

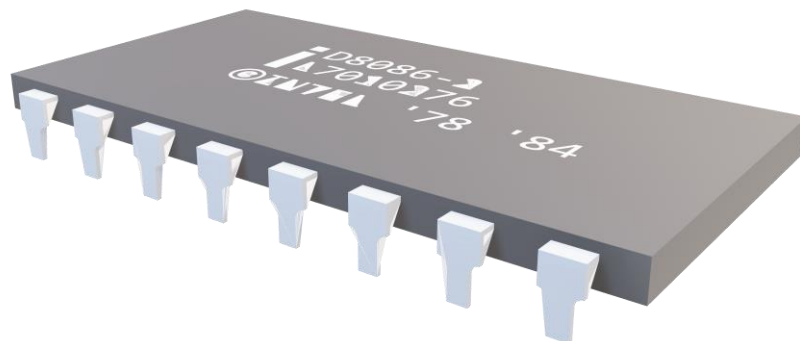


Mikroračunarski sistemi I deo računskih vežbi

Asistenti:

Vladimir Simić: vladimir.simic@elfak.ni.ac.rs, L3

Nenad Petrović: nenad.petrovic@elfak.ni.ac.rs, 323

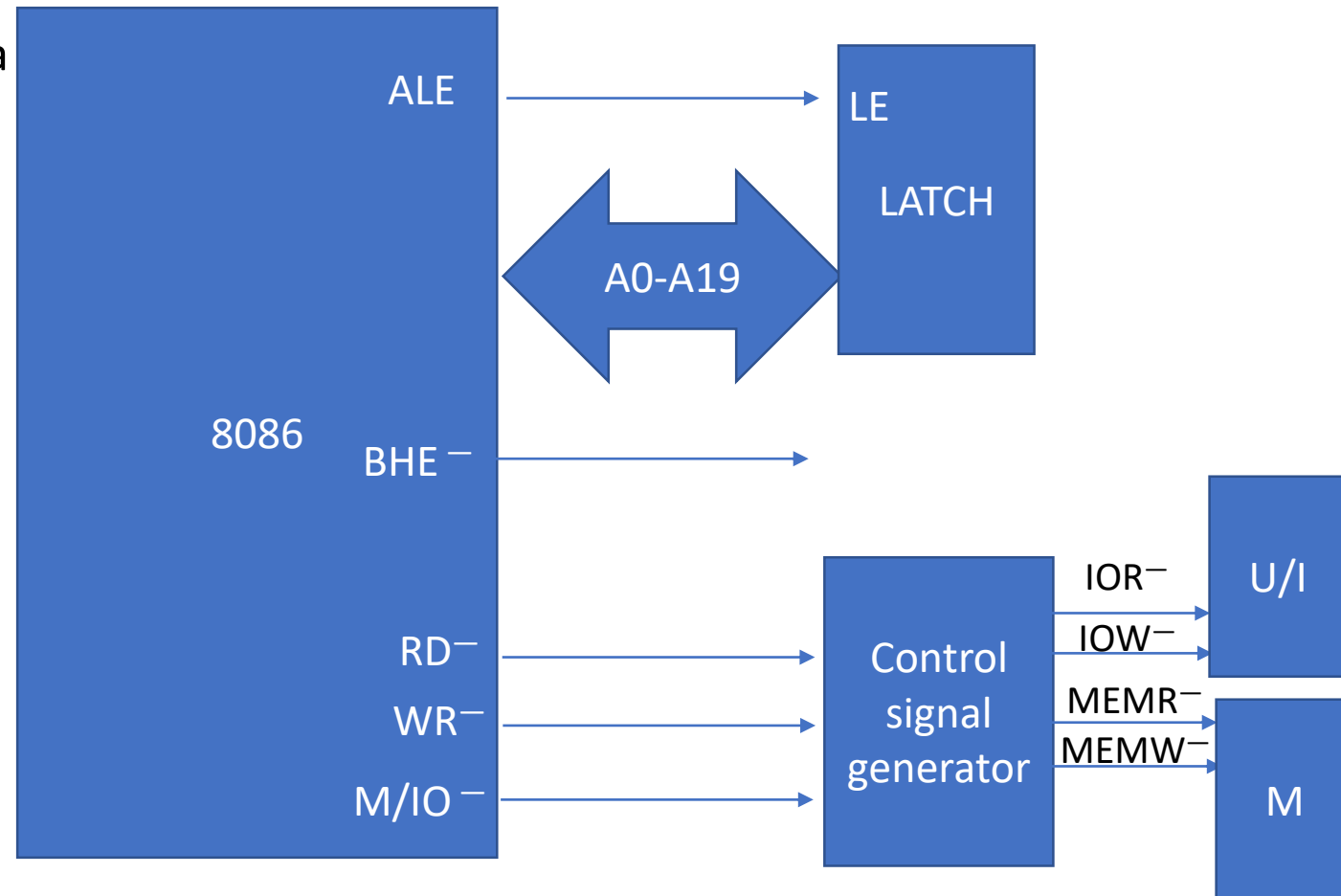


Memorijska organizacija 8086

1. termin računskih vežbi

Značajni pinovi 8086

- ALE
 - A0-A15 multipleksirane sa linijama podataka
 - Upis adrese u latch
 - U sledećem ciklusu: podaci
- Selekcija memorijske banke:
 - BHE⁻: neparna
 - A0⁻: parna
- Čitanje/upis: RD⁻ /WR⁻
- M/IO⁻
 - M: memorija (MOV), A0-A19
 - IO: U/I (IN, OUT), A0-A7/A0-A15
 - IN dst, source
 - OUT dst, source



Memorijska organizacija 8086 - pregled

- Dve vrste pristupa: byte i word
- Logički bajtovski adresni prostor u opsegu: 0-0xFFFFF
- 2 memorijske banke: parna (niža) i neparna (viša)
- Parna banka
 - D7-D0
 - Selekcija bitom A0 (BLE –)
- Neparna banka
 - D15-D8
 - Selekcija bitom BHE –
- A19-A1 adresne linije se koriste za selekciju bajtova iz banaka

8-bitne memorijske banke kod 8086

Neparna banka (viša)

FFFFFF
FFFFD
FFFFB
...
...
00001

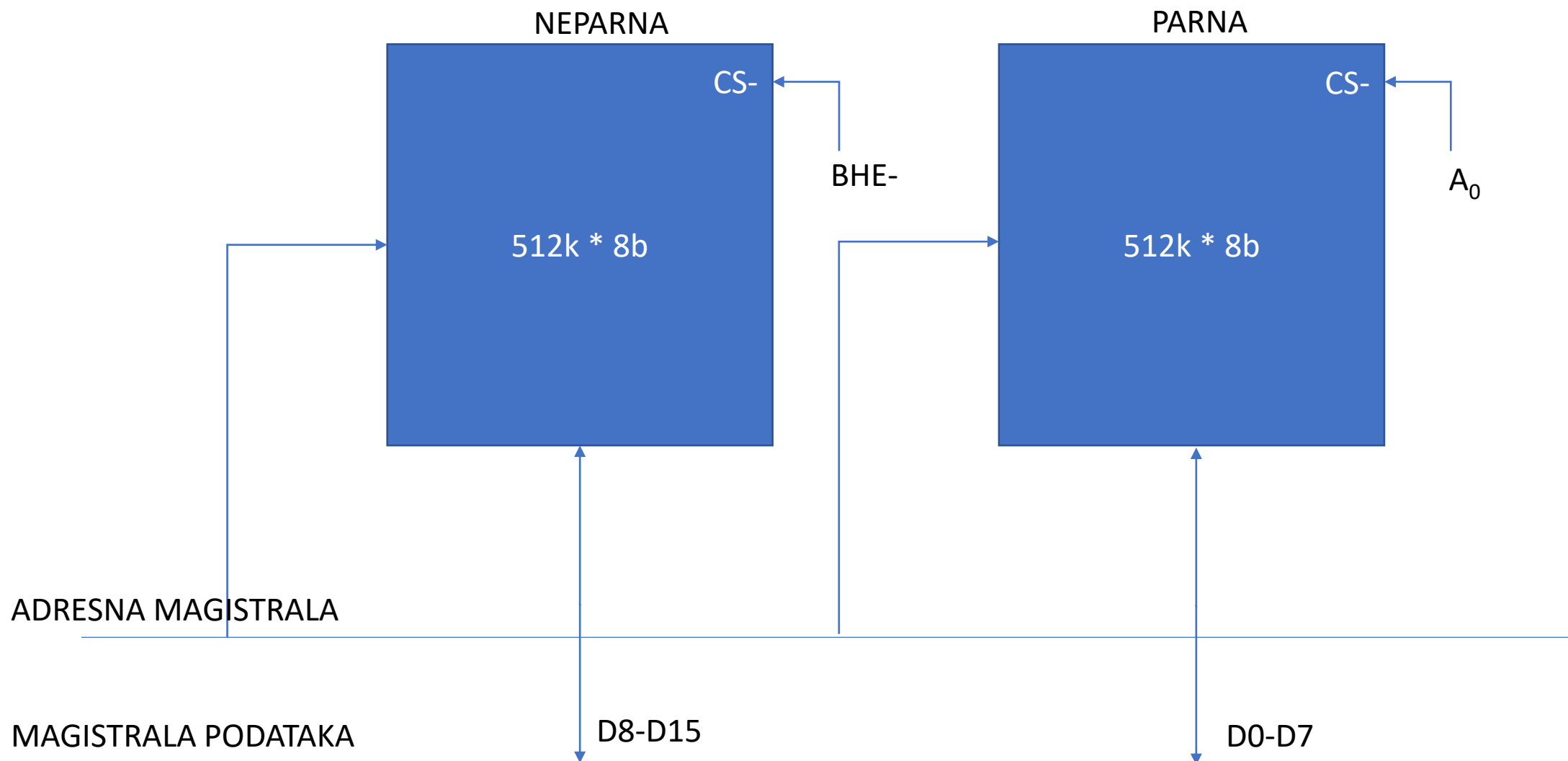
BHE[−]

Parna banka (niža)

FFFFE
FFFFC
FFFFA
...
...
00000

BLE[−]

Ilustracija



Selekcija memorijskih banaka

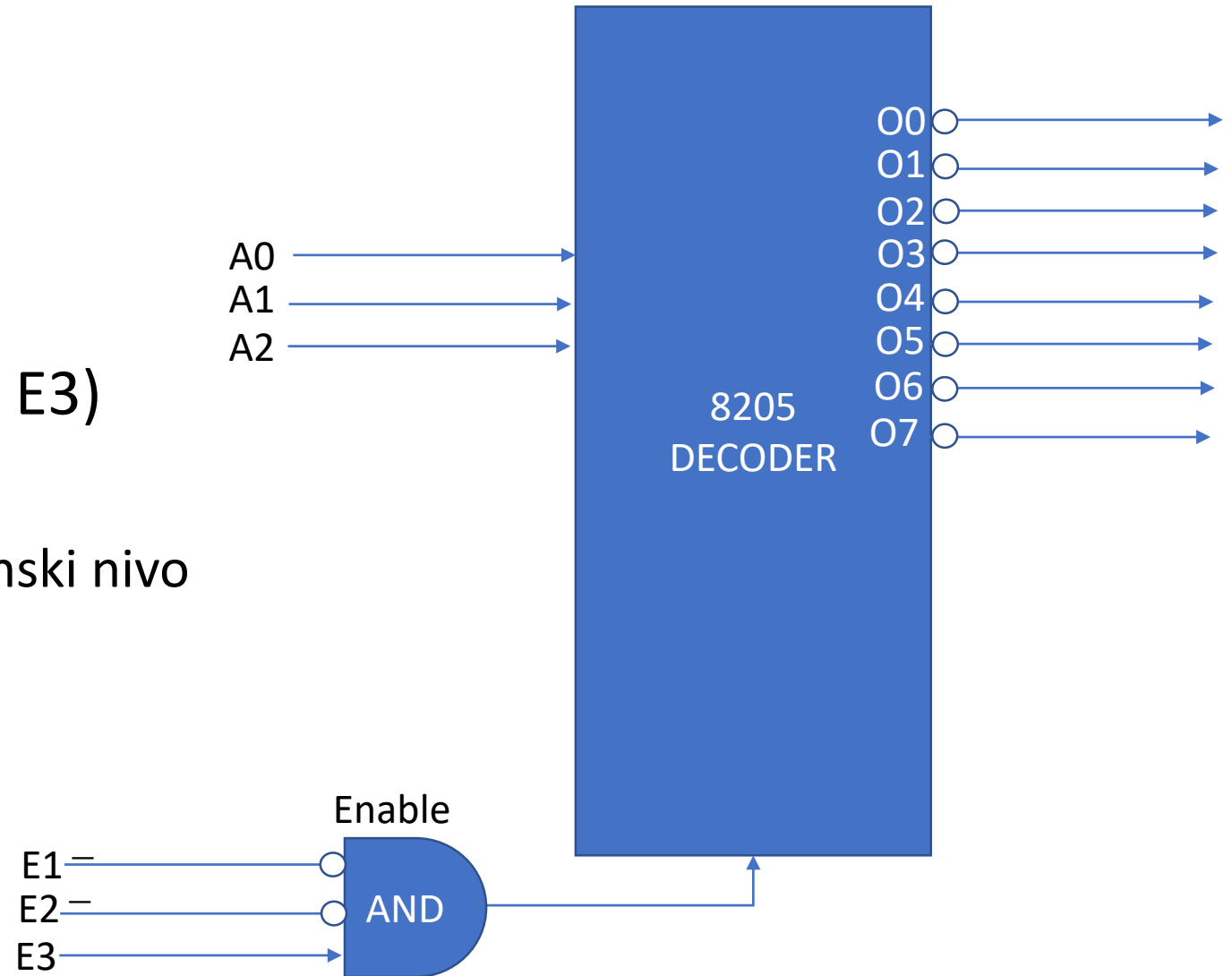
BHE ⁻	BLE ⁻ (A ₀)	Funkcija
0	0	Obe banke, 16-bit transfer
0	1	Viša (neparna) banka, 8-bit transfer
1	0	Niža (parna) banka, 8-bit transfer
1	1	Banke nisu omogućene

Primeri memorijskog pristupa

Primer	BHE [−]	A ₀	Pristup	Broj ciklusa
MOV AL, [4C00h]	1	0	8bit	1
MOV AL, [4C01h]	0	1	8bit	1
MOV AX, [4C00h]	0	0	16bit	1
MOV AX, [4C01h]	*	*	16bit	2: prvo neparna (01), pa parna (10)

8205 dekode

- 1 od 8 binarni dekode
- Ulazni pinovi A0, A1, A2
- Trostruki enable ($E1^-$, $E2^-$, E3)
- Izlazni pinovi O1-O7
 - Selektovani izlaz: nizak naponski nivo
 - Svi ostali: visok



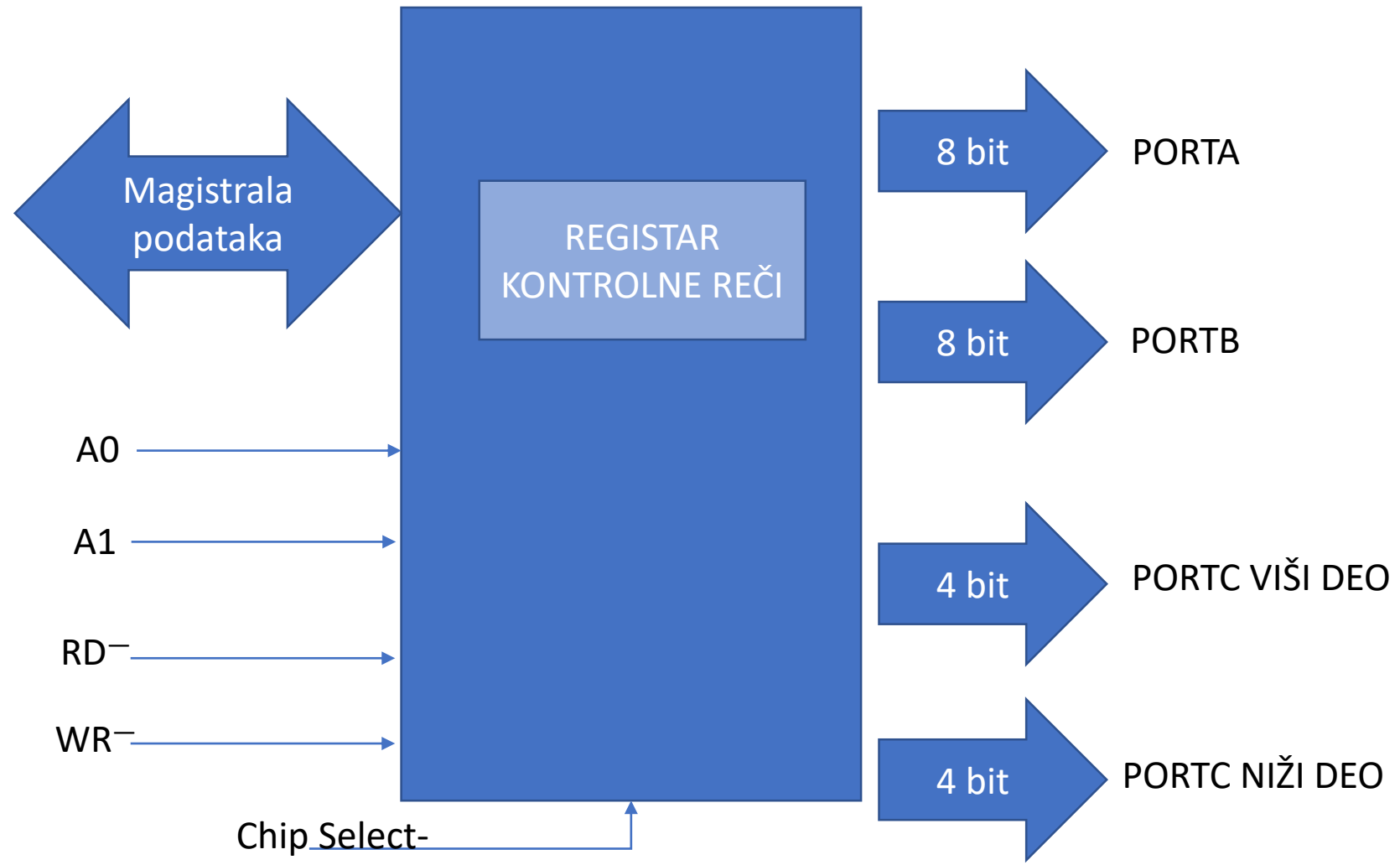
Komponenta 8255

1. termin računskih vežbi

Uvod

- Komponenta za paralelni ulaz/izlaz
- Koristi se da bi processor komunicirao sa periferijama (da ne troši svoje pinove za ulazne/izlazne uređuje)
- Ulazni/izlazni portovi se koriste za pribavljanje/upis podataka u toku komunikacije sa eksternim uređajima
- Primeri uređaja:
 - LED
 - tastature
 - senzori
 - displeji

Struktura 8255



Značajni pinovi

- D0-D7: 8 bit magistrala između komponente i CPU
- CS-: Selekcija čipa
- Signali čitanja i pisanja
 - RD[−]
 - WR[−]
- Tri 8-bit porta:
 - PA0-PA7
 - PB0-PB7
 - PC0-PC7
- A0 i A1: adresiranje portova i internog registra kontrolne reči

Adresiranje portova i kontrolnog registra

A_1	A_0	Adresira
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Registar kontrolne reči

Struktura kontrolne reči

D7	D6	D5	D4	D3	D2	D1	D0
0- SET/RESET	00 – MODE0 01- MODE1 1X – MODE2 MOD GRUPE A		0 -PORTA OUT	0 – PORTC HIGHER OUT	0 - MODE0	0 – PORTB OUT	0 – PORTC LOWER OUT
1 - I/O MODE			1 – PORTA IN	1 – PORTC HIGHER IN	1 - MODE1	1- PORTB IN	1 - PORTC LOWER IN
			GRUPA A		MOD GRUPE B	GRUPA B	

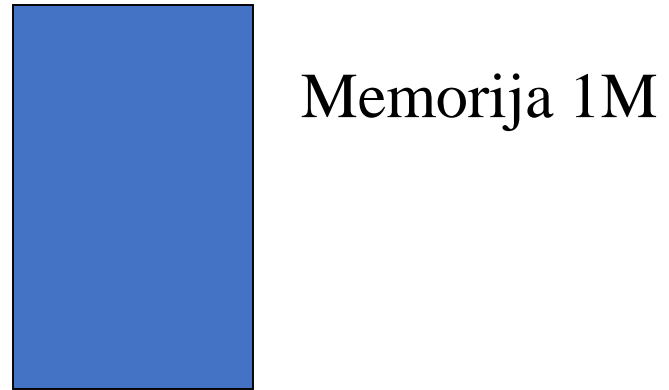
Modovi rada 8255

- Bit set/reset mode (D7=0)
 - Koristi se samo za set/reset bitova porta C.
 - D3, D2, D1 se koriste za izbor bita porta C
 - D0 određuje operaciju: 0-RESET, 1-SET
- I/O mode (D7=1)
 - Mode 0 – jednostavan ulaz/izlaz: portovi A, B, C viši, C niži ulazni ili izlazni
 - Mode 1 – handshake: A i B ulazni ili izlazni, C se koristi kao kontrolni
 - Mode 2- bidirectional: port A se koristi kao ulazni i izlazni (uzastopno), menja smer u toku rada, a port C kao kontrolni

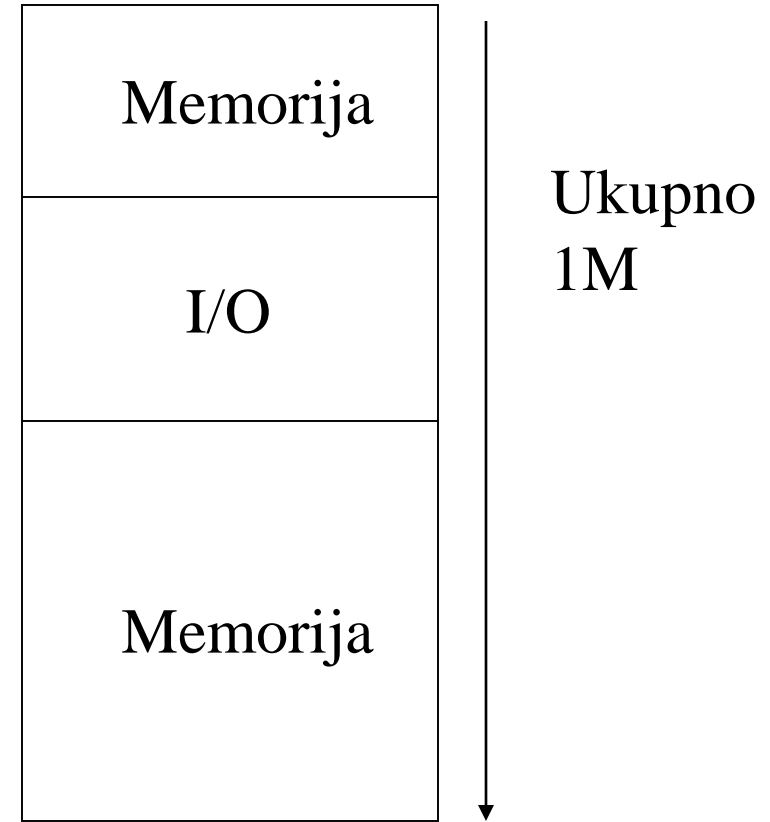
8255 interfacing

- Dva načina povezivanja sa 8086:
 - IO-mapirani ulaz-/izlaz (koriste se IN, OUT)
 - Memorijski-mapirani ulaz/izlaz (koristi se MOV)
- IO-mapirani ulaz/izlaz:
 - Kontrolni signali IOR – /IOW – generisani mehanizmom 8086
 - Adrese portova u ovom slučaju 8/16-bit
 - Chip select logika dekodira A3-A7
- Memorijski-mapirani ulaz/izlaz:
 - Kontrolni signali MEMR – /MEM W – generisani logikom za generisanje kontrolnih signala
 - Adrese portova u ovom slučaju 20-bit
 - Chip select logika dekodira A3-A19

Ilustracija



IO-mapirani

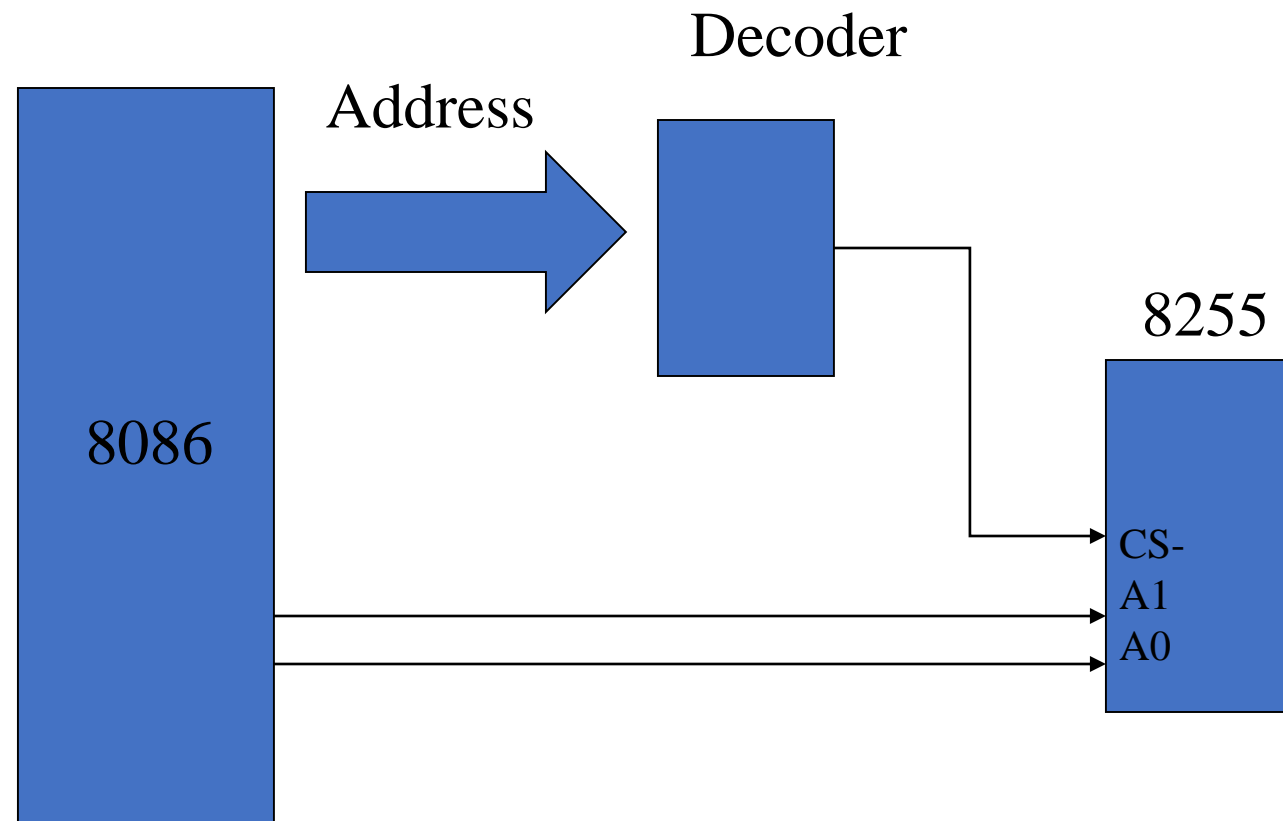


Memorijski-mapirani

Razlike IO-mapiranog i memorijski- mapiranog U/I

Memorijski-mapirani	IO-mapirani
Koristi se MOV instrukcija: MOV AL, [2000] MOV [2010], AL	Koriste se IN i OUT: IN AL, 30h OUT DX, AL
20 bit adrese - A19-A0, zahteva da se učitava DS Primer: port 35000H MOV AX, 3000H MOV DS,AX MOV AL, [5000H]	16/8 bit adrese – A15-A0/A7-A0
MEMR, MEMW kontrolni signali	IOR, IOW
Maksimalan broj portova (prednost): 2^{20}	2^{16}
Mane: <ul style="list-style-type: none">- zalazi u memorijski prostor- dovodi do fragmentacije memorijskog prostora- komplikovanje dekodiranja	Mane: <ul style="list-style-type: none">- manji broj portova

Primena dekodera



Zadatak 1

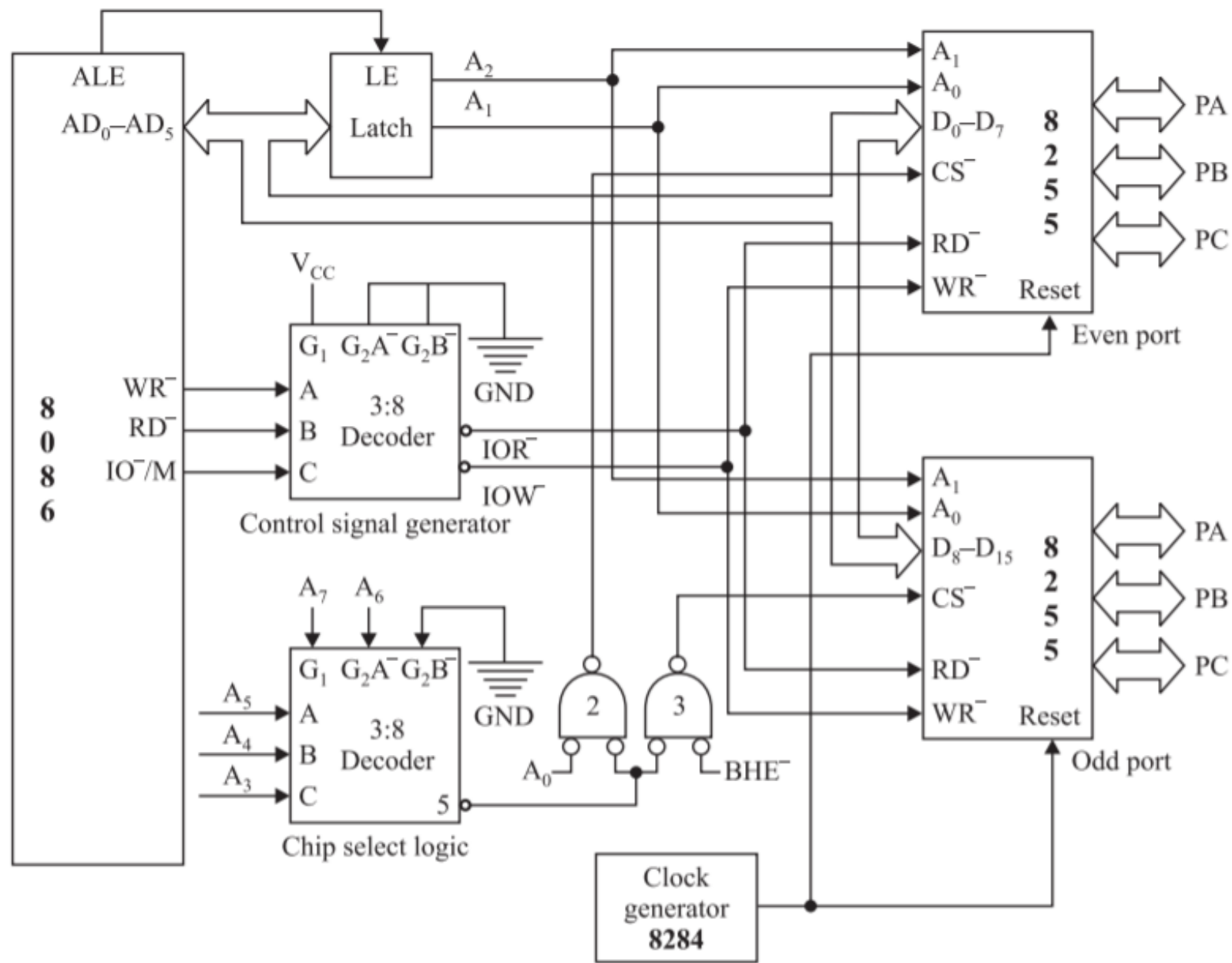
- Za mikroprocesor iAPX 8086 projektovati mikroračunarski sistem sa osam tastera i 8 LED dioda uz pomoć dve komponente 8255A. Na port A prve komponente treba vezati diode, a na port B druge komponente tastere. Inicijalno nijedna dioda ne svetli. Nakon toga posle pritiska nekog od tastera treba da zasvetli odgovarajuća dioda i treba da svetli sve dok je taster pritisnut. Napisati adekvatne procedure.
 - A) Koristi se IO-mapirani ulaz/izlaz (8bit adrese). Komponenta 1 je na adresi A8h, a Komponenta 2 je na A9h.
 - B) Koristi se memorijski-mapirani ulaz/izlaz: Komponenta 1 je na adresi 83FE8h, a Komponenta 2 je na 83FE9h.

Struktura rešenja zadatka

- Adresni prostor
- Šema povezivanja komponenti sa 8086
- Asemblerski kod za 8086
 - Konfiguracija komponenti
 - Logika i obrada

IO-mapirani ulaz/izlaz - Adrese

A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀		
1	0	1	0	1	0	0	0	= A8H = Port A	} Even port
1	0	1	0	1	0	1	0	= AAH = Port B	
1	0	1	0	1	1	0	0	= ACH = Port C	
1	0	1	0	1	1	1	0	= AEH = CWR	
1	0	1	0	1	0	0	1	= A9H = Port A	} Odd port
1	0	1	0	1	0	1	1	= ABH = Port B	
1	0	1	0	1	0	0	1	= ADH = Port C	
1	0	1	0	1	1	1	1	= AFH = CWR	
Used to enable the decoder									



Rešenje a)

- IO-mapirani ulaz/izlaz
- 8 bit IO adrese
- IN i OUT

```
NAME PROGRAM

DATA SEGMENT
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA

START:

MOV AX,DATA
MOV DS,AX

;ZABRANA PREKIDA
CLI

;KONFIGURACIJA
;KOMPONENTA1
MOV AL,80h
OUT AEh,AL

;KOMPONENTA2
MOV AL,82h
OUT AFh,AL

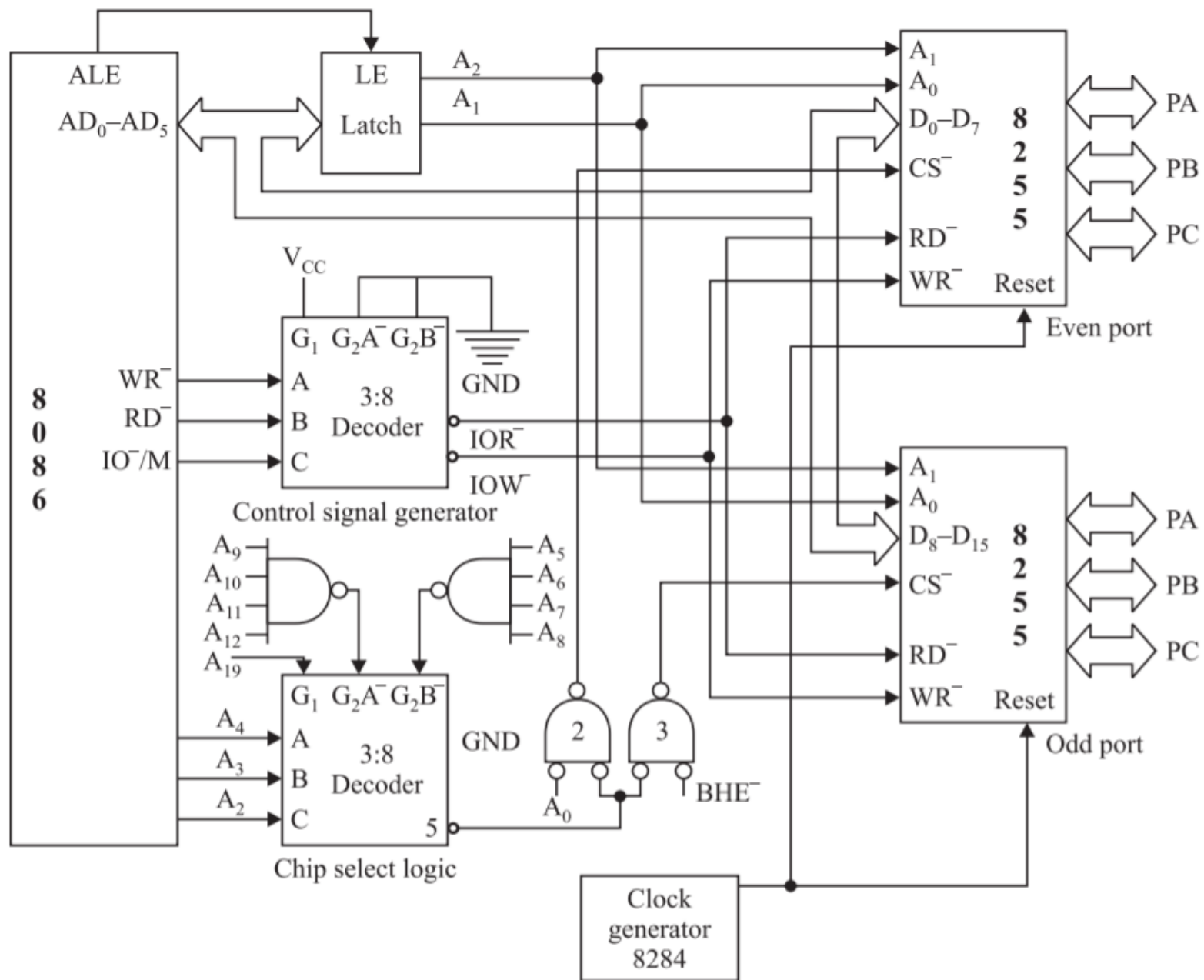
PETLJA:
;ČITANJE SA PORTB KOMPONENTE2
IN AL,ABh
;UPIS NA PORTA KOMPONENTE1
OUT A8h,AL
JMP PETLJA

MOV AX,4C02h
INT 21h

CODE ENDS
END START
END
```

Memorijski mapirani ulaz/izlaz - Adrese

A ₁₉	A ₁₈	A ₁₇	A ₁₆	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀		
1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	0	1	0	0	0	= 83FE8H	} Even port
1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	0	1	0	1	0	= 83FEAH	
1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	0	1	1	0	0	= 83FECH	
1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	0	1	1	1	0	= 83FEEH	
1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	0	1	0	0	1	= 83FE9H	} Odd port
1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	0	1	0	1	1	= 83FEBH	
1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	0	1	1	0	1	= 83FEDH	
1	X	X	X	X	X	1	1	1	1	1	1	1	1	1	0	1	1	1	1	= 83FEFH	



Rešenje b)

- Memorijski-mapirani ulaz/izlaz
- MOV umesto IN i OUT
- U loaderu se konfiguriše početak DS
 - Početak segmenta: 83FEh
 - Adresa(20bit)=Segment(83FE0)+Offset(000E)

```
NAME PROGRAM

DATA SEGMENT
DB 100 DUP (?)
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA

START:

MOV AX, DATA
MOV DS, AX

; ZABRANA PREKIDA
CLI

; KONFIGURACIJA
; KOMPONENTA1
MOV AL, 80h
MOV DS: [000E], AL

; KOMPONENTA2
MOV AL, 82h
MOV DS: [000F], AL

PETLJA:
; ČITANJE SA PORTB KOMPONENTE2
MOV AL, DS: [000B]
; UPIS NA PORTA KOMPONENTE1
MOV DS: [0008], AL
JMP PETLJA

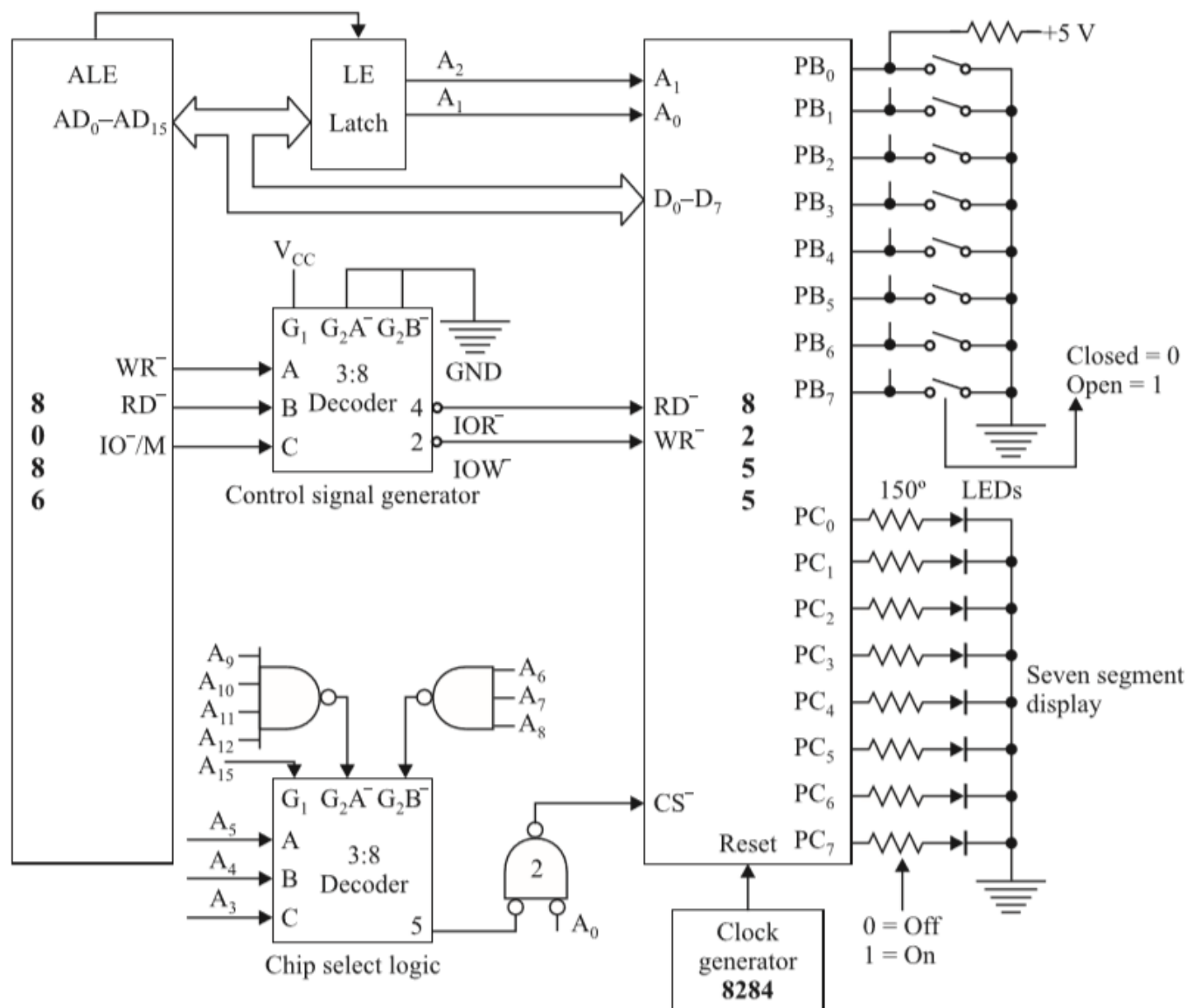
MOV AX, 4C02h
INT 21h

CODE ENDS
END START
END
```

Primer za 16bit IO

- Example 12.8, str. 411 u knjizi Sunil Mathur, "MICROPROCESSOR 8086: Architecture, Programming and Interfacing".
- Pretpostavljamo da imamo osam prekidača i jedan 7s displej povezan na 8086 preko 8255

A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	Even ports
1	X	X	1	1	1	1	1	1	1	1	0	1	0	0	0	= 9FF8H = Port A
1	X	X	1	1	1	1	1	1	1	1	0	1	0	1	0	= 9FEAH = Port B
1	X	X	1	1	1	1	1	1	1	1	0	1	1	0	0	= 9FECH = Port C
1	X	X	1	1	1	1	1	1	1	1	0	1	1	1	0	= 9FEFH = CWR



Program:

```

UP:    MOV BX, 2000H
        MOV DX, 9FEAH
        IN AL, DXH
        CMP AL, 00H
        JZ UP
        XLAT
        MOV DX, 9FECH
OUT DX, AL
CALL DELAY
JMP UP.

```

Reference

- Sunil Mathur, “MICROPROCESSOR 8086: Architecture, Programming and Interfacing”, str. 402-412, 2011.

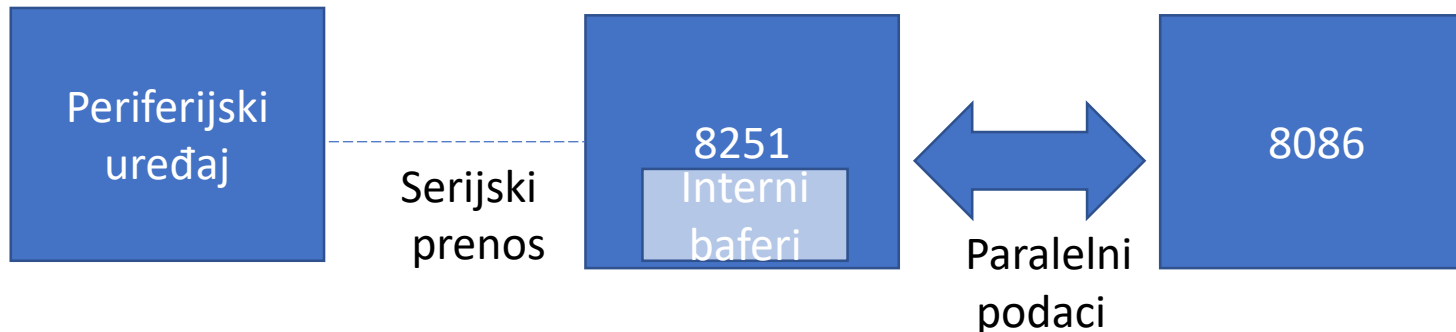
8251A USART

Universal Synchronous Asynchronous Receiver Transmitter

2. termin računskih vežbi

Uvod

- 8251A je programabilni komunikacioni interfejs namenjen sinhronom i asinhronom serijskom prenosu podataka
- Slanje i prijem podataka sa perifernih uređaja od/ka 8086
- Cilj je da uređaj bude “transparentan” za procesor, odnosno da se ponaša kao jednostavan ulaz/izlaz za bajtovski orijentisane podatke koje prima/šalje procesor



Signalizacija kod serijskog prenosa

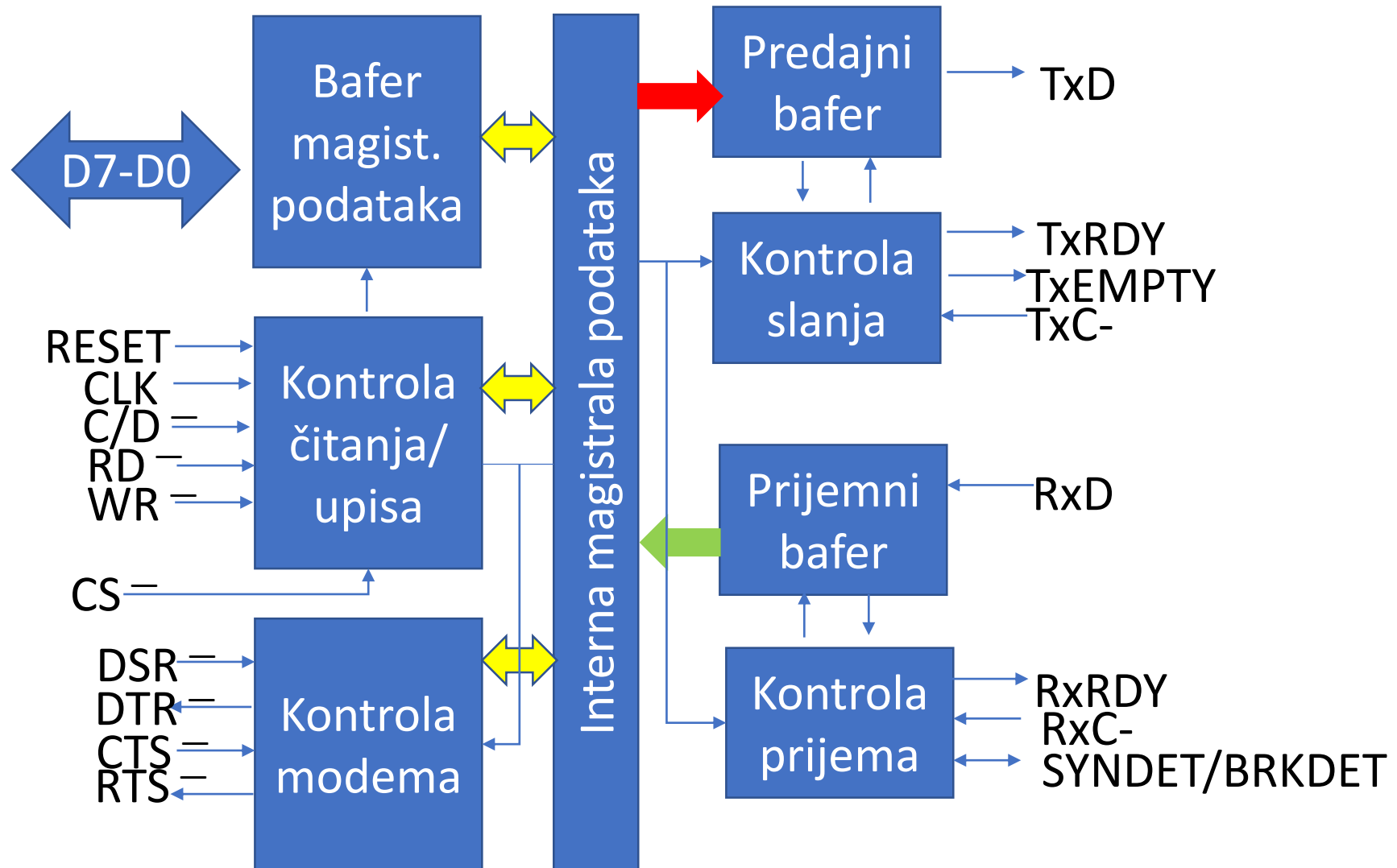
- Marking je stanje kada nema prenosa
- Prenos počinje kada signal padne sa 1 na 0
- Signal počinje start bitom koji traje 1 interval
- Šalje se 5 do 8 bitova podataka
- Opcioni bit parnosti
- Stop-bit može trajati 1, 1.5 ili 2 bitska intervala



Arhitektura 8251A

- 8251 se sestoji od 5 blokov:
 - Bafer magistrale podataka
 - Kontrona logika čitanja/upisa
 - Predajnik
 - Primalac
 - Kontrola modema

Interna struktura



Bafer magistrale podataka

- 8 bit magistrala podataka (linije D0-D7)
- Čitanje i upis sa/u 8251:
 - komandne reči
 - podataka
 - statusa

Blok kontrolne logike čitanja/upisa

- Kontrolna logika
 - Interfejs sa 8086, određuje funkcije čipa na osnovu sadržaja kontrolne reči i nadzor protoka podataka
- 6 ulaznih signala
- 3 baferska registra
 - Data register
 - Control register
 - Status register

Ulazni signali kontrolne logike

- CS –
 - Chip Select : Kada postane 0, obavlja se komunikacija sa 8086
- C/ D – – Control/Data :
 - 1: adresira se kontrolni ili statusni registar
 - 0: obraća se baferu podataka
- WR – :
 - Kada je 0, smer 8086->8251
 - Upis podataka ili kontrolne reči
- RD – :
 - Kada je 0, smer 8086<-8251
 - Čitanje podataka ili statusa
- RESET :
 - Kada se postavi na 1, resetuje 8251 i vraća u idle stanje
- CLK :
 - Taktni ulaz obično povezan na sistemski clock za komunikaciju sa mikroprocesorom

Kontrolni registar

- Registar kontrolne reči treba da dobije dve 8-bit nezavisne reči:
 - mode word
 - command word
- Mode word:
 - Parametri prenosa
- Command word:
 - Omogućava prijem i predaju podataka

Mode word – asinhroni prenos

- Broj stop bitova
- Bit parnosti
- Broj bitova podataka po prenetom karakteru
- Mod faktora brzine serijskog prenosa (baud rate factor)
 - faktor (određen bitima D1 i D0) predstavlja odnos između signala takta na ulazima TxC i RxC i željenog baud-rate-a.

S2	S1	EP	PEN	L2	L1	B2	B1
Broj stop bitova: 01 – 1 stop bit 10 – 1.5 stop bit 11 – 2 stop bita		Bit parnosti: 00 – disable 01 – neparna 10 – disable 11- parna		Broj bitova po karakteru: 00-5b 01-6b 10 -7b 11-8b		Baud rate factor: 00-Sync 01-1x 10-16x 11-64x	

Mode word – sinhroni prenos

- Broj karaktera sinhronizacije
- Mod sinhronizacije
- Bit parnosti
- Broj prenetih bitova podataka
- Baud rate isti kao RxC i TxC takt
- Indikator novog podatka na liniji je prijem jednog ili dva sync karaktera (konfigurabilan broj karaktera)

SCS	ESD	EP	PEN	L2	L1	0	0
Broj sync karaktera: 0- dva 1- jedan	Sync mod: 0-interna 1-eksterna	Bit parnosti: 00 – disable 01 – neparna 10 – disable 11- parna		Broj bitova po karakteru: 00-5b 01-6b 10 -7b 11-8b		Nisu u upotrebi	

Command word

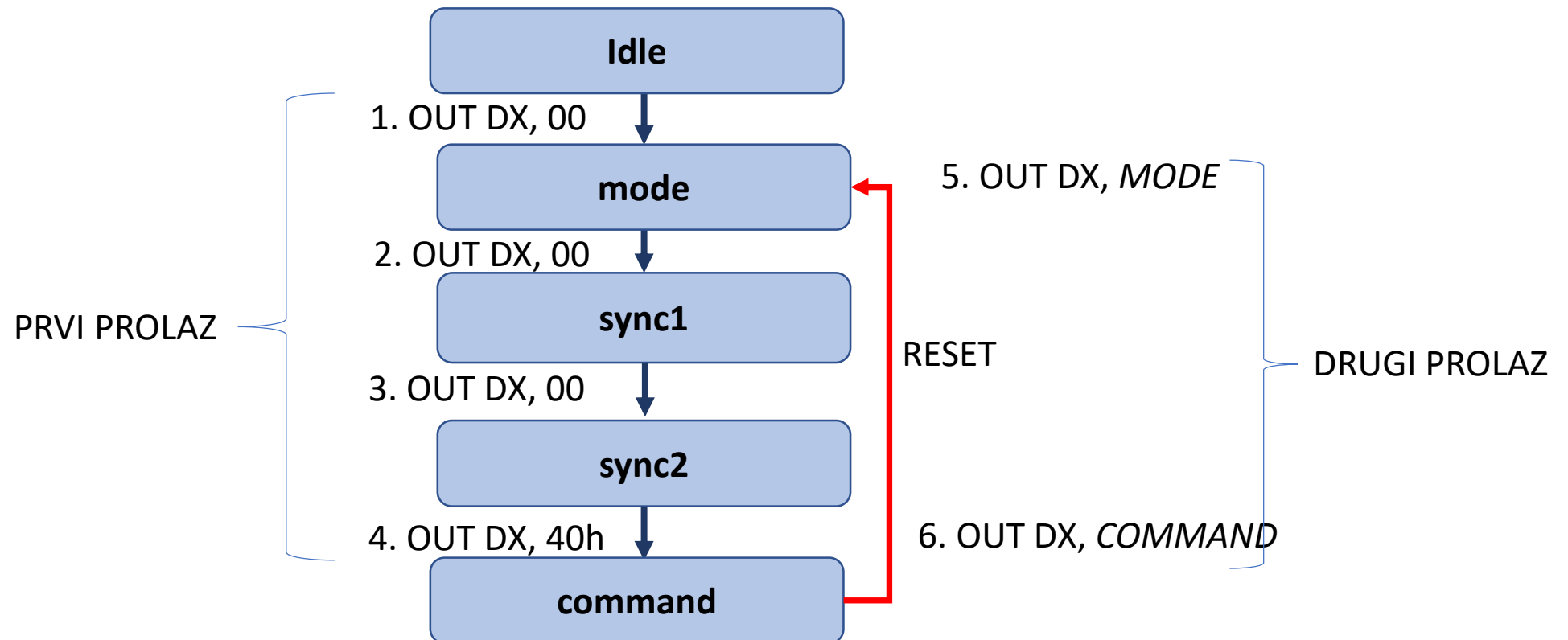
EH	IR	RTS	ER	SBRK	RXE	DTR	TXEN
0:Normal 1:Hunt mode	0:Normal 1:Internal reset	0:DTR->1 1: DTR->0	0:Normal 1: Reset error flag	0:Normalan režim 1:Slanje karaktera prekida	Omogućiti prijemnik	0:DTR->1 1: DTR->0	Omogućiti predajnik

Incijalizacija 8251

- Nakon paljenja uređaja, ne možemo tačno znati u kojem režimu rada se našao uređaj (da li očekuje MODE, SYNC CHARACTER ili COMMAND instrukciju)
- Zbog toga je potrebno izvršiti “worst-case” inicijalizacionu sekvencu
 - pretpostavićemo da uređaj očekuje MODE instrukciju, pa ćemo ga inicijalizovati u sinhroni režim rada, sa 2 sinhronizaciona karaktera – ovo postićemo upisom na kontrolnu adresu 3 uzastopna bajta kojima su vrednosti svih bita = 0
 - zatim softverski resetujemo uređaj, tako što šaljemo COMMAND instrukciju, u kojoj bit D6 ima vrednost 1

Dijagram stanja 8251

- Svaka kontrolna reč upisana u kontrolni registar posle MODE reči se tumači kao COMMAND reč, što znači da se COMMAND reč može promeniti bilo kada.
- Međutim, neophodno je resetovati 8251 pre nego da se upiše MODE reč.



Statusni registar

- Proverava status perifernog uređaja i sadrži informacije o greškama nastalih u prenosu:
- Framing Error
 - Nije detektovan start ili stop bit
- Overrun Error
 - Procesor nije pročitao prethodni podatak koji se nalazio u internom baferu
- Parity Error
 - Greška parnosti

DSR	SYNDET BRKDET	FE	OE	PE	TxE	RxRDY	TxRDY
Data set ready	Sync detect/break error	Framing error	Overrun error	Parity error	Predajnik prazan	Prijemnik spreman	Predajnik spreman

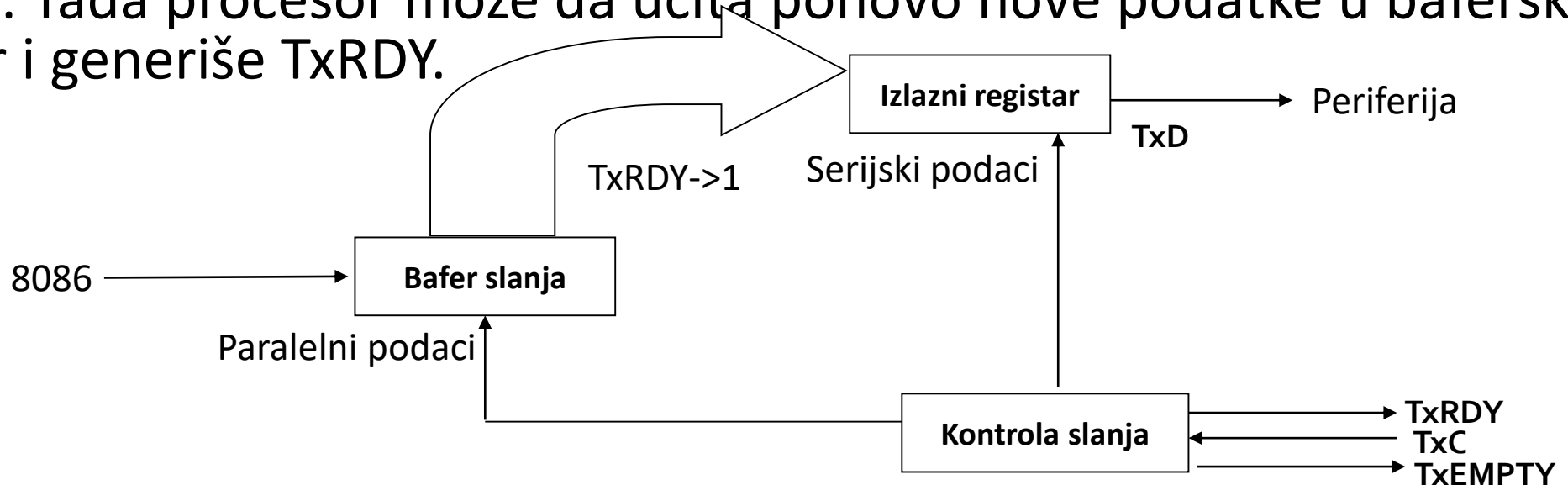
Registar podataka

- Koristi se kao U/I port kada je C/D signal na 0.
- Režimi rada

CS ⁻	C/D ⁻	WR ⁻	RD ⁻	Operacija
0	0	1	0	8086<-bafer podataka
0	0	0	1	8086->bafer podataka
0	1	0	1	8086->kontrolni registar
0	1	1	0	8086<-statusna reč
1	×	×	×	Čip nije selektovan

Predajnik

- Prihvata paralelne podatke sa 8086 i konvertuje ih u serijske
- Predajnik ima dva bafera:
 - Baferski registar za držanje 8-bit paralelnih podataka
 - Izlazni registar koji konvertuje paralelne podatke u serijske
- Kada je izlazni registar prazan, podaci se prebacuju iz bafera u izlazni registar. Tada procesor može da učitava ponovo nove podatke u baferski registar i generiše TxRDY.

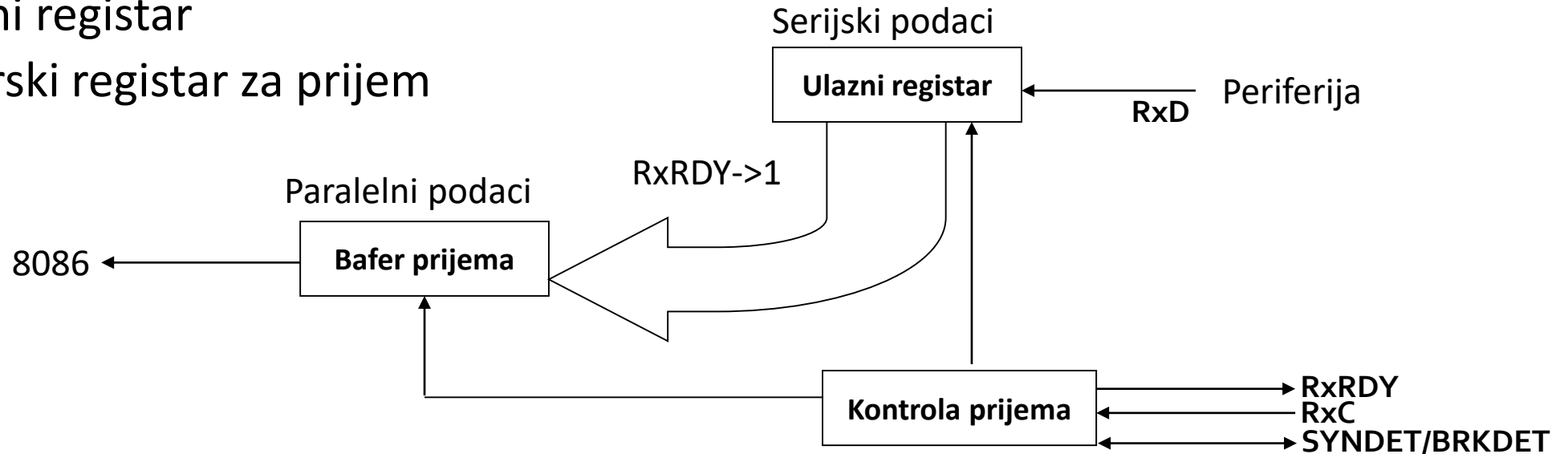


Signali predajnika

- 8086 upisuje bajt u baferski registar za slanje
- Kada je izlazni registar prazan, sadržaj bafera se prebacuje u izlazni
- 3 izlana signala i jedan ulazni:
 - TxD - Transmitted Data Output : Izlazni signal za prenos podataka na periferijske uređaje
 - TxC - Transmitter Clock Input : Ulazni signal za kontrolu brzine slanja ka 8251
 - TxRDY - Transmitter Ready : Izlazni signal koji označava da je baferski registar prazan i da 8255 može primiti sledeći bajt
 - TxE - Transmitter Empty : Izlazni signal koji označava da je izlazni registar prazan

Prijemnik

- Prihvata serijske podatke sa RxD pina i pretvara ih u paralelne
- Posедуje dva registra:
 - Ulazni registar
 - Baferski registar za prijem



Prijemnik - objašnjenje

- Prihvata serijske podatke i konvertuje ih u paralelne
- Dvostruko je baferovan:
 - Ulazni registar za prijem serijskih podataka i konverziju u paralelne
 - Baferski registar za držanje paralelnih podataka
- Kada je napon linije RxD nizak, kontrolna logika pretpostavlja da je to START bit, čeka u periodu od pola bita, zatim ponovo čita liniju.
- Ako je i dalje nizak napon na liniji, onda ulazni registar prihvata naredne bitove, dok se ne popuni do 8bit i učitava u baferski registar
- 8086 čita paralelne podatke iz baferskog registra.
- Kada su sa ulaznog registra učitani paralelni podaci u baferski registar, linija RxRDY dobija visok naponski nivo
- RxC (clock signal) kontroliše brzinu kojom su bitovi primljeni od strane 8251
- U asinhronom modu, signal SYNDET/BRKDET označava prekid u prenosu podataka
- U sinhronom modu, signal SYNDET/BRKDET označava prijem karaktera sinhronizacije

Signali prijemnika

- RxRDY - Receiver Ready Output: Izlazni signal, postaje 1 kada 8251 ima bajt u baferskom registru i spreman je da ga pošalje ka 8086
- RxD - Receive Data Input : Bitovi se na ovoj liniji primaju serijski
- RxC - Receiver Clock Input : Taktni signal za kontrolu brzine kojom se bitovi primaju od strane 8251

Slanje i prijem podataka

- Dva načina:
 - Interrupt
 - Polling
- Interrupt:
 - Za generisanje interrupta ka procesoru koriste se linije RxRDY i TxTDY
 - Mehanizam biće detaljnije obrađen u sledećem terminu računskih vežbi
- Polling:
 - zasniva se na proveru bita statusnog registra
 - čitamo i proveravamo u petlji bitove: D0 (za slanje – TxRDY) ili D1 (za prijem - RxRDY) sve dok ne dobiju vrednost 1
 - zatim šaljemo/čitamo podatak, a analogno mehanizmu interrupt-a, nakon slanja/čitanja, odgovarajući bit u statusnom registru će se resetovati, sve dok uređaj nije spreman da primi novi podatak za slanje, odnosno dok ne primi novi podatak sa periferije, kada se odgovarajući bit u statusnom registru ponovo postavlja na visok nivo

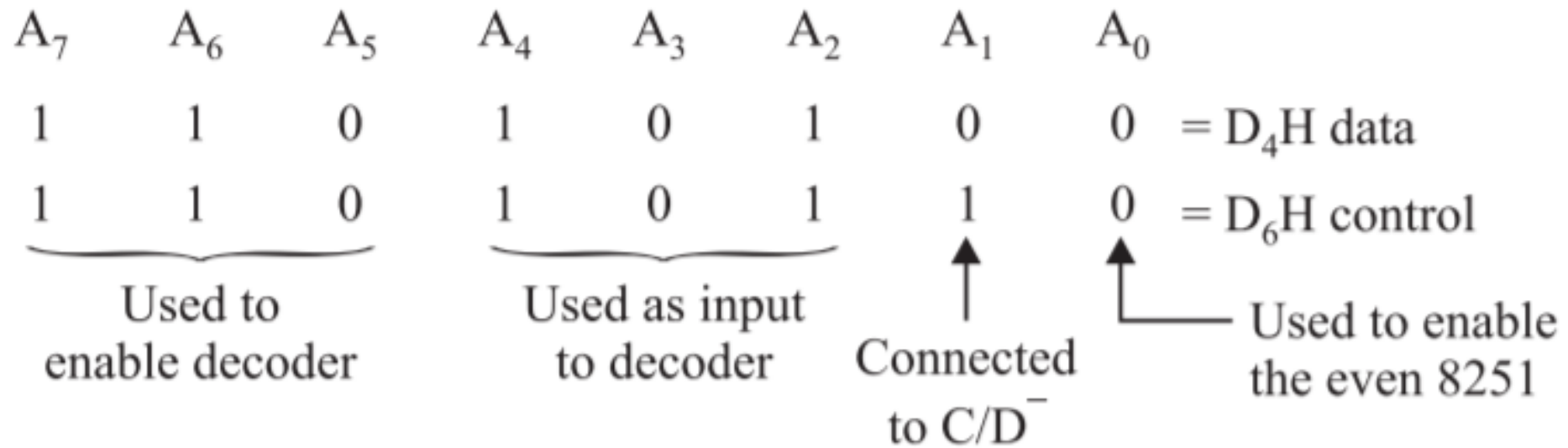
8251 interfacing

- Dva načina:
 - I/O mapirani ulaz/izlaz
 - Memorijki mapirani ulaz-izlaz

Zadatak 2

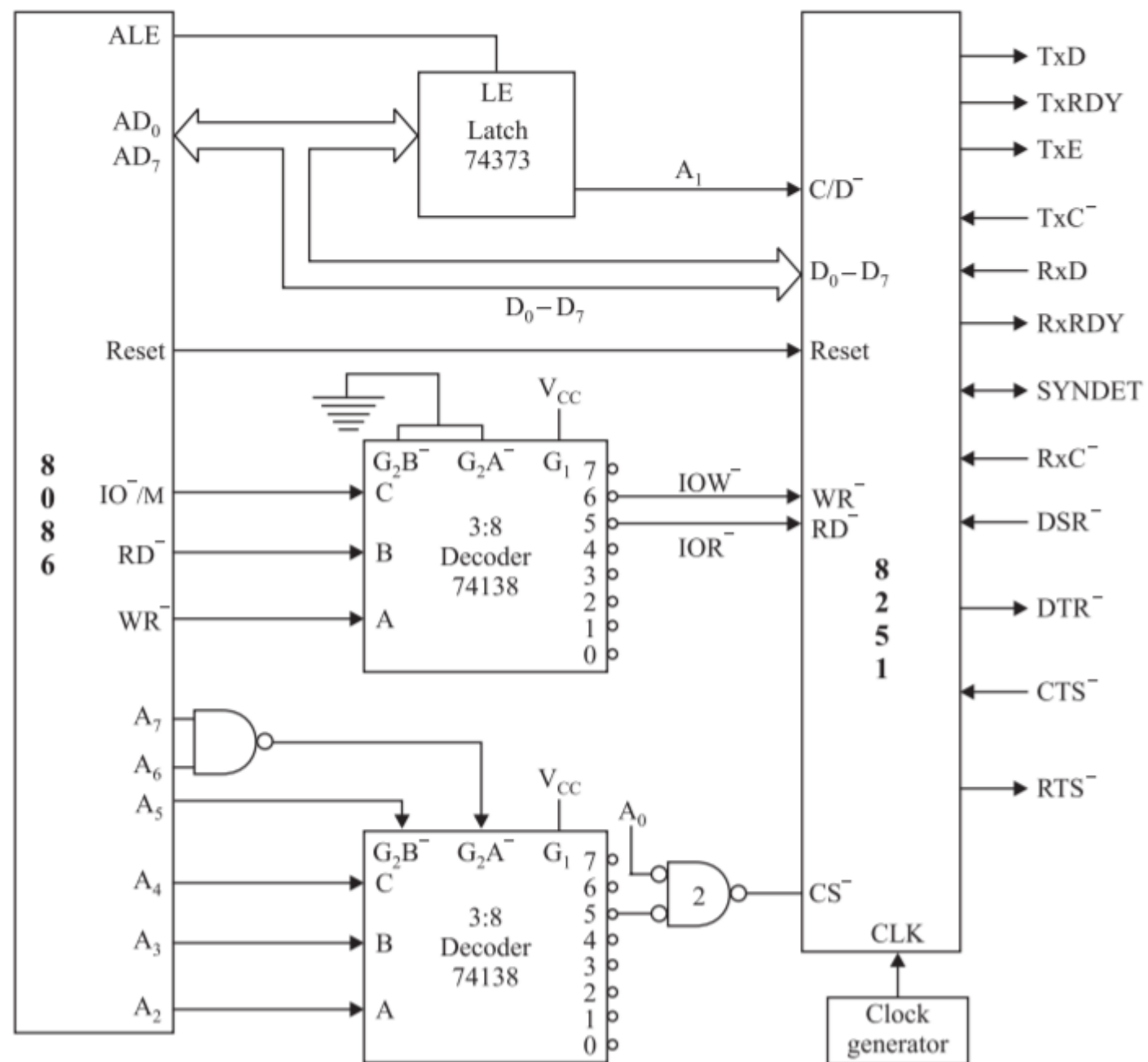
- Za mikroprocesor iAPX 8086 projektovati mikroračunarski sistem za primo-predaju podataka uz pomoć komponente 8251A. Komponentu treba isprogramirati za prijem i predaju po N 8-bitnih podataka bez bita parnosti sa jednim stop bitom, koristeći brzinu od 16x. Prijem i predaju organizovati ispitnim petljama. Komponenta je vezana na U/I adresu D4h. U slučaju greške pri prijemu nastaviti slanje. Za vezivanje 8251 na 8086 se koristi:
 - A) IO mapirani U/I, 8-bit adrese
 - B) Memorijski mapirani U/I, počinje sa adrese 800D4h

Adrese



Zad2 a)

- IO-mapirani ulaz/izlaz



Rešenje – I deo

- Inicijalizacija
- Konfiguracija komponente 8251

```
NAME PROGRAM2
DATA SEGMENT
N DB 100
SEND DB 100 DUP (2Eh) ; bafer za slanje
RECV DB 100 DUP (?) ; bafer za prijem
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START:
MOV AX, DATA
MOV DS, AX
; CLI - zabrana prekida
; Konfiguracija 8251
; Dovedi komponentu 8251 u stanje za prijem komande
; Pogledati dijagram stanja
MOV AL, 00h
MOV DX, D6h
OUT DX, AL
OUT DX, AL
OUT DX, AL
; Resetuj komponentu
; 0100 0000, jedinica za INTERNAL RESET
MOV AL, 40h
OUT DX, AL

; Kontrolna reč MODE 0100 1110
; 1 stop bit, bez bita parnosti, 8b se prenosi, 16x
MOV AL, 4Eh
OUT DX, AL
; Kontrolna reč COMMAND 0001 0101
; Reset error flag i omogućiti prijem i predaju
MOV AL, 15h
OUT DX, AL
; STI
; Brojači na 0
XOR SI, SI
XOR DI, DI
```

Rešenje – II deo

- Slanje
- Prijem

```
PETLJA:
CMP SI, N
JL SLANJE
CMP DI, N
JL PRIJEM
JMP KRAJ

SLANJE:
CMP SI, N ; brojač poslatih
JZ PRIJEM
;čitamo STATUS
MOV DX, D6h
IN AL,DX
TEST AL, 01h ; provera TxRDY == 1
JZ PRIJEM ; ako nije spreman
MOV DX, D4h
MOV AL, SEND[SI]
OUT DX, AL ; prenos u 8251
INC SI ; uvećanje poslatih
```

```
PRIJEM:
CMP DI, N; brojač primljenih
JZ PETLJA
;status
MOV DX, D6h
IN AL, DX
;ako nema greške CONTINUE
TEST AL,38h
JZ CONTINUE
;Simulira kraj prijema
MOV DI,N
;ne skače više na prijem a slanje
nastavlja kao što treba
JMP PETLJA

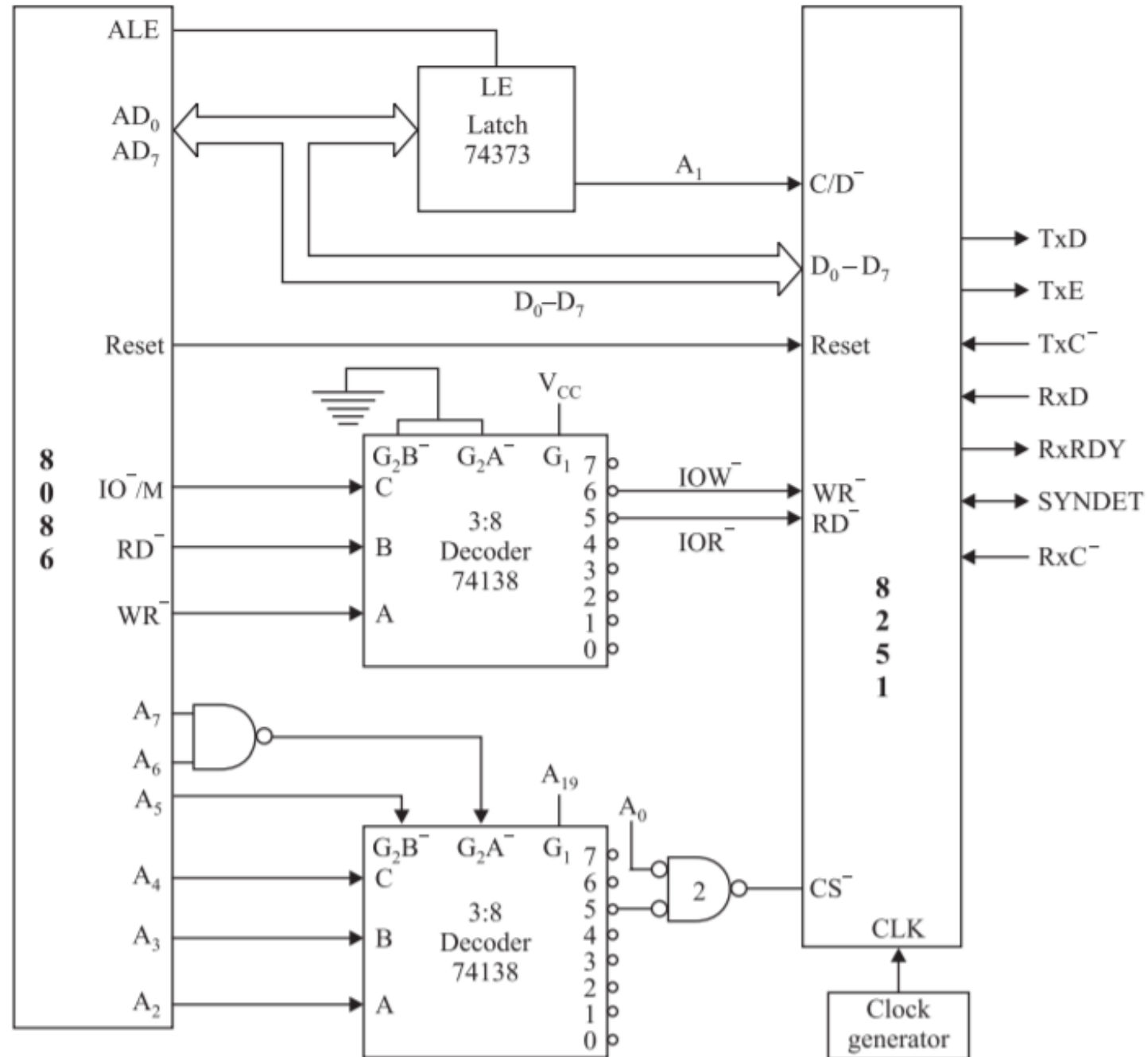
CONTINUE:
TEST AL, 02h ; provera RxRDY == 1
JZ PETLJA
MOV DX, D4h
IN AL, DX ; čitanje podatka
MOV RECV[DI], AL
INC DI ; uvećanje primljenih
JMP PETLJA

KRAJ:
MOV AH, 4Ch
INT 21h
CODE ENDS
END START
END
```

Zad2. b)

- Memorijski mapirani
- DS=8000h
- Offset
 - Za podatke [00D4h]
 - Za komande [00D6h]

$A_{19} \ A_{18} \ \dots \ A_8 \ A_7 \ A_6 \ A_5 \ A_4 \ A_3 \ A_2 \ A_1 \ A_0$
 $1 \ 000 \ \dots \ 0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 0 = 800 \ D4H = \text{Adress of data}$
 $1 \ 000 \ \dots \ 0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 = 800 \ D6H = \text{Address of command}$



Reference

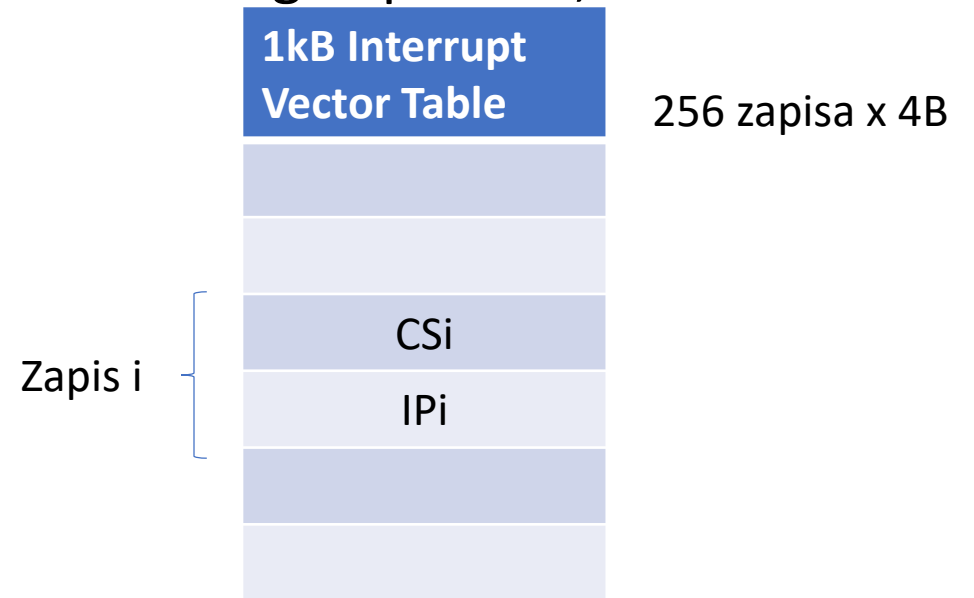
- Sunil Mathur, “MICROPROCESSOR 8086: Architecture, Programming and Interfacing”, str. 325-336, 2011.

Kontroler prekida 8259

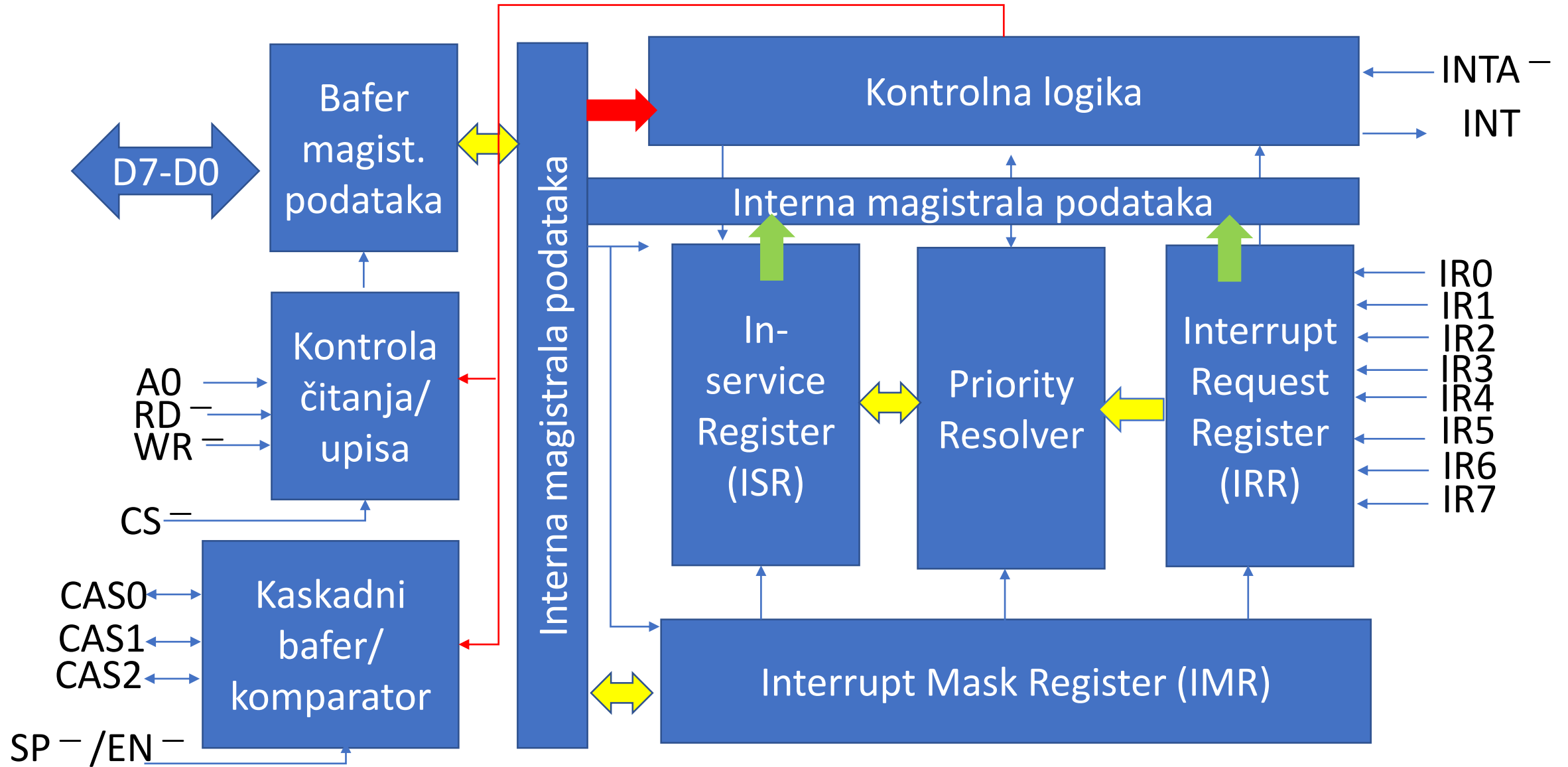
3. termin računskih vežbi

Tabela vektora prekida

- Prvi kilobajt memorije 8086 ima specijalnu namenu – tabela vektora prekida
- Zapis i tabele prekida je podeljen na 2 dela:
 - CS $_i$ koji označava adresu segmenta koda prekidne procedure
 - IP $_i$ koji označava pokazivač instrukcije prekidne procedure
- Zapis u tabeli prekida je veličine 4B, što znači da kada stigne prekid i , biće opslužen procedurom sa adresom $4i$
- Za ovakvu realizaciju prekida koristi se 8259A



Interna struktura 8259A



Blokovi mehanizma prekida

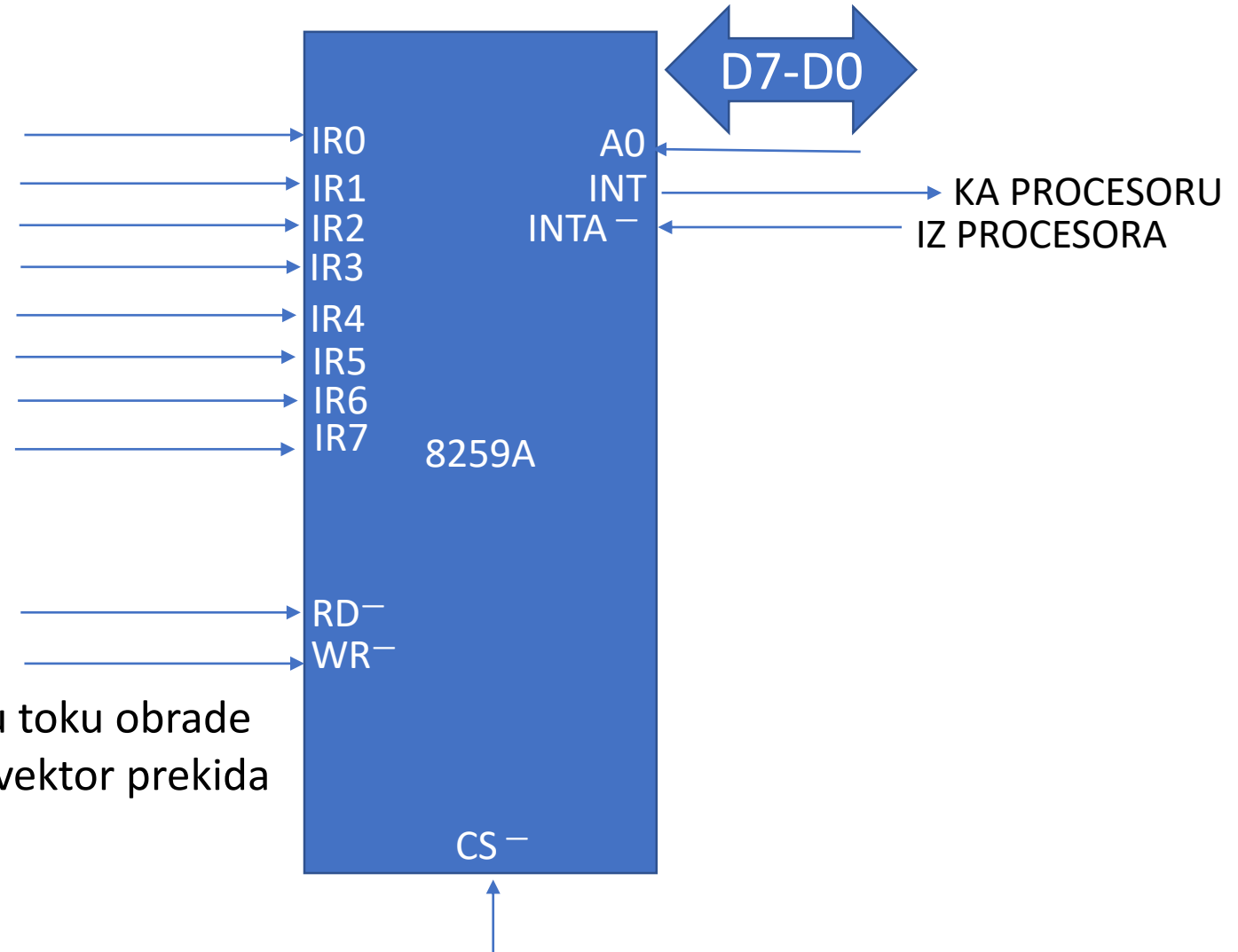
- IMR (Interrupt Mask Register)
- IRR (Interrupt Request Register)
- ISR (Inservice Register)
- Priority Resolver

Opis blokova za prekide

- IRR (Interrupt Request Register)
 - Pamćenje trenutno aktivnih zahteva za prekid
 - Svaki ulaz povezan sa odgovarajućim bitom
 - Bez obzira na mod, ulaz IR mora ostati aktivan do silazne ivice prvog impulsa INTA signala
- Priority Resolver
 - Mehanizam prioriteta prekida
- ISR (Inservice Register)
 - Za svaki ulaz po jedan bit
 - Zahtev sa i-tog ulaza se opslužuje => i-ti bit registra postavljen na 1
- IMR (Interrupt Mask Register)
 - Služi za maskiranje pojedinih ulaza
 - Redni broj bita jednak broju ulaza
 - 0 – demaskiranje
 - 1 – maskiranje

Najbitniji pinovi 8259A

- D0-D7
 - Šalje vektora prekida ka 8086
- A0
 - Adresiranje internih registara
- INT
 - Prekida proces
- INTA
 - Potvrda prekida
 - Dva impulsa:
 - Prvi: 8259 zabeleži da je prekid u toku obrade
 - Drugi: kaže mu da na liniji stavi vektor prekida



Sekvenca prihvatanja prekida

- Stiže zahtev na IR
- Šalje se zahtev na INT
- Procesor šalje na INTA 8259 prvi negativni impuls
 - Silazna ivica zamrzava stanje u svim 8259A da bi se prioriteti pravilno izračunali
 - Na uzlaznu ivicu master postavlja CAS linije
- U ISR se postavlja bit najvišeg prioriteta iz IRR (0 najveći prioritet) - koji pripada kontroleru čiji će se zahtev opslužiti
- Procesor šalje na INTA drugi negativni impuls
 - Odgovarajući 8259A šalje broj ulaza za zahtev koji je upravo prihvaćen
- Odgovarajući ISR bit se briše automatski u ovom trenutku ili se čeka odgovarajuća EOI komanda

Kontrolne reči kod 8259

- ICW1

- ICW2

- ICW3

- ICW4

- OCW1

- OCW2

- OCW3

ICW1

- A0=0

A7	A6	A5	1	LTIM	ADI	SNGL	IC4
A7-A5 adrese vektora interapta (samo za 80/85 mod) Za 8086 nebitno – stavljamo sve 0				0:Okidanje na ivicu (uzlaznu) 1:Okidanje na nivo (jedinica)	Nije relevantno za 8086	0:više od 1, kaskadno 1: jedan 8259	0: ICW4 nije potreban 1: ICW4 potreban

ICW2

- A0=1
- A7-A3: viših 5 bitova interrupt vektora
- D2-D0: setuje ih 8259 i predstavljaju redni broj IR linije na kojoj je prekid došao

Interrupt	A7	A6	A5	A4	A3	D2	D1	D0
IR0						0	0	0
IR1						0	0	1
IR2						0	1	0
IR3						0	1	1
IR4						1	0	0
IR5						1	0	1
IR6						1	1	0
IR7						1	1	1

ICW3

- Master, A0=1

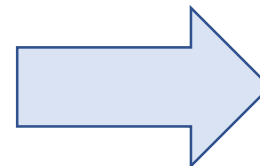
- informacija na kojim linijama postoji slave
- 1 : na odgovarajućem ulazu je slave

S7	S6	S5	S4	S3	S2	S1	S0

- Slave, A0=1

- Njegova adresa se pojavljuse na CAS linijama
- Odgovara rednom broju linije na masteru na koju je taj slave povezan

0	0	0	0	0	ID2	ID1	ID0



Interrupt	ID2	ID1	ID0
IR0	0	0	0
IR1	0	0	1
IR2	0	1	0
IR3	0	1	1
IR4	1	0	0
IR5	1	0	1
IR6	1	1	0
IR7	1	1	1

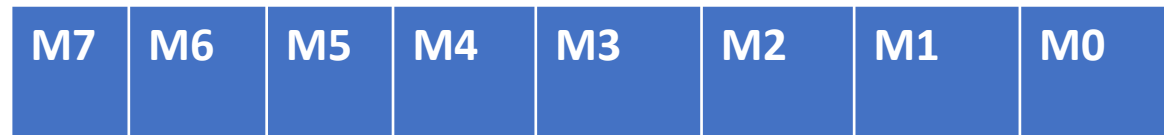
ICW4

- A0=1
- AEOI – Govori o načinu završetka prekida, tj. kako da procesor obavesti 8259 da je obrada prekida završena na strani procesora.
 - 1 (auto): aktivira automatsko brisanje bita u ISR registru po prihvatanju odgovarajućeg zahteva (uzlazna ivica drugog INTA impulsa)
 - 0 (normal): radi se o softverskom prekidu i podrazumeva se slanje OCW2 kontrolne reči neposredno pre instrukcije IRET

D7	D6	D5	SFNM	BUF	M/S	AEOI	mPM
0	0	0	0: NOT SPECIAL FULLY NESTED 1: SPECIAL FULLY NESTED	0X – NON BUFFERED 10 – BUFFERED MODE/SLAVE 11 – BUFFERED MODE/MASTER		0: NORMAL 1: AUTO	0: MCS-80/85 MODE 1: 8086/8088 MODE

OCW1

- A0=1
- Jedinica na poziciji i označava da je prekid sa IRi zabranjen
- 1 – MASK SET
- 0 – MASK RESET



OCW2

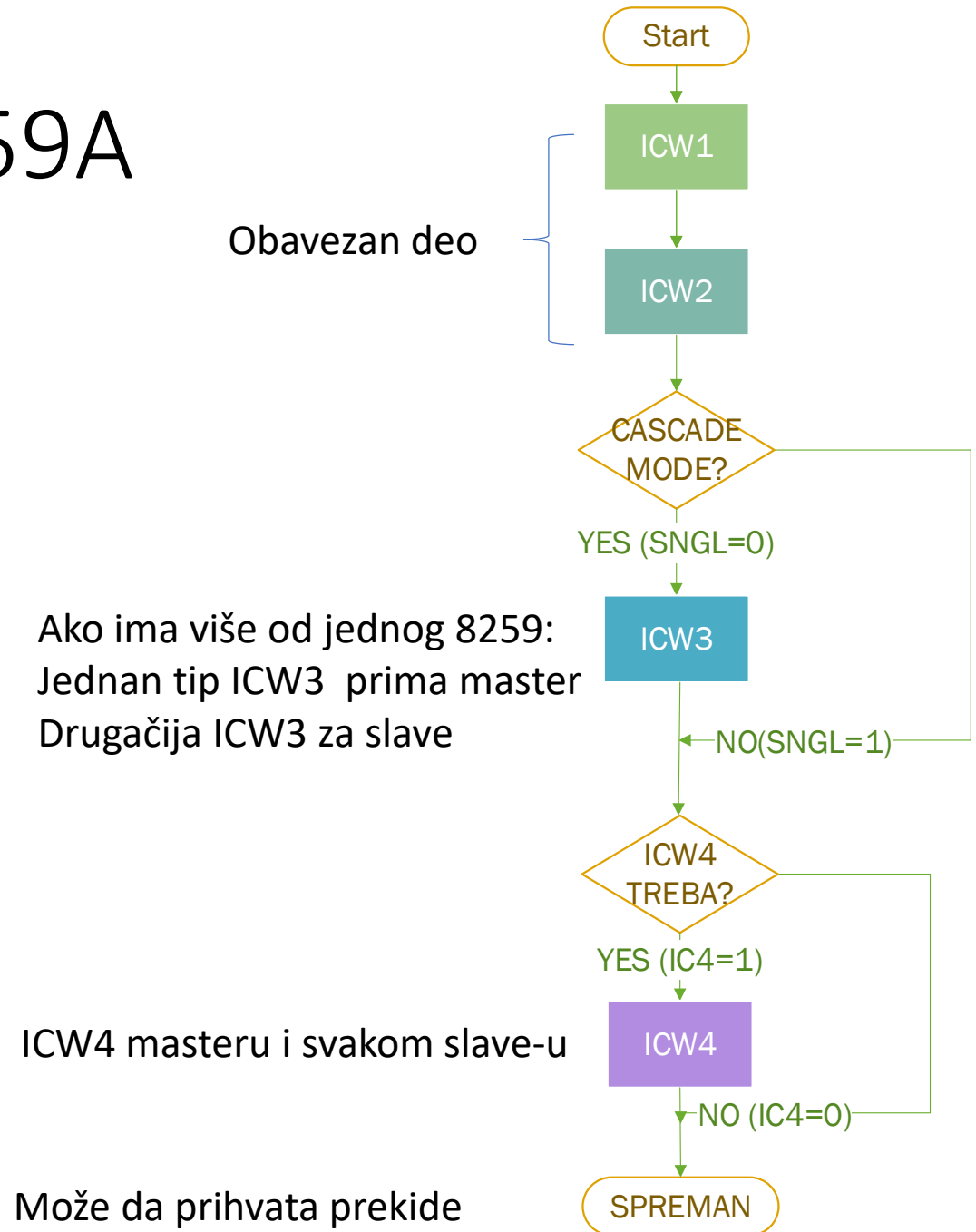
- $A_0=0$

	R	SL	EOI	0	0	L2	L1	L0
END OF INTERRUPT	000 – NON-SPECIFIC EOI COMMAND					IR nivo (0-7) na koji treba da reaguje		
AUTO-ROTATION	011 - SPECIFIC EOI COMMAND							
	101 – ROTATE ON SPECIFIC EOI COMMAND							
	100– ROTATE IN AUTOMATIC EOI MODE (SET)							
	000 – ROTATE IN AUTOMATIC EOI MODE (CLEAR)							
SPECIFIC ROTATION	111- ROTATE ON SPECIFIC EOI COMMAND							
	110 – SET PRIORITY COMMAND							
	010 – NO OPERATION							

OCW2 - objašnjenje

- Dve vrste komandi
 - NON-SPECIFIC – odnosi se na upravo odsluženi zahtev (najvećeg prioriteta)
 - SPECIFIC – zadaje se i broj ulaza na koji se komanda odnosi
 - Način da kontroler sazna kada se koja rutina završila i da osnovu toga radi sa prioritetima
- EOI:
 - Komanda za brisanje odgovarajućeg bita u ISR
 - Način da kontroler sazna kada se koja rutina završila i ispravno radi sa prioritetima
- Automatsko rotiranje:
 - U AEIOI modu, nakon opsluživanja dolazi do rotiranja prioriteta, taj nivo postaje najnižeg prioriteta.
- Specifično rotiranje tako da zadati ulaz postane najnižeg prioriteta
- Ako je upravo odslužen i-ti prioriteti su (od najvišeg ka najnižem)
 - $i+1, i+2, \dots, 7, 0, 1, \dots, i$

Sekvenca inicijalizacije 8259A



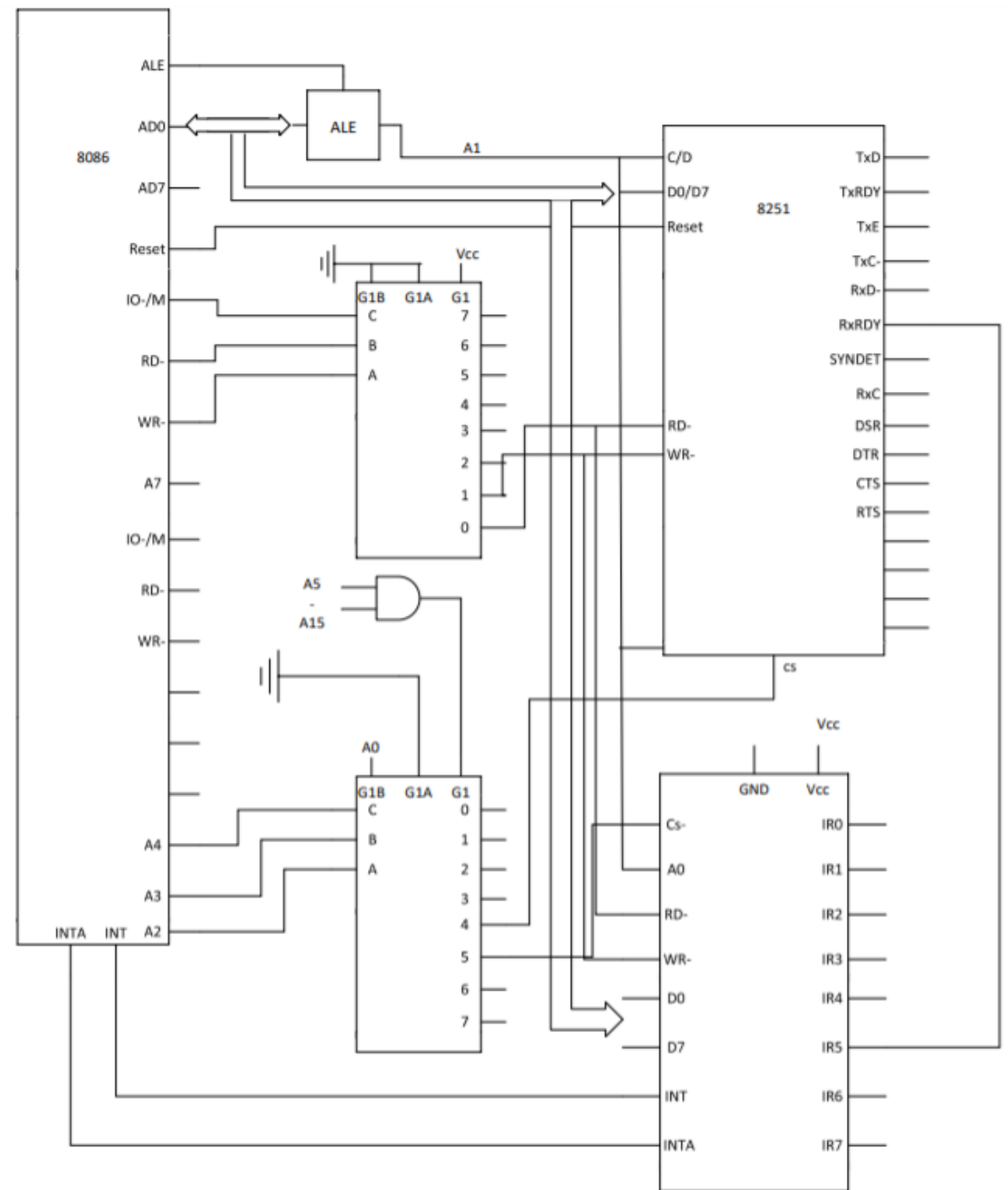
Zadatak 3

- Za mikroprocesor iAPX 8086 projektovati mikroračunarski sistem za predaju i prijem podataka koristeći komponentu 8251A. Podaci se šalju od adrese SEND, a primaju od adrese REC. Broj podataka za predaju i prijem je 64. Komponentu treba isprogramirati za istovremeni prijem i predaju 5-bitnih podataka sa parnim bitom parnosti, 1.5 stop bitom i brzinom 16x. Predaju organizovati ispitnom petljom, a prijem preko prekida 173, koji se generiše na priključku RxRDy. Komponenta 8251 nalazi se na U/I adresi 0xFFFF0.

Adrese

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	8251 (FFF0)
1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	8251 (FFF2)
1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	8259 (FFF4)
1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	8259 (FFF6)

Šema povezivanja



Mode word (8251)

- 0xB2

S2	S1	EP	PEN	L2	L1	B2	B1
Broj stop bitova: 01 – 1 stop bit 10 – 1.5 stop bit 11 – 2 stop bita		Bit parnosti: 00 – disable 01 – neparna 10 – disable 11- parna		Broj bitova po karakteru: 00-5b 01-6b 10 -7b 11-8b		Baud rate factor: 00-Sync 01-1x 10-16x 11-64x	
1	0	1	1	0	0	1	0

Command word (8251)

- 0x15

EH	IR	RTS	ER	SBRK	RXE	DTR	TXEN
0:Normal 1:Hunt mode	0:Normal 1:Internal reset	0:DTR->1 1: DTR->0	0:Normal 1: Reset error flag	0:Normalan režim 1:Slanje karaktera prekida	Omogućiti prijemnik	0:DTR->1 1: DTR->0	Omogućiti predajnik
0	0	0	1	0	1	0	1

ICW1

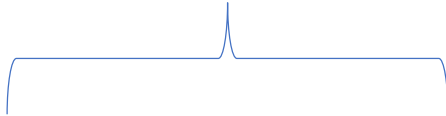
- 0x13

A7	A6	A5	1	LTIM	ADI	SNGL	IC4
A7-A5 adrese vektora interapta (samo za 80/85 mod) Za 8086 nebitno – stavljamo sve 0				0:Okidanje na ivicu (uzlaznu) 1:Okidanje na nivo (jedinica)	Nije relevantno za 8086	0:više od 1, kaskadno 1: jedan 8259	0: ICW4 nije potreban 1: ICW4 potreban
0	0	0	1	0	0	1	1

ICW2

- 173 = 10101101(bin)
- 0xA8

Nije relevantno u toku
inicijalizacije



Interrupt	A7	A6	A5	A4	A3	D2	D1	D0
IR0	1	0	1	0	1	0	0	0
IR1						0	0	1
IR2						0	1	0
IR3						0	1	1
IR4						1	0	0
IR5						1	0	1
IR6						1	1	0
IR7						1	1	1

ICW4

- 0x03

D7	D6	D5	SFNM	BUF	M/S	AEOI	mPM
0	0	0	0: NOT SPECIAL FULLY NESTED 1: SPECIAL FULLY NESTED	0X – NON BUFFERED 10 – BUFFERED MODE/SLAVE 11 – BUFFERED MODE/MASTER		0: NORMAL 1: AUTO	0: MCS-80/85 MODE 1: 8086/8088 MODE
0	0	0	0	0	0	1	1

OCW1

- 0xDF
- 173 = 10101101(bin)
- 168+5=173

M7	M6	M5	M4	M3	M2	M1	M0
1	1	0	1	1	1	1	1

Program – inicijalizacija i konfiguracija

```
EXTRN PREKID:FAR ;pomocna procedura za obradu prekida
PUBLIC RECV, BRULAZ ;vidljivo i iz drugih modula
```

```
DATA SEGMENT
```

```
    BRULAZ DB 0
```

```
    ;baferi primljenih i poslatih podataka
```

```
    SEND DB 64 DUP (3Fh)
```

```
    RECV DB 64 DUP (?)
```

```
DATA ENDS
```

```
STEK SEGMENT
```

```
    BOS DW 256 DUP (?)
```

```
    TOS LABEL WORD
```

```
STEK ENDS
```

```
CODE SEGMENT
```

```
ASSUME CS:CODE, DS:DATA, SS:STEK
```

```
START:
```

```
    ;inicijalizacija segmenata
```

```
    MOV AX, DATA
```

```
    MOV DS, AX
```

```
    MOV AX, STEK
```

```
    MOV SS, AX
```

```
    LEA SP, TOS
```

```
    ;inicijalizacija komponenti
```

```
    CLI
```

```
    ;8251
```

```
    MOV AL, 00h
```

```
    MOV DX, 0FFF2h
```

```
    OUT DX, AL
```

```
    OUT DX, AL
```

```
    OUT DX, AL
```

```
    MOV AL, 40h
```

```
    OUT DX, AL
```

```
    MOV AL, B2h ;slanje mode kontrolne reci
```

```
    OUT DX, AL
```

```
    MOV AL, 15h ;slanje komandne kontrolne reci
```

```
    OUT DX, AL
```

```
    ;8259
```

```
    MOV DX, 0FFF4h
```

```
    MOV AL, 13h ;ICW1, A0=0
```

```
    OUT DX, AL
```

```
    MOV DX, 0FFF6h
```

```
    MOV AL, 0A8h ;ICW2
```

```
    OUT DX, AL
```

```
    MOV AL, 03h ;ICW4
```

```
    OUT DX, AL
```

```
    MOV AL, 0DFh ;OCW1
```

```
    OUT DX, AL
```

```
    ;inicijalizacija tabele vektora prekida
```

```
    MOV AX, OFFSET PREKID
```

```
    MOV [173*4], AX
```

```
    MOV AX, SEG PREKID
```

```
    MOV [173*4+2], AX
```

```
    MOV CX, 64
```

```
    MOV SI, 0
```

```
    STI ;dozvola prekida
```


Program - glavna petlja

```
PETLJA:
CEKAJ:
    MOV DX, 0FFF2h
    IN AL, DX ;AL -- status
    TEST AL, 38h ;test na gresku koja se moze javiti u prijemu
    JNZ GRESKA
    TEST AL, 01h ;TxRDY == 1?
    JZ CEKAJ ;ako nije spreman, cekaj da bude
    MOV DX, 0FFF0h
    MOV AL, SEND[SI]
    OUT DX, AL
    INC SI
    LOOP PETLJA ;64 prolaza
CEKAJ1: ;cekanje na prekidnu proceduru
    CMP BRULAZ 64
    JL CEKAJ1
    MOV DX, 0FFF2h ;testiranje na gresku
    IN AL, DX
    TEST AL, 38h
    JZ KRAJ

GRESKA: ;kod za upravljanje greskom
KRAJ:
    MOV AH, 4Ch
    INT 21h
CODE ENDS
END START
```

Eksterna procedura za prekide

```
;Radi paralelno i vrsi prijem podataka
EXTRN RECV:BYTE, BRULAZ:BYTE
PUBLIC PREKID
PROCED SEGMENT
PREKID PROC FAR
ASSUME CS:PROCED
    ;cuvanje konteksta procesora
    PUSHF; automatski se stavlja na stek
    PUSH AX
    PUSH DI
    PUSH DX
    MOV DX, 0FFF0h ;bez testiranja na spremnost jer je RxRDY sigurno 1
    IN AL, DX
    MOV DI, BRULAZ
    MOV RECV[DI], AL
    INC BRULAZ ;uvecanje brojaca primljenih
    ;vracanje konteksta procesora
    POP DX
    POP DI
    POP AX
    POPF
    IRET ;vracanje iz procedure
PREKID ENDP
PROCED ENDS
```

Zadatak 4

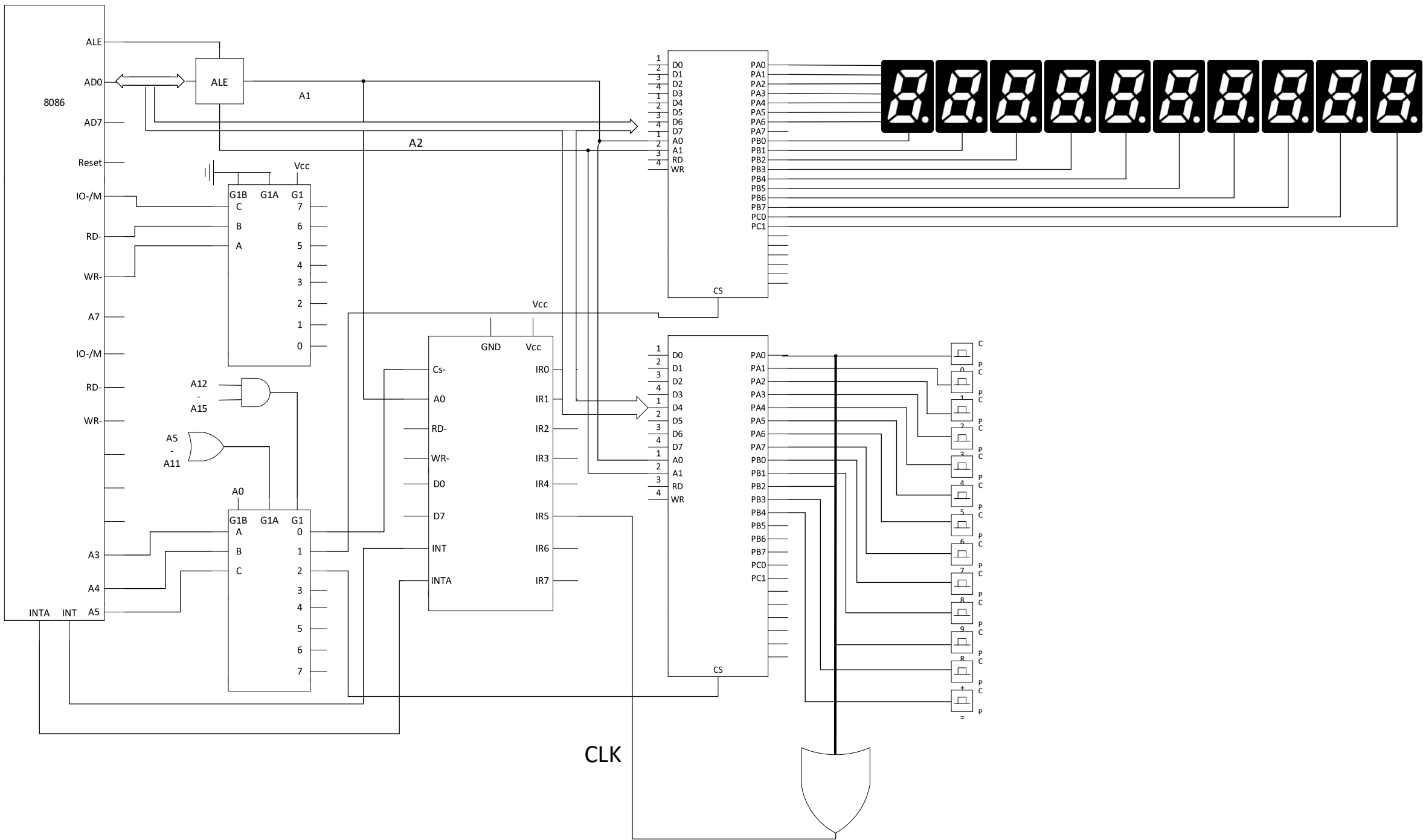
- Projektovati mikroračunarski sistem za kasu na bazi mikroprocesora iA PX8086. Sistem sadrži 10 sedmosegmentnih displeja i tastaturu sa tasterima od 0 do 9, +, = i RESET taster. Pritiskom na RESET briše se trenutno stanje, dok pritiskom na = generiše prikaz zbira svih unetih iznosa. Napisati procedure koje će na displejima ispisivati ono što je otkucano na tastaturi, kao i ukupan iznos za plaćanje. Komponenta 8259 je na adresi 0F000h, a redni broj prekida najvišeg prioriteta je 117. Za realizaciju displeja koristiti tehniku osvežavanja pri čemu jedan pin odgovara jednom tasteru.

Adrese

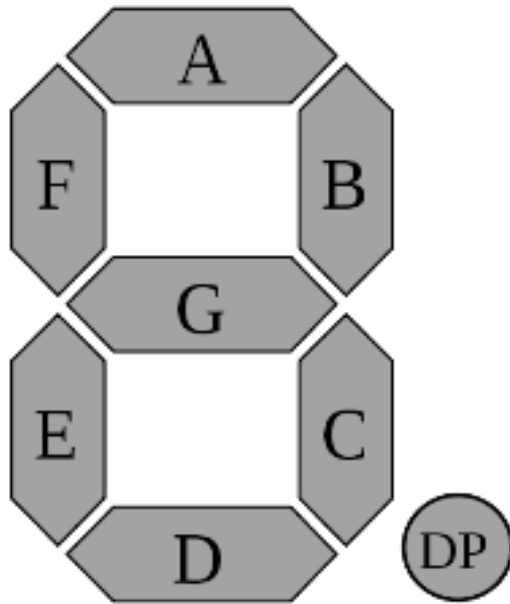
Komponenta	$A_{15} - A_{12}$	$A_{11} - A_8$	$A_7 - A_4$	$A_3 - A_0$	
8259	1111	0000	0000	0000	F000h
	1111	0000	0000	0010	F002h
	1111	0000	0000	1100	F00Ch PC
8255-1	1111	0000	0000	1110	F00Eh CTRL
	1111	0000	0000	1000	F008h PA
	1111	0000	0000	1010	F00Ah PB
	1111	0000	0001	0100	F014h PC
8255-2	1111	0000	0001	0110	F016h CTRL
	1111	0000	0001	0000	F010h PA
	1111	0000	0001	0010	F012h PB

Povezivanje

- 7s displeji: 8255-1
 - Segmenti povezani na PORTA A0-A6
 - Pojedinačni displeji povezani na PORTB (B0-B7) i PORTC (C0 i C1)
- Tasteri: 8255-2
 - 0-7 povezani na PORTA
 - 8, 9, RESET, +, = na PORTB



7s displej



PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0		
	a	b	c	d	e	f	g	code	value
0	1	1	1	1	1	1	0	7Eh	0
0	0	1	1	0	0	0	0	30h	1
0	1	1	0	1	1	0	1	6Dh	2
0	1	1	1	1	0	0	1	79h	3
0	0	1	1	0	0	1	1	33h	4
0	1	0	1	1	0	1	1	5Bh	5
0	1	0	1	1	1	1	1	5Fh	6
0	1	1	1	0	0	0	0	70h	7
0	1	1	1	1	1	1	1	7Fh	8
0	1	1	1	1	0	1	1	7Bh	9

Rešenje

- DEFC:
 - kodovi cifara
- POZC:
 - gde je unos cifre
- DISPLAY:
 - niz prikazanih cifara
 - Ako je 1357, uneto u mem DISPLAY je 7531
- ROTOR:
 - koji će displej biti osvežen

```
EXTRN Taster_proc:far, Osvez_proc:far
PUBLIC DISPLAY, ZBIR, ROTOR, DEFC, POZC
```

```
DATA segment
```

```
    DEFC db 7eh, 30h, 6dh, 79h, 33h, 5bh,
    5fh, 70h, 7fh, 7bh
```

```
    DISPLAY db 10 dup (0)
```

```
    ZBIR db 10 dup (0)
```

```
    ROTOR dw 0
```

```
    POZC dw 0
```

```
DATA ends
```

```
STACK segment
```

```
    bos dw 256 dup (?)
```

```
    tos label word
```

```
STACK ends
```

```
CODE segment
```

```
    assume CS:CODE, DS:DATA, SS:STACK
```

```
start:
```

```
    mov AX, DATA
```

```
    mov DS, AX
```

```
    mov AX, STACK
```

```
    mov SS, AX
```

```
    lea SP, tos
```

```
    cli
```


8259 inicijalizacija

```
; 8259 inic.  
; ICW1  
mov DX, 0F000h  
mov AL, 13h  
out DX, AL  
; ICW2  
mov DX, 0F002h  
mov AL, 70h  
out DX, AL  
; ICW4  
mov AL, 03h  
out DX, AL  
; OCW1  
mov AL, 5Fh  
out DX, AL
```

ICW1

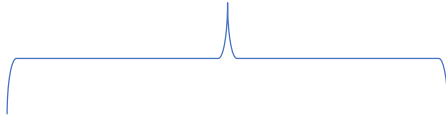
- 0x13

A7	A6	A5	1	LTIM	ADI	SNGL	IC4
A7-A5 adrese vektora interapta (samo za 80/85 mod) Za 8086 nebitno – stavljamo sve 0				0:Okidanje na ivicu (uzlaznu) 1:Okidanje na nivo (jedinica)	Nije relevantno za 8086	0:više od 1, kaskadno 1: jedan 8259	0: ICW4 nije potreban 1: ICW4 potreban
0	0	0	1	0	0	1	1

ICW2

- 117 = 01110101(bin)
- 0x70

Nije relevantno u toku
inicijalizacije



Interrupt	A7	A6	A5	A4	A3	D2	D1	D0
IR0	0	1	1	1	0	0	0	0
IR1						0	0	1
IR2						0	1	0
IR3						0	1	1
IR4						1	0	0
IR5						1	0	1
IR6						1	1	0
IR7						1	1	1

ICW4

- 0x03

D7	D6	D5	SFNM	BUF	M/S	AEOI	mPM
0	0	0	0: NOT SPECIAL FULLY NESTED 1: SPECIAL FULLY NESTED	0X – NON BUFFERED 10 – BUFFERED MODE/SLAVE 11 – BUFFERED MODE/MASTER		0: NORMAL 1: AUTO	0: MCS-80/85 MODE 1: 8086/8088 MODE
0	0	0	0	0	0	1	1

OCW1

- 0x5F
- Taster – viši prioritet
 - 117 = 01110101(bin)
- Refresh - niži
 - 119 = 01110111(bin)
- $112 + 5 = 117$
- $112 + 7 = 119$

M7	M6	M5	M4	M3	M2	M1	M0
0	1	0	1	1	1	1	1

Inicijalizacija 8255-1 i 8255-2

```
    ; 8255-1 inic.  
    mov DX, 0F00Eh  
    mov AL, 80h  
    out DX, AL  
    ; 8255-2 inic.  
    mov DX, 0F016h  
    mov AL, 92h  
    out DX, AL
```

Struktura kontrolne reči 8255-1

- Displej – output A, B i C
- 0x80

D7	D6	D5	D4	D3	D2	D1	D0
0- SET/RESET	00 – MODE0		0 -PORTA OUT	0 – PORTC HIGHER OUT	0 - MODE0	0 – PORTB OUT	0 – PORTC LOWER OUT
1 - I/O MODE	01- MODE1						
	1X – MODE2		1 – PORTA IN	1 – PORTC HIGHER IN	1 - MODE1	1- PORTB IN	1 - PORTC LOWER IN
	MOD GRUPE A		GRUPA A		MOD GRUPE B	GRUPA B	
1	0	0	0	0	0	0	0

Struktura kontrolne reči 8255-2

- Tastatura – input A i B
- 0x92

D7	D6	D5	D4	D3	D2	D1	D0
0- SET/RESET	00 – MODE0 01- MODE1 1X – MODE2 MOD GRUPE A		0 -PORTA OUT	0 – PORTC HIGHER OUT	0 - MODE0	0 – PORTB OUT	0 – PORTC LOWER OUT
1 - I/O MODE			1 – PORTA IN	1 – PORTC HIGHER IN	1 - MODE1	1- PORTB IN	1 - PORTC LOWER IN
			GRUPA A		MOD GRUPE B		GRUPA B
1	0	0	1	0	0	1	0

Inicijalizacija tabele prekida

- $4 * 117 = 468$
- $4 * 117 + 2 = 470$
- $117 + 2 = 119$
- $119 * 4 = 476$
- $119 * 4 + 2 = 478$

```
; Tabela prekida inic.  
mov [468], OFFSET Taster_proc  
mov [470], SEG Taster_proc  
mov [476], OFFSET Osvez_proc  
mov [478], SEG Osvez_proc  
sti  
main:  
    jmp main  
mov AX, 4C02h int 21h  
CODE ends  
end start
```

Osvežavanje displeja

- Prekidne i pomoćne procedure

```
; prekidne i pomocne procedure
PUBLIC Osvez_proc, Taster_proc
EXTRN DISPLAY:byte, ZBIR:byte, ROTOR:byte, DEFC:byte, POZC:word

PCODE SEGMENT
    assume CS: PCODE
```

- Prekidna procedura za osvežavanje

- ROTOR

- Shift jedinice za broj pozicija ROTOR
- 0x0200-0000 0010 0000 0000

```
Osvez_proc proc far
    pushf
    push AX
    push CX
    push DX
    push SI
    cli
    mov AL, ROTOR
    inc AL
    cmp AL, 10
    jl daljel
    mov AL, 0
daljel:
    mov ROTOR, AL
    mov AH, 0
    mov SI, AX
    mov AL, DISPLAY[SI]
    mov SI, AX
    mov AL, DEFC[SI]
    mov DX, 0F008h
    out DX, AL
    mov AX, 0200h
    mov CL, ROTOR
    shr AX, CL
    mov DX, 0F00Ah
    out DX, AL
    mov DX, 0F00Ch
    mov AL, AH
    out DX, AL
    sti
    pop SI
    pop DX
    pop CX
    pop AX
    popf
    iret
Osvez_proc endp
```

Pritisak tastera

- Prekidna procedura
- Prva petlja
 - Da li taster na PORTA?
 - SHIFT udesno i traži poziciju 1
 - BX čuva vrednost od 0 do 7
 - 0-7
- PORTB
 - 8, 9, RESET, +, =
 - Posebne labela

```
Taster_proc proc far
    assume CS: PCODE
    pushf
    push AX
    push DX
    push BX
    cli
    mov DX, 0F010h
    in AL, DX
    cmp AL, 0
    je portb
    mov BX, 0
    poredi:
        cmp AL, 1
        je nasao
        shr AL, 1
        inc BX
        jmp poredi
    nasao:
        call Cifra_proc
        jmp kraj
    portb:
        mov DX, 0F012h
        in AL, DX
        cmp AL, 1
        je losam
        cmp AL, 2
        je ldevet
        cmp AL, 4
        je lreset
        cmp AL, 8
        je lplus
        jmp ljednako
    losam:
        mov BX, 8
        call Cifra_proc
        jmp kraj
    ldevet:
        mov BX, 9
        call Cifra_proc
        jmp kraj
    lplus:
        call Plus_proc
        jmp kraj
    ljednako:
        call Jednako_proc
        jmp kraj
    lreset:
        call Reset_proc
    kraj:
        sti
        pop BX
        pop DX
        pop AX
        popf
        iret
Taster_proc endp
```

Cifra

- Pomoćna procedura
- Umeće cifru na prvo
- Ostatak SHIFT udesno (DISPLAY)

```
Cifra_proc proc near
pushf
push SI
push AX
mov SI, POZC
cmp SI, 10
je kraj
petlja:
    cmp SI, 0
    je novacifra
    dec SI
    mov AL, DISPLAY[SI]
    inc SI
    mov DISPLAY[SI], AL
    dec SI
    jmp petlja
novacifra:
    mov DISPLAY[SI], BL
    inc POZC
kraj:
    pop AX
    pop SI
    popf
    ret
Cifra_proc endp
```

Reset

- Pomoćna procedura

```
Reset_proc proc near
pushf
push SI
push CX
xor SI, SI
mov CX, 10
petlja:
    mov byte ptr DISPLAY[SI], 0
    mov byte ptr ZBIR[SI], 0
    inc SI
loop petlja

pop CX
pop SI
popf ret
Reset_proc endp
```

Jednako

- Pomoćna procedura
- Poslednje uneto doda na prethodi zbir i kopira niz zbir u DISPLAY

```
Jednako_proc proc near
pushf
push SI
push AX
push CX
call Plus_proc
xor SI, SI
mov CX, 10
petlja:
    mov AL, ZBIR[SI]
    mov DISPLAY[SI], AL
    inc SI
loop petlja
pop CX
pop AX
pop SI
popf
ret
Jednako_proc endp
```

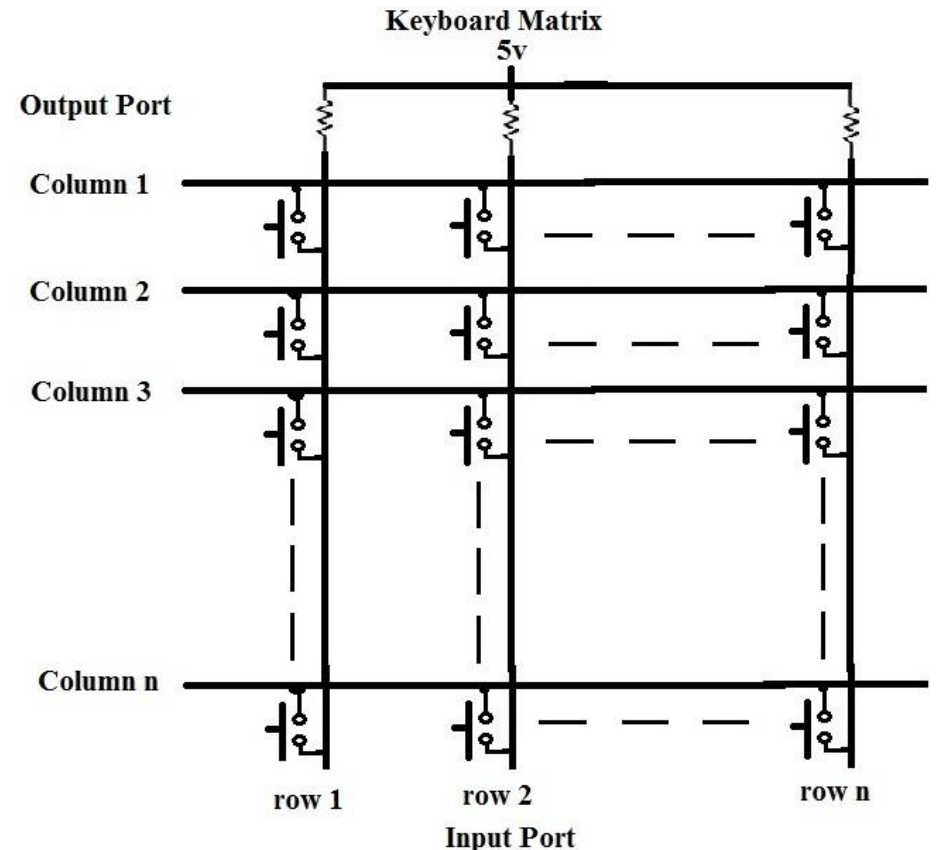
Plus

- Pomoćna procedura
- Sabira do tada sakupljeno i poslednje uneto

```
Plus_proc proc near
    pushf
    push AX
    push SI
    xor SI, SI
    mov AH, 0
    petlja:
        mov AL, DISPLAY[SI]
        add AL, ZBIR[SI]
        add AL, AH
        mov AH, 0
        cmp AL, 10
        jl v2
        mov AH, 1
        sub AL, 10
        v2:
        mov ZBIR[SI], AL
        mov DISPLAY[SI], 0
        inc SI
        cmp SI, 9
    jbe petlja
    mov POZC, 0
    pop SI
    pop AX
    popf
    ret
Plus_proc endp
PCODE ends END
```

Druga varijanta:SCAN algoritam matrice tastature

- U prethodnom primeru za $m \times n$ tastera $\rightarrow m \times n$ pinova
- Za matricnu tastaturu
 - $m \times n$ tastera $\rightarrow m + n$ pinova
- Princip rada:
 - Kada ništa nije pritisnuto, sve kolone 1
 - Kada je taster pritisnut \rightarrow ta kolona 0
 - Za proveru dovoljno jedna 0
 - Potrebno je utvrditi koji taster je to
 - Prvo se detektuje kolona, pa vrsta
 - Taster može biti samo u jednoj vrsti, pa mikroprocesor izbacuje 0 samo u toj jednoj vrsti
 - Ako se nađe 0 na ulaznom portu, mikroprocesor zna da je u toj vrsti pritisnut taster
 - Obrnuto, ako ulazni port ima sve 1, pritisnuti taster nije u toj vrsti, pa mikroprocesor bira sledeću vrstu dok ne nađe. Onda bi se kolona identifikovala (jednom) nulom na ulaznom portu

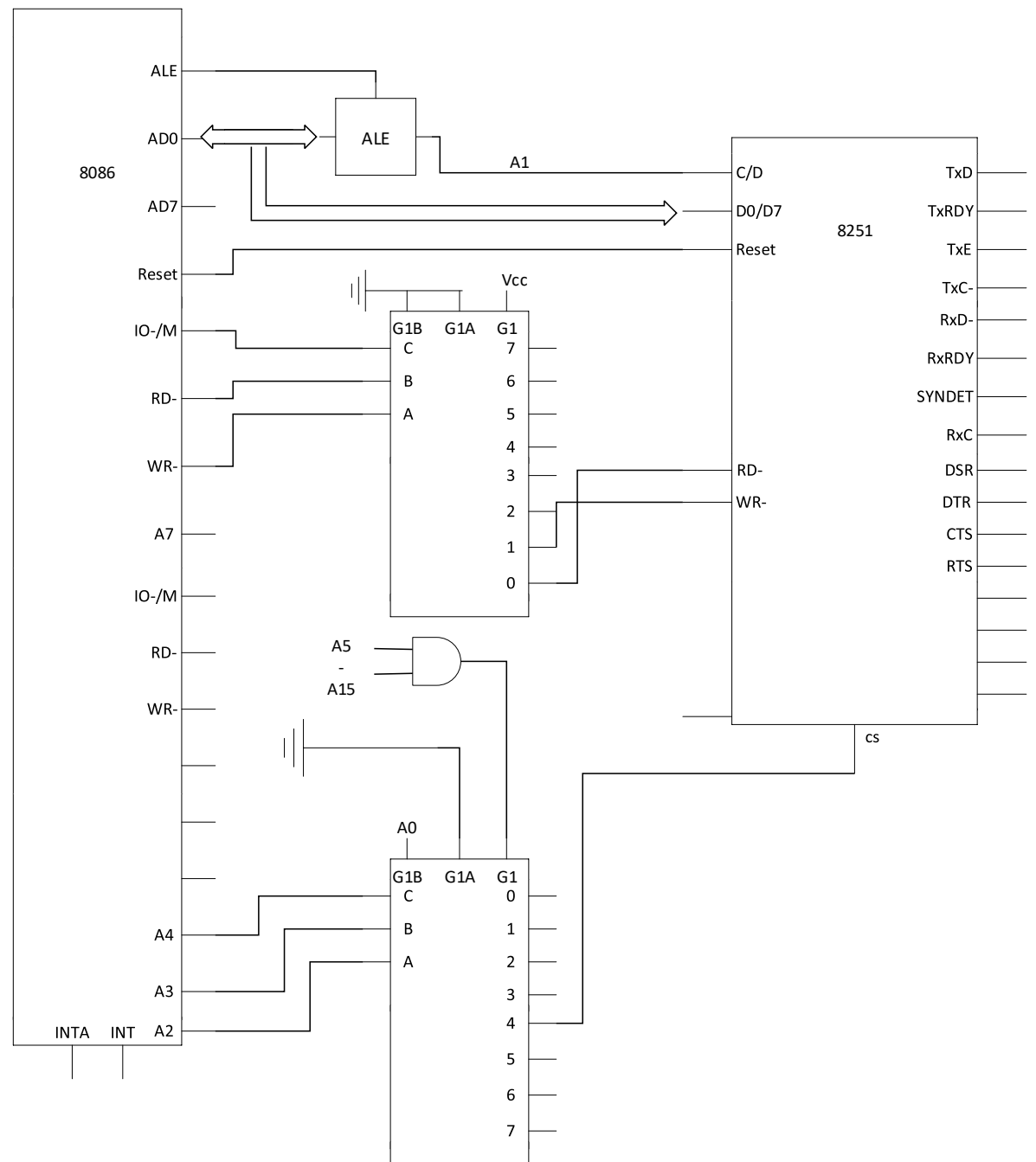


Zadatak 5

- Za mikroprocesor iAPX8086 projektovati sistem koji obavlja prenos uz zamenu podataka. Prima se neprekidan niz podataka sa komponente 8251A i šalje nazad tako što se svaki primljeni podatak koji je jednak sa podatkom na lokaciji SEC zamenjuje sekvencom '0xFF 0xFF 0xFF' u odlaznom nizu.
- Ostali podaci se neizmenjeni šalju nazad. Prijem i slanje sekvenci obaviti koristeći ispitne petlje. Za privremeno čuvanje podataka koristiti niz veličine 128 na lokaciji BUFFER.
- U slučaju da u prijemnom baferu nema dovoljno mesta za prijem novih podataka ignorisati primljene podatke sve do oslobađanja mesta u baferu.
- Podaci se primaju i šalju kao **osmobitni** sa **parnim** bitom parnosti brzinom **64x** i slanjem **dva stop bita**.
- U slučaju pojave greške u prijemu obustaviti dalji prijem, isprazniti bafer slanjem do tada primljenih podataka i završiti program. Obezbediti prijem novih podataka kada komponenta nije spremna za slanje i obrnuto. Komponenta 8251A je na UI adresi 0xFFFF0.

Adrese

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	8251 (FFF0)
1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	8251 (FFF2)



Rešenje

```
name zad5
data segment
    broj1 db 0
    full db 0
    empty db 1
    buffer db 128 dup(?)
    sec db 10h
data ends
code segment
    assume cs:code,
ds:data
```

```
start:
    mov ax, data
    mov ds, ax
    ;8251 inic.
    cli
    mov al, 00h
    mov dx, 0FFF2h
    out dx, al
    out dx, al
    out dx, al
    mov al, 40h
    out dx, al
    mov al, 0FFh
    out dx, al
    mov al, 15h
    out dx, al
    sti

    xor si, si
    xor di, di
```

Mode word (8251)

- 0xFF

S2	S1	EP	PEN	L2	L1	B2	B1
Broj stop bitova: 01 – 1 stop bit 10 – 1.5 stop bit 11 – 2 stop bita		Bit parnosti: 00 – disable 01 – neparna 10 – disable 11- parna		Broj bitova po karakteru: 00-5b 01-6b 10 -7b 11-8b		Baud rate factor: 00-Sync 01-1x 10-16x 11-64x	
1	1	1	1	1	1	1	1

Command word (8251)

- 0x15

EH	IR	RTS	ER	SBRK	RXE	DTR	TXEN
0:Normal 1:Hunt mode	0:Normal 1:Internal reset	0:DTR->1 1: DTR->0	0:Normal 1: Reset error flag	0:Normalan režim 1:Slanje karaktera prekida	Omogućiti prijemnik	0:DTR->1 1: DTR->0	Omogućiti predajnik
0	0	0	1	0	1	0	1

Petlja

petlja:

```
    mov dx, 0FFF2h
    in al, dx
    test al, 38h
    jnz greska
    test al, 02h
    jz slanje
    call primi
```

slanje:

```
    test al, 01h
    jz petlja
    call posalji
    jmp petlja
```

greska:

```
    call flush
    mov ax, 4c02h
    int 21h
```

- Status 8251

DSR	SYNDET BRKDET	FE	OE	PE	TxE	RxRDY	TxRDY
Data set ready	Sync detect/break error	Framing error	Overrun error	Parity error	Predajnik prazan	Prijemnik spreman	Predajnik spreman

Prijem podataka

- 7fh
 - 0111 1111 - po modulu dužine bafera
 - Kada postane 1000 0000, obriše se i krene od 0

```
primi proc near
    pushf
    push ax
    push dx
    mov dx, 0FFF0h
    in al, dx
    cmp al, sec
    je specijalan
    cmp brojel, 128
    je kraj
    mov buffer[di], al
    inc di
    and di, 7fh
    inc brojel
    jmp kraj
specijalan:
    cmp brojel, 125
    ja kraj
    mov buffer[di], 0FFh
    inc di
    and di, 7fh
    mov buffer[di], 0FFh
    inc di
    and di, 7fh
    mov buffer[di], 0FFh
    inc di
    and di, 7fh
kraj:
    pop dx
    pop ax
    popf
    ret
primi endp
```


Slanje podataka

```
posalji proc near
    pushf
    push ax
    push dx
    mov dx, 0FFF0h
    cmp brojel, 0
    je kraj
    mov al, buffer[si]
    out dx, al
    inc si
    and si, 7fh
    dec brojel
kraj:
    pop dx
    pop ax
    popf
posalji endp
```

Flush

- Poziva se kada dođe do greške u prijemu podataka

```
flush proc near
    pushf
    push ax
    push dx
    push cx
    xor cx, cx
    mov cl, brojel
petlja2:
cekaj:
    mov dx, 0FFF2h
    in al, dx
    test al, 01h
    jz cekaj
    mov dx, 0FFF0h
    mov al, buffer[si]
    inc si
    and si, 7fh
    loop petlja2
    pop cx
    pop dx
    pop ax
    popf
flush endp
code ends
end start
```