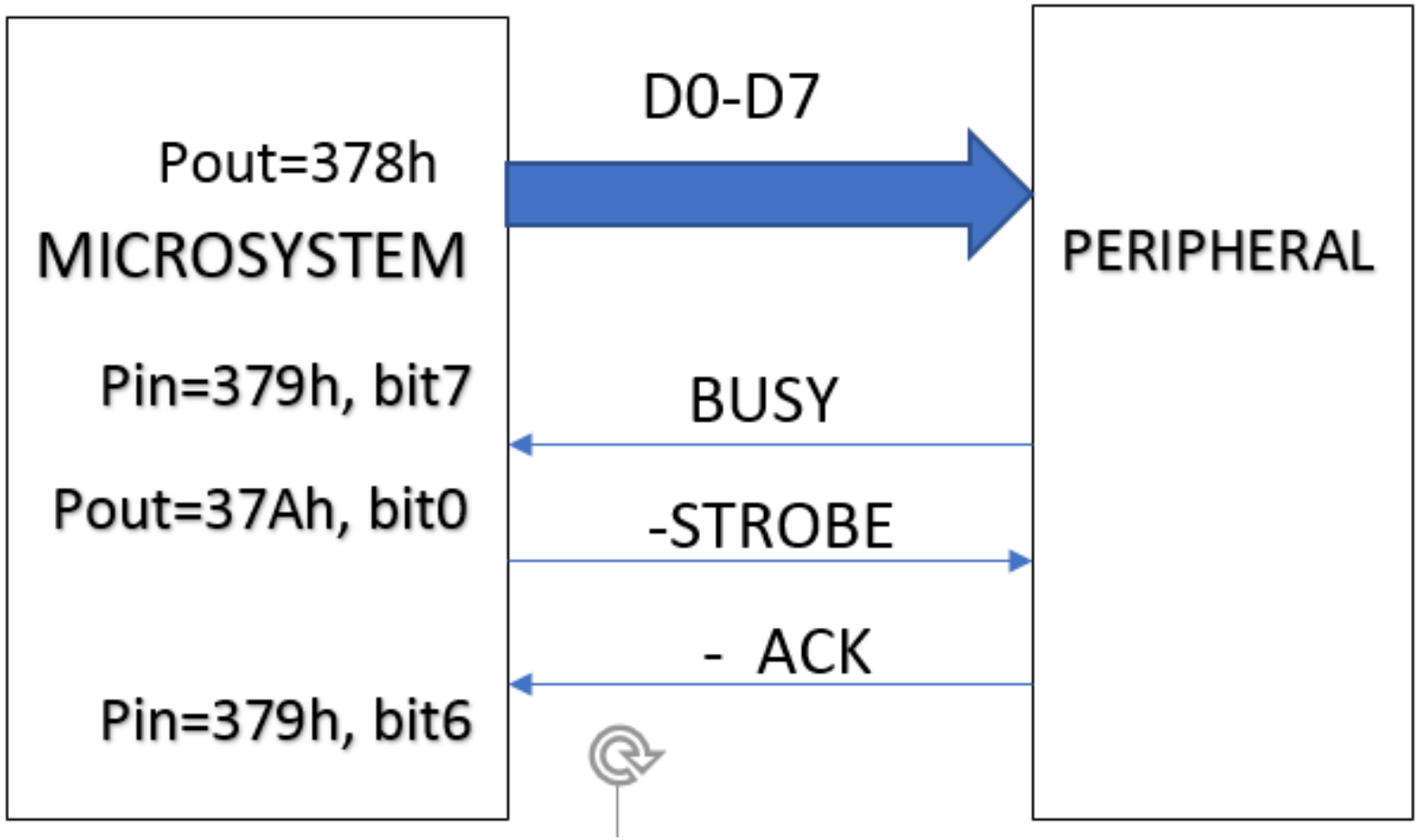
```
C0          EQU     48h
CTRL        EQU     4Eh
CTRLW       EQU     35h; 00110101b
CDIV        DW      1000
DATA_P      equ     40h
SAMPLES     DW      4000   DUP(0)
COUNT       DW      4000
……................
;initializing 8259A
;saving interrupt vector in IVT type 08h
            mov ax,0
            mov es,ax
            mov wordptr es:22h ; seg ACQ
            mov wordptr es:20h ; offset ACQ
            mov al, CTRLW       ;TIMER
            out  CTRL,al
            mov  ax,CDIV
            out C0,al
            mov al,ah
            out C0,al
            sti         ;IF=1
            mov si,offset SAMPLES
et0:        cmp COUNT,0
            jnz et0
            cli   ……................
```

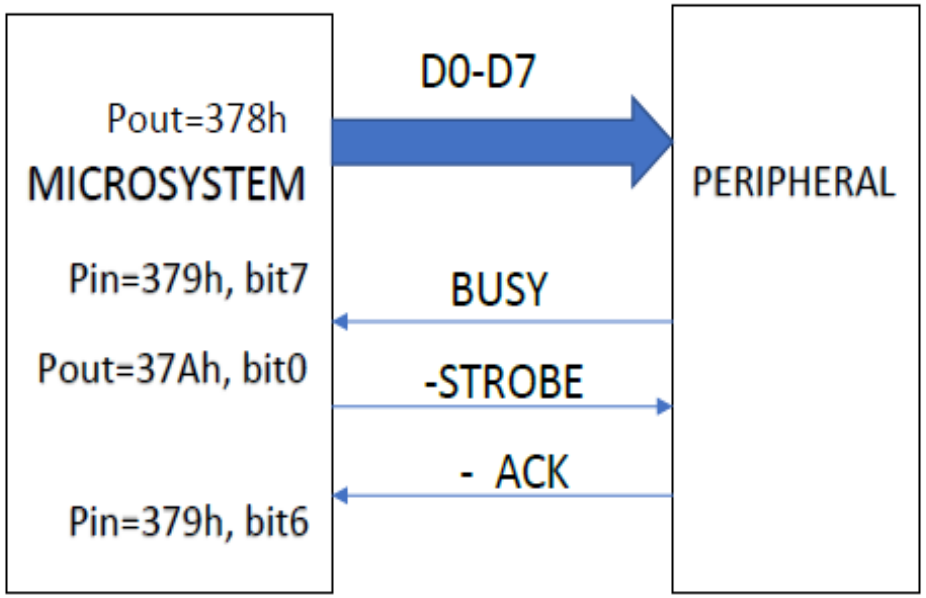
```
ACQ PROC FAR
        Push ax
        In ax,DATA_P
        Mov cx,4
        Shr ax,cl
        Mov [si],ax
        Dec COUNT
        Inc si
        Inc si
        Sti    ;IF=1
        ;EOI if necesary
        Pop ax
        IRET
ACQ   ENDP
```

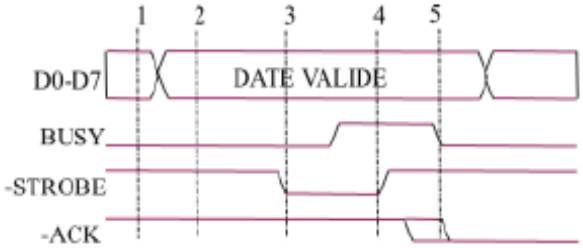


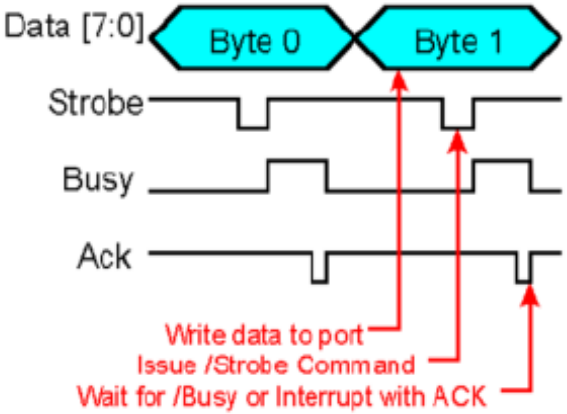
5. Modify the sheet in order to make the sampling acquisition by DMA. Write the program sequences which allow the hardware to work. (programing DMAC, start conversion, ...)

6. A peripheral is connected to a microsystem with a processor on the SPP parallel port as in the figure below. The data exchange between the peripheral and the system is done according to the Centronics protocol, see the diagram. It is required to write the program sequence that transfers 100 bytes of data from: BUF DB 100 DUP (?).









1. The **Busy** signal in the status register is checked
2. The data register is written to port (if BUSY=0)
3. If **Busy** is not active, the signal **-Strobe** in the control register is asserted
4. If **Busy** is active, the **-Strobe** signal is deactivated.
5. When **–Ack** signal becomes active, the receiving of the byte by the printer is confirmed to the PC.

```
BUF   DB   100DUP(?)
PDATA EQU  378H
PCON EQU   37AH
PSTARE EQU 379H
..........................
MOV SI,OFFSET BUF
MOV CX,100
MOV AL,1
OUT PCON,AL  ;STROBE=1
ET1: IN AL,PSTARE
TEST AL,80H     ;BUSY=0?
JNZ ET1
MOV AL,[SI]  ; DATE>>PORT
OUT PDATA,AL
XOR AL,AL
OUT PCON,AL ; STROBE=0
ET2: IN AL,PSTARE
TEST AL,80H ; BUSY=0?
JZ ET2
MOV AL,1  ;  if BUSY=1
OUT PCON, AL ;STROBE=1
ET3: IN AL,PSTARE ;
TEST AL,40H ; ACK=0?
JNZ ET3   ; IF ACK=1 Jump
ET4: IN AL, PSTARE ;test BUSY=1
TEST AL,80H
JZ ET4  ; ; IF BUSY=1 Jump
INC SI
LOOP ET1
.......................
```