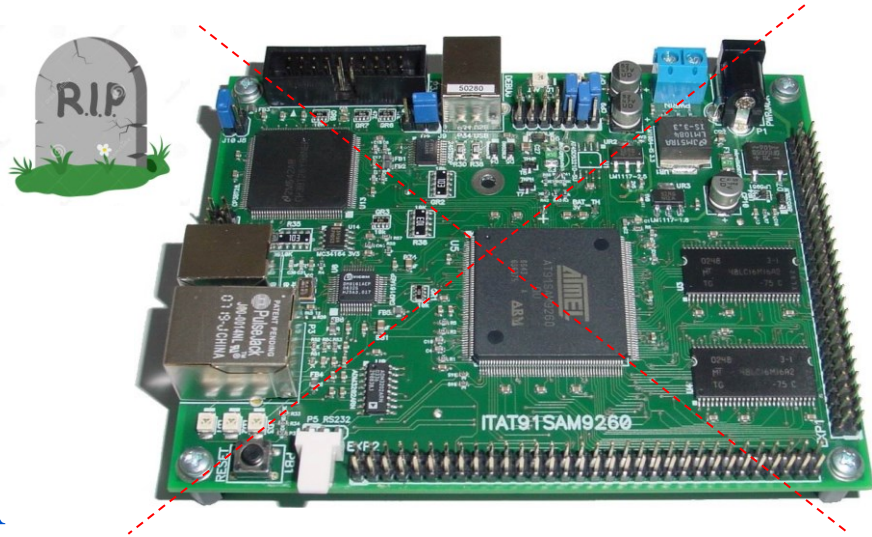
Računalniška arhitektura RA

Računalnik STM32H750-DK

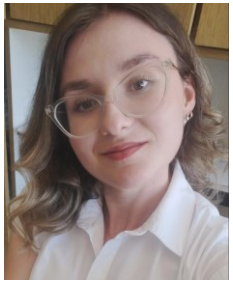
- Računalnik FRI-SMS
  - Mikrokrmilnik AT91SAM9260 iz družine mikrokrmilnikov ARM9



Ekipa RA

Asistentke, asistenti

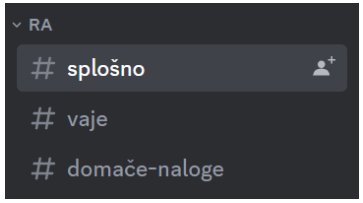
Žiga Pušnik
ziga.pusnik@fri....



Romanela Lajić
romanela.lajic@fri....

Mira Trebar
mira.trebar@fri...

Tutorji



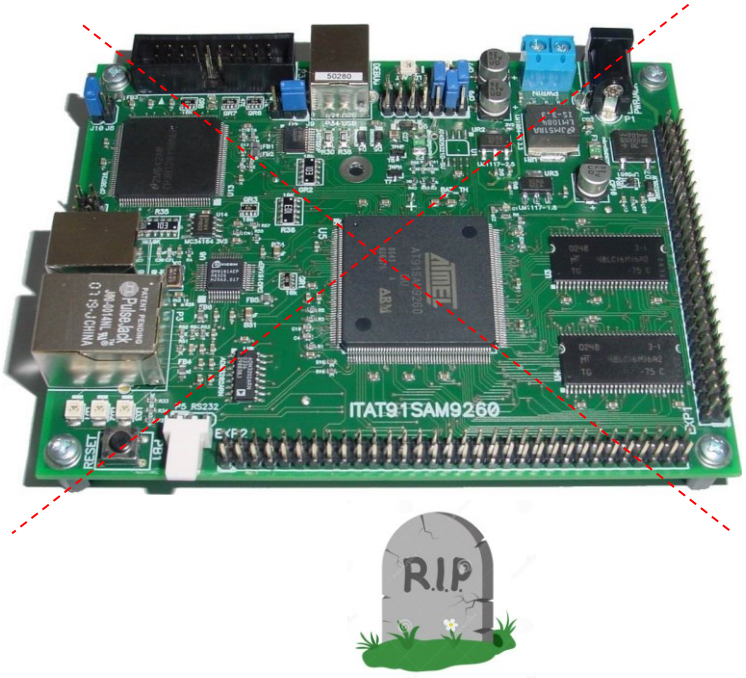
https://discord.gg/nmzjQU7me7

Robert Rozman
rozman@fri.uni-lj.si

Računalniška arhitektura RA

Računalnik STM32H750-DK

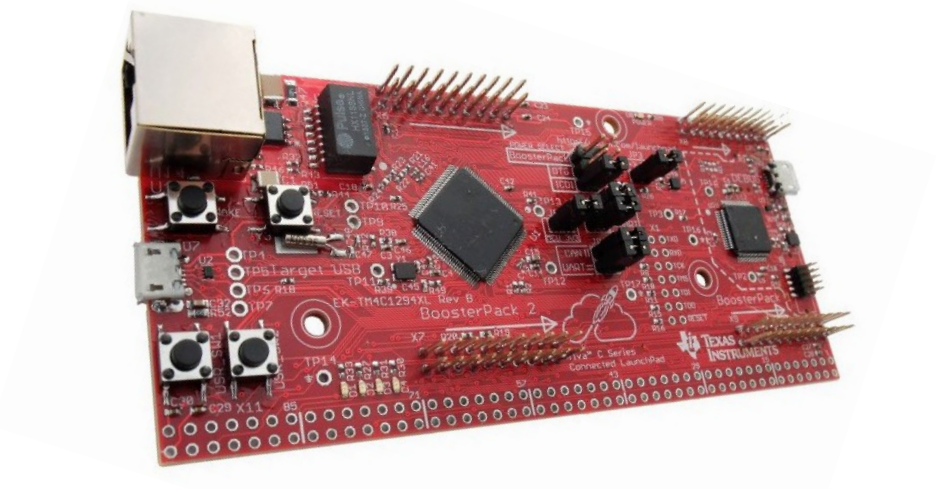
- Računalnik FRI-SMS
  - Mikrokrmilnik AT91SAM9260 iz družine mikrokrmilnikov ARM9

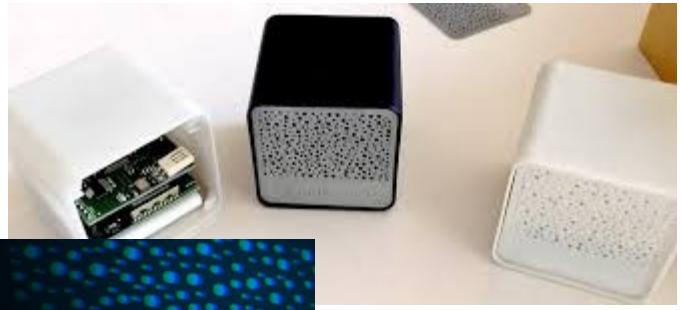


LAB 1.1 Splošne informacije

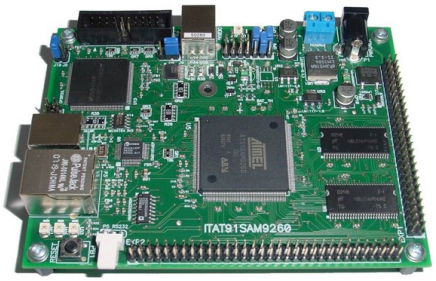
Laboratorijske vaje RA

- Spoznati osnove računalniške arhitekture s praktičnega vidika
- Razumeti delovanje računalnika (ARM) s programiranjem v zbirnem jeziku
- Podrobnejši vpogled:
  - v delovanje računalnika
  - v izvajanje programov na računalniku
- Vsebinske nadgradnje -> predmeti Organizacija računalnikov, Vhodno izhodne naprave in ostali





Vsebina vaj



- Potrebne osnove s predavanj (npr. pomnilniški naslov, vsebina, …)
- Jedro: Programiranje v zbirnem jeziku ARM
- Oblika:
  - Sprotne vaje (2. praktični vaji) + domača naloga
- Tri preverjanja (november, december, januar)
- Priprava na izpit (avditorne naloge)
- Predmetni seminar po dogovoru z asistentom
- Video gradiva (ponovitev, utrditev):
  - Računalniška arhitektura (RA) (sharepoint.com)

Ocenjevanje

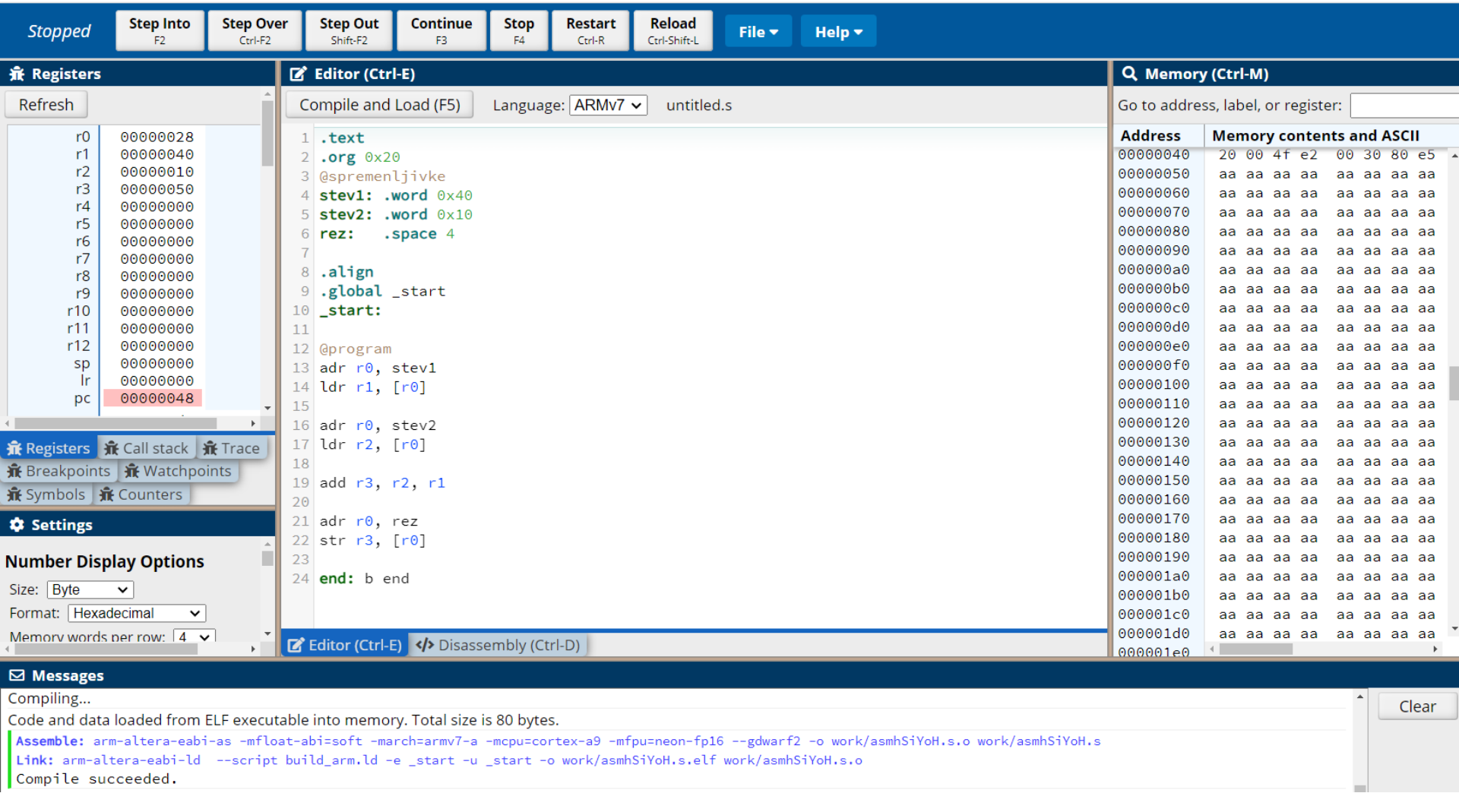
Vaje prispevajo 50% h končni oceni in morajo biti opravljene naslednje obveznosti:

- Uspešno opraviti sprotne naloge in biti prisoten na laboratorijskih vajah
- Uspešno oddati in zagovarjati domačo nalogo,
- Tri preverjanja (80 + 100 + 120 točk)
  - skupaj potrebno zbrati vsaj 150 točk (50%)
  - ni omejitev na posameznih preverjanjih

- Ocena vaj velja le v tekočem študijskem letu. Kdor v istem letu ne opravi predmeta v celoti, mora prihodnje leto ponovno opraviti vaje.

Spletni simulator cpulator

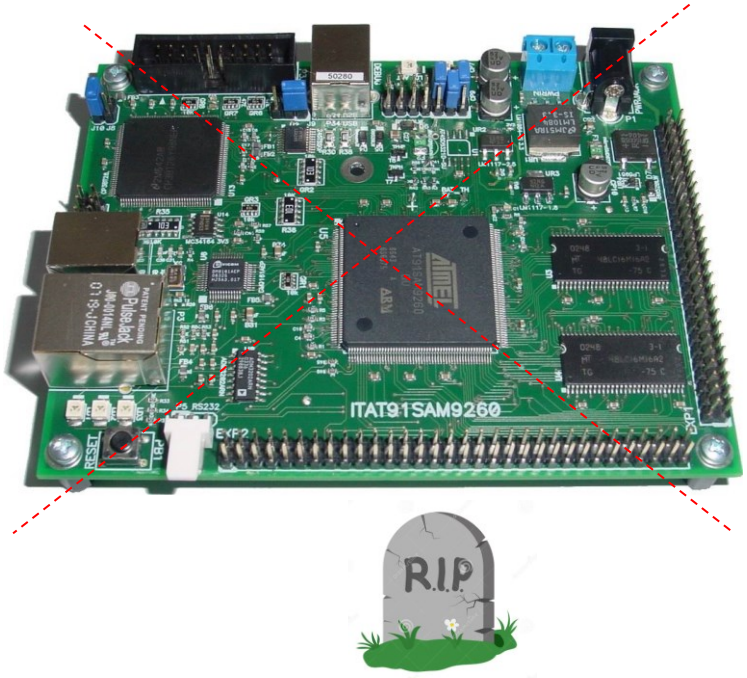
- https://cpulator.01xz.net/?sys=arm
- začetni projekt RA:
  - https://cpulator.01xz.net/?sys=arm&loadasm=share/sg8LlNt.s



Računalniška arhitektura RA

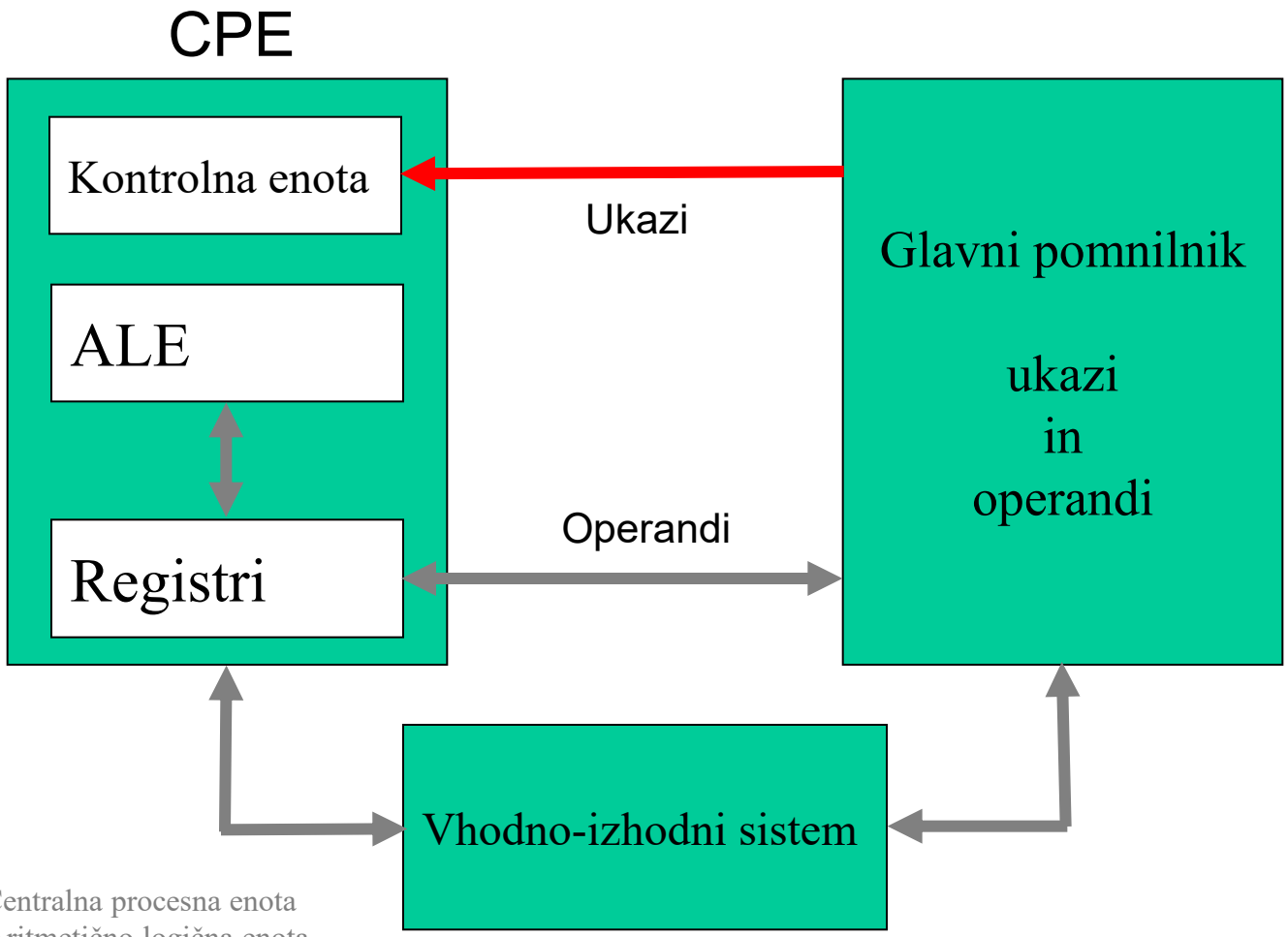
Računalnik STM32H750-DK

- Računalnik FRI-SMS
  - Mikrokrmilnik AT91SAM9260 iz družine mikrokrmilnikov ARM9



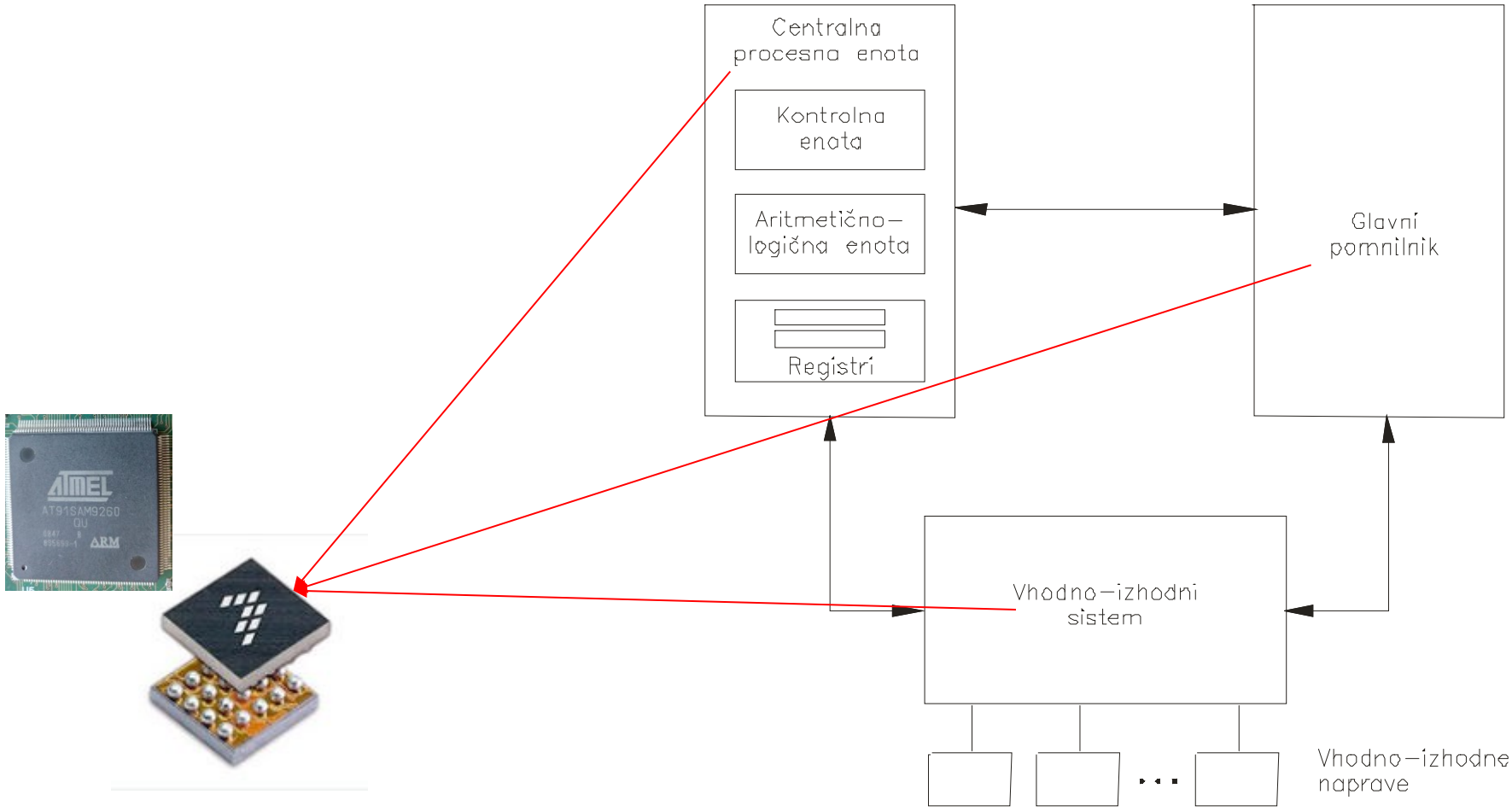
LAB 1.2 Von Neumannov model (VN)

Von Neumannov računalniški model



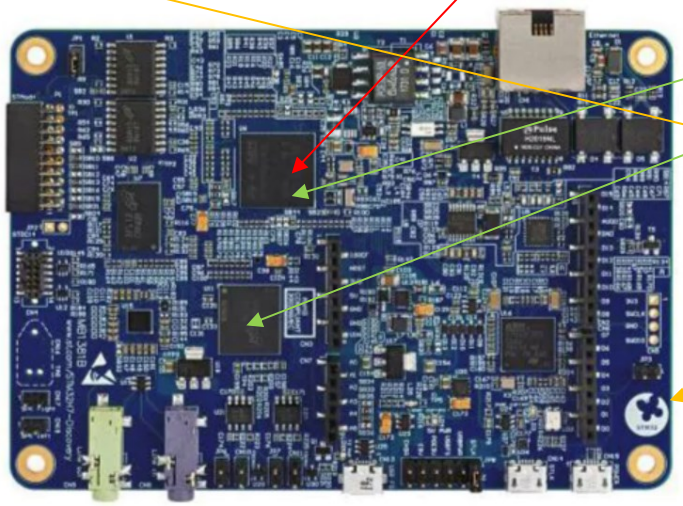
CPE – Centralna procesna enota
ALE – Aritmetično logična enota

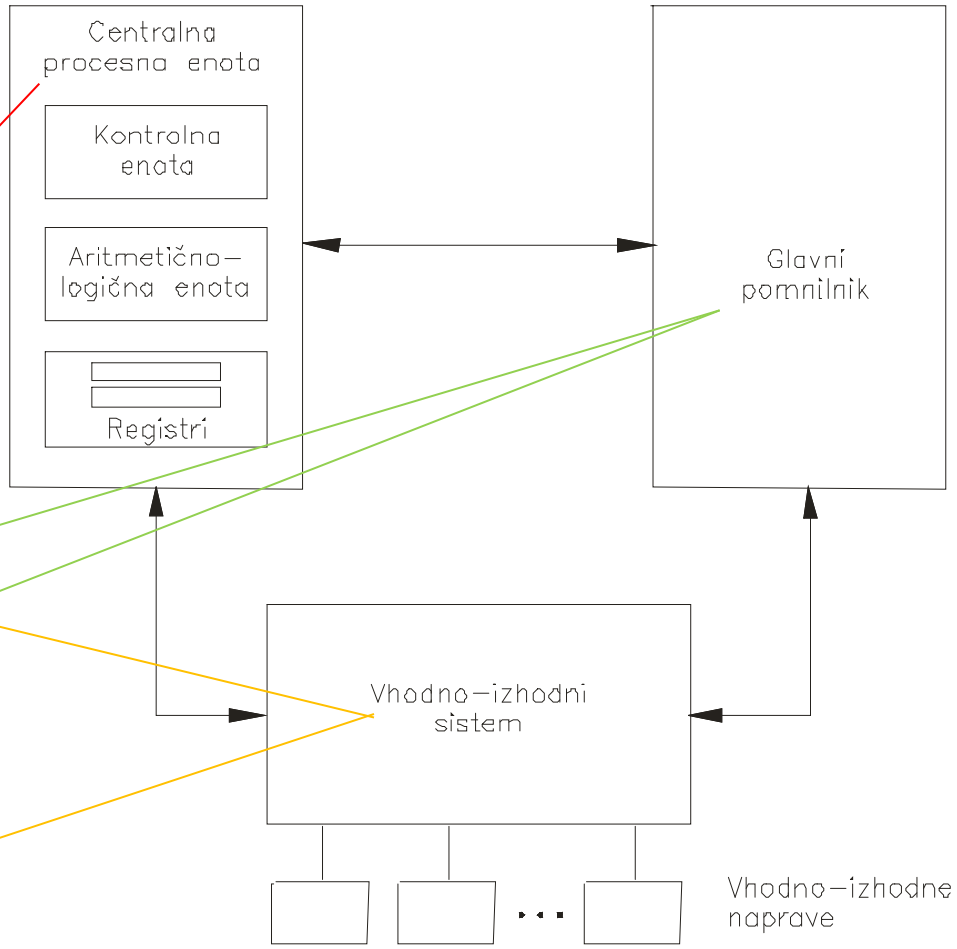
Osnovni model računalnika



Mikrokrmilniki

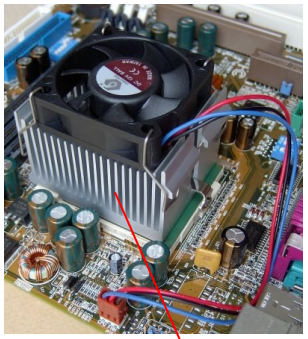
Osnovni model računalnika

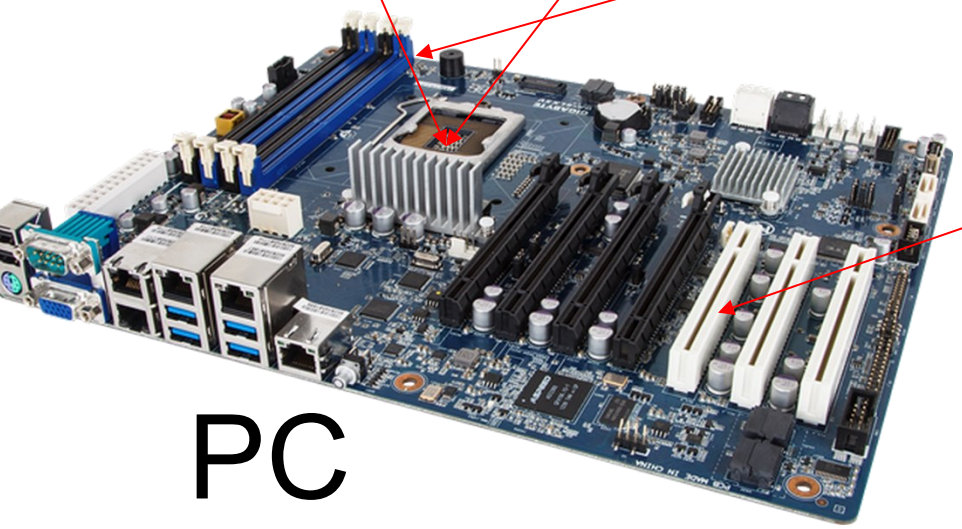




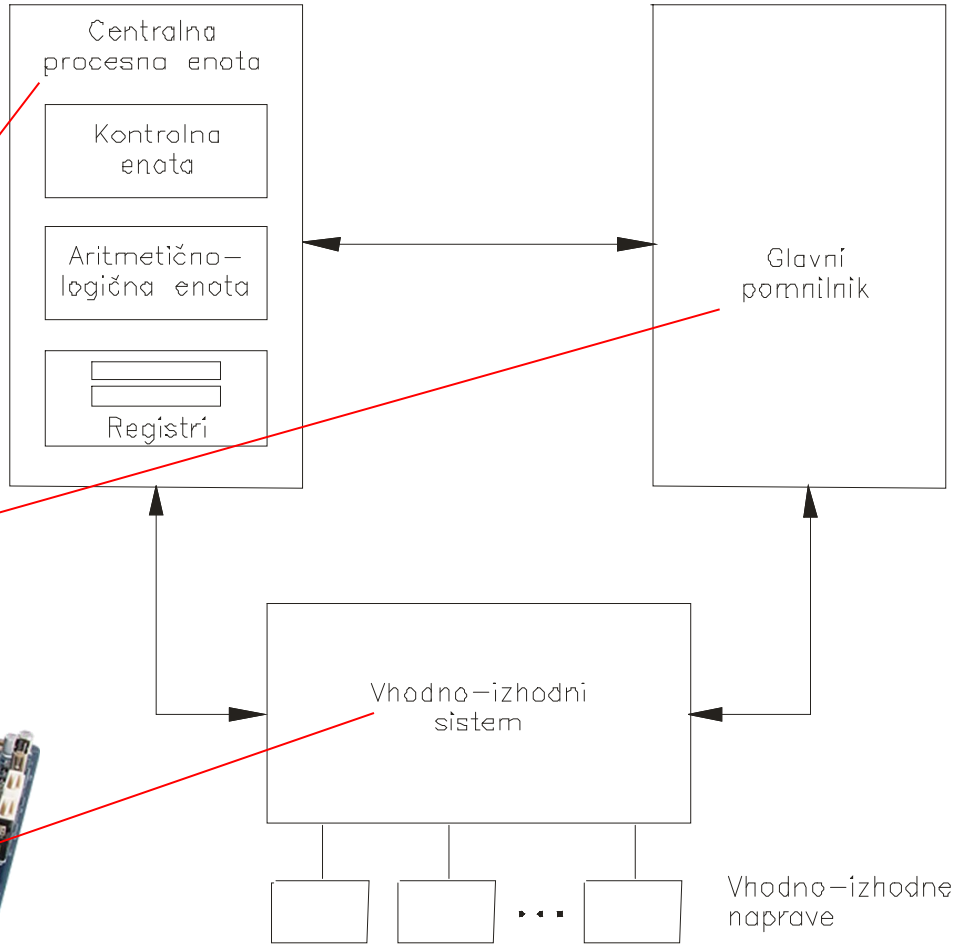
STM32H750-DK

Osnovni model računalnika





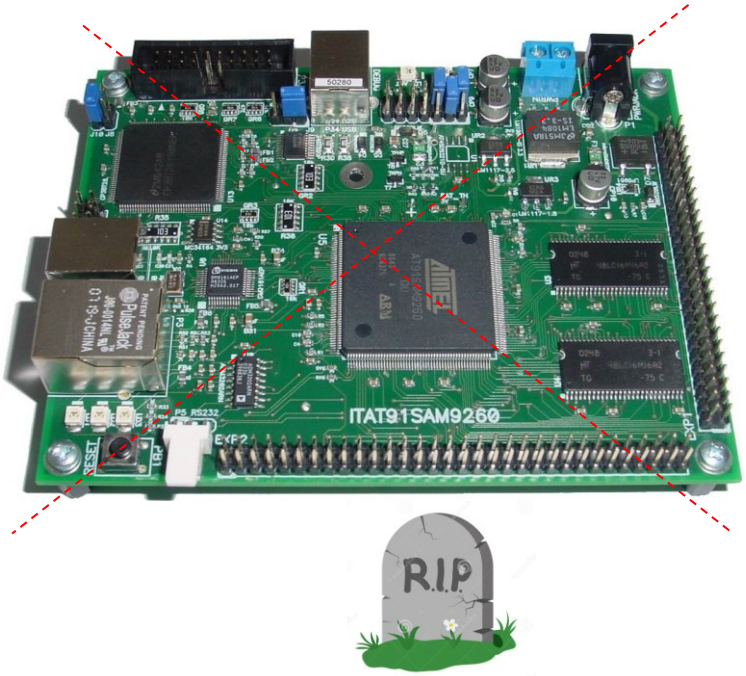
PC



Računalniška arhitektura RA

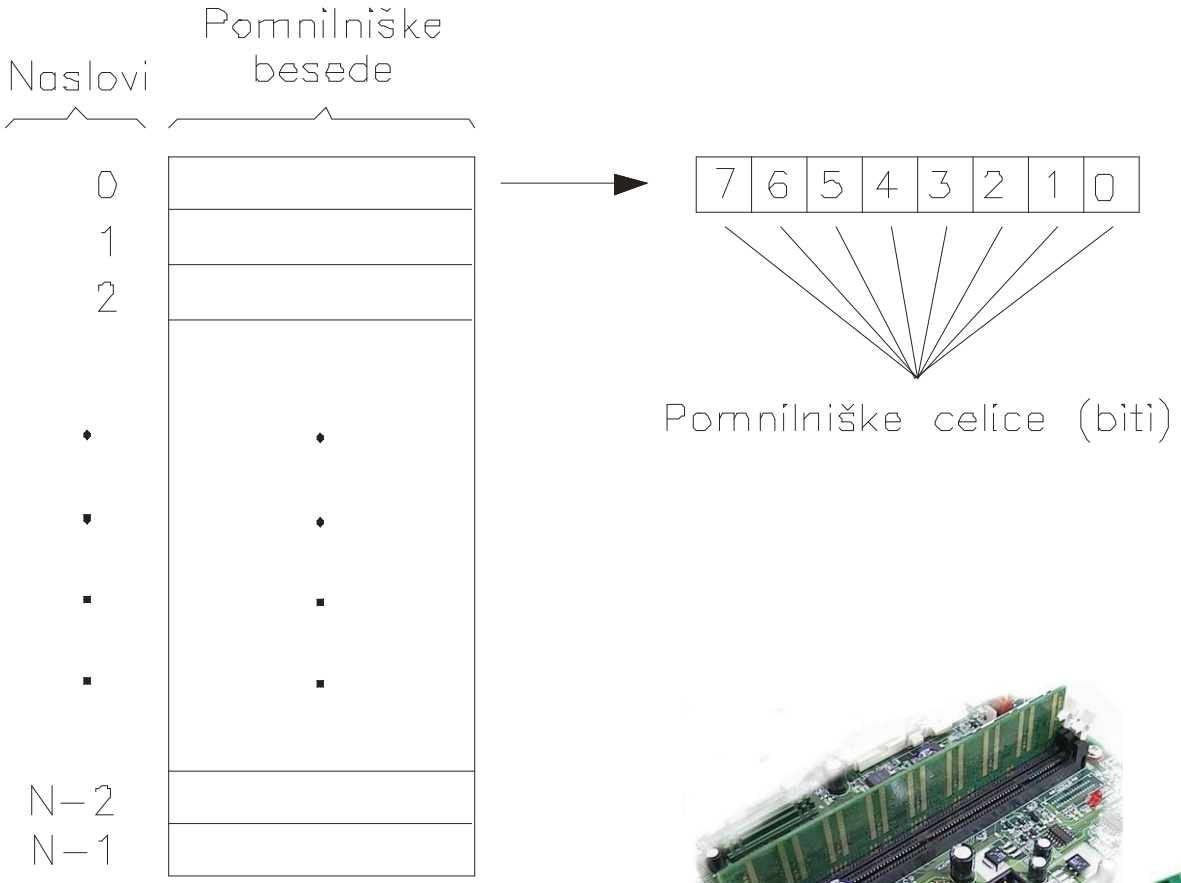
Računalnik STM32H750-DK

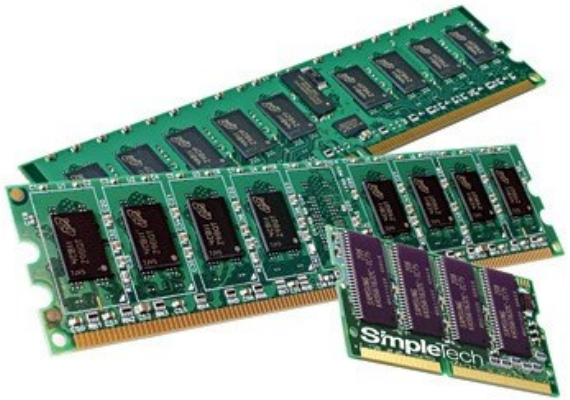
- Računalnik FRI-SMS
  - Mikrokrmilnik AT91SAM9260 iz družine mikrokrmilnikov ARM9

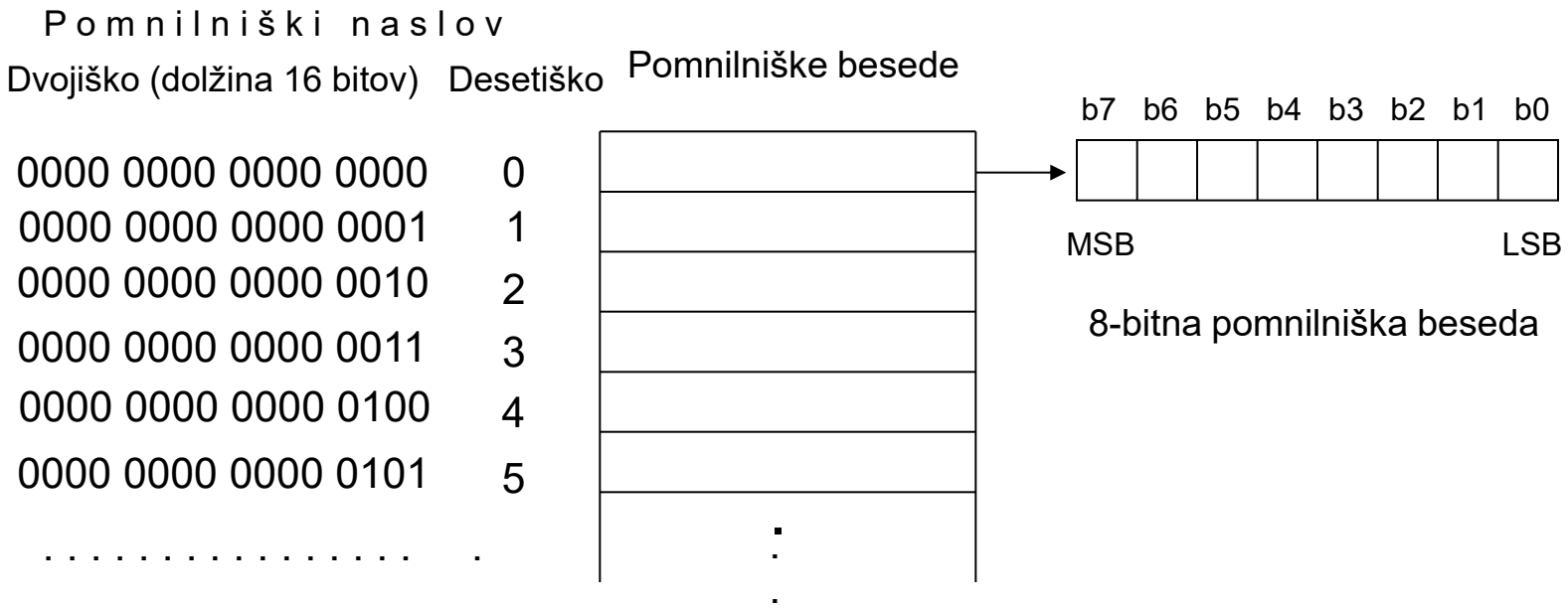


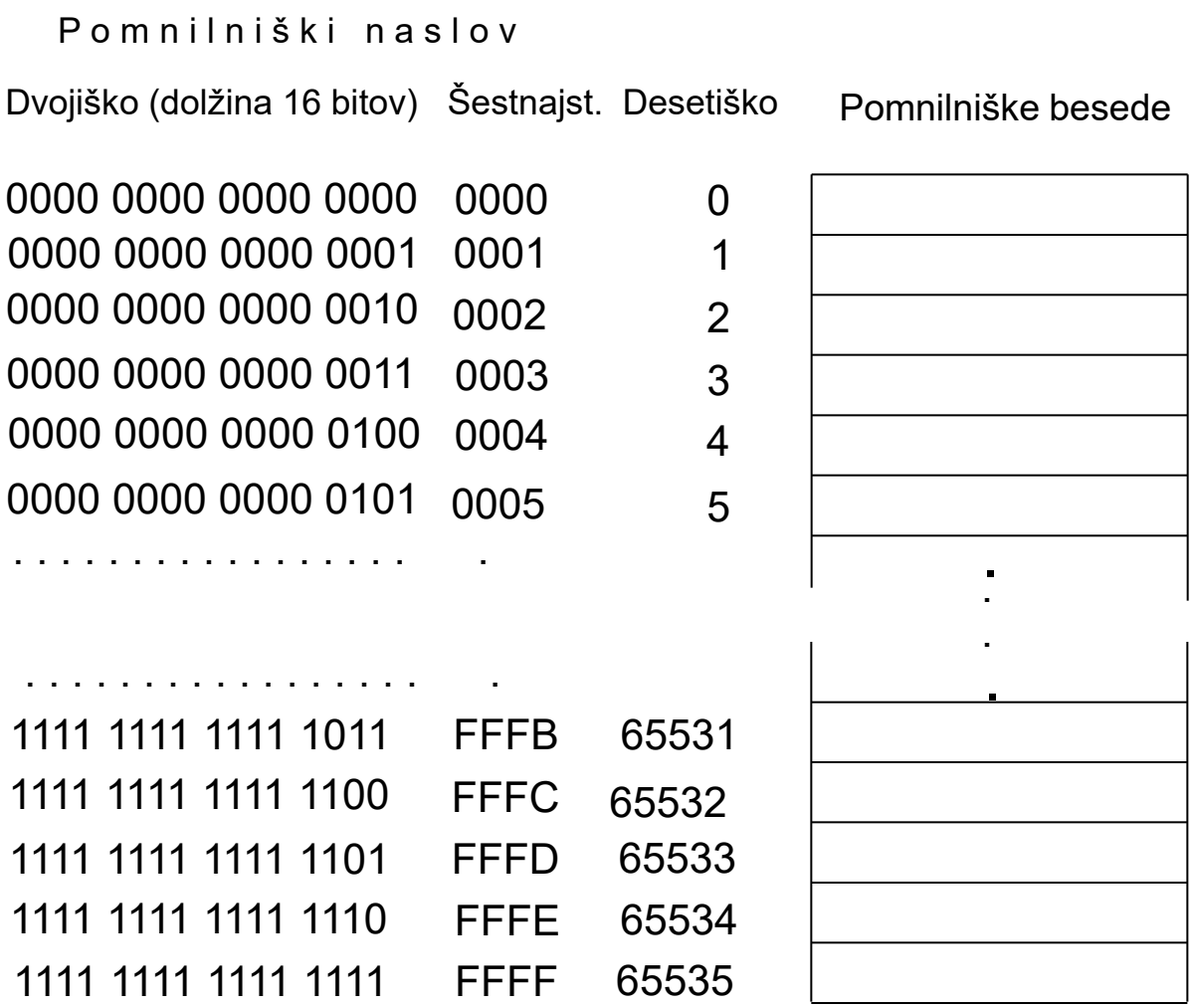
LAB 1.3 Pomnilnik

Kaj je pomnilnik ?



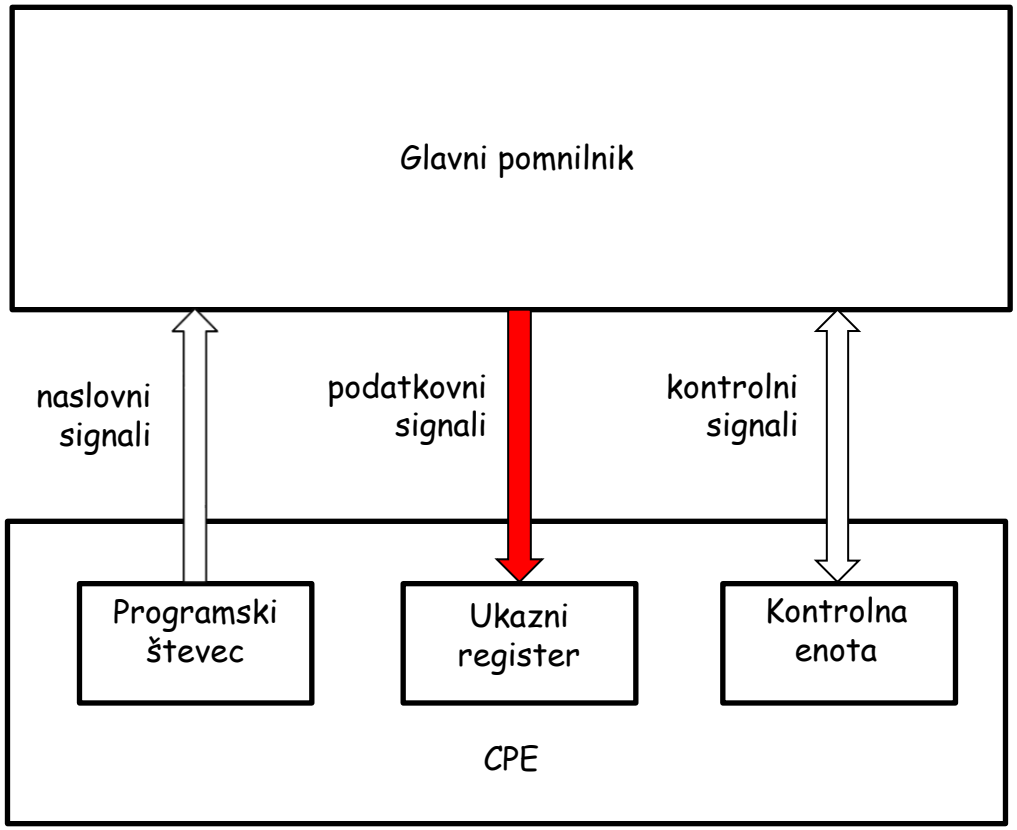



P o m n i l n i š k i n a s l o v
Dvojiško (dolžina 16 bitov) Desetiško
0000 0000 0000 0000 0
0000 0000 0000 0001 1
0000 0000 0000 0010 2
0000 0000 0000 0011 3
0000 0000 0000 0100 4
0000 0000 0000 0101 5
Pomnilniške besede
b7 b6 b5 b4 b3 b2 b1 b0
MSB
LSB
8-bitna pomnilniška beseda


P o m n i l n i š k i n a s l o v
Dvojiško (dolžina 16 bitov)
Šestnajst.
Desetiško
Pomnilniške besede
0000 0000 0000 0000 0000 0
0000 0000 0000 0001 0001 1
0000 0000 0000 0010 0002 2
0000 0000 0000 0011 0003 3
0000 0000 0000 0100 0004 4
0000 0000 0000 0101 0005 5
. . . . . .
. . . . . .
1111 1111 1111 1011 FFFB 65531
1111 1111 1111 1100 FFFC 65532
1111 1111 1111 1101 FFFD 65533
1111 1111 1111 1110 FFFE 65534
1111 1111 1111 1111 FFFF 65535

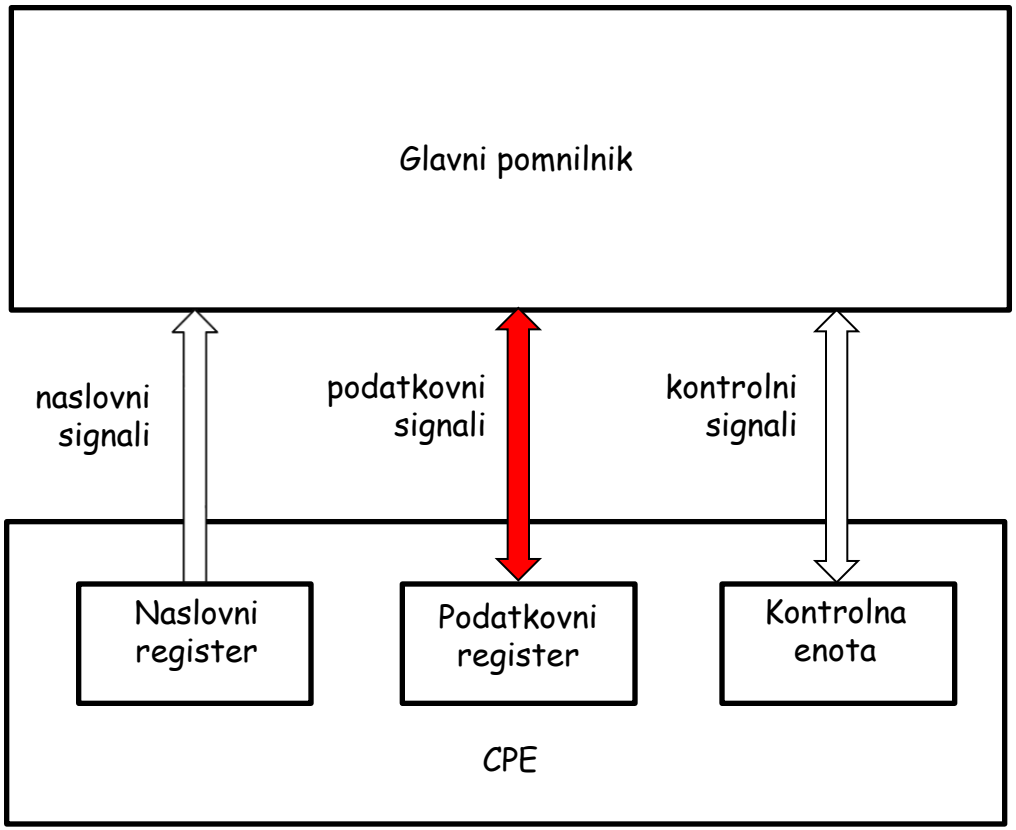
Kako CPE dostopa do glavnega pomnilnika?

Primer za ukaze:



Kako CPE dostopa do glavnega pomnilnika?

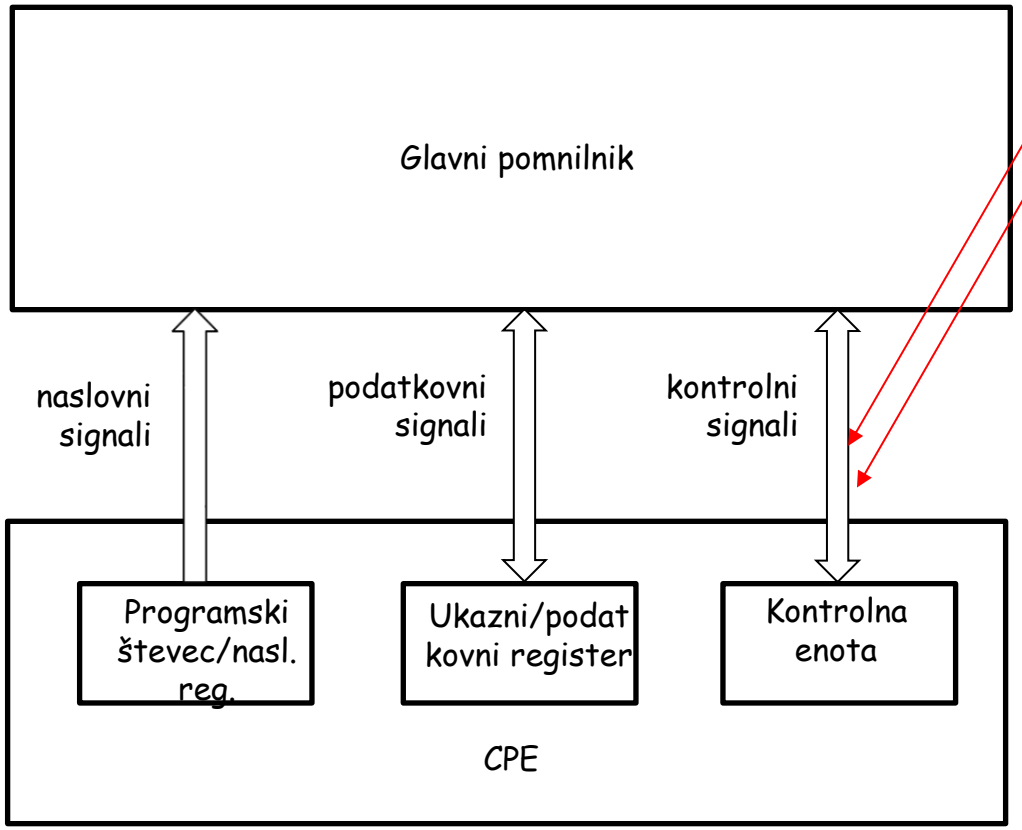
Primera za operande:



Povezava CPE <-> glavni pomnilnik

Vodilo = skupina povezav (naslovno, podatkovno, kontrolno, …)

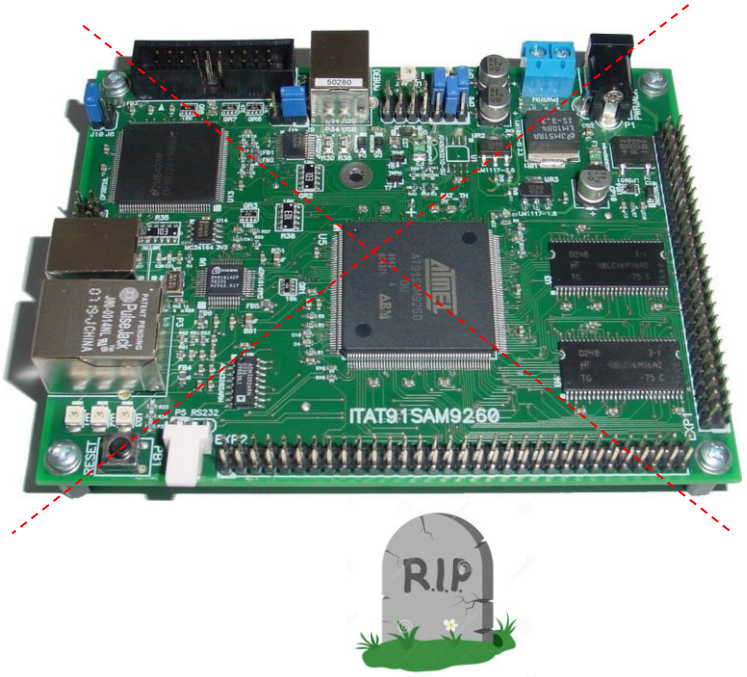
Linija = povezava
Signal = vsebina, ki se prenaša po povezavi (1bit)



Računalniška arhitektura RA

Računalnik STM32H750-DK

- Računalnik FRI-SMS
  - Mikrokrmilnik AT91SAM9260 iz družine mikrokrmilnikov ARM9

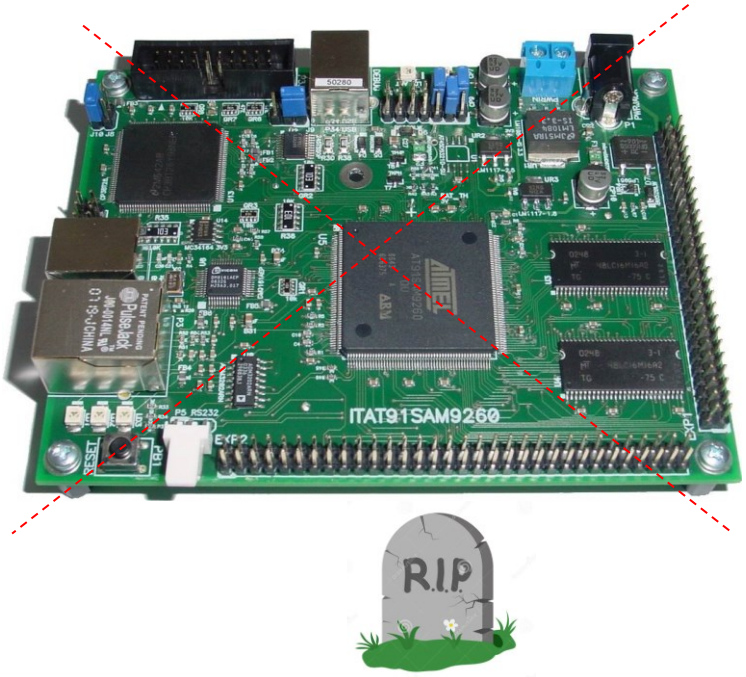


LAB 1.4 Številski sistemi (BIN,HEX) na hitro

Računalniška arhitektura RA

Računalnik STM32H750-DK

- Računalnik FRI-SMS
  - Mikrokrmilnik AT91SAM9260 iz družine mikrokrmilnikov ARM9

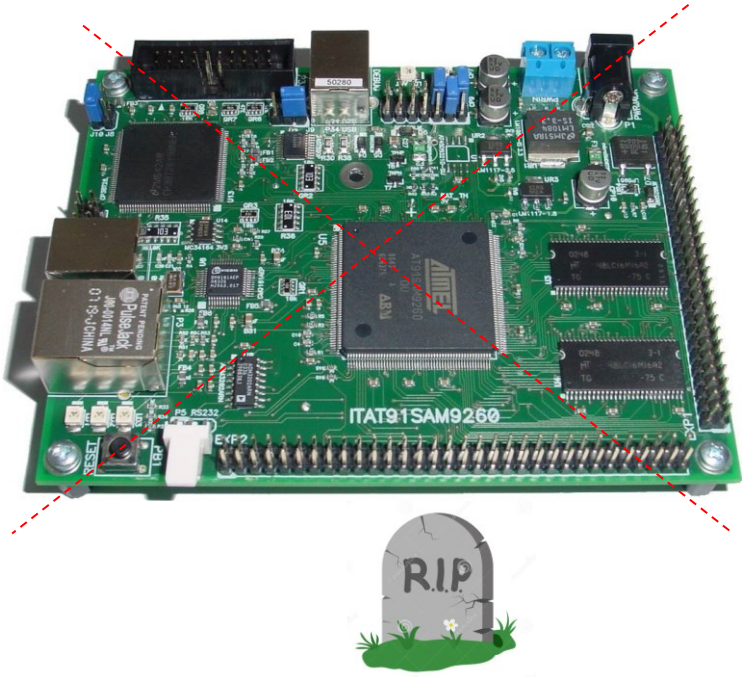


LAB 1.5 Pravilo tankega/debelega konca na hitro

Računalniška arhitektura RA

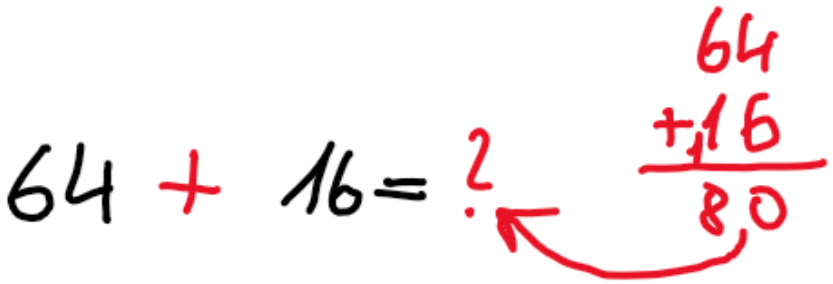
Računalnik STM32H750-DK

- Računalnik FRI-SMS
  - Mikrokrmilnik AT91SAM9260 iz družine mikrokrmilnikov ARM9



LAB 1.6 Seštevanje – človek, python, zbirnik

Človek (zgled: 64 + 16 = 80)



Python (zgled: REZ = STEV1 + STEV2)

Seštevanje spremenljivk v Pythonu.

http://goo.gl/YXQ5qN

Python 2.7

```
1  STEV1=0x40
2  STEV2=0x10
3  REZ = STEV1 + STEV2
4  print (" STEV1 = " + hex(STEV1) + "\n+STEV2 = " + hex(STE
```

Frames Objects

Global frame	
STEV1	64
STEV2	16
REZ	80

Print output (drag lower right corner to resize)

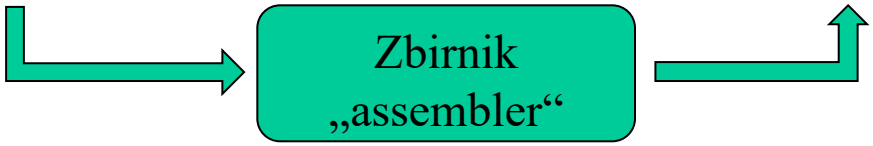
```
 STEV1 = 0x40
+STEV2 = 0x10
-------------
   REZ = 0x50
```

Zbirni jezik (zgled: rez=stev1+stev2)

Seštevanje spremenljivk v zbirniku ARM. Prenesite pripravljen projekt na e-učilnici.

Vrednosti spremenljivk so shranjene v pomnilniku. Operacije realiziramo s programom z naslednjimi ukazi:

Zbirni jezik	Opis ukaza	Strojni jezik
adr r0, stev1	R0 ← nasl. stev1	0xE24F0014
ldr r1, [r0]	R1 ← M[R0]	0xE5901000
adr r0, stev2	R0 ← nasl. stev2	0xE24F0018
ldr r2, [r0]	R2 ← M[R0]	0xE5902000
add r3, r2, r1	R3 ← R1 + R2	0xE0823001
adr r0, rez	R0 ← nasl. rez	0xE24F0020
str r3, [r0]	M[R0] ← R3	0xE5803000

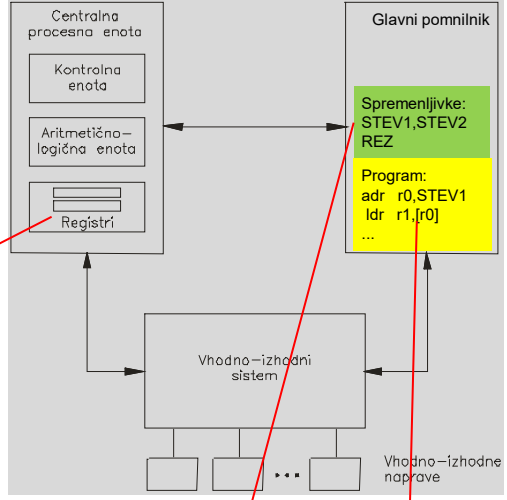


Ukaze izvajajte po korakih in opazujte vrednosti registrov in vrednosti spremenljivk v pomnilniku.

Praktično delo: vsota dveh števil

https://cpulator.01xz.net/?sys=arm&loadasm=share/s8zU3xx.s

Zbirni jezik	Opis ukaza	Strojni jezik
adr r0, stev1	R0 ← nasl. stev1	0xE24F0014
ldr r1, [r0]	R1 ← M[R0]	0xE5901000
adr r0, stev2	R0 ← nasl. stev2	0xE24F0018
ldr r2, [r0]	R2 ← M[R0]	0xE5902000
add r3, r2, r1	R3 ← R1 + R2	0xE0823001
adr r0, rez	R0 ← nasl. rez	0xE24F0020
str r3, [r0]	M[R0] ← R3	0xE5803000



Stopped | Step Into F2 | Step Over Ctrl-F2 | Step Out Shift-F2 | Continue F3 | Stop F4 | Restart Ctrl-R | Reload Ctrl-Shift-L | File | Help

Registers

Refresh

r0	00000000	
r1	00000000	
r2	00000000	
r3	00000000	
r4	00000000	
r5	00000000	
r6	00000000	
r7	00000000	
r8	00000000	
r9	00000000	
r10	00000000	
r11	00000000	
r12	00000000	
sp	00000000	
lr	00000000	
pc	0000002c	
cpsr	000001d3	NZCVI SVC
spsr	00000000	NZCVI ?

s0	00000000
s1	00000000
s2	00000000
s3	00000000
s4	00000000
s5	00000000
s6	00000000

Disassembly (Ctrl-D)

Go to address, label, or register: 00000000

Address	Opcode	Disassembly
		STEV1:
00000020	00000010	andeq r0, r0, r0,
		STEV2:
00000024	00000040	andeq r0, r0, r0,
		REZ:
00000028	00000000	andeq r0, r0, r0
		9 .align
		11 .global _start
		12 _start:
		14 adr r0,STEV1
		_start:
0000002c	e24f0014	adr r0, 0x20 (0
00000030	e5901000	15 ldr r1, [r0]
		17 adr r0,STEV2
00000034	e24f0018	adr r0, 0x24 (0
00000038	e5902000	18 ldr r2, [r0]
0000003c	e0813002	20 add r3, r1, r2
		22 adr r0,REZ
00000040	e24f0020	adr r0, 0x28 (0
00000044	e5803000	23 str r3, [r0]
		26 end: b er
		end:
00000048	eafffffe	b 0x48 (0x48: enc

Memory (Ctrl-M)

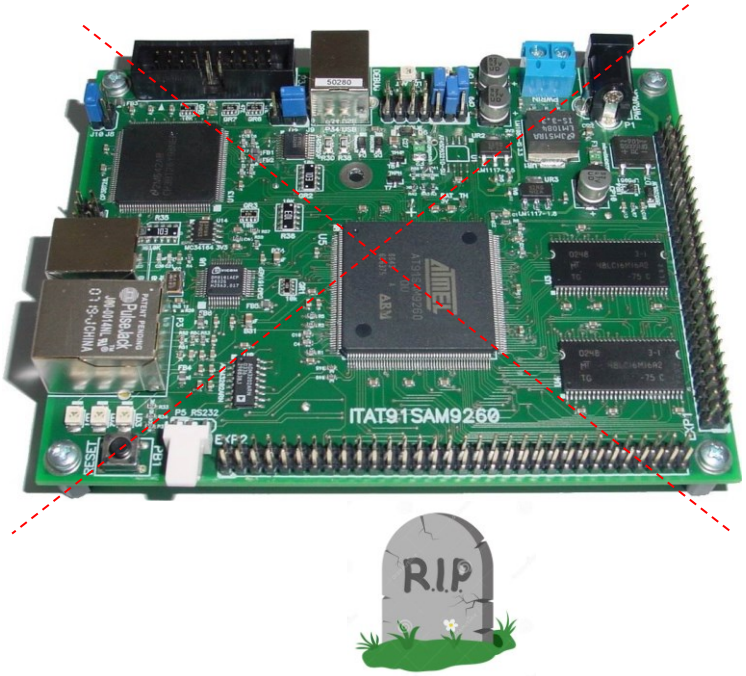
Go to address, label, or register:

Address	Memory contents and ASCII			
00000000	00000000	00000000	00000000	00000000
00000010	00000000	00000000	00000000	00000000
00000020	00000010	00000040	00000000	e24f0014
00000030	e5901000	e24f0018	e5902000	e0813002
00000040	e24f0020	e5803000	eafffffe	00000000
00000050	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000060	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000070	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000080	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000090	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
000000a0	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
000000b0	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
000000c0	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
000000d0	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
000000e0	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
000000f0	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000100	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000110	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000120	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000130	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000140	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000150	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa
00000160	aaaaaaaa	aaaaaaaa	aaaaaaaa	aaaaaaaa

Računalniška arhitektura RA

Računalnik STM32H750-DK

- Računalnik FRI-SMS
  - Mikrokrmilnik AT91SAM9260 iz družine mikrokrmilnikov ARM9



LAB 1.7 Predloge za zapiske

Python (zgled: REZ = STEV1 + STEV2)

Frames Objects

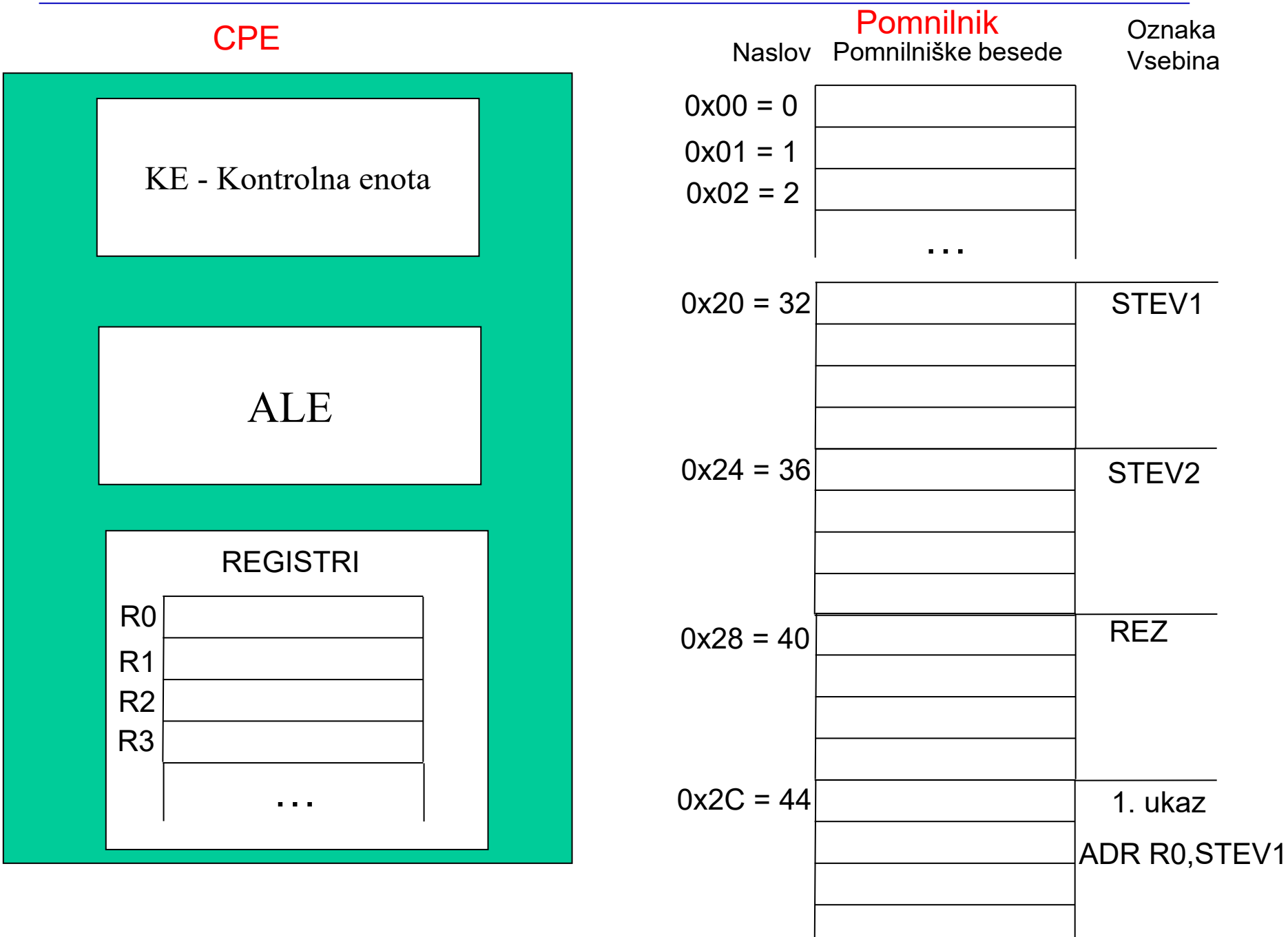
Global frame	
STEV1	64
STEV2	16
REZ	80

Python 2.7

```
1  STEV1=0x40
2  STEV2=0x10
3  REZ = STEV1 + STEV2
4  print (" STEV1 = " + hex(STEV1) + "\n+STEV2 = " + hex(STE
```

http://goo.gl/YXQ5qN

Zgled: izvedba programa za seštevanje dveh števil



Zgled: izvedba programa za seštevanje dveh števil

UKAZI

	Strojni jezik	Zbirni jezik	Opis ukaza	Komentar
1.	0xE24F0014	adr r0, stev1	R0 ← nasl. stev1	
2.	0xE5901000	ldr r1, [r0]	R1 ← M[R0]	
3.	0xE24F0018	adr r0, stev2	R0 ← nasl. stev2	
4.	0xE5902000	ldr r2, [r0]	R2 ← M[R0]	
5.	0xE0823001	add r3, r2, r1	R3 ← R1 + R2	
6.	0xE24F0020	adr r0, rez	R0 ← nasl. rez	
7.	0xE5803000	str r3, [r0]	M[R0] ← R3	

Pravilo tankega in debelega konca / Big vs. Little Endian

MSB LSB

0 x AA BB CC DD

Debeli konec
Big Endian

Tanki konec
Little Endian