Hello, world!

Van Code naar Zand

Hello, world!

Van Code naar Zand

```
puts("Hello, world!")
```

```
start:
        MOV C, hello
        MOV D, 232
        CALL print
        HLT

print:
        PUSH A
        PUSH B
        MOV B, 0
```

```
1F 0F 48 65 6C 6C 6F
20 57 6F 72 6C 64 21
00 06 02 02 06 03 E8
38 18 00 32 00 32 01
06 01 00 03 00 02 05
03 00 12 02 12 03 15
01 02 27 1F 36 01 36
00 39
```

LW 04/2025

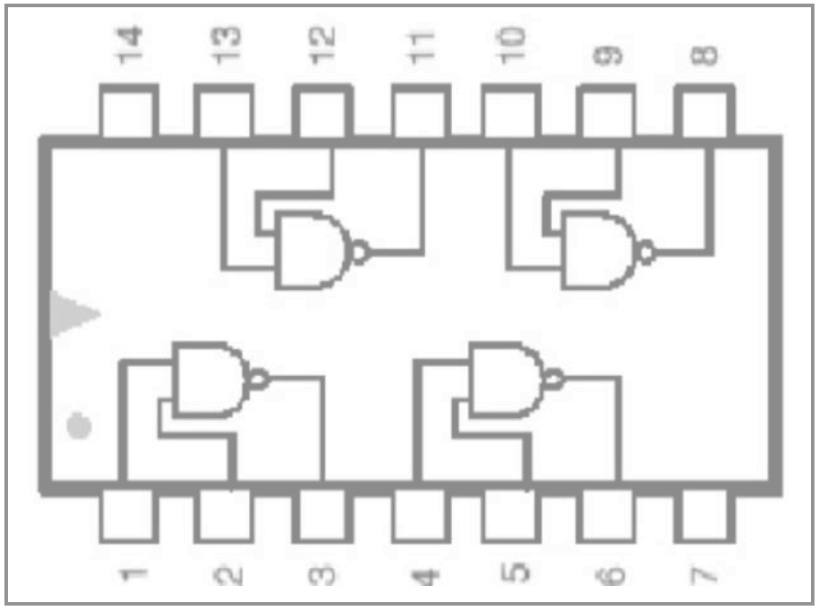
Hello, world!

Van Code naar Zand

```
puts("Hello, world!")
```

```
start:
        MOV C, hello
        MOV D, 232
        CALL print
        HLT

print:
        PUSH A
        PUSH B
        MOV B, 0
```

```
1F 0F 48 65 6C 6C 6F
20 57 6F 72 6C 64 21
00 06 02 02 06 03 E8
38 18 00 32 00 32 01
06 01 00 03 00 02 05
03 00 12 02 12 03 15
01 02 27 1F 36 01 36
00 39
```



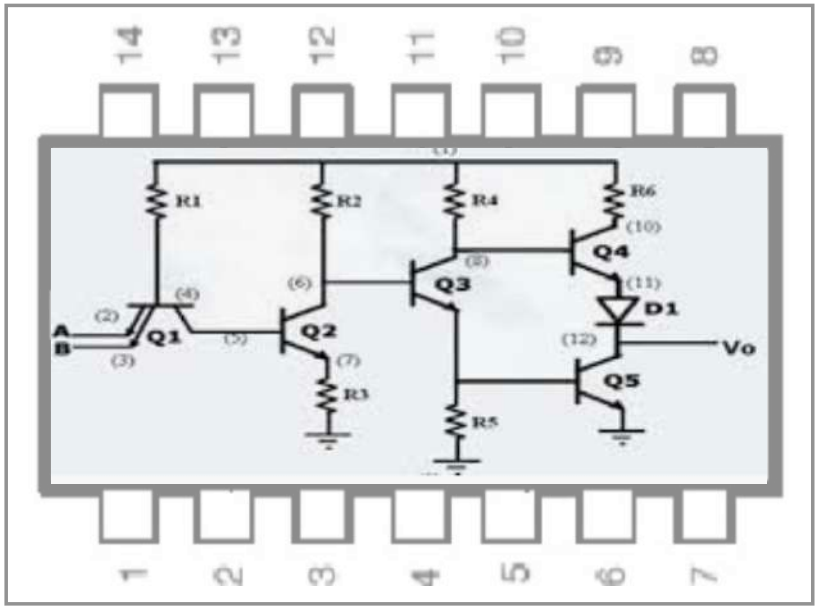
Hello, world!

Van Code naar Zand

```
puts("Hello, world!")
```

```
start:
        MOV C, hello
        MOV D, 232
        CALL print
        HLT

print:
        PUSH A
        PUSH B
        MOV B, 0
```

```
1F 0F 48 65 6C 6C 6F
20 57 6F 72 6C 64 21
00 06 02 02 06 03 E8
38 18 00 32 00 32 01
06 01 00 03 00 02 05
03 00 12 02 12 03 15
01 02 27 1F 36 01 36
00 39
```



Enkele uitdagingen rond 1935

- CPU registers & schakel-logica
- Het werkgeheugen
  - *Er was al veel bekend:*
    - *~1819: Charles Babbage's difference (analytical) engine*
    - *~1840: George Boole's binary logic*

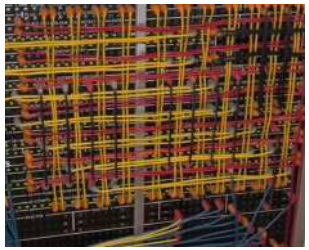
Taal: "assembler"

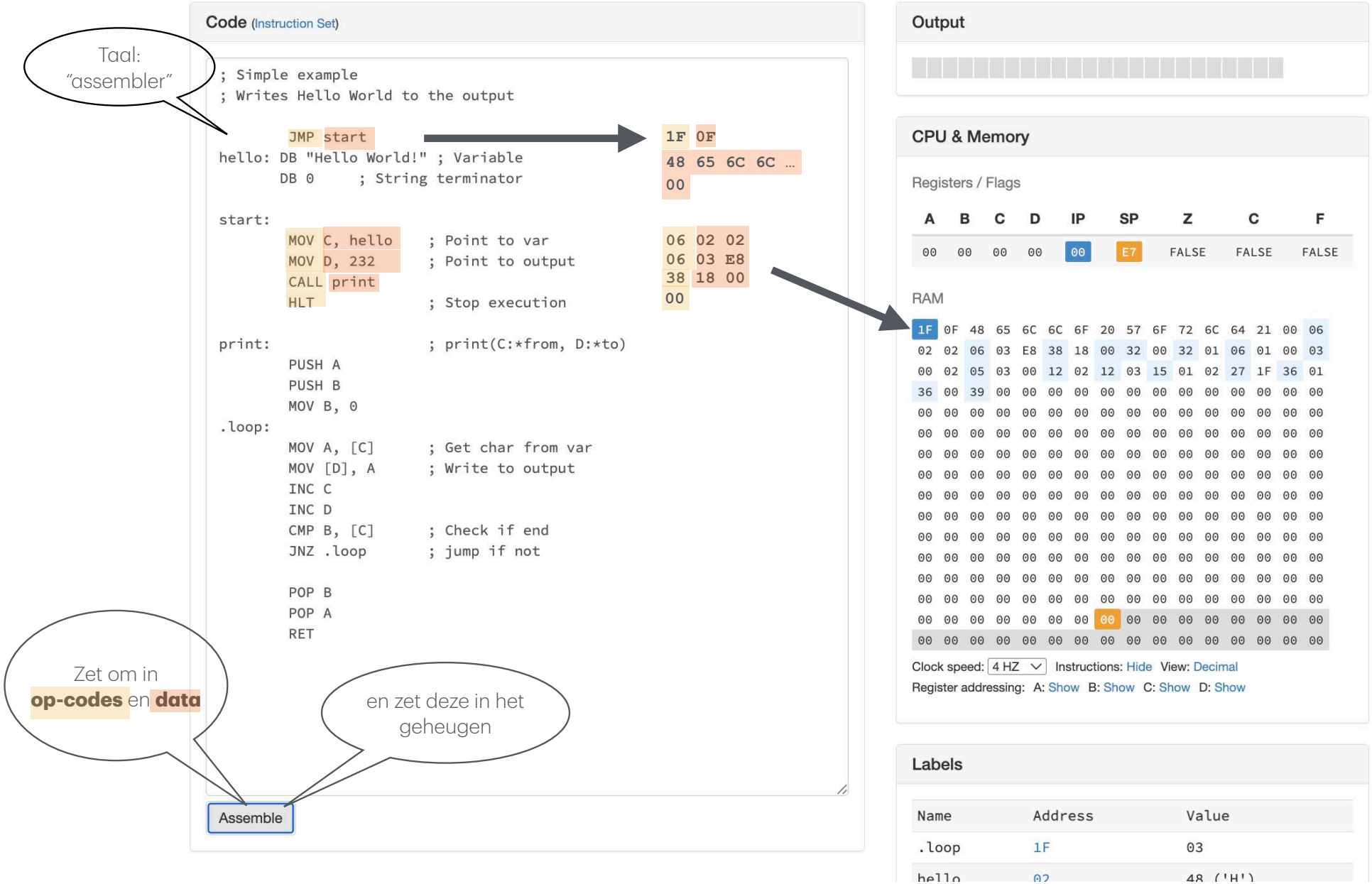
Operation code en Data

```
; Simple example
; Writes Hello World to the output

        JMP start
hello:  DB "Hello World!" ; Variable
        DB 0      ; String terminator

start:
        MOV C, hello     ; Point to var
        MOV D, 232       ; Point to output
        CALL print
        HLT              ; Stop execution
```

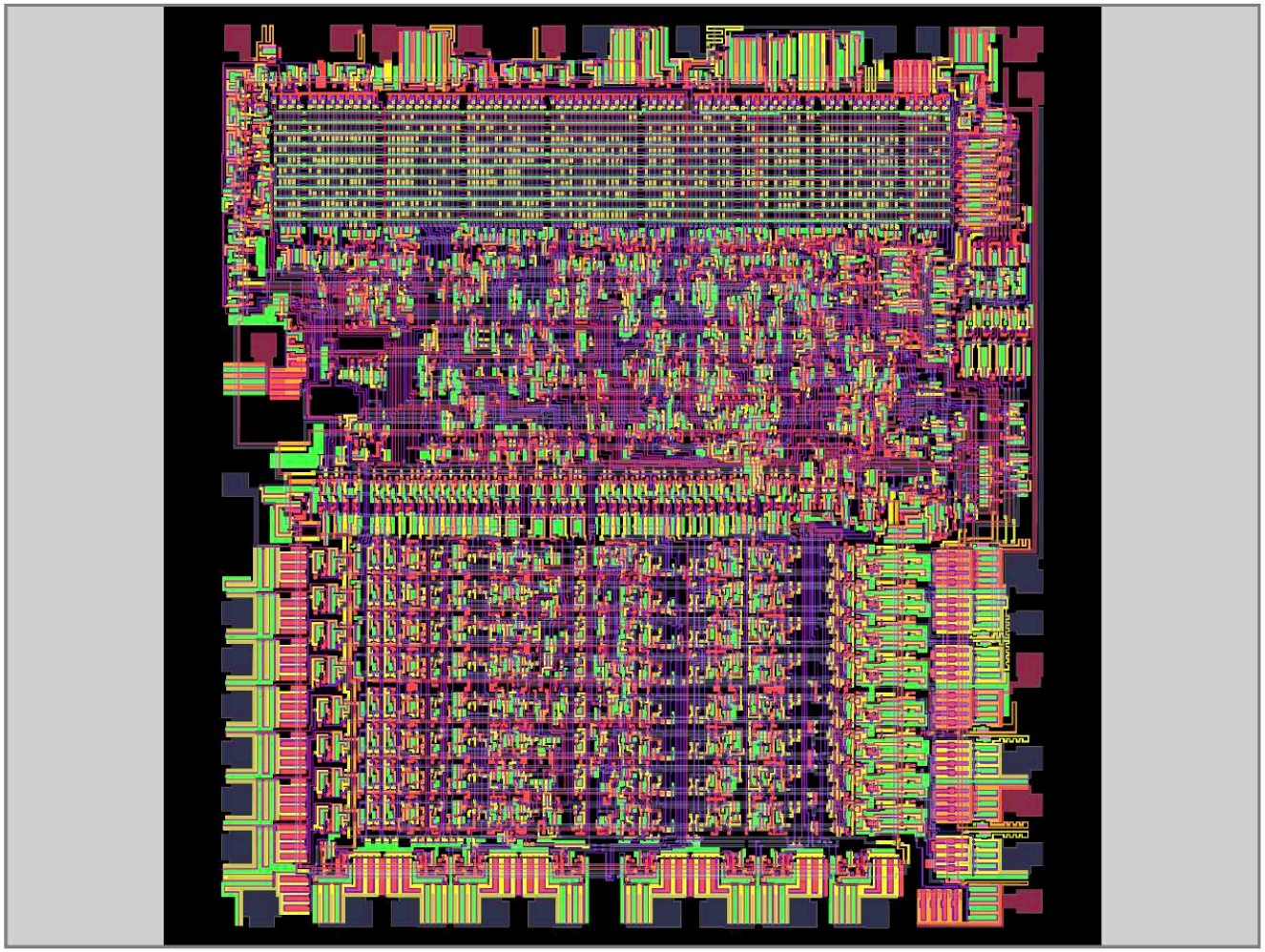
assembleren

```
1F 0F
48 65 6C 6C …
00

06 02 02
06 03 E8
38 18 00
00
```




Taal: "assembler"
Code (Instruction Set)
; Simple example
; Writes Hello World to the output
JMP start
hello: DB "Hello World!" ; Variable
DB 0 ; String terminator
start:
MOV C, hello ; Point to var
MOV D, 232 ; Point to output
CALL print
HLT ; Stop execution
print: ; print(C:*from, D:*to)
PUSH A
PUSH B
MOV B, 0
.loop:
MOV A, [C] ; Get char from var
MOV [D], A ; Write to output
INC C
INC D
CMP B, [C] ; Check if end
JNZ .loop ; jump if not
POP B
POP A
RET
1F 0F
48 65 6C 6C …
00
06 02 02
06 03 E8
38 18 00
00
Zet om in op-codes en data
en zet deze in het geheugen
Assemble
Output
CPU & Memory
Registers / Flags
A B C D IP SP Z C F
00 00 00 00 00 E7 FALSE FALSE FALSE
RAM
1F 0F 48 65 6C 6C 6F 20 57 6F 72 6C 64 21 00 06
02 02 06 03 E8 38 18 00 32 00 32 01 06 01 00 03
00 02 05 03 00 12 02 12 03 15 01 02 27 1F 36 01
36 00 39 00 00 00 00 00 00 00 00 00 00 00 00 00
Clock speed: 4 HZ Instructions: Hide View: Decimal
Register addressing: A: Show B: Show C: Show D: Show
Labels
Name Address Value
.loop 1F 03
hello 02 48 ('H')

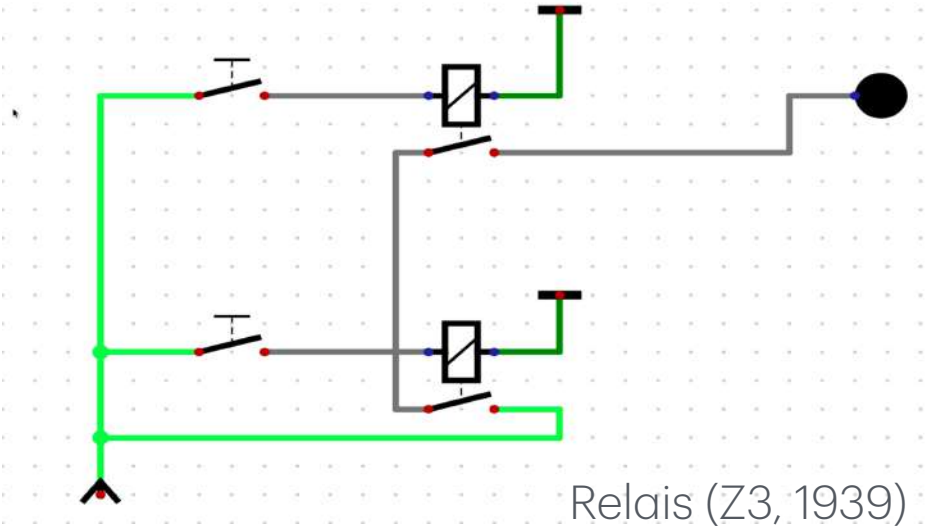
The Visual 6502



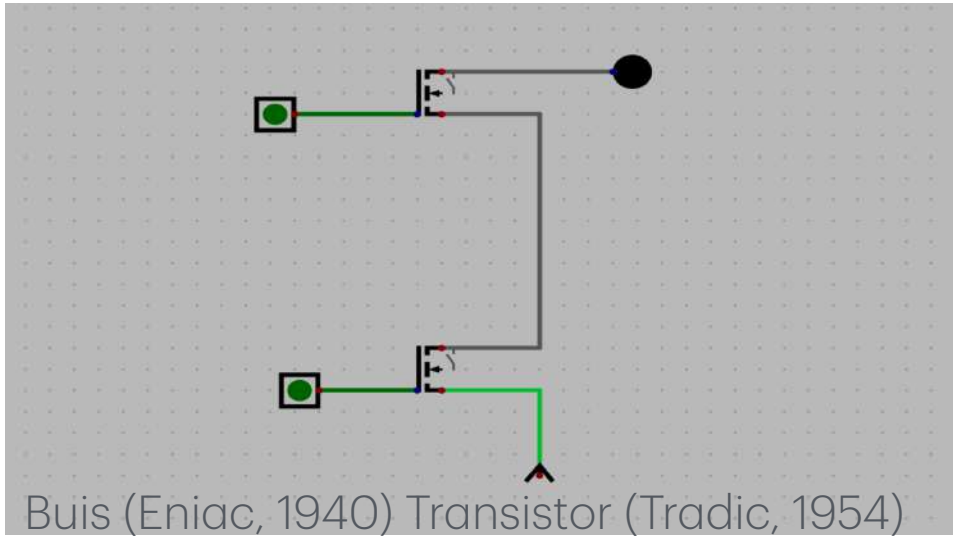
... or try Advanced

halfcyc:0 phi0:0 AB:0000 D:a9 RnW:1
PC:0000 A:aa X:00 Y:00 SP:fd nv-BdIZc
Hz: 1.0

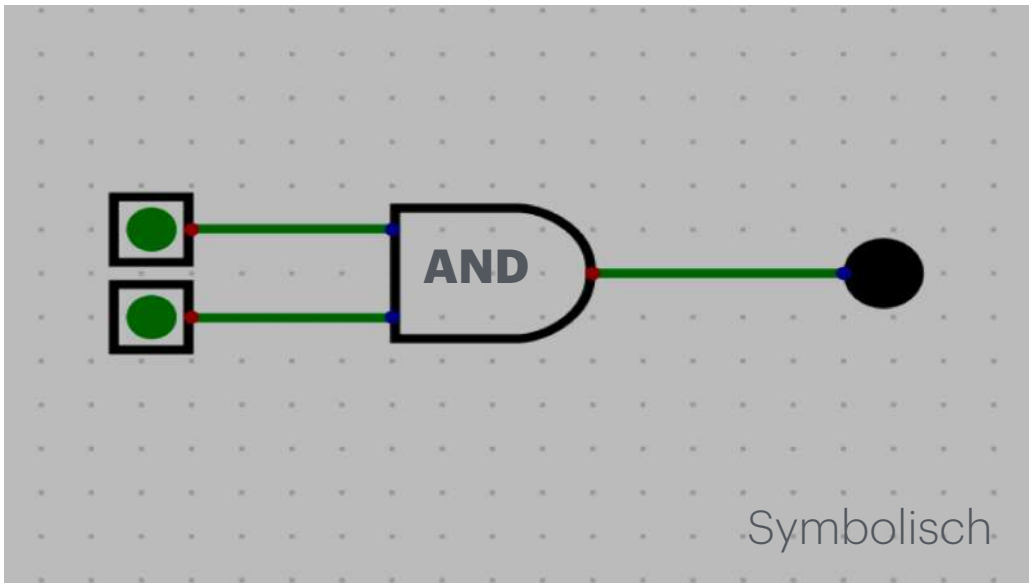
```
0000: a9 00 20 10 00 4c 02 00 00 00 00 00 00 00 00 40
0010: e8 88 e6 0f 38 69 02 60 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```



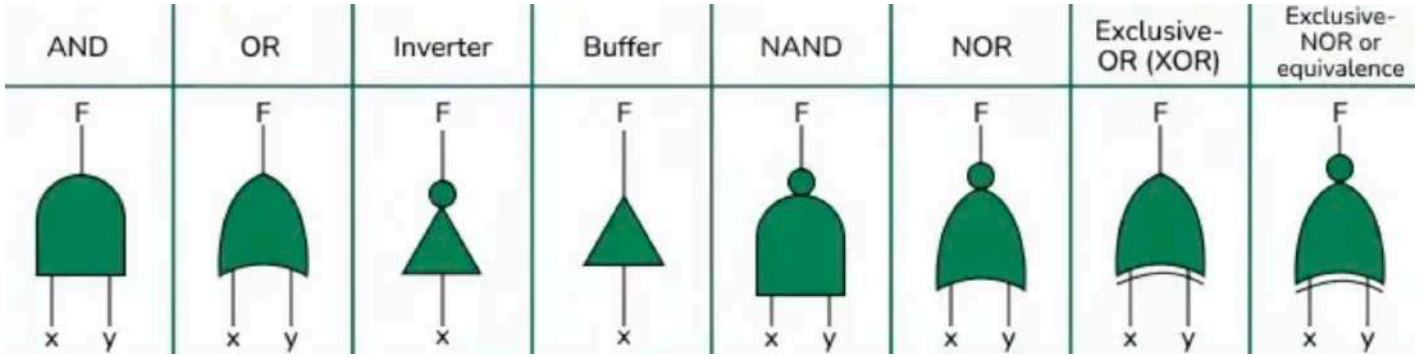
Relais (Z3, 1939)



Buis (Eniac, 1940) Transistor (Tradic, 1954)



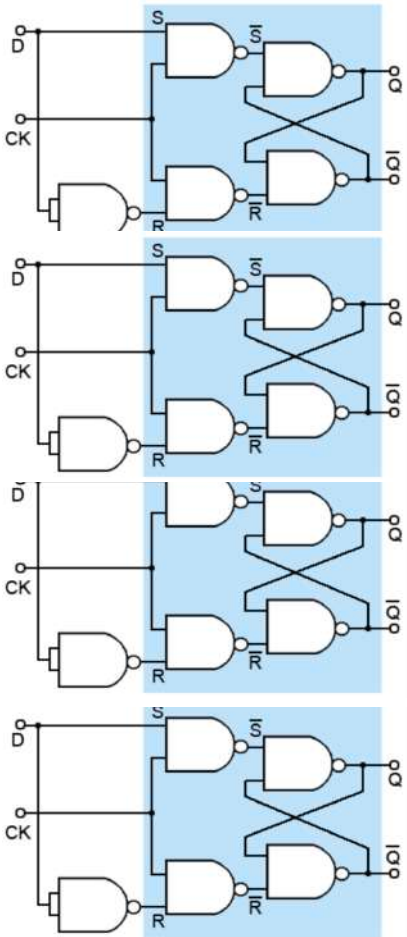
Symbolisch



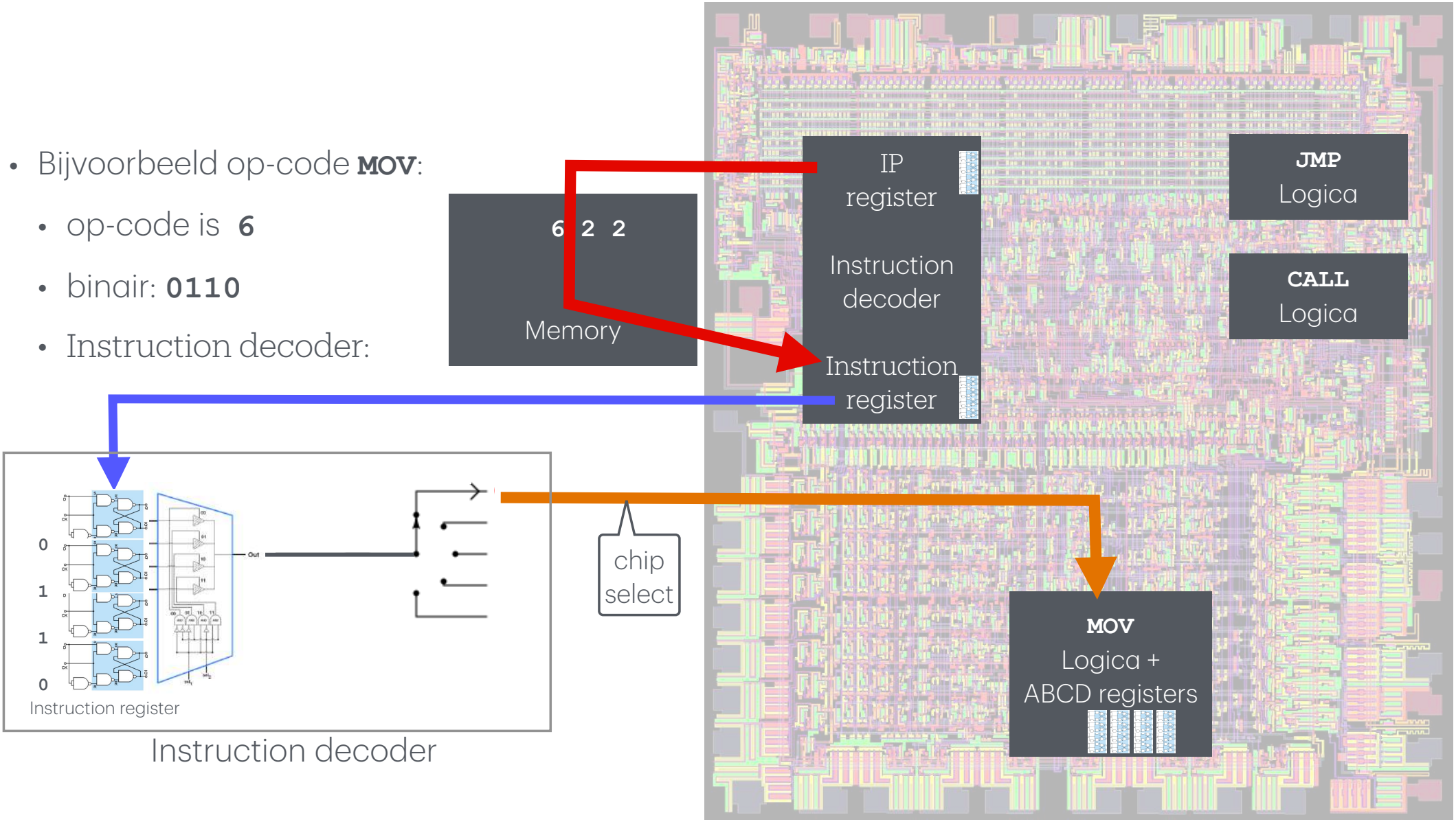
Processor	Transistor count
6502	4528
Quad-core + GPU Core i7 Ivy Bridge	1,400,000,000
Quad-core + GPU Core i7 Haswell	1,400,000,000
Dual-core Itanium 2	1,700,000,000
Six-core Core i7 Ivy Bridge E	1,860,000,000

CPU “memory”: Registers

- 4, 8, 16, 32, 64 bits opslag:
  - Instruction pointer
  - Instruction register
  - Stack pointer
  - Algemene registers (A, B, etc)
- Een register bestaat uit D(ata) “flip flops"
  - 1 flop flop bestaat uit 5 gates
    - dat zijn dus ca. 15 relais, buizen of transistoren



- Bijvoorbeeld op-code **MOV**:
  - op-code is **6**
  - binair: **0110**
  - Instruction decoder:

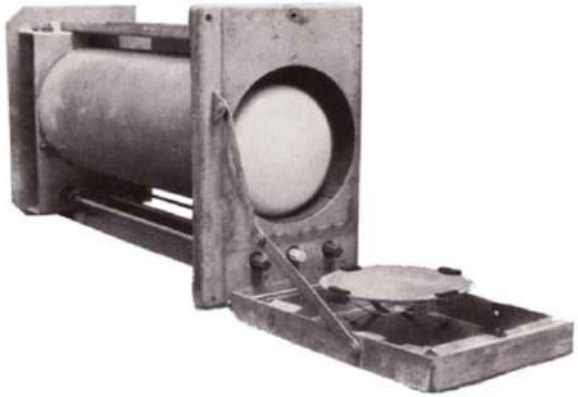


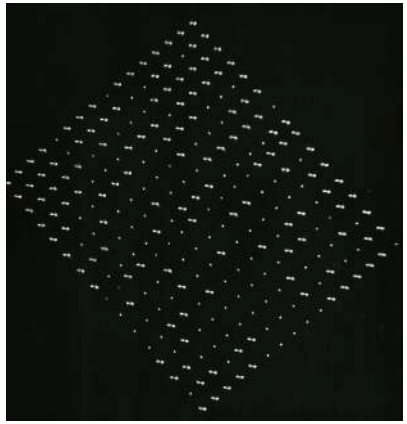
Werkgeheugen

Een overzicht vanaf 1947

1947: Williams-Kilburn Tube (CRT)

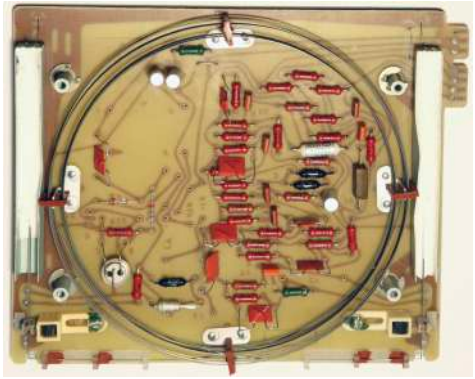
- Capaciteit: 256 tot 2560 bits
- Een bit is in feite een fosfor-eilandje van elektronen dat zeer kort bestaat
  - dus refreshen door te lezen en te (her)schrijven
- Doordat de elektronenstraal elke positie kan raken is dit feitelijk Random Access Memory
- Sneller dan Mercury Delay Line Memory maar snel ontregeld
- Gebruikt in de Manchester Mark 1, UNIVAC 1103, IBM 701, Strela-1 (Sovjet).



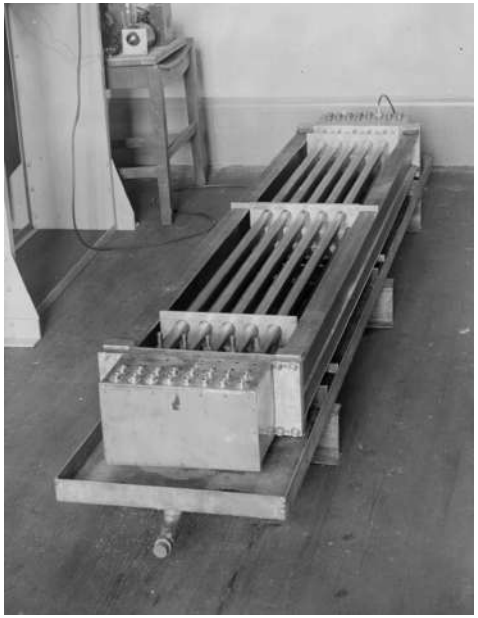


1949: Mercury Delay Line

- Ontwikkeld (1920) om analoge signalen te vertragen
- Slaat een bit op als een geluidstrilling in een buis met kwik
  - als de puls aan er de andere zijde uitkomt wordt deze weer geïnjecteerd aan het begin (recirculation time: milliseconds)
- Sequentieel access: wachten tot de “positie” verschijnt aan het einde
  - leestijd is een functie van circulatietijd en positie
- Capaciteit: enkele duizenden bits
- Gebruikt in de EDVAC en UNIVAC 1

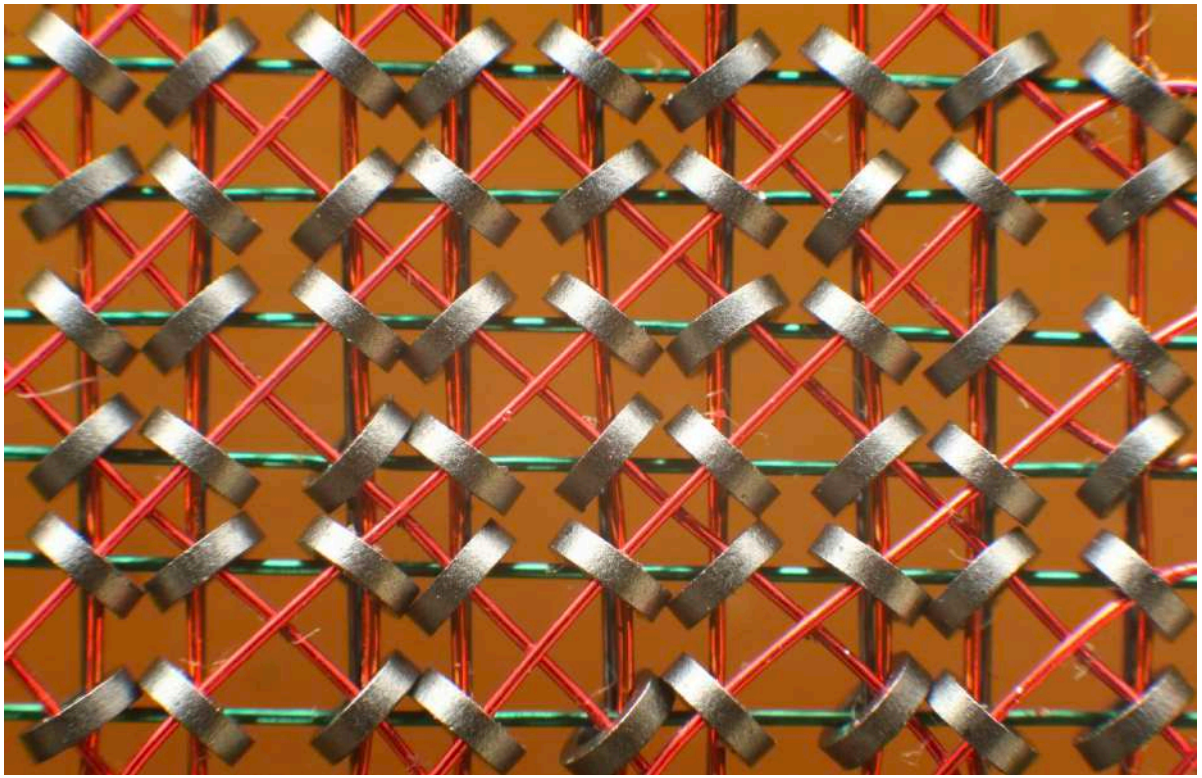


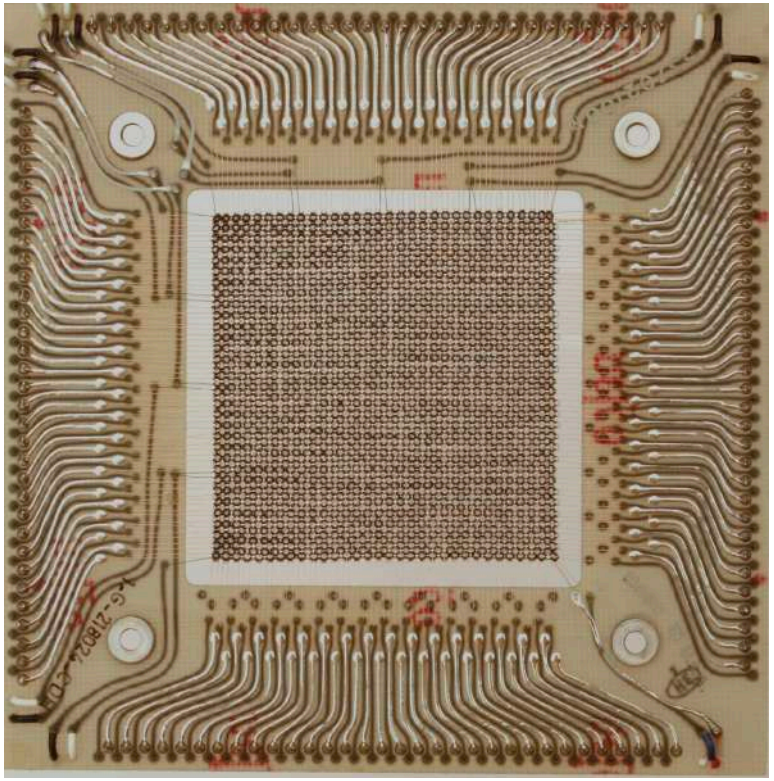
Acoustic delay line



EDSAC mercury delay line

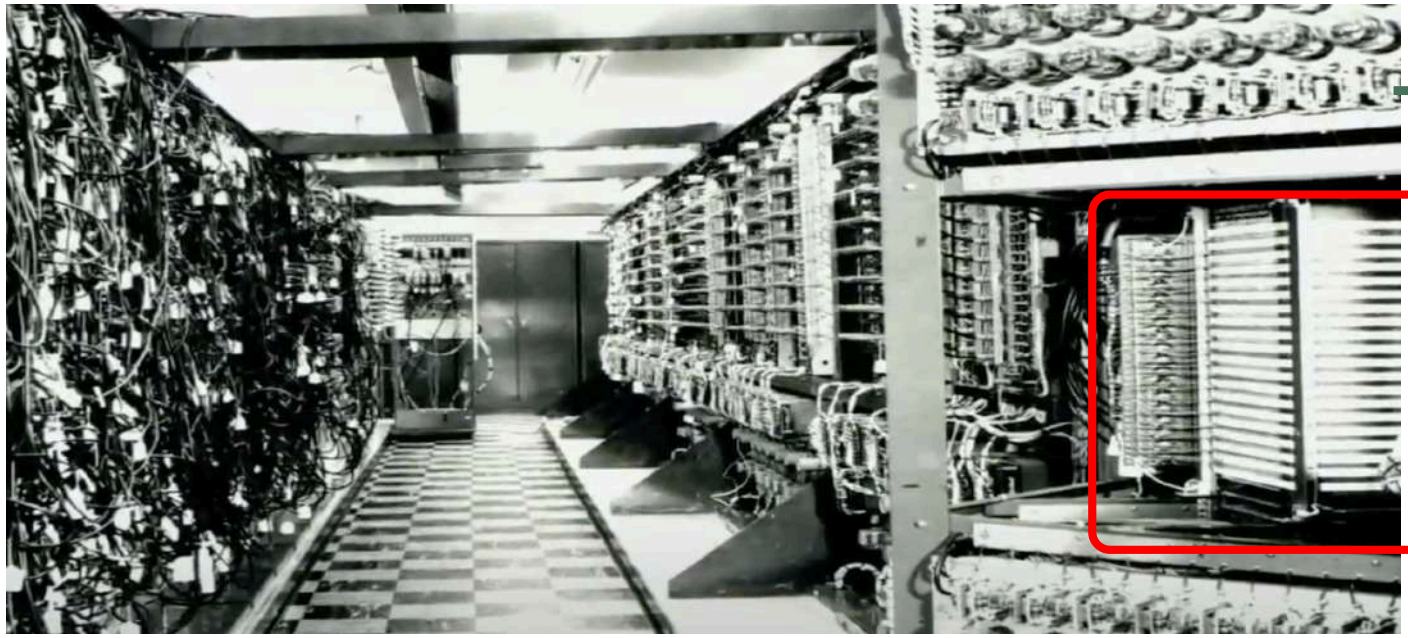
~1950: Core memory (ringkern geheugen)





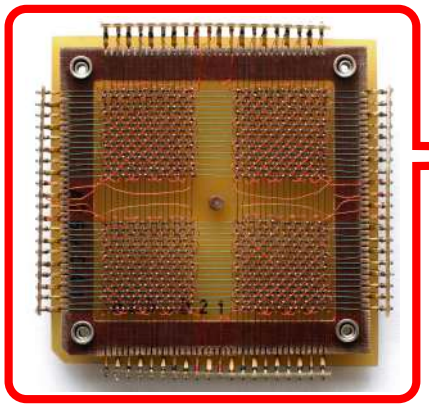
diameter: 2.5 mm in 1953 0.33 mm in 1966, in gebruik tot midden jaren ’70.

1951: Whirlwind I, eerste computer met core memory (32 kBit)

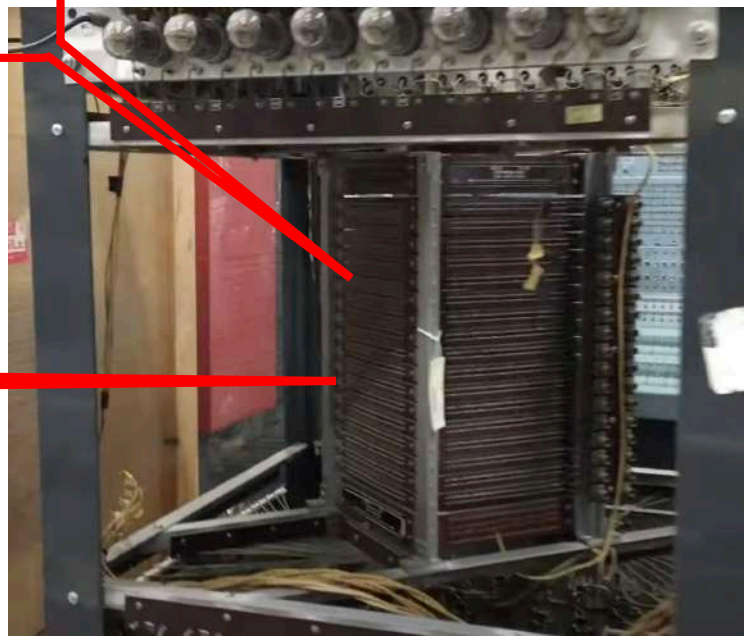


5000 buizen voor de logica en de registers

10.000 kg, 192 m2

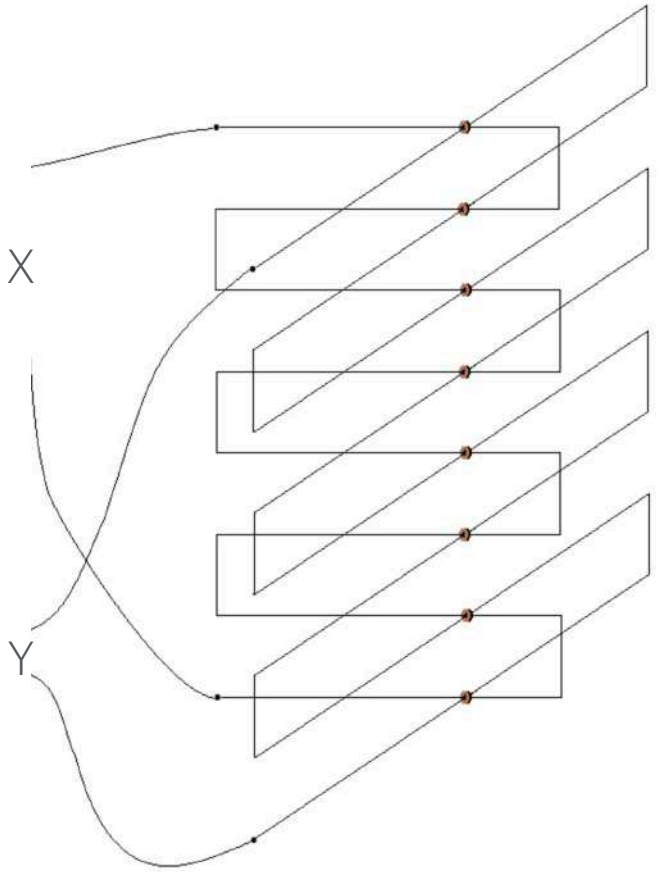


Stacked planes

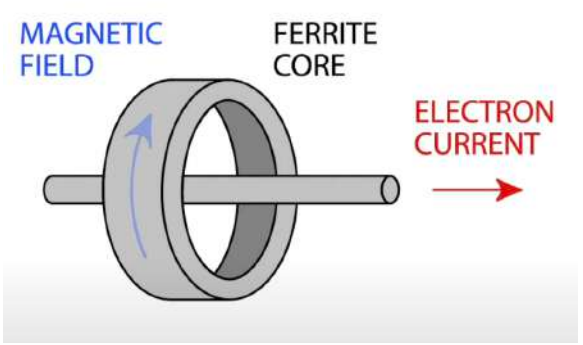


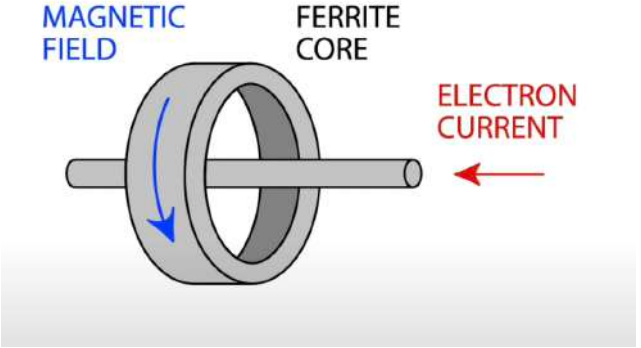
Geen bits maar Words

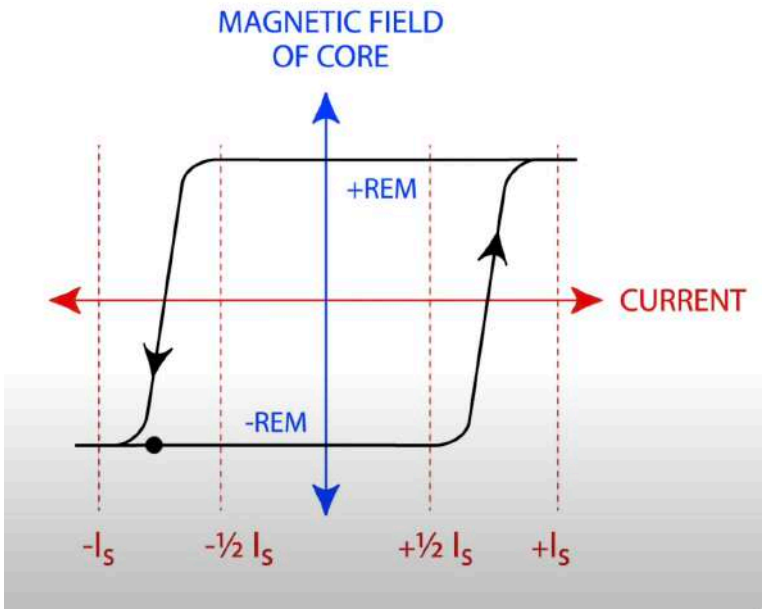
- Op-codes en adressen zijn een veelvoud van bits (4, 8, ..., 64) (Words)
- Memory planes waren gestapeld en er liep maar één X en één Y draad door alle planes
- Bij 8 planes kun je dus met één X/Y setting 8 bits uitlezen: een Word

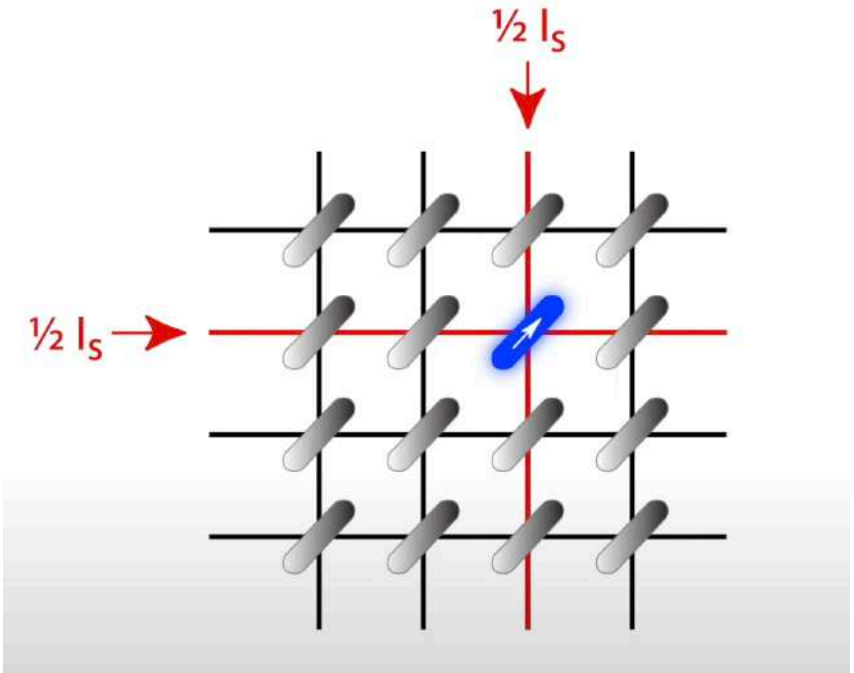


Schrijven







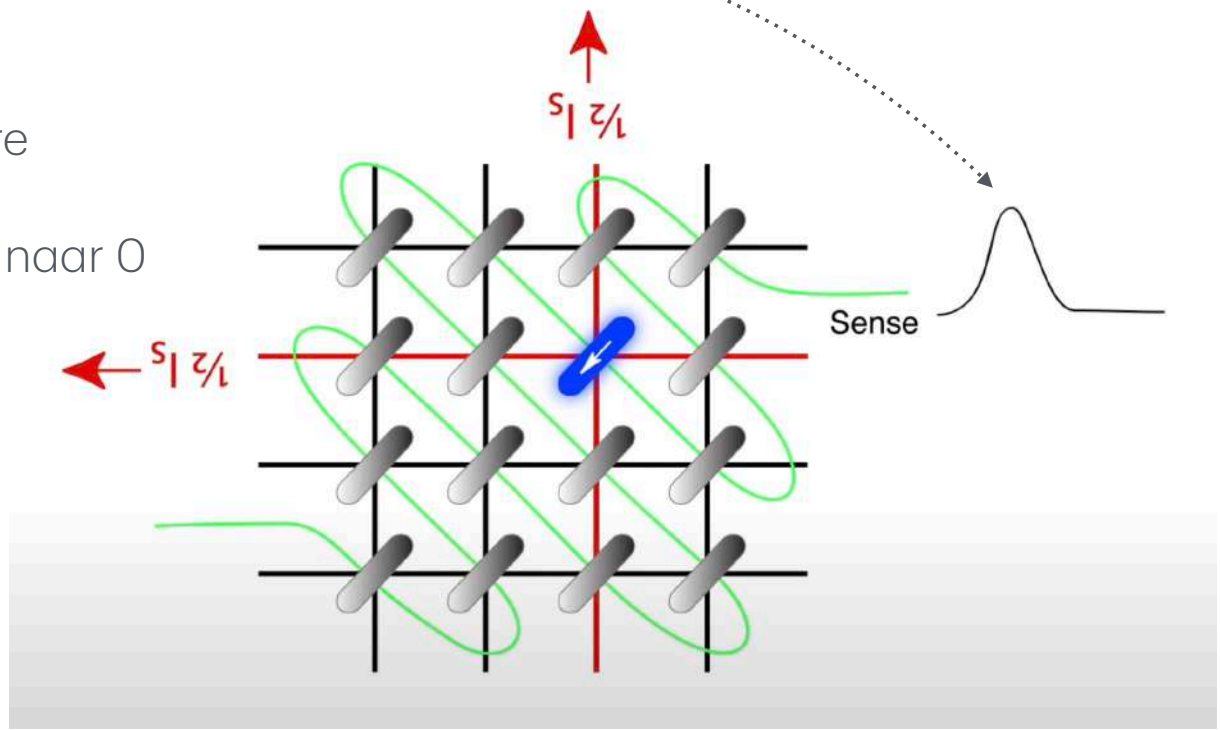


Magnetiseren lukt pas bij volledige stroomsterkte (saturatie)

Lezen

Magnetisme “lezen” kan alleen door het magnetisch veld weer te wijzigen:

- Het flippen van een 1 naar 0 veroorzaak een puls.....
  - of geen puls als het bit al een 0 was
- De puls wordt gelezen via een Sense wire
- Helaas is het bit nu wel veranderd van 1 naar 0
  - er moet nu een re-write plaatsvinden



Herschrijven

Na een “read” moet dus een re-write volgen

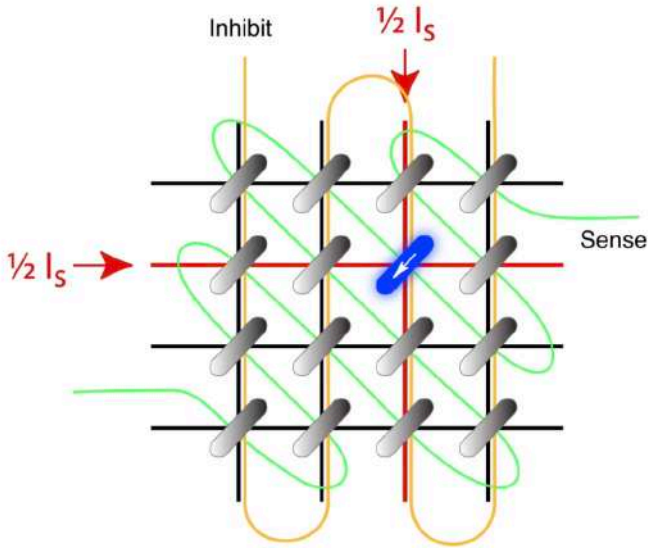
Het meest eenvoudig zou zijn om weer een write op de gelezen core te doen, maar dat kan niet door het ontwerp van stacked planes, de overige X/Y bits zouden dan ook veranderen.

Daarom is er een 4e draad: de *Inhibit* wire. Net als de Sense wire loopt er één Inhibit wire door alle cores (per plane).

Het doel van de Inhibit wire is om optioneel stroom te leveren in de tegenovergestelde richting van de schrijfstroom. Dit zal een deel van het magnetische veld neutraliseren en voorkomen dat de 7 niet betrokken cores veranderen.

In latere ontwerpen was er maar 1 extra draad nodig: de Sense en Inhibit wire zijn dan dezelfde.

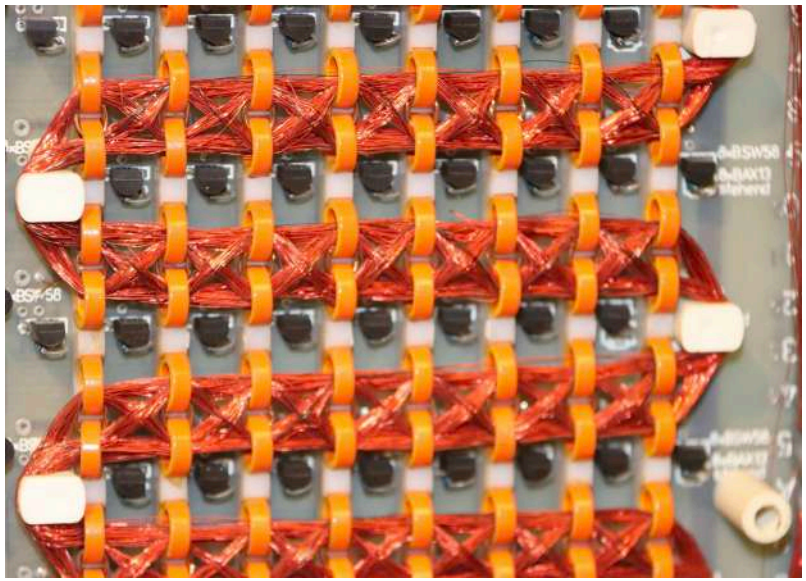
Het aantal wires en het benodigde amperage was aanzienlijk, de IBM 1620 (1959) had een veel slimmer (en ingewikkelder) ontwerp.

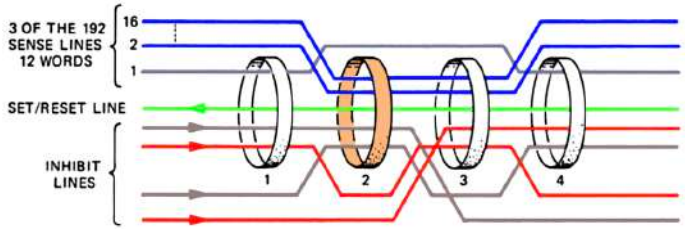


Core Rope Memory

- Een programma kan worden opgeslagen in ROM
  - Core Rope Memory is Read Only Memory
- Toegepast voor ondermeer het Apollo project (i.v.m. beperkte ruimte)
- Als de Sense wire door de core loopt, telt die core als een 1, anders is die core een 0
  - Codeer **`1011`**

- 1 core kan vaker gebruikt worden doordat er (max.) 192 Sense wires door geweven kunnen worden
  - Codeer **`0110`** erbij:





http://apolloguidance.computer/rope/

Show notes

- https://www.computerhistory.org/timeline/memory-storage/
- https://en.wikipedia.org/wiki/Magnetic-core_memory
- https://schweigi.github.io/assembler-simulator/ (simulator)
- https://adamgerhant.github.io/cpusimulation/
- https://en.wikipedia.org/wiki/Williams_tube
- https://en.wikipedia.org/wiki/Delay-line_memory
- https://www.i-programmer.info/babbages-bag/151-cpu.html?start=2
- http://www.visual6502.org/JSSim/ (simulator)
- https://www.youtube.com/watch?v=Wcj3PsK0M90 (build a d-type flip flop with tubes)
- https://www.youtube.com/watch?v=AwsInQLmjXc (core memory explained)
- http://apolloguidance.computer/rope/ (simulator)
- https://www.righto.com/2019/07/software-woven-into-wire-core-rope-and.html
- https://www.angelfire.com/oh3/ebjoew/IBM_1620_Core_Memory.html

Vragen?

Logo of LEO Computers Ltd 1954 until 1963

Also known as	Lyons Electronic Office I
Manufacturer	J. Lyons and Co.
Generation	1
Release date	1951; 74 years ago
CPU	@ 500 kHz
Memory	2K (2048) 35-bit words (i.e., $8\frac{3}{4}$ kilobytes) (ultrasonic delay-line memory based on tanks of mercury)
Removable storage	paper tape readers and punches, fast punched card readers and punches, and a 100 line a minute tabulator
Predecessor	EDSAC
Successor	LEO II