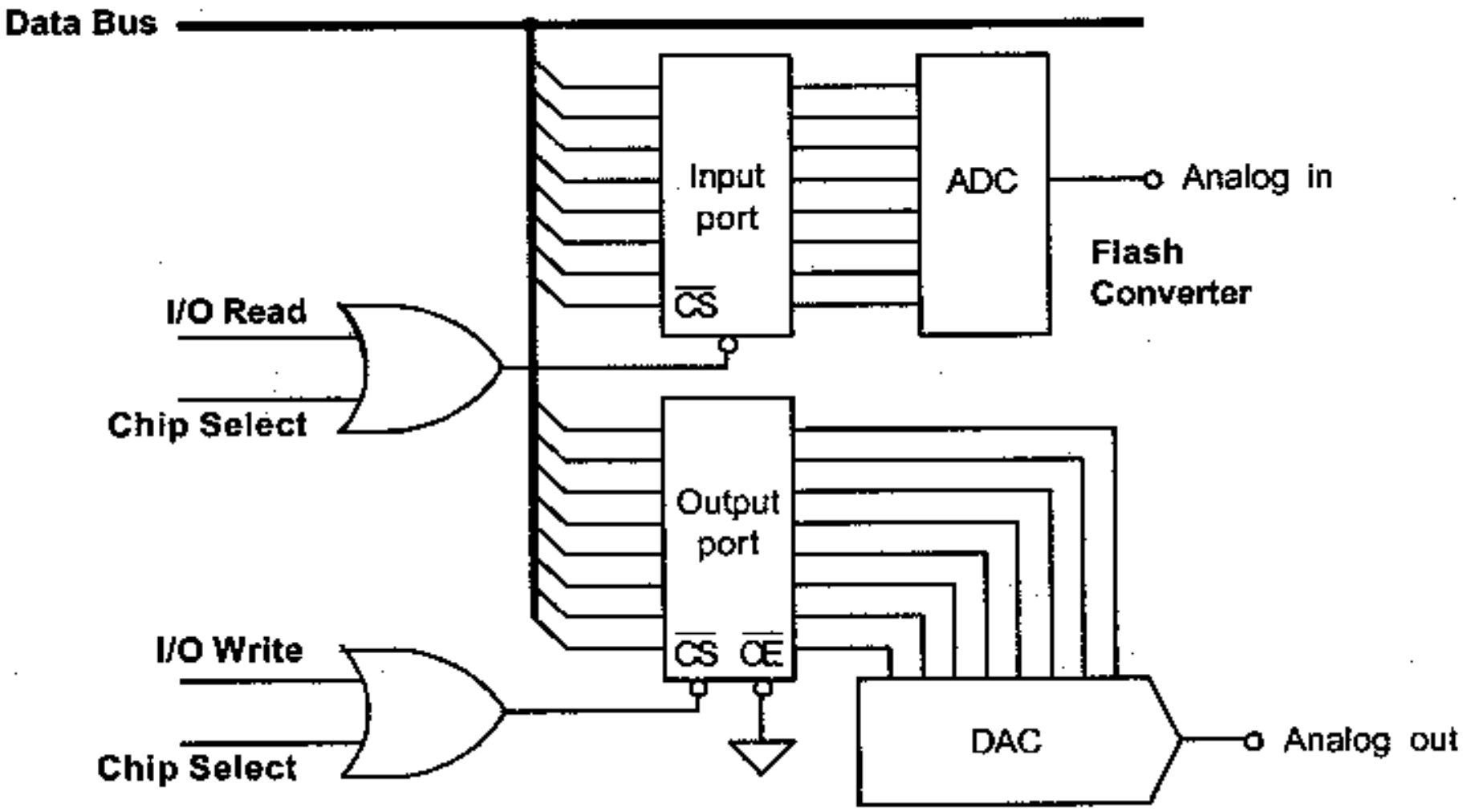
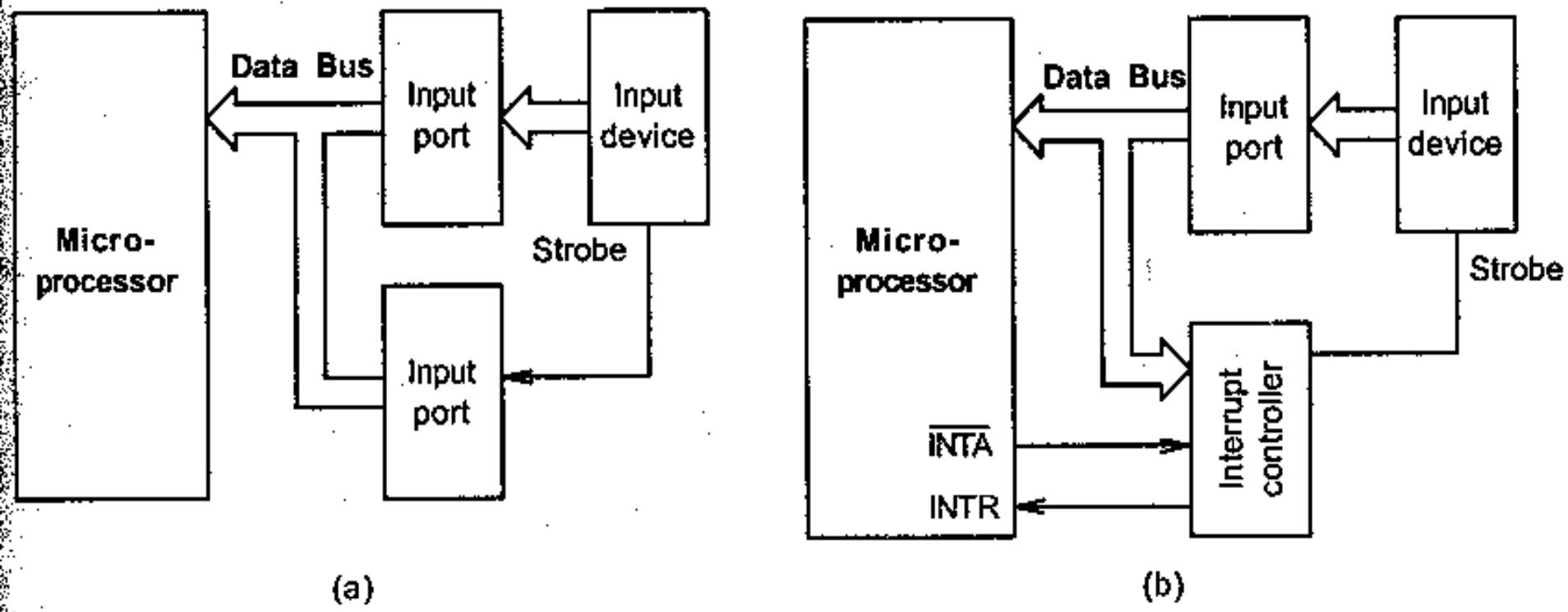
Applications

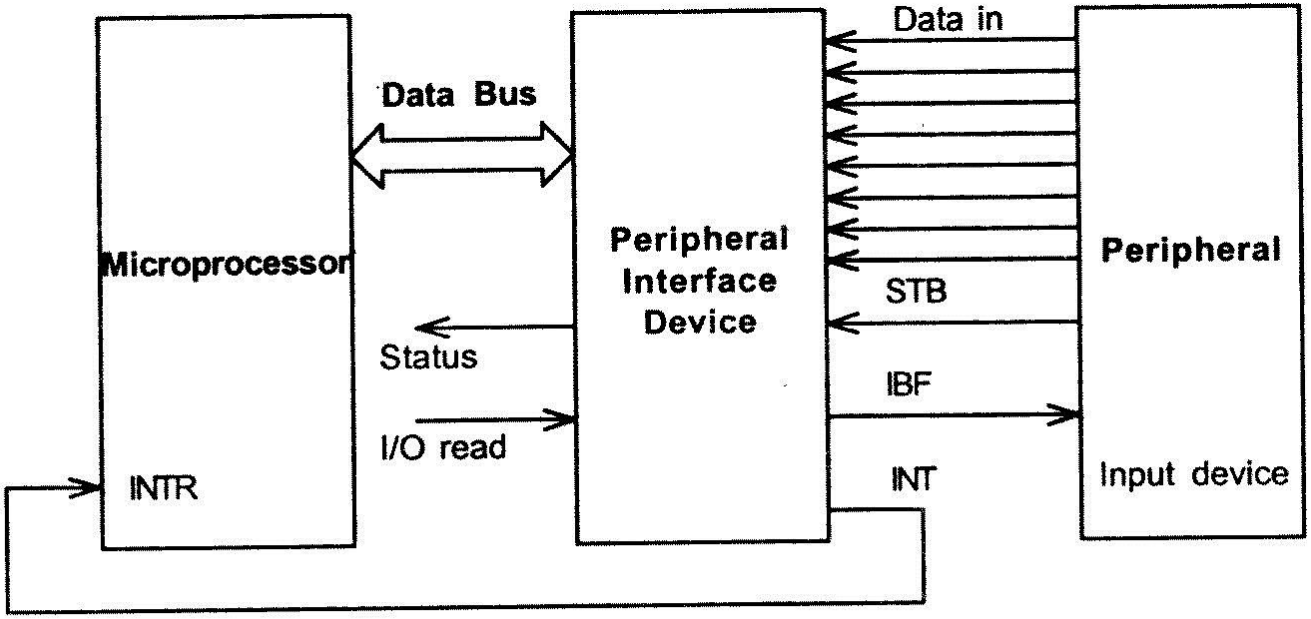
ADC interfacing to Microprocessor



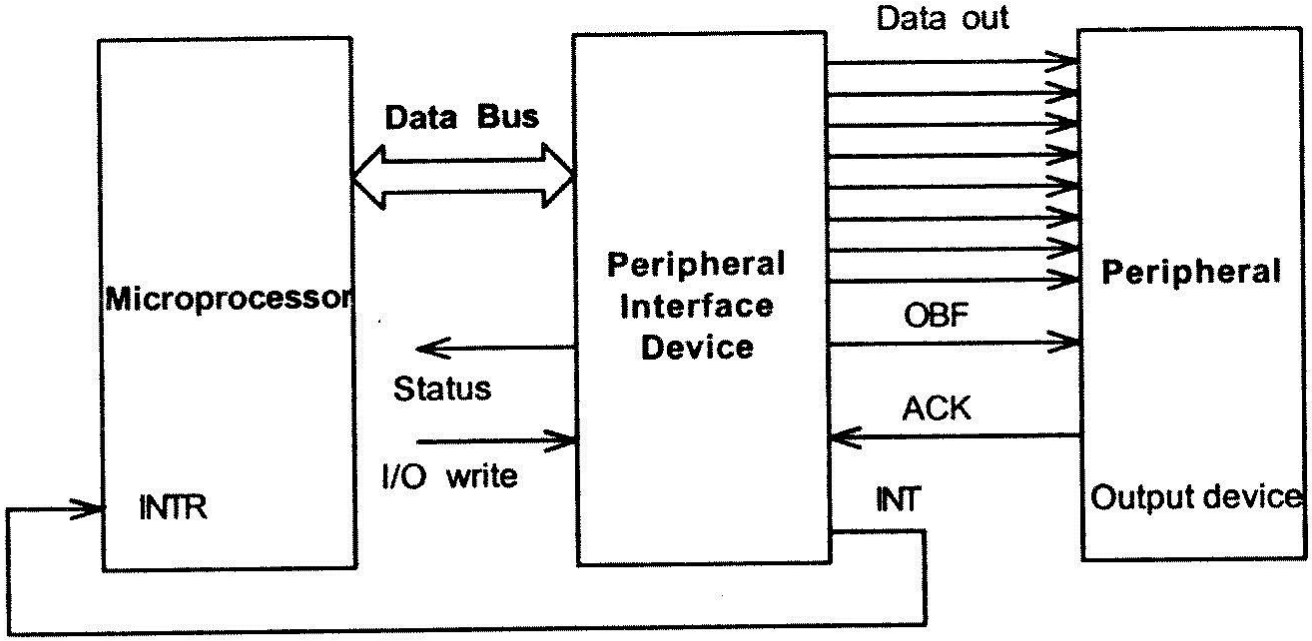
Data transfer by simple I/O.



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



(a)



(b)

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

ADConverter - AD 574A

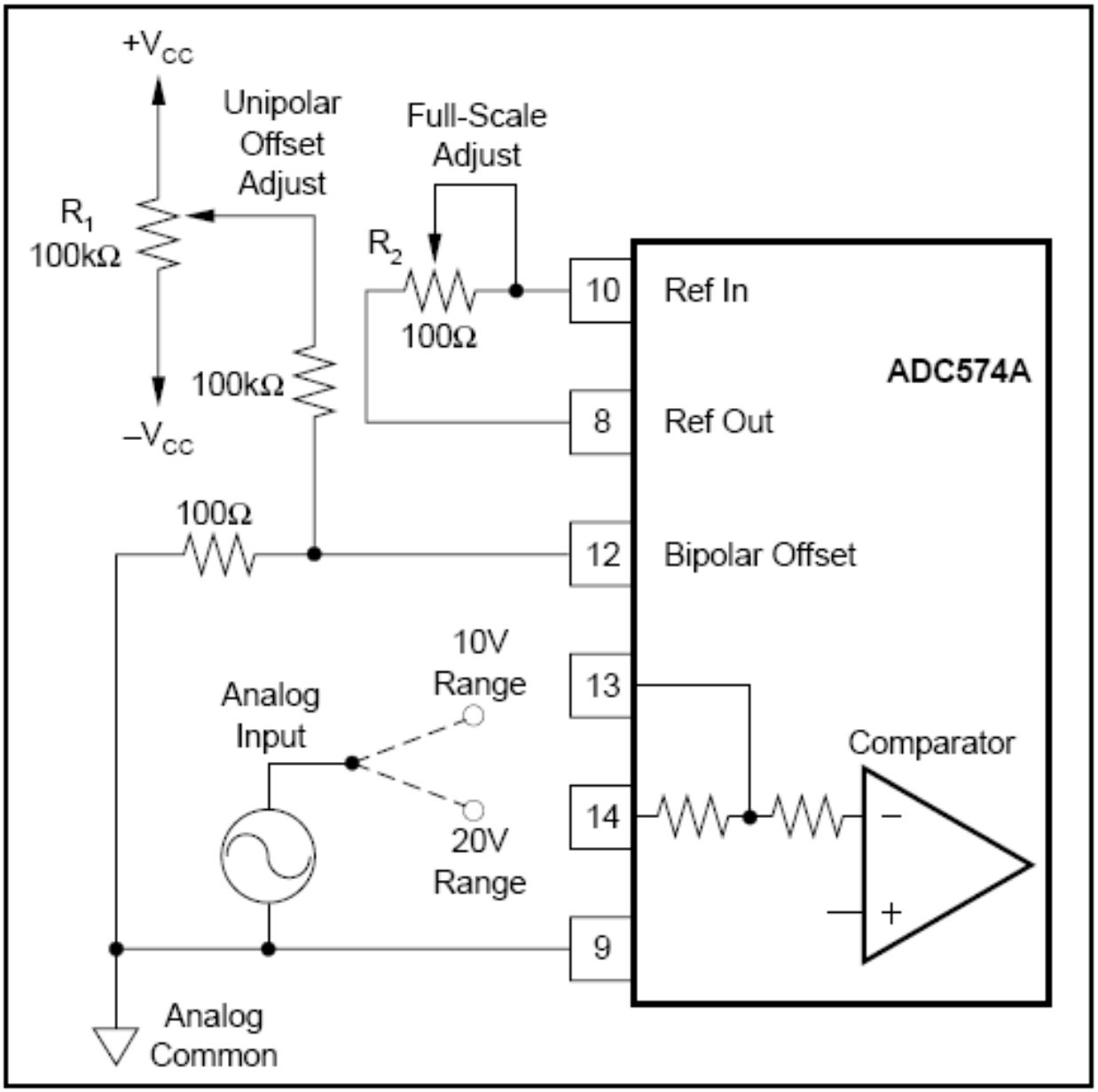


FIGURE 2. Unipolar Configuration.

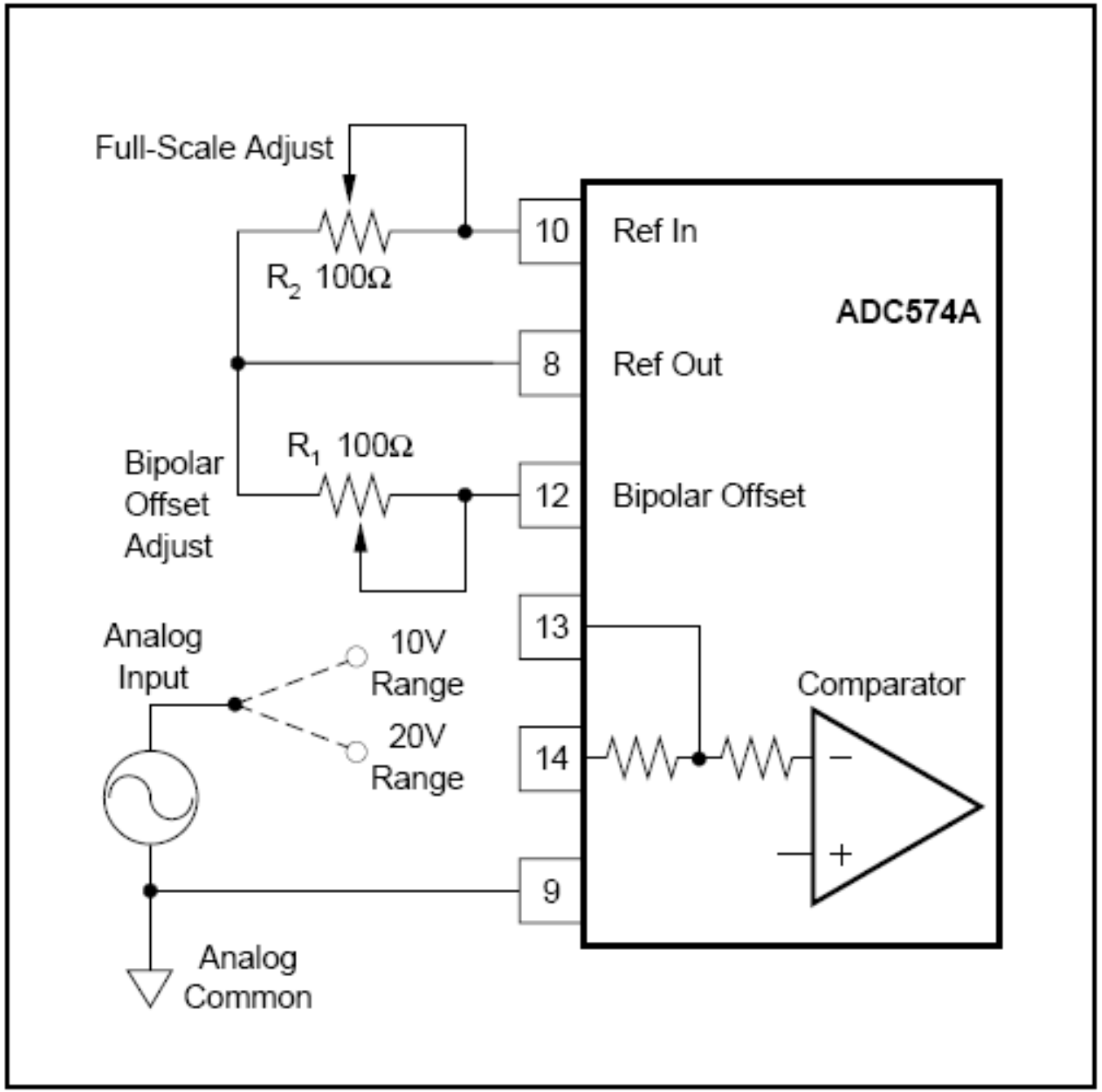


FIGURE 3. Bipolar Configuration.

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	2
↑	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
1	0	↓	X	1	Initiate 8-bit conversion	
1	0	1	1	X	Enable 12-bit output	2
1	0	1	0	0	Enable 8 MSBs only	
1	0	1	0	1	Enable 4 LSBs plus 4 trailing zeros	

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

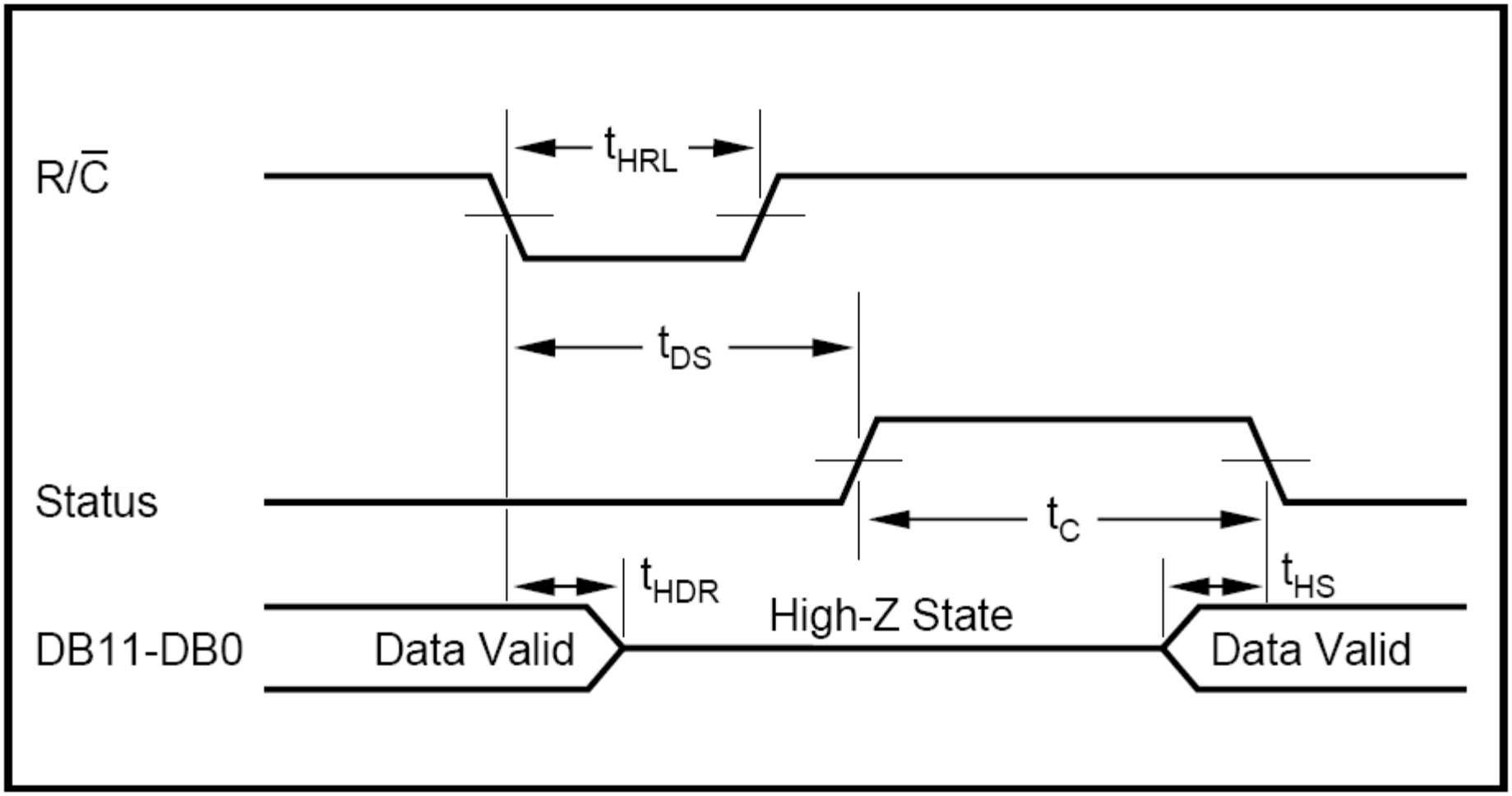
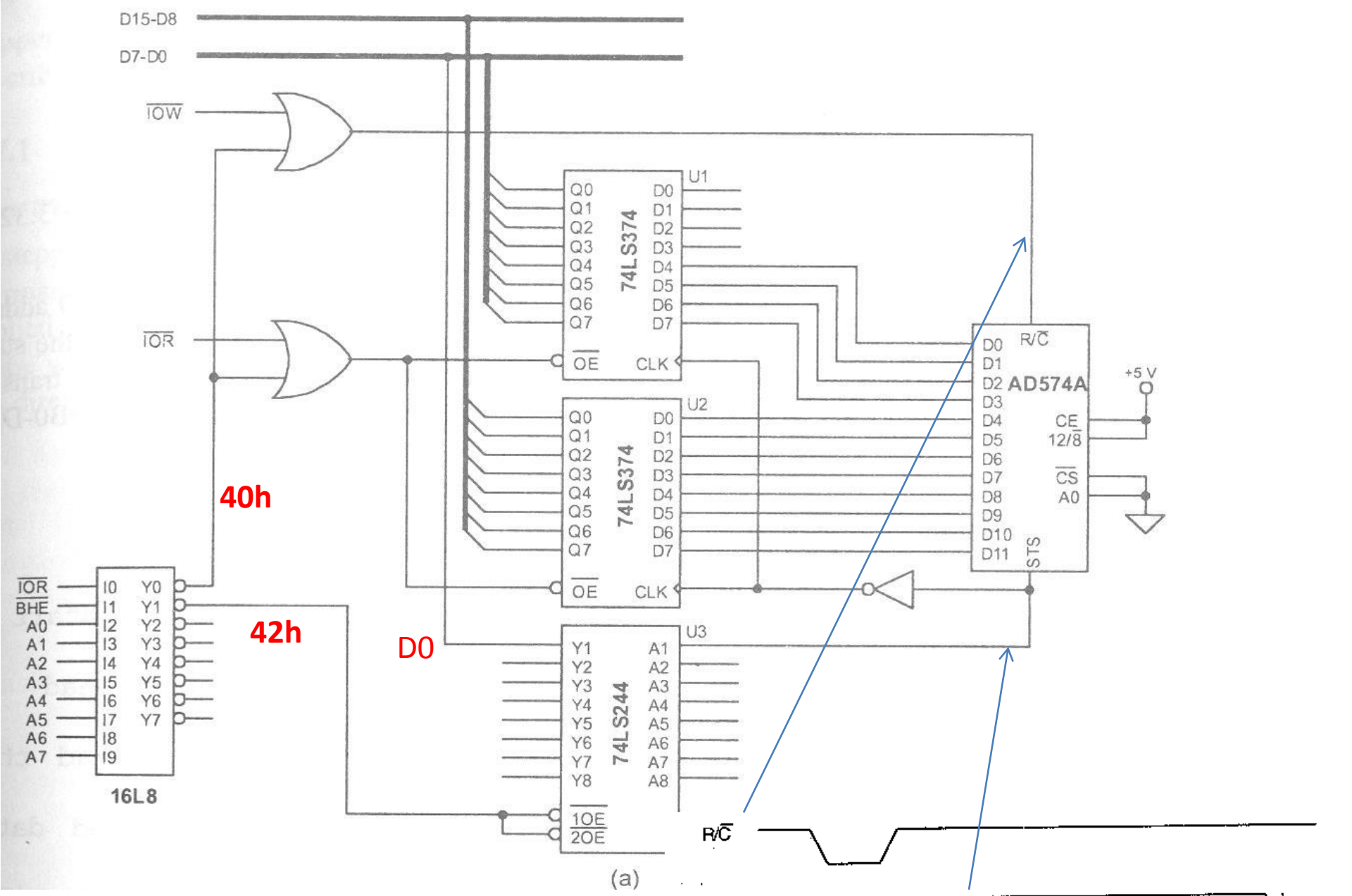
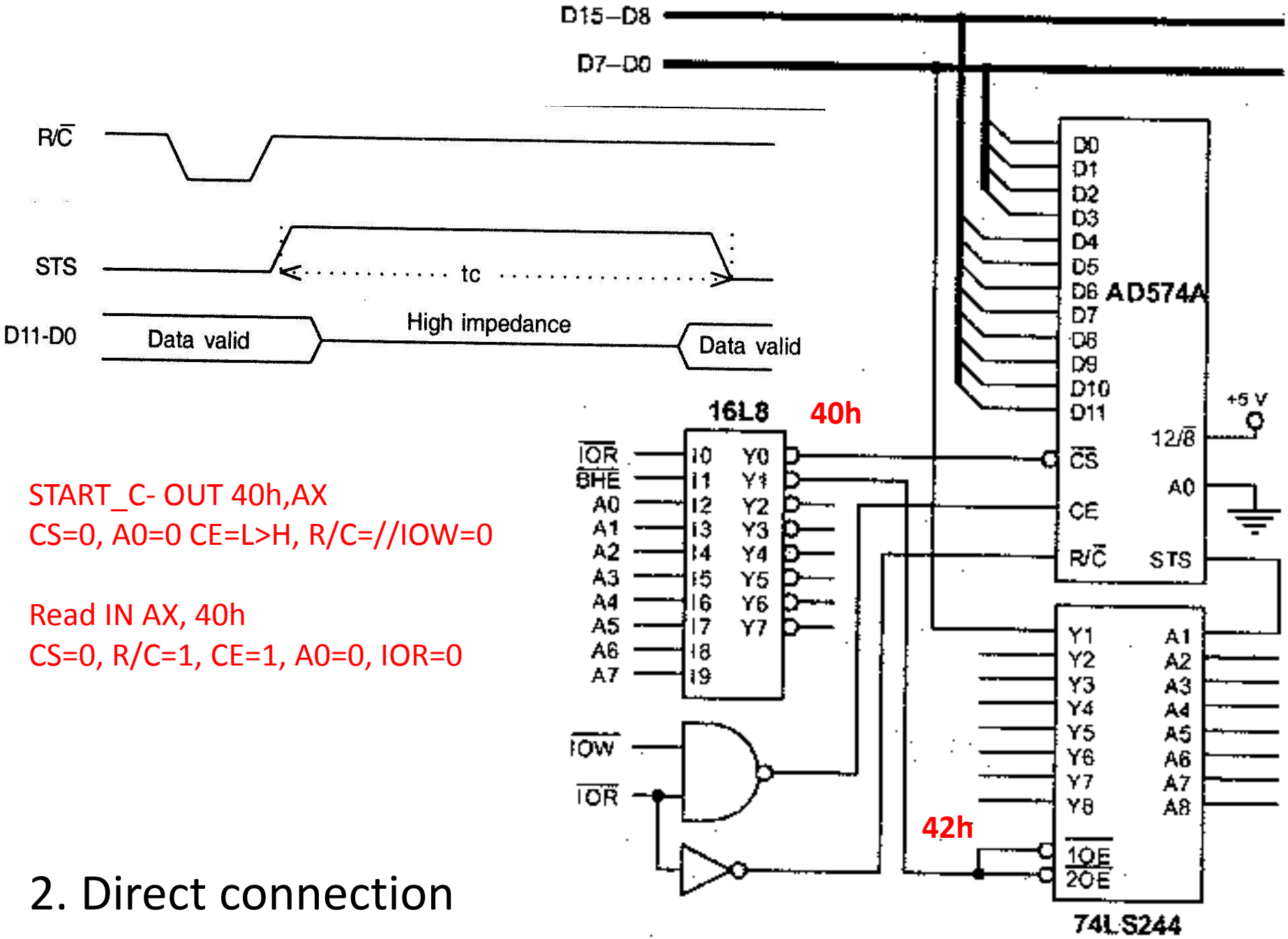


FIGURE 4. R/$\overline{C}$ Pulse Low—Outputs Enabled After Conversion.



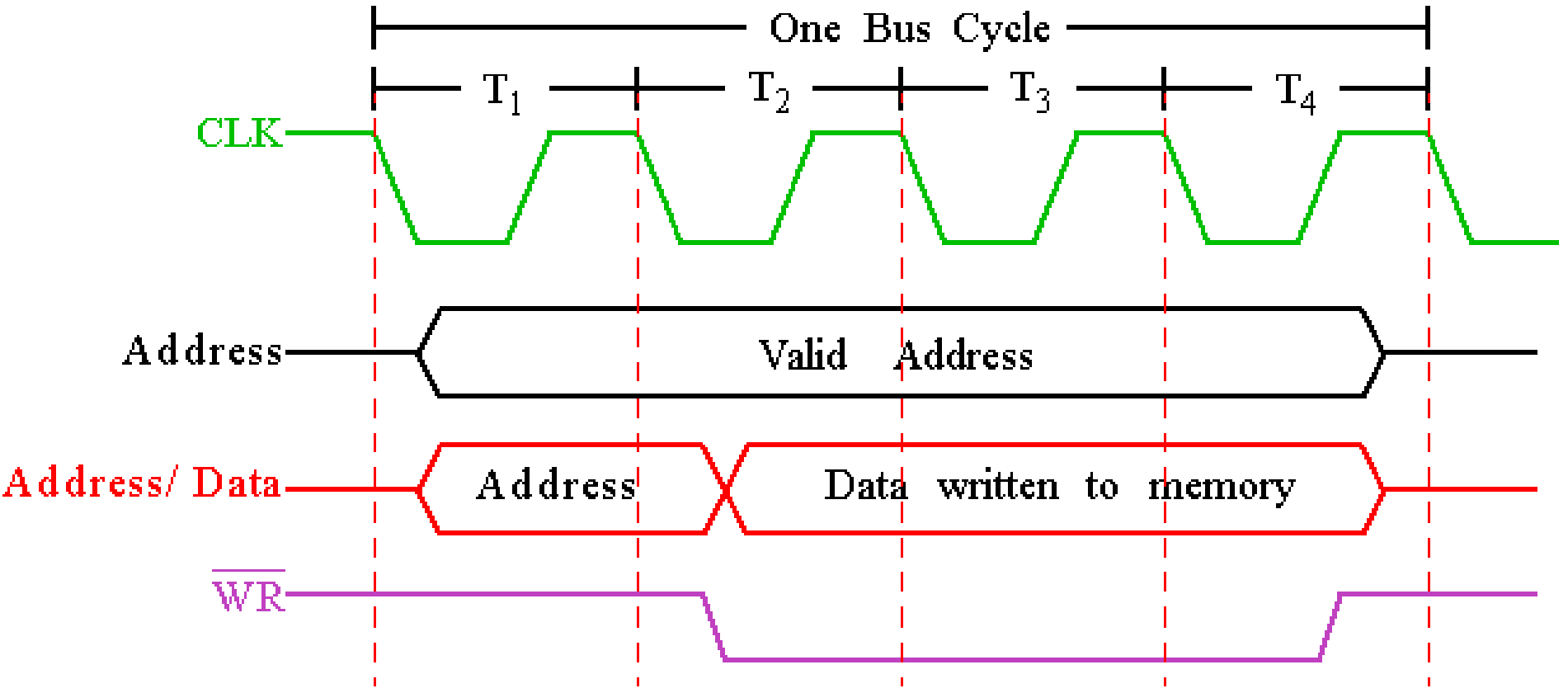
1. Stand-alone configuration



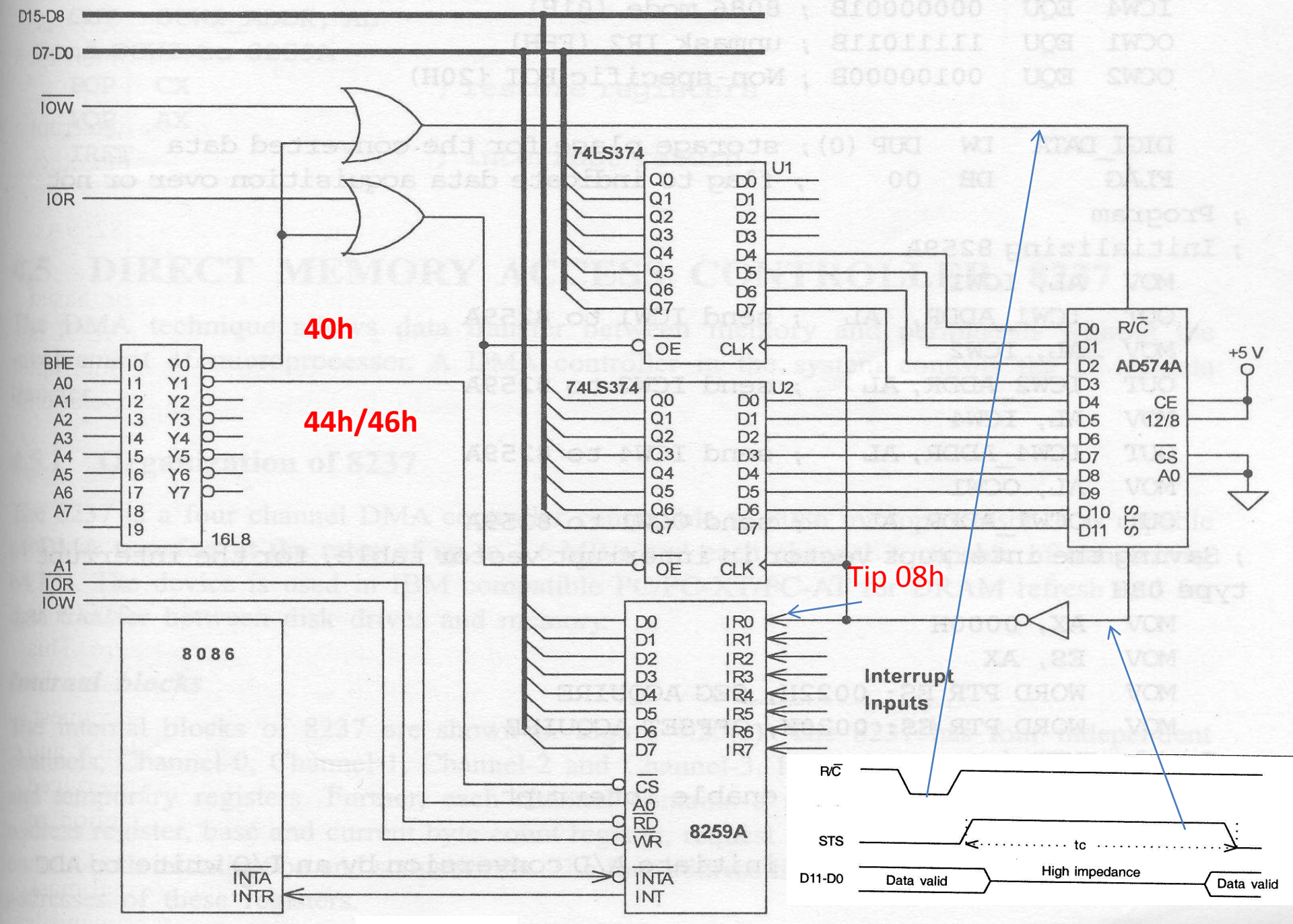
START_C- OUT 40h,AX
CS=0, A0=0 CE=L>H, R/C=//IOW=0

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

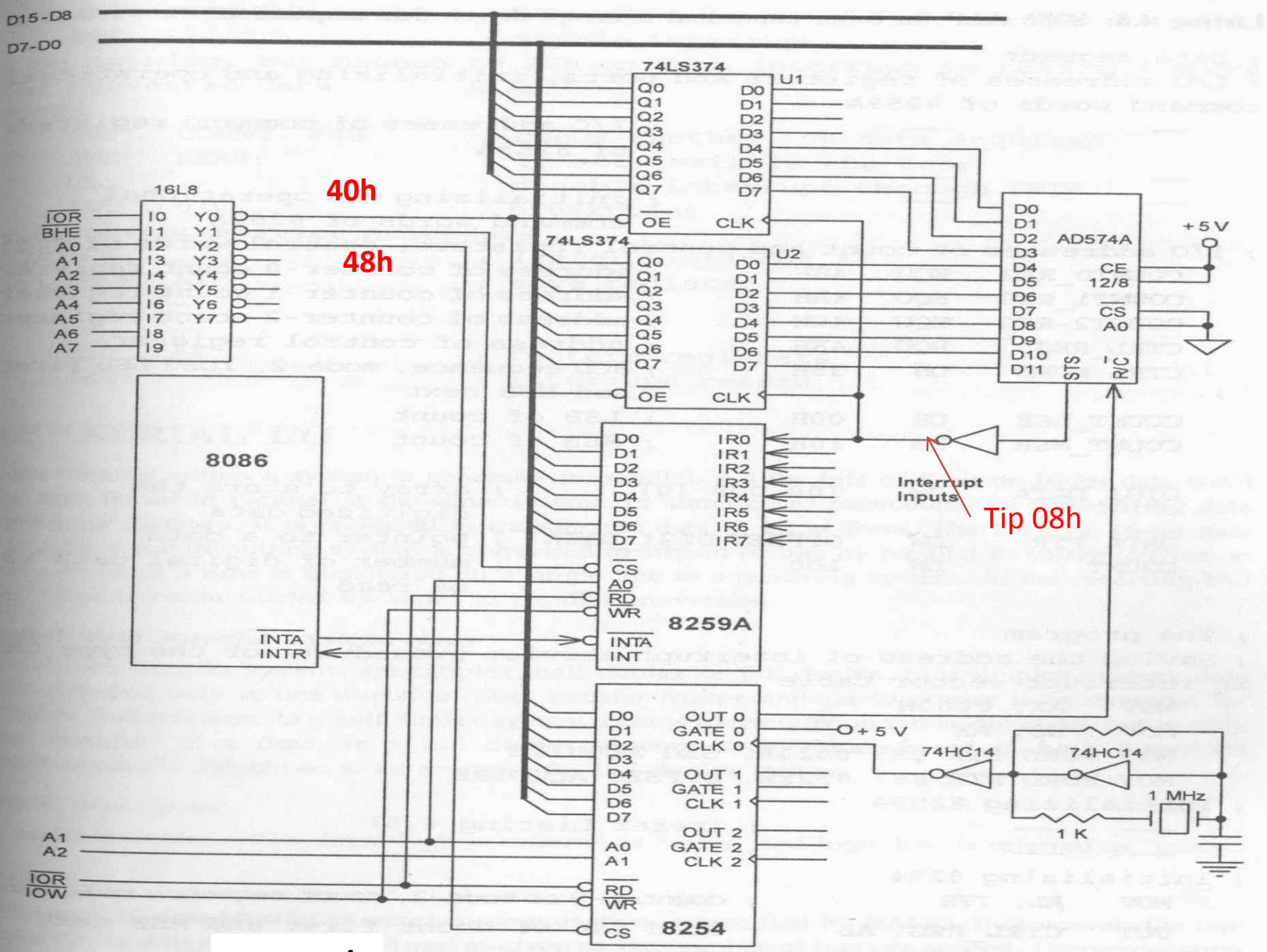
2. Direct connection



Simplified 8086 Write Bus Cycle



3. Interrupt driven data acquisition.



4. Timer controlled interrupt driven data acquisition.

5. Modificati schema astfel ca achizitia esantioanelor sa se faca prin DMA.
Scrieti secventele de program care stau la baza functionarii schemei (programare DMAC, generare start conversie, ...)

Q.1 *Explain Data Acquisition System with example.*

Ans. : Refer section 13.6.

Q.2 *Interface a typical 12-bit DAC with 8255 and write program to generate triangular waveform of period 10 ms. The CPU runs at 5 MHz clock frequency.*

[May-2005, 10 Marks, May-2007, 10 Marks]

Ans. :

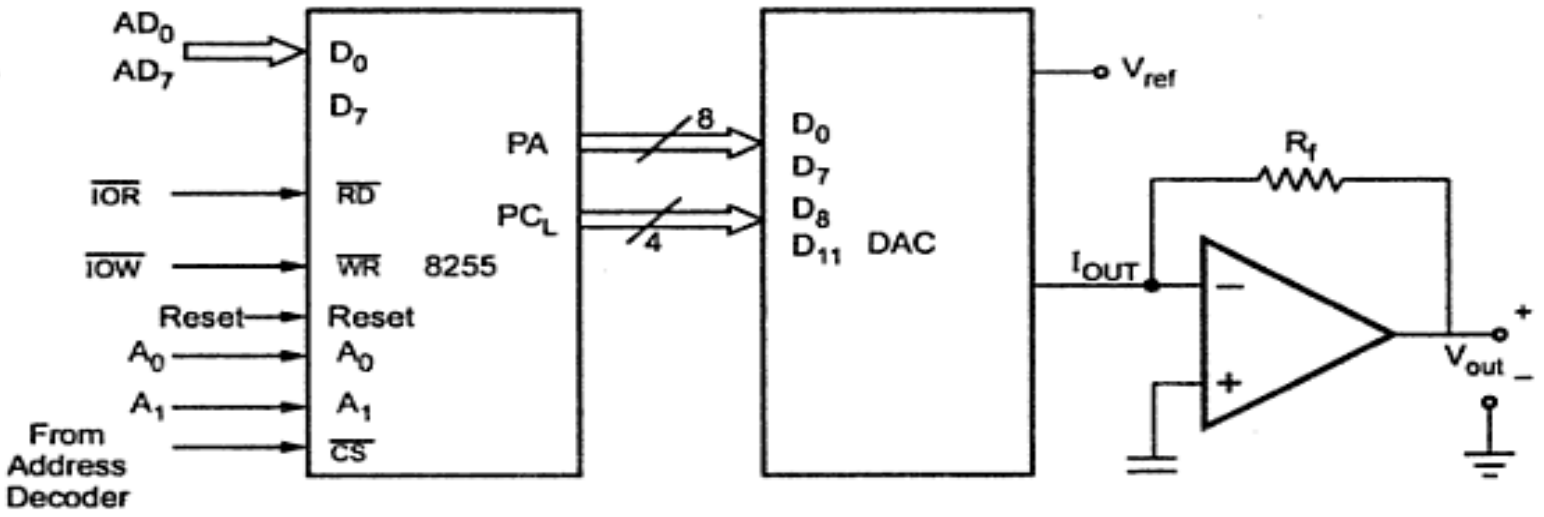


Fig. 1

In triangular waveform period for 1 ramp = 10 ms/2 = 5 ms and minimum time required to transfer data is about 40 μs.

Therefore, steps in one ramp = $\frac{5\,\text{ms}}{8\,\mu\text{s}}$ = 625. So to reach from 0 – 4096 we have 625 steps i.e. we have increment the digital count by 4096/625 = 6.55 ≈ 7

With increment of 7 we get 585 steps and for each data transfer we get time $= \frac{5\,ms}{585} = 8.547\ \mu s.$

The clock frequency of 8086 is given 5 MHz,

$$\therefore \quad \text{Time for one clock cycle} = \frac{1}{5\,MHz} = 0.2\ \mu s$$

$$\text{Therefore, clock cycles in one loop} = \frac{8.547\ \mu s}{0.2\ \mu s} \approx 43$$

Program :

```
START :  MOV CX, 0FFFH
         MOV BX, 0000H        Clock cycles
BACK  :  MOV AL, BL    ...        2
         OUT PA, AL               10
         MOV AL, BH                2
         OUT PC, AL               10
         INC BX                    2
         NOP                       3    Delay Adjustment
         INC DX                    2    Delay Adjustment
         DEC CX                    2
         JNZ BACK                 10
                                  ----
                                  43
```

```
Mov cx,585          4
Mov bx,0            4
Back: Mov al,bl     2
Out PA,al          10
Mov al,bh           2
Out PC, al         10
Add bx,7            4
Mov bx,bx           2
Dec cx              3
Jnz Back           10
                   43
```

```
         MOV CX, 0FFFH
BACK1 :  DEC BX
         MOV AL, BL
         OUT PA, AL
         MOV AL, BH
         OUT PC, AL
         NOP
         INC DX
         DEC CX
         JNZ BACK1
         JMP START
```

```
Mov cx,585          4
Back1: Sub bx,7     4
....
....
Jnz back1
Jmp start
```