

[-november 2021-]



[-november 1971-]



[-historie intel-]

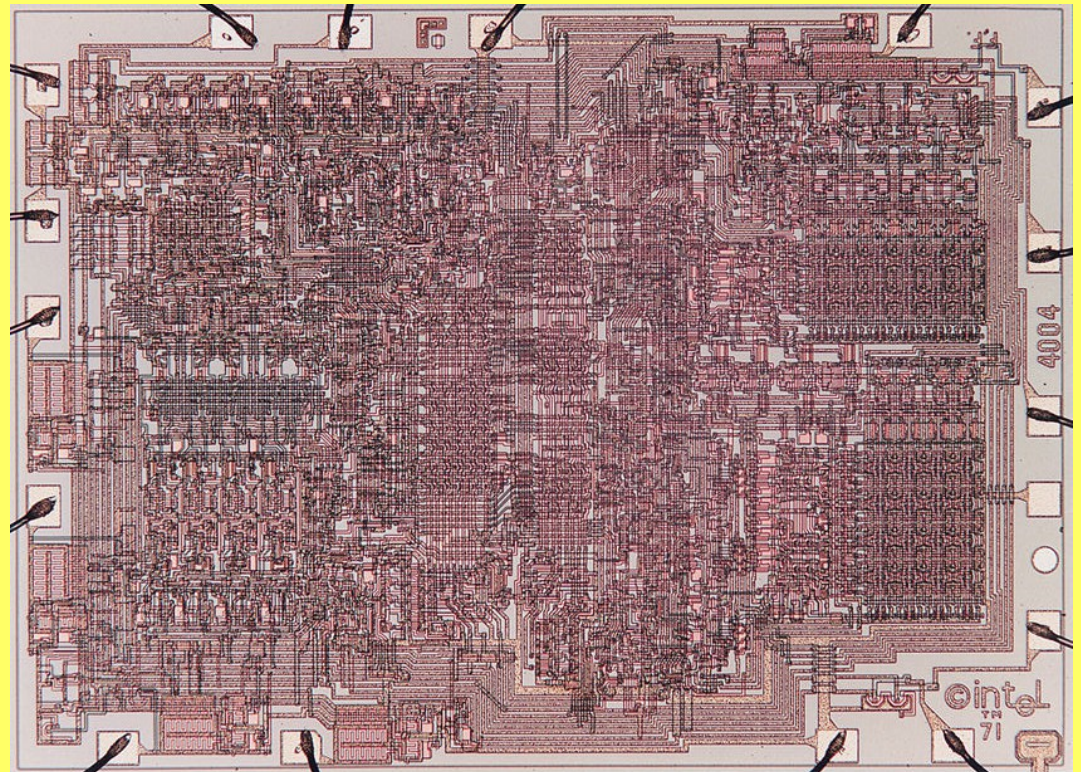
- Historie Intel:
 - Het bedrijf is in 1968 door Robert Noyce en Gordon Moore opgericht
 - Kwamen beide bij Fairchild Semiconductor vandaan
 - Santa Clara, Californië
 - Startkapitaal 2,5 miljoen dollar, in 2 dagen bijeengebracht.
 - De derde werknemer was Andrew Grove
 - DRAM 1103, 1 kilobit
 - Best verkopende RAM in 1969 - 1970

[-historie 4004-]

- Historie 4004:
 - 1969, Ted Hoff en Stan Mazor komen bij Intel
 - 1970, Federico Faggin komt bij Fairchild vandaan waar hij aan de SGS-MOS techniek gewerkt had
 - Busicom vraagt om 12 chips te ontwerpen
 - Masatoshi Shima
 - Voorjaar 1971 eerste 4004 afgeleverd
 - Zomer 1971, Busicom 141-FP
 - Intel koopt rechten terug
 - 15 November, advertentie in "Electronic news"

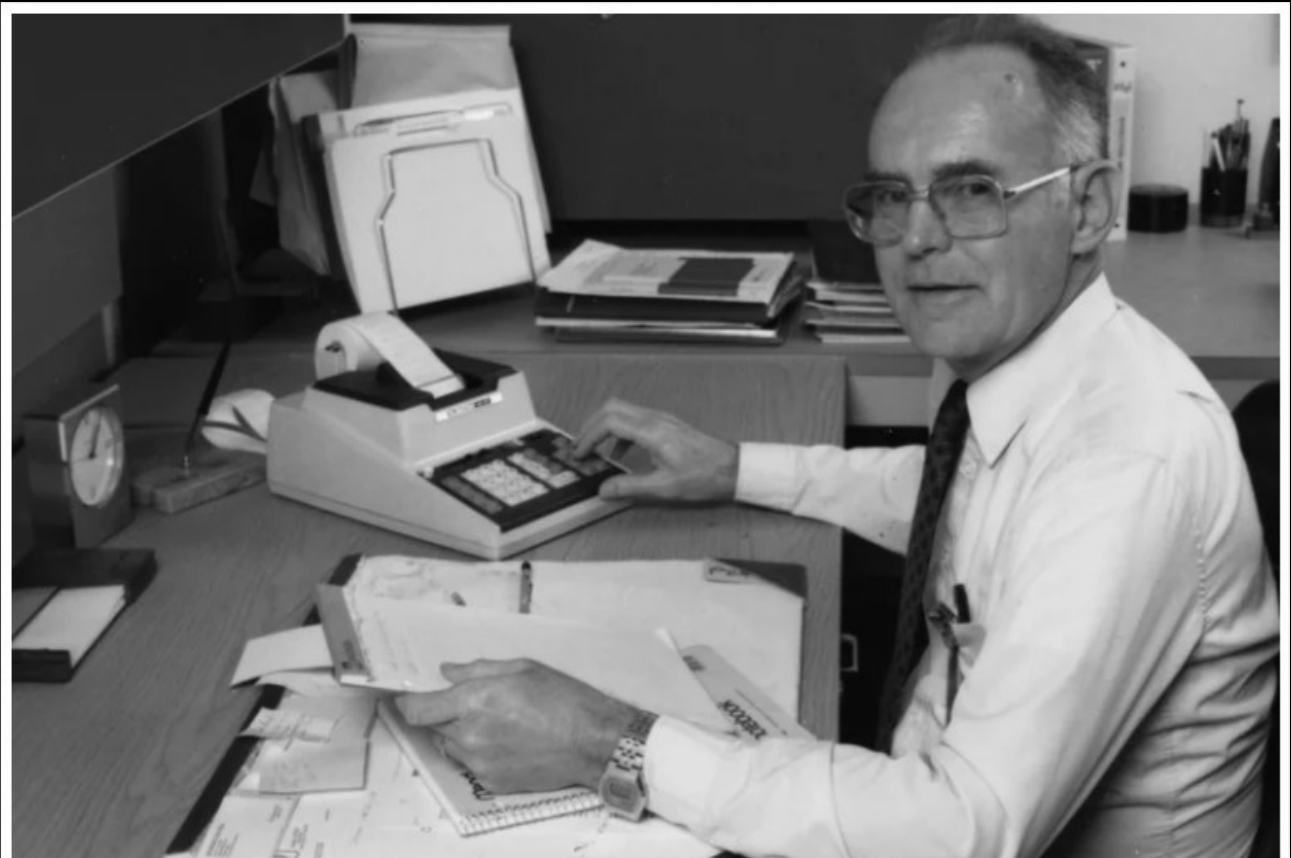
[-historie 4004-]

- Eigenschappen 4004:
 - Silicon Gate Technology (SGT) Metal Oxide Semiconductor
 - 2300 transistors
 - pMOS
 - 4 bits databus
 - 16 Pins IC
 - 740 kHz klok
 - 46 instructies
 - 4 K adres ruimte
 - $16 * 4$ bits registers



[-uitstapje **intel**-]

- Neven activiteiten.
- Horloge



Intel co-founder Gordon Moore, circa 1980, wearing his Microma wristwatch.

Courtesy of Intel

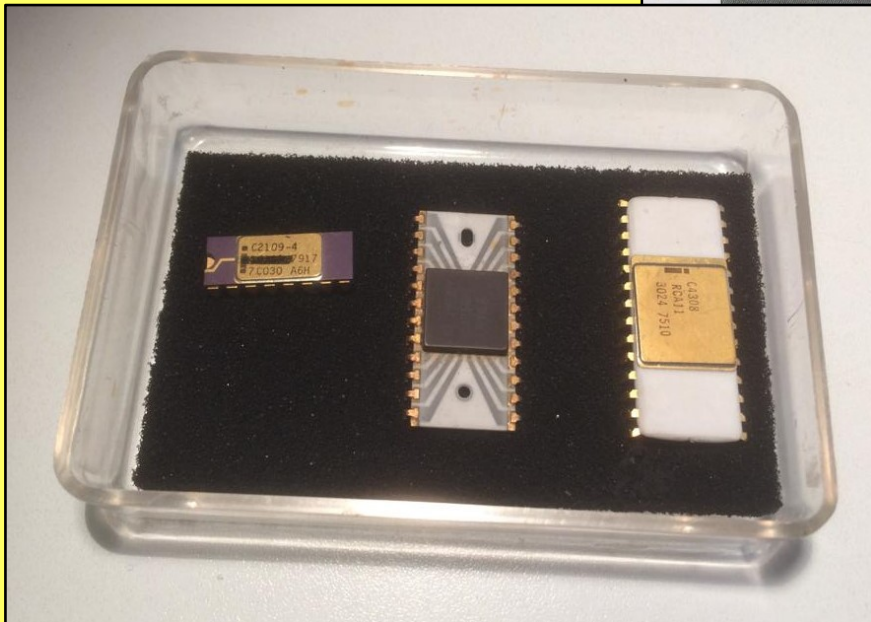
[-ik en intel-]

- Wat deed ik
rond 1970
- Hoe kwam ik
in contact met
elektronica,
computers en de
Intel 4004



[-intel-]

- Jan Wubben

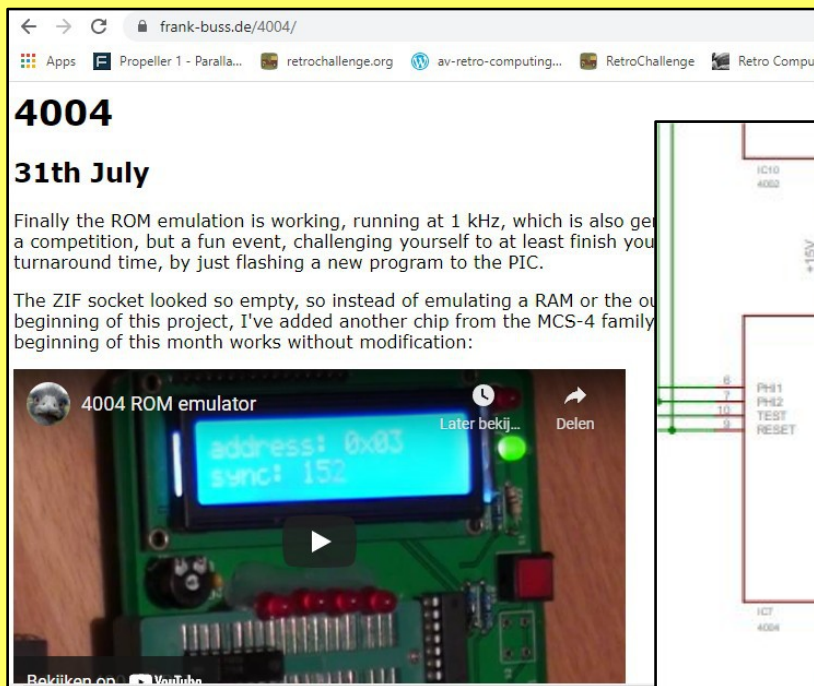


[-intel-]

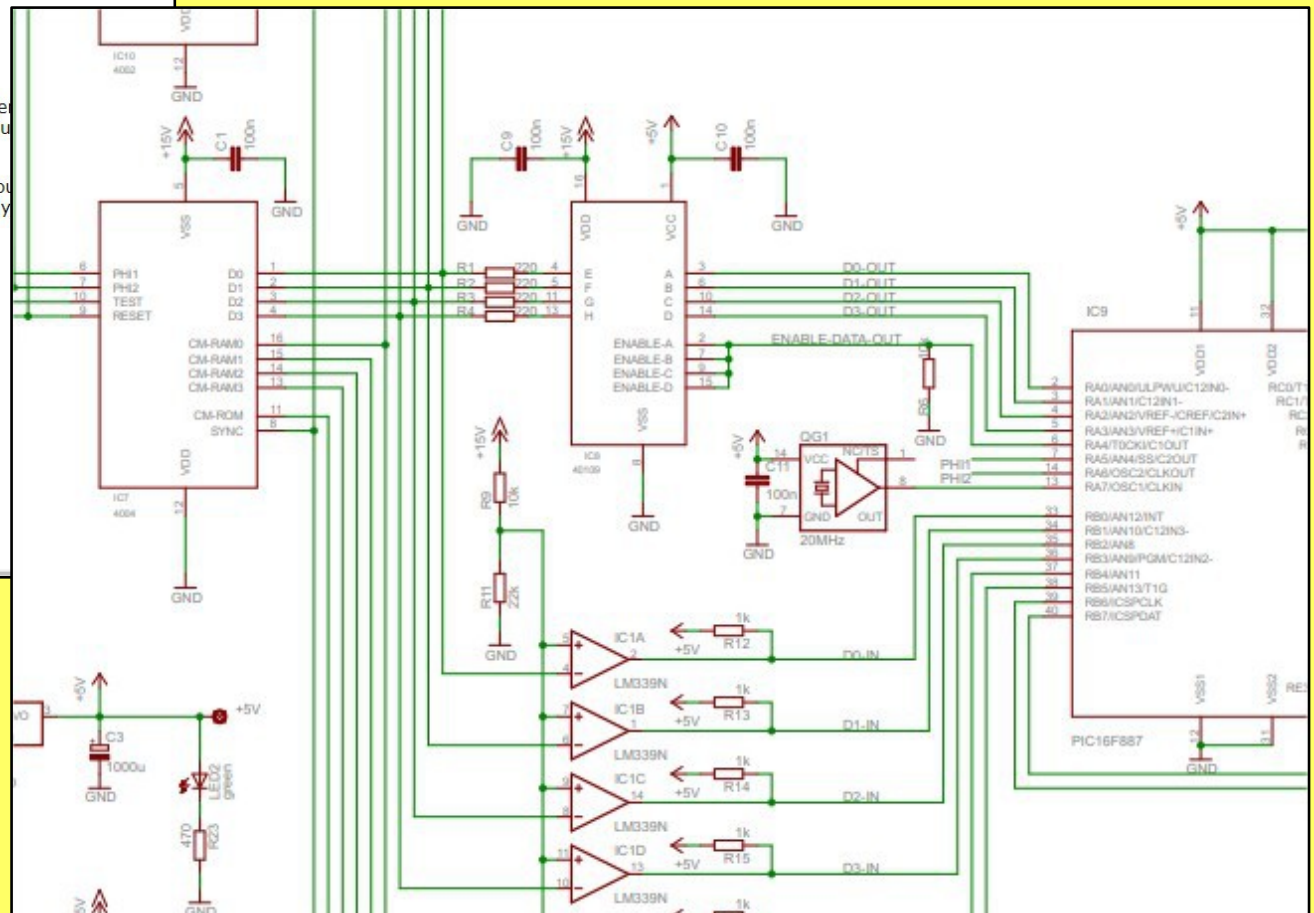
- Ruim 10 jaar in opslag
- Museum stuk
- Werkt hij nog ?
- Hoe van start ?
- Hij eet geen brood !
- ???

[-Idee intel-]

- Frank Buss: <https://frank-buss.de/4004/>



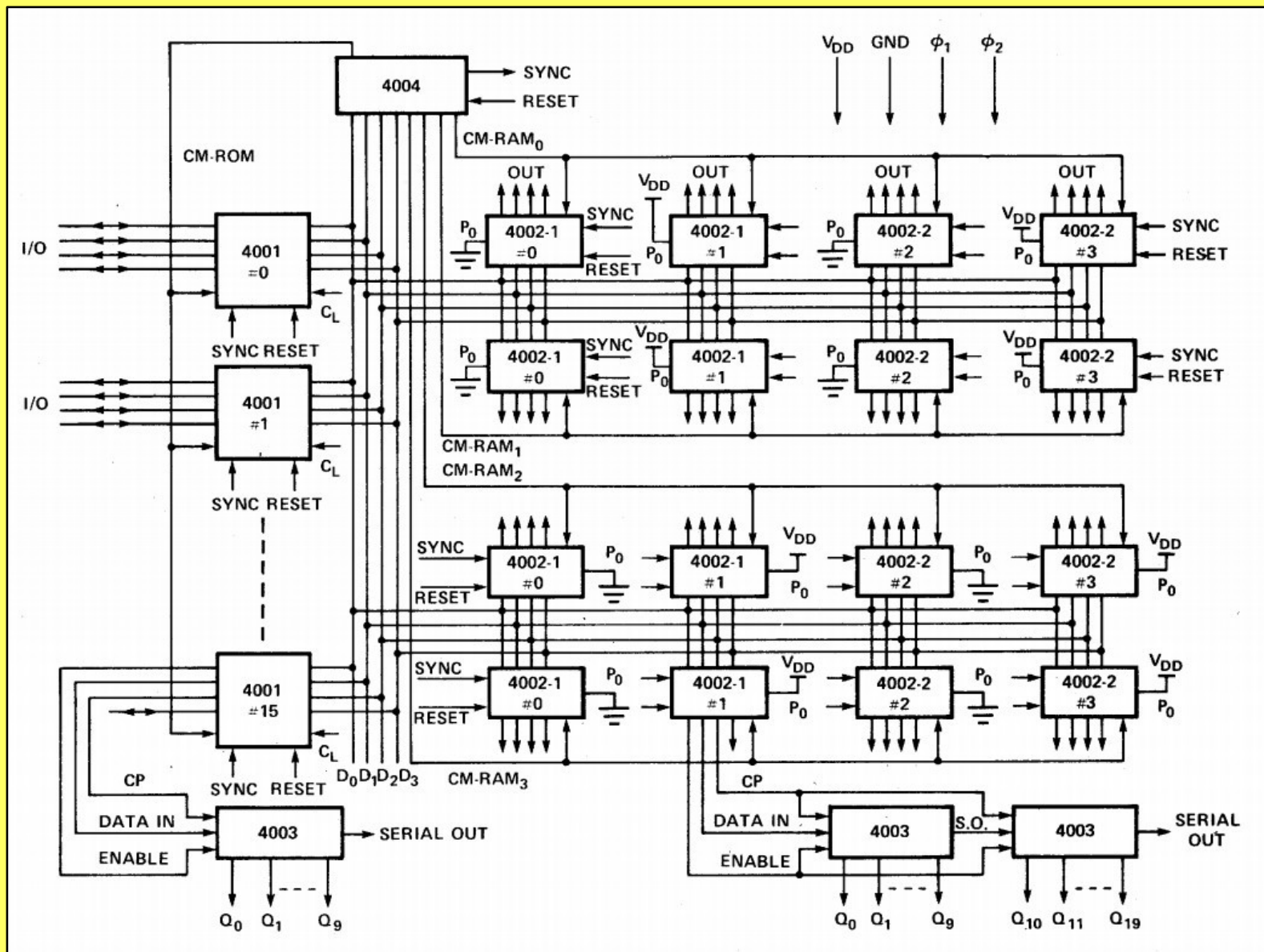
Retrochallenge 2012



[-MCS 4 intel-]

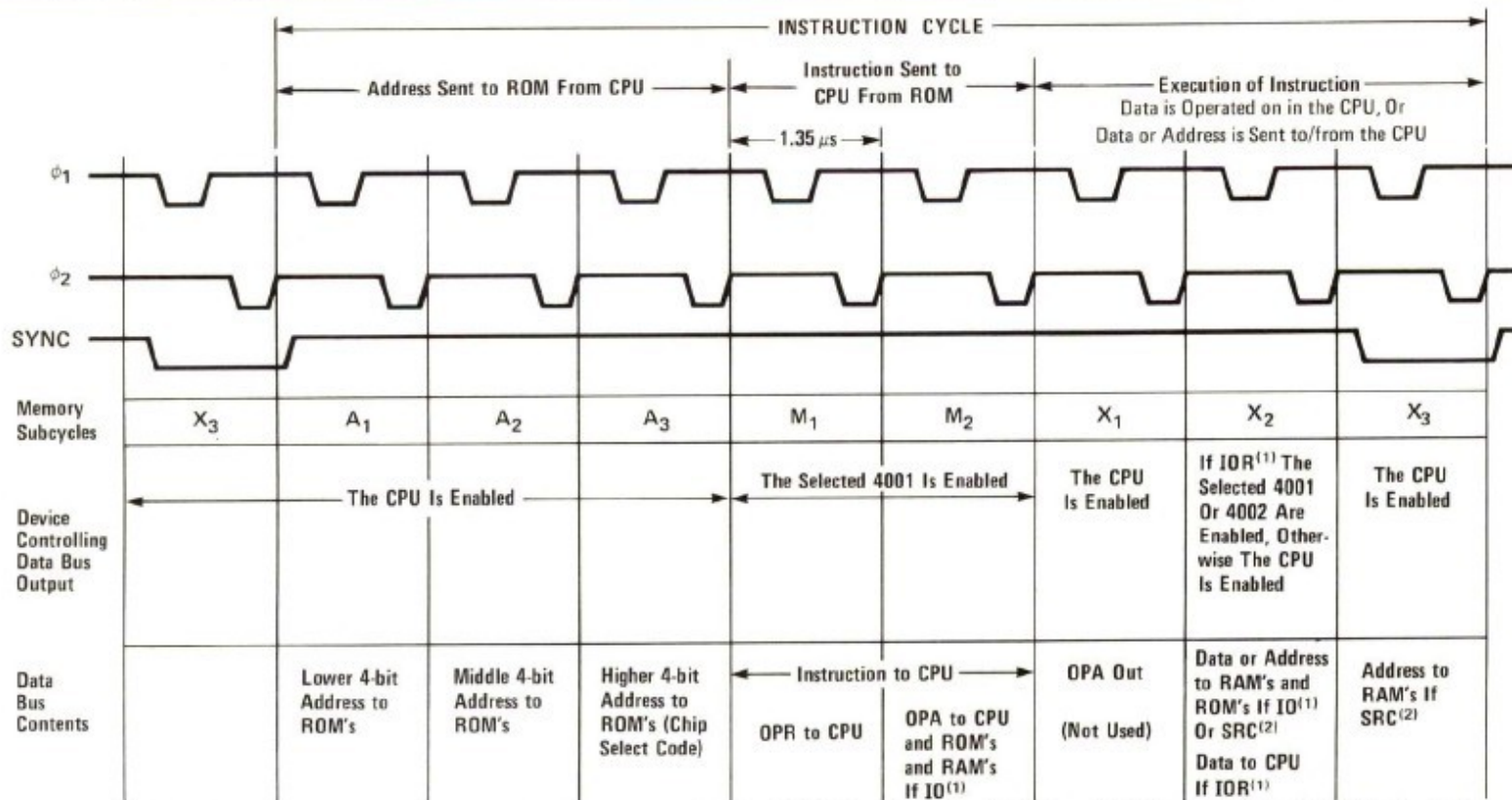
- Welke IC's en functie:
- 4001 – 256 byte ROM, 4 bit I/O
- 4002 – 320 bit RAM, 4 bit O
- 4003 – serieel in, 10 bit parallel uit, schuifregister
- 4004 – microprocessor, 4 bit, 740 kHz
- Latere uitbreiding:
- 4008 – adres uitbreiding tbv EPROM's
- 4009 – adres uitbreiding tbv I/O

[-MCS 4 intel-]



[-MCS 4 intel-]

MCS-4 Operation



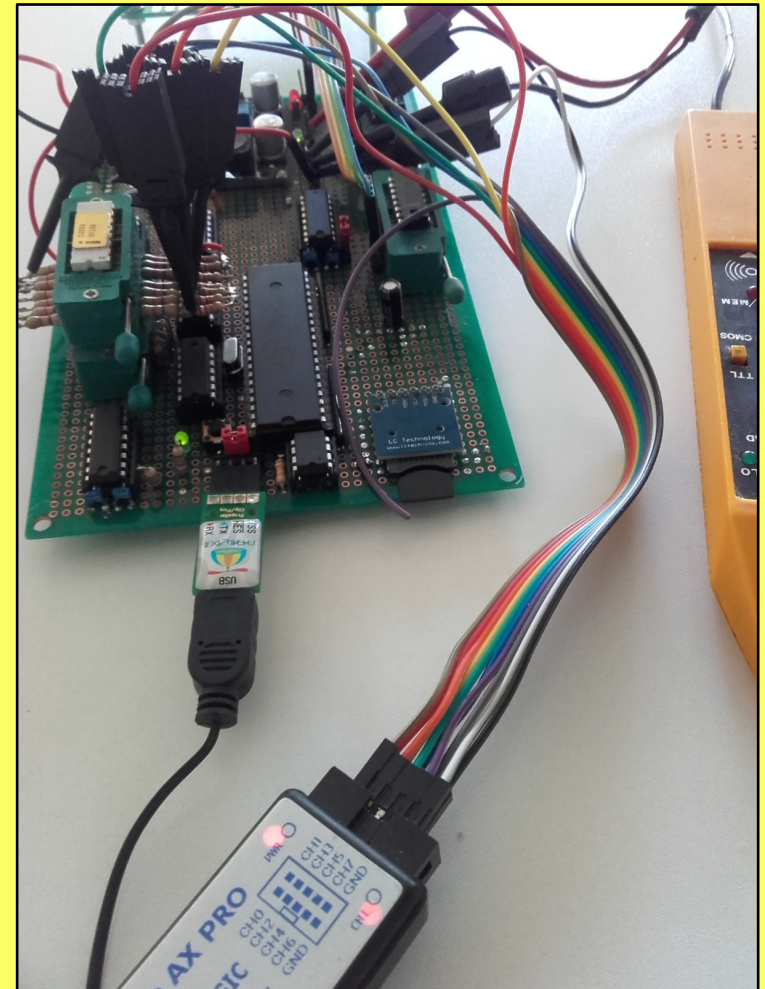
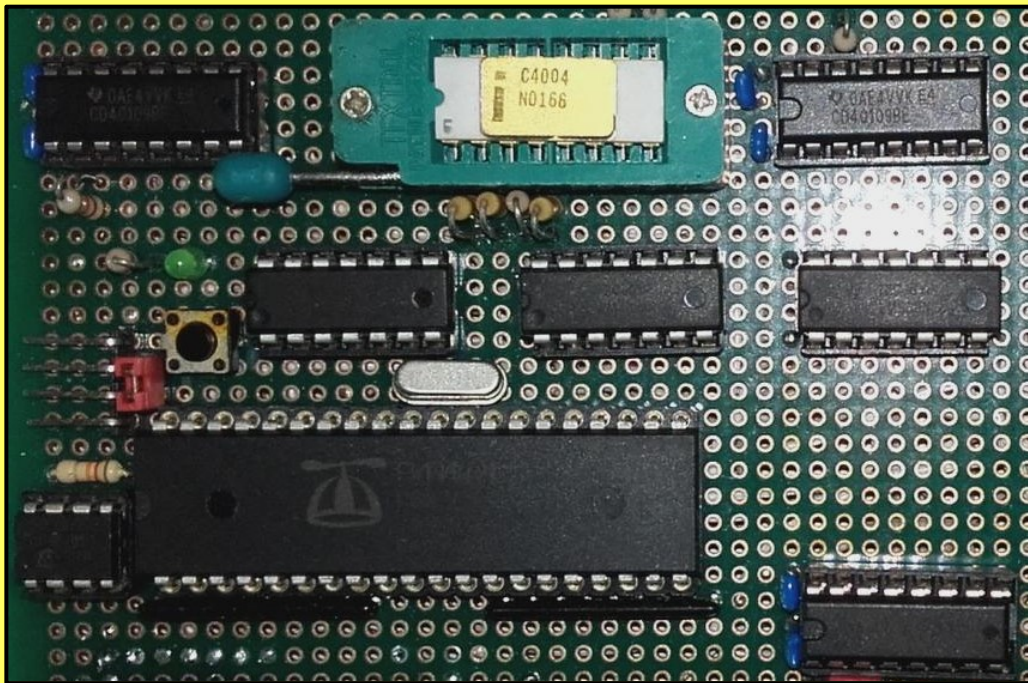
(1) IO instructions control the flow of information between accumulator in CPU, I/O lines in ROM's and RAM's and RAM storage. IOR stands for IO Read. In this case the CPU will receive data from RAM storage locations or I/O input lines of 4001's.

(2) The SRC instruction designates the chip number and address for a following IO instruction.

Figure 2. MCS-4 Basic Instruction Cycle

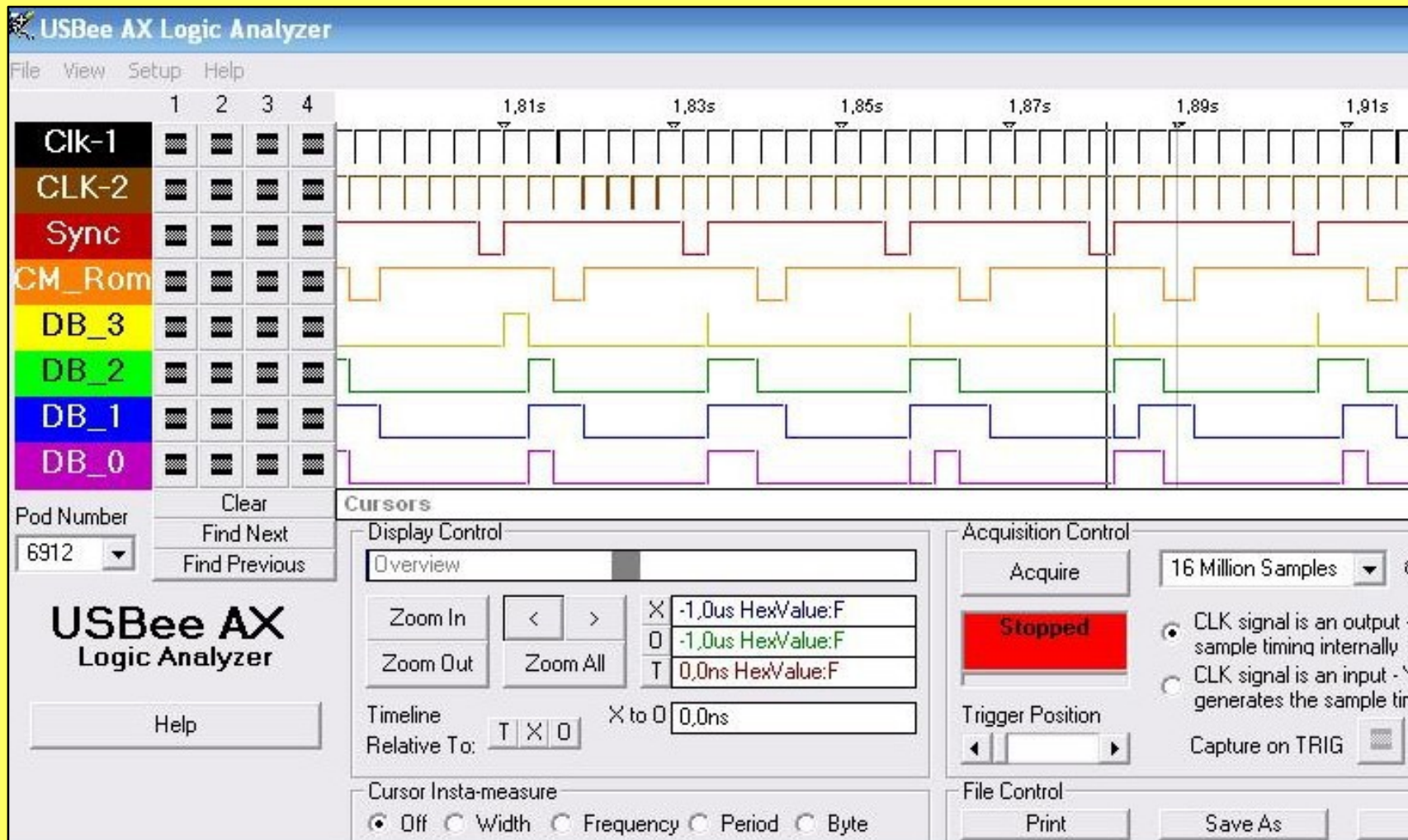
[-Mijn C4004 1-]

- Propellerchip als ROM en klok.



[-Mijn C4004 2-]

- Klok --> Sync resultaat



[-Mijn C4004 3-]

- NOP, NOP, NOP, NOP, Jmp naar 0

[Those instructions preceded by an asterisk (*) are 2 word instructions that occupy 2 successive locations in ROM]

MACHINE INSTRUCTIONS (Logic 1 = Low Voltage = Negative Voltage; Logic 0 = High Voltage = Ground)

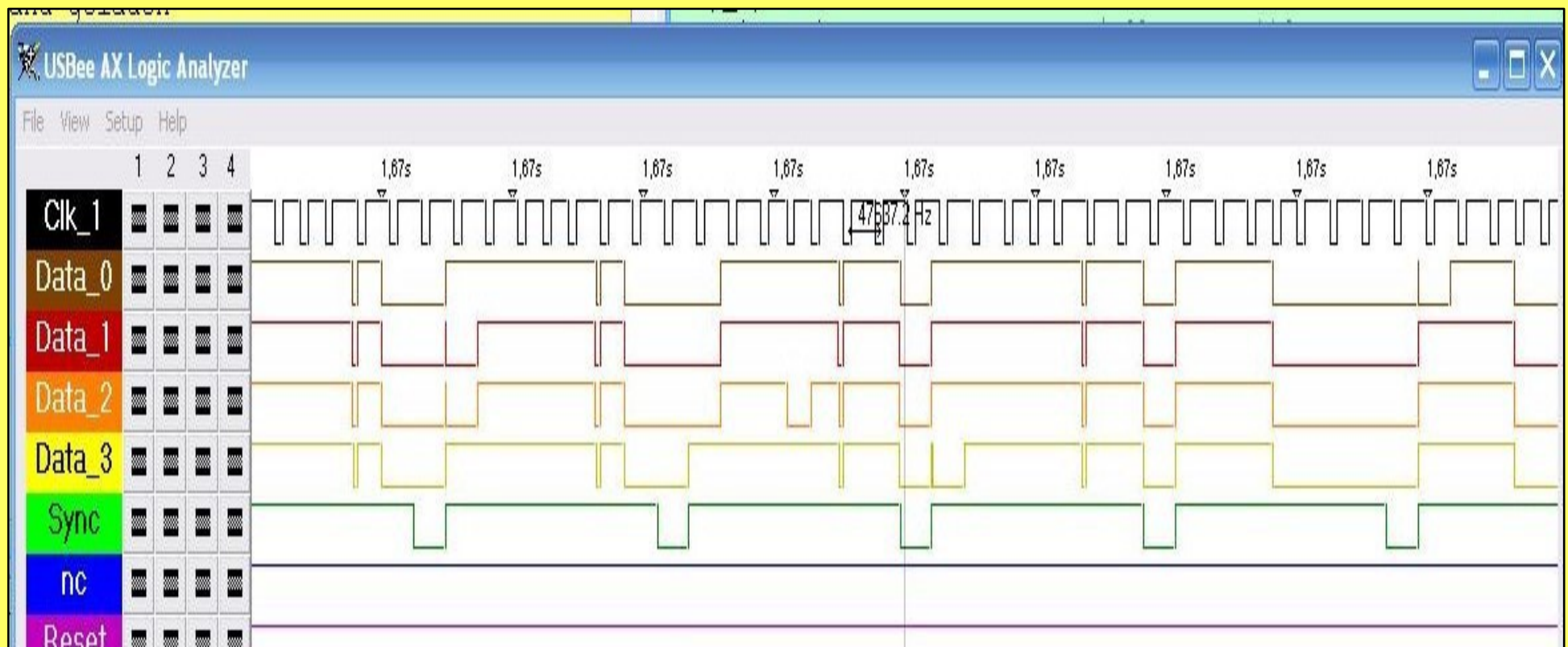
MNEMONIC	OPR D ₃ D ₂ D ₁ D ₀	OPA D ₃ D ₂ D ₁ D ₀	DESCRIPTION OF OPERATION
NOP	0 0 0 0	0 0 0 0	No operation.
*JCN	0 0 0 1 A ₂ A ₂ A ₂ A ₂	C ₁ C ₂ C ₃ C ₄ A ₁ A ₁ A ₁ A ₁	Jump to ROM address A ₂ A ₂ A ₂ A ₂ , A ₁ A ₁ A ₁ A ₁ (within the same ROM that contains this JCN instruction) if condition C ₁ C ₂ C ₃ C ₄ ⁽¹⁾ is true, otherwise skip (go to the next instruction in sequence).
*FIM	0 0 1 0 D ₂ D ₂ D ₂ D ₂	R R R 0 D ₁ D ₁ D ₁ D ₁	Fetch immediate (direct) from ROM Data D ₂ , D ₁ to index register pair location RRR. ⁽²⁾
SRC	0 0 1 0	R R R 1	Send register control. Send the address (contents of index register pair RRR) to ROM and RAM at X ₂ and X ₃ time in the Instruction Cycle.
FIN	0 0 1 1	R R R 0	Fetch indirect from ROM. Send contents of index register pair location 0 out as an address. Data fetched is placed into register pair location RRR.
JIN	0 0 1 1	R R R 1	Jump indirect. Send contents of register pair RRR out as an address at A ₁ and A ₂ time in the Instruction Cycle.
*JUN	0 1 0 0 A ₂ A ₂ A ₂ A ₂	A ₃ A ₃ A ₃ A ₃ A ₁ A ₁ A ₁ A ₁	Jump unconditional to ROM address A ₃ , A ₂ , A ₁ .
*JMS	0 1 0 1 A ₂ A ₂ A ₂ A ₂	A ₃ A ₃ A ₃ A ₃ A ₁ A ₁ A ₁ A ₁	Jump to subroutine ROM address A ₃ , A ₂ , A ₁ , save old address. (Up 1 level in stack.)
INC	0 1 1 0	R R R R	Increment contents of register RRRR. ⁽³⁾
*ISZ	0 1 1 1 A ₂ A ₂ A ₂ A ₂	R R R R A ₁ A ₁ A ₁ A ₁	Increment contents of register RRRR. Go to ROM address A ₂ , A ₁ (within the same ROM that contains this ISZ instruction) if result ≠ 0, otherwise skip (go to the next instruction in sequence).
ADD	1 0 0 0	R R R R	Add contents of register RRRR to register RRRR.

[-Mijn C4004 3-]

- NOP, NOP, NOP,
- NOP, Jmp naar 0

```
source code:  
NOP  
NOP  
NOP  
NOP  
NOP  
JUN $0000
```

```
object code:  
00 00 00 00 00 40 00
```



[-Mijn C4004 4-]

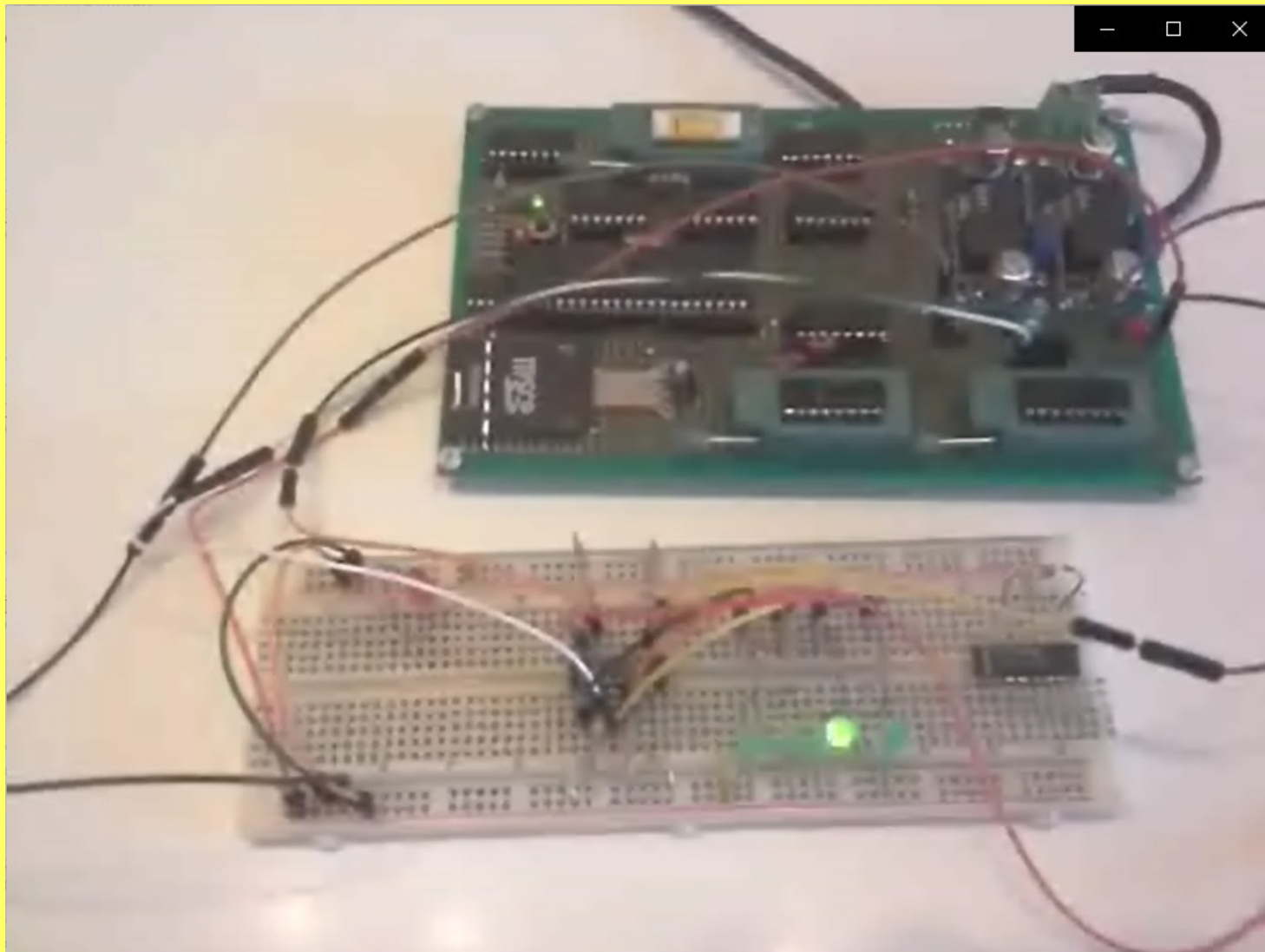
- RAM als uitgang

```
; Toggel output op RAM

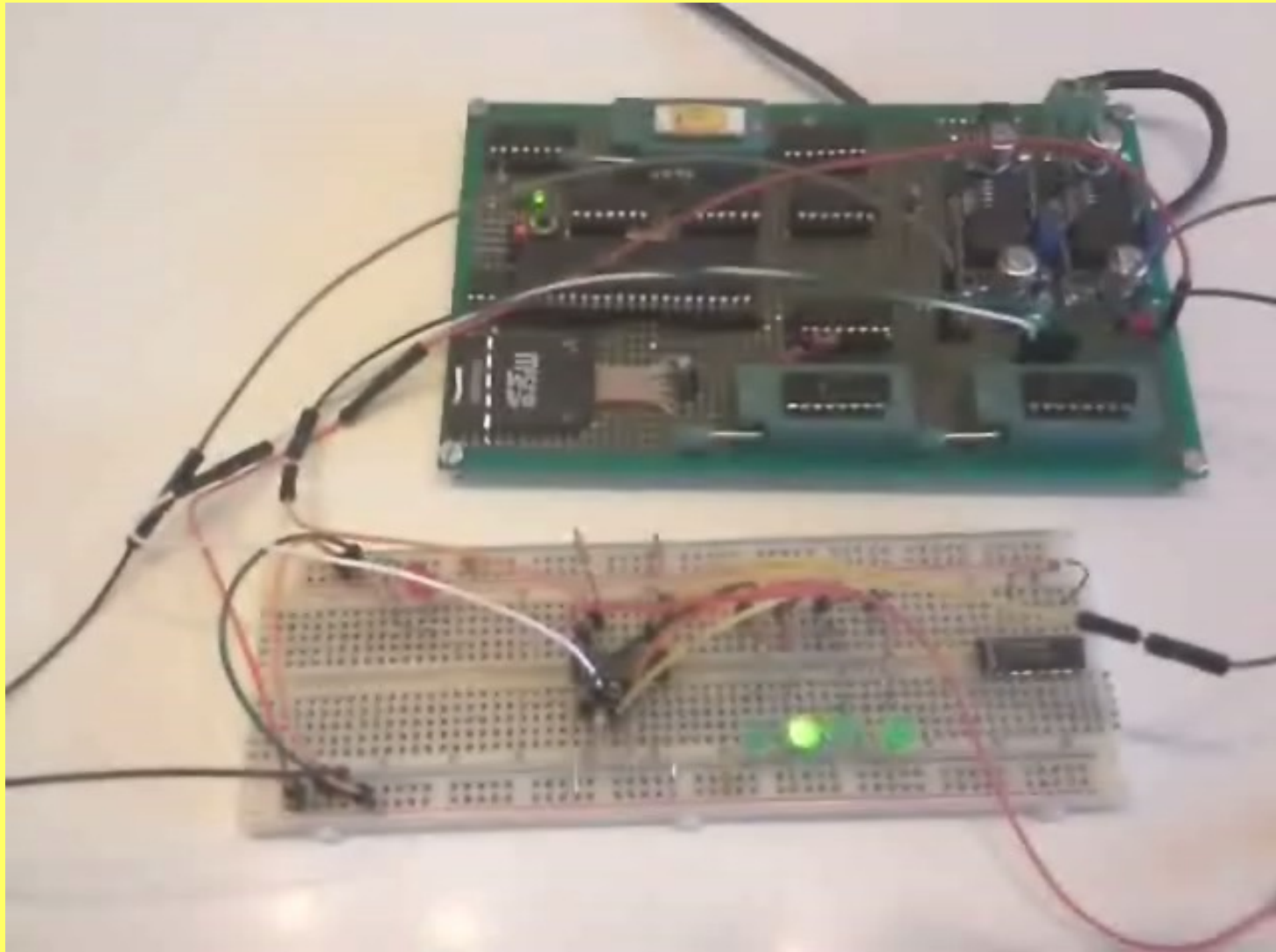
START
    FIM R2R3, $40    ; RAM uitgang adres

BEGIN
    SRC R2R3          ; laad RAM adres
    LDM 0              ; laad accu met waarde
    WMP                ; zet op RAM-IO
    LDM 15             ; laad accu met waarde
    WMP                ; zet op RAM-IO
    JUN BEGIN
```

[-Mijn C4004 4-]



[-Mijn C4004 4-]



[-Mijn C4004 5-]

- ROM als ingang --> naar RAM

```
;      Lees ROM, schrijf naar RAM

START
      FIM R0R1, $10      ; ROM_1 adres
      FIM R2R3, $40      ; RAM   adres

BEGIN
      CLB                ; Clear accu
      SRC R0R1           ; R0R1 naar ROM/RAM
      RDR                ; Lees input ROM in accu
      NOP                ; Niets
      NOP
      SRC R2R3           ; R2R3 naar ROM/RAM
      WMP                ; Schrijf accu naar RAM
      NOP
      NOP
      NOP
      JUN BEGIN          ; Spring naar begin
```


[-Mijn C4004 5-]

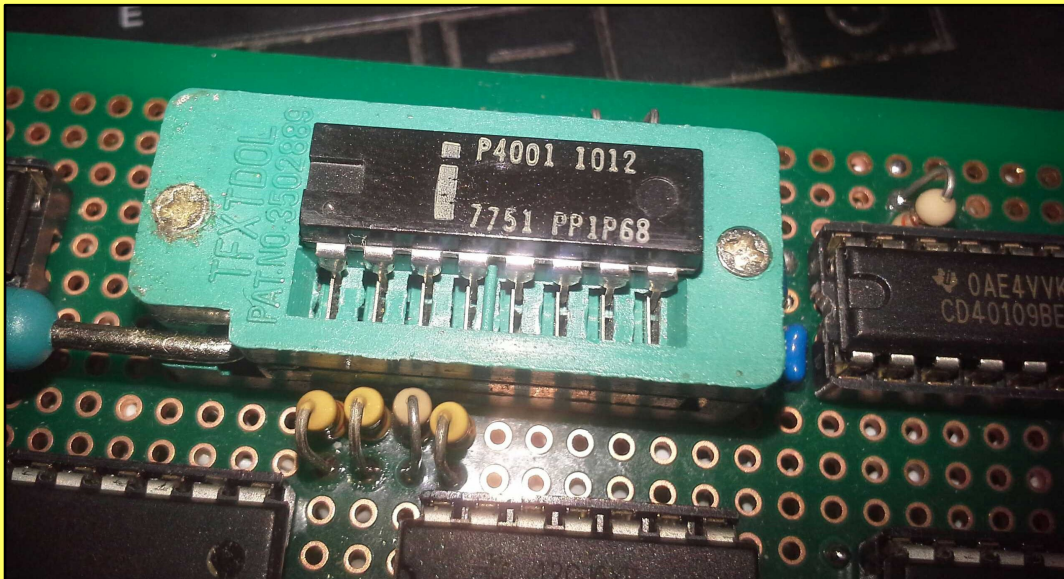


[-Mijn C4004 5-]



[-Mijn C4004 5-]

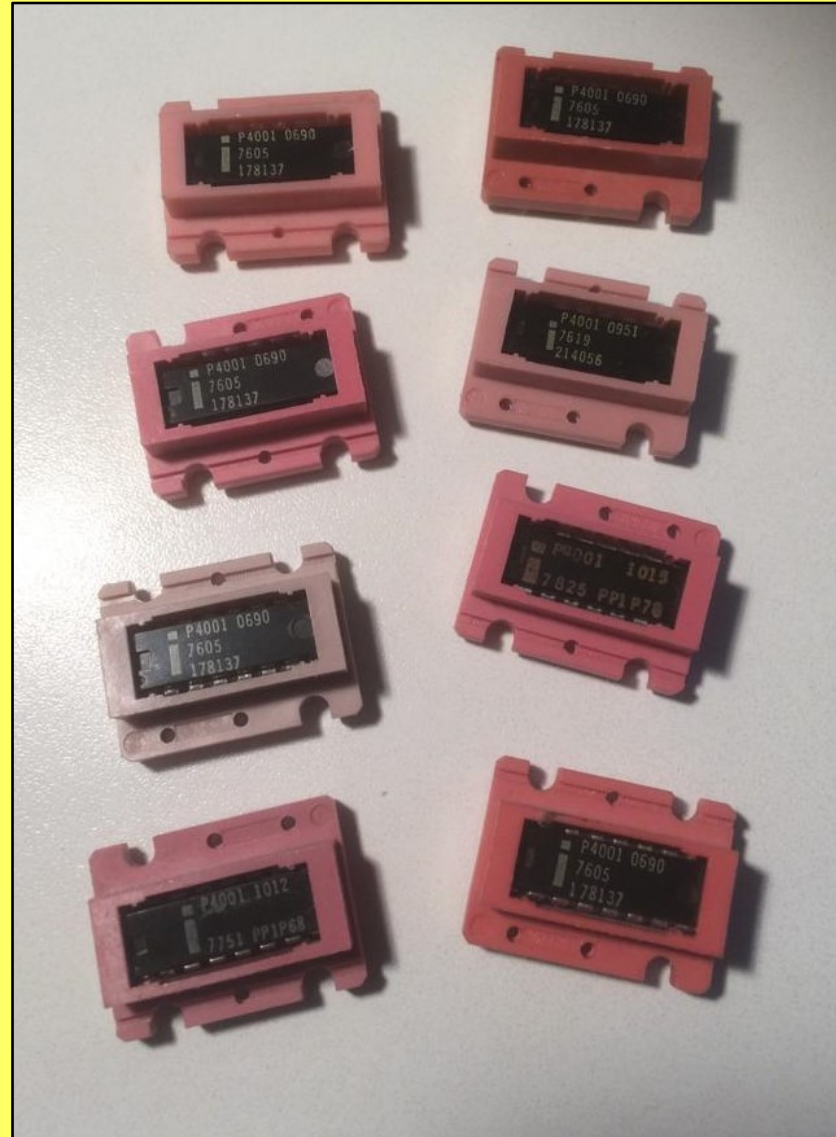
- ROM als ingang,
problemen:
 - Adressering !!!!



1100	0000	0111	0000	0000
1101	0000	0111	0000	0000
1110	0000	0111	0000	0000
1111	0000	0111	0000	0000
0000	0000	1000	0100	1000
0001	0000	1000	0000	1101
0010	0000	1000	0000	0000
0011	0000	1000	0000	0000
0100	0000	1000	0100	1000
0101	0000	1000	0000	1101
0110	0000	1000	0000	0000
0111	0000	1000	0000	0000
1000	0000	1000	0100	1000
1001	0000	1000	0011	0000
1010	0000	1000	0000	0000
1011	0000	1000	0100	1000
1100	0000	1000	0011	0000
1101	0000	1000	1101	0001
1110	0000	1000	1111	1101
1111	0000	1000	0101	1000
0000	0000	1001	0000	0000

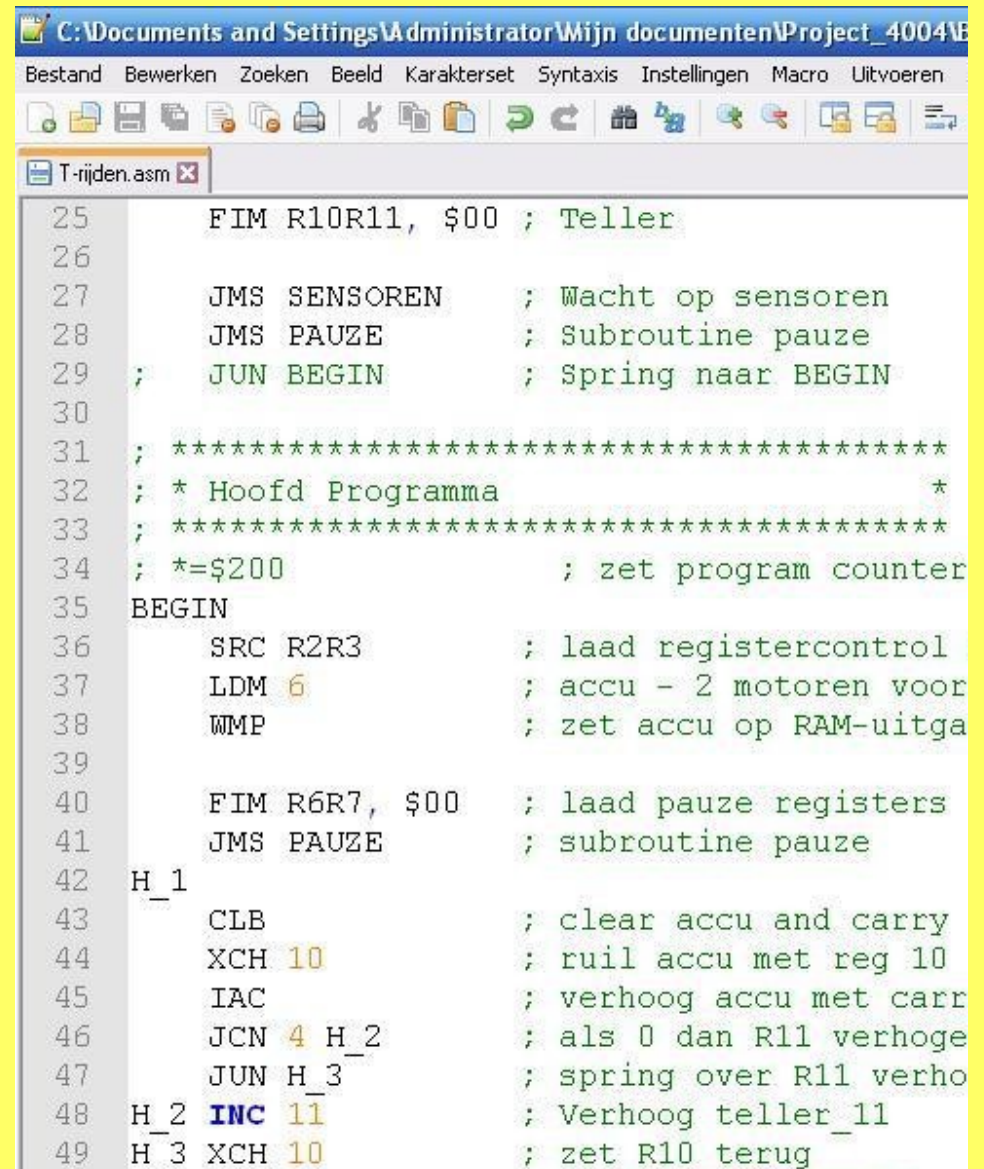
[-Mijn C4004 5-]

- 10 stuks gekocht
- NIET ROM-0
- En 4 ingangen
- Geen garantie !!



[-Mijn C4004 6-]

- Programma flow:
 - 1: Notepad, in assembler een programma maken



The screenshot shows a Notepad window titled 'C:\Documents and Settings\Administrator\Mijn documenten\Project_4004\T-rijden.asm'. The window contains assembly code for a program named 'T-rijden.asm'. The code is written in a mix of uppercase and lowercase letters, with some words in green and others in black. The code is organized into sections, with labels like 'H_1', 'H_2', and 'H_3' used to mark specific points in the program. The code includes instructions for setting up registers, loading data, and performing operations like clearing the accumulator and carrying over values. Comments in Dutch provide context for the instructions, such as 'Wacht op sensoren' (Wait for sensors) and 'Spring naar BEGIN' (Jump to BEGIN).

```
25      FIM R10R11, $00 ; Teller
26
27      JMS SENSOREN    ; Wacht op sensoren
28      JMS PAUZE       ; Subroutine pauze
29      ; JUN BEGIN     ; Spring naar BEGIN
30
31      ; *****
32      ; * Hoofd Programma
33      ; *****
34      ; *=$200         ; zet program counter
35      BEGIN
36      SRC R2R3        ; laad registercontrol
37      LDM 6           ; accu - 2 motoren voor
38      WMP             ; zet accu op RAM-uitga
39
40      FIM R6R7, $00   ; laad pauze registers
41      JMS PAUZE       ; subroutine pauze
42      H_1
43      CLB             ; clear accu and carry
44      XCH 10          ; ruil accu met reg 10
45      IAC             ; verhoog accu met carr
46      JCN 4 H_2       ; als 0 dan R11 verho
47      JUN H_3         ; spring over R11 verho
48      H_2 INC 11       ; Verhoog teller_11
49      H_3 XCH 10       ; zet R10 terug
```

[-Mijn C4004 6-]

- Programma flow:
 - 2: Assembleren: e4004.szyc.org/asm.html

Intel 4004 Microprocessor Assembler home

source code:

```
; *****  
; * Lijn_volgen.asm av_retro AV_Tech *  
; *  
; * Processor: Intel C4004 *  
; * Robot: Intel-Inside *  
; * SD-kaart: 5 *  
; *  
; * d20161101 Begin Lijn_volgen *  
; *  
; *****  
; *
```

object code:

```
20 10 22 40 24 00 26 00  
50 37 50 2E 23 D6 E1 21  
EA F1 F6 12 25 F6 F1 F6  
12 1C 40 0C 23 D2 E1 26  
0E 50 2E 40 0C 23 D4 E1  
26 0E 50 2E 40 0C 74 2E  
75 2E 76 2E 77 2E C0 F0  
21 EA 14 37 C0
```

listing: \$003C

```
pass 1: done  
pass 2  
  
0000: START  
0000: FIM R0R1 $10 20 10  
0002: FIM R2R3 $40 22 40  
0004: FIM R4R5 $00 24 00  
0006: FIM R6R7 $00 26 00  
0008: JMS SENSOR 50 37  
000A: JMS PAUZE 50 2E  
000C: BEGIN  
000C: SRC R2R3 23
```

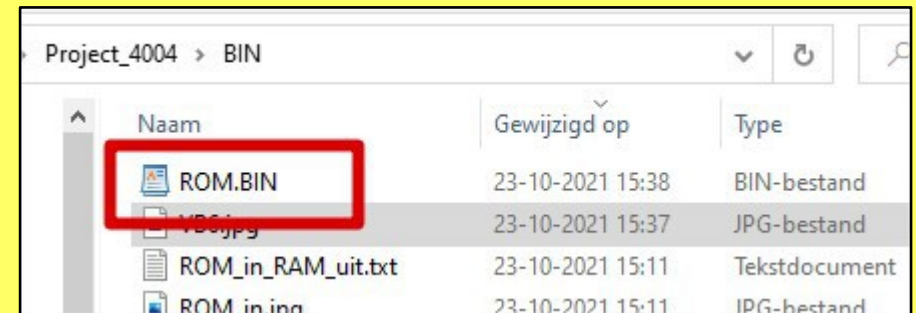
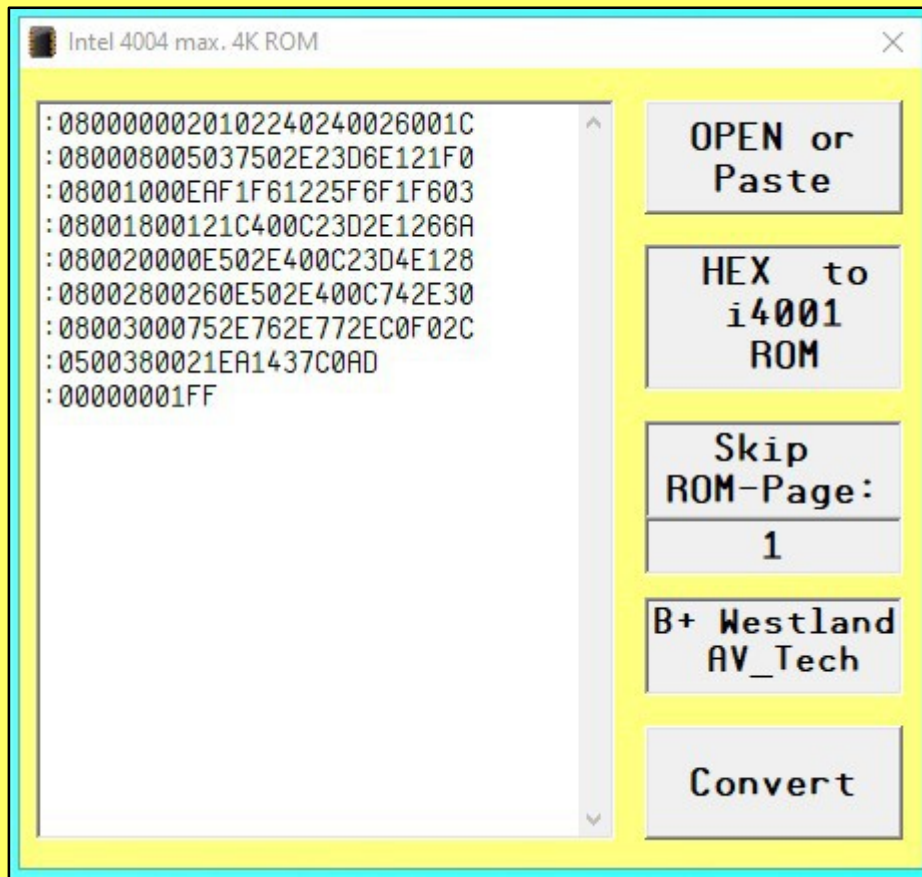
hex code:

```
:0800000020102240240026001C  
:080008005037502E23D6E121F0  
:08001000EAF1F61225F6F1F603  
:08001800121C400C23D2E1266A  
:080020000E502E400C23D4E128  
:08002800260E502E400C742E30  
:08003000752E762E772EC0F02C  
:0500380021EA1437C0AD  
:00000001FF
```

generate code

[-Mijn C4004 6-]

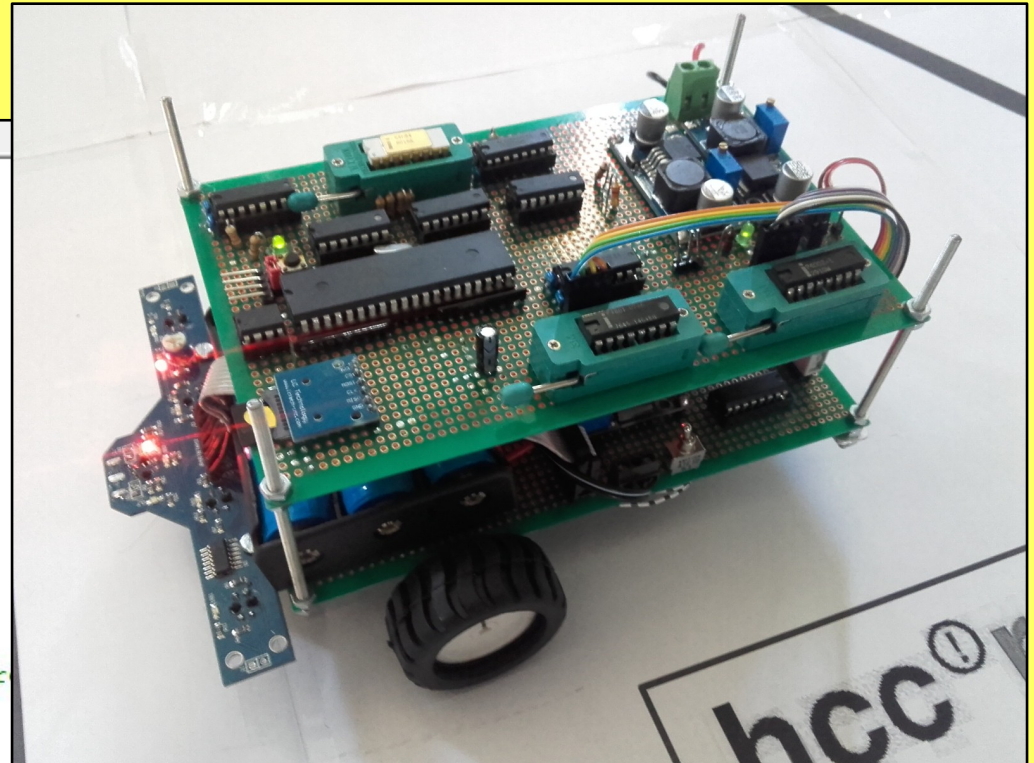
- Programma flow:
 - 3: Omzetten in binaire bestand via VB programma



[-Mijn C4004 7-]

- Resultaat: Lijnvolger.

```
Lijn_volgen.asm
16 ; *****
17 ; * Initialisatie van Registers *
18 ; *****
19 START
20     FIM R0R1, $10    ; ROM-1 ingang adres
21     FIM R2R3, $40    ; RAM-1 uitgang adres
22     FIM R4R5, $00    ; PAUZE registers
23     FIM R6R7, $00    ; PAUZE registers
24
25     JMS SENSOREN     ; Wacht op sensoren
26     JMS PAUZE        ; Subroutine pauze
27
28 ; *****
29 ; * Hoofd Programma *
30 ; *****
31 BEGIN
32     SRC R2R3          ; laad registercontrol RAM-adres
33     LDM 6              ; accu - 2 motoren vooruit
34     WMP               ; zet accu op RAM-uitgang
35
36     SRC R0R1          ; laad ROM adres
37     RDR              ; Lees input ROM in accu
38     CLC              ; Clear carry
39     RAR              ; Rotate accu right door carry (1e bit)
40     JCN 2 H_LM        ; test op accu=0/c=1
41 ;                   ; ja -> jump -> lbl
42 ;                   ; 2=carry, 4=accu
43     RAR              ; Rotate accu right door carry (2e bit)
44     CLC              ; Clear carry
45     RAR              ; Rotate accu right door carry (3e bit)
46     JCN 2 H_RM        ; test op accu=0/c=1
47     JUN BEGIN
```



[-Info intel-]

- Bron vermelding:
 - Intel - 4004 op internet
 - <https://frank-buss.de/4004/>
 - e4004.szyc.org/asm.html
 - <https://avretro.wordpress.com/>

hcc! retro



**50 Jaar intel 4004
microprocessor
1971 - 2021**