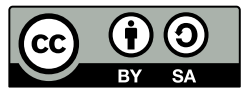
Kompiuterių architektūra. Įvadas

Saulius Gražulis

Vilnius, 2020

Vilniaus universitetas, Matematikos ir informatikos fakultetas
Informatikos institutas



Kurso organizacija

https://emokymai.vu.lt/course/view.php?id=15112

Vertinimo strategija	Svoris %	Atsiskaitymo laikas	Vertinimo kriterijai
Apklausos prieš paskaitas	10	10 min. prieš paskaitą ar praktikos darbą.	4-klausimų apklausa iš keleto praeitų paskaitų temų (1 ir antro lygio klausimai pagal Blūmo (Bloom) klasifikaciją) naudojant elektronines mokymo priemones (Moodle, Open edX ar pan).
Tarpinis kontrolinis	15	semestro vidurys	Apytikriai 30 klausimų apklausa iš visų praeitų dalykų (1 iki 9 lygio klausimai pagal Bloom klasifikaciją) naudojant elektronines mokymo priemones (Moodle, Open edX ar pan).
Pratybų (laboratoriniai) darbai	50	pagal pratybų vadovų paskelbtą grafiką	Praktikos darbai atliekami pagal pratybų vadovų nurodytas užduotis ir atsiskaitomi bei vertinami pagal kiekvieno pratybų vadovo nurodytą grafiką bei kriterijus.
Savarankiškas konkrečios architektūros studijavimas	10	semestro pabaiga	Studentai pateikia maždaug 4 psl. (A4, 9pt) techninį aprašymą apie jiems paskirtą ir savarankiškai išnagrinėtą kompiuterių architektūrą, arba 5 min. žodinį pristatymą su skaidrėmis (PDF formatu). Žodinis pristatymas galimas studentams, pasiekusiems gerų rezultatų kurso metu ir gali būti užskaitomas kaip egzaminas (sąlygas žr. punkte „Egzaminas“).
Egzaminas	15	semestro pabaiga	Apytikriai 30 klausimų apklausa iš viso kurso dalykų (1 iki 9 lygio klausimai pagal Bloom klasifikaciją)

Reikalavimai egzaminui

Kad studentai būtų prileisti prie egzamino, jie turi:

1. Atlikti bent vieną praktikos darbą ir surinkti teigiamą (didesnį už nulį) pratybų balą;
2. Sukaupti suminį balą už darbą per semestrą (iš praktikos darbų, tarpinio kontrolinio, darbo paskaitose, galimai kitų dėstytojų paskirtų užduočių), kad, parašius egzamino kontrolinį, būtų įmanoma pasiekti patenkinamo pažymio balą (t. y. balą, užtikrinantį bent pažymį „5“);
3. Ypač gerai pasirodžiusiems semestro metu studentams gali būti leidžiama laikyti išankstinį egzaminą, padarant žodinį pranešimą dėstytojo paskirtu laiku užsiėmimų metu. Norint laikyti išankstinį egzaminą, būtina:
    1. Surinkti bent bent 60% semestro metu galimų gauti teorijos balų (pvz. bent 150 iš dabar prieinamų 250 balų);
    2. Atsiskaityti laiku visus praktikos darbus;
    3. Turėti teigiamas praktikos vadovo rekomendacijas.

Jei norinčių laikyti egzaminą iš anksto studentų yra daugiau, negu leidžia turimas užsiėmimų laikas, pirmenybė suteikiama studentams, surinkusiems daugiau balų.

Reikalavimai egzaminui

Dalyko aprašas

			Egzamino kontrolinis yra būtinas visiems nepriklausomai nuo surinkto balų skaičiaus, išskyrus tuos studentus, kurie gavo leidimą laikyti egzaminą iš anksto ir kuriems užskaitytas kaip egzaminas žodinis savarankiško darbo pristatymas. Studentams, neatvykusiems į egzaminą, žiniaraštyje bus žymima „neatvyko“. Egzamine būtina surinkti bent 50% egzamino balo.
Viso	100		Galutinis pažymys gaunamas susumavus už visas veiklas surinktus balus, padalinant sumą iš 100 ir su apvalinant iki *didesnio* sveiko skaičiaus (t.y. 0.001 apvalinama link 1; 9.1 apvalinama iki 10).

Kurso struktūra

- Pirma pusė: supratimas, kaip veikia kompiuteris ir procesorius loginių schemų lygyje; modeliavimas Logisim programa;
- Antra pusė: pažintis su keliomis paplitusiomis ir savo sukurtomis architektūromis ir jų programavimas asemblerio kalba;

Nuotolinių konferencijų taisyklės

- Mikrofoną įsijungia tik tas, kas kalba; visų kitų mikrofonai turi būti **išjungti**;
- Kalbantysis **turi** įsijungti vaizdo kamerą (VU taisyklės!);
- Kai dėstytojas užduoda klausimą, studentai **turi** atsakyti, pakeldami ranką ir, gavę žodį, įjungę mikrofoną, arba parašydami forume (VU taisyklės!);
- Jei norite paklausti klausimą, naudokite funkciją „pakelti ranką“ ir/arba rašykite forume;
- Paskaitų vaizdo ir garso įrašai saugomi autorių teisių; juos viešinti už VU ribų ir be VU/dėstytojo sutikimo **negalima**;
- Jei nutrūksta ryšys, **nepalikite** paskaitos – dėstytojas įjungs alternatyvų kanalą per kelias minutes;

Kodėl reikia studijuoti kompiuterių architektūrą?

> *Most computer users have an incorrect, but useful, cognitive metaphor for computers in which the user says (or types or clicks) something and a mystical, almost intelligent or magical, behavior happens.*
>
> Charles W. Kann (2016)

- Tik suprasdami kompiuterių architektūrą, galime ją efektyviai programuoti
- Praplečia mūsų akiratį – galų gale suprasime, kaip **tai** veikia
- Paprasčiausiai įdomu :)

Kas yra kompiuteris?

- Kompiuteris yra **skaitmeninis**:

Emural screenshot

- Kompiuteris yra **automatinis**:

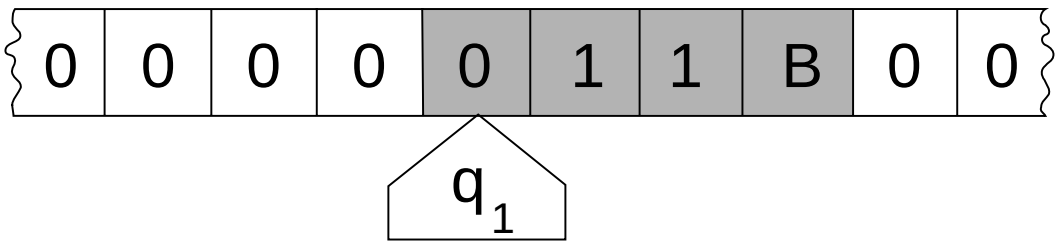
Wikipedia: Skylab. Image by NASA, public domain.

Kas yra kompiuteris?

- Kompiuterį galima **perprogramuoti**:

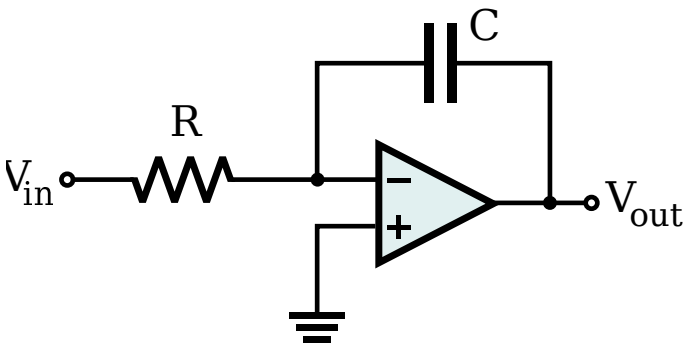
```
#! /usr/bin/perl

use strict;
use warnings;

my $selected_line;

while( <> ) {
    $selected_line = $_ if rand() < 1/$.;
}

print $selected_line;
```

- Kompiuteris gali išspręsti **bet kokį matematiškai** įmanomą uždavinį (Turing 1937; Wikipedia 2020):



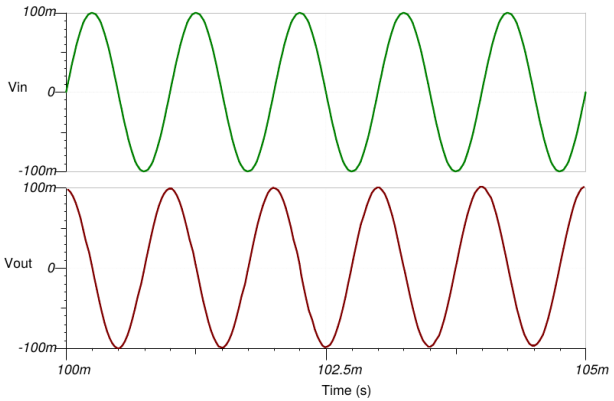
Nynexman4464 [CC-BY-SA 3.0]

Pastaba: Analoginės skaičiavimo mašinos



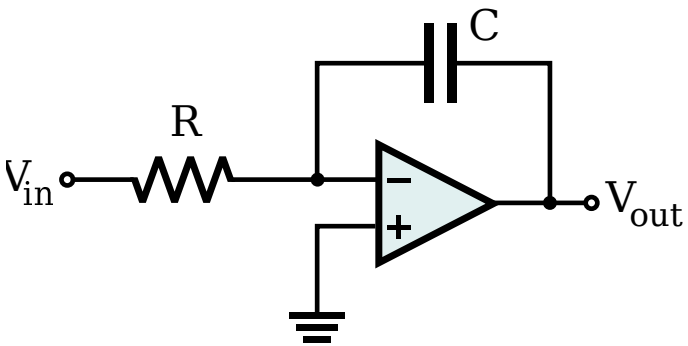
Inductiveload [Public domain]

$$V_{\text{out}}(t) = -\frac{1}{RC}\int_0^t V_{\text{in}}(t)dt$$



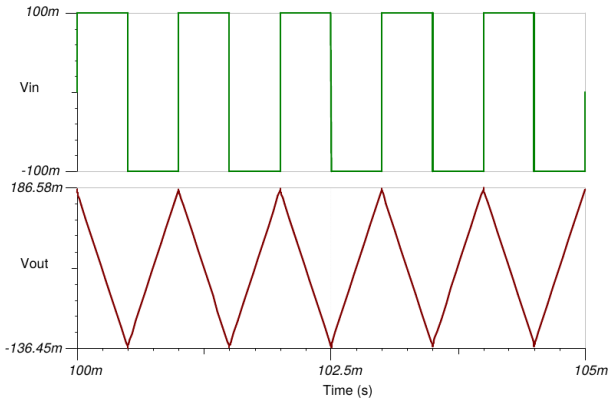
(Texas Instruments 2019)

Pastaba: Analoginės skaičiavimo mašinos



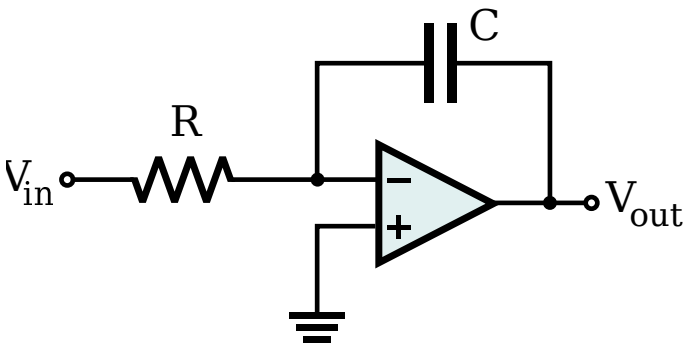
Inductiveload [Public domain]

$$V_{\text{out}}(t) = -\frac{1}{RC}\int_0^t V_{\text{in}}(t)dt$$



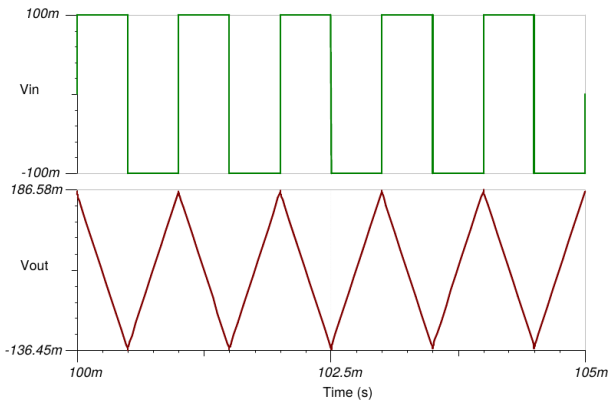
(Texas Instruments 2019)

Pastaba: Analoginės skaičiavimo mašinos

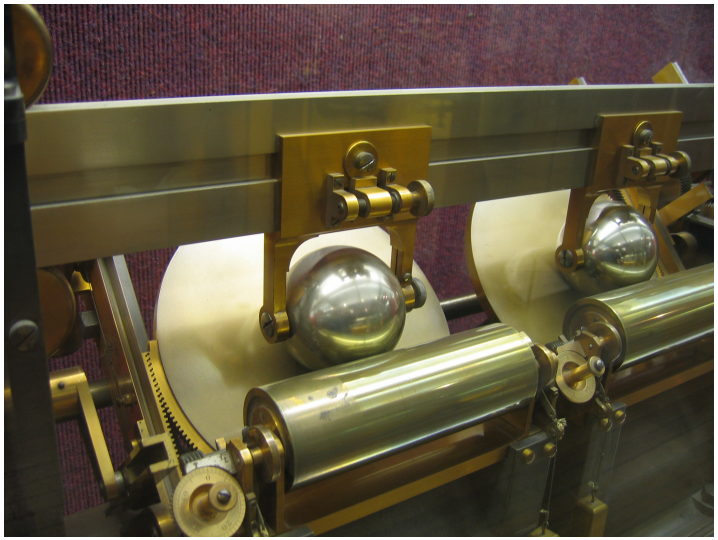


Inductiveload [Public domain]

$$V_{\text{out}}(t) = -\frac{1}{RC}\int_0^t V_{\text{in}}(t)dt$$

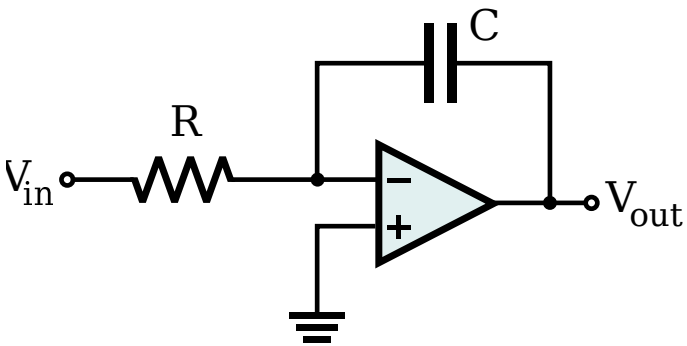


(Texas Instruments 2019)



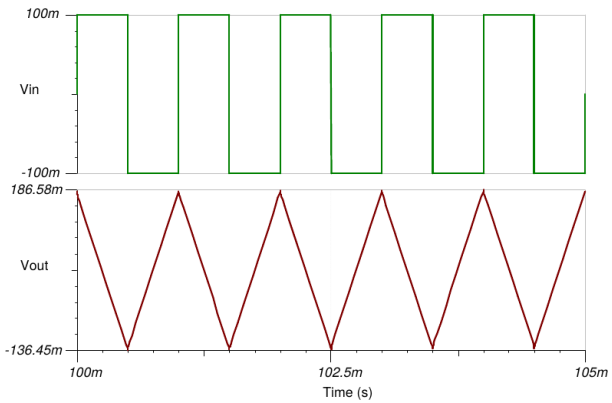
Potvynių prognozavimo mašina, apie 1878
Andy Dingley [CC BY 3.0]

Pastaba: Analoginės skaičiavimo mašinos

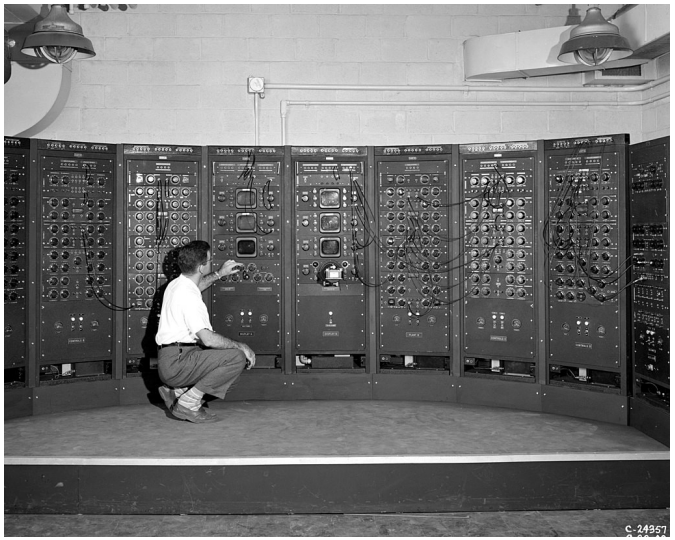


Inductiveload [Public domain]

$$V_{\text{out}}(t) = -\frac{1}{RC}\int_0^t V_{\text{in}}(t)dt$$

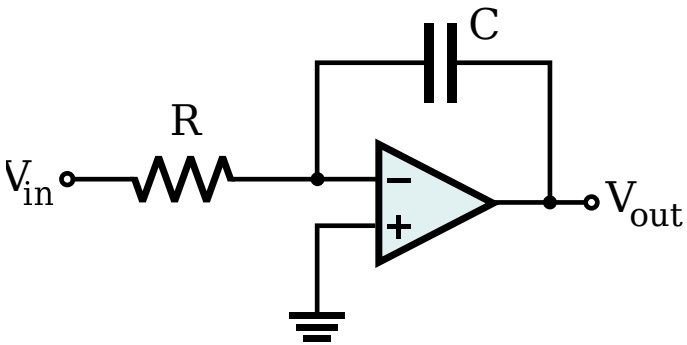


(Texas Instruments 2019)



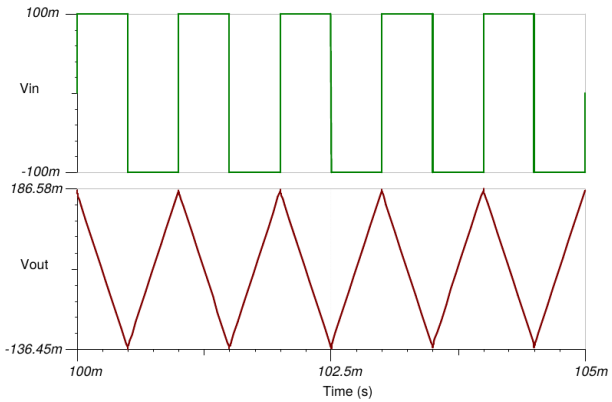
Elektroninis analoginis kompiuteris, apie 1949
NASA [Public domain]

Pastaba: Analoginės skaičiavimo mašinos

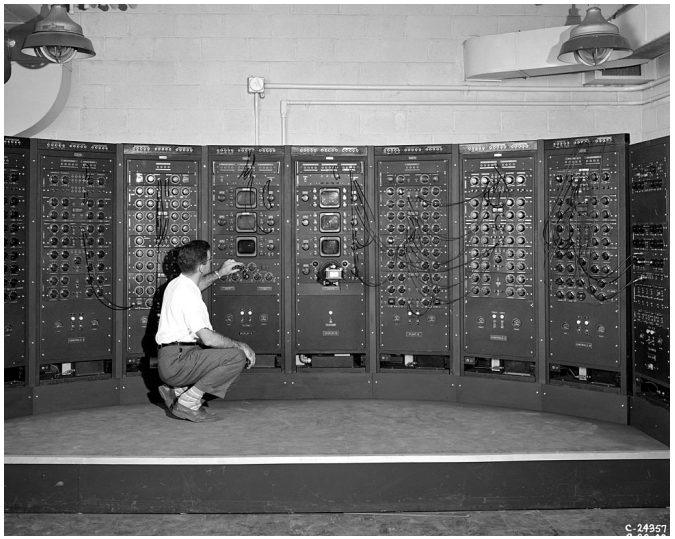


Inductiveload [Public domain]

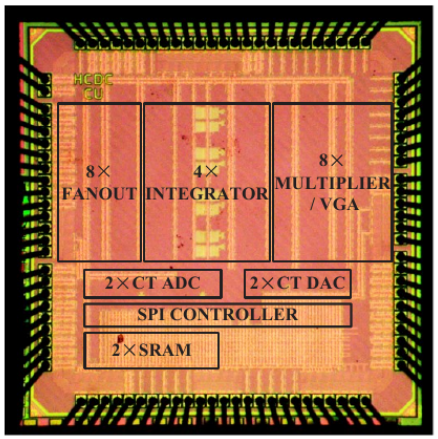
$$V_{\text{out}}(t) = -\frac{1}{RC}\int_0^t V_{\text{in}}(t)dt$$



(Texas Instruments 2019)

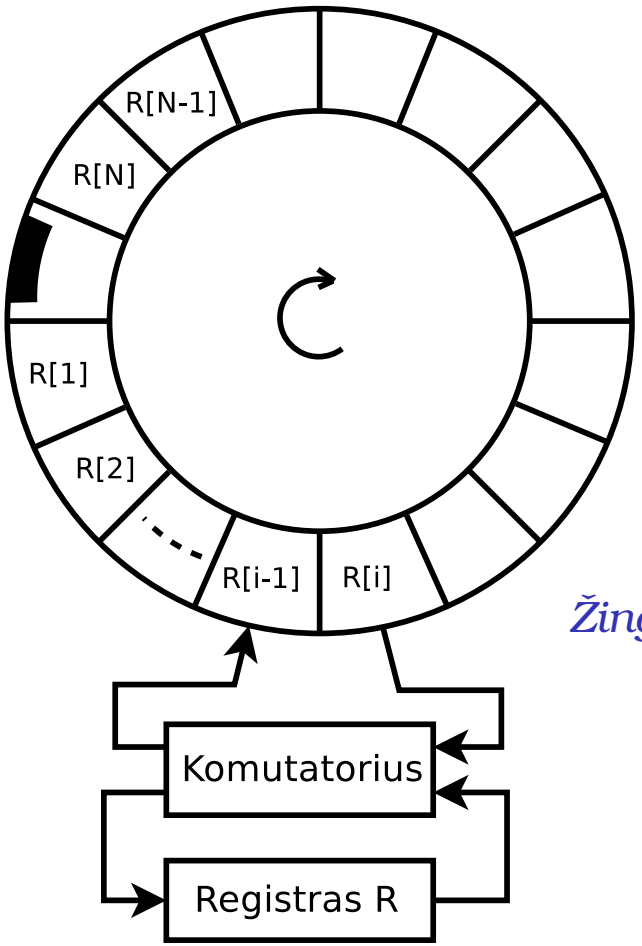


Elektroninis analoginis kompiuteris, apie 1949
NASA [Public domain]



(Guo et al. 2015)

“Burbuliukų” mašina



Žingsnis 1. Nustatyti $R \leftarrow R_1$.
(R – vidinis mašinos registras.)

Žingsnis i. visiems $1 < i \leq N$:

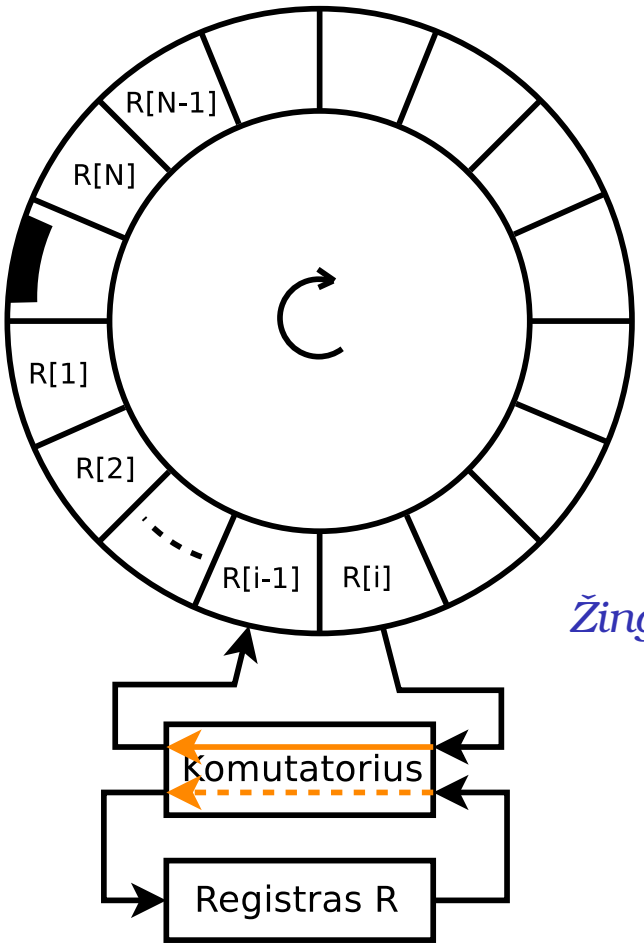
Arba

i. nustatyti $R_{i-1} \leftarrow R, R \leftarrow R_i$, arba
ii. nustatyti $R_{i-1} \leftarrow R_i$, paliekant R nepakeistą.

Žingsnis $N+1$. Nustatyti $R_N \leftarrow R$.

(Knuth 1997), sk. 5.3.4 (psl. 246)

“Burburiukų” mašina



Žingsnis 1. Nustatyti $R \leftarrow R_1$.
(R – vidinis mašinos registras.)

Žingsnis i. visiems $1 < i \leq N$:

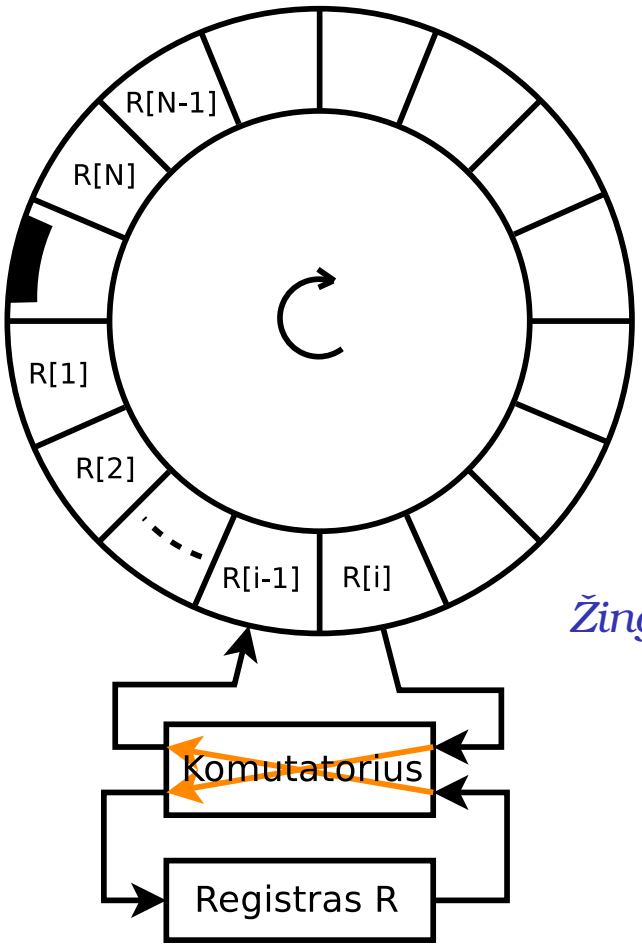
Arba

i. nustatyti $R_{i-1} \leftarrow R, R \leftarrow R_i$, arba
ii. nustatyti $R_{i-1} \leftarrow R_i$, paliekant R nepakeistą.

Žingsnis $N+1$. Nustatyti $R_N \leftarrow R$.

(Knuth 1997), sk. 5.3.4 (psl. 246)

“Burbuliukų” mašina



Žingsnis 1. Nustatyti $R \leftarrow R_1$.
(R – vidinis mašinos registras.)

Žingsnis i. visiems $1 < i \leq N$:

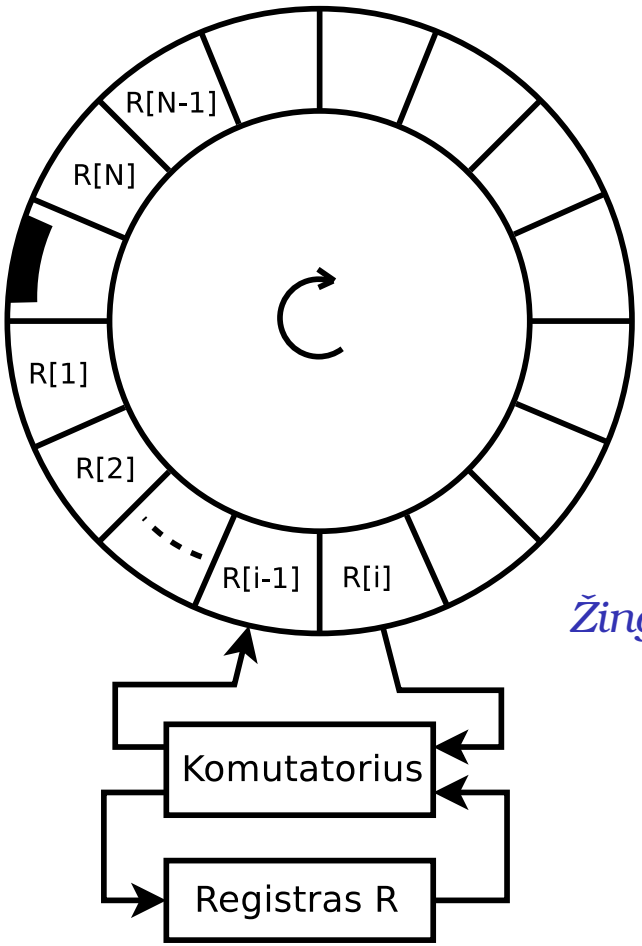
Arba

i. nustatyti $R_{i-1} \leftarrow R, R \leftarrow R_i$, arba
ii. nustatyti $R_{i-1} \leftarrow R_i$, paliekant R nepakeistą.

Žingsnis $N+1$. Nustatyti $R_N \leftarrow R$.

(Knuth 1997), sk. 5.3.4 (psl. 246)

“Burbuliukų” mašina



Žingsnis 1. Nustatyti $R \leftarrow R_1$.
(R – vidinis mašinos registras.)

Žingsnis i. visiems $1 < i \leq N$:

Arba

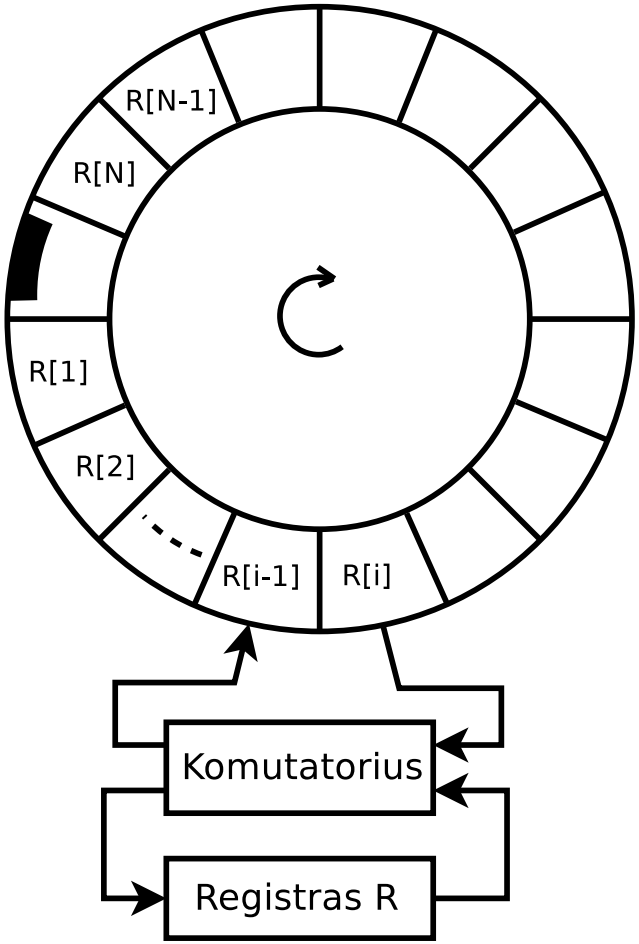
i. nustatyti $R_{i-1} \leftarrow R, R \leftarrow R_i$, arba
ii. nustatyti $R_{i-1} \leftarrow R_i$, paliekant R nepakeistą.

Žingsnis $N+1$. Nustatyti $R_N \leftarrow R$.

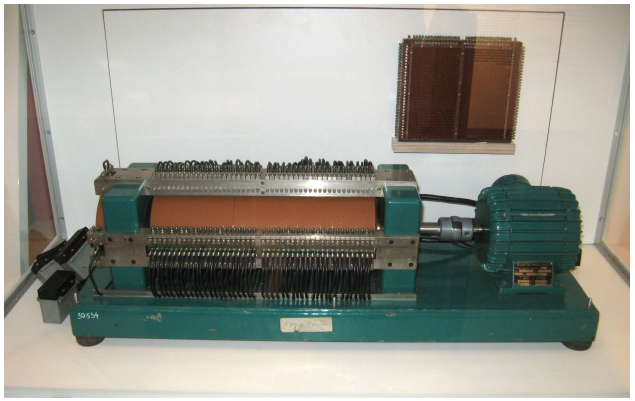
Pasirodo, šiam automatui optimalus rikiavimo algoritmas yra rikiavimas “burbuliuko” metodu! (Howard B. Demuth, PhD Thesis, 1956)

(Knuth 1997), sk. 5.3.4 (psl. 246)

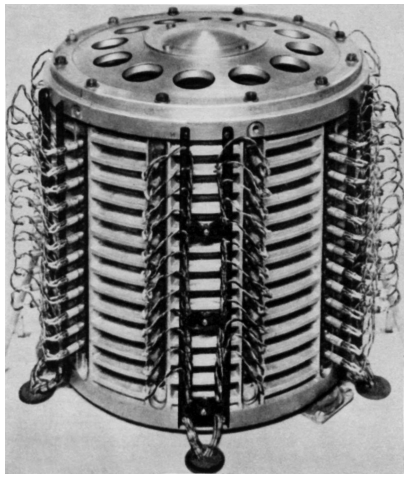
“Burbuliukų” mašina



(Knuth 1997), sk. 5.3.4 (psl. 246)



Wikipedia: the BESK drum memory



Wikipedia: the Polish ZAM-41 drum memory

’float’ tipo skaičių palyginimas

```c
#include <stdio.h>
#include <math.h>

int main()
{
    float a = 0.0, b = 0.0;
    float ratio = a/b;

    printf( "%f\n", ratio );
    printf( "%f\n", a );

    if( a <= ratio ) {
        printf( "YES\n" );
    } else {
        printf( "NO\n" );
    }

    if( a > ratio ) {
        printf( "YES\n" );
    } else {
        printf( "NO\n" );
    }

    return 0;
}
```

’float’ tipo skaičių palyginimas

```c
#include <stdio.h>
#include <math.h>

int main()
{
    float a = 0.0, b = 0.0;
    float ratio = a/b;

    printf( "%f\n", ratio );
    printf( "%f\n", a );

    if( a <= ratio ) {
        printf( "YES\n" );
    } else {
        printf( "NO\n" );
    }

    if( a > ratio ) {
        printf( "YES\n" );
    } else {
        printf( "NO\n" );
    }

    return 0;
}
```

```
+ ./float-comparisons
-nan
0.000000
NO
NO
```

’float’ tipo skaičių palyginimas

```c
#include <stdio.h>
#include <math.h>

int main()
{
    float a = 0.0, b = 0.0;
    float ratio = a/b;

    printf( "%f\n", ratio );
    printf( "%f\n", a );

    if( a <= ratio ) {
        printf( "YES\n" );
    } else {
        printf( "NO\n" );
    }

    if( a > ratio ) {
        printf( "YES\n" );
    } else {
        printf( "NO\n" );
    }

    return 0;
}
```

```
+ ./float-comparisons
-nan
0.000000
NO
NO
```

Slankaus kablelio skaičiams (kompiuteriuose atvaizduotiems realiems skaičiams) žemiau pateiktas teiginys **negalioja**:

$$\neg(a \leq b) \Rightarrow a > b$$

Raketos “Patriot” klaida...

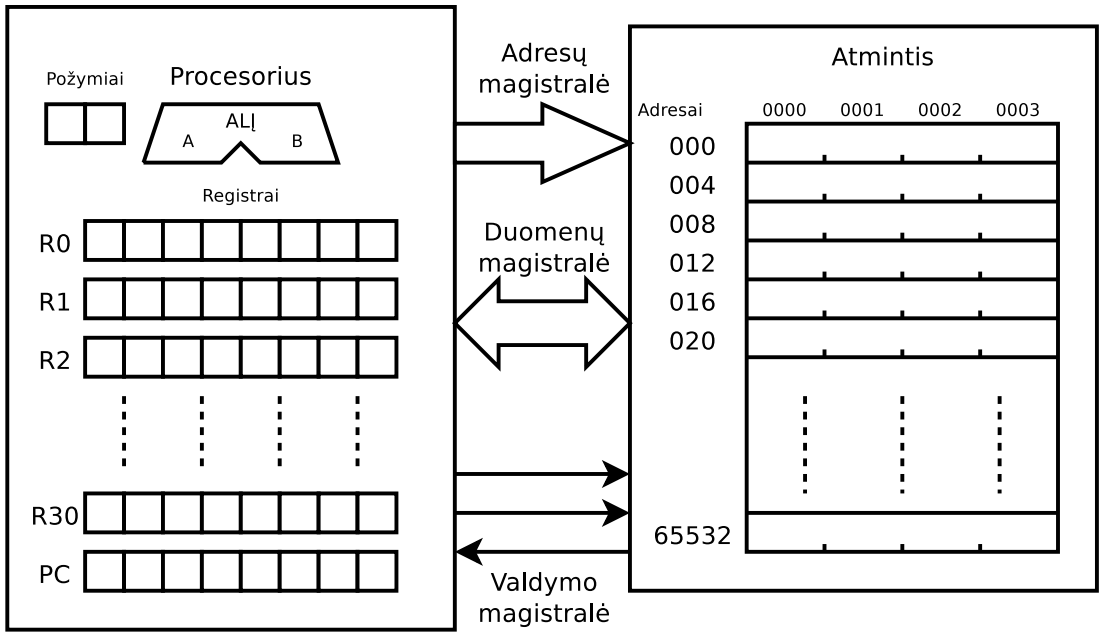
American Patriot Missile

[CC-BY-SA]

- Sistemos kompiuteris buvo **dvejetainis**;
- Sistemos darbo laikas buvo fiksuojamas **24 bitų** registre;
- Sistemos laikrodis matavo **1/10 sekundės** intervalais;
- Skaičius 1/10, paverstas į dvejetainę sistemą yra **begalinė** dvejetainė trupmena – jo negalime atvaizduoti tiksliai dvejetainėje mašinoje;
- Apvalinimo klaida buvo: $0.1_{10} - 0.0001\,1001\,1001\,1001\,1001\,1000_2 = 9.5 \times 10^{-08}_{10}$
- Per 100 val. susikaupia paklaida: $9.5 \times 10^{-08} ds \cdot 10 \frac{ds}{s} \cdot 3600 \frac{s}{h} \cdot 100 h = \mathbf{0.34s}$
- Per **0.34s** atskrendanti Scud raketa nukeliauja daugiau nei pusę kilometro ir “pasprunka” nuo “Patriot” sistemos...

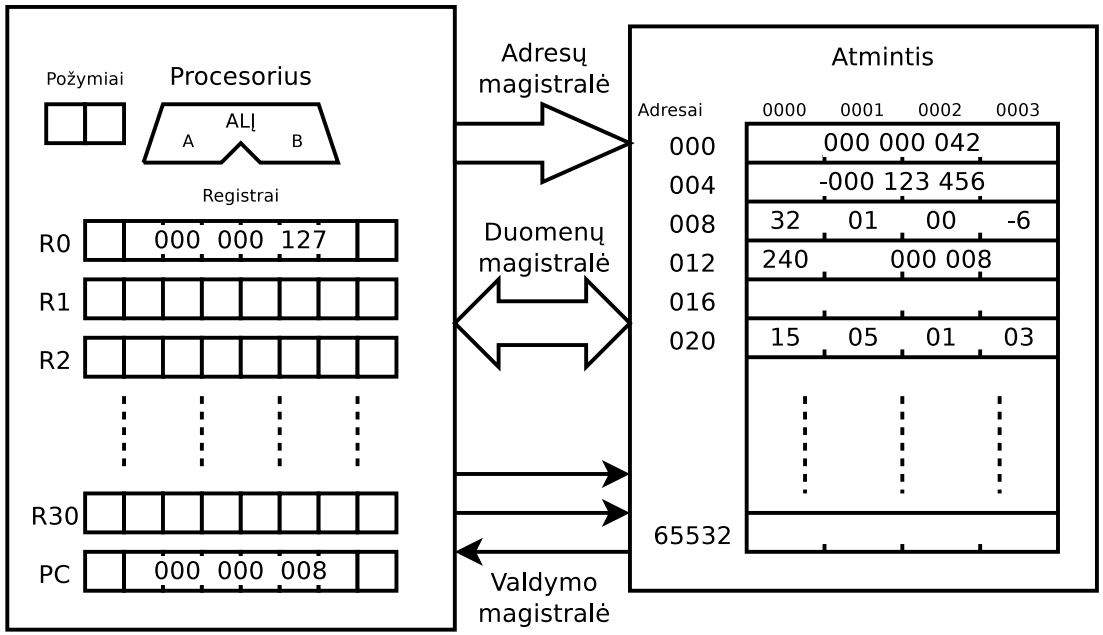
Apibendrinta kompiuterio schema

Fon Noimano (von Neumann) architektūra

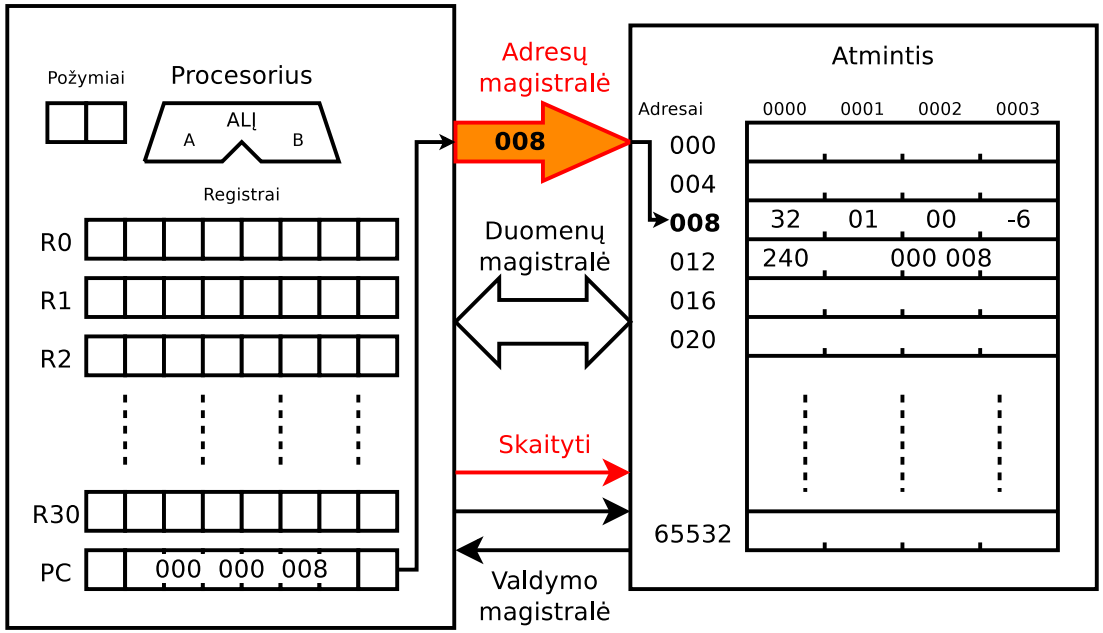


Apibendrinta kompiuterio schema

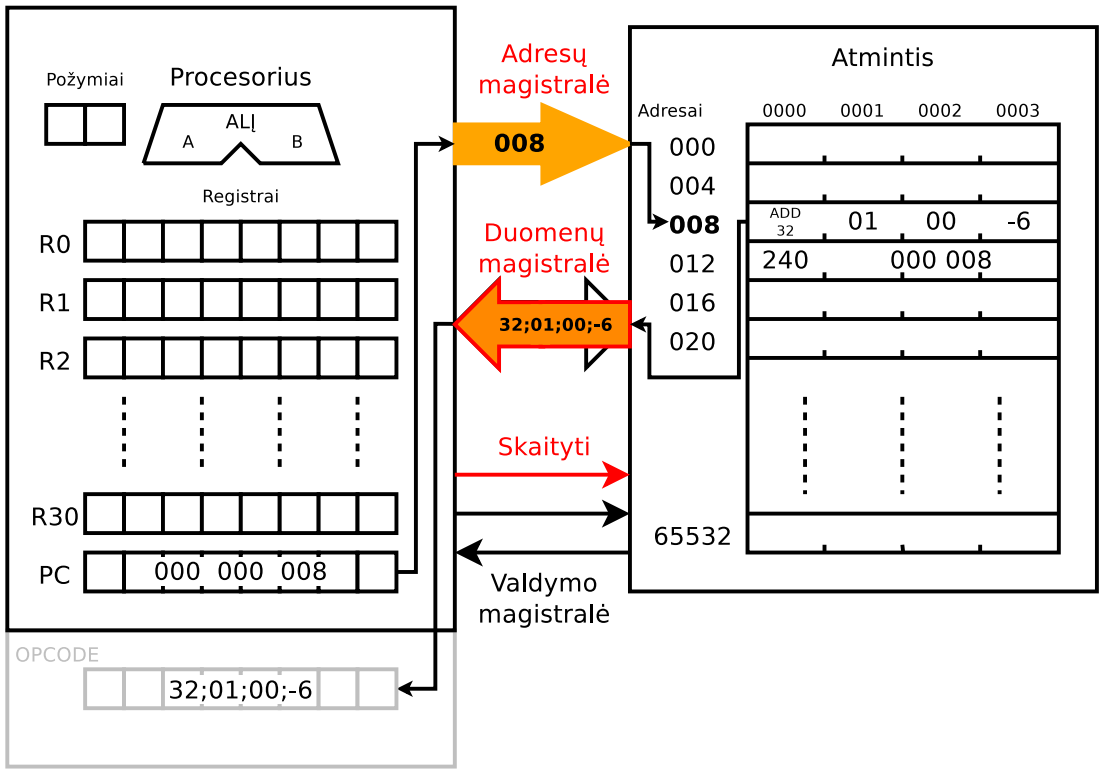
Fon Noimano (von Neumann) architektūra



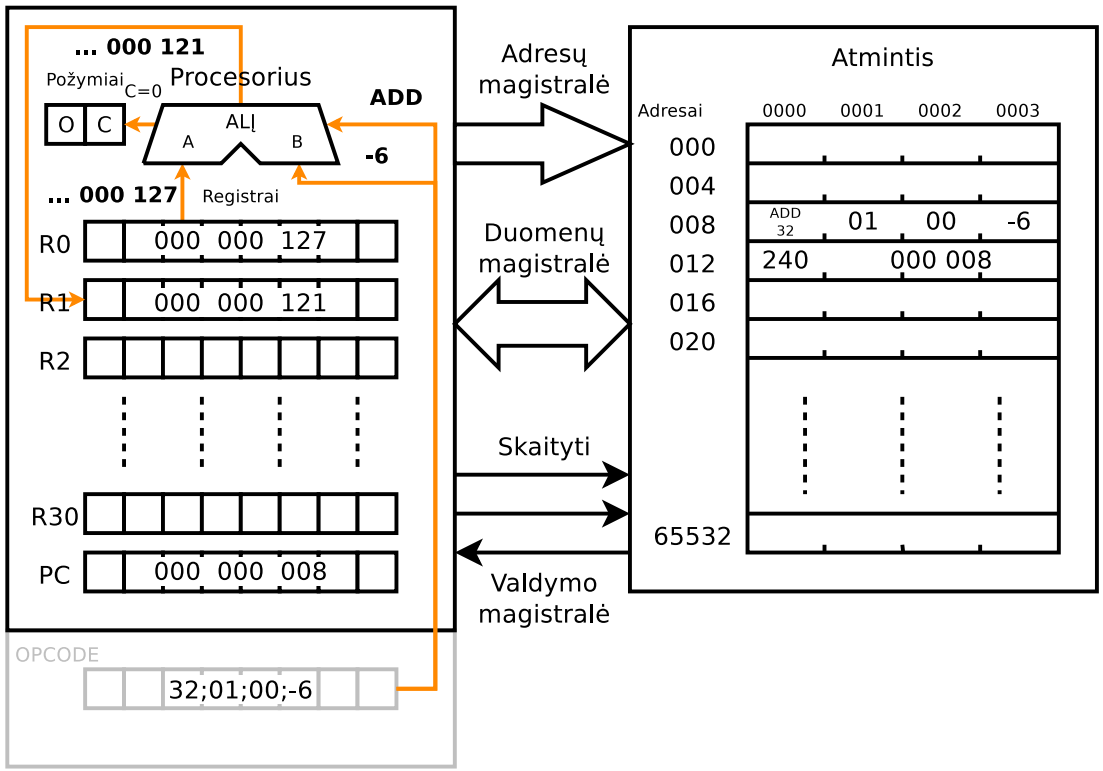
Komandos vykdymo ciklas



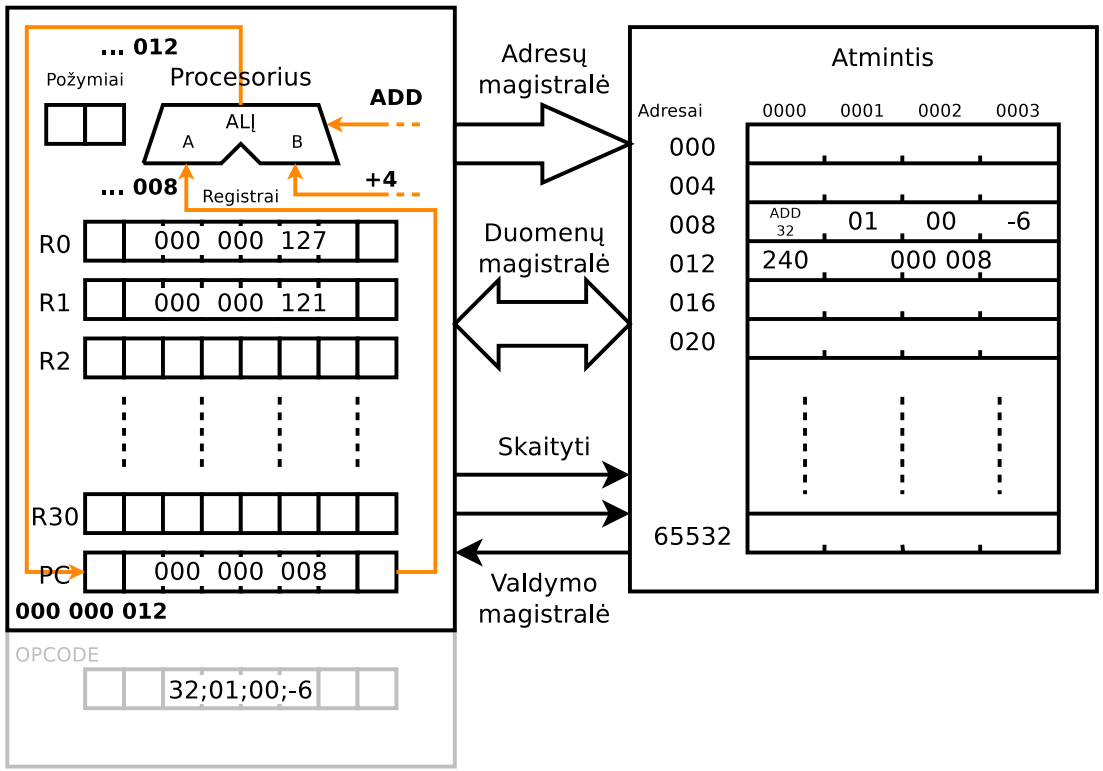
Komandos vykdymo ciklas



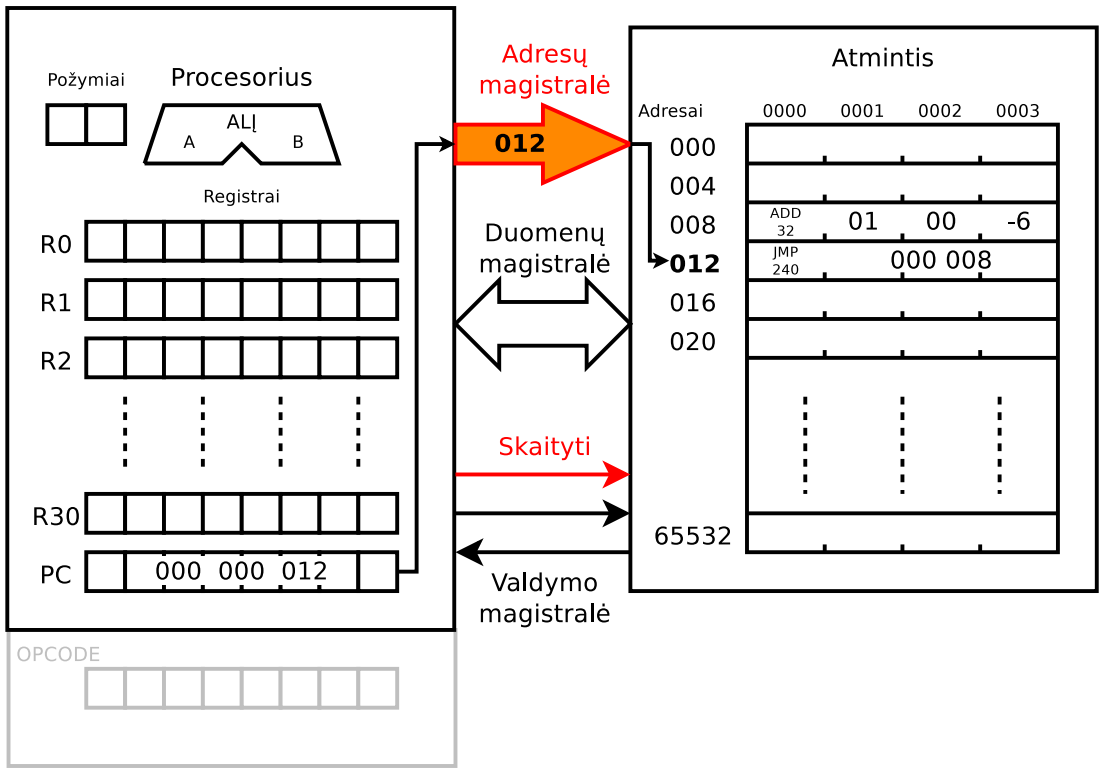
Komandos vykdymo ciklas



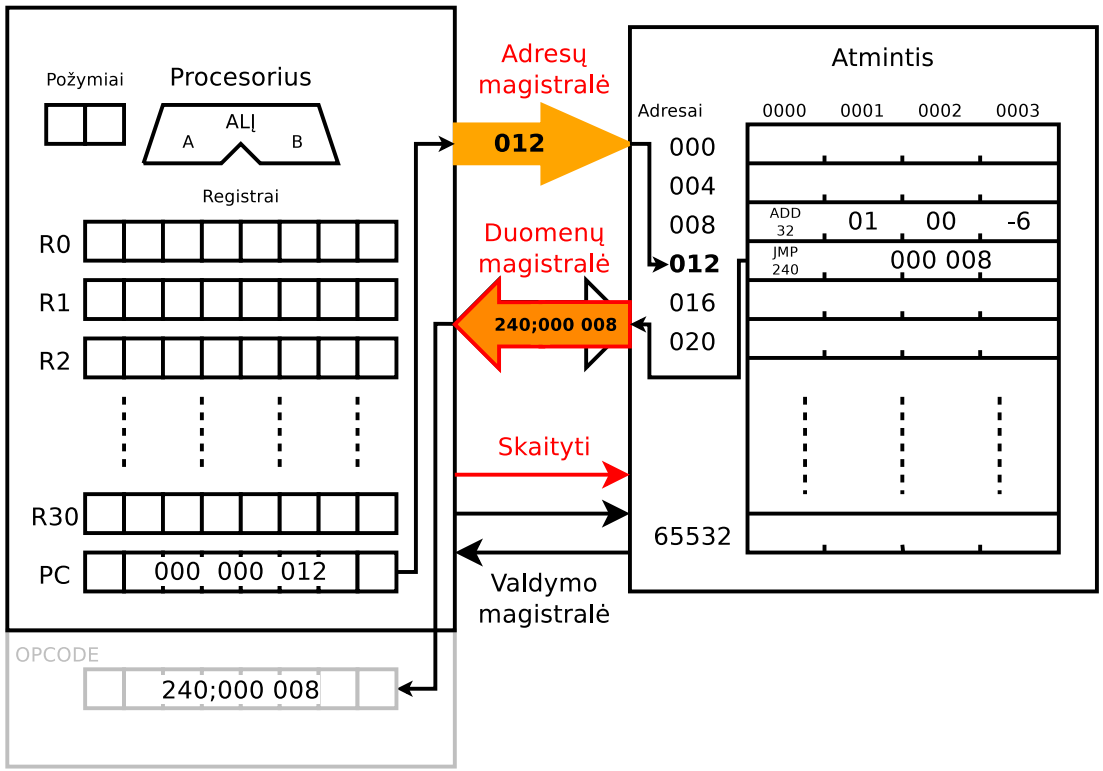
Komandos vykdymo ciklas



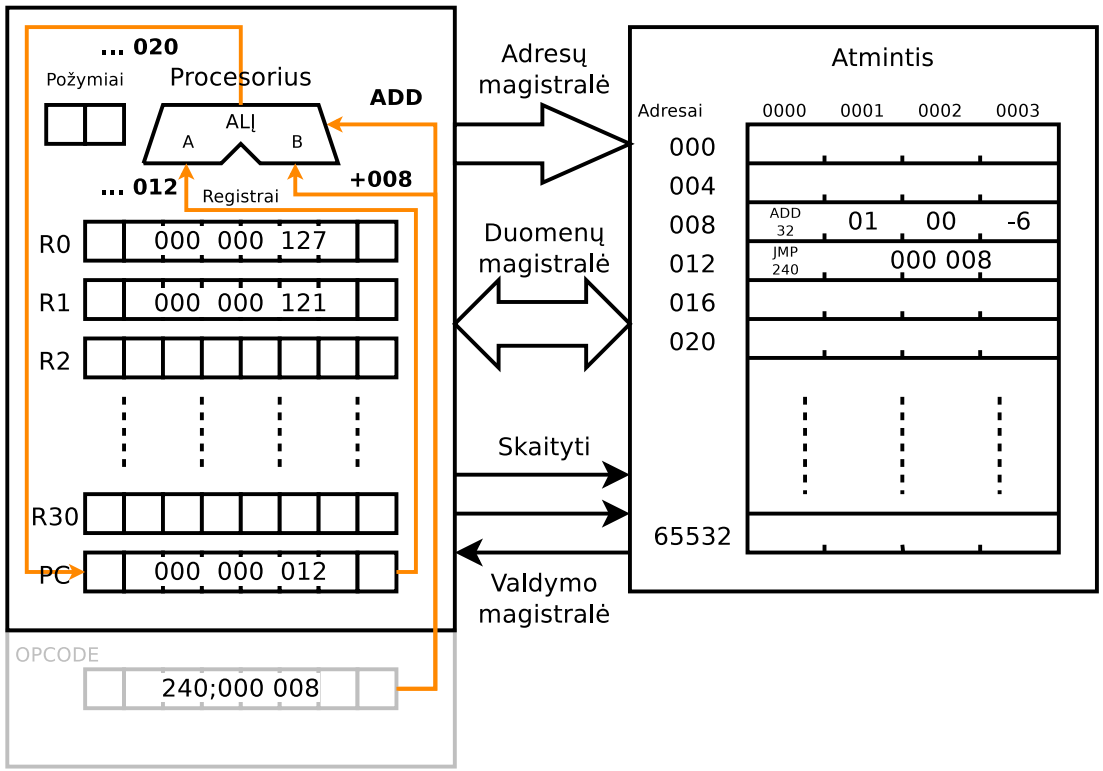
Komanda JMP


Požymiai
Procesorius
ALĮ
A
B
Registrai
R0
R1
R2
R30
PC
000 000 012
OPCODE
Adresų magistralė
012
Duomenų magistralė
Skaityti
Valdymo magistralė
Atmintis
Adresai
0000 0001 0002 0003
000
004
008
012
016
020
65532
ADD 32
01 00 -6
JMP 240
000 008

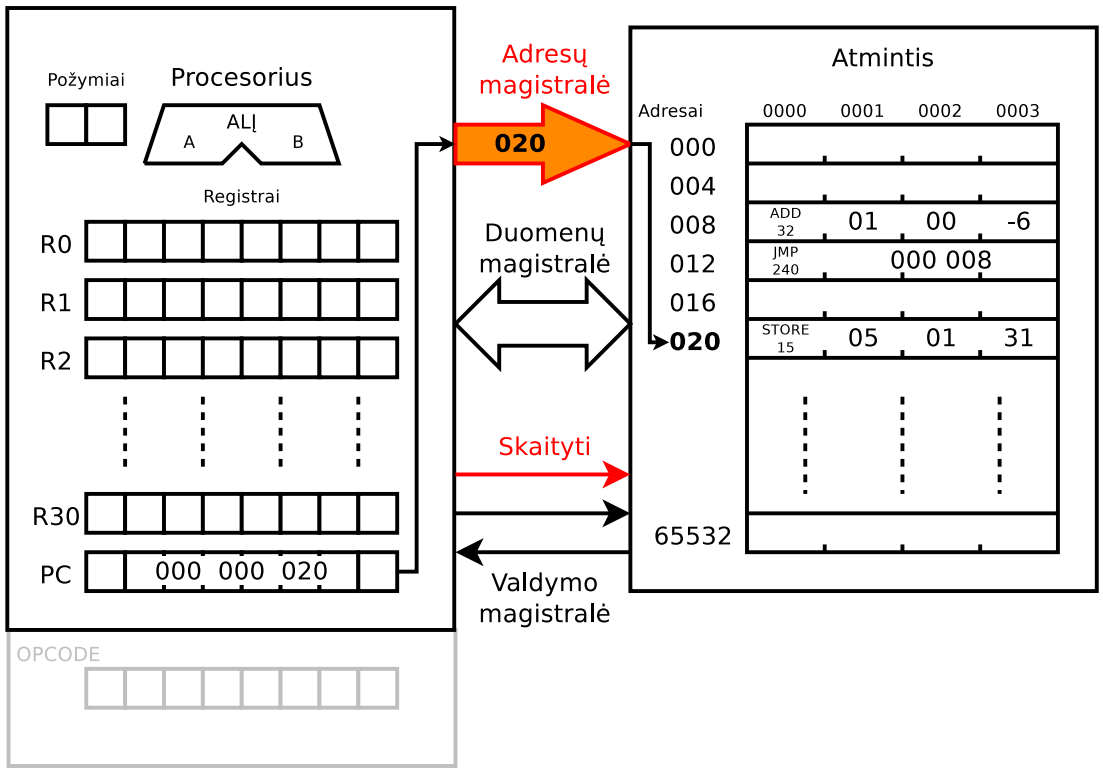
Komanda JMP


Procesorius
Požymiai
ALĮ
A
B
Registrai
R0
R1
R2
R30
PC
000 000 012
OPCODE
240;000 008
Adresų magistralė
012
Duomenų magistralė
240;000 008
Skaityti
Valdymo magistralė
Atmintis
Adresai
0000 0001 0002 0003
000
004
008
012
016
020
65532
ADD 32
01 00 -6
JMP 240
000 008

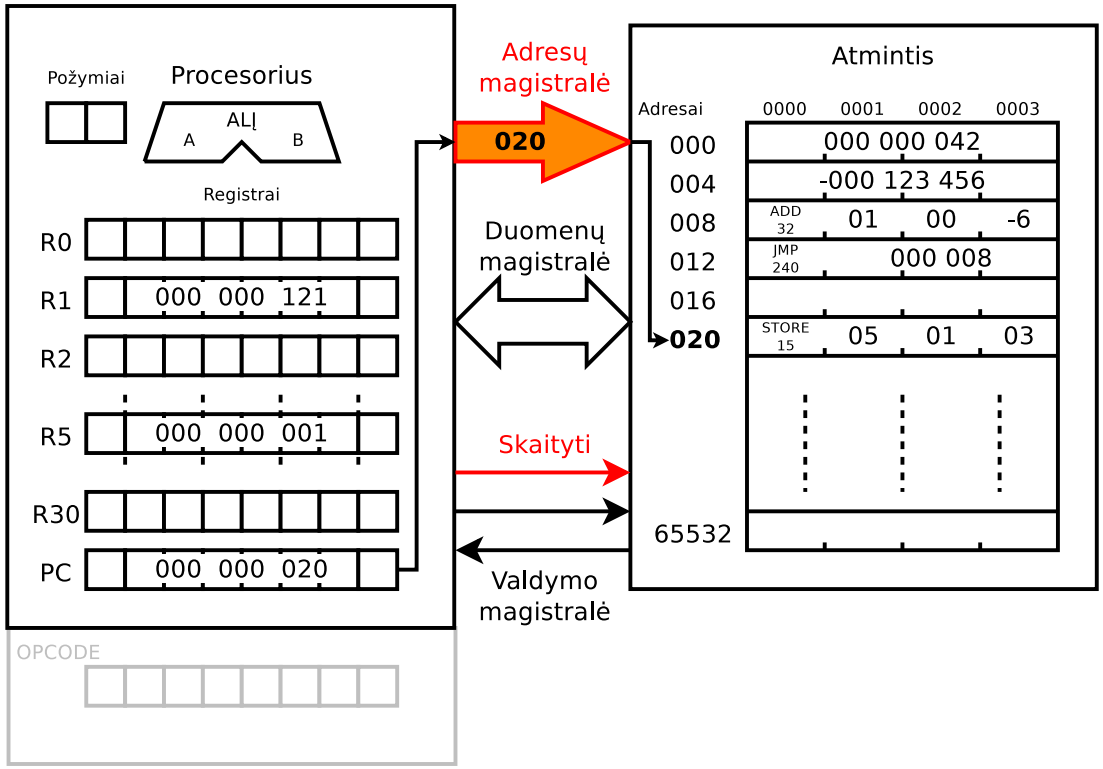
Komanda JMP


... 020
Požymiai
Procesorius
ADD
ALĮ
A
B
... 012
Registrai
+008
R0
000 000 127
R1
000 000 121
R2
R30
PC
000 000 012
OPCODE
240;000 008
Adresų magistralė
Duomenų magistralė
Skaityti
Valdymo magistralė
Atmintis
Adresai
0000
0001
0002
0003
000
004
008
012
016
020
65532
ADD 32
01
00
-6
JMP 240
000 008

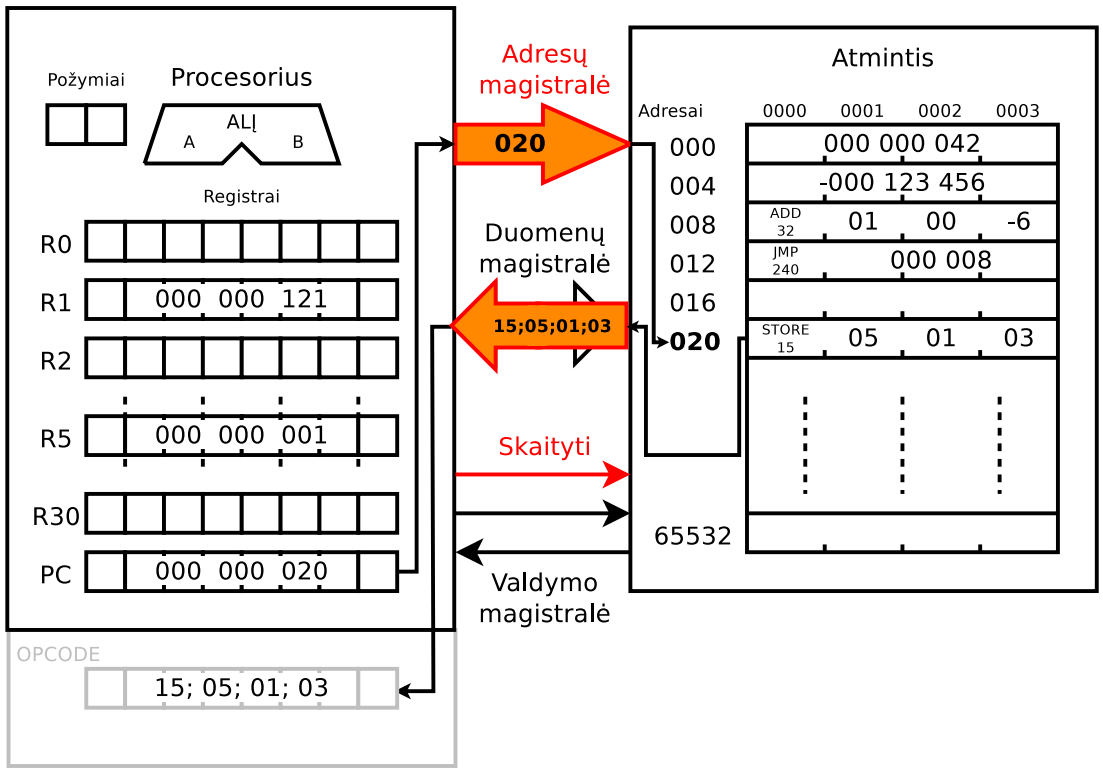
Komanda JMP


Požymiai
Procesorius
ALĮ
A
B
Registrai
R0
R1
R2
R30
PC
000 000 020
OPCODE
Adresų magistralė
020
Duomenų magistralė
Skaityti
Valdymo magistralė
Atmintis
Adresai
0000 0001 0002 0003
000
004
008
012
016
020
65532
ADD 32
01 00 -6
JMP 240
000 008
STORE 15
05 01 31

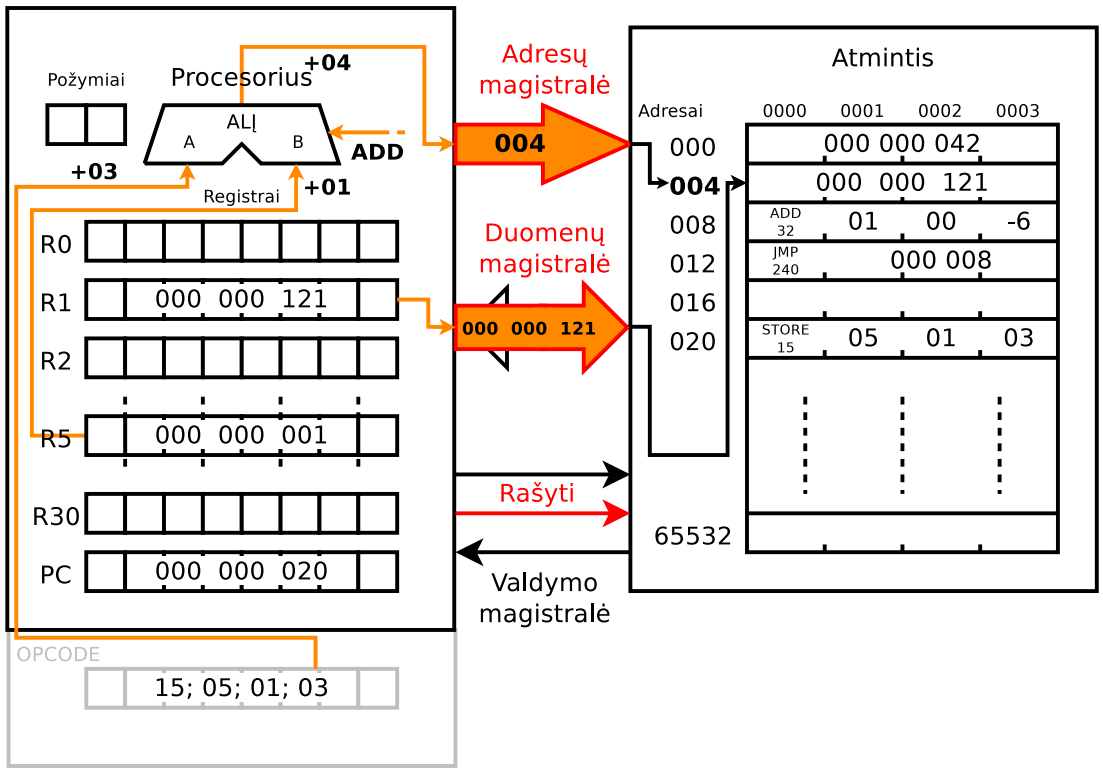
Komanda STORE



Komanda STORE

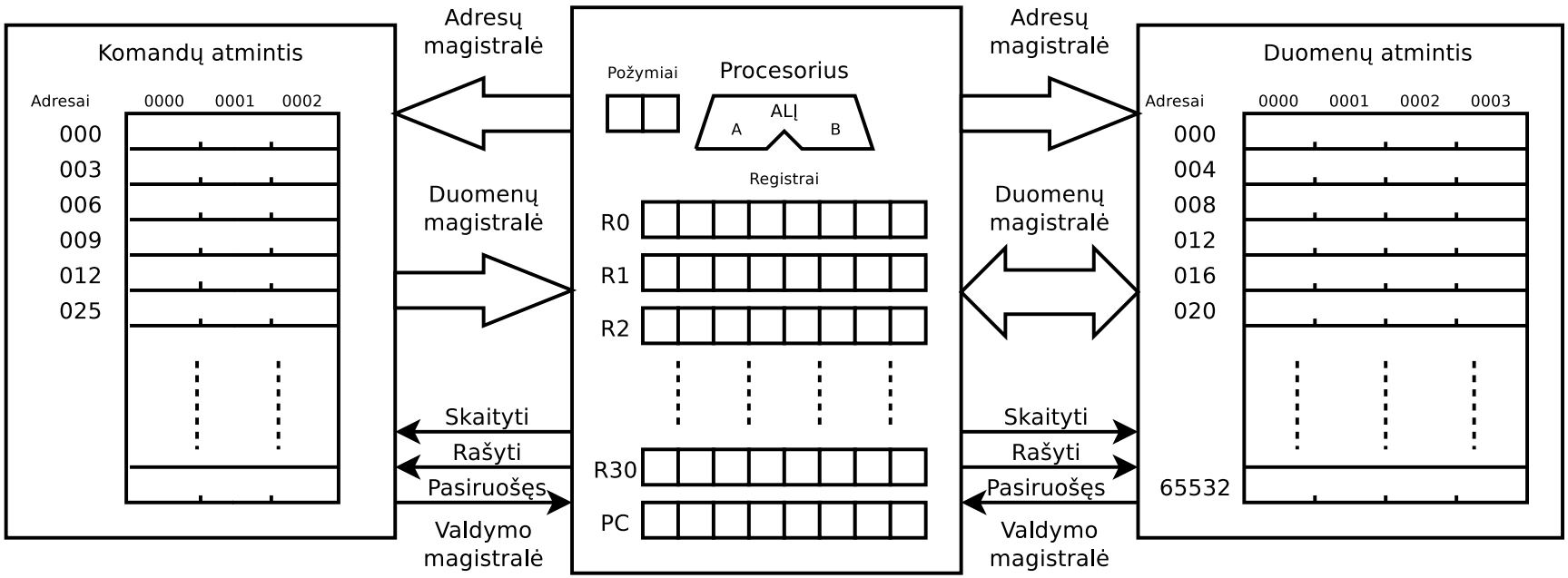


Komanda STORE



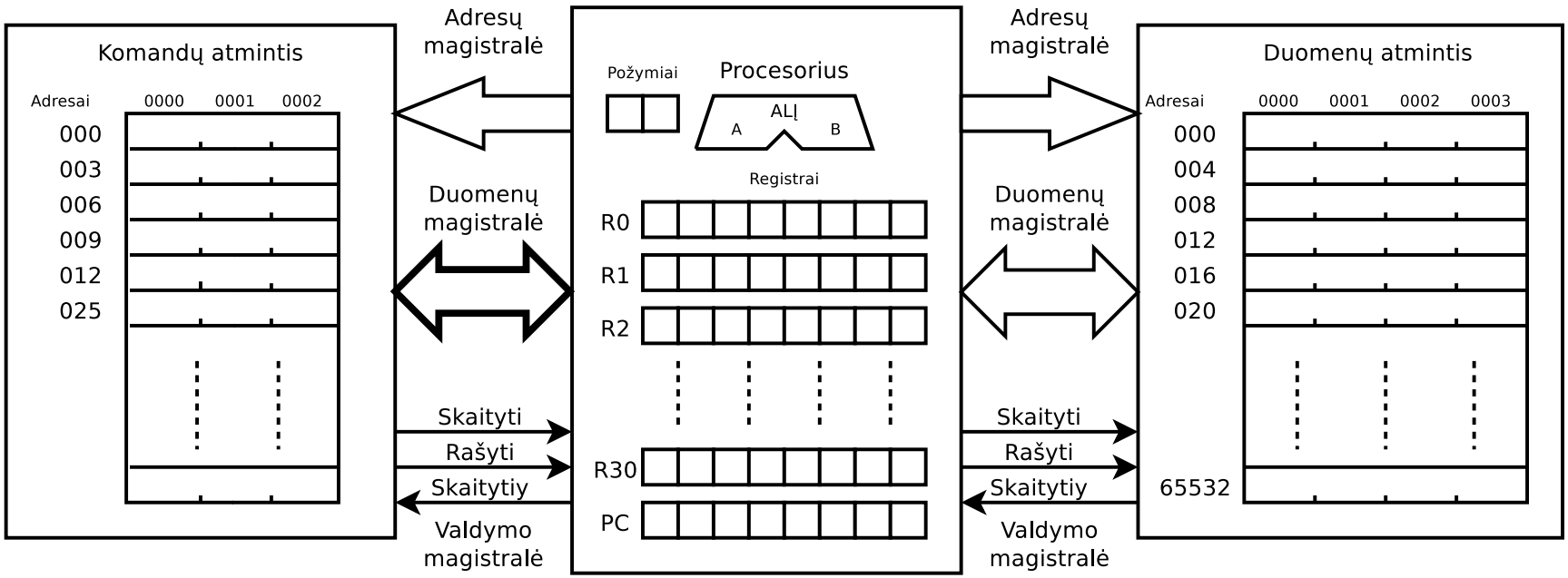
Harvardo architektūra

Klasikinė Harvardo architektūra

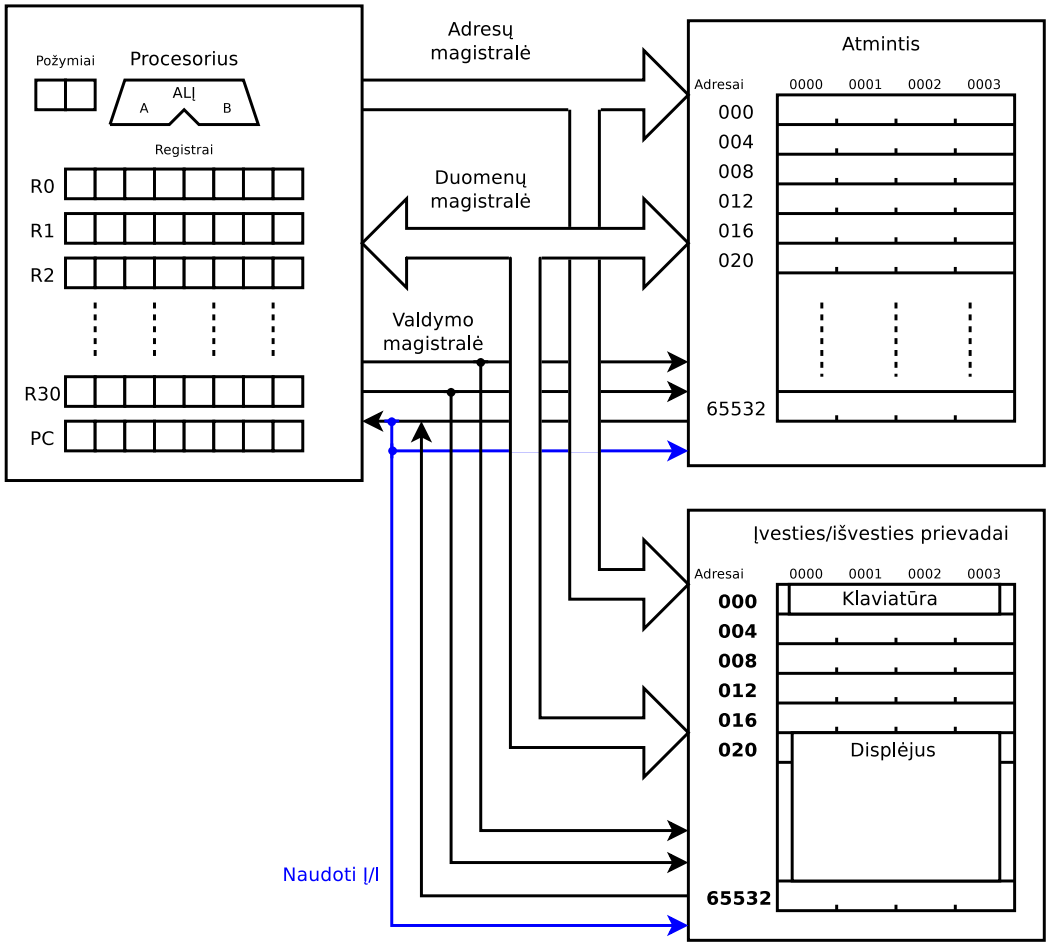


Harvardo architektūra

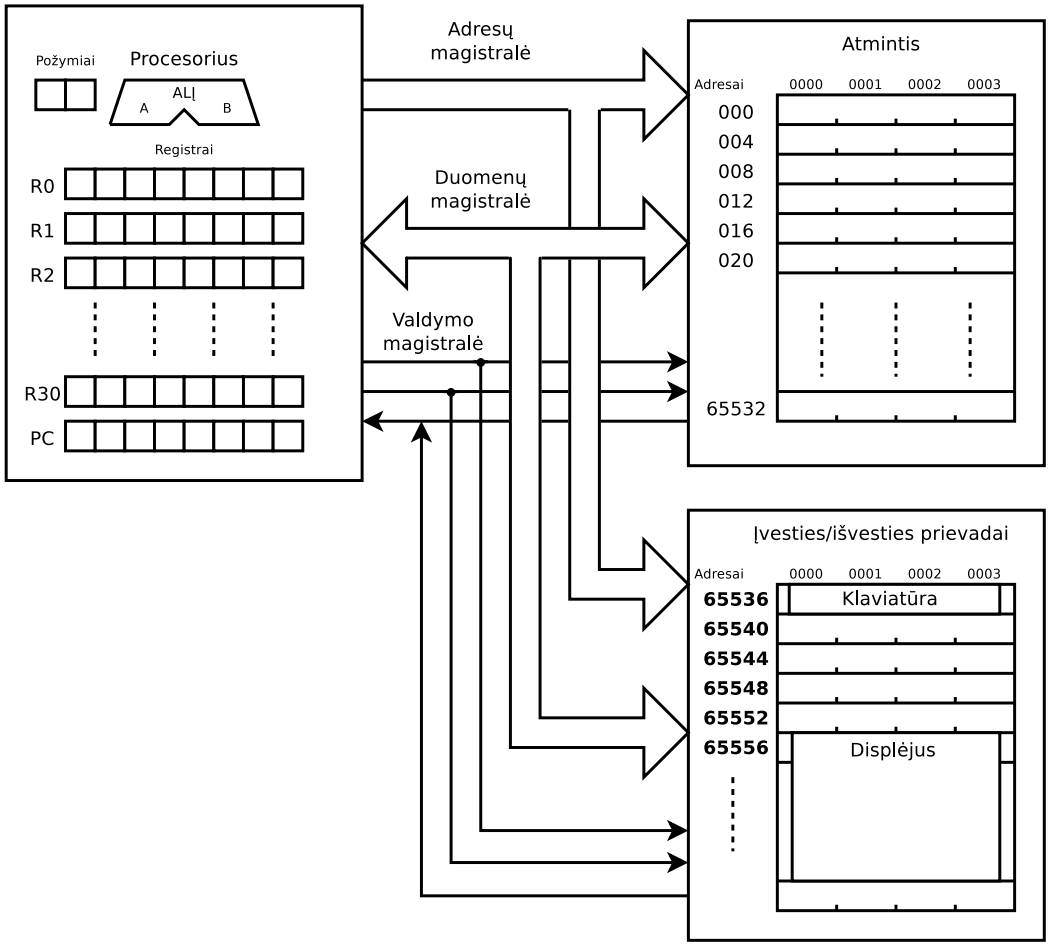
Modifikuota Harvardo architektūra



Atskiros atminties ir Į/I adresų erdvės



Bendra atminties ir Į/I adresų erdvė

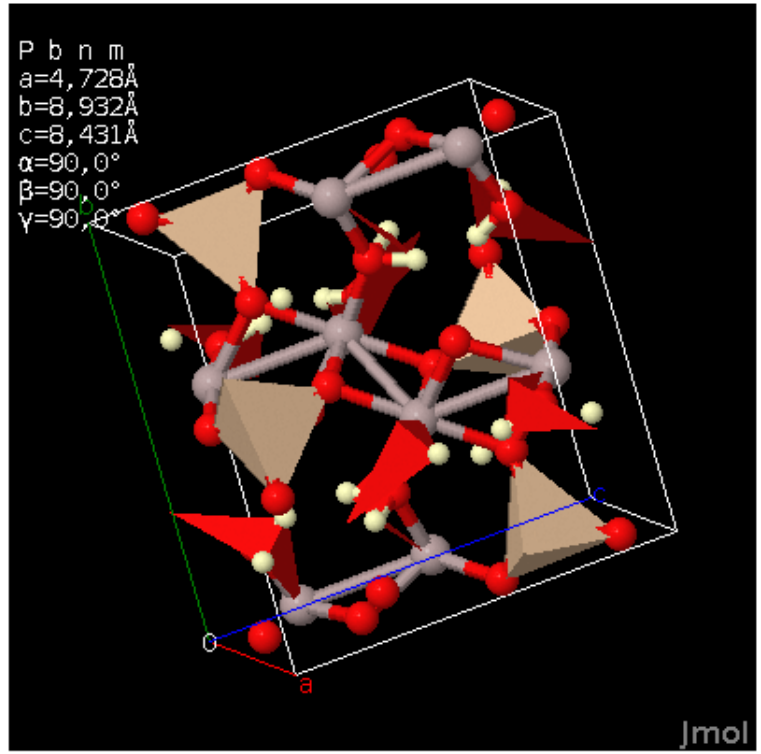


Apibendrinimas

- Skaitmeniniai kompiuteriai yra automatiniai, programuojami, universalūs įrenginiai (bet yra/buvo ir analoginiai kompiuteriai);
- Kompiuterių architektūrą reikia suprasti, jei norime programuoti juos teisingai ir efektyviai
- Egzistuoja įvairios kompiuterių architektūros (pvz. fon Noimano, Harvardo), bet visais atvejais esminiai komponentai yra procesorius ir atmintis;
- Procesorius skaito komandas iš atminties ir pagal tas komandas atlieka įvairius veiksmus su operandais iš atminties ar iš procesoriaus registrų;

Klausimai?

http://en.wikipedia.org/wiki/Topaz



Coordinates	2207377.cif
Original IUCr paper	HTML

http://www.crystallography.net/2207377.html

Šaltiniai

Demuth, Howard B. (Oct. 1956). "Electronic Data Sorting". PhD thesis. Stanford University.

Guo, Ning et al. (Sept. 2015). "Continuous-time hybrid computation with programmable nonlinearities". In: *ESSCIRC Conference 2015 - 41st European Solid-State Circuits Conference (ESSCIRC)*. IEEE, pp. 279–282. DOI: 10.1109/esscirc.2015.7313881.

Kann, Charles W. (2016). *Implementing a One Address CPU in Logisim*. WWW: https://open.umn.edu/opentextbooks/textbooks/implementing-a-one-address-cpu-in-logisim. Gettysburg College. URL: https://open.umn.edu/opentextbooks/formats/989.

Knuth, Donald E. (July 1997). *The art of computer programming. Sorting and searching*. Vol. 3. Addison-Wesley. ISBN: 978-02-0189-685-5.

Texas Instruments (2019). *Integrator circuit*. Tech. rep. Texas Instruments, pp. 1–6. URL: https://www.ti.com/lit/an/sboa275a/sboa275a.pdf.

Turing, A. M. (1937). "Computability and λ-definability.". English. In: *J. Symb. Log.* 2, pp. 153–163. ISSN: 0022-4812; 1943-5886/e. DOI: 10.2307/2268280.

Wikipedia (2020). *Turing machine*. URL: https://en.wikipedia.org/wiki/Turing_machine.