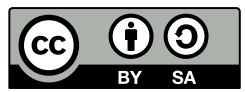
Mašinos kodas. Asembleris

Saulius Gražulis

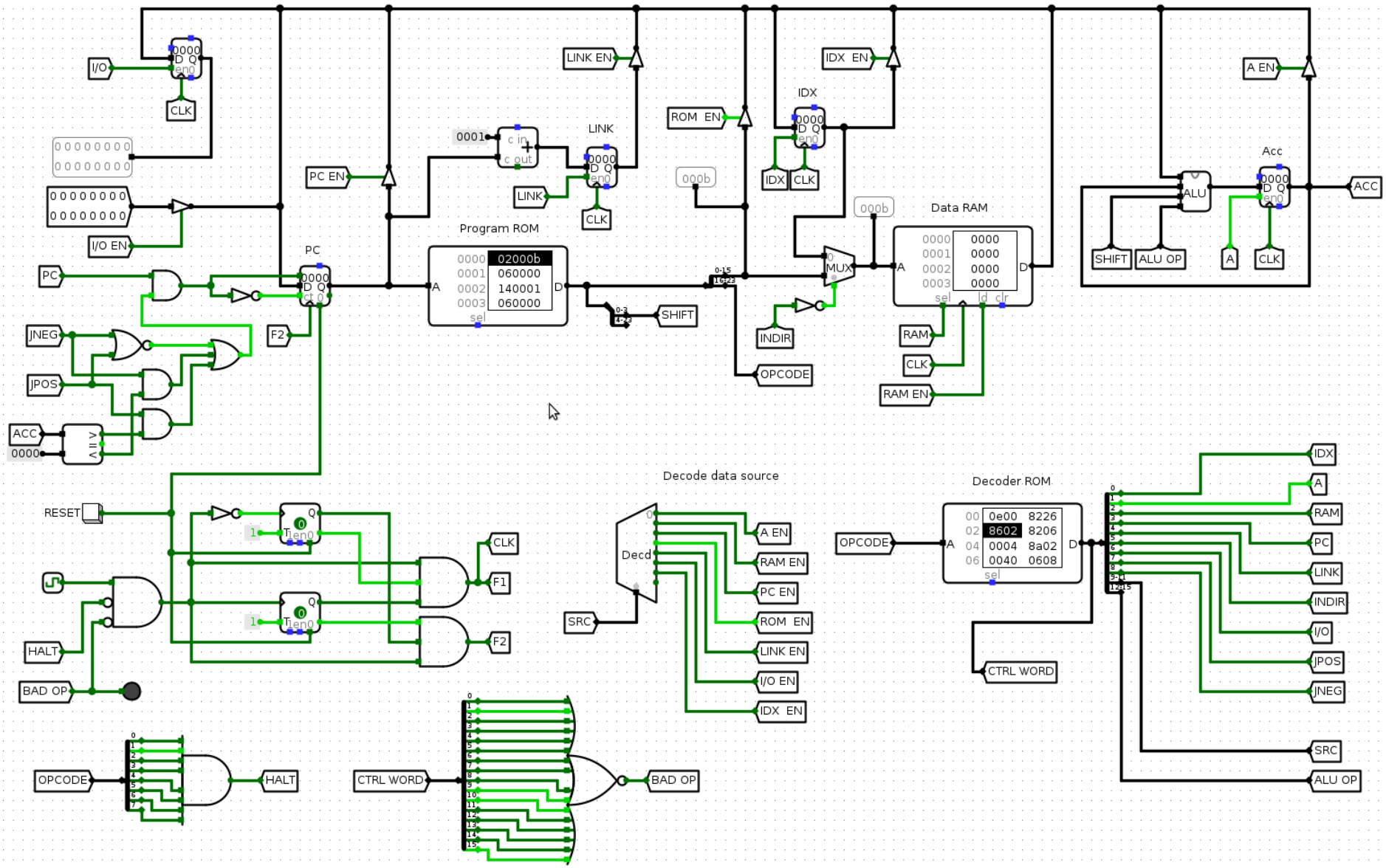
Vilnius, 2020

Vilniaus universitetas, Matematikos ir informatikos fakultetas
Informatikos institutas

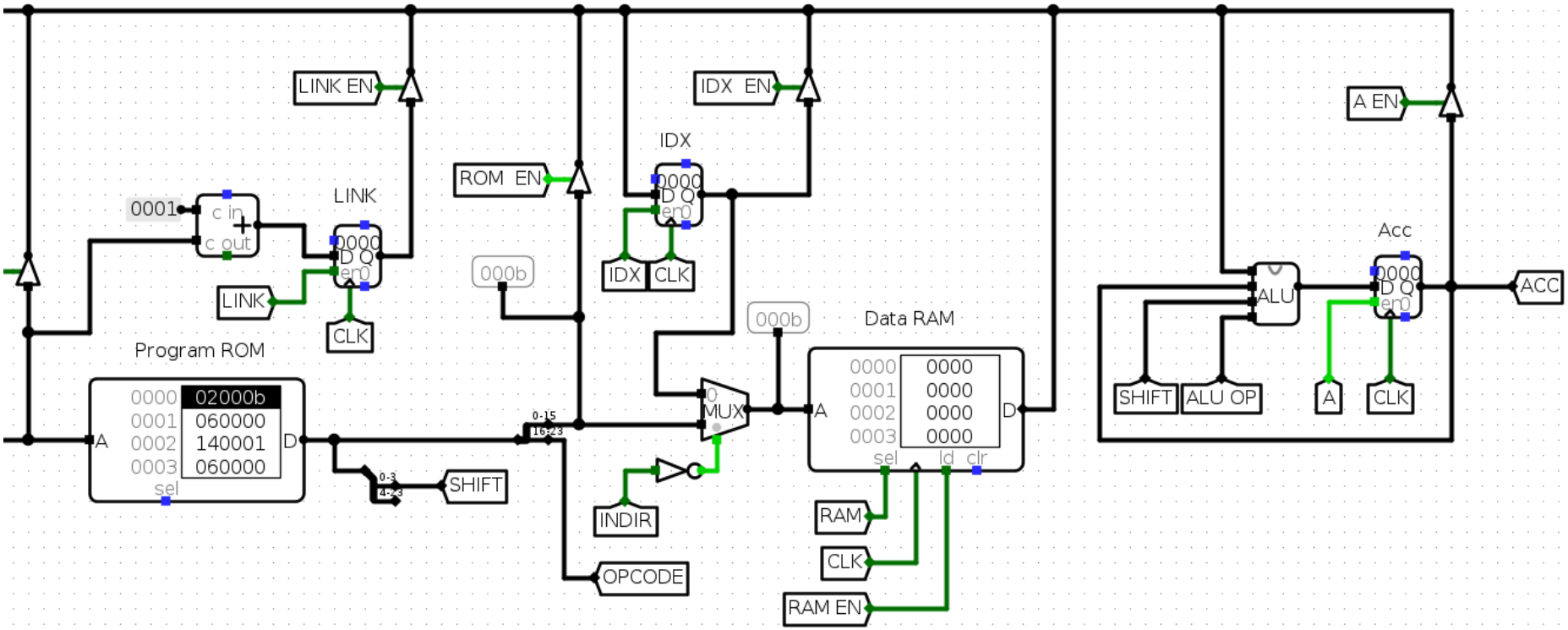
Šį skaidrių rinkinį galima kopijuoti, kaip nurodyta Creative Commons Attribution-ShareAlike 4.0 International licenzijoje



Paprastas Harvardo architektūros procesorius



Komandų atmintis



Paprastas algoritmas

Fibonači skaičiai

```c
int main()
{
    /* last two members of the
       sequence: */
    int a, b;
    int i;
    /* number of sequence members
       to produce: */
    int N = 21;

    a = b = 1;

    for( i = 1; i <= N; i++ ) {
        int t = b;
        b += a;
        printf( "%d\t%d\n", i, b );
        a = t;
    }

    return 0;
}
```

Žingsnis 0. Nustatyti $a \leftarrow 1$, $b \leftarrow 1$. (a ir b yra du paskutiniai sekos skaičiai)

Žingsnis i. visiems $1 \leq i \leq N$:

i. laikinai išsaugoti b: $t \leftarrow b$;
ii. nustatyti $b \leftarrow a + b$;
iii. išvesti b;
iv. nustatyti $a \leftarrow t$;

Žingsnis $N + 1$. Pabaiga

Mašinos kodas

```c
int main()
{
    /* last two members of the
       sequence: */
    int a, b;
    int i;
    /* number of sequence members
       to produce: */
    int N = 21;

    a = b = 1;

    for( i = 1; i <= N; i++ ) {
        int t = b;
        b += a;
        printf( "%d\t%d\n", i, b );
        a = t;
    }

    return 0;
}
```

```
v2.0 raw
020001
040000
040001
020000
040002
030001
0B0000
060000
040001
0C0000
040000
030002
190001
040002
1A0015
0A0005
FFFFFF
```

Asemblerio pavyzdys

```c
int main()
{
    /* last two members of the
       sequence: */
    int a, b;
    int i;
    /* number of sequence members
       to produce: */
    int N = 21;

    a = b = 1;

    for( i = 1; i <= N; i++ ) {
        int t = b;
        b += a;
        printf( "%d\t%d\n", i, b );
        a = t;
    }

    return 0;
}
```

```
        ;; Fibonacci numbers
A:      DS   1
B:      DS   1
I:      DS   1
N:      EQU  21
        ;;
        LDC  1
        ST   A
        ST   B
        LDC  0
        ST   I
LOOP:   LD   B
        ADD  A
        OUT  0
        ST   B
        SUB  A
        ST   A
        LD   I
        ADDC 1
        ST   I
        SUBC N
        JNEG LOOP
END:    HALT
```

Komandų mnemonikos

Adresas	Mašinos kodas	Mnemonika		Reikšmė
0000	020001	LDC	1	Įkelti konstantą 1 į akumuliatorių
0001	040000	ST	0	Išsaugoti akumuliatorių adresu 0
0002	040001	ST	1	
0003	020000	LDC	0	
0004	040002	ST	2	
0005	030001	LD	1	Įkelti [0001][1] į akumuliatorių
0006	0B0000	ADD	0	Pridėti prie akumuliatoriaus [0000]
...				
000D	1A0015	SUBC	15H	Atimti 21 iš akumuliatoriaus
000E	0A0005	JNEG	0005	Pereiti į prog. adresą 5, jei akum. < 0
000F	FFFFFF	HALT		Sustabdyti procesorių

[1] [0001] – duomenų atminties celės, kurios adresas yra 0001, turinys.

Komandų operandai

```
N   EQU   21
A   EQU   0
B   EQU   1
I   EQU   2
```

Adresas	Mašinos kodas	Mnemonika		Reikšmė
0000	020001	LDC	1	Įkelti konstantą 1 į akumuliatorių
0001	040000	ST	A	Išsaugoti akumuliatorių adresu A
0002	040001	ST	B	
0003	020000	LDC	0	
0004	040002	ST	I	
0005	030001	LD	B	Įkelti [B][2] į akumuliatorių
0006	0B0000	ADD	A	Pridėti prie akumuliatoriaus [A]
...				
000D	1A0015	SUBC	N	Atimti N (= 21) iš akumuliatoriaus
000E	0A0005	JNEG	0005	Pereiti į 0005, jei akumuliatorius < 0

[2] [B] – duomenų atminties celės, kuri talpina kintamąjį B, turinys.

Komandų operandai

```
N   EQU   21
A   DS    1
B   DS    1
I   DS    1
```

Adresas	Mašinos kodas	Mnemonika		Reikšmė
0000	020001	LDC	1	Įkelti konstantą 1 į akumuliatorių
0001	040000	ST	A	Išsaugoti akumuliatorių adresu A
0002	040001	ST	B	
0003	020000	LDC	0	
0004	040002	ST	I	
0005	030001	LD	B	Įkelti [B][2] į akumuliatorių
0006	0B0000	ADD	A	Pridėti prie akumuliatoriaus [A]
...				
000D	1A0015	SUBC	N	Atimti N (= 21) iš akumuliatoriaus
000E	0A0005	JNEG	0005	Pereiti į 0005, jei akumuliatorius < 0

[2] [B] – duomenų atminties celės, kuri talpina kintamąjį B, turinys.

Žymės

```
      ORG   100
N  :  EQU   21
A  :  DS    1
B  :  DS    1
I  :  DS    1
```

Adresas	Mašinos kodas	Mnemonika		Reikšmė
	020001	LDC	1	Įkelti konstantą 1 į akumuliatorių
	040000	ST	A	Išsaugoti akumuliatorių adresu A
	040001	ST	B	
	020000	LDC	0	
	040002	ST	I	
LOOP:	030001	LD	B	Įkelti atminties celę B į akumuliatorių
	0B0000	ADD	A	Pridėti prie akumuliatoriaus A turinį
...				
	1A0015	SUBC	N	Atimti N (= 21) iš akumuliatoriaus
	0A0005	JNEG	LOOP	Pereiti į LOOP, jei akumuliatorius < 0

Asemblerio programa

Žymė	Operacija	Operandai	Komentarai
	ORG	100	
N:	EQU	21	; Kiek sekos narių reikia suskaičiuoti
A:	DS	1	
B:	DS	1	
I:	DS	1	
	LDC	1	; Įkelti 1 į akumuliatorių
	ST	A	; Išsaugoti akumuliatorių celėje A
	ST	B	
	LDC	0	
	ST	I	
LOOP:	LD	B	; Įkelti B turinį į akumuliatorių
	ADD	A	; Pridėti A turinį prie akumuliatoriaus
...			
	SUBC	N	; Atimti 21 iš akumuliatoriaus
	JNEG	LOOP	; Pakartoti ciklą, jei akumuliatorius < 0

Adresavimo režimai

Režimas	**Veiksmai**	**Aprašymas**
Betarpiškas	A $\leftarrow$ operand	Naudojama reikšmė, nurodyta komandoje
Tiesioginis	A $\leftarrow$ [operand]	Reikšmės *adresas* nurodytas komandoje
Registrinis	A $\leftarrow$ IDX	Naudojama reikšmė yra registre
Netiesioginis	A $\leftarrow$ [IDX]	Reikšmės *adresas* yra registre
Indeksinis	A $\leftarrow$ operand[IDX]	Reikšmės adresas yra indekso registro reikšmė + operandas

Programuotojui matomas procesoriaus modelis

Programuotojo požiūriu, apie kompiuterį ar procesorių reikia nurodyti:

- **Procesoriaus registrus**, matomus programuotojui
- Procesoriaus **komandų sistemą (ISA)**
- Procesoriaus **adresavimo režimus**
- Procesoriaus **atminties organizaciją**
- Procesoriaus **įvestį ir išvestį**

(Gutierrez-Osuna 2000)

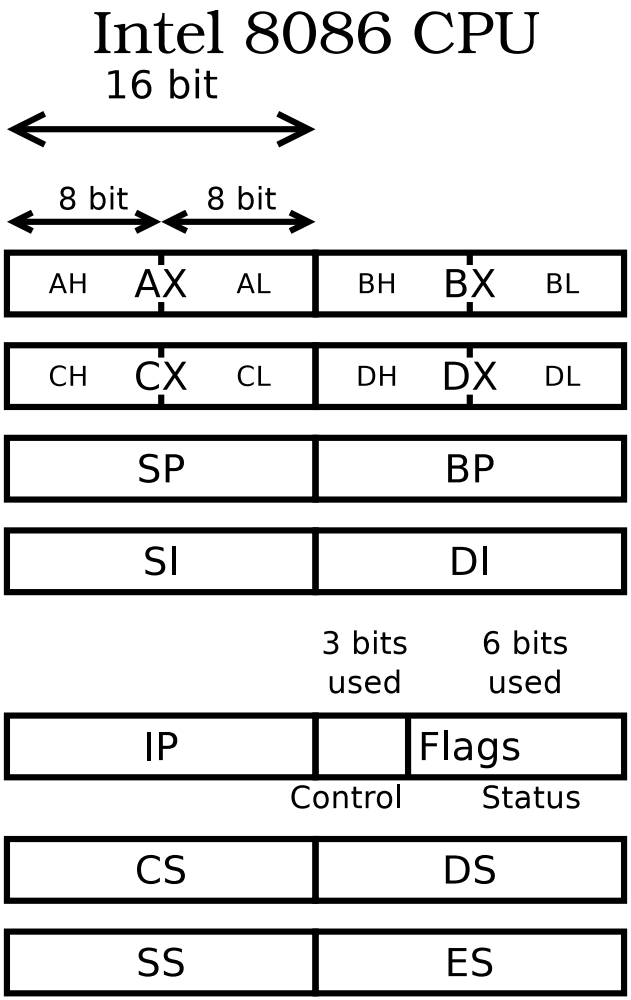
Harvardo architektūros 16 bitų procesorius

- Registrai: `A`, `PC`, `IDX`
- Komandų formatas:

OPC (8 bitai)	ADDR/IMM (16 bitų)

- Žodžio plotis:
  - duomenų žodžio – 16 bitų
  - komandų žodžio – 24 (8 + 16) bitai
- Adreso plotis: 16 bitų
- Adresuojamas: žodis
- Adresų erdvė: vientisa
- Komandų sistema: akumuliatorinė architektūra; `LD`, `LDC`, `ST`, `ADD`, `JNZ`, `JNEG`, `...`
- Įvestis/išvestis: prievadai atskiroje adresų erdvėje; pertraukimų nėra; vienas įvesties ir vienas išvesties prievadas

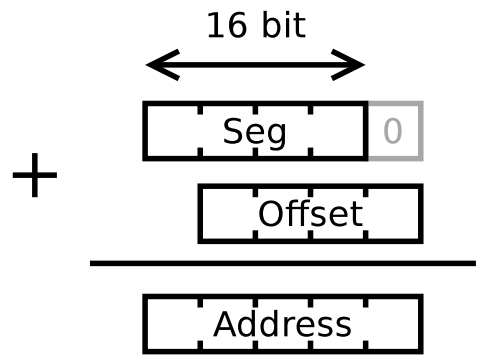
Registrai



(Intel 1979)

Segmentinė adresacija

Segmento adresas pastumiamas 4-iais bitais ir sudedamas su postūmiu:



Apibendrinimas

- Mašinos kalba yra dvejetainiai skaičiai, paprastai užrašomi šešioliktainėje sistemoje
- Komandų mnemonikos žymiai palengvina operacijų aprašymą ir įsiminimą
- Kad palengvinti programavimą mašinos kalba, naudojami asembleriai
- Pagrindinės asemblerio sąvokos: žymės, operacijos, operandai, pseudoinstrukcijos, komentarai
- Kompiuterio ar procesoriaus architektūrą galime aprašyti, nurodydami keletą charakteristikų, matomų programuotojui

Šaltiniai

Gutierrez-Osuna, Ricardo (2000). *Lecture 2: MC68000 architecture*. URL: http://courses.cs.tamu.edu/rgutier/ceg411_f01/l2.pdf.

Intel (Oct. 1979). *Intel 8086 Family User's Manual*. Intel Corporation. URL: https://edge.edx.org/c4x/BITSPilani/EEE231/asset/8086_family_Users_Manual_1_.pdf.