3.2 MiMo - model CPE

ponedeljek, 02. november 2020 17:37

Značilnosti :

- pomnilniška beseda 16 bitov
- pomnilniški naslov 16 bitni
- dolžina ukazov 16 ali 32 bitov (2 formata)
  - Format 1 : (primer **ADD R1,R2,R3** # R1<-R2+R3, R1=Dreg, R2=Sreg, R3=Treg)
  - Format 2 : (primer **LI R1, 100** # R1<-100)

FORMAT 1:

Op.koda	Treg	Sreg	Dreg
7	3	3	3

ADD R1,R2,R3

FORMAT 2:

Op.koda	Treg	Sreg	Dreg
7	3	3	3
16 bitni tak. operand			
16			

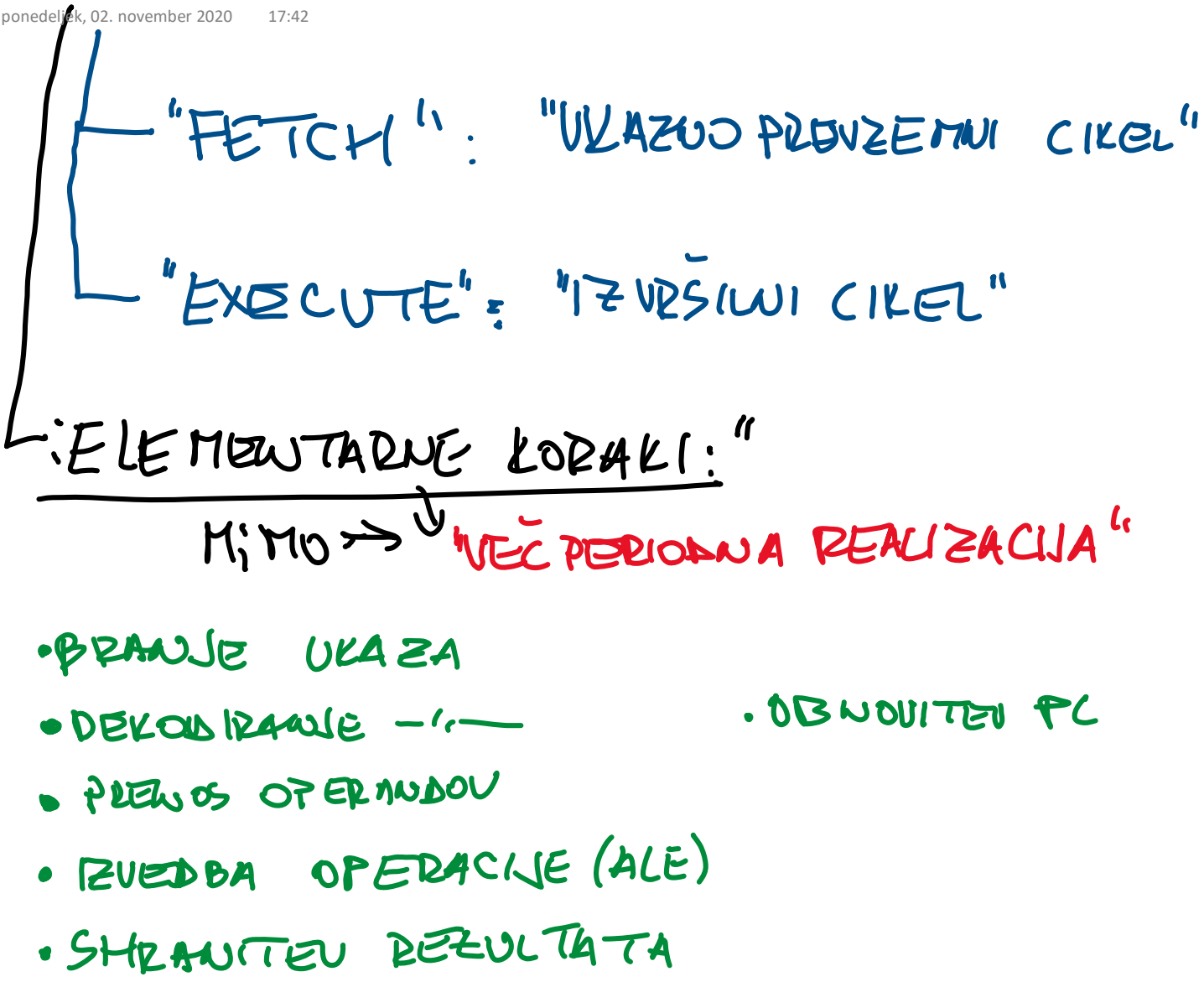
LI R1, 100

- registri: 8 x 16b R0-R7
  - 8x 16bitnih splošno namenskih registrov R0-R7
- operandi (pomnilniški dostopi) samo 16b
- pomnilniško preslikan vhod/izhod
- ~~prekinitve~~ NI!

MiMo temelji na tem viru: http://minnie.tuhs.org/Programs/UcodeCPU/index.html

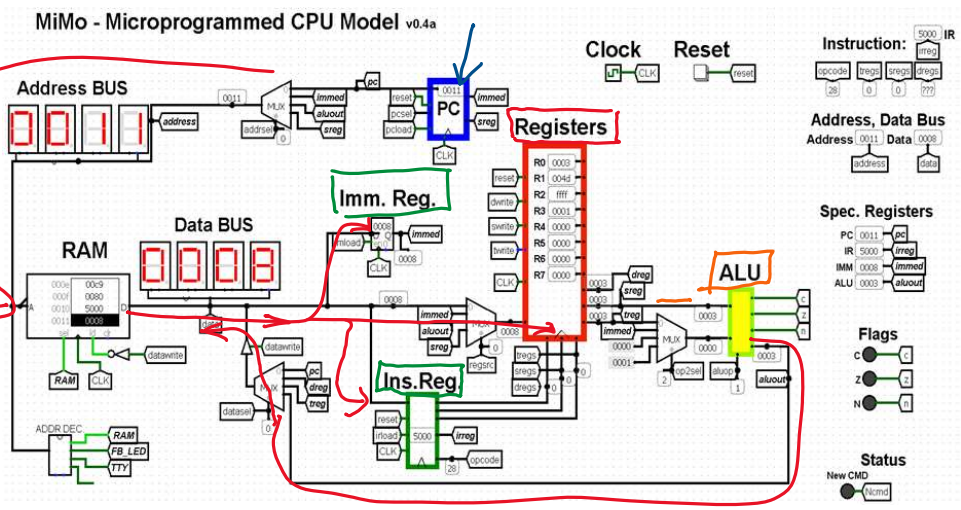
3.2.1 Izvrševanje ukazov

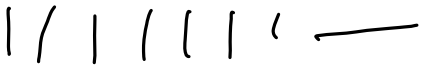
ponedeljek, 02. november 2020 17:42



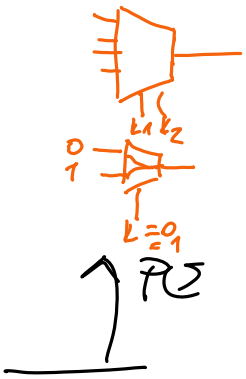
3.2.2 Podatkovna enota

ponedeljek, 02. november 2020 17:42



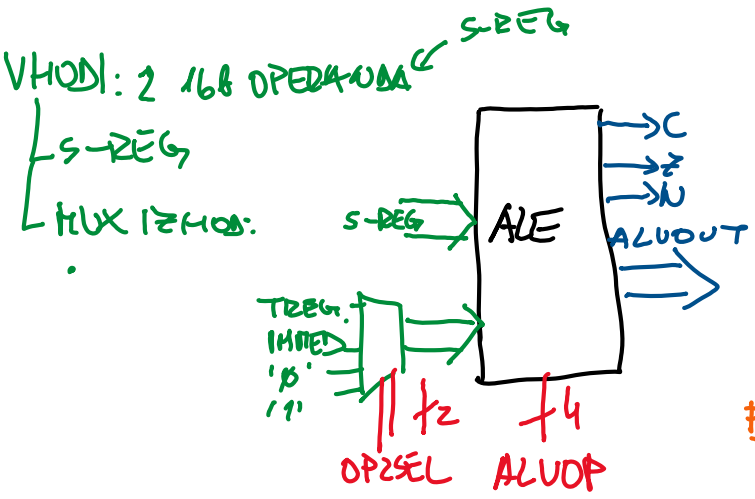


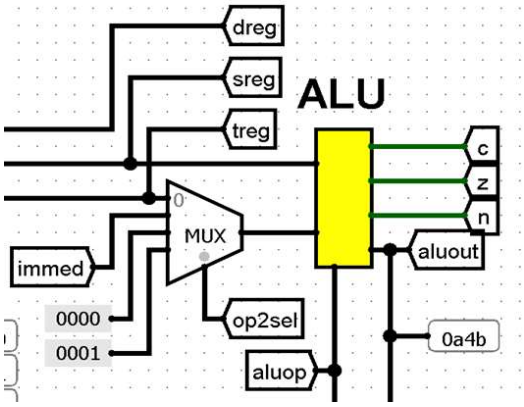
KE



ALE

ponedeljek, 02. november 2020 17:45





PRIMERI:

ADD R_D, R_S, R_T

ALUOP = 0 (+)

OP2SEL = TREG

JNEZ R_S, IMMED

$R_S \neq 0$: ALUOP = 1 (−)

$R_S - 0 = ?$ OP2SEL = "0"

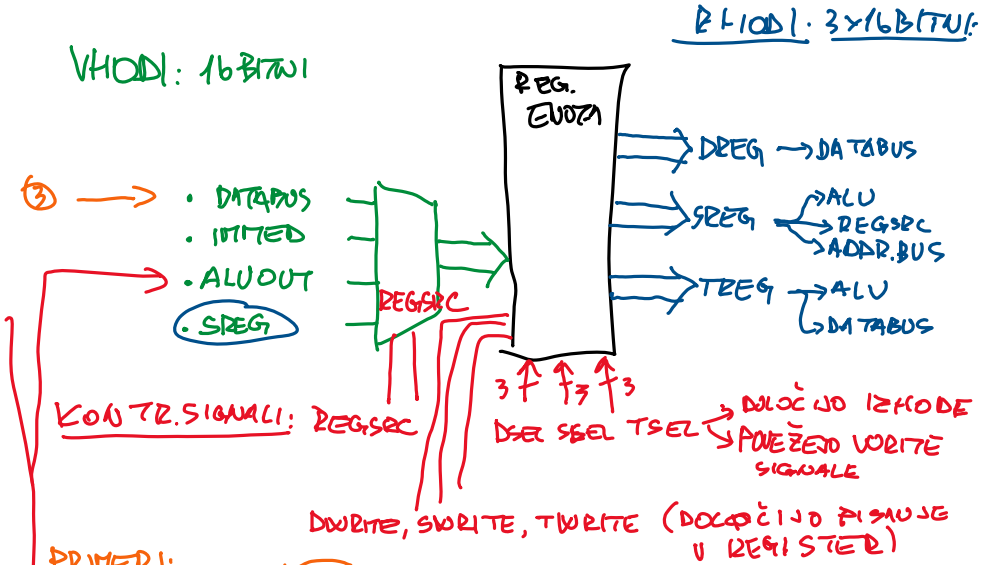
IZHODI:

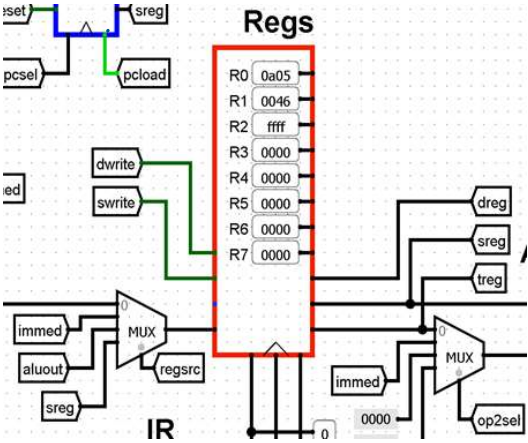
- 16b REZULTAT: ALUOUT
- ZASTAVICE: C, Z, N

KONTR. SIG.: - OP2SEL (2b) → DOLOČI 2. OPERAND
- ALU OP (4b) → DOLOČI OPERACIJO

Registri

ponedeljek, 02. november 2020 17:46





PRIMERI:

ADD R_D, R_S, R_T (DSEL SSEL TSEL)

REGSRC = ALU OUT

DWRITE = 1

MOV R_D, R_S

REGSRC = SREG

DWRITE = 1

LI Rd, IMMED (PREMOS TAK. OP. V Rd)

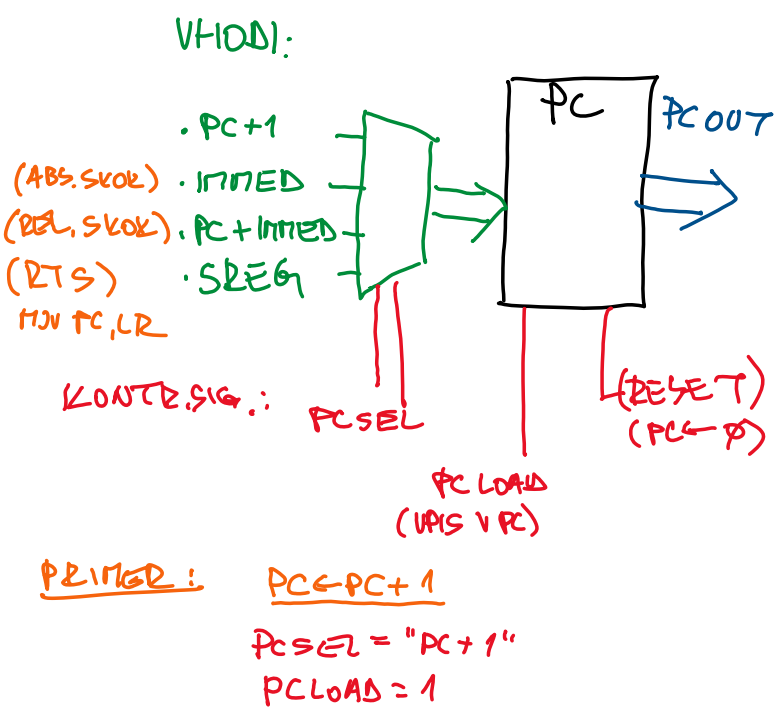
REGSRC = DATA BUS

DWRITE = 1

ADRSEL = PC

PC - programski števec

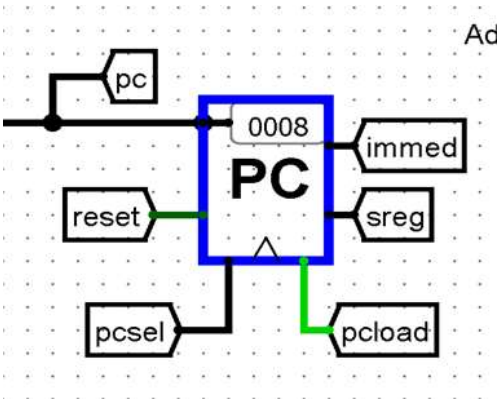
ponedeljek, 02. november 2020 17:49

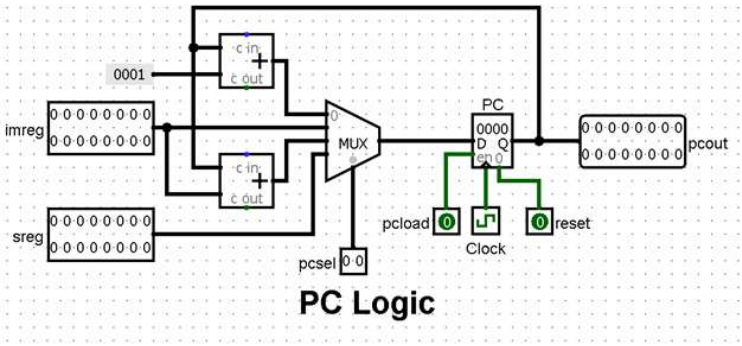


PRIMER: PC←PC+1

PCSEL = "PC+1"

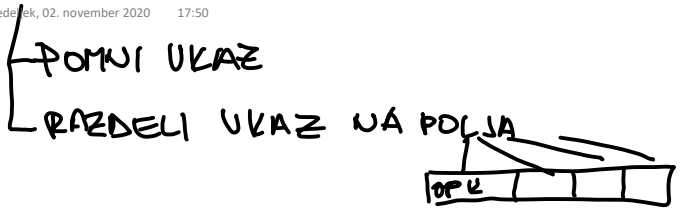
PCLOAD = 1

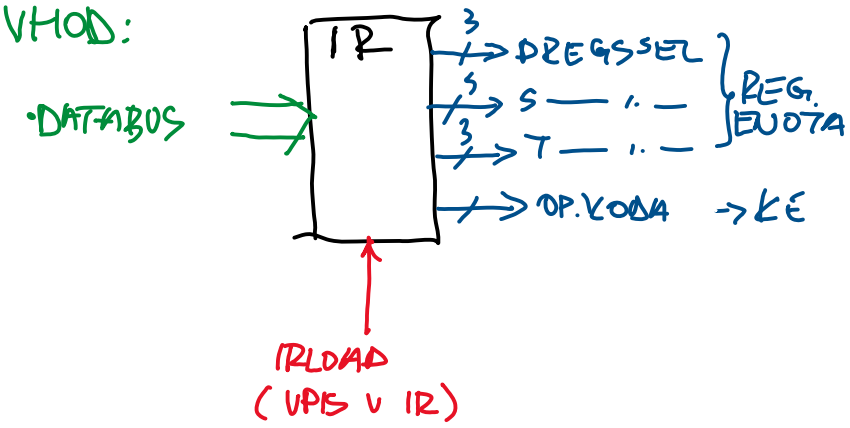


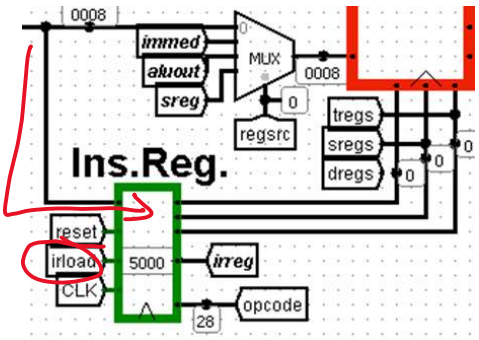


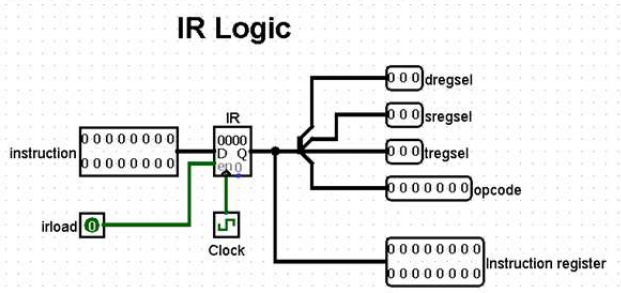
IR - ukazni register

ponedeljek, 02. november 2020 17:50









PRIMER: BRANJE UKAZA (FETCH)

IRLOAD = 1

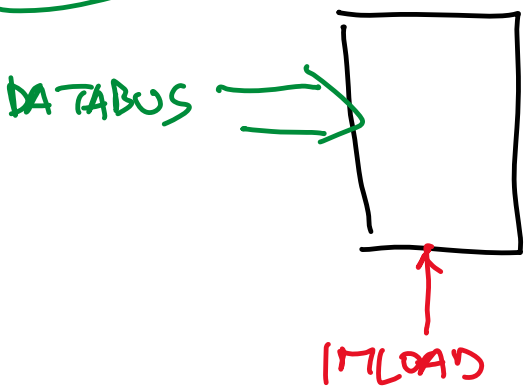
ADDRSEL = PC

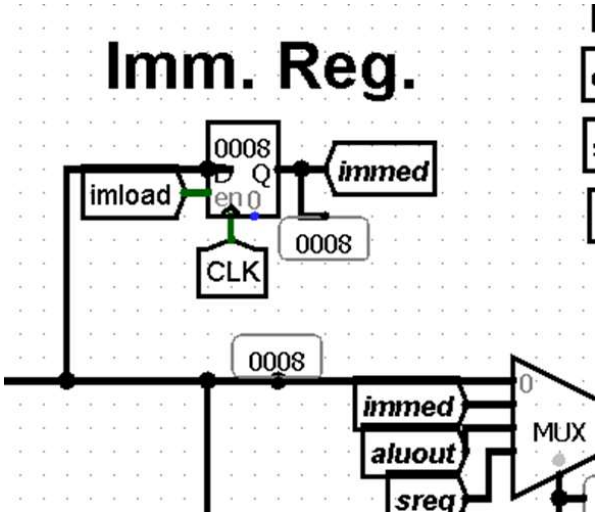
IM - Takojšnji register

ponedeljek, 02. november 2020 17:51

POMNI TAK. OPERAND

VHOD:





PRIMERA:

JNEZ R1, LOOP

- BRANJE TAK.OP.

IMLOAD = 1

(POMNIMO TAK.OP., KER JE TO SKOČNI NASLOV)

LI R_D, IMMED

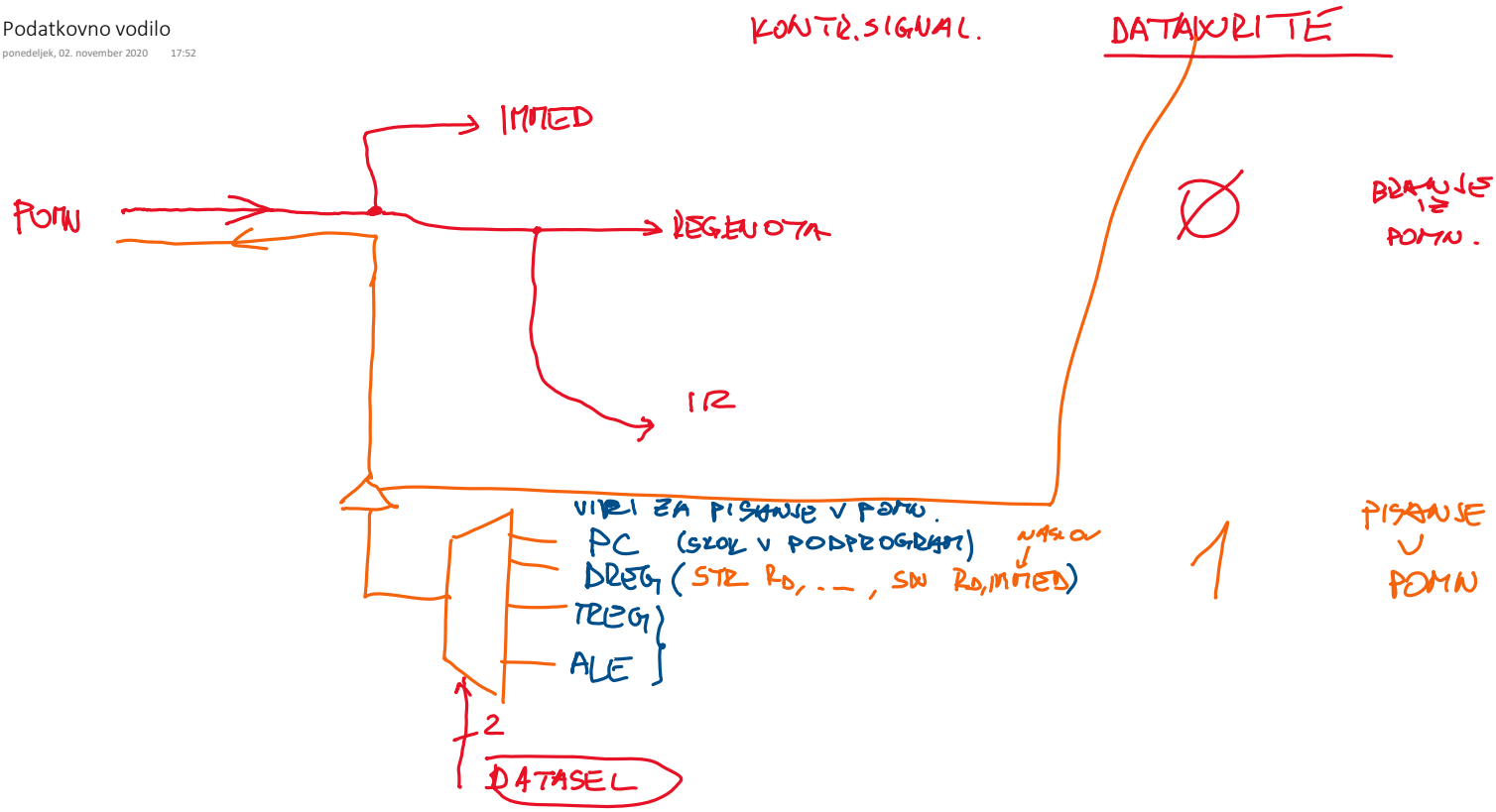
- VPIS TAK. OP. V R_D

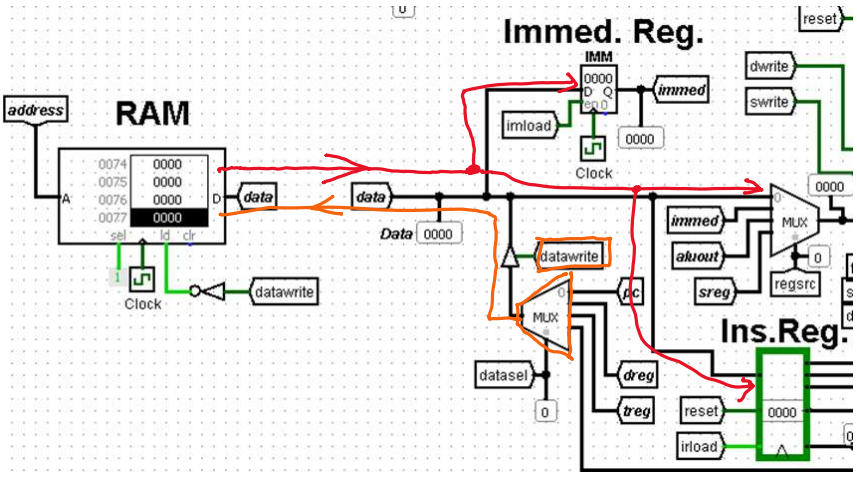
IMLOAD = 0

(PIŠEMO DIREKTNO V REGISTER) ← POMNENJE NI POTREBNO

Podatkovno vodilo

ponedeljek, 02. november 2020 17:52





PRIMER: SW Rd, IMMED

DATASEL = DREG

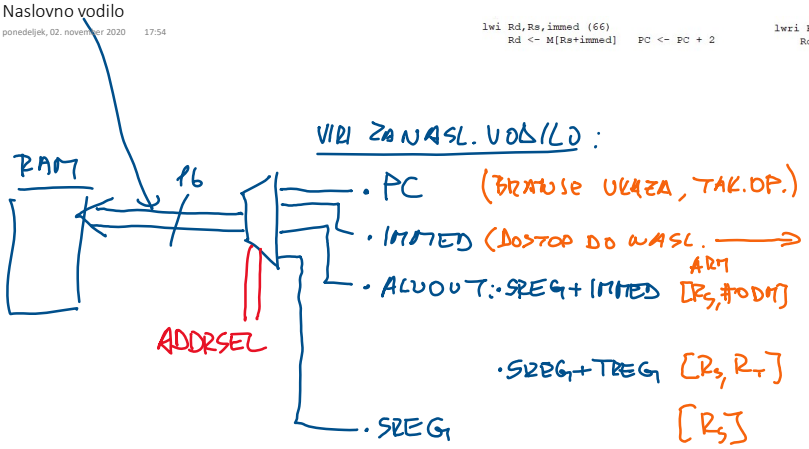
DATAWRITE = 1

ADDRSEL = IMMED (NASLOV V PCMN.)

Naslovno vodilo

ponedeljek, 02. november 2020 17:54

```
lwi Rd,Rs,immed (66)
    Rd <- M[Rs+immed]    PC <- PC + 2

lwri Rd,Rs,Rt (73)
    Rd <- M[Rs+Rt]       PC <- PC + 1
```

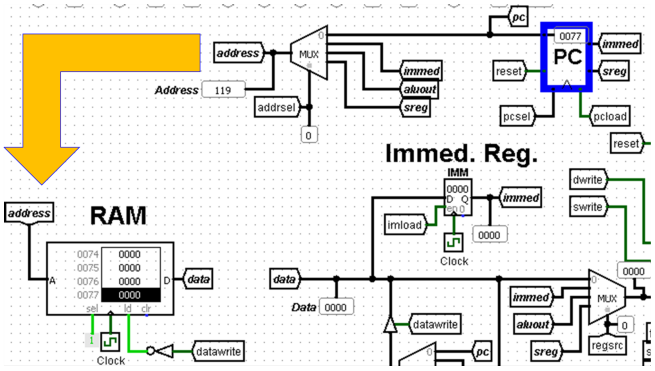


SW Rd, IMMED

DATASEL = DREG

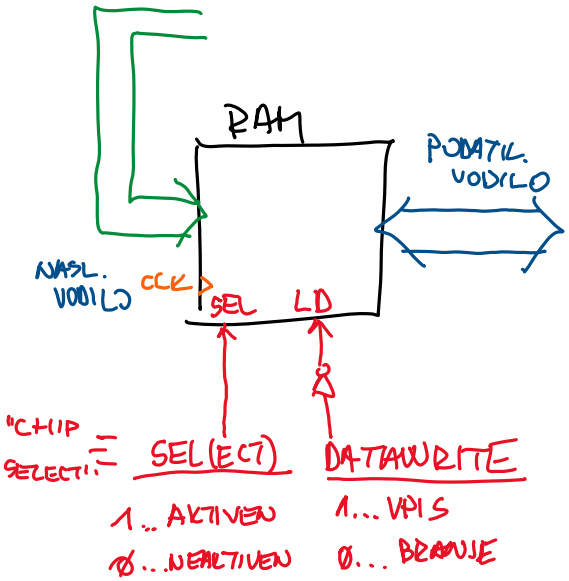
DATAWRITE = 1

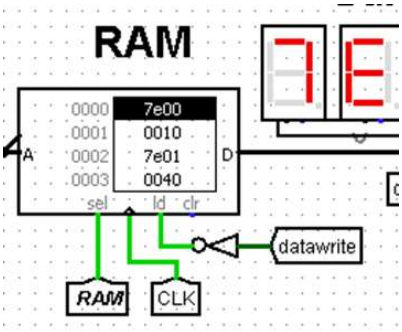
ADDRSEL = IMMED (NASLOV V POMO.)



RAM pomnilnik – 14-bitni

ponedeljek, 02. november 2020 17:54





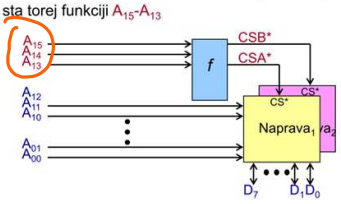
"CHIP SELECT" = SEL(ECT) DATAWRITE

1.. AKTIVEN 1... VPIS

0.. NEAKTIVEN 0... BRANJE

Izbira čipa (CS)

- **Kako priključimo dve (ali več) naprav na vodilo?**
  - Naenkrat mora biti izbran samo en čip (ali nobeden)
  - Za izbiro uporabimo naslednje signale:
    - R/W*, Naslov(A_0-A_{15})
- **Uporabni so biti, ki niso povezani na naslovne signale naprav (A_{15}-A_{13})**
- CSA* in CSB* sta torej funkciji A_{15}-A_{13}



MIMO { A15 A14 A13 } A0

NEPOPOLNO NASL. DEKODIRANJE

NASLOVNO DEKODIRANJE

00 RAM

01 FB_LED

10 TTY

11 ?

NASLOV

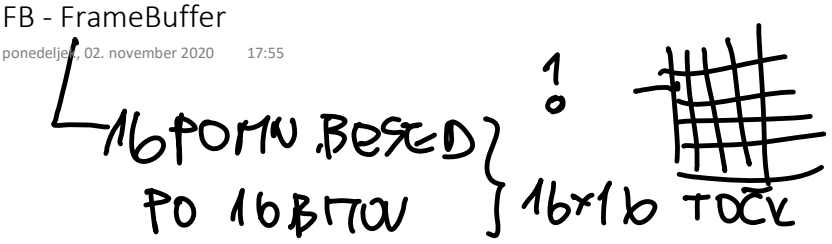
0 – 16383

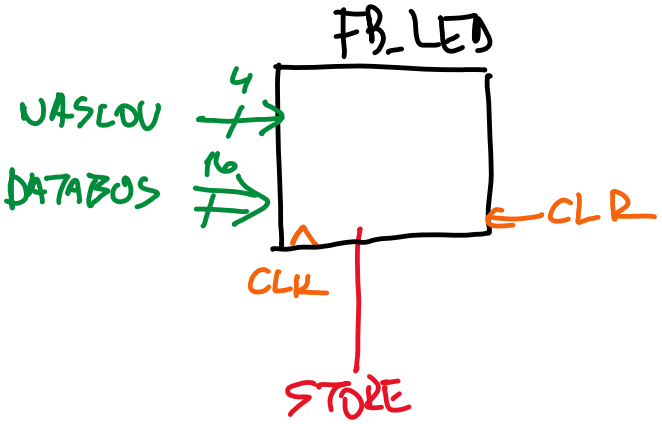
16384 – 32767 ←

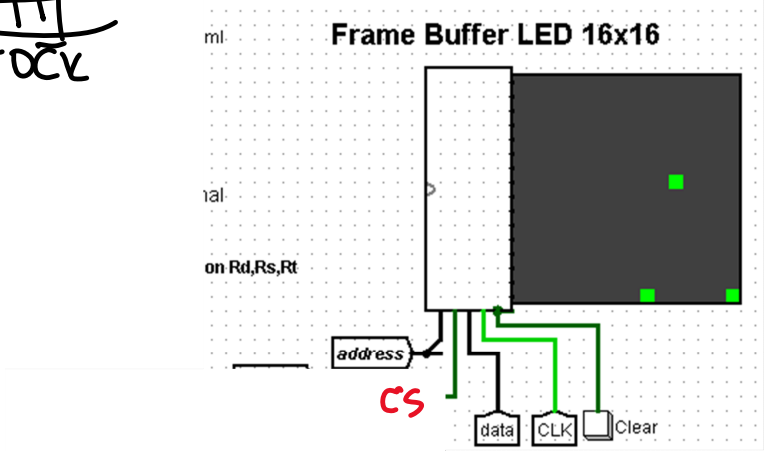
32768 –

FB - FrameBuffer

ponedeljek, 02. november 2020 17:55



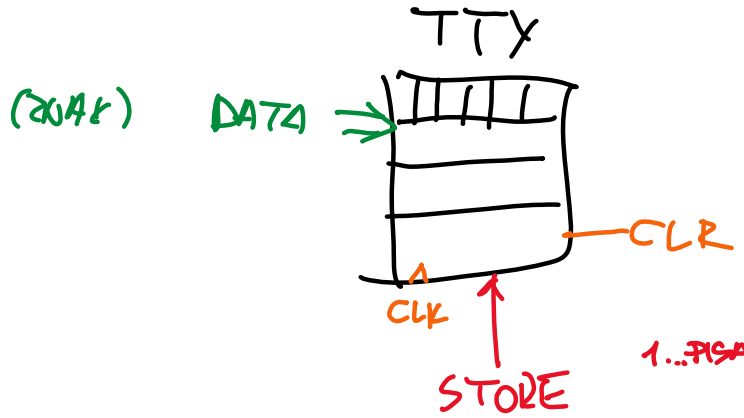


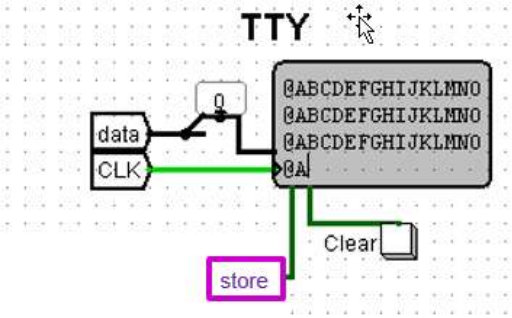


TTY - Serijski terminal

ponedeljek, 02. november 2020 17:56

- 4 VRSTICE PO 16 ZNAKOV





3.2.3 Kontrola enota

ponedeljek, 02. november 2020 17:57

→ NALOGA: • IZVEDBA UKAZOV

↓

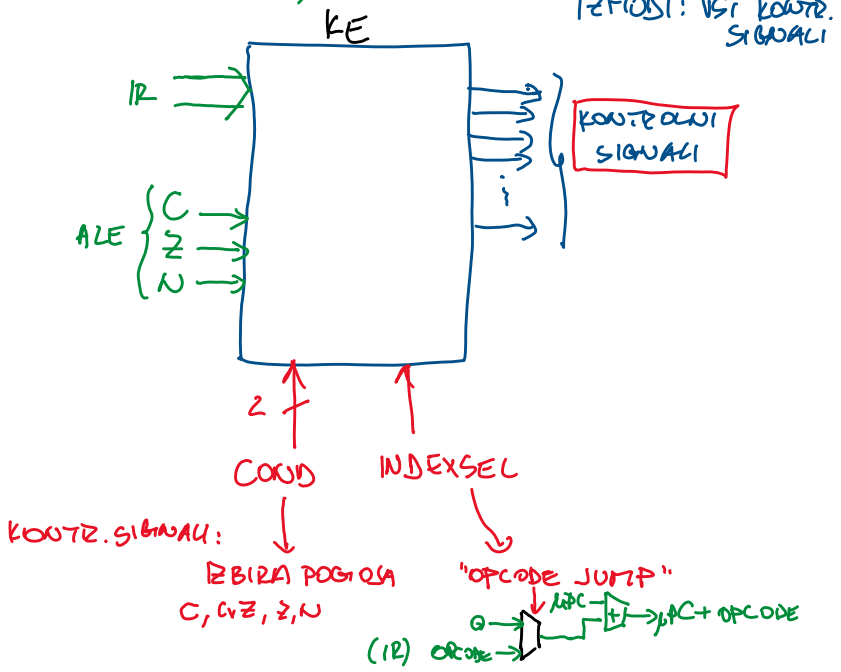
VSAK UKAZ PO VSEH ELEMENTARNIH KORAKIH

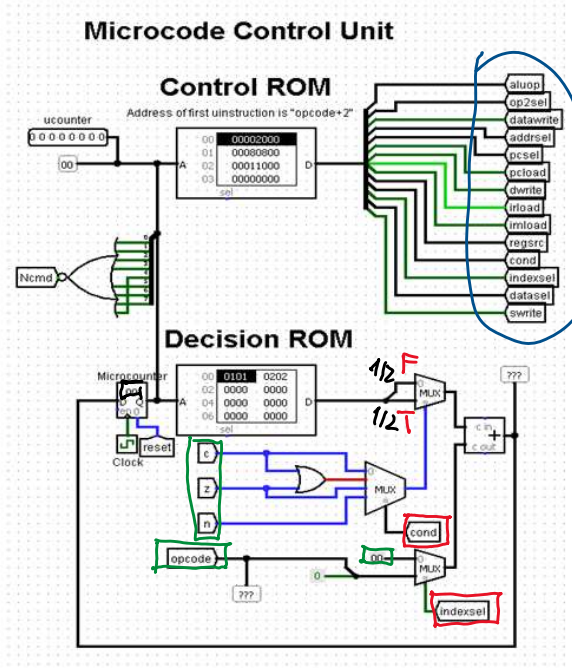
1 ELEMENT. KORAK ≡ 1 μ-UKAZ

→ MORA DOLOČITI:

- STANJE KONTROLNIH SIGNALOV
- NASLEDNJI μ-UKAZ

VHODI: • UKAZ (IR)
• ZASTAVICE (ALE)

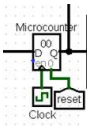




SESTAVA KE:

- μPC → DOLOČA NASLOV μ-UKAZA

MIKRO



- 2 KONTROLNA POMNILNIKA

  - 32-BITNI "CONTROL ROM" ← KONTR. SIG.
    - DOLOČA STANJE KONTR. SIGNALOV (23x)

    1 POMN. BES. → 1 μ UKAZ (KONTR. SIGNALI)

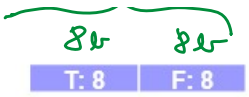
  - 16-BITNI "DESICION ROM" ← NASL. μ-UKAZ
    - DOLOČA NASLOV NASL. μ-UKAZA

8b 8b

T: 8 F: 8

PRIMERA: T F

(5, 5) ... BREZPOG. SKOK



PRIMERA:
T F
(5,5) ... BREZPOG. SKOK
(5,4) ... T: μPC ← 5, F: μPC ← 4

PRIMERA ŽA ICE:

① OPCODE_JUMP:
- INDEXSEL = 1
- μPC ← NASLOV NASL. μVILA ZA (Z + OPCODE)
  ↓
  μPC

② IF Z THEN PCINCR ELSE JUMP
LABELI
J
- COND = Z (2)
- DECISION ROM

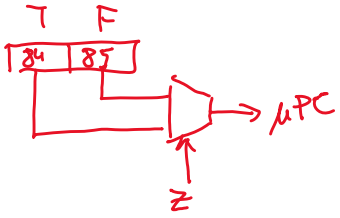
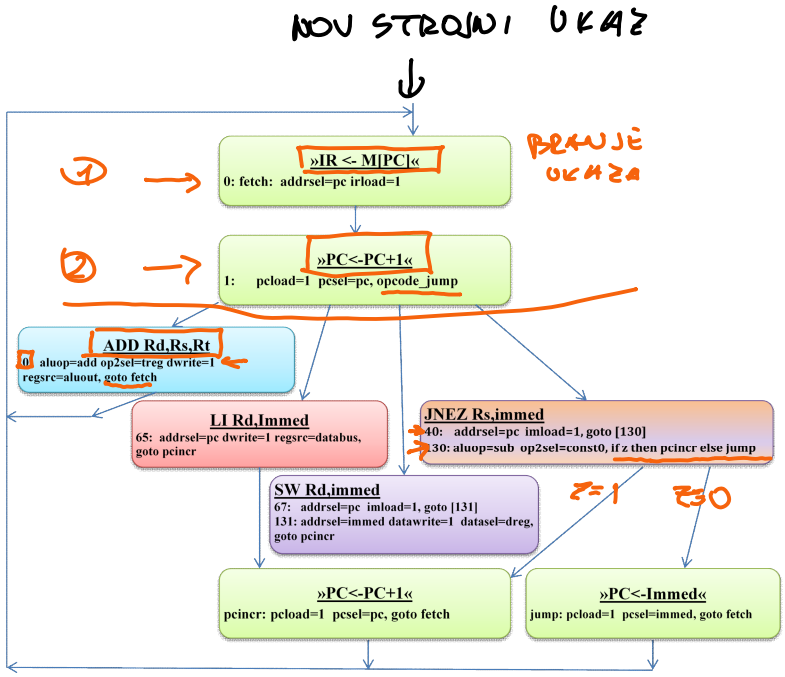


Diagram prehajanja stanj

ponedeljek, 09. november 2020 20:59

- OBIČAJEN KONCEPT OPISOVANJA KE
- RAZVIDNA IZVEDBA UKAZOV
- OSNOVA OPTIMIZACIJ ③



KOMENTAR

- Mikroukaz : (63: addrsel=pc dwrite=1 regsrc=databus, goto pcincr)

VRSTICA

Op.koda	kontrolni signali	naslednji mikroukaz

- OP.KODA
- LOG.OZNAKA

kontr. signal	opisna vrednost	enota
aluop	add, sub, mul, div, rem, and, or, xor, nand, nor, not, lsl, lsr, asr, rol, ror	ALE
op2sel	treg, immed, const0, const1	ALE
addrsel	pc, immed, aluout, sreg	nasl. vodilo
pcsel	pc, immed, pcimmed, sreg	PC
regsrc	databus, immed, aluout, sreg	registri
cond	c, corz, z, n	kontr. enota

NASL. μ-UKAZ:

- PRAZNO → NASLEDNJI μ-UKAZ
- "GOTO OZNAKA" → V VRSTICO "OZNAKA"
- "IF POGOJ THEN OZNAKA1 ELSE OZNAKA2" (POGOJ ↔ COND)
- "IF POGOJ THEN OZNAKA" → ISCIČE NASL. μ-UKAZ
- "OPCODE_JUMP" → SKOK NA 2+OPCODE (SAMO 1K/UKAZ)

datoteka basic_microcode.def

```
fetch:   addrsel=pc irload=1                    # Address=PC, Load IR register
         pcload=1 pcsel=pc, opcode_jump         # PC=PC+1, jump to 2+OPC

# ALU operation '+' on Rd,Rs,Rt
0:       aluop=add op2sel=treg dwrite=1 regsrc=aluout, goto fetch

# JNEZ Rs,immed
40:      addrsel=pc imload=1
         aluop=sub op2sel=const0, if z then pcincr else jump

# li Rd,Immed
63:      addrsel=pc dwrite=1 regsrc=databus, goto pcincr

# Rd->M[immed]
65:      addrsel=pc imload=1
         addrsel=immed datawrite=1 datasel=dreg, goto pcincr

pcincr:  pcload=1 pcsel=pc, goto fetch

jump:    pcload=1 pcsel=immed, goto fetch
```

OP.KODA → 4012=42 → 130+

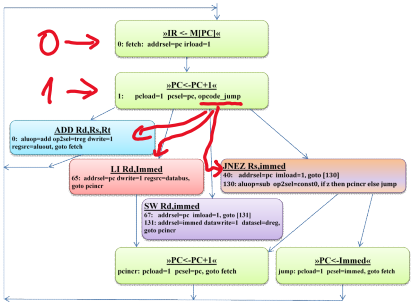
LOG.OZNAKA

- RAZPORED μUKAZOV V KONTR. POMN:

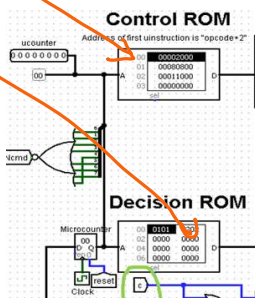
Naslov	Vsebina
0	BRANJE UKAZA
1	PC←PC+1 , OPCODE_JUMP
2-129	PRVI μ-UKAZI ZA VSEH 128 MOŽNIH STROJNIH UKAZOV
130+ 0x82+	VSI PREOSTALI μUKAZI

0..127

2+ 130+



.\MICRO_ASSEMBLER.EXE BASIC_MICROCODE.DEF → UCONTROL.ROM, UDECISIONROM → v MIMO



datoteka **basic_microcode.def**

```
fetch:   addrsel=pc irload=1                    # Address=PC, Load IR register
         pcload=1 pcsel=pc, opcode_jump         # PC=PC+1, jump to 2+OPC

# ALU operation '+' on Rd,Rs,Rt
0:       aluop=add op2sel=treg dwrite=1 regsrc=aluout, goto fetch

# JNEZ Rs,immed
40:      addrsel=pc imload=1
         aluop=sub op2sel=const0, if z then pcincr else jump

# li Rd,Immed
63:      addrsel=pc dwrite=1 regsrc=databus, goto pcincr

# Rd->M[immed]
65:      addrsel=pc imload=1
         addrsel=immed datawrite=1 datasel=dreg, goto pcincr

pcincr:  pcload=1 pcsel=pc, goto fetch

jump:    pcload=1 pcsel=immed, goto fetch
```

se prevede v indexsel, pcload

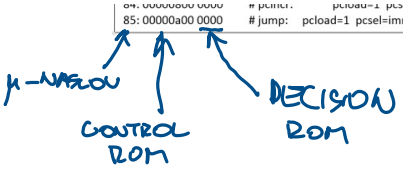
```
00: 00002000 0101    # fetch: addrsel=pc irload=1
01: 00080800 0202    #          pcload=1 pcsel=pc, opcode_jump
02: 00011000 0000    # 0:       aluop=add op2sel=treg dwrite=1 regsrc=aluout,    goto fetch
2a: 00004000 8282    # 40:      addrsel=pc imload=1                              (goto 82)
41: 00001000 8484    # 63:      addrsel=pc dwrite=1 regsrc=databus,              goto pcincr
43: 00004000 8383    # 65:      addrsel=pc imload=1                              (goto 83)
82: 00040021 8485    #          aluop=sub op2sel=const0,                         if z then pcincr else jump
83: 001000c0 8484    #          addrsel=immed datawrite=1 datasel=dreg,          goto pcincr
84: 00000800 0000    # pcincr:  pcload=1 pcsel=pc,                               goto fetch
85: 00000a00 0000    # jump:    pcload=1 pcsel=immed,                            goto fetch
```

42:

μ-ZBIRNIK

μ-NASLOV

```
84: 00000800 0000    # pcinc:    pcload=1  pcsel=pc,                    goto fetch
85: 00000a00 0000    # jump:     pcload=1  pcsel=immed,                 goto fetch
```



3.2.5 Zbirnik

ponedeljek, 09. november 2020 22:01

- **Primer (testni):**

```
main:   li    r0, 0           # r0 is the running sum
        li    r1, 100         # r1 is the counter
        li    r2, -1          # Used to decrement r1
loop:   add   r0, r0, r1      # r0= r0 + r1
        add   r1, r1, r2      # r1--
        jnez  r1, loop        # loop if r1 != 0
        sw    r0, 256         # Save the result
inf:    jnez  r2, inf         # loop if r1 != 0 -> loop forever
```

PROGRAM

ZBIRNIK

```
0000: 00007e00  0111111000000000        main:   li      r0, 0
0001: 00000000  0000000000000000
0002: 00007e01  0111111000000001                li      r1, 100
0003: 00000064  0000000001100100
0004: 00007e02  0111111000000010                li      r2, -1
0005: 0000ffff  1111111111111111
0006: 00000040  0000000001000000        loop:   add     r0, r0, r1
0007: 00000089  0000000010001001                add     r1, r1, r2
0008: 00005008  0101000000001000                jnez    r1, loop
0009: 00000006  0000000000000110
000a: 00008200  1000001000000000                sw      r0, 256
000b: 00000100  0000000100000000
000c: 00005010  0101000000010000        inf:    jnez    r2, inf
000d: 0000000c  0000000000001100
```

OPK

NASLOVI HEX BIN

RAM POMN.

.\ASSEMBLER.EXE IME.S → IME.RAM

Zbirnik – primeri ukazov

rdeče: trenutno že implementirani ukazi v modelu MiMo v04b.

assembler.pl (zbirnik), list_of_instructions.txt (dokumentacija) :

- add Rd,Rs,Rt (0) — Rd <- Rs + Rt, PC <- PC + 1
- sub Rd,Rs,Rt (1) — Rd <- Rs – Rt, PC <- PC + 1
- …
- jeqz Rs,immed (39) — if Rs == 0, PC <- immed else PC <- PC + 2
- jnez Rs,immed (40) — if Rs != 0, PC <- immed else PC <- PC + 2
- …
- beq Rs,Rt,immed (46) — if Rs == Rt, PC <- PC + immed else PC <- PC + 2
- bne Rs,Rt,immed (47) — if Rs != Rt, PC <- PC + immed else PC <- PC + 2
- …
- li Rd,immed (63) — Rd <- immed, PC <- PC + 2
- sw Rd,immed (65) — M[immed] <- Rd, PC <- PC + 2
- …
- lw Rd,immed (64) — Rd <- M[immed], PC <- PC + 2
- lwi Rd,Rs,immed (66) — Rd <- M[Rs+immed], PC <- PC + 2
- swi Rd,Rs,immed (67) — M[Rs+immed] <- Rd, PC <- PC + 2

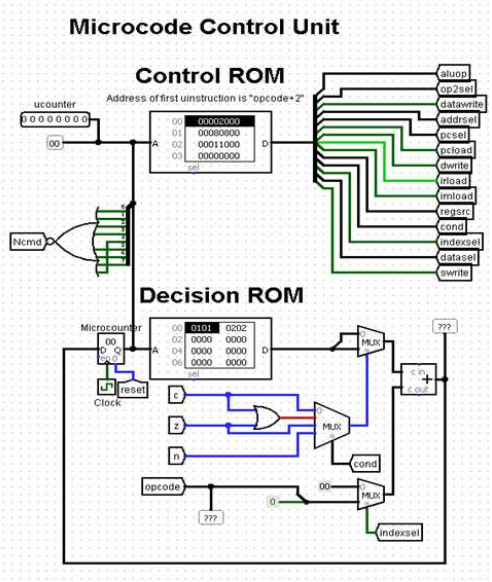
OP. KODA OPIS

3.2.6 Mikroprogramirana vs trdoožičena CPE

ponedeljek, 09. november 2020 22:01

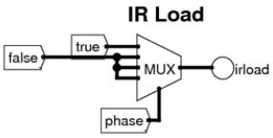
datoteka **basic_microcode.def**

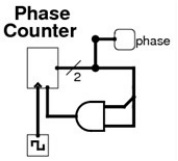
```
fetch:   addrsel=pc irload=1                    # Address=PC, Load IR register
         pcload=1  pcsel=pc, opcode_jump        # PC=PC+1, jump to 2+OPC

# ALU operation '+' on Rd,Rs,Rt
0:       aluop=add  op2sel=treg  dwrite=1  regsrc=aluout, goto fetch

# JNEZ Rs,immed
40:      addrsel=pc  imload=1
         aluop=sub  op2sel=const0, if z then pcincr else jump

# li Rd,Immed
63:      addrsel=pc  dwrite=1  regsrc=databus, goto pcincr

# Rd->M[immed]
65:      addrsel=pc  imload=1
         addrsel=immed  datawrite=1  datasel=dreg, goto pcincr

pcincr:  pcload=1  pcsel=pc, goto fetch

jump:    pcload=1  pcsel=immed, goto fetch
```

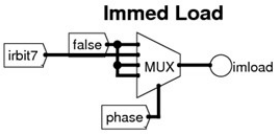


Elementarni koraki:

- 0 .. Branje ukaza -> IR
- 1 .. Format 2: branje operanda
  Format 1: nop
- 2 .. Vse operacije :
  - ALE, skok, reg.write, R/W from Mem







irbit7:

- 0 .. 8-bitni ukaz (1bajt)
- 1 .. 16-bitni ukaz (2 bajta)