

Kompiuterių architektūra. Įvadas

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](#) licencijoje



Kurso organizacija

<https://emokymai.vu.lt/mod/resource/view.php?id=2554>

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 vadovą paskelbtą grafiką	Praktikos darbai atliekami pagal pratybų vadovą nurodytas užduotis ir atskirkaitomi 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. žodinių pristatymą su skaidrumais (PDF formatu). Žodinis pristatymas galimas studentams, pasiekusiems gerų rezultatų kurso metu ir gali būti užskaitomas kaip egzaminas (salygas žr. punkte „Egzaminas“).
Egzaminas	15	semestro pabaiga	Apytikriai 30 klausimų apklausa iš viso kurso dalykų (1 iki 9 lygio klausimai pagal Bloom klasifikaciją) naudojant elektronines mokymo priemones (Moodle, Open edX ar pan).

Reikalavimai egzaminui

Egzaminas	15	semestro pabaiga	<p>Aptykriai 30 klausimų apklausa iš viso kurso dalykų (1 iki 9 lygio klausimai pagal Bloom klasifikaciją) naudojant elektronines mokymo priemones (Moodle, Open edX ar pan).</p> <p>Kad studentai būtų prileisti prie egzamino, jie turi:</p> <ol style="list-style-type: none">1. Atliliki bent vieną praktikos darbą ir surinkti bent 30% galimų praktikos darbų balų (pratybų balai neapvalinami);2. Surinkti suminį balą už darbą per semestrą (iš praktikos darbų, tarpinio kontrolinio, darbo paskaitose, galimai kitų dėstytojų paskirtų užduočių), kad, parašius egzamino kontrolinį, būtų įmanoma surinkti patenkinamo pažymio balą (t. y. balą, užtikrinantį bent pažymį „5“);3. Ypač gerai pasirodžiusiems semestro metu studentams gali būti leidžiama laikyti išankstinį egzaminą, padarant žodini pranešimą dėstytojo paskirtu laiku užsiémimų metu. Norint laikyti išankstinį egzaminą, būtina:<ol style="list-style-type: none">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. <p>Jei norinčių laikyti egzaminą iš anksto studentų yra daugiau, negu leidžia turimas užsiémimų laikas, pirmenybė suteikiama studentams, surinkusiems daugiau balų.</p>
-----------	----	------------------	---

Reikalavimai egzaminui

<https://emokymai.vu.lt/mod/resource/view.php?id=53944>

		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 atsakyti bent 1/2 visų klausų ir 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).

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 autoriu 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

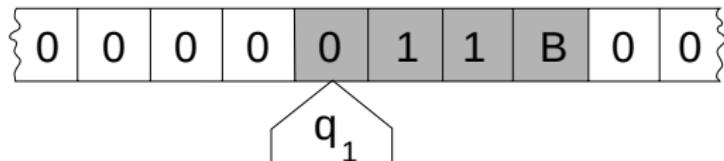
use strict;
use warnings;

my $selected_line;

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

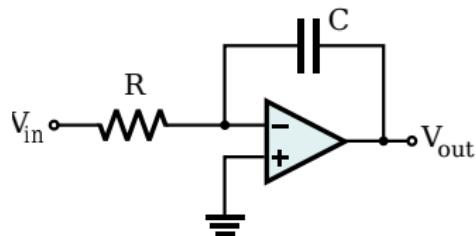
print $selected_line;
```

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



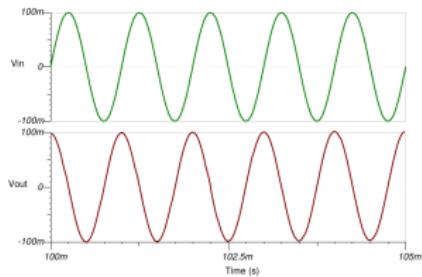
Nynexman4464 [CC-BY-SA 3.0]

Pastaba: Analoginės skaičiavimo mašinos



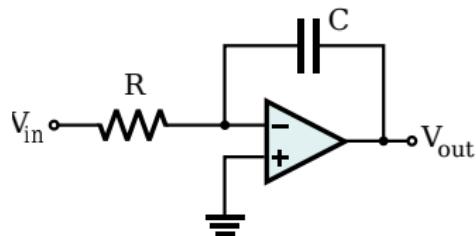
Inductiveload [Public domain]

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



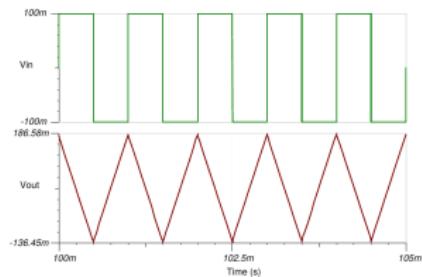
(Texas Instruments 2019)

Pastaba: Analoginės skaičiavimo mašinos



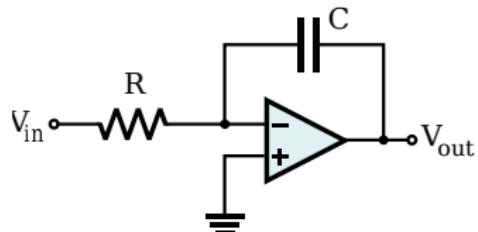
Inductiveload [Public domain]

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



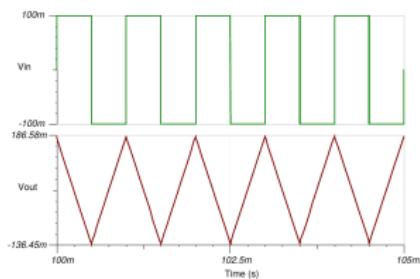
(Texas Instruments 2019)

Pastaba: Analoginės skaičiavimo mašinos



Inductiveload [Public domain]

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

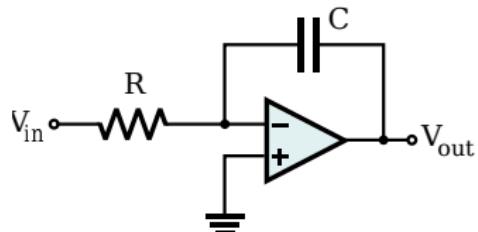


(Texas Instruments 2019)



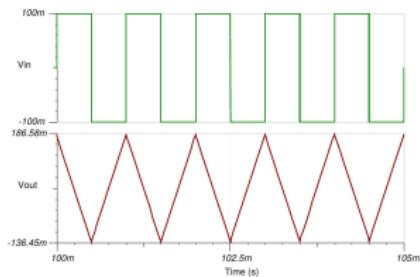
Potvynių prognozavimo mašina, apie 1878
Andy Dingley [CC BY 3.0]

Pastaba: Analoginės skaičiavimo mašinos

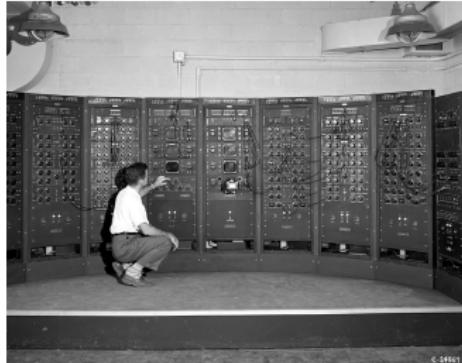


Inductiveload [Public domain]

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

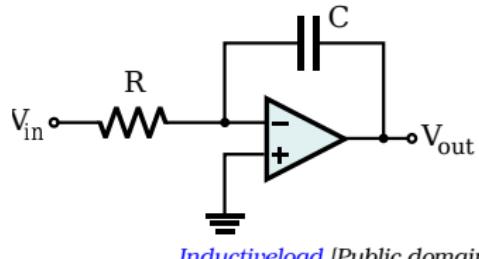


(Texas Instruments 2019)

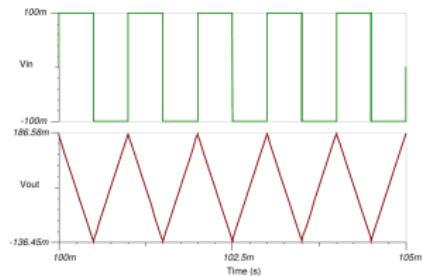


*Elektroninis analoginis kompiuteris, apie 1949
NASA [Public domain]*

Pastaba: Analoginės skaičiavimo mašinos



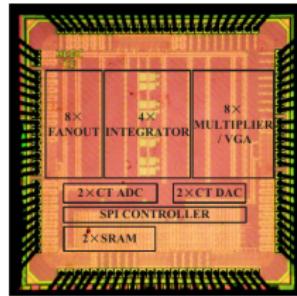
$$V_{out}(t) = -\frac{1}{RC} \int_0^t V_{in}(t) dt$$



(Texas Instruments 2019)

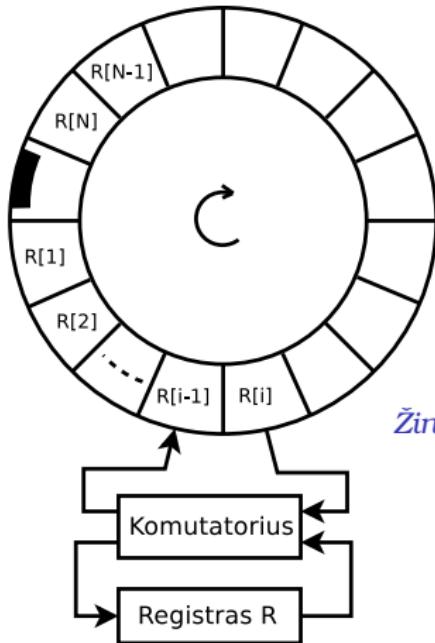


*Elektroninis analoginis kompiuteris, apie 1949
NASA [Public domain]*



(Guo et al. 2015)

“Burbuliukų” mašina



Žingsnis 1. Nustatyti $R \leftarrow R_1$.

(R – vidinis mašinos registras.)

Žingsnis i. visiems $1 < i \leq N$:

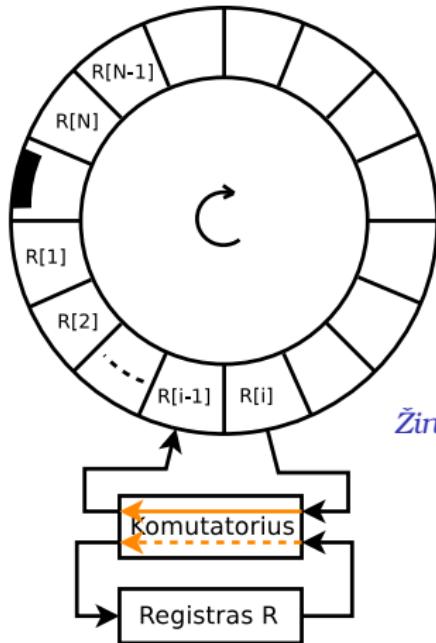
Arba

- i nustatyti $R_{i-1} \leftarrow R$, $R \leftarrow R_i$, arba
- fi 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 $1 < i \leq N$:

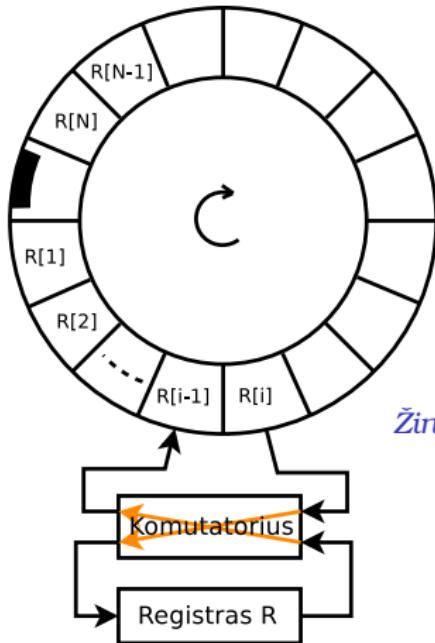
Arba

- i nustatyti $R_{i-1} \leftarrow R$, $R \leftarrow R_i$, arba
- fi 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 $1 < i \leq N$:

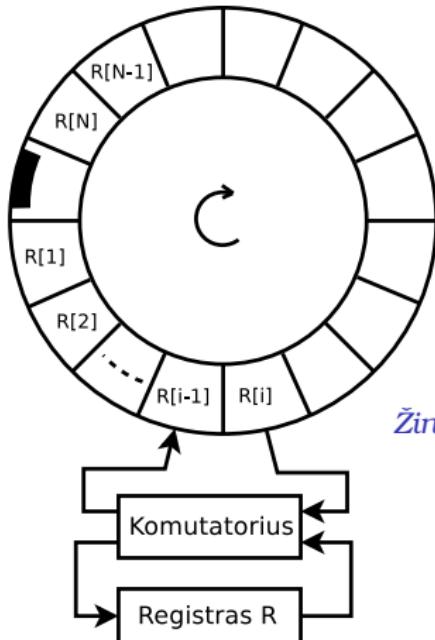
Arba

- i nustatyti $R_{i-1} \leftarrow R$, $R \leftarrow R_i$, arba
- fi 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

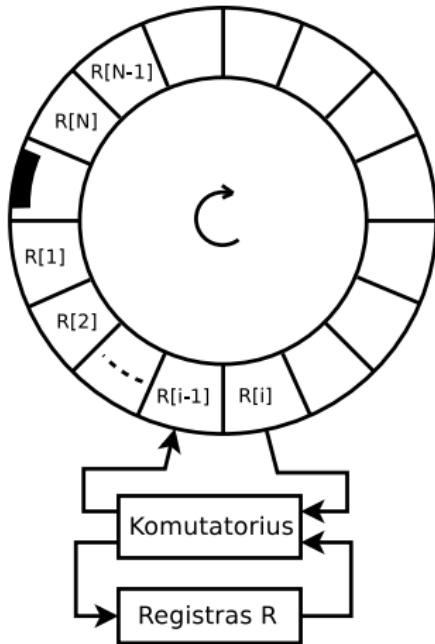
- i nustatyti $R_{i-1} \leftarrow R$, $R \leftarrow R_i$, arba
- fi nustatyti $R_{i-1} \leftarrow R_i$, paliekant R nepakeistą.

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

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

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

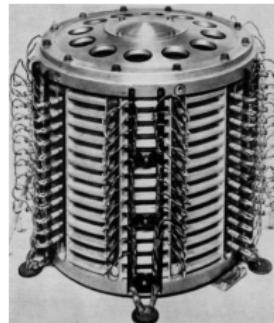
“Burbuliukų” mašina



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



Wikipedia: the [BESK drum memory](#)



Wikipedia: the Polish [ZAM-41 drum memory](#)

'float' tipo skaičių palyginimas

```
#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

```
#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

```
#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) žemaiu
pateiktas teiginys **negalioja**:

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

Raketos “Patriot” klaida...



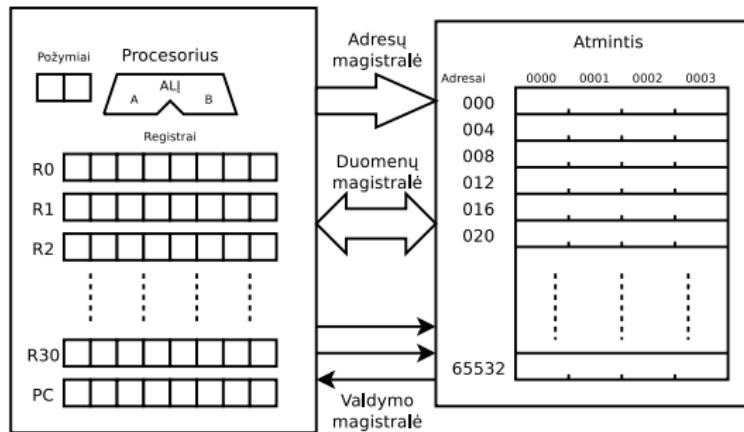
American Patriot Missile

[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:
$$0.1_{10} - 0.0001\ 1001\ 1001\ 1001\ 1000_2 = \\ 9.5 \times 10^{-10}$$
- Per 100 val. susikaupia paklaida:
$$9.5 \times 10^{-10} ds \cdot 10^{\frac{ds}{s}} \cdot 3600 \frac{s}{h} \cdot 100 h = \mathbf{0.34s}$$
- Per **0.34s** atskrendanti Scud raketa nukeliauja daugiau nei pusę kilometro ir “pasprunka” nuo “Patriot” sistemos...

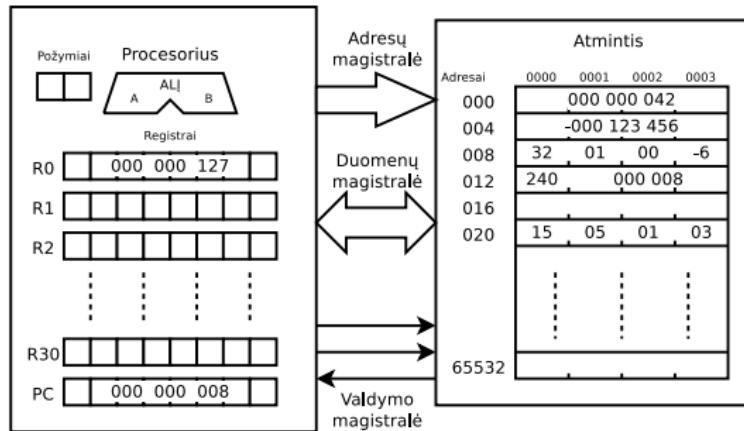
Apibendrinta kompiuterio schema

Fon Noimano (von Neumann) architektūra

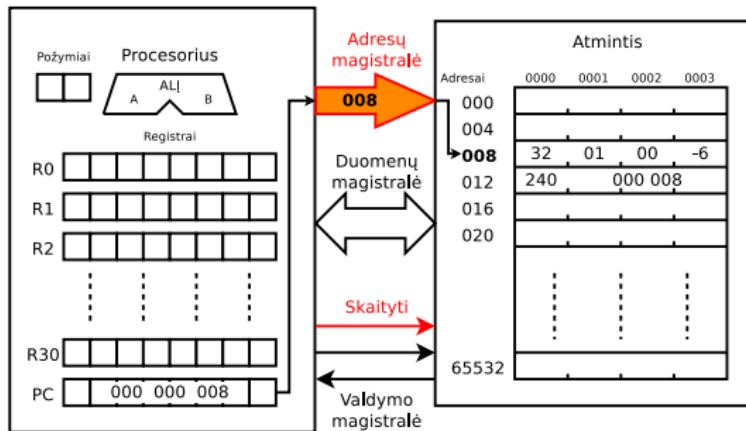


Apibendrinta kompiuterio schema

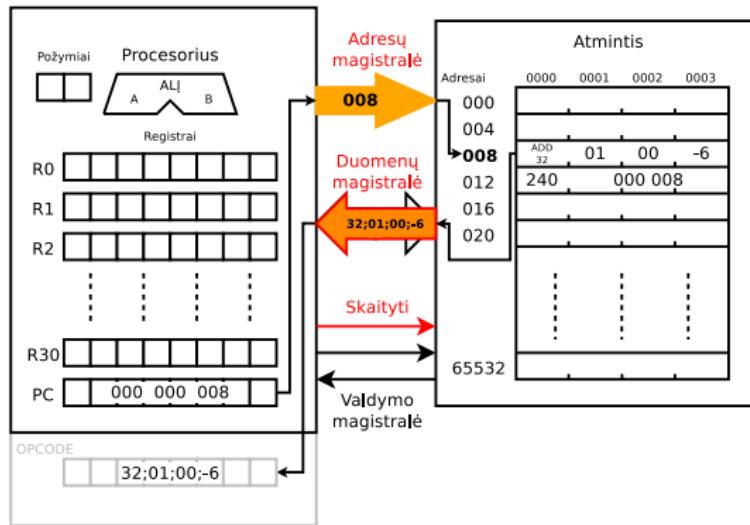
Fon Noimano (von Neumann) architektūra



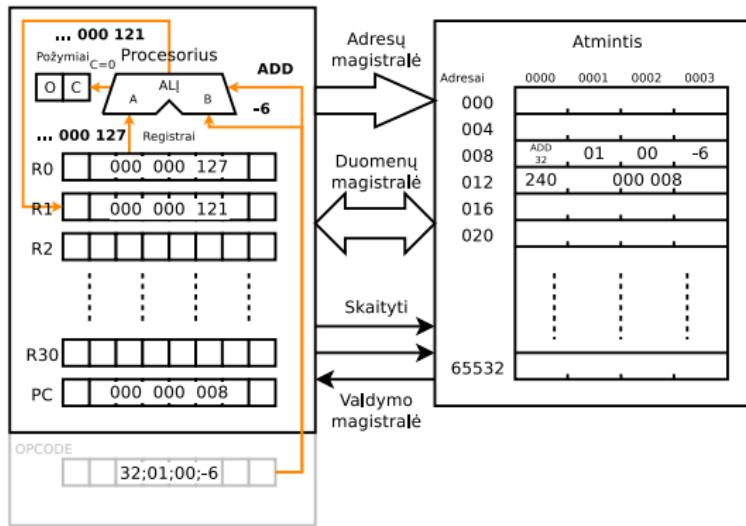
Komandos vykdymo ciklas



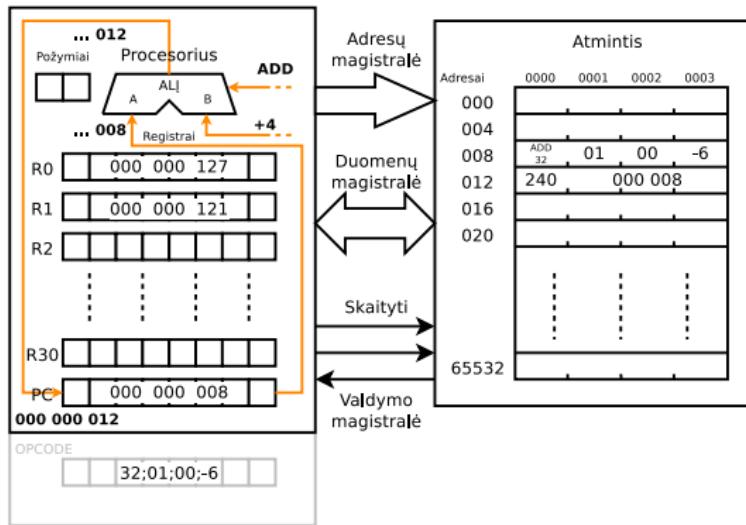
Komandos vykdymo ciklas



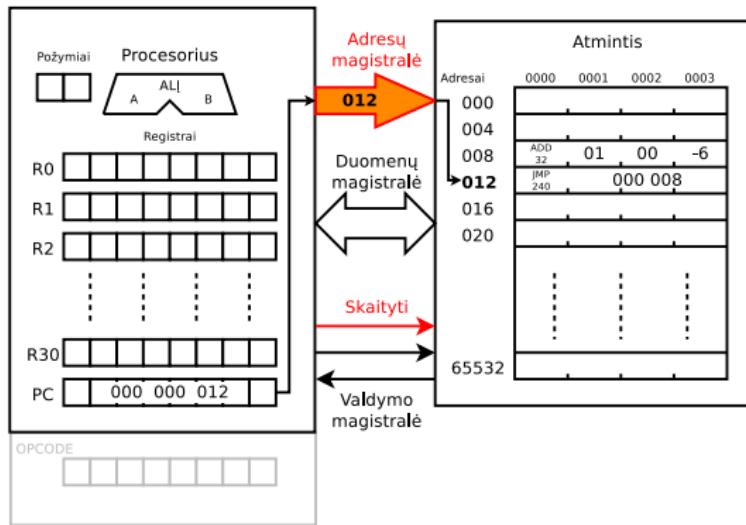
Komandos vykdymo ciklas



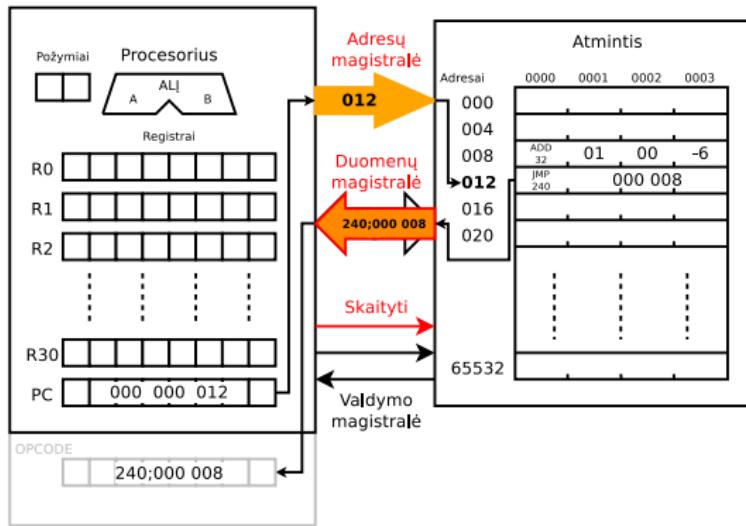
Komandos vykdymo ciklas



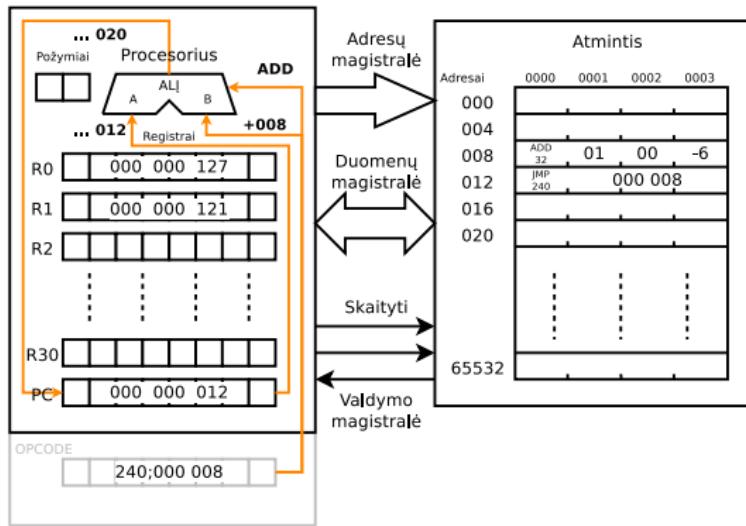
Komanda JMP



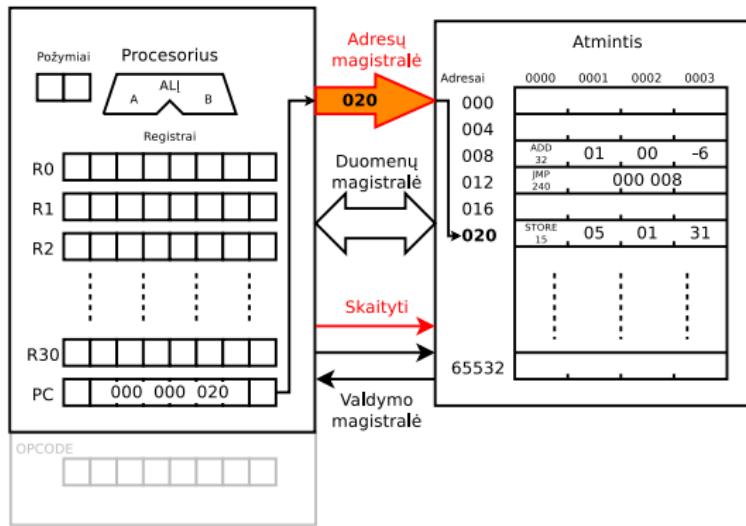
Komanda JMP



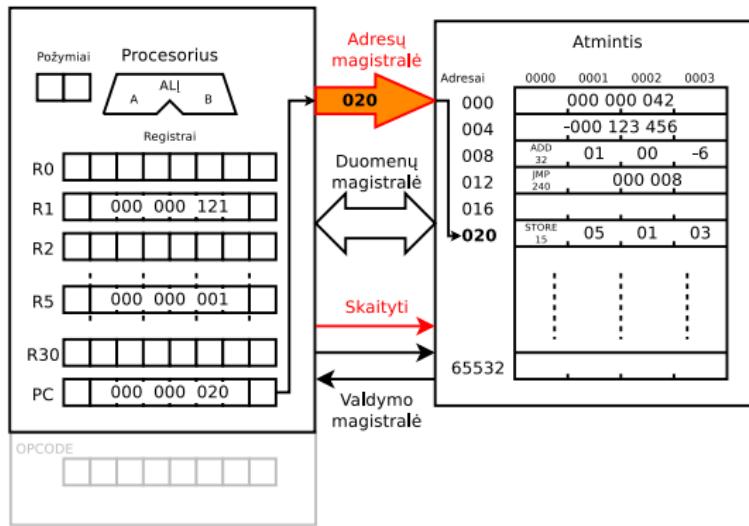
Komanda JMP



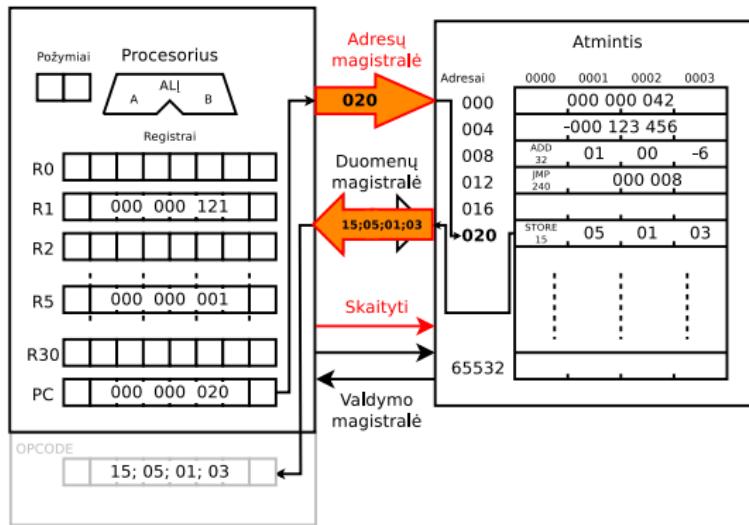
Komanda JMP



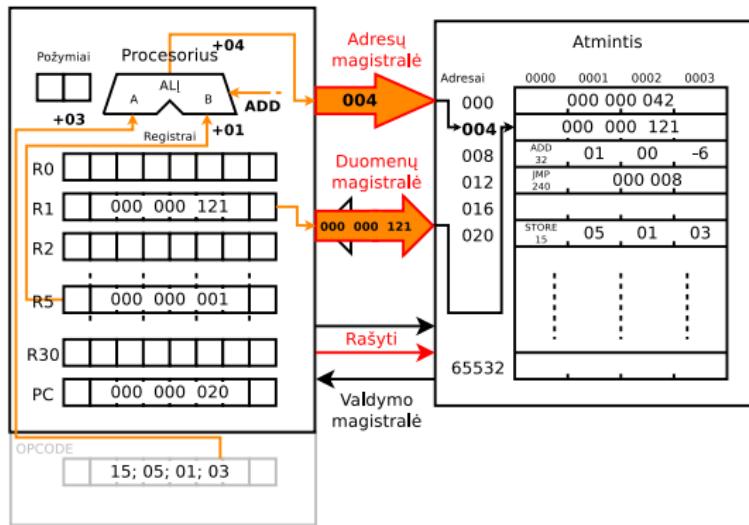
Komanda STORE



Komanda STORE

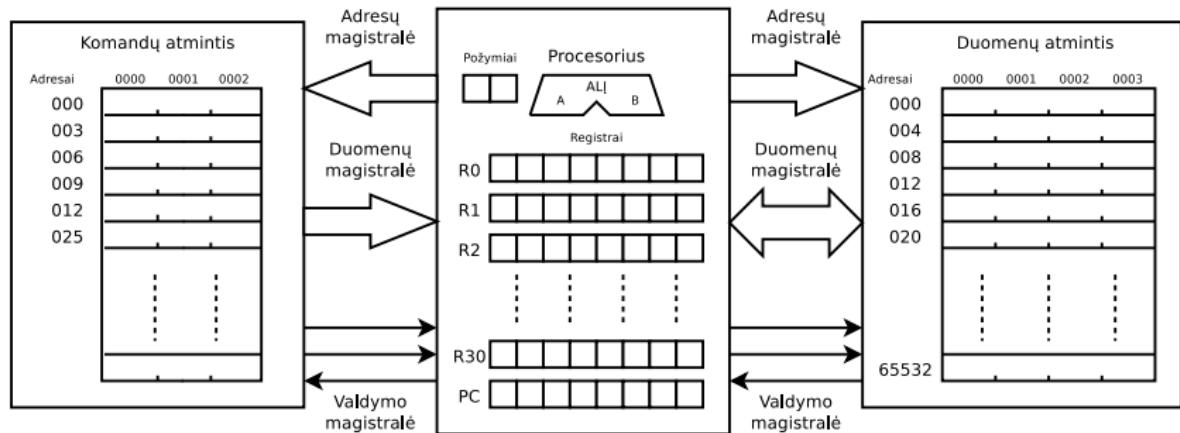


Komanda STORE



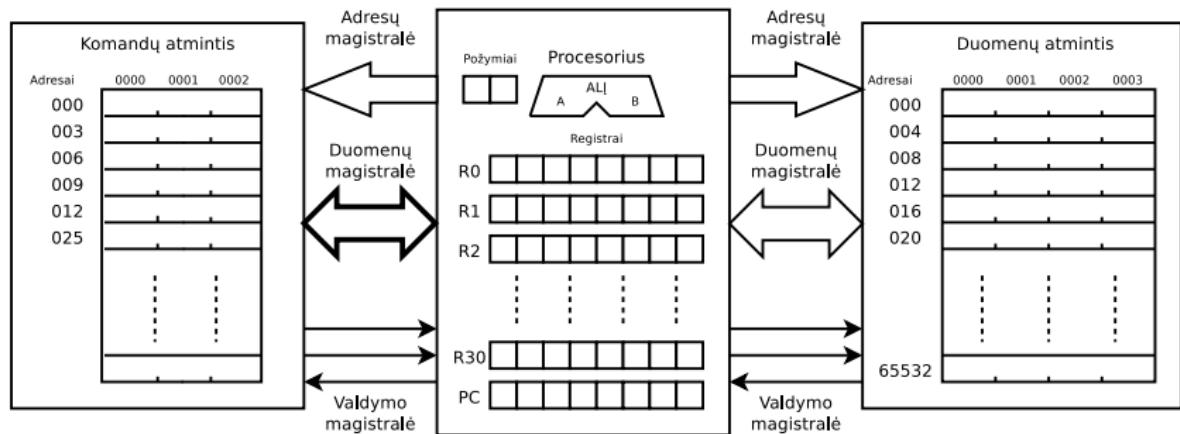
Harvardo architektūra

Klasikinė Harvardo architektūra



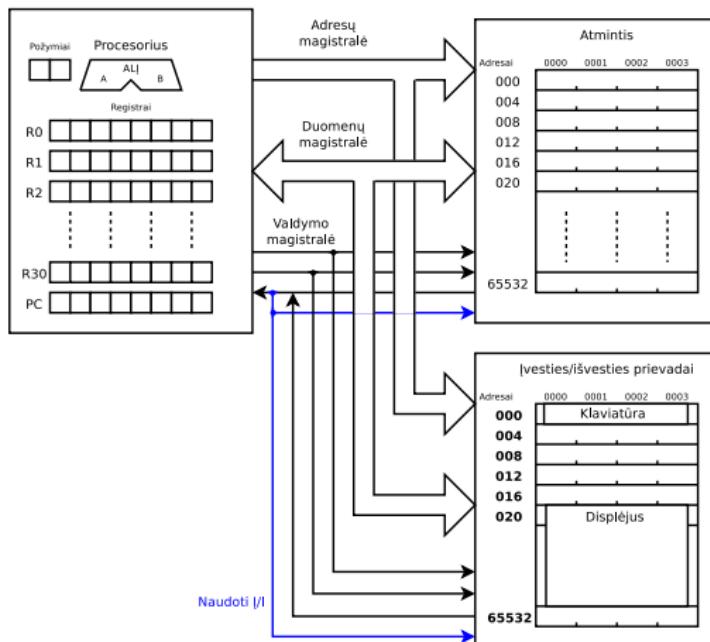
Harvardo architektūra

Modifikuota Harvardo architektūra



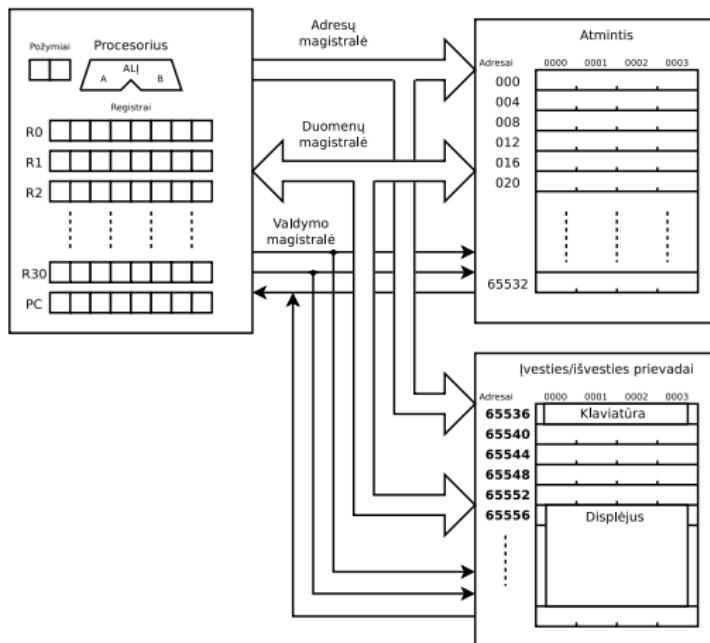
Ivestis ir išvestis

Atskiros atminties ir I/O adresų erdvės



Ivestis ir išvestis

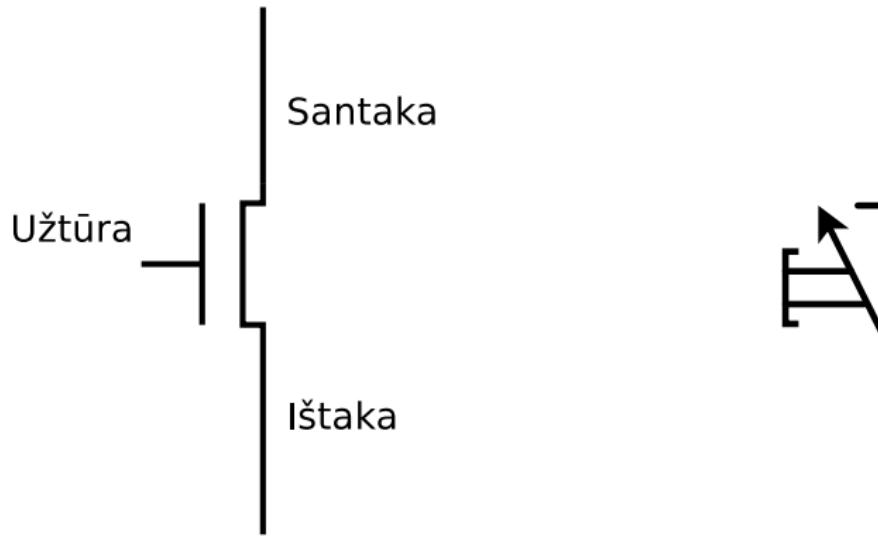
Bendra atminties ir I/I adresų erdvė



Valdomas perjungiklis – tranzistorius

FET – Field-effect transistor

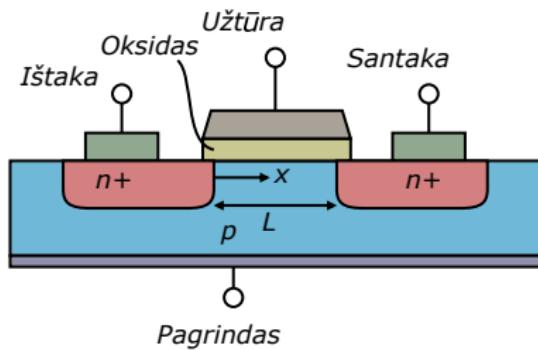
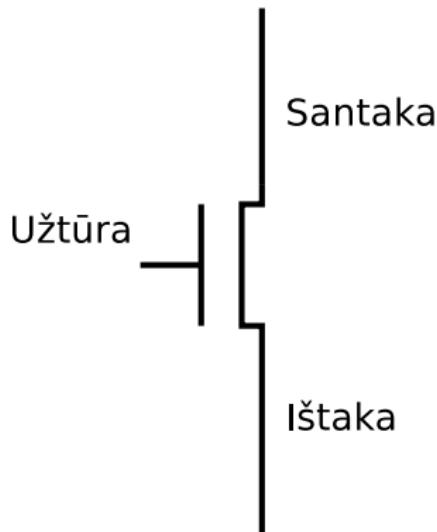
MOSFET – Metal-oxide-semiconductor field-effect transistor



Valdomas perjungiklis – tranzistorius

FET – Field-effect transistor

MOSFET – Metal–oxide–semiconductor field-effect transistor

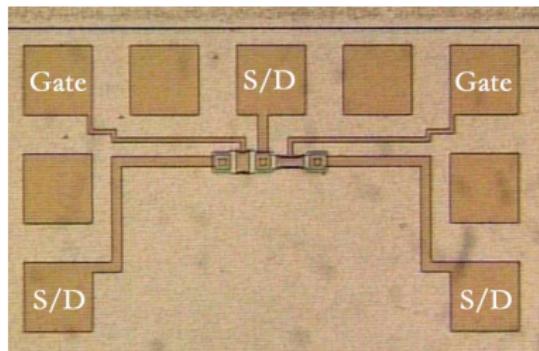
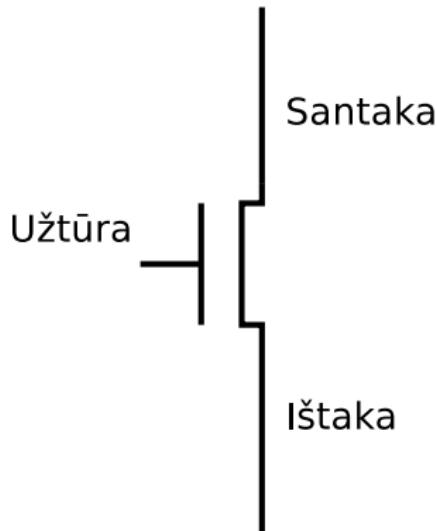


Cyril Buttay [CC BY-SA 3.0]

Valdomas perjungiklis – tranzistorius

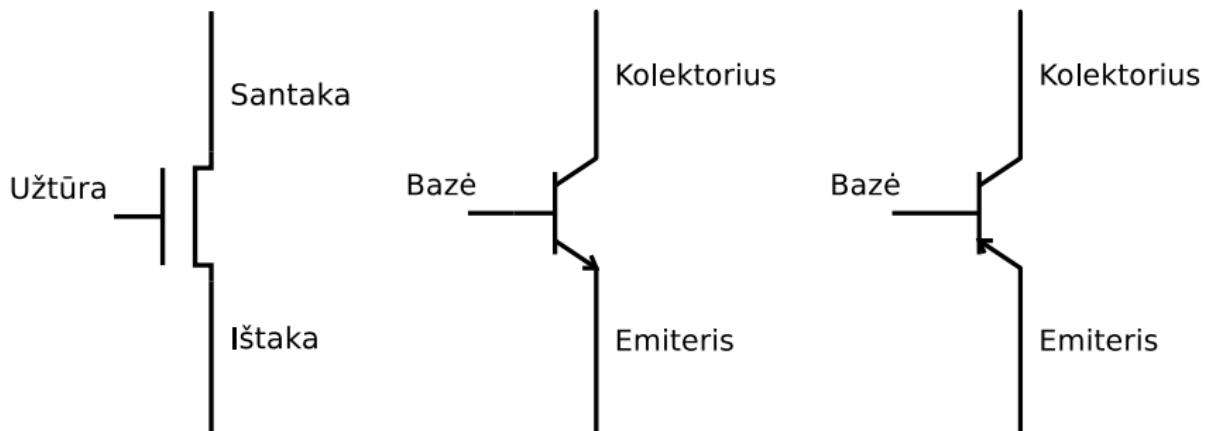
FET – Field-effect transistor

MOSFET – Metal–oxide–semiconductor field-effect transistor

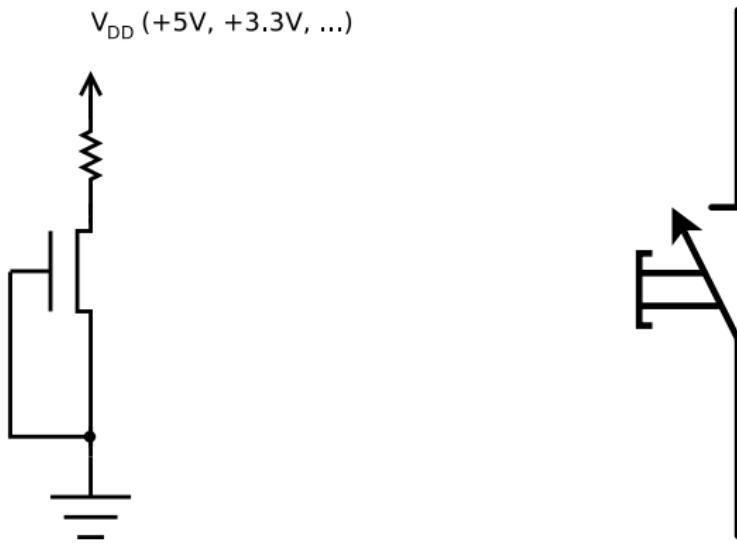


Dick Lyon [Public domain]

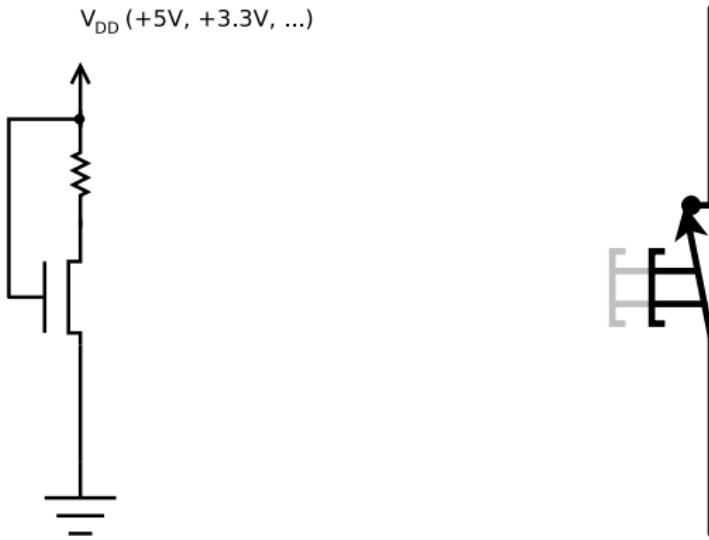
Lauko ir bipolariniai tranzistoriai



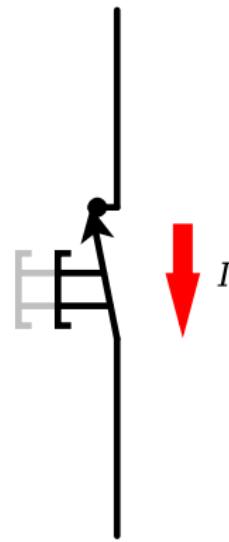
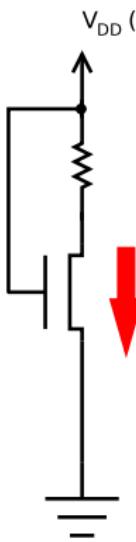
Tranzistoriaus savybės



Tranzistoriaus savybės



Tranzistoriaus savybės



Du tranzistorių tipai



n-tipo tranzistorius

\Leftrightarrow

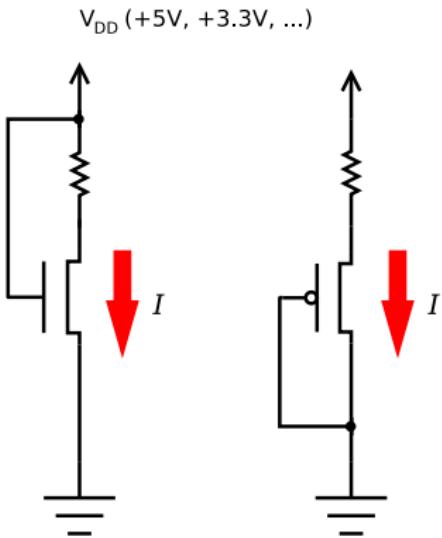


p-tipo tranzistorius

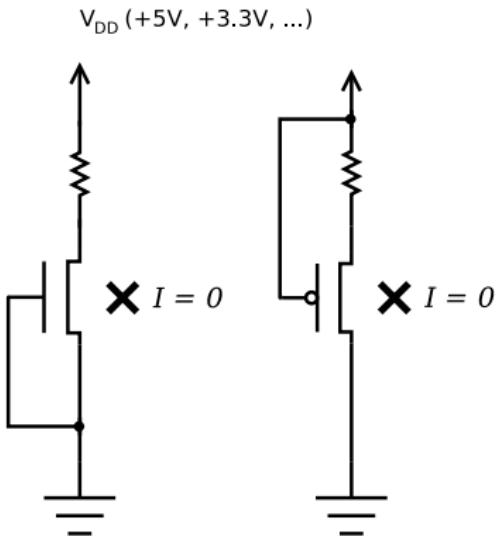
\Leftrightarrow



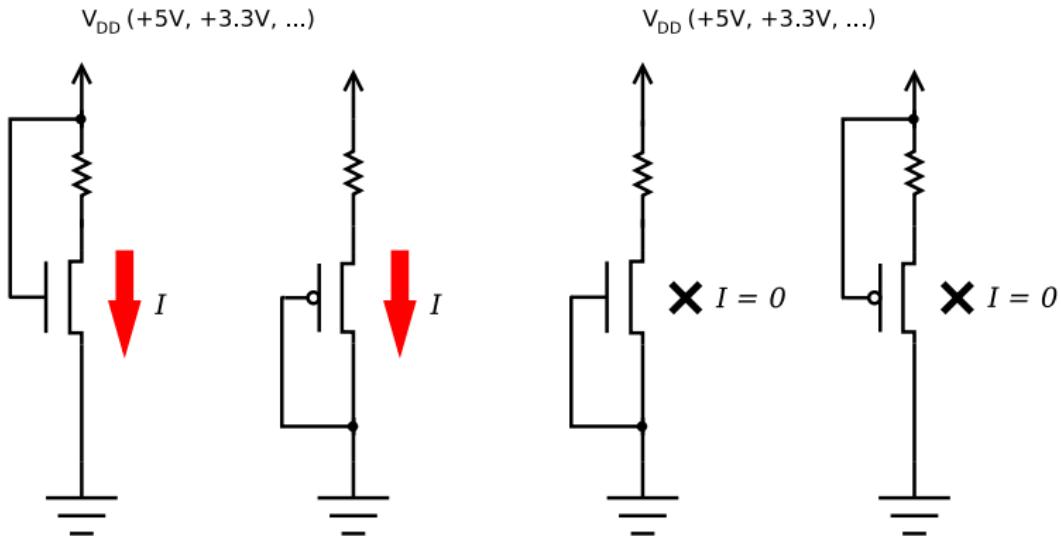
N-tipo ir p-tipo tranzistorių palyginimas



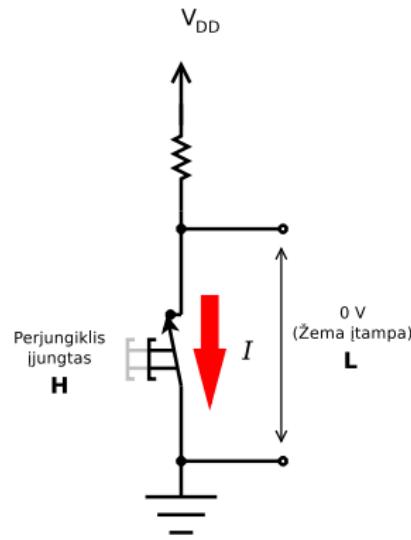
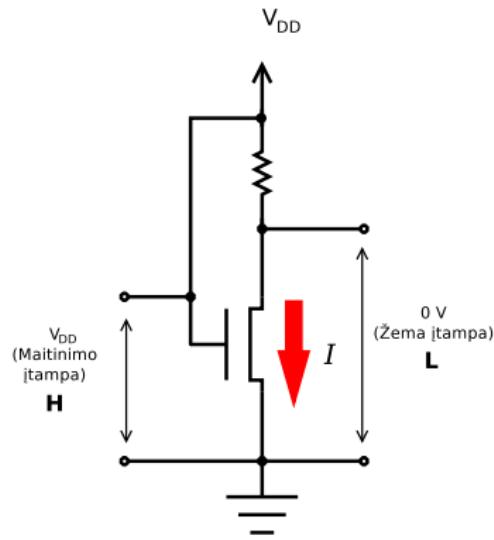
N-tipo ir p-tipo tranzistorių palyginimas



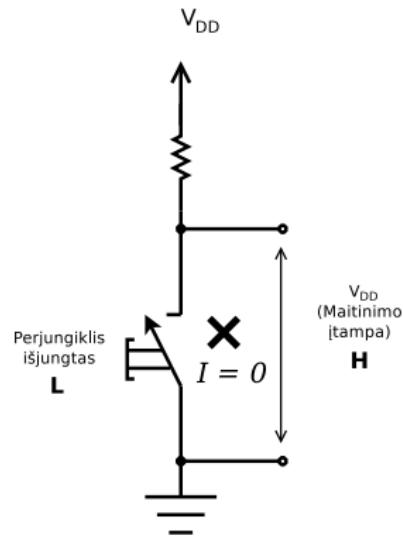
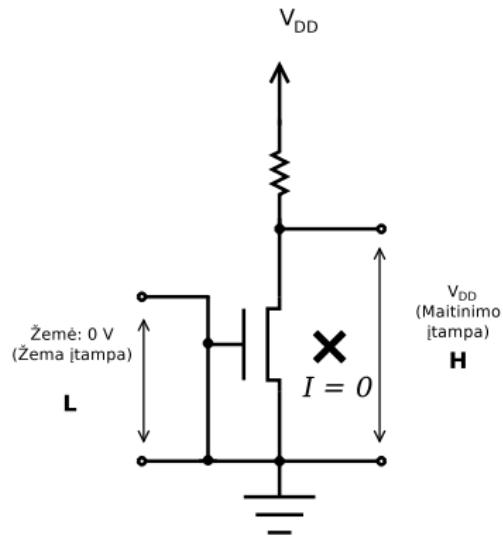
N-tipo ir p-tipo tranzistorių apžvalga



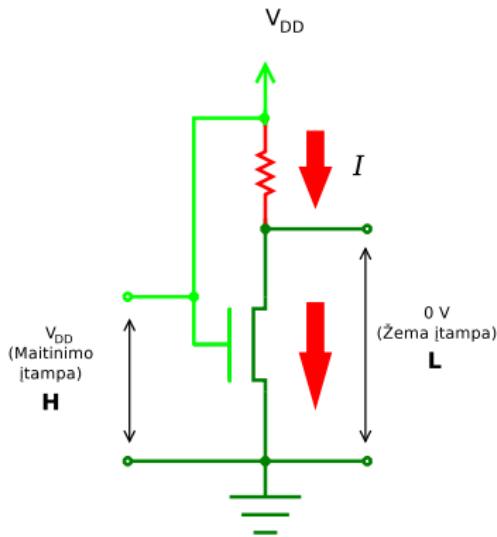
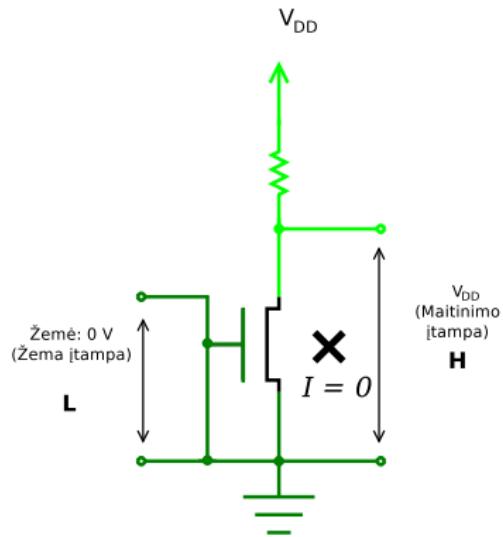
Tranzistorinių perjungiklių įtampos



Tranzistorinių perjungiklių įtampos



Priešingos tranzistorinio perjungiklio įtampos

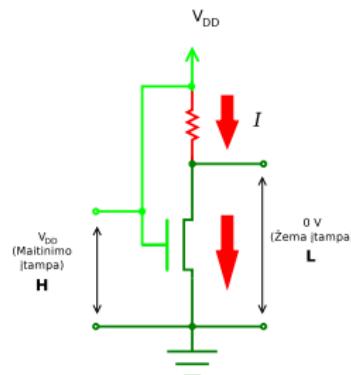
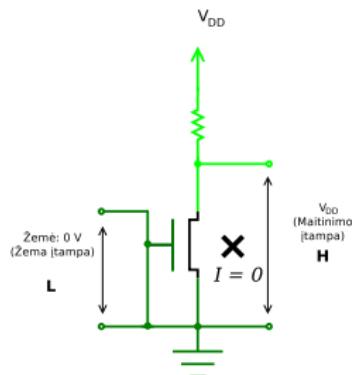


Invertorius (loginė f-ja NE/NOT)

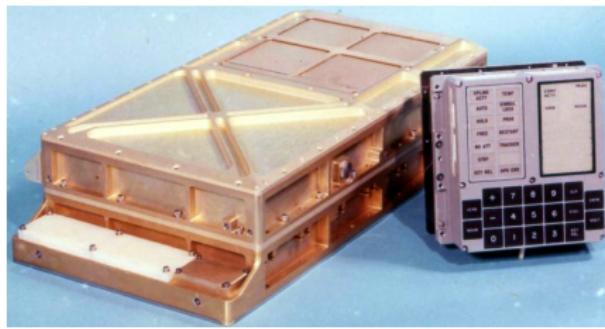
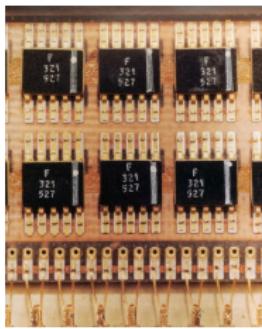
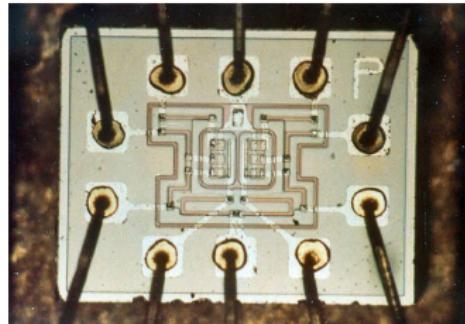
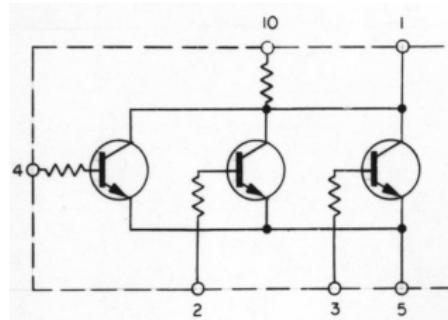
L: Žemės potencialas ($\approx 0V$)
H: Maitinimo įtampa (5V, 3.3V, ...)

L: 0
H: 1
L: False
H: True

Ivestis	Išvestis	Ivestis	Išvestis	Ivestis	Išvestis
L	H	0	1	False	True
H	L	1	0	True	False

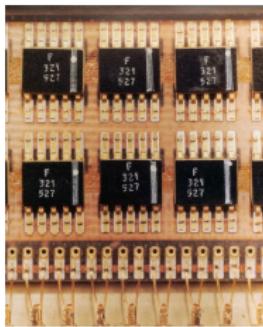
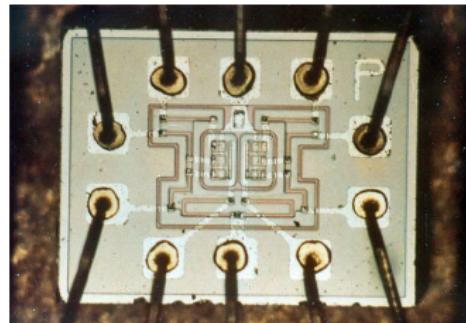
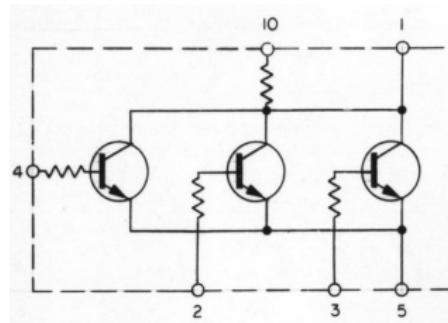


Istorinės loginės schemas: Apolono valdymo kompiuteris



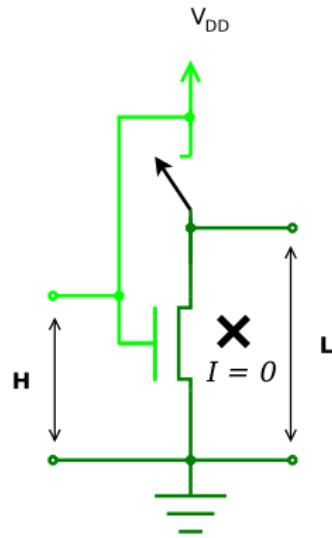
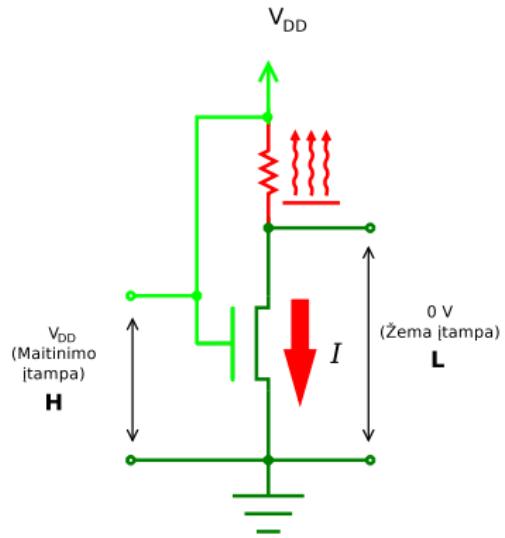
All AGC images: Wikipedia. [Apollo guidance computer](#), (viewed 2020-08-10). All images by NASA, public domain.

Istorinės loginės schemas: Apolono valdymo kompiuteris

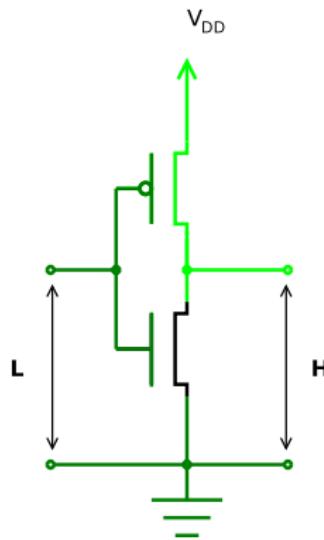
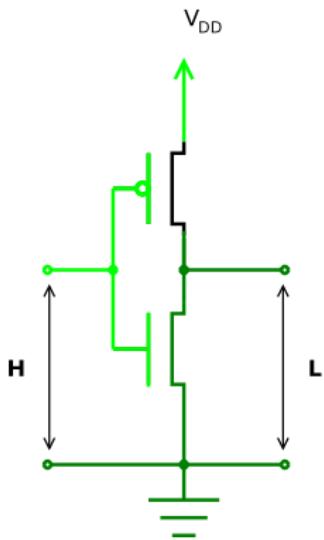


Lunar Landing Module image: by [Neil Armstrong](#) (viewed 2020-08-10). [Public domain]

Išsklaidyta galia

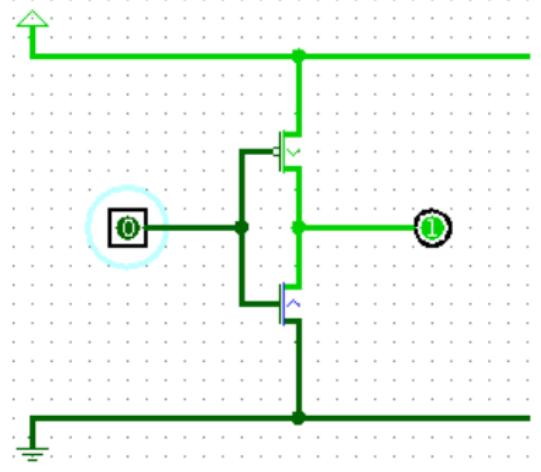
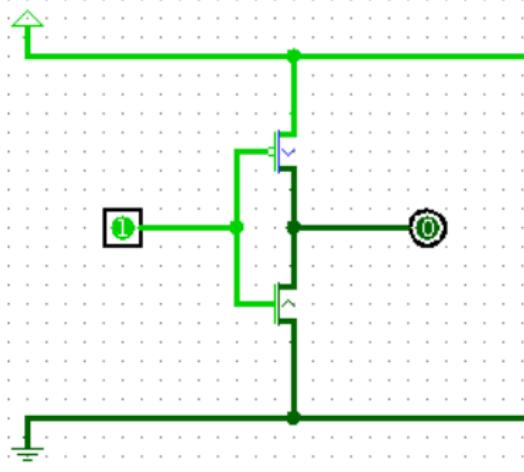


CMOS ventilis NE/NOT



NOT ventilis, sumodeliuotas Logisim

Logisim yra puiki programa skaitmeninių įrenginių konstravimui ir modeliavimui.



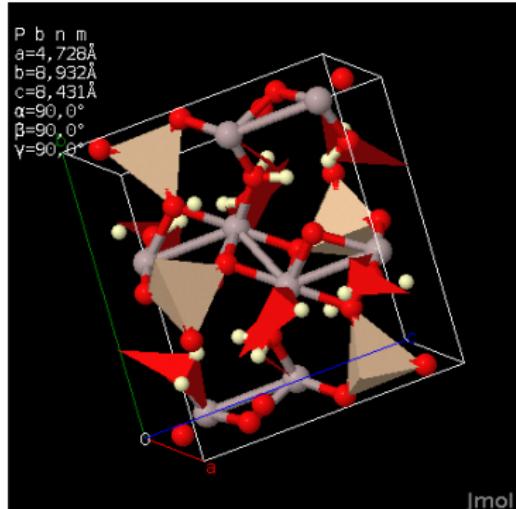
Apibendrinimas

- 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;
- Visus kompiuterio mazgus galima pagaminti iš valdomų perjungiklių; šiuolaikiniuose kompiuteriuose kaip perjungikliai naudojami KMOP lauko tranzistoriai.

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.