Digitális technika II. házi feladat

2013.03.05

Név: **Szabó Norbert**	Gyak: *Horváth Tamás (K2.)*	**236**	Kód: **AJD5YL**
Feladat beadás: 2013.04.30 a gyakorlaton	*Pótbeadás:* 2013.05.22. 9:30 - 10:30 - IB.310		

Beadáskor ezt a feladatlapot a megoldáshoz csatolni kell. A feladatokat külön lapon, kézírással oldja meg. A beadandó anyaghoz útmutatót a tárgy honlapján (www.iit.bme.hu/digit2) talál! Hibás megoldás javítására a pótbeadás alkalmával van lehetőség.

1. Illesszen 8085-ös mikroprocesszor alapú sínre **2764** típusú EPROM és **5565** típusú RAM memóriákat úgy, hogy az alábbi címtartományokat fedjék le:
   1. **0000h-1FFFh** EPROM
   2. **2000h-3FFFh** EPROM
   3. **0000h-1FFFh vagy 8000h-9FFFh*** RAM

   * A 80h I/O címre írt adattal lehessen állítani, hogy a 0000h címen EPROM vagy a RAM memória látszódjon. (Ha a kiírt érték 0 az EPROM, ha 1 a RAM látszik) Gondoskodjon róla, hogy RESET után mindig az EPROM látszódjon! Ha a RAM a 0-ás címtől látszik, legyen írásvédett. Tervezze meg a szükséges IO egységet.

   A sín jelei:

   $SA0...SA15, SD0...SD7, \overline{SMRD}, \overline{SMWR}, \overline{SIORD}, \overline{SIOWR}, SIO/\overline{M}, \overline{SREADY}, SS0, SS1, SClkOut, SresetOut$

   a. Rajzolja fel a memória modul blokkvázlatát. (Figyeljen a jelek konzisztens elnevezésére!)
   b. Rajzolja fel a memóriamodul címtérképét és a címdekóder egységét.
   c. Rajzolja fel az adatbusz meghajtó áramkör-vezérlő logikát.
   d. Adja meg a memória-áramkörök bekötését!
   e. Rajzolja fel a READY logikát a következő paraméterek figyelembevételével:
      a RAM memóriák READY logikája **1** WAIT állapotot,
      az EPROM memóriák READY logikája kizárólag olvasásra **0** WAIT állapotot
      iktasson közbe a műveletvégzés közben!
   f. Tervezze meg a feladatban kért I/O egységet (dekódoló, flip-flop)!

2. Készítse el a következő assembly szubrutint, amellyel a RAM memória tesztelhető.

   a. Írjon ***KITOLT*** szubrutint, amely a **HL** regiszterpárban egy kezdőcímet, a **DE** regiszterpárban egy hossz értéket kap, az így meghatározott memóriablokkot kitölti úgy, hogy minden páratlan címen elhelyezkedő byte az 0AAh, páros címen elhelyezkedő byte az 55h értéket tartalmazza.

   b. Írjon ***ELLENOR*** szubrutint, amely a **HL** regiszterpárban egy kezdőcímet, a **DE** regiszterpárban egy hossz értéket kap és ellenőrzi, hogy a memóriablokk rekeszei a ***KITOLT*** szubrutin által beírt értékeket tartalmazzák-e? A szubrutin **CY=0-val** jelezze, ha hibát talált. Ilyenkor a HL regiszterpár az utolsó (legmagasabb memóriacímű) **megtalált hiba címét**, a BC regiszterpár pedig a hibásnak talált **byte-ok darabszámát** tartalmazza. Ha nincs hiba CY=1, BC=0 és HL a memóriablokk első elemére mutat.

   c. Írjon programrészletet, amely a processzor SID bemenetén fellépő 0→1 átmenet hatására a ***KITOLT*** és ***ELLENOR*** szubrutinok segítségével ellenőrzi a RAM területet. A teszt indulását és befejeződését a SOD kimeneten egy-egy 3 ms idejű impulzussal jelezze. Az időzítés meghatározásánál vegye figyelembe, hogy a program a feladatban meghatározott EPROM memóriában fut! Figyeljen arra, hogy a tesztelés alatt a RAM terület ne legyen írásvédett.

   A szubrutinokat úgy írja meg, hogy a működéshez előírt regisztereken kívül más regiszterek értékét ne rontsák el! A szubrutinokat lássa el megjegyzésekkel és készítsen fejlécet is!

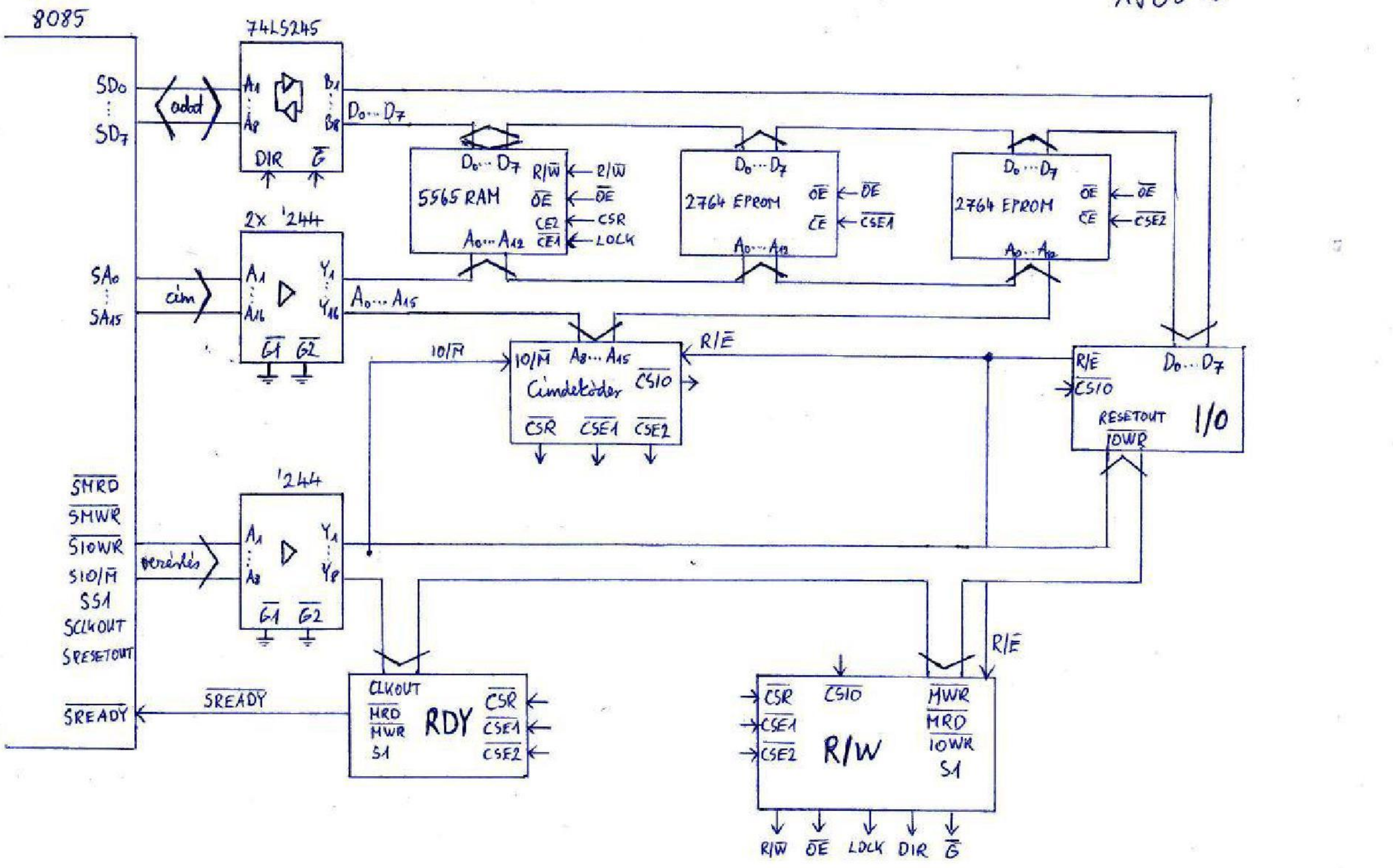
Nyilatkozat:

Alulírott Szabó Norbert nyilatkozom, hogy a házi feladatot meg nem engedett segédeszköz használata nélkül saját magam oldottam meg.

Dátum: 2013. .... Aláírás:.....

SZABÓ NORBERT
AJO5YL

1.a)

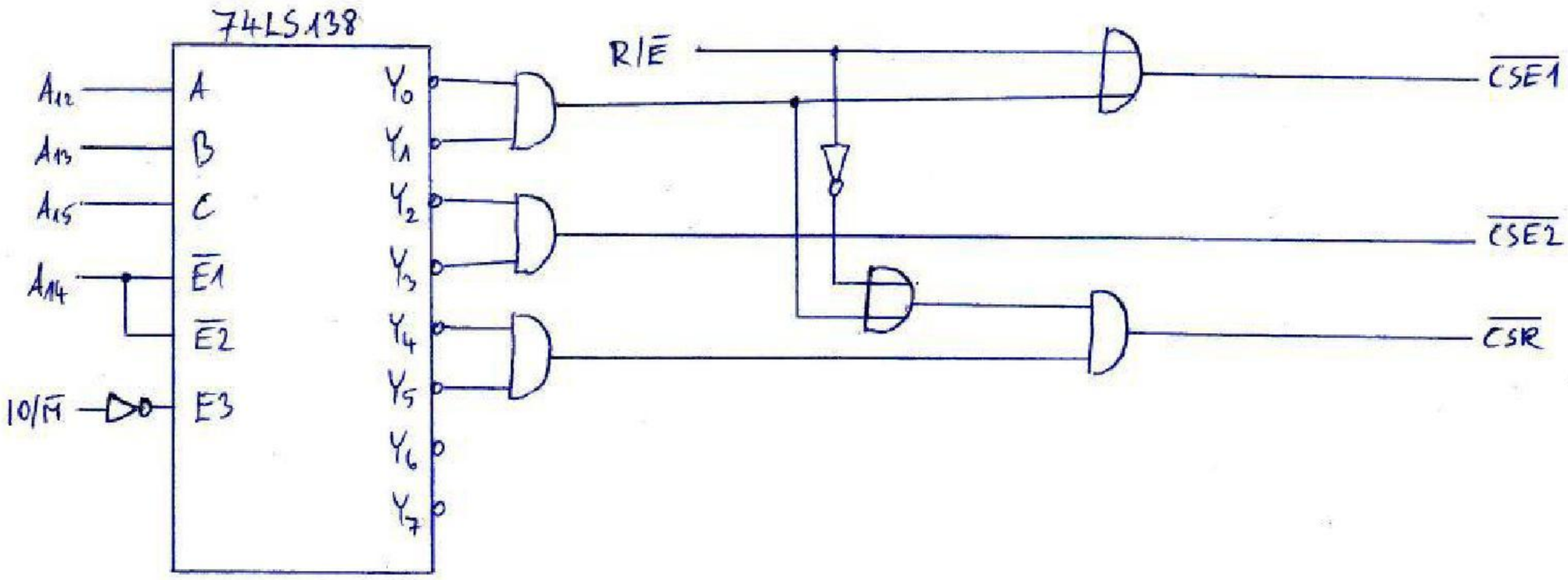


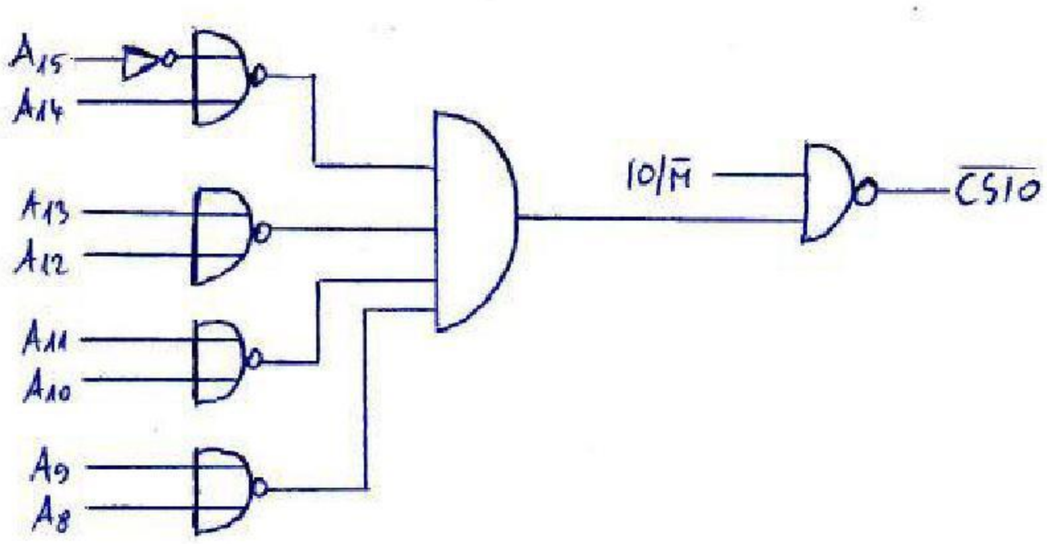
SZABÓ NORBERT
AJDSYL

1.b)

CPU	A_{15}	A_{14}	A_{13}	A_{12}	MEMÓRIA
0000H 0FFFH	0	0	0	0	8K RAM: 2000H 1FFFH vagy EPROM1: 0000H 1FFFH
1000H 1FFFH	0	0	0	1	
2000H 2FFFH	0	0	1	0	8K EPROM2: 0000H 1FFFH
3000H 3FFFH	0	0	1	1	
X	—	—	—	—	X
8000H 8FFFH	1	0	0	0	8K RAM: 0000H 1FFFH
9000H 9FFFH	1	0	0	1	

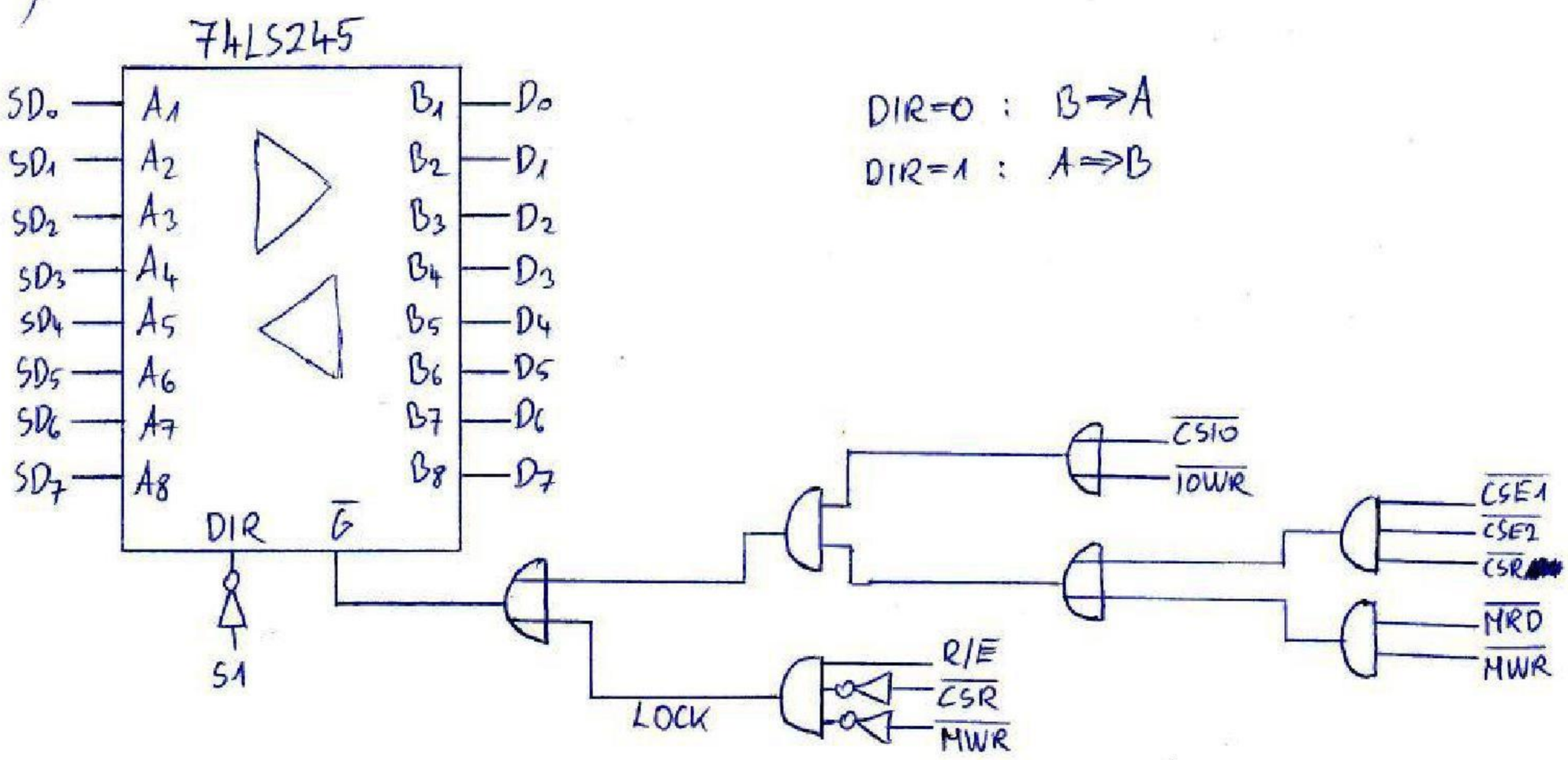
10 80H adat: R/$\overline{E}$





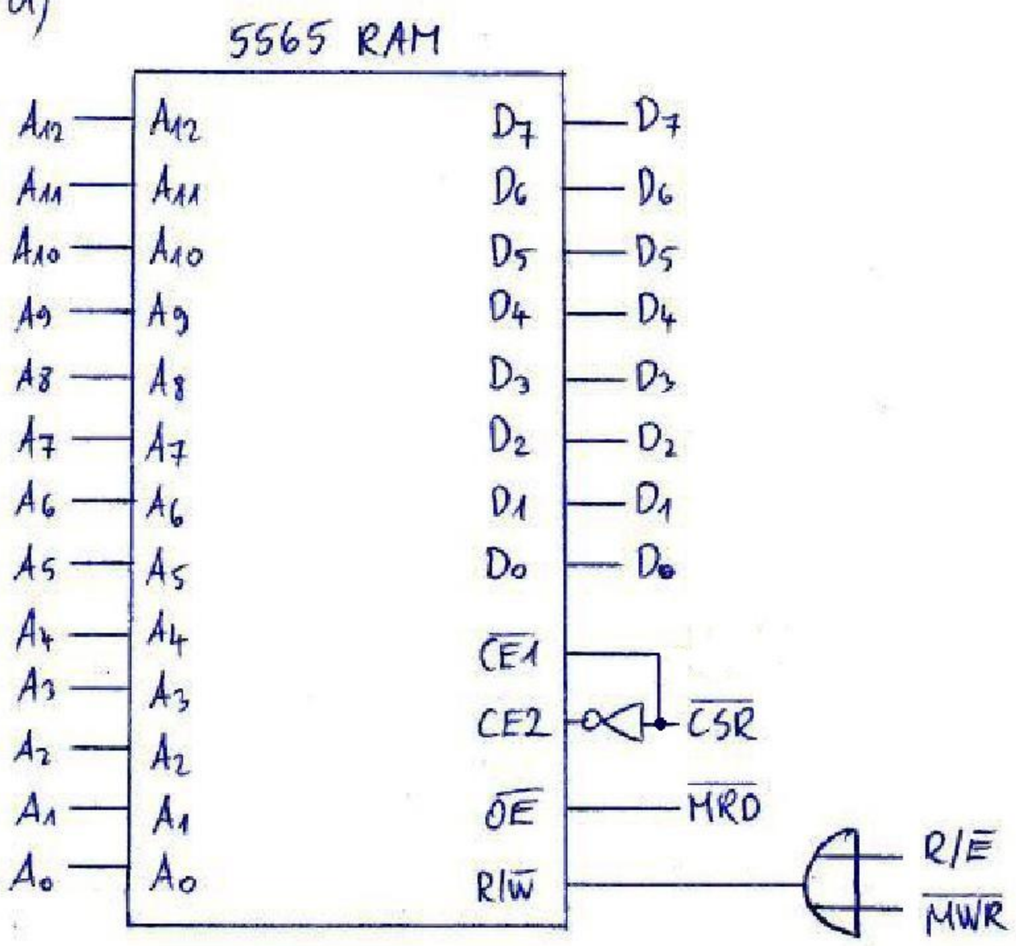
SZABÓ NORBERT
AJD5YL

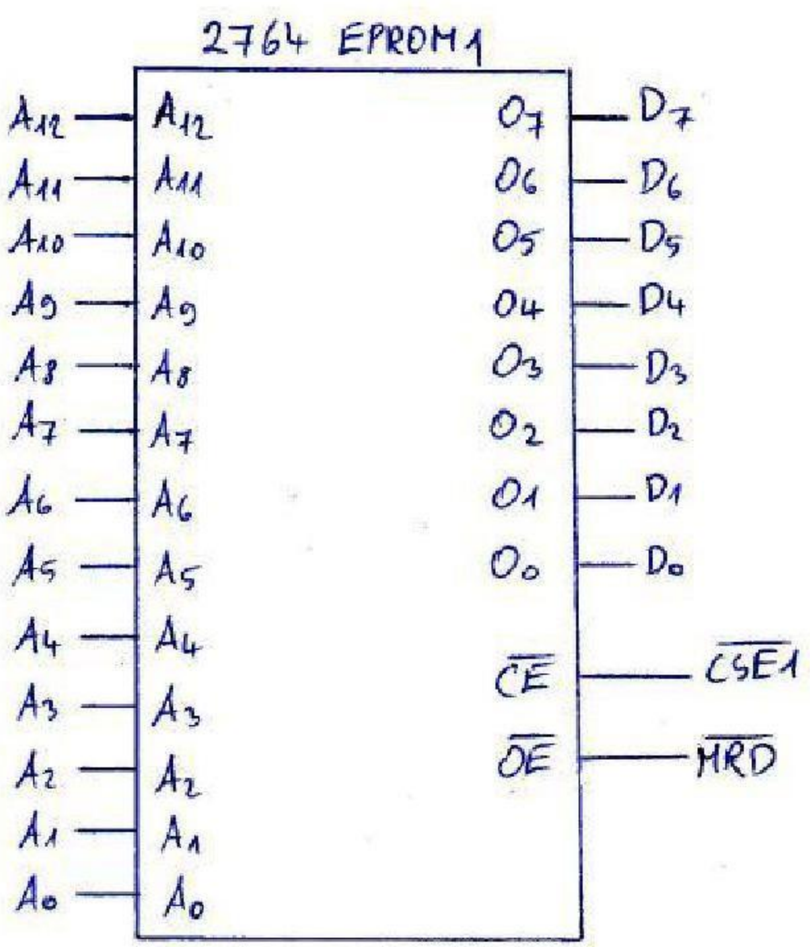
1.c)

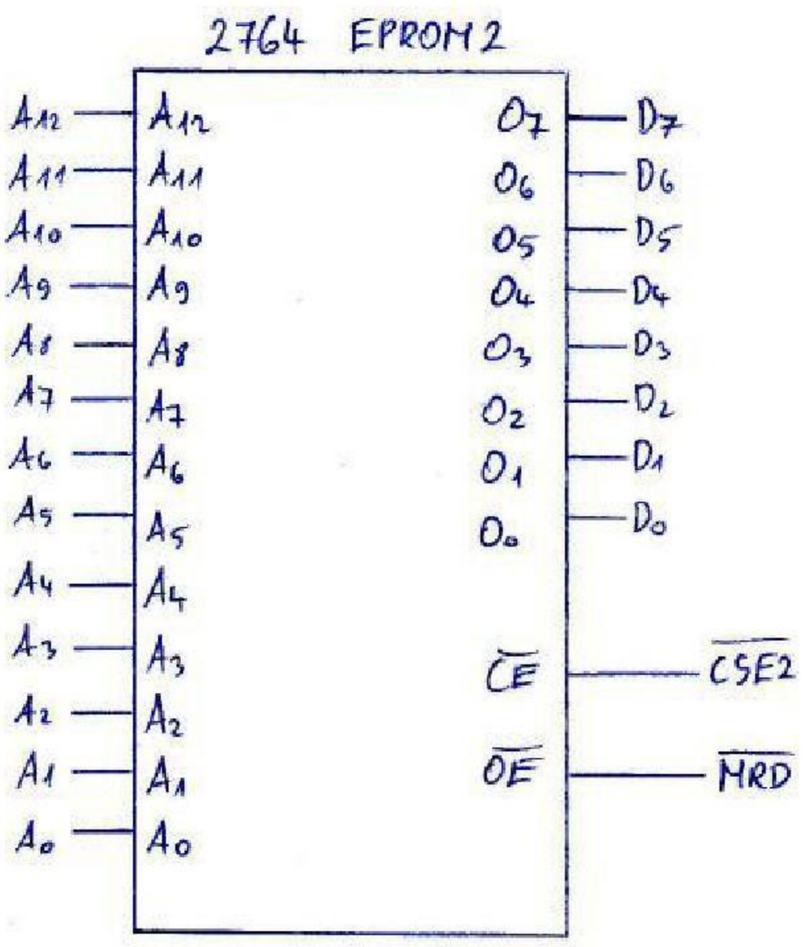


Ha R/E=1, akkor a 0000H címen a RAM lathnik, tehat irhato-olvashato.

d)

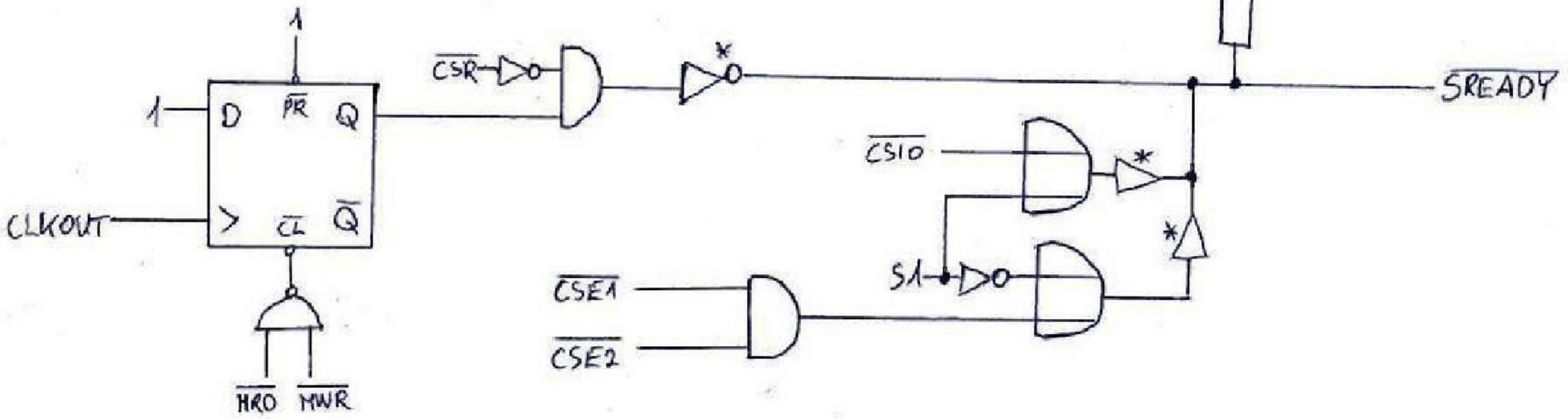




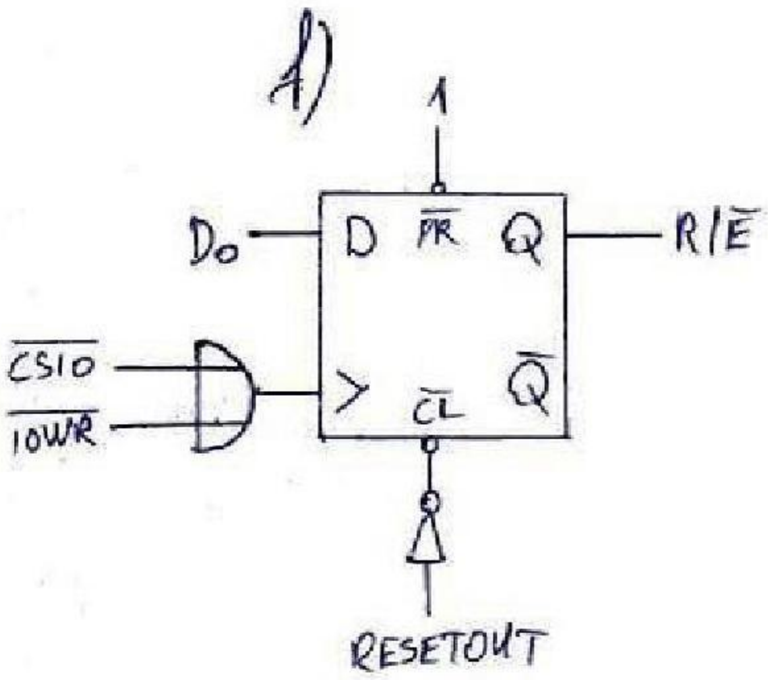


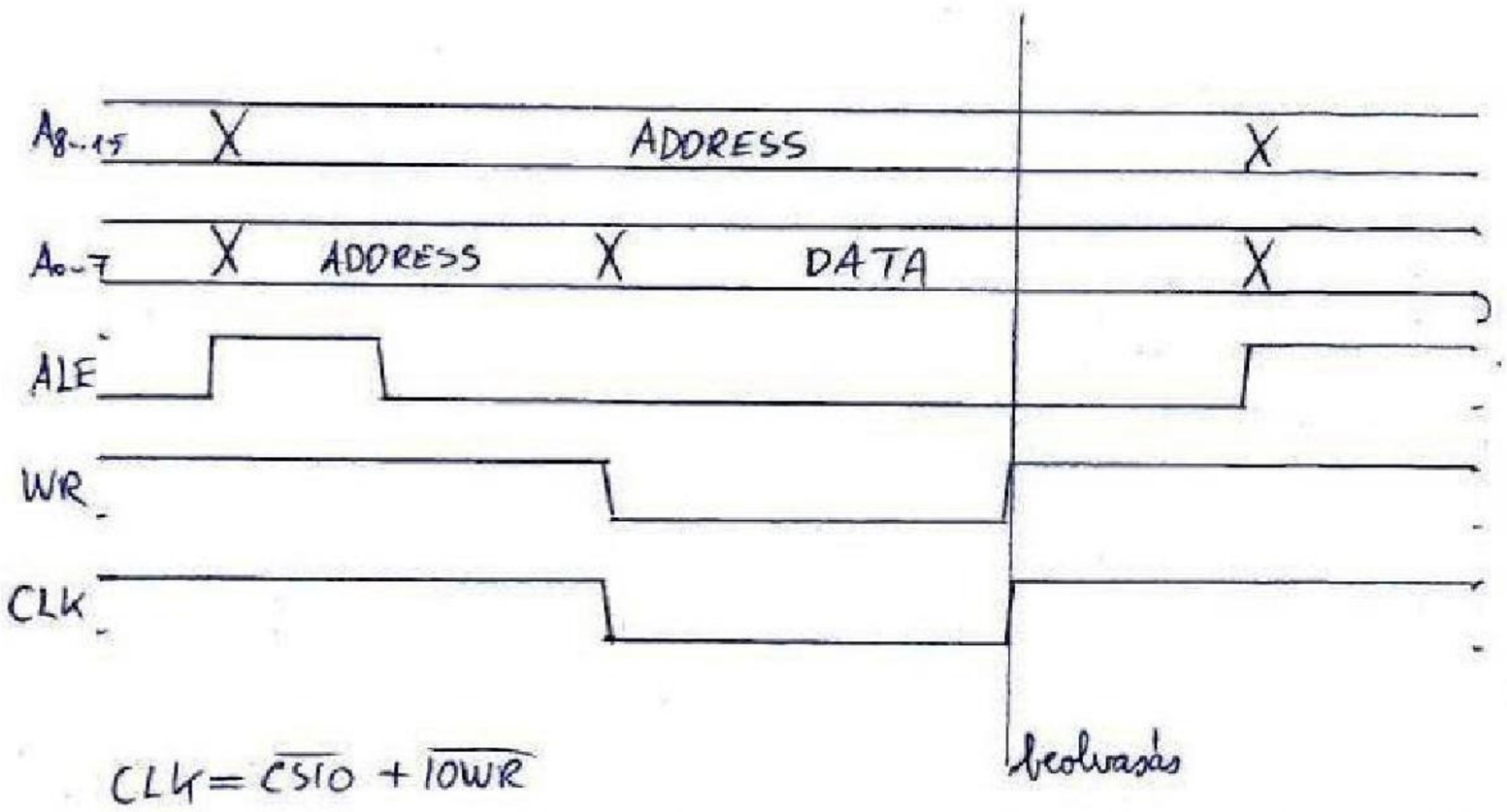
SZABÓ NORBERT
AJD5YL

1.e)



1)





$CLK = \overline{CS10} + \overline{IOWR}$

Ha nem a megadott cím van kiválasztva, $\overline{CS10} = 1$, $CLK = 1$ végig!

Ha a megadott cím van kiválasztva, $\overline{CS10} = 0$, $CLK = \overline{IOWR}$

SZABÓ NORBERT
AJD5YL

12.a)

```
; KITOLT rutin
; HL : kezdőcím
; DE : hossz

PAROSDATA EQU 55H
PARATLANDATA EQU 0AAH

KITOLT: PUSH PSW
        PUSH H
        PUSH D ; regiszterek mentése
        MOV A,D
        ORA E
        JZ KVEGE ; ha DE tartalma nulla, vége
        MOV A,L
        ANI 01H ; kezdőcím páros/páratlan
        JNZ KPARATLAN
KPAROS: MVI A, PAROSDATA ; páros cím esetén
        XCHG ; DE és HL cseréje
        STAX D ; DE által mutatott címre írjuk az akkumulátor tartalmát
        XCHG
        DCX D ; hátralévő méret csökken
        INX H ; következő cím
        MOV A,D
        ORA E
        JZ KVEGE ; ha DE tartalma nulla, vége
KPARATLAN: MVI A, PARATLANDATA ; páratlan cím
        XCHG
        STAX D
        XCHG
        DCX D
        INX H
        MOV A,D
        ORA E
        JZ KVEGE
        JMP KPAROS ; ha DE nem nulla, folytatjuk
KVEGE: POP D
        POP H
        POP PSW ; visszaállítás
        RET
```

SZABÓ NORBERT
AJD5YL

2.b/

```
;ELLENOR szubrutin
;HL: kezdőcím
;DE: hossz

PAROSDATA EQU 55H
PARATLANDATA EQU 0AAH
ELLENOR: PUSH D
         PUSH PSW
         PUSH H ;mentés
         LXI B, 0H ;kezdetben nulla hiba
         MOV A,L
         ANI 01H ;kezdőcím páros/páratlan
         JNZ EPARATLAN
EPAROS:  MOV A,M
         XRI PAROSDATA ;flag állítás
         JZ OK1 ;ha megfelelő értékvolt, ugrunk
         INX B ;egyébként hibák száma nő
         INX SP
         INX SP ;stack pointer visszaállítása
         PUSH H ;felülírjuk a címet
OK1:     DCX D ;hátralévő hossz csökken
         INX H ;cím nő
         MOV A,D
         ORA E
         JZ EVEGE ;ha E és D is 0, vége
EPARATLAN: MOV A,M ;ugyanez páratlan címre
         XRI PARATLANDATA
         JZ OK2
         INX B
         INX SP
         INX SP
         PUSH H
OK2:     DCX D
         INX H
         MOV A,D
         ORA E
         JZ EVEGE ;ha DE nem 0, folytatjuk
         JMP EPAROS
EVEGE:   MOV A,B
         ORA C
         JZ NINCSHIBA ;ha BC nullát tartalmaz, míves hiba
VANHIBA: POP H ;utolsó hiba címe
         POP PSW
         STC
         CMC ;CY=0
         JMP FINISH
NINCSHIBA: POP H ;kezdőcím
         POP PSW
         STC ;CY=1
FINISH:  POP D
         RET
```

SZABÓ NORBERT
AJD5YL

2.c)
```
; RAM terület tesztelése: 8000H-9FFFH
; indítás: SID lefutó élen 0->1 átmenet
; eredmény: CY=1, ha nincs hiba; CY=0, ha van hiba

RAMCIM  EQU 8000H
RAMSIZE EQU 2000H
IOCIM   EQU 80H
IODATA  EQU 0H ; RAM nem írásvédett
SETSOD  EQU 0C0H ; 11000000B
CLEARSOD EQU 40H ; 01000000B
W3MS    EQU 384 ; számolás alul

        MVI A, IODATA
        OUT IOCIM ; RAM 8000H címtől látszik, nem írásvédett
WAIT:   RIM
        ANI 80H ; első bit vizsgálata (SID)
        JNZ WAIT ; ha induláskor már 1, várunk
WAIT2:  RIM
        ANI 80H
        JZ WAIT2 ; 0->1 átmenetig várunk
        CALL IMPULZUS
        LXI H, RAMCIM ; kezdőcím
        LXI D, RAMSIZE ; hossz
        CALL KITOLT
        CALL ELLENOR
        CALL IMPULZUS
        ...
        RET

IMPULZUS: PUSH PSW
        PUSH H
        MVI A, SETSOD ; SOD=1 és engedélyezve
        SIM ; SOD 1-be áll
        LXI H, W3MS
VAR:    DCX H ; vissaszámol
        MOV A, H
        ORA L
        JNZ VAR ; várunk, amíg HL tartalma nem nulla
        MVI A, CLEARSOD ; SOD=0 és engedélyezve
        SIM ; SOD 0-ba áll
        POP H
        POP PSW
        RET
```

} 3ms

Az EPROM 0 wait-tel működik!

CPU órajele: 3,072 MHz
⇓
Egy fázis: 325,5 ns

Két SIM között:
- 1db LXI – 10 fázis
- xdb DCX – 6x fázis
- xdb MOV – 4x fázis
- xdb ORA – 4x fázis
- x-1db JNZ – 10·(x-1) fázis
- 1db JNZ – 7 fázis
- 1db MVI – 7 fázis

24x+14 fázis

$\frac{3\,000\,000\ ns}{325,5\ ns} \approx 9217$

$24x + 14 = 9217$

$x \approx 384$