ŠTEVILSKI SESTAVI

1. DESETIŠKI

a) cifre: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
b) osnova: 10
c) primer:

$$234 = 2 \cdot 10^2 + 3 \cdot 10^1 + 4 \cdot 10^0$$

2. DVOJIŠKI (binarni)

a) cifre: 0, 1
b) osnova: 2
c) primer:

$$1011 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 8 + 2 + 1 = 11_{(10)}$$

3. ŠESTNAJSTIŠKI (heksadecimalni)

a) cifre: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
b) osnova: 16
c) primer:

$$2AC = 2 \cdot 16^2 + 10 \cdot 16^1 + 12 \cdot 16^0 = 2 \cdot 256 + 10 \cdot 16 + 12 \cdot 1 = 684$$

4. PRETVARJANJE ŠTEVIL

– *desetiško ⇒ dvojiško*

$13 \div 2 = 6 + 1$
$6 \div 2 = 3 + 0$
$3 \div 2 = 1 + 1$
$1 \div 2 = 0 + 1$

$13_{(10)} = 1101_{(2)}$

dvojiško ⇒ desetiško

$$1010 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 8 + 2 = 10_{(10)}$$

ali krajše

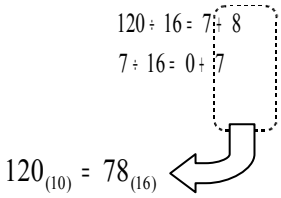
z utežmi:

$$\begin{array}{cccc} \ldots & 2^3 & 2^2 & 2^1 & 2^0 \\ \ldots & 8 & 4 & 2 & 1 \end{array}$$

primer

$$13 = 8 + 4 + 1 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 1101_{(2)}$$

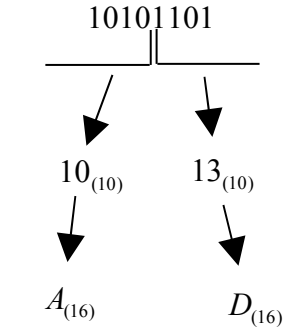
– *desetiško ⇒ šestnajstiško*



šestnajstiško ⇒ desetiško

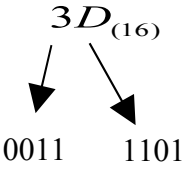
$$78_{(16)} = 7 \cdot 16^1 + 8 \cdot 16^0 = 7 \cdot 16 + 8 = 120_{(10)}$$

– *dvojiško ⇒ šestnajstiško*



$10101101_{(2)} = AD_{(16)}$

šestnajstiško ⇒ dvojiško



$3D_{(16)} = 00111101_{(2)} = 111101_{(2)}$

5. TIPI PODATKOV

bit = 1 cifra
byte (bajt) = 8 bitov
word = 16 bitov = 2 byta

6. ŠTEVILA S PREDZNAKOM

Najvišji bit v bytu nosi informacijo o predznaku (0 = + ; 1 = –)

- eniški komplement
- dvojiški komplement
- primeri pretvarjanja med pozitivnimi in negativnimi števili

PREKLOPNA ALGEBRA
(Boolova algebra)

1. *Načini opisovanja logičnih funkcij:*
   - pravilnostna tabela
   - funkcijska enačba (Boolova enačba)
   - stikalni načrt
   - funkcijski načrt (s simboli)
   - Veitchev diagram (KV diagram)

2. *Poskus z lučjo*

   vklop in izklop stikala → luč se prižge ali ugasne

   stikalo = x stanje luči = f(x)
   1 = vklop (gori)
   0 = izklop (ne gori)

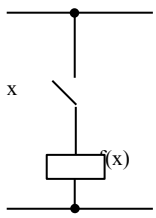
   Opišemo dogajanje:
   - pravilnostna tabela

x	f(x)
0	0
1	1

   - funkcijska enačba

   f(x) = x

   - stikalni načrt



   - funkcijski načrt (s simboli)

   - Veitchev diagram (KV diagram)

   število polj diagrama = 2^n
   vsaka spremenljivka pokrije polovico diagrama
   vpišemo v polja log. 1, kjer je spremenljivka (ali kombinacija spremenljivk) aktivna (postavi vrednost funkcije na log. 1)

x	1
$\overline{x}$	

3. *Povezava dveh stikal*

a) vzporedna vezava

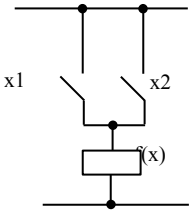
- pravilnostna tabela

x1	x2	f(x)
0	0	0
0	1	1
1	0	1
1	1	1

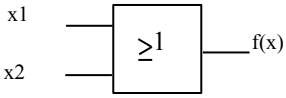
- funkcijska enačba

$$f(x) = x1+x2$$

- stikalni načrt



- funkcijski načrt (s simboli)



- Veitchev diagram (KV diagram)

	x1	$\bar{x1}$
x2	1	1
$\bar{x2}$	1	

b) zaporedna vezava

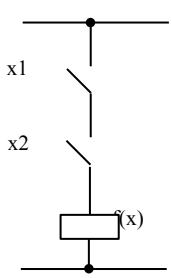
- pravilnostna tabela

x1	x2	f(x)
0	0	0
0	1	0
1	0	0
1	1	1

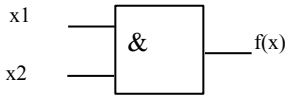
- funkcijska enačba

$$f(x) = x1 \bullet x2$$

– stikalni načrt



– funkcijski načrt (s simboli)



– Veitchev diagram (KV diagram)

	x1	$\bar{x}1$
x2	1	
$\bar{x}2$		

4. *Osnovne operacije v preklopni algebri*

– ALI operacija

x1	x2	f(x)
0	0	0
0	1	1
1	0	1
1	1	1

– IN operacija

x1	x2	f(x)
0	0	0
0	1	0
1	0	0
1	1	1

– negacija

x	f(x)
0	1
1	0

TEOREMI PREKLOPNE ALGEBRE
(pravila)

$0 + 0 = 0$
$1 + 0 = 1$
$1 + 1 = 1$
$0 \cdot 0 = 0$
$0 \cdot 1 = 0$
$1 \cdot 1 = 1$

$x \cdot 0 = 0$
$x \cdot 1 = x$
$x + 0 = x$
$x + 1 = 1$
$\overline{\overline{x}} = x$

$x \cdot x = x$
$x \cdot \overline{x} = 0$
$x + x = x$
$x + \overline{x} = 1$

komutativnost

$x1 \cdot x2 = x2 \cdot x1$
$x1 + x2 = x2 + x1$

asociativnost

$x1 \cdot x2 \cdot x3 = (x1 \cdot x2) \cdot x3 = x1 \cdot (x2 \cdot x3)$
$x1 + x2 + x3 = (x1 + x2) + x3 = x1 + (x2 + x3)$

distributivnost

$x1 \cdot (x2 + x3) = x1 \cdot x2 + x1 \cdot x3)$
$x1 + (x2 \cdot x3) = (x1 + x2) \cdot (x1 + x3)$

De Morganova teorema

$\overline{x1 \cdot x2} = \overline{x1} + \overline{x2}$
$\overline{x1 + x2} = \overline{x1} \cdot \overline{x2}$

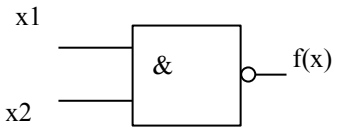
} velja enako tudi za več spremenljivk hkrati

OSNOVNE LOGIČNE FUNKCIJE DVEH SPREMENLJIVK

Poleg že obdelanih osnovnih log. funkcij si poglejmo še naslednje:

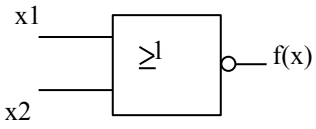
- *NEIN* (NAND) Schefferjeva f.

$$f = \overline{x1 \cdot x2} = x1 \mid x2$$



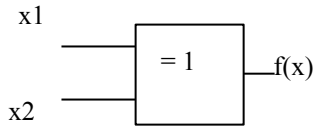
- *NEALI* (NOR) Pierceova f.

$$f = \overline{x1 + x2} = x1 \downarrow x2$$



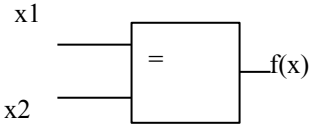
- *antivalenca - ekskluzivni ALI* (XOR)

$$f = x1 \oplus x2$$



- *ekvivalenca* (XNOR)

$$f = \overline{x1 \oplus x2}$$



POENOSTAVLJANJE LOG. FUNKCIJ

1. *Poenostavljanje funkcijskih enačb*

   Uporabljamo teoreme in lastnosti preklopne algebre.

   Primeri:

   – **2 spremenljivki:**

x1	x2	f(x)
0	0	1
0	1	1
1	0	0
1	1	0

$$f(x) = m_0 + m_1 = \bar{x}_1 \cdot \bar{x}_2 + \bar{x}_1 \cdot x_2 = \bar{x}_1 \cdot (\bar{x}_2 + x_2) = \bar{x}_1 \cdot 1 = \bar{x}_1$$

x1	x2	f(x)
0	0	1
0	1	1
1	0	1
1	1	0

$$f(x) = m_0 + m_1 + m_2 = \bar{x}_1 \cdot \bar{x}_2 + \bar{x}_1 \cdot x_2 + x_1 \cdot \bar{x}_2 = \bar{x}_1 \cdot \bar{x}_2 + \bar{x}_1 \cdot x_2 + \bar{x}_1 \cdot \bar{x}_2 + x_1 \cdot \bar{x}_2 =$$
$$= \bar{x}_1 \cdot (\bar{x}_2 + x_2) + \bar{x}_2 \cdot (\bar{x}_1 + x_1) = \bar{x}_1 + \bar{x}_2$$

   – **3 spremenljivke:**

x1	x2	x3	f(x)
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

$$f(x) = m_0 + m_2 + m_4 + m_6 = \bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 + \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 + x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 + x_1 \cdot x_2 \cdot \bar{x}_3 =$$
$$= \bar{x}_2 \cdot \bar{x}_3 \cdot (\bar{x}_1 + x_1) + x_2 \cdot \bar{x}_3 \cdot (\bar{x}_1 + x_1) = \bar{x}_2 \cdot \bar{x}_3 + x_2 \cdot \bar{x}_3 = \bar{x}_3 \cdot (\bar{x}_2 + x_2) = \bar{x}_3$$

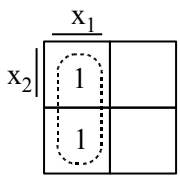
x1	x2	x3	f(x)
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

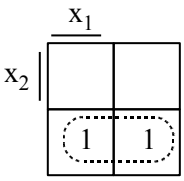
$$f(x) = m_0 + m_1 + m_3 = \bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 + \bar{x}_1 \cdot \bar{x}_2 \cdot x_3 + \bar{x}_1 \cdot x_2 \cdot x_3 = \bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 + \bar{x}_1 \cdot \bar{x}_2 \cdot x_3 +$$
$$+ \bar{x}_1 \cdot \bar{x}_2 \cdot x_3 + \bar{x}_1 \cdot x_2 \cdot x_3 = \bar{x}_1 \cdot \bar{x}_2 \cdot (\bar{x}_3 + x_3) + \bar{x}_1 \cdot x_3 \cdot (\bar{x}_2 + x_2) = \bar{x}_1 \cdot \bar{x}_2 + \bar{x}_1 \cdot x_3 = \bar{x}_1 \cdot (\bar{x}_2 + \cdot x_3)$$

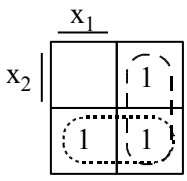
2. *Veitchev diagram*

Primeri:

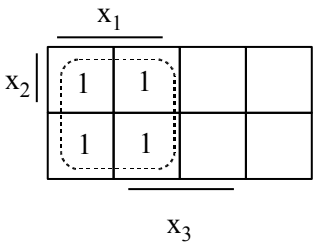
- 2 spremenljivki

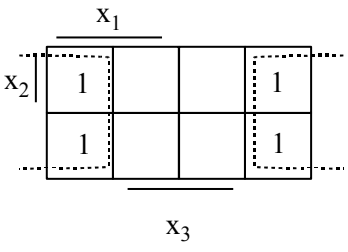


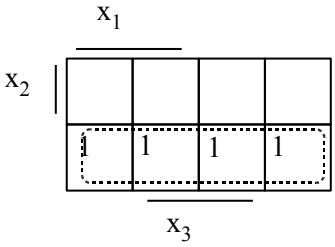


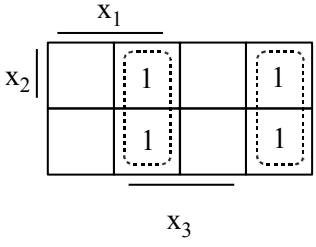


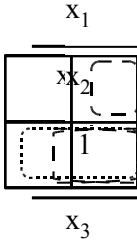
- 3 spremenljivke









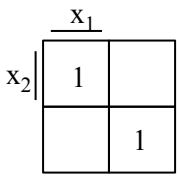


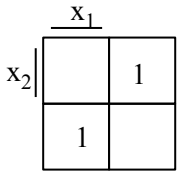
3. *Upoštevanje redundantnih kombinacij*

x1	x2	f(x)
0	0	x
0	1	1
1	0	0
1	1	0

x1	x2	x3	f(x)
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	x
1	0	0	0
1	0	1	x
1	1	0	x
1	1	1	1

4. *Poenostavljanje z uporabo XOR in XNOR funkcij*





$f = x_1 \cdot x_2 + \bar{x}_1 \cdot \bar{x}_2 = \overline{x_1 \oplus x_2}$

$f = x_1 \cdot \bar{x}_2 + \bar{x}_1 \cdot x_2 = x_1 \oplus x_2$

LOGIČNA VEZJA

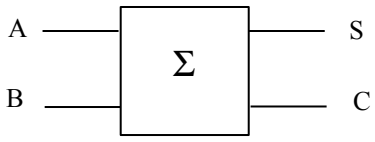
SEŠTEVALNIK

1. *Polovični seštevalnik*

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$S = \overline{A} \cdot B + A \cdot \overline{B}$
$C = A \cdot B$

$S = A \oplus B$
$C = A \cdot B$

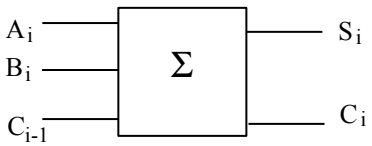


2. *Popolni seštevalnik*

A_i	B_i	C_{i-1}	C_i	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$S_i = A_i \cdot \overline{B}_i \cdot \overline{C}_{i-1} + \overline{A}_i \cdot B_i \cdot \overline{C}_{i-1} + \overline{A}_i \cdot \overline{B}_i \cdot C_{i-1} + A_i \cdot B_i \cdot C_{i-1}$
$S_i = \overline{C}_{i-1} \cdot (A_i \cdot \overline{B}_i + \overline{A}_i \cdot B_i) + C_{i-1} \cdot (\overline{A}_i \cdot \overline{B}_i + A_i \cdot B_i)$
$S_i = \overline{C}_{i-1} \cdot (A_i \oplus B_i) + C_{i-1} \cdot (\overline{A_i \oplus B_i})$
$S_i = C_{i-1} \oplus (A_i \oplus B_i)$

$C_i = A_i \cdot B_i \cdot \overline{C}_{i-1} + A_i \cdot \overline{B}_i \cdot C_{i-1} + \overline{A}_i \cdot B_i \cdot C_{i-1} + A_i \cdot B_i \cdot C_{i-1}$
$C_i = C_{i-1} \cdot (A_i \cdot \overline{B}_i + \overline{A}_i \cdot B_i) + A_i \cdot B_i \cdot (\overline{C}_{i-1} + C_{i-1})$
$C_i = C_{i-1} \cdot (A_i \oplus B_i) + A_i \cdot B_i$

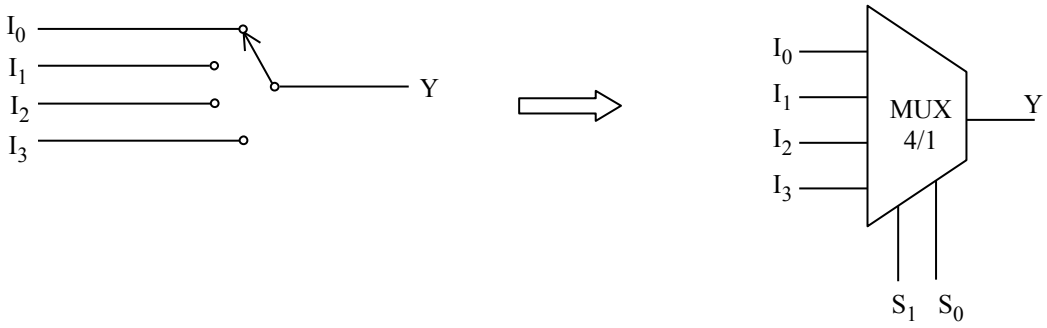


3. *Večbitni seštevalnik*

- polovični
- popolni

MULTIPLEKSER

Delovanje multiplekserja:

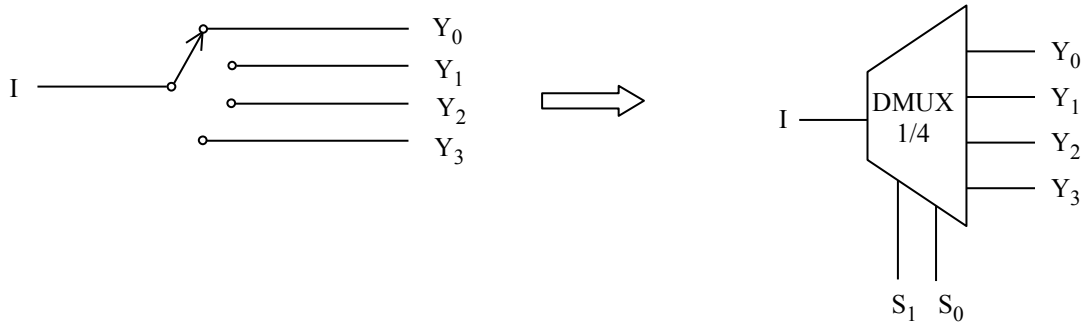


Pravilnostna tabela:

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

DEMULTIPLEKSER

Delovanje demultiplekserja:



Pravilnostna tabela:

S_1	S_0	I	Y_0	Y_1	Y_2	Y_3
0	0	1	0	1	1	1
0	1	1	1	0	1	1
1	0	1	1	1	0	1
1	1	1	1	1	1	0

X	X	0	1	1	1	1

PRIMERJALNIK (KOMPARATOR)

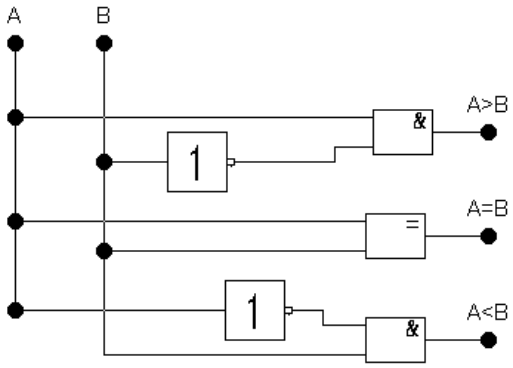
- *primerjava dveh binarnih števil A in B*
  Rezultat primerjave je lahko:

  – A>B
  – A=B
  – A<B

  } TRIJE IZHODI!

Enobitni komparator:

A	B	A>B	A=B	A<B
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0



Večbitni so v integrirani obliki (IC).

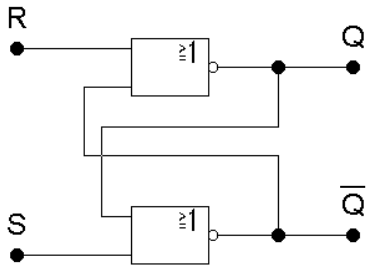
SEKVENČNA VEZJA

Spominski elementi:
- asinhroni (časovno neusklajeno delovanje) – zaporedna povezava elementov,
- sinhroni (časovno usklajeno delovanje) – poljubna povezava elementov.

POMNILNE CELICE

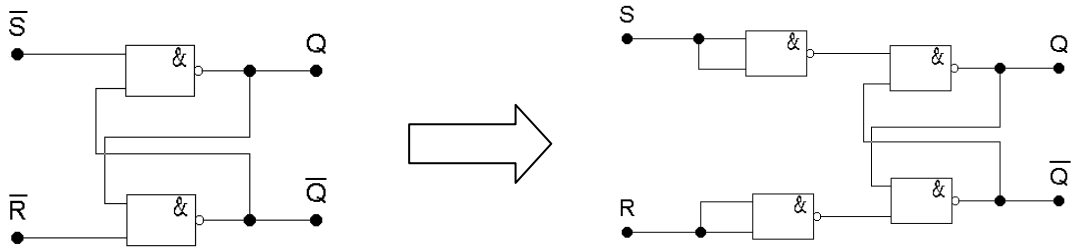
Osnovna pomnilna celica je *flip-flop* ali *bistabil*. Shrani lahko en bit informacije.

1. *RS flip-flop*
   a) asinhroni
      - izvedba z NOR vrati



S	R	Q	$\overline{Q}$	
0	0	X	X	ohrani prejšnje stanje
0	1	0	1	
1	0	1	0	
1	1	0	0	prepovedano stanje

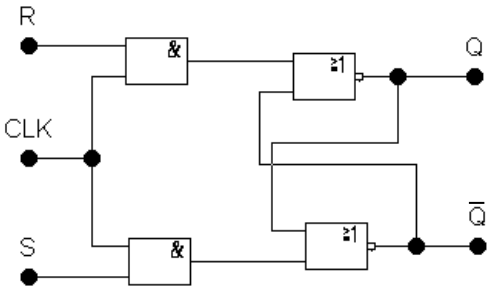
      - izvedba z NAND vrati



S	R	$\overline{Q}$	Q	
0	0	1	1	prepovedano stanje
0	1	1	0	
1	0	0	1	
1	1	X	X	ohrani prejšnje stanje

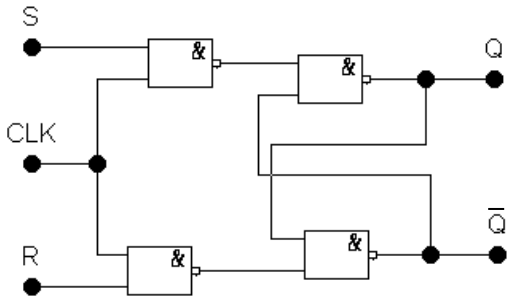
b) sinhroni

– izvedba z NOR vrati



S	R	CLK	Q	$\overline{Q}$	–
0	0	0	X	X	ohrani prejšnje stanje
0	0	1	X	X	ohrani prejšnje stanje
0	1	0	X	X	ohrani prejšnje stanje
0	1	1	0	1	
1	0	0	X	X	ohrani prejšnje stanje
1	0	1	1	0	
1	1	0	X	X	ohrani prejšnje stanje
1	1	1	0	0	prepovedano stanje

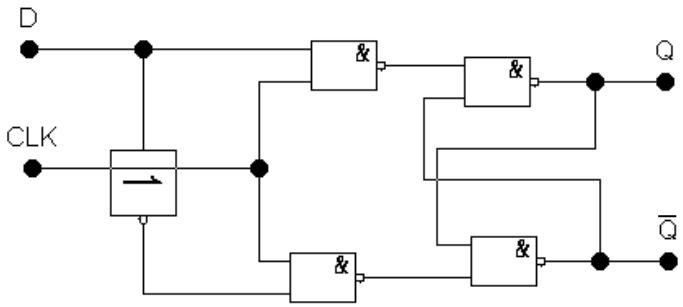
– izvedba z NAND vrati



S	R	CLK	Q	$\overline{Q}$	–
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

2. *D flip-flop*

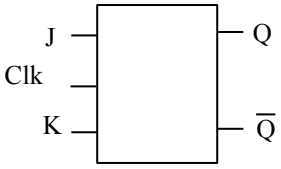
Slabost RS flip-flopa je prepovedano stanje – odpravimo ga z negatorjem na R vhod. Dobimo D flip-flop, ki nima prepovedanega stanja!



D	CLK	Q	$\overline{Q}$	
0	0	X	X	ohrani prejšnje stanje
0	1	0	1	
1	0	X	X	ohrani prejšnje stanje
1	1	1	0	

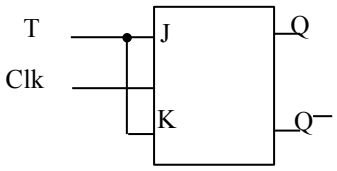
3. *JK flip-flop*

Je sinhrona celica, ki ima dva vhoda in nima slabosti RS flip-flopa (prepovedanega stanja).



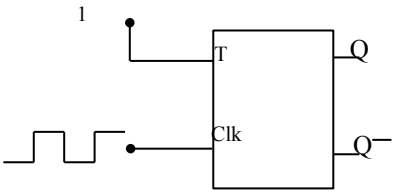
J	K	Q_n	Q_{n+1}	
0	0	0	0	ohrani prejšnje stanje
0	0	1	1	ohrani prejšnje stanje
0	1	0	0	briše
0	1	1	0	briše
1	0	0	1	postavi
1	0	1	1	postavi
1	1	0	1	spremeni prejšnje stanje
1	1	1	0	spremeni prejšnje stanje

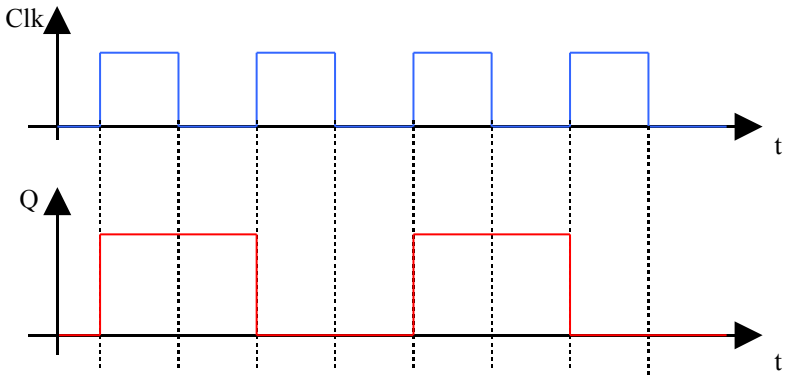
4. *T flip-flop*



T	Q_n	Q_{n+1}	
0	0	0	ohrani prejšnje stanje
0	1	1	ohrani prejšnje stanje
1	0	1	spremeni prejšnje stanje
1	1	0	spremeni prejšnje stanje

Delovanje na primeru (delilnik frekvence)!





5. *Krmiljenje pomnilnih celic*

Ločimo krmiljenje celic z:
- logičnimi vrednostmi,
- bokom impulza (pozitivni, negativni bok) – npr. CLK vhod.

Pomnilne celice imajo lahko še naslednje vhode:
- vhod, ki omogoča ali onemogoča vpis vrednosti (ENABLE, LATCH ENABLE) – D celica,
- vhode za postavitev ali brisanje izhoda (SET, RESET) – JK, T celica.

Ti vhodi so lahko aktivni na log. 0 ali 1 (oznaka)!

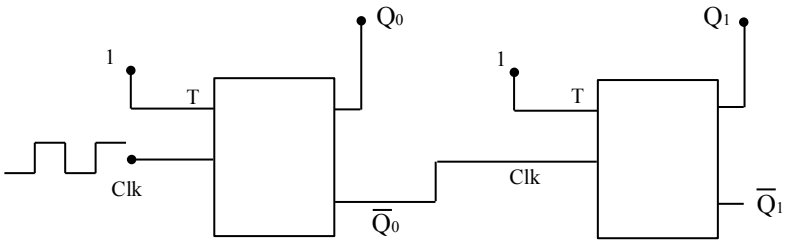
ŠTEVNIKI IN BINARNI DELILNIKI

So sekvenčna vezja, sestavljena iz osnovnih pomnilnih celic. Modul štetja števnika nam pove, koliko različnih stanj lahko zavzame (npr. m=10 pomeni stanja od 0 do 9). Števnike delimo na asinhrone in sinhrone. Lahko imajo še naslednje možnosti:

- vpis vrednosti (paralelni vpis),
- brisanje (resetiranje),
- različno smer štetja (UP/DOWN).

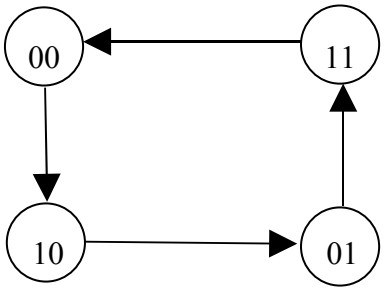
1. *Asinhroni števniki*

Osnovna celica je T flip-flop.



2. *Sinhroni števniki*

Uporabimo sinhroni RS ali JK flip-flop – večinoma JK! Delovanje lahko predstavimo tudi z diagramom stanj. Sestavljajo ga krogi ter usmerjene povezave. V krogih so izhodna stanja števnika, usmerjene povezave med krogi pa kažejo delovanje vezja (prehode stanj števnika). Ob povezavah so lahko zapisane tudi vhodne vrednosti (dodatni vhod-i), ki povzročajo posamezne prehode stanj, ter izhodna stanja posebnega izhoda – če le-ta obstaja.



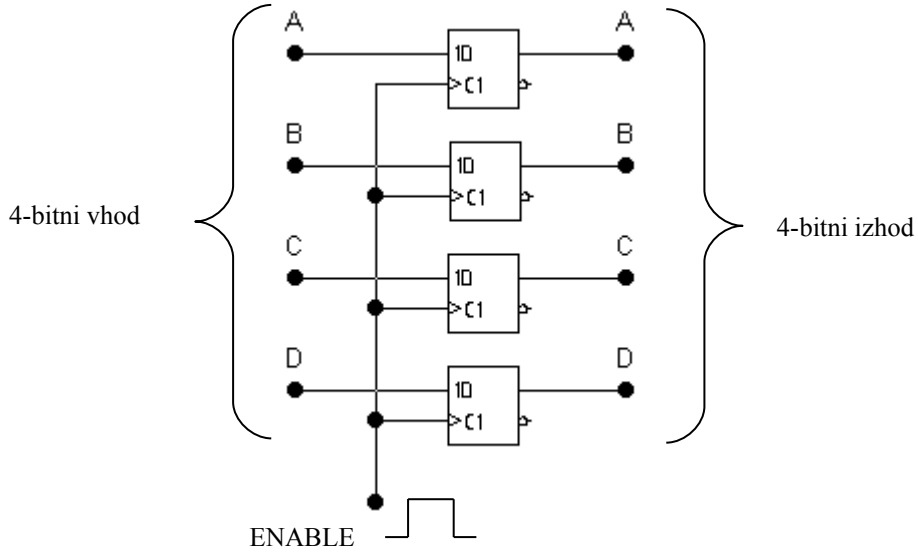
REGISTRI

So sekvenčna vezja, sestavljena iz RS, D ali JK flip-flopov. Število uporabljenih celic določa dolžino registra (4, 8, 16, 32 bitni). Namenjeni so shranjevanju podatkov (pomnilniški blok) ali pomikanju vsebine registra (pomikalni registri).

1. *Pomnilniški blok*

So osnovne enote nekaterih pomnilnikov ter drugih vezij (števnikov, dekoderjev, prekodirnikov in podobno).

Primer uporabe – latch ali zatič:



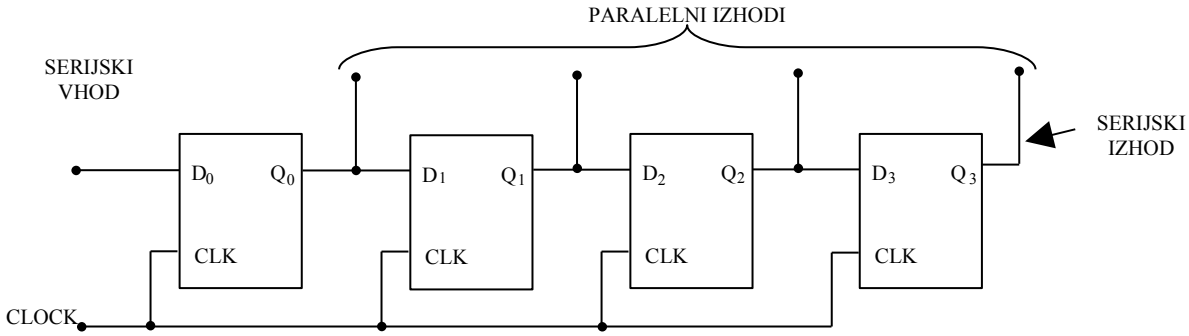
2. *Pomikalni registri*

Omogočajo pomik vsebine registra v eno ali drugo stran (levo ali desno). Pomik se izvrši ob aktivnem CLK impulzu. Ločimo naslednje izvedbe:

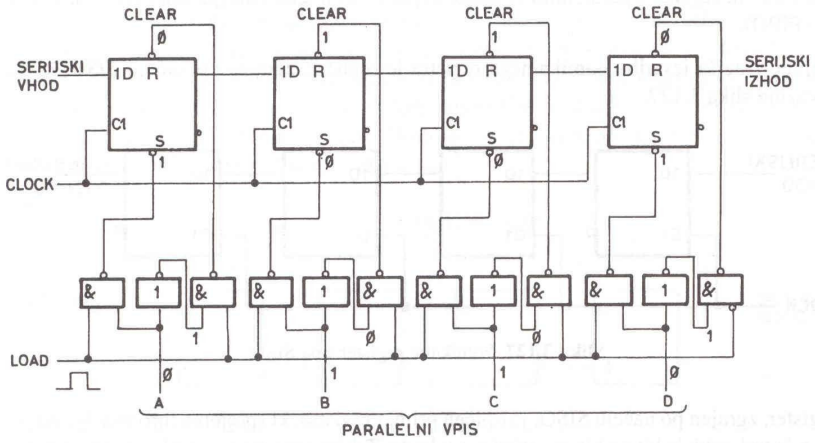
- pomikalni register s serijskim vhodom in serijskim izhodom (SISO)
- pomikalni register s serijskim vhodom in paralelnim izhodom (SIPO)
- pomikalni register s paralelnim vhodom in serijskim izhodom (PISO)
- pomikalni register s paralelnim vhodom in paralelnim izhodom (PIPO)

Paralelni vpis podatkov se izvrši asinhrono s pomočjo dodatnega vhoda (LOAD). Tak register ima tudi serijski vhod. Če ima pomikalni register paralelni izhod, je zadnji izhod hkrati tudi serijski izhod. Nekateri pomikalni registri imajo možnost izbire pomikanja v levo ali desno.

a) Realizacija z D celicami



b) Pomikalni registri s paralelnim vpisom



Pomikalni register tipa PISO

POMNILNIKI

1. Delitev pomnilnikov:

a) *po fizikalnem principu zapisa informacije*
- magnetni zapis (trdi, mehki diski)
- svetlobni zapis (CD, DVD)
- stanje celice (SRAM, EPROM, ...)
- naboj kondenzatorja (DRAM)

b) po načinu dostopa do pomnilniških lokacij
- naključni (RAM, ROM, PROM, ...)
- zaporedni
- krožni
- direktni dostop

2. Karakteristični parametri pomnilnikov
- kapaciteta ali obseg (št. lokacij)
- dostopni čas
- možnost spremembe vpisane informacije
- obstojnost vpisane informacije
- cena na bit

3. Pomnilniki z naključnim dostopom

Vsaka celica ima svoj naslov, dostopni čas je enak do vsake celice. Priključki:
- naslovni (adresni) A0, A1, ... – število odvisno od kapacitete
- podatkovni D0 – D7 (O0 – O7) za 8-bitne pomnilnike
- kontrolni (napajanje, **C**hip **S**elect, **O**utput **E**nable, **R**ead/**W**rite, **PGM** – programiranje)

a) bralni pomnilniki

ROM (Read Only Memory)

Je pomnilnik, kateremu je že ob izdelavi vnesena vsebina, ki se pozneje na da več spremeniti. Enobitna informacija je v celici (dioda, tranzistor) s programirljivo povezavo. Obstoječa povezava pomeni eno log. stanje (0), neobstoječa (prekinjena) pa drugo log. stanje (1). Informacijo zadrži tudi po odklopu napajanja.

PROM (Programmable Read Only Memory)

Matrika spominskih celic je sestavljena iz diod. Vsaka dioda pa ima oslabljeno mesto na vodniku (varovalka). Programiranje se opravi tako, da se prežgejo s tokovnim sunkom oslabljena mesta – varovalke. Ko je pomnilnik programiran, ga ne moremo več reprogramirati (spremeniti vsebine), zato ga uporabljamo kot ROM. Oslabljeno mesto se prekine z napetostjo nad 20V. Informacijo zadrži tudi po odklopu napajanja.

EPROM (Erasable-and-Programmable ROM)

Omogočajo večkratno vpisovanje informacij (programiranje). Spominske celice sestavljajo MOSFET tranzistorji s plavajočimi vrati (FAMOS). Programiramo jih s pomočjo napetosti, višje od napajalne (ponavadi s programatorji). Brišemo jih z UV svetlobo (okence). Informacijo zadrži tudi po odklopu napajanja.

Označevanje: 27xxx xxx=št. kbitov (2764 = 64kbitni = 65536bitni/8 =8192bytni – 8kbytov)

http://xtronics.com/memory/how_EPROM-works.htm

http://www.howstuffworks.com/rom4.htm

EEPROM (Electrically-Erasable-and-Programmable ROM)

Po zgradbi je zelo podoben EPROM-u, le da omogoča brisanje informacij s pomočjo električne napetosti.

Flash EEPROM omogoča brisanje po sektorjih in ne samo celotnega pomnilnika.

http://en.wikipedia.org/wiki/USB_flash_drive#Technology
http://en.wikipedia.org/wiki/Flash_memory#NAND_memories

b) bralno-pisalni pomnilniki

SRAM

Pomnilna celica je zatič (latch) – kompleksnejši ⇒ manjša kapaciteta na isti površini in višja cena/bit.

Informacijo ohranja samo, ko je priključen na napajanje. Ko je informacija vpisana, je ni potrebno obnavljati. Ima kratek dostopni čas.

Označevanje: 62xxx xxx=št. kbitov (6264 = 64kbitni = 65536bitni/8 =8192bytni – 8kbytov)

DRAM

Pomnilna celica je tranzistor s kondenzatorjem – manj kompleksnejši od SRAM-a ⇒ večja kapaciteta na isti površini in nižja cena/bit.

Informacijo ohranja kratek čas in samo, ko je priključen na napajanje. Zato jo je potrebno obnavljati (refresh). Ima daljši dostopni čas ter manjšo porabo kot SRAM.

http://computer.howstuffworks.com/framed.htm?parent=flash-memory.htm&url=http://www.kingston.com/tools/umg/default.asp
(vrste spominskih modulov)

D/A PRETVORNIKI

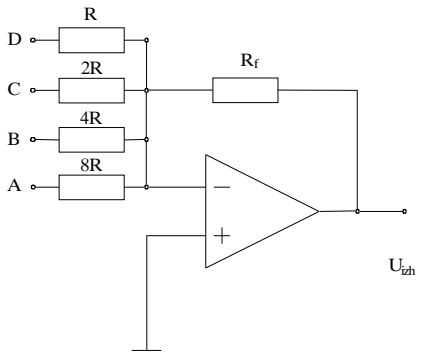
Podatki D/A pretvornikov:

a) Ločljivost
   - št. vhodov (bitov)
   - najmanjša sprememba izhodne napetosti ali toka

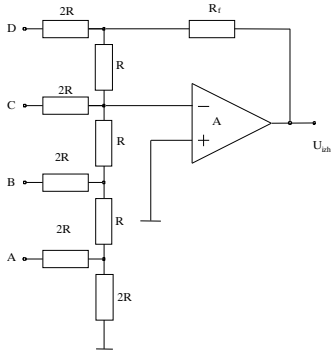
b) Čas pretvorbe

c) Napake pretvornika

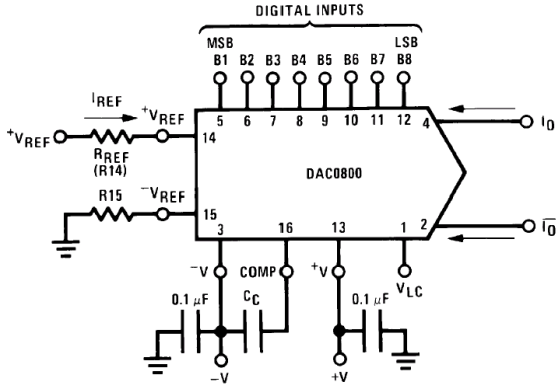
1. Utežnostni D/A pretvornik



2. Lestvičasti D/A pretvornik

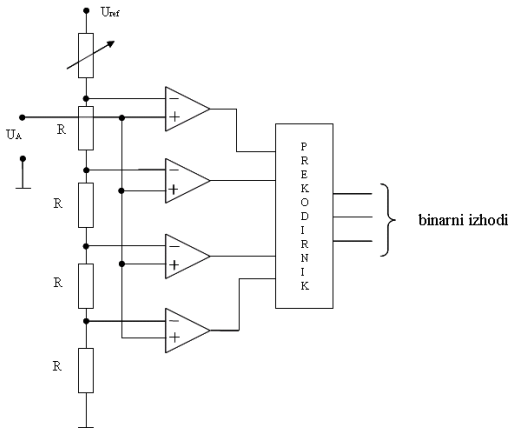


3. Integrirani D/A pretvornik DAC 0800

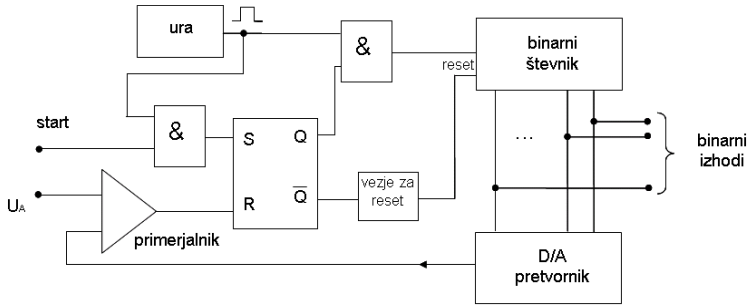


A/D PRETVORNIKI

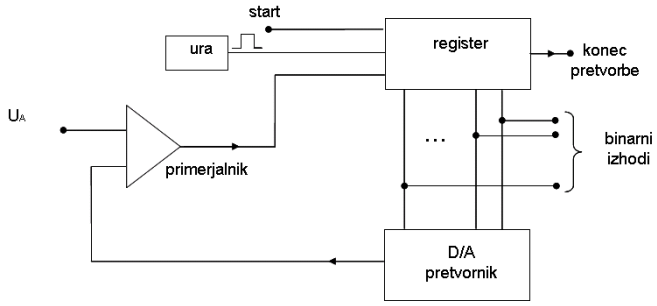
1. A/D pretvornik s paralelnimi komparatorji (Flash converter)



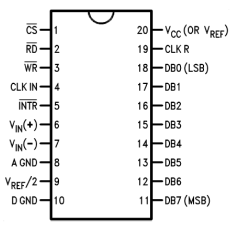
2. Startno-števni pretvornik



3. A/D pretvornik s postopnim približevanjem (s sukcesivno aproksimacijo)



4. Integrirani A/D pretvornik ADC0801



KRMILJA

Vrste krmilij:

- trajno ožičena:
  - z elektromehanskimi elementi
  - z moduli elektronskih elementov
- programirljiva
  - mikroprocesorski sistemi
  - krmilniki (PLC)

Prednosti programirljivih krmilij:

- prilagodljivost uporabniku (več možnosti prilagajanja željam naročnika)
- preprostejša nadgradnja in sprememba krmilja
- povezljivost posameznih delov krmilja v celoto
- varčevanje z energijo
- lažji nadzor in upravljanje objekta
- ...

Glavne komponente programirljivih krmilij:

- programirljiva krmilna naprava (mikrokontroler, PLK)
- dajalniki stanj (tipke, stikala, senzorji ipd)
- izvršilne naprave oz. aktuatorji (luči, motorji, grelci in izmenjevalci toplote, ventili itd)

Dajalniki stanj:

- tipke in stikala

- senzorji temperature
  - NTK
  - PTK
  - termočlen

- svetlobni senzorji
  - fotocelice
  - svetlobne zavese
  - IR senzorji gibanja

- senzorji sile in tlaka
  - senzorji sile
  - senzorji tlaka tekočin, plinov

- pozicijska, varnostna in končna stikala (kontaktna in brezkontaktna)
  - kontaktna stikala
  - induktivna in kapacitivna stikala
  - magnetna stikala
  - ultrazvočni senzorji

- dajalniki položaja
  - inkrementalni dajalniki
  - absolutni dajalniki

- senzorji nevarnih plinov in tekočin

Vrste dajalnikov stanj:

- pasivni (rabijo izvor napetosti oz. toka)
- aktivni (vsebujejo izvor napetosti oz. toka)
- "inteligentni" (krmilna naprava komunicira s takimi napravami)

Primer mehanskih stikal:
http://int.catalog.moeller.net/en/default.asp?Form=6&PRGR=100000524&Typ=3

Katalog Omron:
http://industrial.omron.eu/en/products/catalogue/default.html

Katalog Allen-Bradley:
http://www.ab.com/sensors/brozine/

Katalog Moeller:
http://int.catalog.moeller.net/en/default.asp?prgrlink=i00512&Form=3

Katalog Hengstler:
http://www.hengstler.com/en/products/shop.php

Katalog Panasonic (avtomatizacija):
http://www.panasonic-electric-works.it/pewit/it/html/66.php

Katalog PDB Turck:
http://pdb.turck.de/catalogue/catalogue.do;jsessionid=8ED8CACD930A03BE140C4F8A0F1426B3?act=showBookmark&favOid=00000001000172690006002 3&lang=en&catId=DE

Vklop močnostnih naprav:

- rele, kontaktor
- triak
- tranzistor
- optospojnik
- polprevodniški rele (Solid State Relay – SSR)

Povezave med komponentami programirljivih krmilij:

- žične (posebej izvedene povezave)
- brezžične
- po omrežni inštalaciji

Prenos signalov v krmilje:

- binarna stanja (0V ali napajalna napetost)
- napetostni nivoji (0V – 10V za krmilnike)
- tokovna zanka (0 (oz.4)mA – 20mA za krmilnike)
- serijski prenos podatkov (binarne večbitne vrednosti): RS-232, RS-485, I2C, SMBus, SPI, 1Wire, ...

MIKROPROCESORSKI SISTEMI

Mikroprocesor in mikrokrmilnik oz.mikrokontroler:

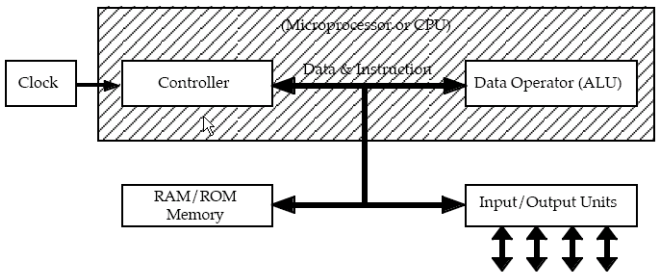
a) Mikroprocesor:
- ALE
- registri

b) Spominske enote:
- (E)EPROM (Flash EEPROM)
- RAM

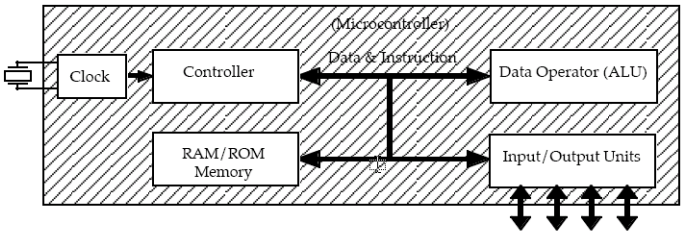
c) Vhodno-izhodne enote:
- paralelni prenos podatkov
- serijski prenos podatkov
- A/D pretvorniki (mikrokontrolerji)

d) Povezave med enotami sistema (vodila)

Osnovna zgradba:
- mikroprocesor



- mikrokontroler



Uporaba mikroprocesorskega sistema:

- krmiljenje strojev in naprav
- krmiljenja in regulacije v proizvodnih procesih
- krmiljenja in regulacije toplotnih in hladilnih sistemov
- alarmni sistemi
- avdio in video naprave
- bela tehnika
- telekomunikacije in drugo ...

KRMILNIKI

Zgradba in značilnosti

So naprave, ki nam omogočajo izvedbo prilagodljivih krmilnih in regulacijskih vezij. Upravlja jih mikrokontroler, ki izvaja realizirani program. Le-tega sestavimo na osebnem računalniku z ustreznim programskim orodjem, ki nam omogoča ta program tudi shraniti v krmilnikovem pomnilniku. Poznamo manjše izvedbe krmilnikov ter zmogljivejše večje (industrijske). Manjše izvedbe imajo določeno število vhodnih in izhodnih priključkov ter možnost komunikacije z osebnim računalnikom. Nekateri imajo še LCD prikazovalnik ter nekaj tipk za upravljanje. Večje (industrijske) sestavljamo v (skoraj) poljubne konfiguracije. Take krmilnike lahko med seboj povezujemo v mrežo, v kateri imamo lahko tudi enega ali več nadzornih računalnikov (SCADA).
Vhodi krmilnikov so lahko digitalni in analogni. Območje vhodne napetosti za analogne vhode je od 0V do 10V, tokovno območje pa je od 0 oz. 4mA do 20mA. To napetost ali tok A/D pretvornik pretvori v večbitno binarno vrednost. Izhodi so močnostni (rele, tranzistor ali triak) in omogočajo vklop bremen s porabo toka do 10A. Napajalne napetosti so:

- 12V/24V DC (nekateri tudi AC)
- 110V-240V AC

Programiranje krmilnikov

- direktno na krmilniku (pogoj je LCD zaslon in tipke – manj uporabno)
- z osebnim računalnikom (proizvajalčev programski paket ter povezava PC ⇔ PLC – RS 232, USB, Ethernet, ...)

Načini programiranja krmilnikov

- funkcijski blokovni diagram (FBD)
- lestvični diagram (LD)
- strukturiran tekst (ST)

Proizvajalci krmilnikov

MITSUBISHI
SIEMENS
KLOCKNER-MOELLER
OMRON
ALLEN BRADLEY
HITACHI
Smarteh
Robotina
.
.
.

Manjši modeli krmilnikov:

MITSUBISHI **Alpha**	www.the-new-alpha.com
SIEMENS **LOGO**	www.siemens.com/logo
KLOCKNER-MOELLER **Easy**	www.klocknermoeller.com
OMRON **ZEN**	www.omron.com
ALLEN BRADLEY **Pico**	www.ab.com/plclogic/pico

Primer krmilnika:

MITSUBISHI Alpha

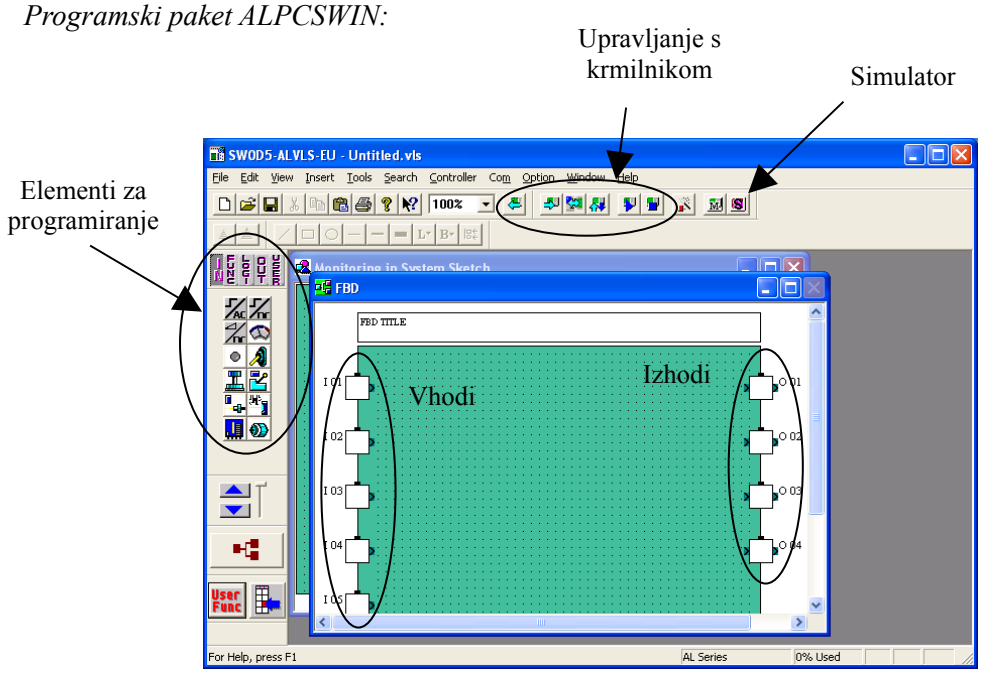
1. Značilnosti:

a) napajalna napetost (24V DC ali 100V – 240V AC)
b) vhodi (4 – 12):
   – digitalni
   – analogni (verzija z DC napajanjem) – 8 bitni A/D pretvornik
c) izhodi (2 – 8)
   – relejski (do 8A)
   – tranzistorski (samo nekatere verzije z DC napajanjem; do 1A)
d) dodatni vhodi (8) – tipke na krmilniku
e) LCD prikazovalnik

2. Programiranje krmilnika

– direktno (na krmilniku)
– s pomočjo računalnika (programski paket ALPCSWIN): omogoča simulacijo delovanja krmilnika ter komunikacijo z njim – lahko ga tudi upravljamo z računalnikom

Programski paket ALPCSWIN:



MIKROKONTROLER MC908GP32

Lastnosti

Osnovni podatki:
- 8-bitna ALE
- 512 B RAM-a
- 32 kB FLASH spomina (EEPROM)
- programiranje v vezju (In-circuit programming) – Monitor ROM
- do 33 I/O priključkov na paralelnih portih (port A – D) – obremenitve do 10mA
- 8 kanalni A/D pretvornik (8-bitni)
- serijska komunikacija
- večfunkcijski časovnik (timer)
- ...

Zgradba in priključki

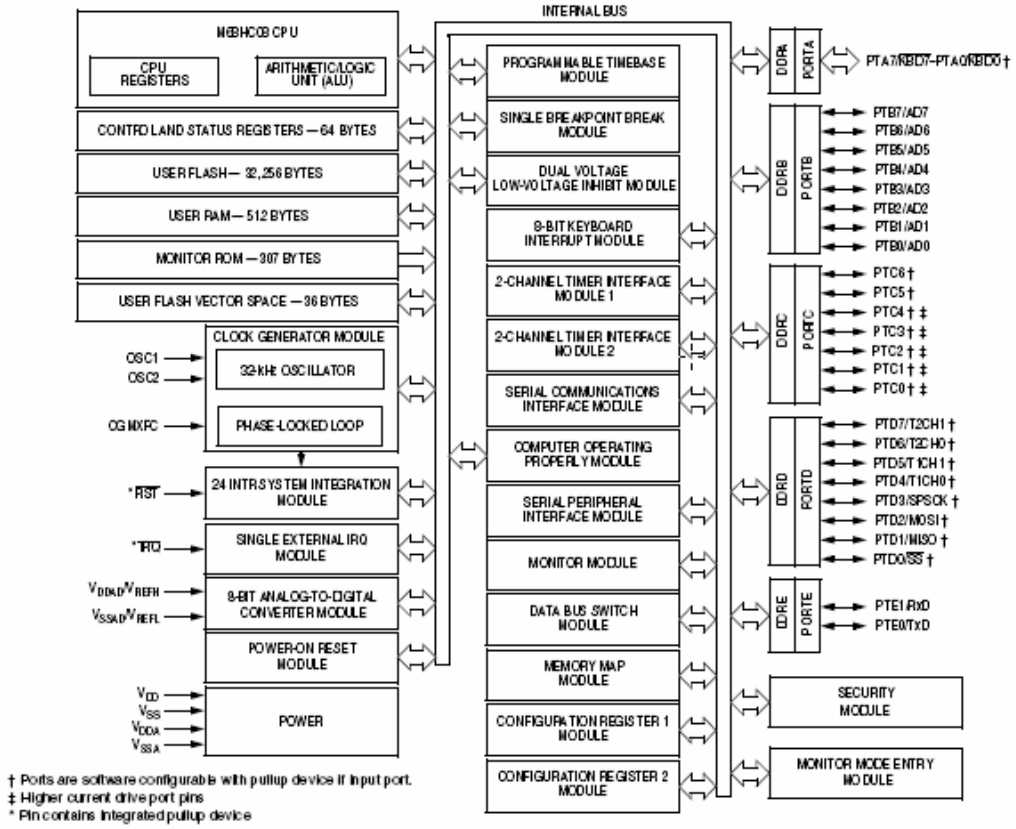


Figure 1-1. MCU Block Diagram

Naslovni prostor mikrokontrolerja (Memory Map)

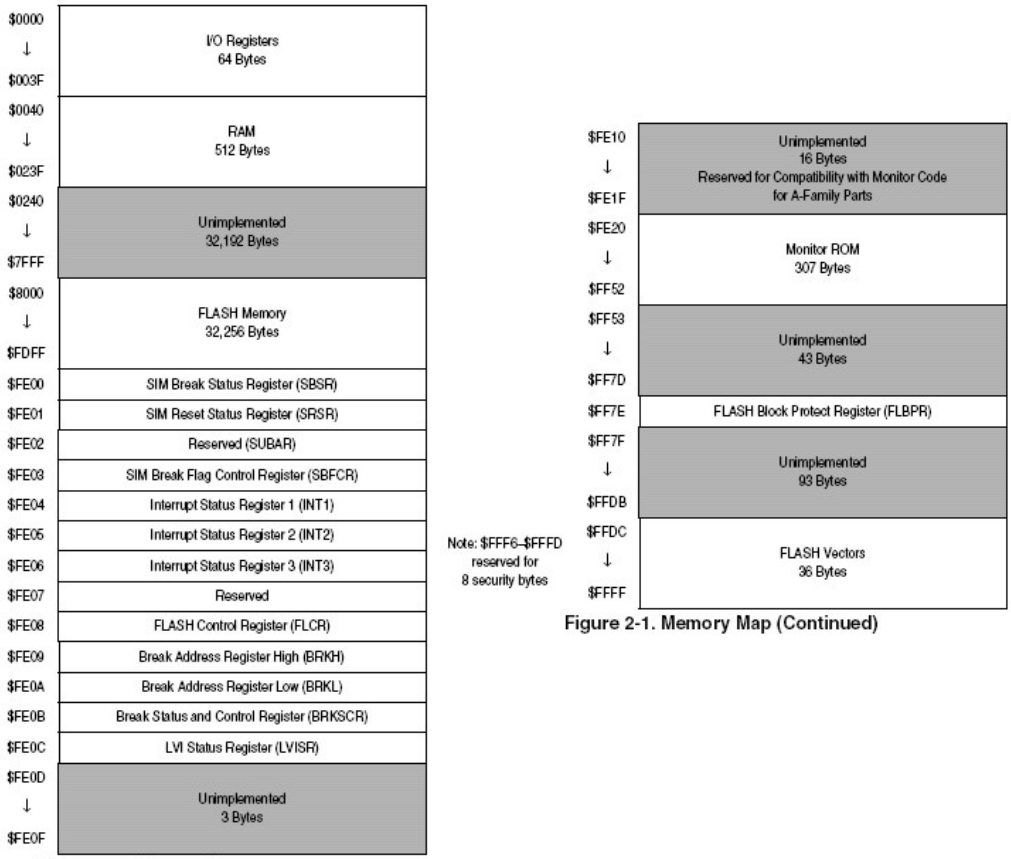


Figure 2-1. Memory Map

Figure 2-1. Memory Map (Continued)

Programski model in programiranje v zbirnem jeziku

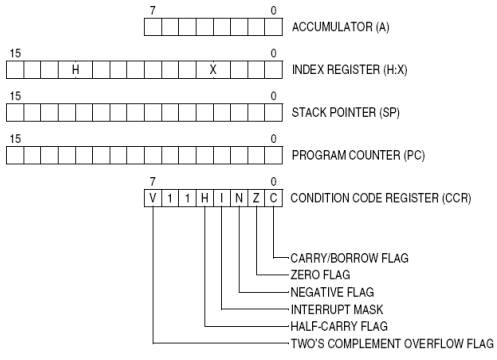


Figure 10-1. CPU Registers

Mikrokontroler sestavlja poleg ostalih enot tudi pet porabniku dostopnih registrov. Ti so:

1. A k u m u l a t o r A : je 8-bitni register, ki se uporablja za shranjevanje podatkov, izvajanje operacij nad podatki ter za shranjevanje rezultatov po izvedenih operacijah aritmetično-logične enote,
2. I n d e k s n i r e g i s t e r : je 16-bitni register; sestavljen je iz dveh delov – X (nižjega dela) in H (višjega dela). Uporabljamo ga pri indeksnem načinu naslavljanja,
3. P r o g r a m s k i š t e v e c : je 16-bitni register - vsebuje naslov naslednje operacije, ki jo mora procesor izvesti,
4. S k l a d o v n i k a z a l e c : je 16-bitni register; vsebuje začetni naslov spominskih lokacij skladovnega pomnilnika. Uporabljamo ga lahko tudi za indeksno naslavljanje,

5. R e g i s t e r s t a n j : je 8-bitni register, od tega sta dva bita neuporabljena, ostali biti pa se postavljajo na ustrezne vrednosti po vsaki izvršeni operaciji, če le-ta na njih vpliva. Opisuje lastnosti rezultata zadnjega izvršenega ukaza.

Pomen posameznih bitov registra stanj:

- C (*carry/borrow*): postavi se na logično vrednost 1, ko nastopi v rezultatu zadnje operacije prenos iz osmega bita naprej;
- V (*overflow*): postavi se na logično vrednost 1, ko nastopi v rezultatu zadnje operacije prekoračitev vrednosti +127 ali -128;
- Z (*zero*): postavi se na logično vrednost 1, ko je rezultat zadnje operacije nič;
- N *(negative)*: postavi se na logično vrednost 1, ko je rezultat zadnje operacije negativno število. V dvojiškem komplementu je z osmimi biti negativno število predstavljeno s sedmimi biti ter osmim (najvišjim) na logični vrednosti 1. Torej je N bit registra stanj enak najvišjemu bitu rezultata zadnje izvršene operacije;
- I *(interupt mask)*: Če je postavljen na logično vrednost 1, preprečuje izvajanje maskirne prekinitve IRQ. Postavi se na logično vrednost 1, ko se začne izvajati prekinitveni program. Z ukazoma SEI in CLI lahko med izvajanjem programa postavljamo ali brišemo I bit in s tem dovolimo ali preprečimo izvajanje prekinitvenih programov za prekinitveno zahtevo IRQ;
- H *(half-carry)*: postavi se na logično vrednost 1, ko nastopi v rezultatu zadnje operacije prenos med četrtim in petim bitom.

Zbirni jezik

Ker je procesor digitalno vezje, razume samo binarno kodirane ukaze. Zato ima vsak ukaz svojo binarno operacijsko kodo. Ker pa je takšno pisanje ukazov človeku manj jasno in pregledno, so nastali razni programski jeziki, ki nam olajšajo programiranje, s pomočjo prevajalnikov pa nato program prevedemo v binarno obliko, ki jo razume procesor. Programski jezik, ki je najbližji računalniku, hkrati pa tudi človeku razumljiv, je zbirni jezik ali asembler. Ukazi v tem jeziku so kratice ali okrajšave angleških izrazov za operacije, ki jih ta ukaz opravi. Tem kraticam pravimo mnemonične kode. Zaradi boljše preglednosti pišemo namesto binarnih vrednosti operacijske kode v šestnajstiški obliki.

Programe lahko pišemo tudi v višjih programskih jezikih in jih nato z ustreznim prevajalnikom prevedemo v binarno obliko (strojno kodo).

Program v zbirnem jeziku pišemo v štirih stolpcih. V prvem je oznaka vrstice (labela), če le-ta obstaja, v drugem je ukaz in v tretjem je operand (razen pri vsebujočem naslavljanju, ko operanda ni). Četrti stolpec lahko uporabimo za komentar (za podpičjem), ker ga prevajalnik ne upošteva. Stolpci so med seboj ločeni vsaj z enim presledkom. Če je cela vrstica komentar, mora biti prvi znak v vrstici zvezdica ali podpičje.

Ukazi in načini naslavljanja

Ta mikrokontroler pozna šest načinov naslavljanja:

- vsebujoče (*inherent* ali *implied*)
- takojšnje (*immediate*)
- neposredno ali direktno (*direct*)
- razširjeno (*extended*)
- relativno (*relative*)
- indeksno (*indexed*)

Vhodno-izhodne enote

- port A

  Registri: $0000 – Data Register (PTA)
  $0004 – Data Direction Register (DDRA)
  $000D – Input Pullup Enable Register (PTAPUE)

- port B

  Registri: $0001 – Data Register (PTB)
  $0005 – Data Direction Register (DDRB)

- port C

  Registri: $0002 – Data Register (PTC)
  $0006 – Data Direction Register (DDRC)
  $000E – Input Pullup Enable Register (PTCPUE)

- port D

  Registri: $0003 – Data Register (PTD)
  $0007 – Data Direction Register (DDRD)
  $000F – Input Pullup Enable Register (PTDPUE)

- port E

  Registri: $0008 – Data Register (PTE)
  $000C– Data Direction Register (DDRE)

Programski paket *WinIDE Development Environment*