

# Architektura komputerów

wer. 17 z drobnymi modyfikacjami!

Wojciech Myszka

2023-10-30 16:18:48 +0100

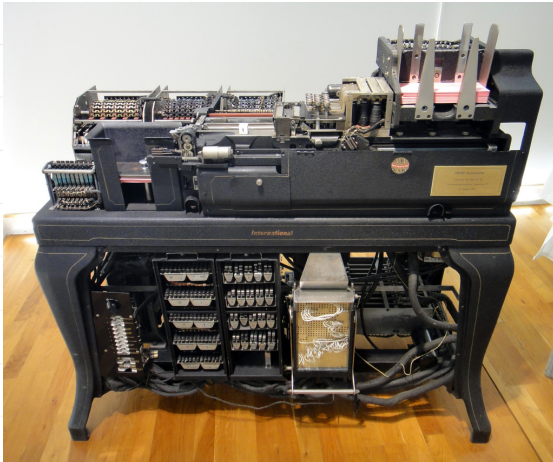


Politechnika Wroclawska

# Karty perforowane



# Kalkulator



IBM 601, 1931



## IBM 601 kalkulator

Maszyna czytała dwie, maksimum ośmiocyfrowe, liczby z karty, mnożyła je przez siebie i wynik dziurkowała na tej samej karcie. Potrafiła również dodawać i odejmować.

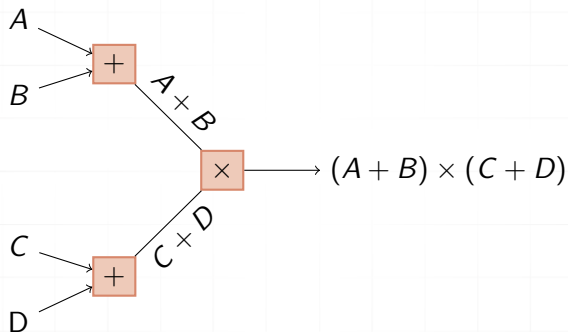
Nie mogła wyników drukować. W tym celu musiała „współpracować” z tabulatorem.

Model zainstalowany w laboratorium Eckerta posiadał funkcję interpolacji (specjalnie zaprojektowaną przez IBM). Właściciel (użytkownik) maszyny — Eckert połączył ją z innymi maszynami (Tabulator, Duplikator) za pomocą własnoręcznie zaprojektowanego elektrycznego łącznika tworząc pierwszy „kombajn” do złożonych obliczeń naukowych.

<http://www.columbia.edu/cu/computinghistory/601.html>



## Idea obliczeń



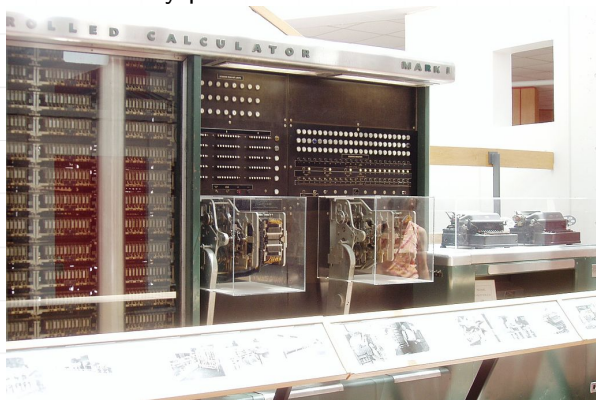
- ▶  $A, B, C, D$  — dane: „strumień” (plik) kart perforowanych
- ▶  $A + B, C + D$  — wyniki częściowe: plik kart perforowanych
- ▶  $(A + B)(C + D)$  — wynik, również plik kart perforowanych

**Rysunek:** Schemat obliczeń z wykorzystaniem maszyn wykonujących proste operacje na kartach perforowanych



# Pierwsze elektromechaniczne komputery. . .

. . . realizowały podobne idee



# John von Neumann

1. Jednym z najważniejszych i największych projektów obliczeniowych lat czterdziestych były obliczenia na potrzeby bomby atomowej.
2. Brał w ich udział **John von Neumann**, genialny matematyk i fizyk (który zajmował się również mechaniką kwantową, teorią gier, informatyką, analizą funkcjonalną...)



# Architektura von Neumanna

Architektura von Neumanna — rodzaj architektury komputera, przedstawionej po raz pierwszy w 1945 roku przez von Neumanna stworzonej wspólnie z Johnem W. Mauchly'ym i Johnem Presperem Eckertem Polega na podziale komputera na trzy podstawowe części:

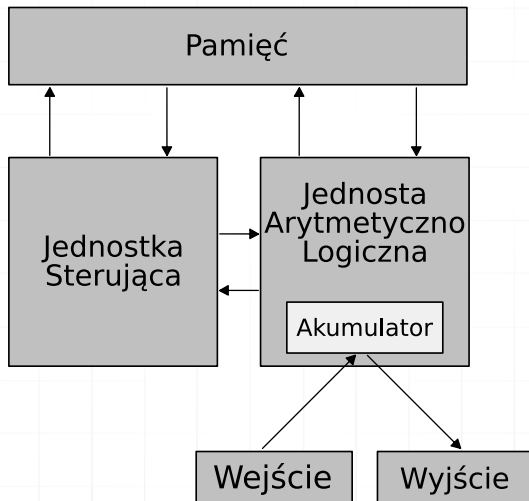
- ▶ procesor (w ramach którego wydzielona bywa część sterująca oraz część arytmetyczno-logiczna);
- ▶ pamięć komputera (zawierająca dane i sam program);
- ▶ urządzenia wejścia/wyjścia.

Rysunek: Schemat architektury von Neumanna





# Architektura von Neumanna



Rysunek: Schemat architektury von Neumanna



# Architektura von Neumanna

Rysunek: Schemat architektury von Neumanna

System komputerowy zbudowany w oparciu o architekturę von Neumanna powinien:

- ▶ mieć skończoną i funkcjonalnie pełną listę rozkazów;
- ▶ mieć możliwość wprowadzenia programu do systemu komputerowego poprzez urządzenia zewnętrzne i jego przechowywanie w pamięci w sposób identyczny jak danych;
- ▶ dane i instrukcje w takim systemie powinny być jednakowo dostępne dla procesora;
- ▶ informacja jest tam przetwarzana dzięki sekwencyjnemu odczytywaniu instrukcji z pamięci komputera i wykonywaniu tych instrukcji w procesorze.



# Architektura von Neumanna

Rysunek: Schemat architektury von Neumanna

Podane warunki pozwalają przełączać system komputerowy z wykonania jednego zadania (programu) na inne bez fizycznej ingerencji w strukturę systemu, a tym samym gwarantują jego uniwersalność.

System komputerowy von Neumanna nie posiada oddzielnych pamięci do przechowywania danych i instrukcji. Instrukcje jak i dane są zakodowane w postaci liczb (binarnych). Bez analizy programu trudno jest określić czy dany obszar pamięci zawiera dane czy instrukcje. Wykonywany program może się sam modyfikować traktując obszar instrukcji jako dane, a po przetworzeniu tych instrukcji — danych — zacząć je wykonywać.



# Schemat komputera



Rysunek: Schemat komputera nawiązujący do architektury von Neumanna



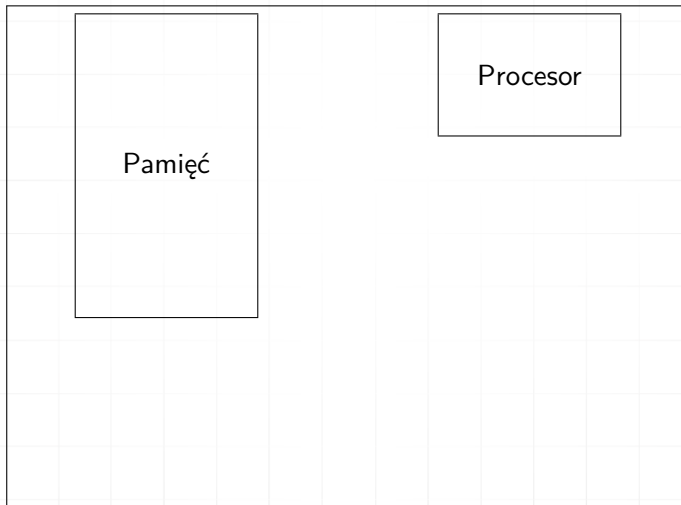
# Schemat komputera



Rysunek: Schemat komputera nawiązujący do architektury von Neumanna



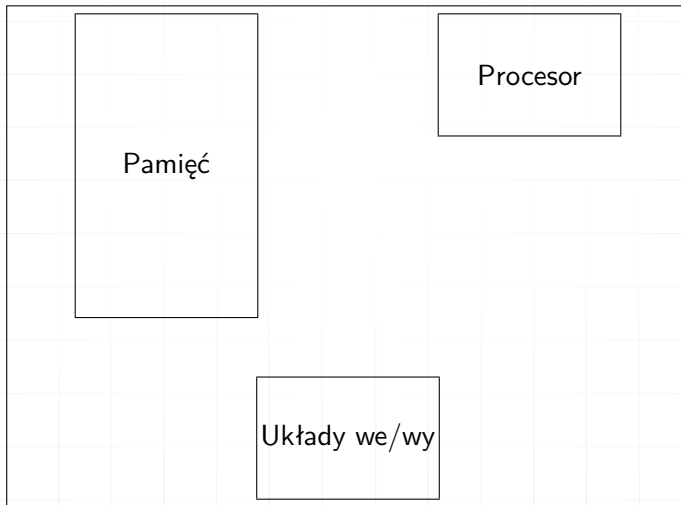
# Schemat komputera



Rysunek: Schemat komputera nawiązujący do architektury von Neumanna



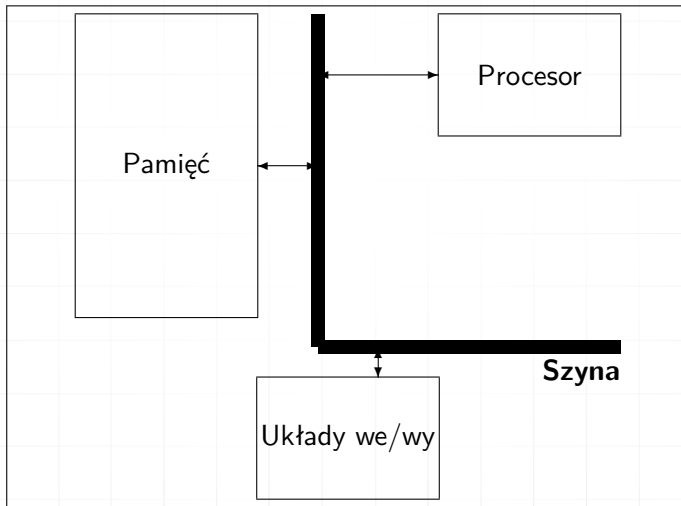
# Schemat komputera



Rysunek: Schemat komputera nawiązujący do architektury von Neumanna



# Schemat komputera

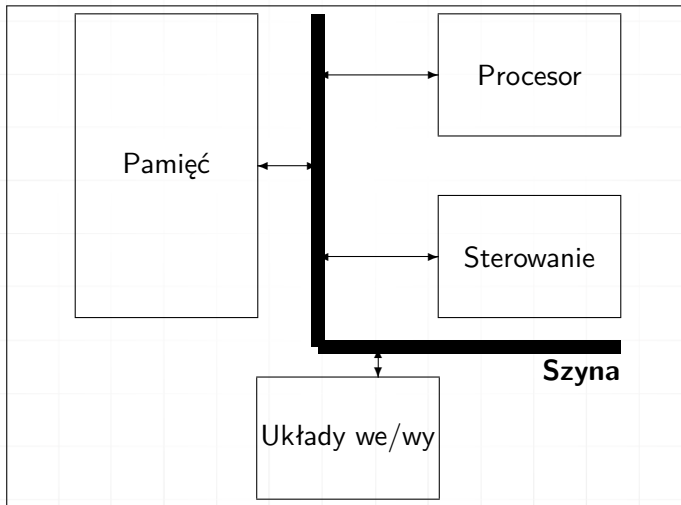


Rysunek: Schemat komputera nawiązujący do architektury von Neumanna





# Schemat komputera



Rysunek: Schemat komputera nawiązujący do architektury von Neumanna



# Schemat komputera I

W modelu tym wyróżniamy:

- ▶ Procesor.
- ▶ Pamięć (i pod tym pojęciem rozumiemy wszystkie rodzaje pamięci: Cache, RAM, ROM, dyski, dyskietki, dyski wymienne — zapisywalne i nie).
- ▶ Urządzenie wejścia/wyjścia (wszystkie urządzenia pozwalające na kontakt ze „światem zewnętrznym”: klawiatura, mysz, karta graficzna, drukarka, czytniki różnego rodzaju).
- ▶ Sterowanie (wszystkie układy elektroniczne zapewniające właściwą komunikację wszystkich wymienionych wyżej urządzeń ze sobą i zapewniający uporządkowany przepływ informacji po szynie/szynach).
- ▶ Magistrala (szyna) — to wszystkie „drogi” łączące wymienione wyżej urządzenia.



## Co to jest komputer

Zanim przejdziemy dalej zastanówmy się co to jest komputer.

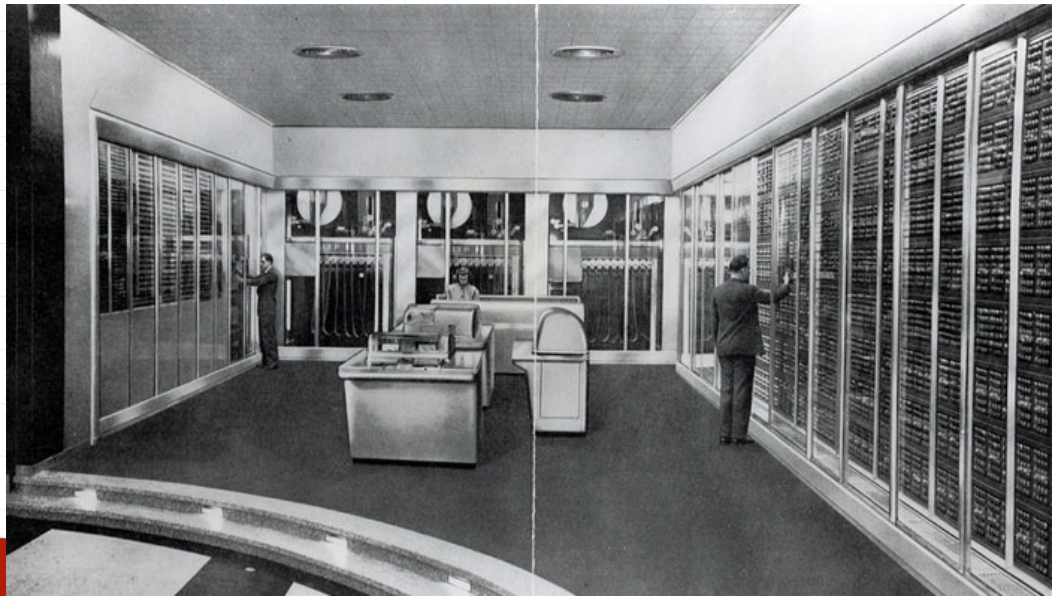


No właśnie...

...co to jest komputer?



???



???



???



???





???



???



???



???

# Sinclair ZX81 Personal Computer - the heart of a system that grows with you.

1980 saw a genuine breakthrough - the Sinclair ZX80, world's first complete personal computer for under £100. Not surprisingly, over 50,000 were sold.

In March 1981, the Sinclair lead increased dramatically. For just £89.95 the Sinclair ZX81 offers even more advanced facilities at an even lower price. Initially, even we were surprised by the demand - over 50,000 in the first 3 months!

Today, the Sinclair ZX81 is the heart of a computer system. You can add 16 times more memory with the ZX RAM pack. The ZX Printer offers an unbeatable combination of performance and price. And the ZX Software Library is growing every day.

**Lower price, higher capability**  
With the ZX81, it's still very simple to teach yourself computing, but the ZX81 packs even greater working capability than the ZX80.

It uses the same micro-processor, but incorporates a new, more powerful 8K BASIC ROM - the 'trained intelligence' of the computer. This chip works in decimals, handles signs and trig, allows you to plot graphs, and builds up animated displays.

And the ZX81 incorporates other operation refinements - the facility to load and save named programs on cassette, for example, and to drive the new ZX Printer.



New BASIC manual

Manuals: ZX81 comes with 200 computer programs. Complete system manual. Additional manuals available for extra programming. 19 ZX80 chips.

**Kit: £49.95**

**Higher specification, lower price - how's it done?**

Quite simply, by design. The ZX80 reduced the chips in a working computer from 40 or so, to 21. The ZX81 reduces the 21 to 4!

The secret lies in a totally new master chip. Designed by Sinclair and custom-built in Britain, this unique chip replaces 18 chips from the ZX80.

**New, improved specification**

● ZX81 micro-processor - new faster version of the famous Z80 chip, widely recognised as the best ever made.

● Unique 'one-touch' key word entry: the ZX81 eliminates a great deal of tedious typing. Key words (RUN, LIST, FILES, etc.) have their own single-key entry.

● Unique syntax-check and report codes identify programming errors immediately.

● Full range of mathematical and scientific functions accurate to eight decimal places.

● Graph-drawing and animated-display facilities.

● Multi-dimensional string and numerical arrays.

● Up to 26 PC/NEXT loops.

● Randomise function - useful for games as well as serious applications.

**Built: £69.95**

**Kit or built - it's up to you!**

You'll be surprised how easy the ZX81 kit is to build: just four chips to assemble (plus, of course the other discrete components) - a few hours' work with a fine-tipped soldering iron. And you may already have a suitable mains adaptor - 400mA at 9V DC nominal unregulated (supplied with built version).

Kit and built versions come complete with all leads to connect to your TV (colour or black and white) and cassette recorder.



**16K-byte RAM pack for massive add-on memory.**

Designed as a complete module to fit your Sinclair ZX80 or ZX81, the RAM pack simply plugs into the existing expansion port at the rear of the computer to multiply your data/program storage by 16. Use it for long and complex programs or as a personal database. Yet it costs as little as half the price of competitive additional memory.

With the RAM pack, you can also run some of the more sophisticated ZX Software - the Business & Household management systems for example.

**sinclair ZX81**

4 Kings Parade, Cambridge, Cambs, CB2 1PR. Tel: 0223 55164 x 2102.

**Available now - the ZX Printer - for only £49.95**

Designed exclusively for use with the ZX81 (and ZX80) with 8K BASIC ROM, the printer offers full alphanumeric and highly sophisticated graphics.

A special feature is COPY, which prints out exactly what is on the whole TV screen without the need for further instructions.

**How to order your ZX81 BY PHONE** - Access, Barclaycard or Trustcard holders can call 01 200 0306 for personal attention 24 hours a day, every day. **BY FREEPOST** - use the no-stamp-needed coupon below. You can pay...

At last you can have a hard copy of your program listings - particularly useful when writing or editing programs.

And of course you can print out your results for permanent records or sending to a friend.

Printing speed is 50 characters per second, with 32 characters per line and 8 lines per vertical inch.

The ZX Printer connects to the rear of your computer - using a stackable connector to you can plug in a RAM pack as well. A roll of paper (50 ft long x 4 in wide) is supplied, along with full instructions.

by cheque, postal order, Access, Barclaycard or Trustcard. EITHER WAY - please allow up to 28 days for delivery. And there's a 14-day money-back option. We want you to be satisfied beyond doubt - and we have no doubt that you will be.

To: Sinclair Research, FREEPOST, Cambridge, Cambs, CB2 1PR.		Order	
City	Item	Code	Total
	Sinclair ZX81 Personal Computer kit, Price includes 200 BASIC programs, manual and mains adaptor	12	69.95
	16K RAM pack	13	39.95
	ZX Printer	14	49.95
	200 BASIC manual	15	99.95
	Basic Adaptor (9V at 400mA) (UK nominal unregulated)	16	59.95
	16K 9V 1A pack	17	49.95
	Sinclair ZX Printer	18	49.95
	Roll of Paper	19	19.95
	Post and Packing		1.00
			<b>TOTAL £</b>
<input type="checkbox"/> Please bill if you require a VAT receipt <input type="checkbox"/> Enclose a cheque/postal order payable to Sinclair Research Ltd, for £			
*Please charge to my Access/Barclaycard/Trustcard account no.			
Name: Mr/Ms/Ms/Ms _____			
Address: _____			
Postcode: _____			
<input type="checkbox"/> FREEPOST - no stamp needed			

???



???



???



???

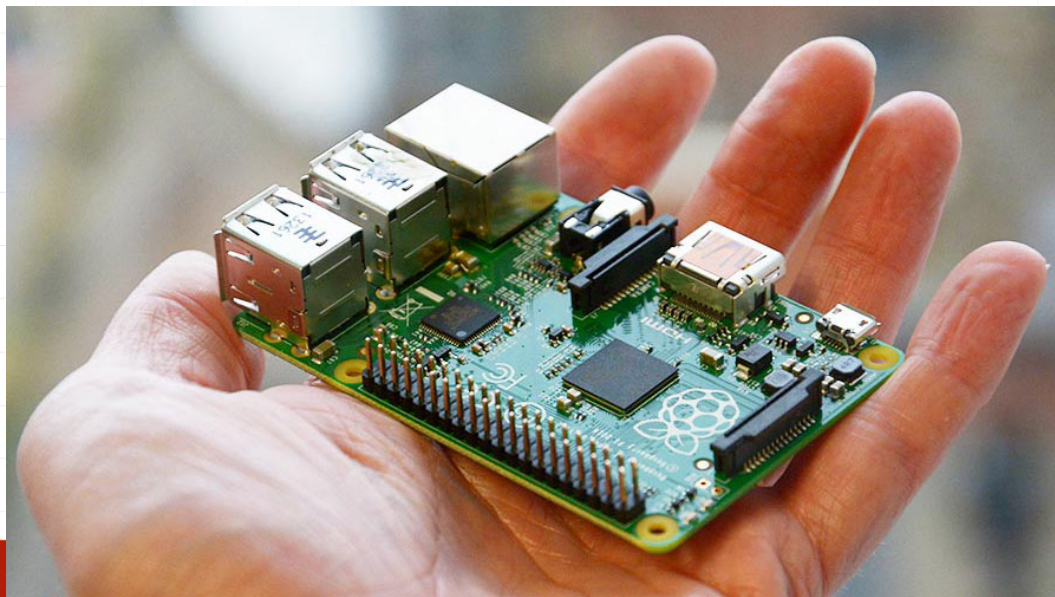




???



???



???



???



# Kalkulator

Wyświetlacz			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷



# Kalkulator

Wyświetlacz			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- Wyobraźmy sobie kalkulator. Taki najprostszy, czterodziałaniowy.



# Kalkulator

Wyświetlacz			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Wyobraźmy sobie kalkulator. Taki najprostszy, czterodziałaniowy.
- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.



# Kalkulator

Wyświetlacz			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Wyobraźmy sobie kalkulator. Taki najprostszy, czterodziałaniowy.
- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu





# Kalkulator

			1
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Wyobraźmy sobie kalkulator. Taki najprostszy, czterodziałaniowy.
- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu



# Kalkulator

12			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Wyobraźmy sobie kalkulator. Taki najprostszy, czterodziałaniowy.
- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu



# Kalkulator

123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Wyobraźmy sobie kalkulator. Taki najprostszy, czterodziałaniowy.
- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu



# Kalkulator

123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Wyobraźmy sobie kalkulator. Taki najprostszy, czterodziałaniowy.
- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu
- ▶ I tu pojawia się podejrzenie, że wprowadzana liczba jest gdzieś zapamiętywana. Jak oderwiemy palec od klawiatury — wartości nie znikają. Pamięć ta jest (jakoś) powiązana z wyświetlaczem (i klawiaturą).



# Kalkulator

123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Wyobraźmy sobie kalkulator. Taki najprostszy, czterodziałaniowy.
- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu
- ▶ I tu pojawia się podejrzenie, że wprowadzana liczba jest gdzieś zapamiętywana. Jak oderwiemy palec od klawiatury — wartości nie znikają. Pamięć ta jest (jakoś) powiązana z wyświetlaczem (i klawiaturą).
- ▶ Musimy zmodyfikować nasz schemat.



# Kalkulator

Wyświetlacz			
Akumulator			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.



# Kalkulator

Wyświetlacz			
Akumulator			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Nie widać pamięci nazwanej tu akumulatorem, ale chyba ona tam jest.



# Kalkulator

Wyświetlacz			
Akumulator			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Nie widać pamięci nazwanej tu akumulatorem, ale chyba ona tam jest.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu i w akumulatorze.





# Kalkulator

1			
1			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Nie widać pamięci nazwanej tu akumulatorem, ale chyba ona tam jest.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu i w akumulatorze.



# Kalkulator

12			
12			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Nie widać pamięci nazwanej tu akumulatorem, ale chyba ona tam jest.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu i w akumulatorze.



# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Nie widać pamięci nazwanej tu akumulatorem, ale chyba ona tam jest.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu i w akumulatorze.



# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Nie widać pamięci nazwanej tu akumulatorem, ale chyba ona tam jest.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu i w akumulatorze.
- ▶ Naciskamy klawisz operacji. Niech to będzie +



# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na pierwszy rzut oka widać klawiaturę (z cyframi i podstawowymi klawiszami operacji) oraz wyświetlacz.
- ▶ Nie widać pamięci nazwanej tu akumulatorem, ale chyba ona tam jest.
- ▶ Gdy zaczynamy naciskać klawisze cyfr (123) odpowiednie informacje pojawiają się na wyświetlaczu i w akumulatorze.
- ▶ Naciskamy klawisz operacji. Niech to będzie +



# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na wyświetlaczu nic się nie zmieniło, ale zmieniło się zachowanie kalkulatora: kolejne wprowadzane wartości powodują skasowanie i zastąpienie wyświetlanej liczby na wyświetlaczu.



# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na wyświetlaczu nic się nie zmieniło, ale zmieniło się zachowanie kalkulatora: kolejne wprowadzane wartości powodują skasowanie i zastąpienie wyświetlanej liczby na wyświetlaczu.
- ▶ Ale pierwsza wprowadzona wartość nie „ginie”. Jest gdzieś zapamiętana i będzie użyta w operacji (dodawania).



# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Na wyświetlaczu nic się nie zmieniło, ale zmieniło się zachowanie kalkulatora: kolejne wprowadzane wartości powodują skasowanie i zastąpienie wyświetlanej liczby na wyświetlaczu.
- ▶ Ale pierwsza wprowadzona wartość nie „ginie”. Jest gdzieś zapamiętana i będzie użyta w operacji (dodawania).
- ▶ Potrzebna jest modyfikacja — musimy dodać kolejną pamięć. Akumulator wykorzystywany będzie podczas wprowadzania danych z klawiatury i do wyświetlania wyników. Liczba tam zawarta zawsze będzie jednym z argumentów operacji dwuargumentowych.





# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷

- ▶ Ale pierwsza wprowadzona wartość nie „ginie”. Jest gdzieś zapamiętana i będzie użyta w operacji (dodawania).
- ▶ Potrzebna jest modyfikacja — musimy dodać kolejną pamięć. Akumulator wykorzystywany będzie podczas wprowadzania danych z klawiatury i do wyświetlania wyników. Liczba tam zawarta zawsze będzie jednym z argumentów operacji dwuargumentowych.
- ▶ Dodatkowa pamięć przechowywać będzie drugi z argumentów.



# Kalkulator

Wyświetlacz			
Akumulator			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷
Pamięć			

Teraz widać już wszystkie (widoczne i nie) elementy składowe kalkulatora.



# Kalkulator

Wyświetlacz			
Akumulator			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷
Pamięć			

Operacje



# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷
Pamięć			

Operacje  
123



# Kalkulator

123			
123			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷
123			

Operacje

123

+



# Kalkulator

55			
55			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷
123			

Operacje

123

+

55



# Kalkulator

178			
178			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷
123			

Operacje

123

+

55

+



# Kalkulator

22			
22			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷
178			

Operacje

123

+

55

+

22





# Kalkulator

200			
200			
1	2	3	+
4	5	6	-
7	8	9	*
0	C	=	÷
178			

Operacje

123

+

55

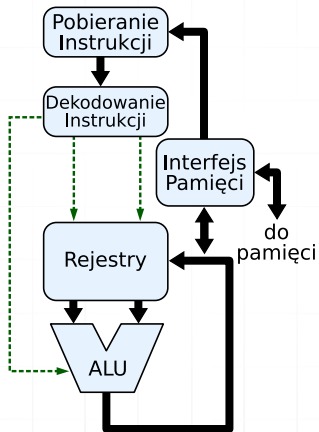
+

22

=



# Processor



Rysunek: Schemat działania procesora



# Podstawowe operacje

## Instrukcje arytmetyczne

- ▶ Ładuj  $\langle \textit{adres pamieci} \rangle$  przepisuje zawartość pamięci o wskazanym adresie do rejestru.
- ▶ Zapisz  $\langle \textit{adres pamieci} \rangle$  przepisuje zawartość akumulatora do pamięci
- ▶ Ładuj  $\langle \textit{liczba} \rangle$  zapisuje liczbę do rejestru
- ▶ Dodaj  $\langle \textit{adres pamieci} \rangle$  do zawartości akumulatora dodaje zawartość komórki o wskazanym adresie (możemy też założyć, że w podobny sposób potrafi policzyć różnicę, iloczyn i iloraz, choć, w rzeczywistości, nie musi to być prawdą).  
Wykonanie każdej operacji zmieniającej zawartość rejestru powoduje ustawienie wskaźników (zero, przepełnienie, ujemne).



# Podstawowe operacje

## Instrukcje na bitach

- ▶ Neguj zmienia znak liczby w akumulatorze
- ▶ And *<adres pamięci>* iloczyn logiczny (bit po bicie dwu słów)
- ▶ Or *<adres pamięci>*
- ▶ Xor *<adres pamięci>* — różnica symetryczna
- ▶ Przesun\_w\_lewo
- ▶ Przesun\_w\_prawo
- ▶ Przesun\_cyklicznie\_w\_lewo
- ▶ Przesun\_cyklicznie\_w\_prawo



# Podstawowe operacje

## Instrukcje sterujące

- ▶ `Skocz <adres pamięci>` bezwarunkowe przekazanie sterowanie do adresu
- ▶ `Skocz_jezeli_zero <adres pamięci>`
- ▶ `Skocz_jezeli_ujemne <adres pamięci>`
- ▶ `Skocz_jesli_nadmiar <adres pamięci>`
- ▶ `Skocz_do_podprogramu <adres pamięci>` bardzo podobne do instrukcji zwykłego skoku, ale dodatkowo zapisuje aktualny stan procesora w specjalnie do tego przeznaczonej pamięci



# Asembler

Bardzo proste działanie:

$A=B+C$



# Asembler

Bardzo proste działanie:

$$A=B+C$$

W komórce o adresie A ma być umieszczony wynik dodawania zawartości komórek o adresie B i C.



# Asembler

Bardzo proste działanie:

$$A=B+C$$

W komórce o adresie A ma być umieszczony wynik dodawania zawartości komórek o adresie B i C.

Realizacja komputerowa:

Ładuj B

Dodaj C

Zapisz A





# Asembler

Bardziej skomplikowany przykład

$$Z = \frac{[(A + B)(C + D)]}{W}$$



# Asembler

Bardziej skomplikowany przykład

$$Z = \frac{[(A + B)(C + D)]}{W}$$

$$T1 = A + B$$



# Asembler

Bardziej skomplikowany przykład

$$Z = \frac{[(A + B)(C + D)]}{W}$$

$$T1 = A + B$$

$$T2 = C + D$$



# Asembler

Bardziej skomplikowany przykład

$$Z = \frac{[(A + B)(C + D)]}{W}$$

$$T1 = A + B$$

$$T2 = C + D$$

$$T3 = T1 * T2$$



# Asembler

Bardziej skomplikowany przykład

$$Z = \frac{[(A + B)(C + D)]}{W}$$

$$T1 = A + B$$

$$T2 = C + D$$

$$T3 = T1 * T2$$

$$Z = T3/W$$



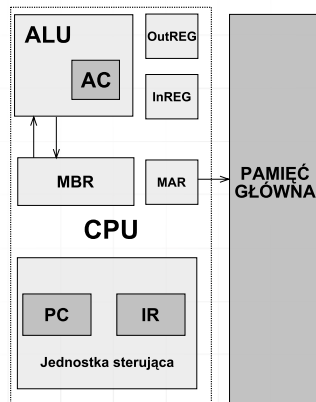
Ten rozdział jest nieobowiązkowy.



# MARIE

## MARIE — A Machine Architecture that is Really Intuitive and Easy

- ▶ notacja dwójkowa, zapis w kodzie dopełnieniowym
- ▶ przechowywanie programu, stała długość słowa
- ▶ adresowanie słowne
- ▶ 4K pamięci głównej (12 bitów na każdy adres)
- ▶ 16-bitowe dane (16-bitowe słowa)
- ▶ 16-bitowe rozkazy (4-bitowy kod operacji + 12-bitowy adres)
- ▶ 16-bitowy akumulator (AC)
- ▶ 16-bitowy rejestr rozkazów (IR)
- ▶ 16-bitowy rejestr bufora pamięci (MBR)
- ▶ 12-bitowy licznik rozkazów (PC)
- ▶ 12-bitowy rejestr adresów pamięci (MAR)
- ▶ 8-bitowy rejestr wejściowy (InREG)
- ▶ 8-bitowy rejestr wyjściowy (OutREG)



# Simulator MARIE

The screenshot shows the MARIE Simulator window with the following components:

- Assembly Code Table:**

	label	opcode	operand	hex
<input type="checkbox"/>	100	LOAD	B	1105
<input type="checkbox"/>	101	ADD	C	3106
<input type="checkbox"/>	102	STORE	A	2104
<input type="checkbox"/>	103	HALT		7000
<input type="checkbox"/>	104	A	DEC	0
<input type="checkbox"/>	105	B	DEC	20
<input type="checkbox"/>	106	C	DEC	30

- Registers:** AC (0, Dec), IR (0000, Hex), MAR (000, Hex), MBR (0000, Hex), PC (100, Hex), INP... (ASCII).
- OUTPUT:** A text area for displaying simulation output, currently empty.
- Memory:** A table showing memory addresses and their contents.

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
090	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
0A0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
0B0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
0C0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
0D0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
0E0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
0F0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
100	1105	3106	2104	7000	0000	0014	001E	0000	0000	0000	0000	0000	0000	0000	0000	0000
110	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

Message: /tmp/tmp/test.mex loaded.





# Zadanie domowe

## Prosty program dla komputera MARIE (na przykład mnożenie dwu liczb)...

Literatura dodatkowa:



MARIE.

<https://pl.wikipedia.org/wiki/MARIE>.

Only in Polish.



Student resources – essentials of computer organization and architecture, second edition.

[http://samples.jbpub.com/9781284123036/9781284136852\\_FMxx\\_Print\\_Final.pdf](http://samples.jbpub.com/9781284123036/9781284136852_FMxx_Print_Final.pdf).

Only Table of Contents and Preface.



Kuo-pao Yang.

MARIE: An introduction to a simple computer.

<https://www2.southeastern.edu/Academics/Faculty/kyang/2013/Spring/CMPS375/ClassNotes/CMPS375ClassNotesChap04.pdf>, 2013.



Linda Null and Julia Lobur.

*The essentials of computer organization and architecture.*

Jones and Bartlett Publishers, Sudbury, Mass., 2006.



Linda Null and Julia Lobur.

MARIE: an introduction to a simple computer.

In *The essentials of computer organization and architecture*. 2006.

[http://samples.jbpub.com/9781449600668/00068\\_CH04\\_Null13e.pdf](http://samples.jbpub.com/9781449600668/00068_CH04_Null13e.pdf).



Linda Null and Julia Lobur.

A guide to the MARIE machine simulator environment, 2010.

<https://cs.msutexas.edu/~simpson/wordpress/wp-content/uploads/2012/12/MarieGuide.pdf>.



# Odwrotna Notacja Polska

Popatrzmy na działanie:

$$3 + 7 \times 5$$

Ile wynosi wynik?



# Odwrotna Notacja Polska

Popatrzmy na działanie:

$$3 + 7 \times 5$$

Ile wynosi wynik?  
50 czy 38?



# Odwrotna Notacja Polska

Popatrzmy na działanie:

$$3 + 7 \times 5$$

Ile wynosi wynik?

**50 czy 38?**

A który jest poprawny?



# Odwrotna Notacja Polska

Popatrzmy na działanie:

$$3 + 7 \times 5$$

Ile wynosi wynik?

**50 czy 38?**

A który jest poprawny?

Czemu tak łatwo znaleźć kalkulator który liczy „źle”?



# „Ważność” działań arytmetycznych

1. potęgowanie
2. mnożenie i dzielenie
3. dodawanie i odejmowanie

Nawiasy mogą ją zmieniać!



# „Ważność” działań arytmetycznych

1. potęgowanie
2. mnożenie i dzielenie
3. dodawanie i odejmowanie

Nawiasy mogą ją zmieniać!

A gdzie jest zmiana znaku?



## Czy jest możliwy zapis jednoznaczny?

Polski logik, Łukasiewicz, wprowadził notację „przedrostkową”. Zamiast  $z = x + y$  zaproponował zapis:

+xy





## Czy jest możliwy zapis jednoznaczny?

Polski logik, Łukasiewicz, wprowadził notację „przedrostkową”. Zamiast  $z = x + y$  zaproponował zapis:

$$+xy$$

Zwracam uwagę że jest on bardzo podobny do zapisu funkcji dwu zmiennych:

$$z = f(x, y)$$

Funkcja **suma** jest też dwuargumentowa:

$$z = +(x, y)$$



Działanie  $3 + 7 \times 5$  oznaczające  $3 + (7 \times 5)$  zapisujemy:

$$\begin{array}{r} + \times 7 \ 5 \ 3 \\ \quad \underbrace{\phantom{7 \ 5 \ 3}} \\ \quad \quad 35 \\ \underbrace{\phantom{+ \times 7 \ 5 \ 3 \ 35}} \\ \quad \quad 38 \end{array}$$

# Odwrotny zapis polski

Utarło się używanie innego zapisu: najpierw podaje się argumenty działania, później samo działanie:

$$xy+$$

Stąd nazwa: „Odwrotna notacja polska”.

Nasze działanie zapisujemy tak:

$$75 \times 3+$$

a to bardziej skomplikowane tak:

$$A B + C D + \times W /$$



# Odwrotna Notacja Polska — stos

Praktyczna realizacja działania

$$A B + C D + \times W /$$

wymaga stosu. I dodatkowych operacji w języku wewnętrznym:

- ▶ `Zapisz_na_stos` przepisuje zawartość akumulatora na stos.
- ▶ `Pobierz_ze_stosu` pobiera ze stosu wartość i przepisuje ją do akumulatora



# Odwrotna Notacja Polska — stos

Praktyczna realizacja działania

$$A B + C D + \times W /$$

wymaga stosu. I dodatkowych operacji w języku wewnętrznym:

- ▶ `Zapisz_na_stos` przepisuje zawartość akumulatora na stos.
- ▶ `Pobierz_ze_stosu` pobiera ze stosu wartość i przepisuje ją do akumulatora

Stos czasami jest nazywany pamięcią LIFO (Last In First Out)...



# Odwrotna Notacja Polska — stos

Praktyczna realizacja działania

$$A B + C D + \times W /$$

wymaga stosu. I dodatkowych operacji w języku wewnętrznym:

- ▶ `Zapisz_na_stos` przepisuje zawartość akumulatora na stos.
- ▶ `Pobierz_ze_stosu` pobiera ze stosu wartość i przepisuje ją do akumulatora

Stos czasami jest nazywany pamięcią LIFO (Last In First Out)...

Kolejka jest nazywana pamięcią FIFO (First In First Out).



## Idea stosu

