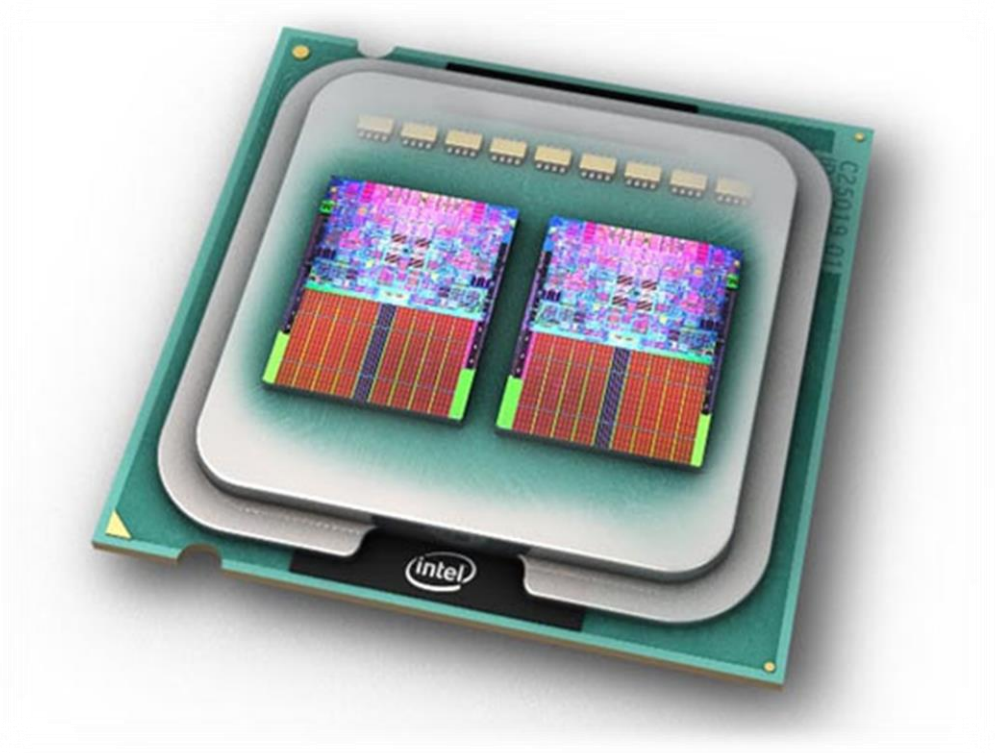


Microcontrollers Inleiding

Auteur: Frank Marchal
Versie: 0.6
Datum: 25/01/2021
Doelgroep: 3de graad elektronica

Inleiding

In deze cursus gaan we de microcontroller (µc) aan de binnenkant proberen te begrijpen. Voor een elektronicus is het van uiterst belang dat we begrijpen wat er allemaal in dat zwarte blokje omgaat.

Eerst gaan we even de tijd terug in om te weten te komen **vanwaar de "chips" vandaan komen.**

Daarna leren we kort de **µc geschiedenis** kennen. Wie ontwierp de eerste microcontroller en hoe ging het dan verder tot de dag van vandaag.

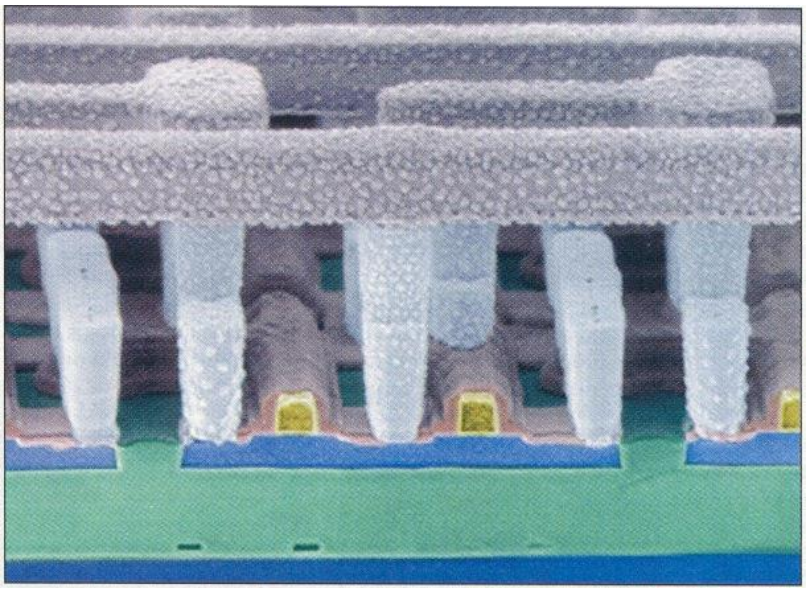
Daarna duiken we in de **binnenkant van de µc** en proberen we te achterhalen hoe alles er samenwerkt. Hoe kunnen we een controller aan het werk zetten en onze data laten versturen of opdrachten laten uitvoeren?

Je komt het allemaal te weten in deze cursus.

Ik dank bij deze ook Peter Dams voor de extra info die hij gaf om deze cursus structuur te geven.

Veel plezier ermee.

Frank Marchal

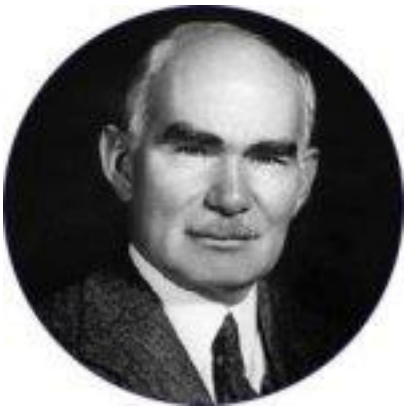


1. IC history

Hier vind je info over de uitvinding en geschiedenis van het IC (integrated circuit):

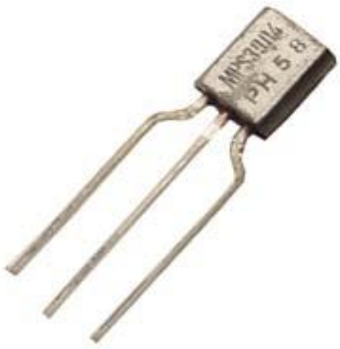
First 50 years of the 20th century: the **vacuum Tube** (invented at **1907** by **Lee De Forest**) was dominating the electronic market.

➔ Problem: fragile, bulky, unreliable, power hungry, and produced considerable heat

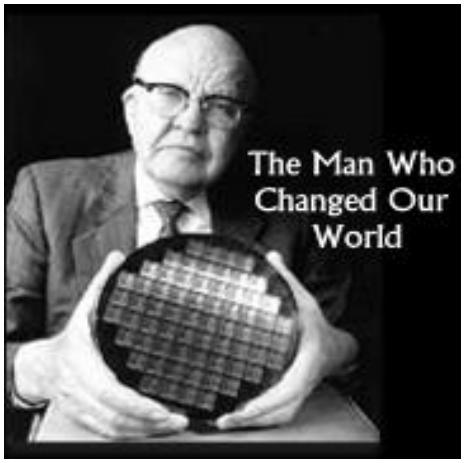


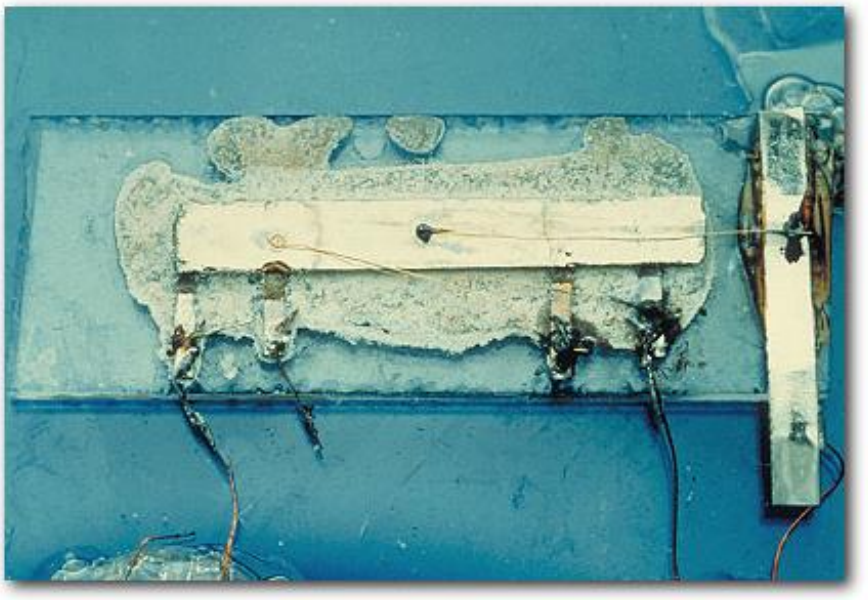
1947: First **Transistor** invented at **Bell** Telephone Laboratories by **John Bardeen and Walter Brattain**.

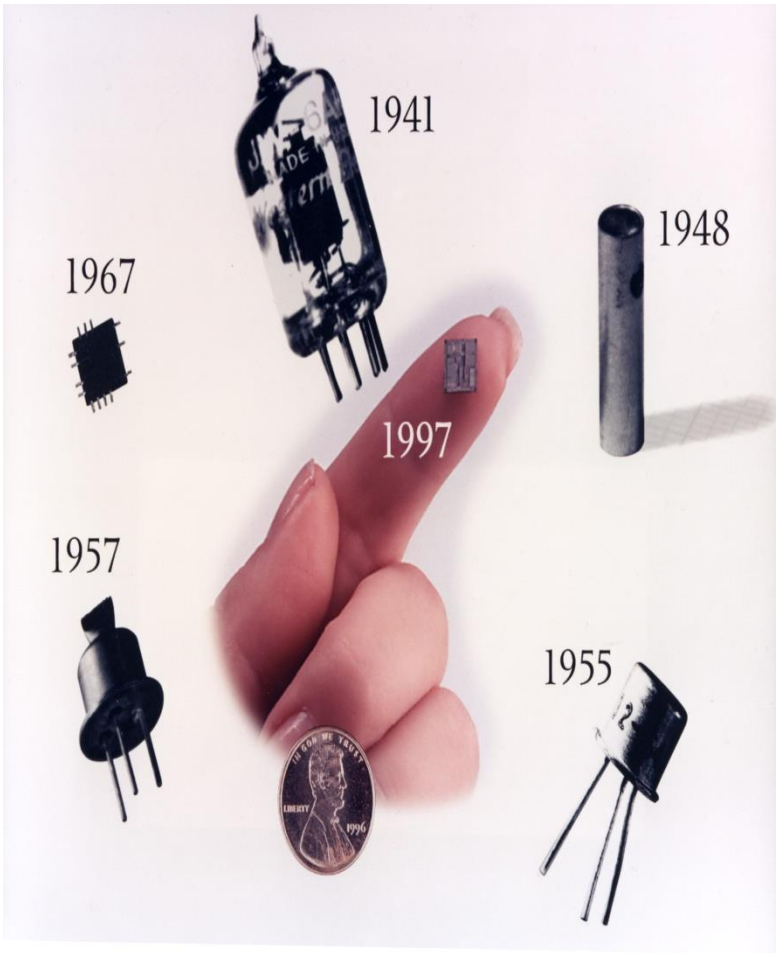
- ➔ Transistors were miniscule in comparison, more reliable, longer lasting, produced less heat, and consumed less power
- ➔ Stimulation for further engineering of more complex electronic circuits
- ➔ Problem: thousand of discrete components should be still soldered by hand, this time-consuming, expensive, and takes a lot of space on a PCB. Also every soldered joint was a potential source of trouble.

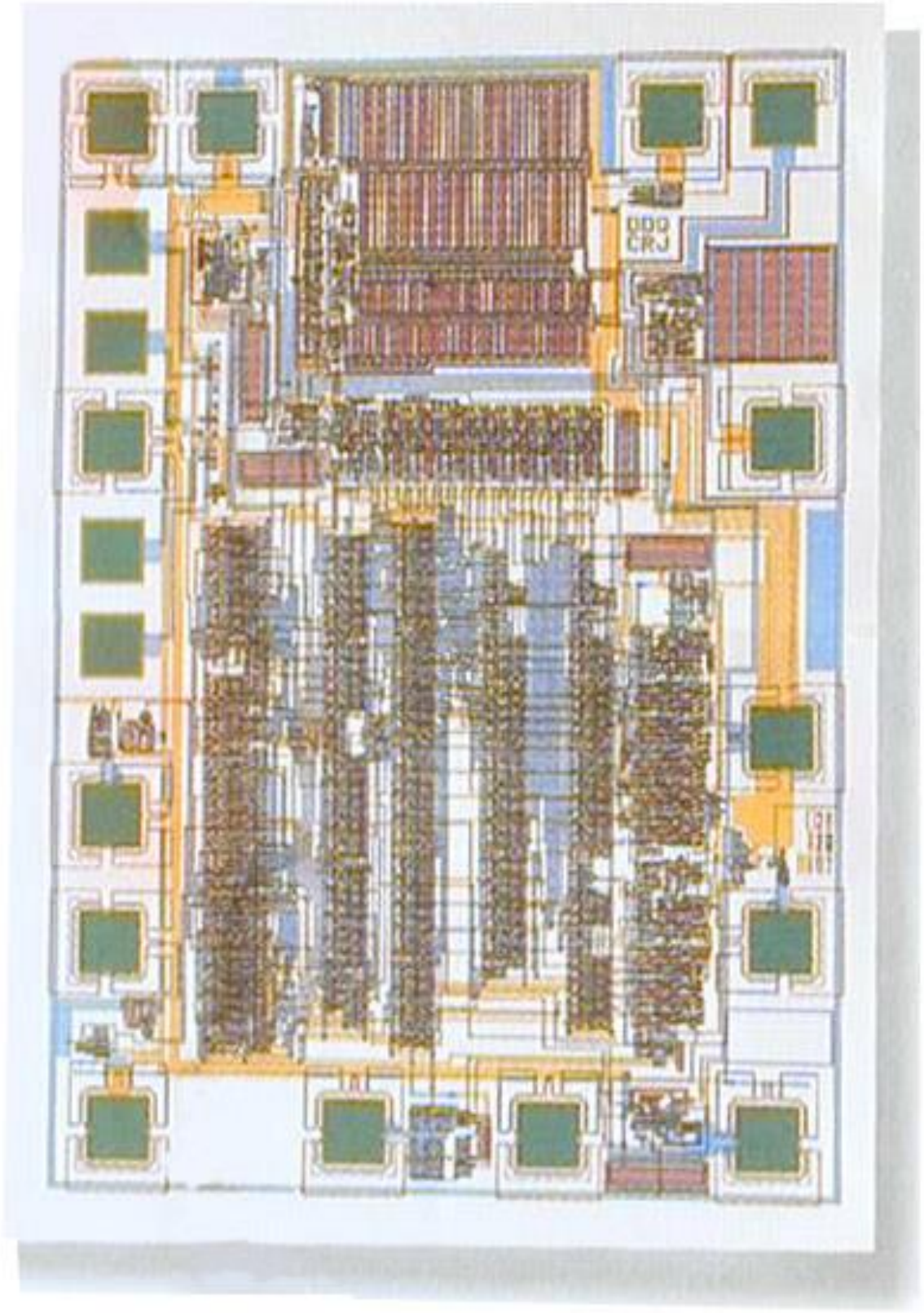


1958: First **Integrated Circuit (IC)** build by **Jack Kilby** at Texas Instruments.

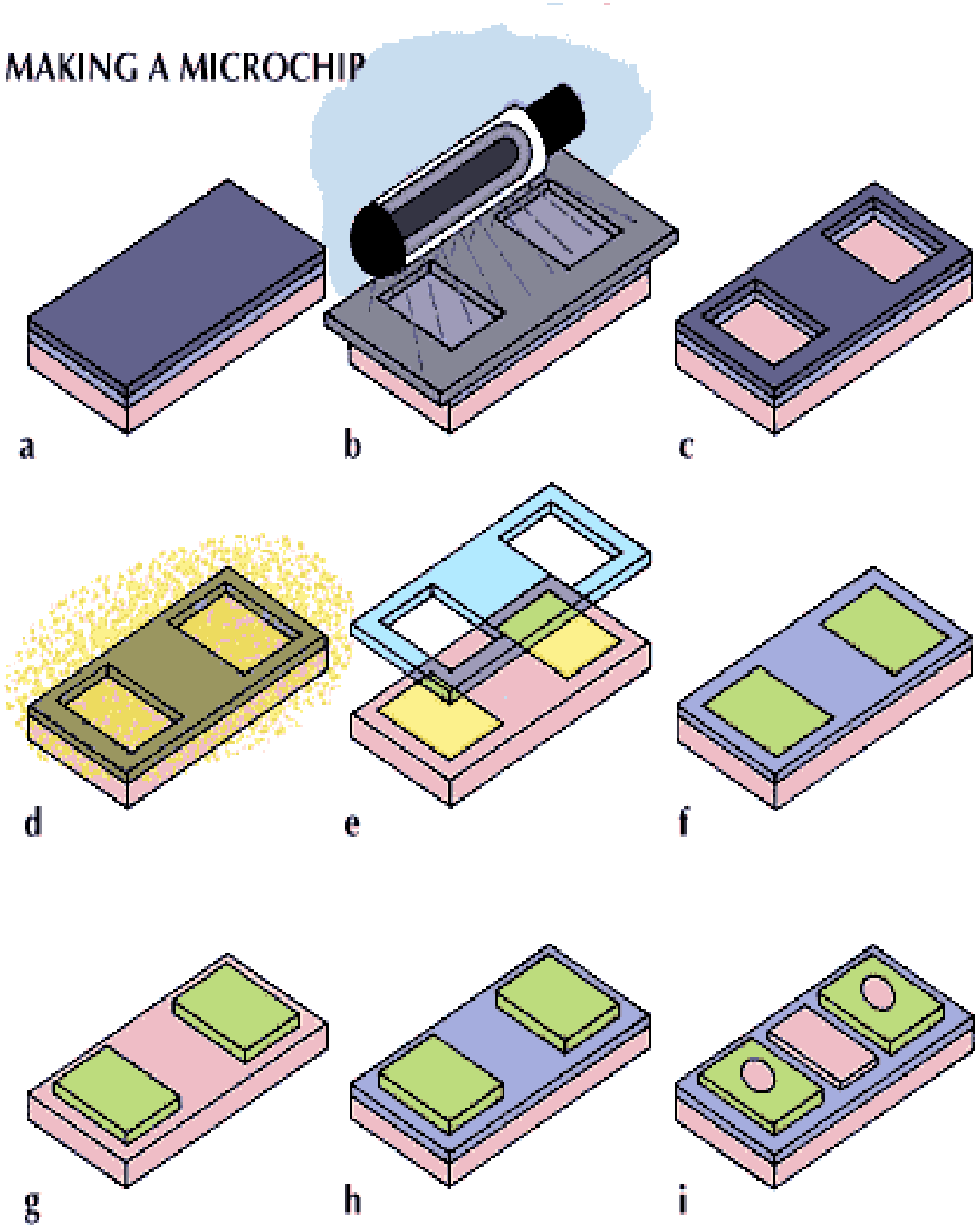






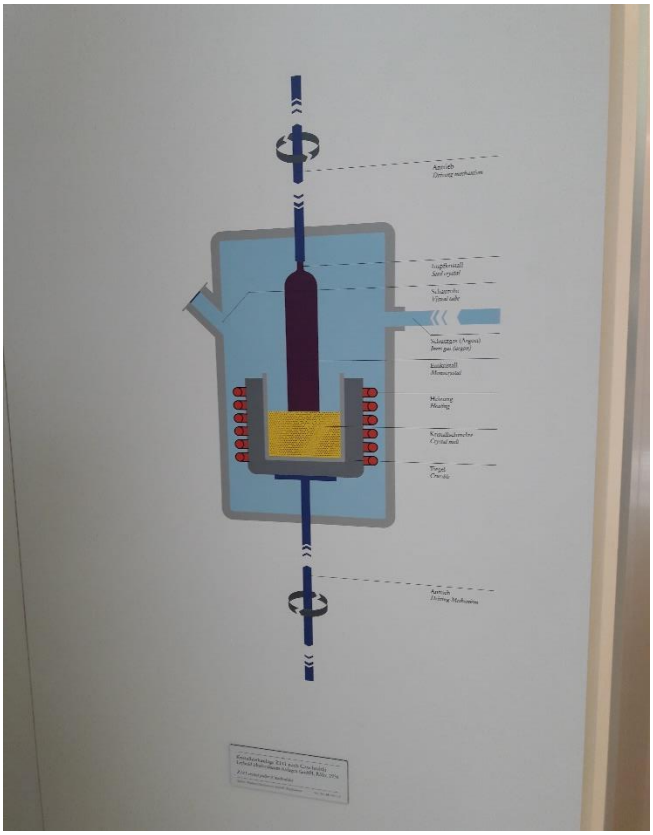


Voorstelling van een DIE (inwendige van een IC waar alle elektronica circuits worden op voorzien).

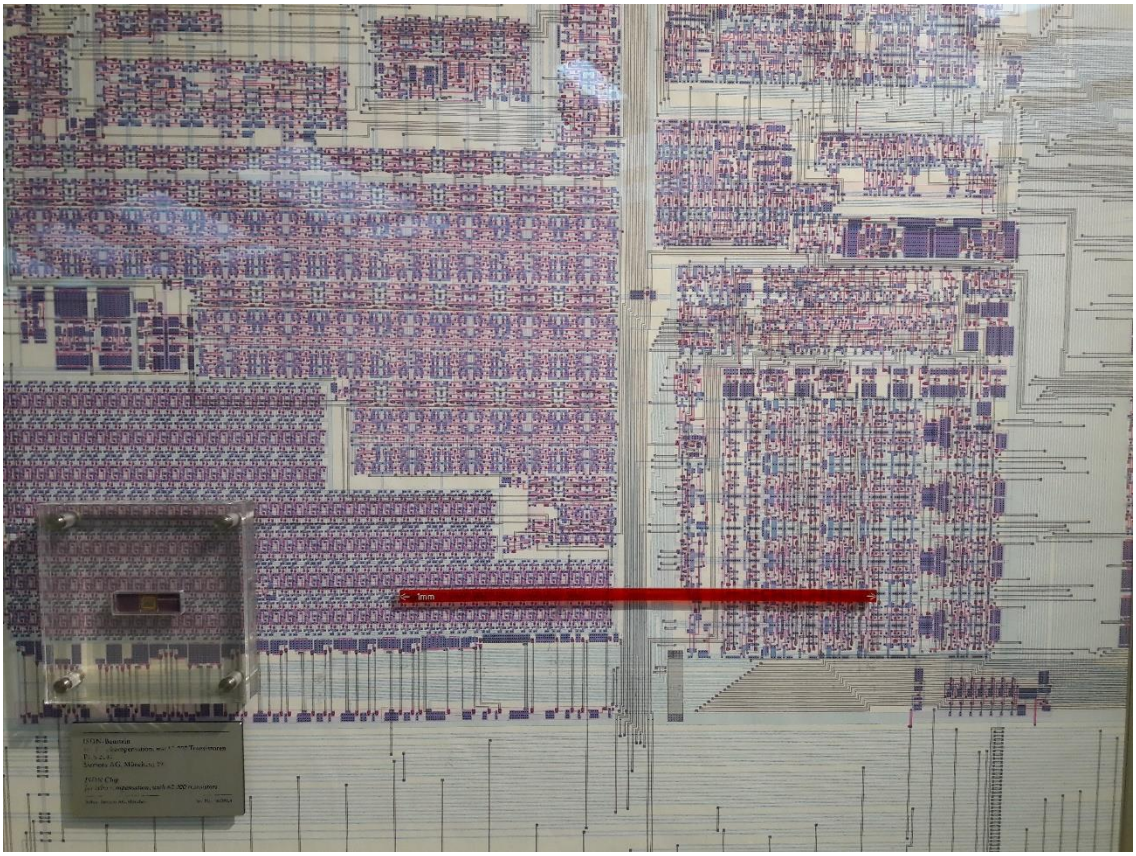


Meer info en filmpjes over deze technologie kan je terugvinden op:

http://newsroom.intel.com/docs/DOC-2476



Op de deze foto's wordt getoond hoe silicium wordt aangemaakt en omgezet tot 1 wafer.



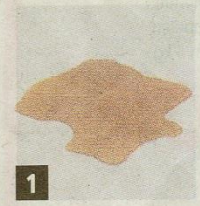
Op deze foto wordt 1mm in het rood aangegeven. Zo kan je je proberen in te beelden hoeveel elektronische circuits er zitten in die 1mm op de chip.

Met deze diamand zaag (30 000 rpm) werden de wafers gesneden. Het zaagblad is 25 µm dik.

meerdere processors in een computer geplaatst. Die verdelen het rekenwerk onder elkaar en maken de computer zo sneller.

Hoe worden chips gemaakt?

Dit is slechts een vereenvoudiging: in werkelijkheid zijn er honderden stappen. Het productieproces vindt plaats in steriele fabrieken die duizend keer properder zijn dan een operatiekamer in een ziekenhuis.

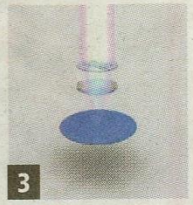


1

Uit zand wordt silicium gehaald. Dat wordt gezuiverd: op een miljard siliciumatomen mag er zich slechts één ander atoom bevinden.

2

Het silicium wordt in een gietvorm gegoten, ongeveer 30cm groot. De gietvorm wordt in plakken gesneden. Die worden gepolijst tot ze zo glad zijn als een spiegel.

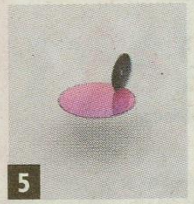


3

Met ultraviolet licht worden fijne patronen in het silicium geëtst. De uitsparingen worden elektrisch geleidend gemaakt. Zo ontstaan transistors. Die regelen waar de elektrische stroom heengaat.

4

Dit is één transistor. Om een idee te geven van de grootte: in het punt op het einde van deze zin passen er 4 miljoen. De transistoren worden verbonden en vormen samen een processor.

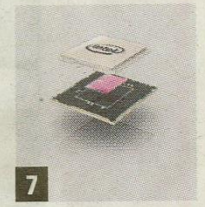


5

De individuele processoren worden getest. Dan worden de plakken silicium versneden. Elk stuk vormt een chip met daarop één of meer processoren.

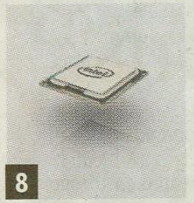
6

Dit is één chip met daarop twee processoren. Ze meet 3 cm en bevat ongeveer 900 miljoen transistoren.



7

De technologie is zo verfijnd dat ze erg gevoelig is aan stof, warmte en vocht. Elke chip krijgt daarom een beschermend jasje.



8

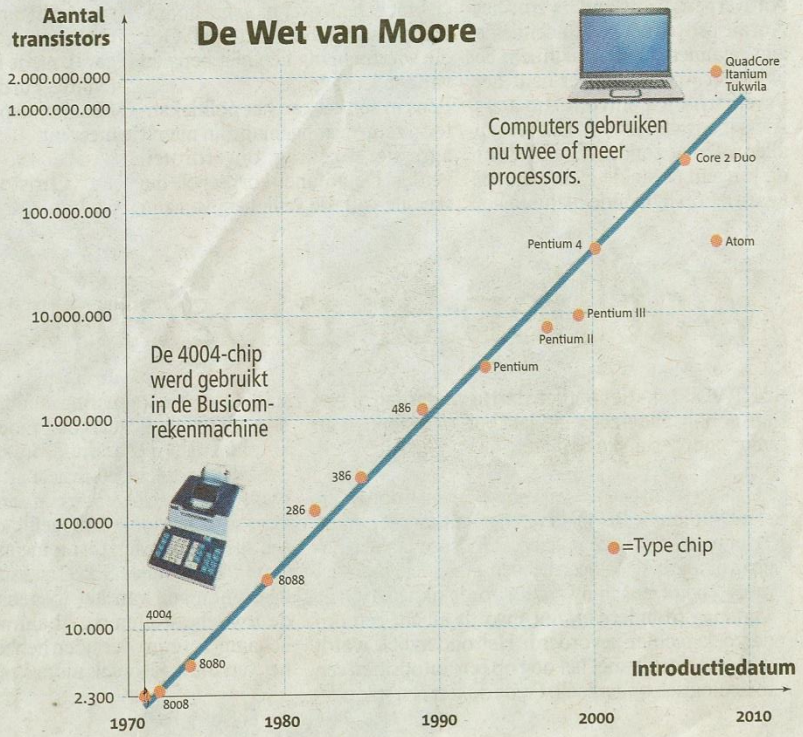
Deze chip is klaar om in een computer te worden gemonteerd. Zonder zo'n chip kan een computer niet rekenen en is hij dus nutteloos.

Infografiek: Ivan De Mey - Samenstelling: Jan Martynowski - Bronnen: Intel, Wikipedia

Wet van Moore

- Gordon Moore, een van de oprichters van Intel, stelde in 1965 dat het aantal transistors op een chip elke twee jaar ongeveer verdubbelt. Dit wordt de Wet van Moore genoemd.
- Doordat computerchips steeds kleiner maar toch krachtiger worden, houdt de Wet van Moore nog steeds stand.
- Eind 2006 kondigde Moore - intussen al met pensioen - aan dat zijn wet niet eeuwig zou meegaan. "Bepaalde limieten kunnen niet overschreden worden", zei hij.
- Tot nu toe houdt de wet nog stand.

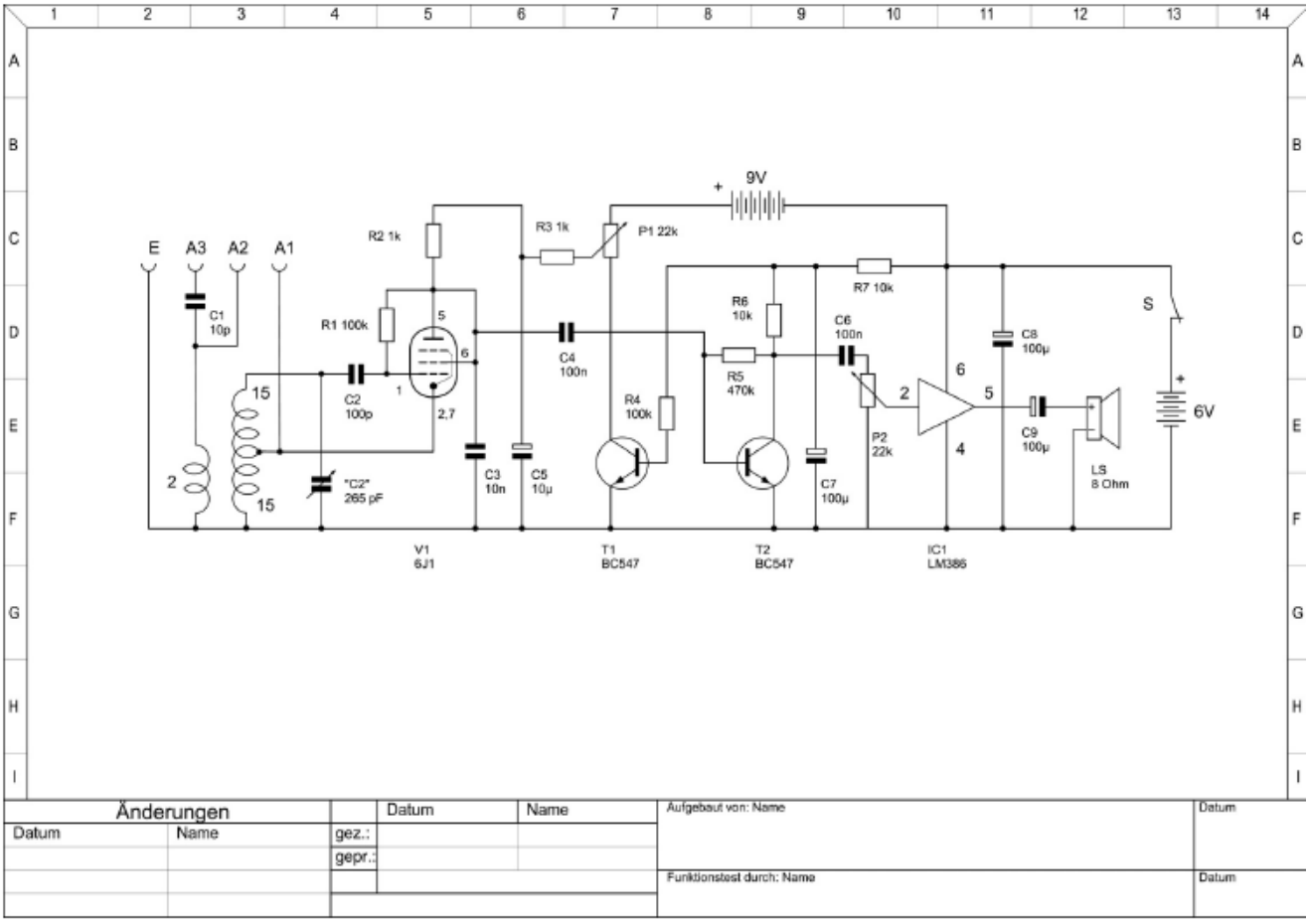
Gordon Moore

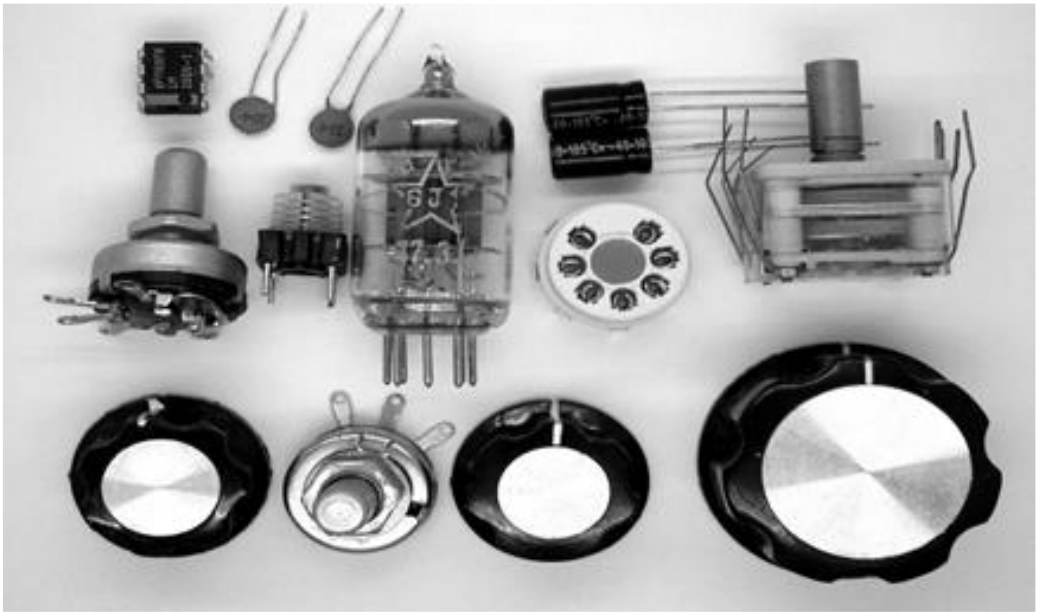


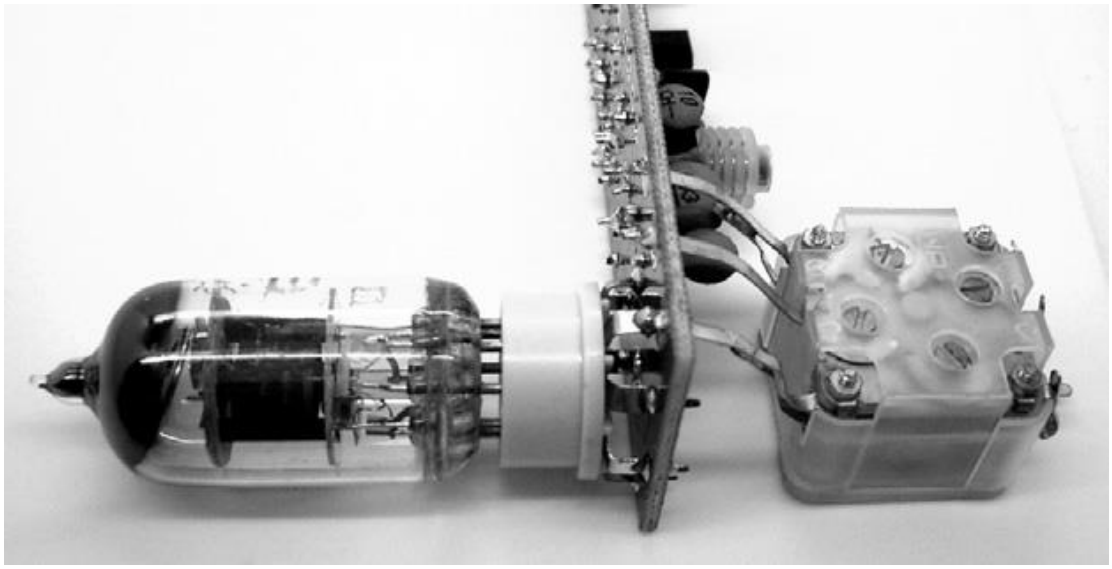
Combinatie van de 3 vorige technologieën:

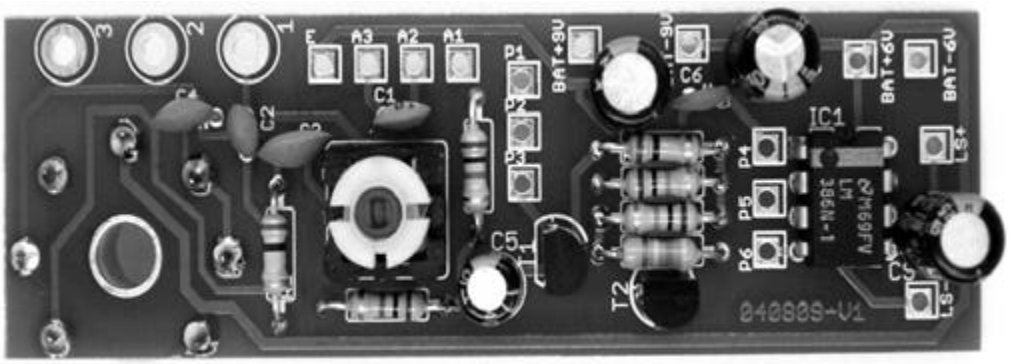
De **zelfbouw AM radio** van Franzis (zie demo)

Duid de 3 verschillende technologieën aan in het schema: buislamp, transistor, opamp (IC).








BAT+9V
BAT-9V
BAT+6V
BAT-6V
IC1
C1
C5
C6
T1
T2
LS+
LS-
040089-V1

2. µC History

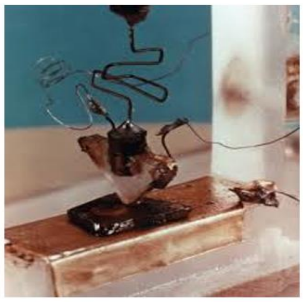
Voordat we spraken van chips bestonden er microcontrollers die zo groot waren **als een gebouw**:

Vacuum tube computers

SAGE Blockhouse/Computer:
10,170m², 250 tons, houses More than 200,000 vacuum tubes @ 3,000,000 Watts

Transistor Computers

- 2nd Generation
- From 1956
- Half a room



The Harwell Dekatron Computer under restoration at the British National Museum of Computing

Invention of ICs

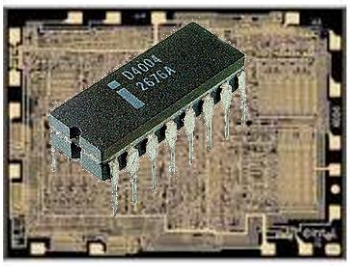
- 3rd generation

IBM 360 made by ICs (1964)

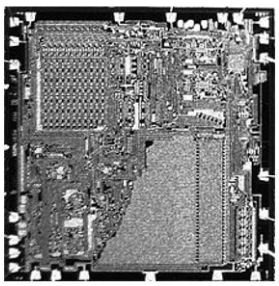
First microprocessors/Microcontrollers

- TI TMS1000
- 4004 (from Intel)
- 6800 (Motorola)

TI TMS1000
(1971-1974)
http://www.antiquetech.com/



Intel 4004 (1971)
www.computerhistory.org

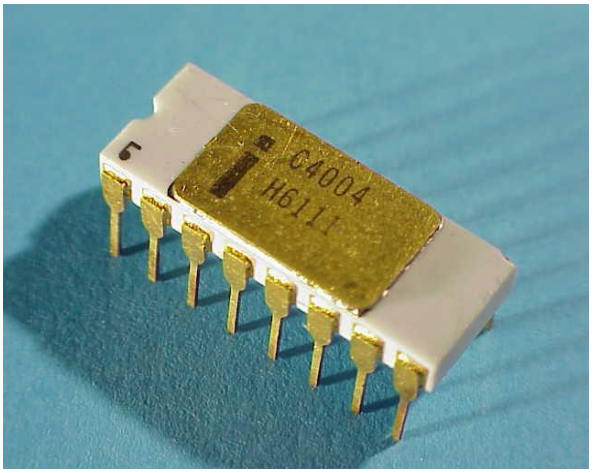


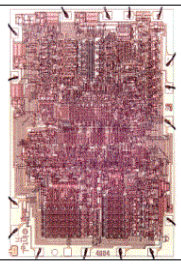
PICO1 (1971)
http://en.wikipedia.org/wiki/Microprocessor

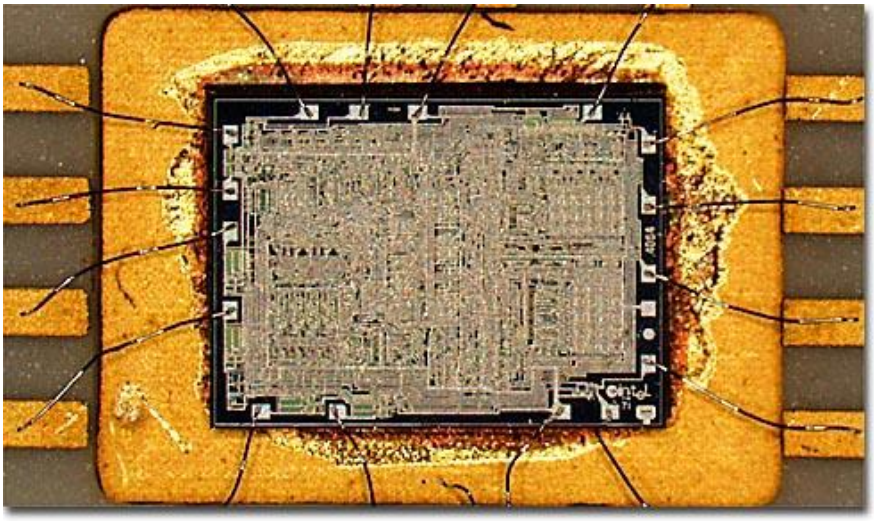
Motorola MC6800
(1974)
http://en.wikipedia.org/wiki/Motorola_6800

De volgende info geeft weer hoe de µC is gegroeid van **een chip voor een kleine rekenmachine tot een heuse krachtige processor of meerdere processors in 1 behuizing**:

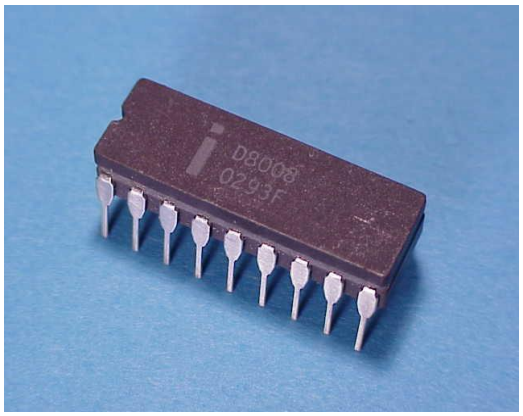
Intel 4004 (first microprocessor chip in 1969 build for datapoint)

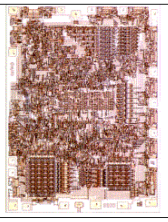


Modello	**4004**
Data di introduzione	15 Novembre 1971
Architettura	4 bit
Larghezza bus	4 bit
Clock	0,108 MHz
MIPS	0,06
Numero di transistor	2.250
Tecnologia	NMOS 10 micron
Spazio di indirizzamento	640b
Layout	

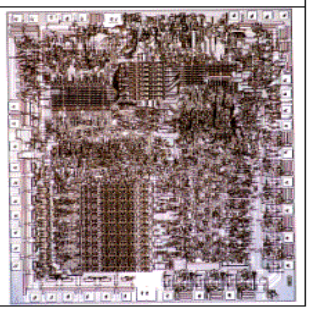


Intel 8008 (same 4004 but other code number)



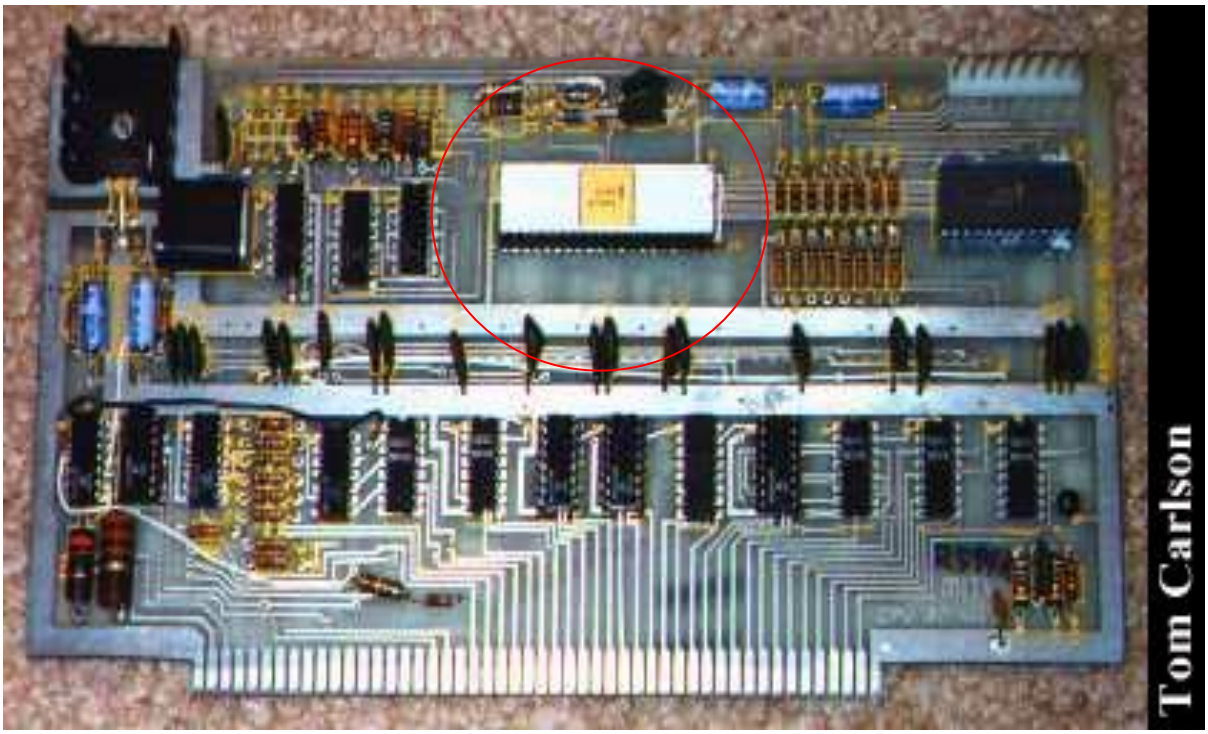
Modello	**8008**
Data di introduzione	1 Aprile 1972
Architettura	8 bit
Larghezza bus	8 bit
Clock	0,200 MHz
MIPS	0,06
Numero di transistor	3.500
Tecnologia	NMOS 10 micron
Spazio di indirizzamento	16kb
Layout	

Intel 8080 (1975, 4500 transistors)

Modello	8080
Data di introduzione	1 Aprile 1974
Architettura	8 bit
Larghezza bus	8 bit
Clock	2 MHz
MIPS	0,64
Numero di transistor	4.500
Tecnologia	NMOS 6 micron
Spazio di indirizzamento	64kb
Layout	

Altair 8800 van MITS (in 1975), eerste microcomputer met de 8080 van Intel

MITS logo



CPU card van Altair met 8080 van Intel

front panel

Paul Allen en Bill Gates maken BASIC programma voor Altair 8080.

Il nastro perforato contenente la versione 1.0
dell'interprete Microsoft BASIC

Eerste BASIC programma op ponsband van MICROSOFT

Paul Allen (seduto) e Bill Gates
alla Lakeside School

Microsoft staff photo, December 7, 1978.

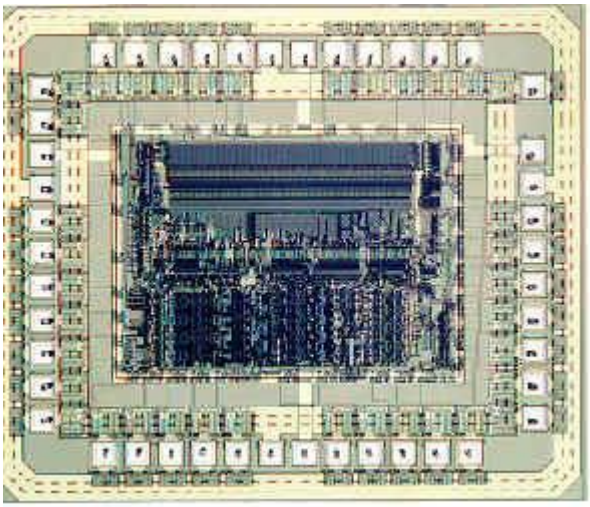
Top row: Steve Wood (left), Bob Wallace, Jim Lane.

Middle row: Bob O'Rear, Bob Greenberg, Marc McDonald, Gordon Letwin.

Bottom row: Bill Gates, Andrea Lewis, Marla Wood, Paul Allen.

MOS Technologies maakt de 6502

Deze wordt het hart van de eerste **APPLE II** personal computer (van Steve Jobs en Steve Wozniak). Ook de **commodore** bevatte deze microprocessor.



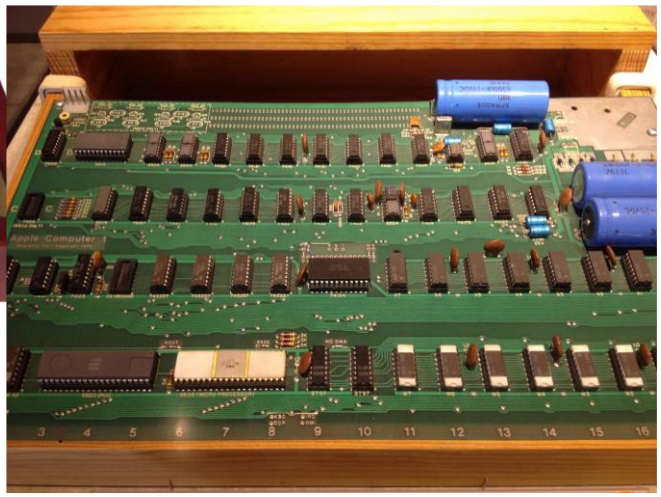
6502

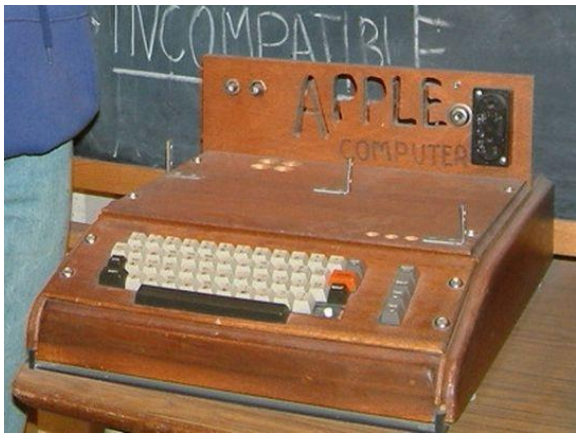
APPLE II

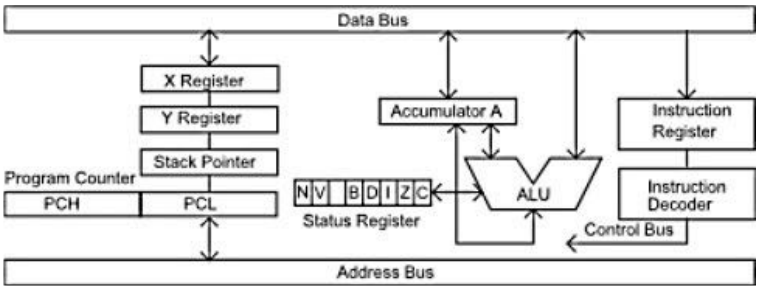
IBM's eerste microcomputer in 1981 (met een 8088 van Intel, 4.77MHz)

Extra: het eerste **Breakout arcade game** werd gemaakt door **Steve Wozniak in 1975 in zijn vrije tijd in zijn tuinhuisje**. Hij noemde het de **Gigatron**. Deze computer had nog **geen eigen microprocessor maar werkte wel als een microprocessor**. Deze bestond gewoon uit **44 losse TTL IC's**. Hij had deze chips mogen lenen van zijn bedrijf en bouwde in zijn vrije tijd een "personal computer". Deze naam was toen in feite nog niet uitgevonden. **Steve Jobs** zal hem hiermee helpen. Ze ontwikkelen een **keyboard** waarmee je de Gigatron kan besturen. Ze gaan dit concept zelf proberen te verkopen nadat HP en Atari geen interesse hebben.

1 jaar later, op 1 april 1976, zou **MOS de 6502 en ZILOG de Z80** lanceren (wel een chip met de complete microprocessor aan boord, lees de Gigatron nu verwerkt in "1" en dezelfde IC). Toen werd vrij snel de **Apple I** geboren. Dit was een **doe-het-zelf soldeerkit**. Je kon voor 666,66 euro deze kopen in de Byte Shop. De computer had nog geen behuizing. Ieder maakte deze zelf. Er werden er **175** verkocht. Dit was een innoverende computer met een kloksnelheid van 1MHz, 7kB RAM en een video terminal. Je programmeerde deze met BASIC.

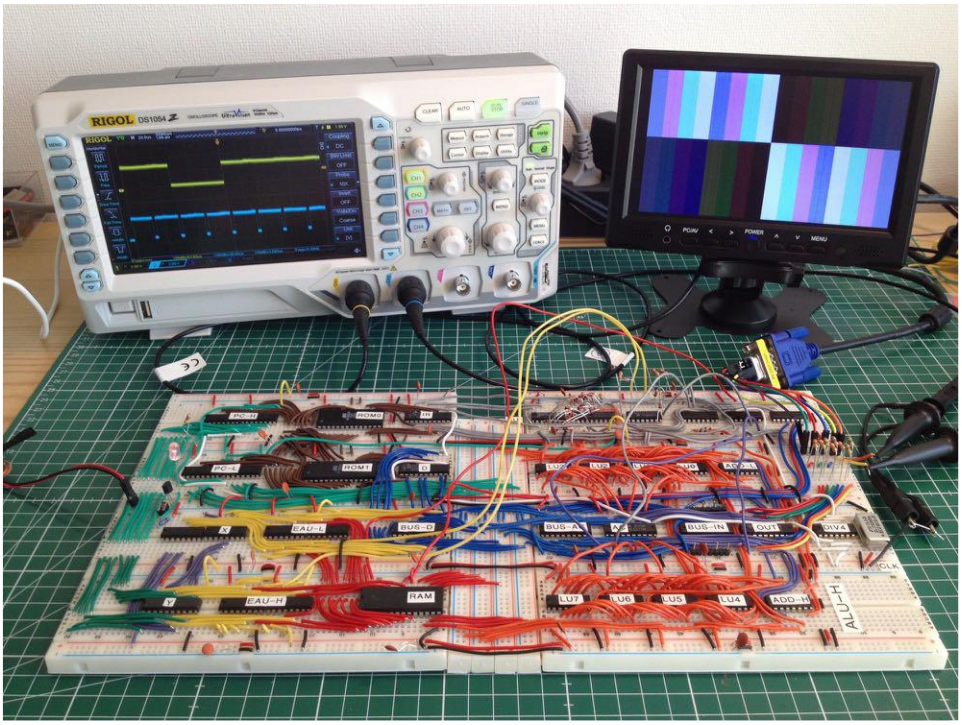






Vandaag kan je de **Gigatron** terug kopen als **TTL soldeerkit** en kan je zo jouw eigen microprocessor maken zonder enige microcontroller chip.

Meer info kan je hierover vinden op https://www.youtube.com/watch?v=4l8i_xOBTPg
en op https://gigatron.io/



Gigatron in prototype fase, zonder microcontroller

Steve Wozniak toont de Apple II, Gebouwd samen met Steve Jobs in 1977. Nu had de PC al een kunststoffen behuizing.

Apple Introduces the First Low Cost Microcomputer System with a Video Terminal and 8K Bytes of RAM on a Single PC Card.

The Apple Computer. A truly complete microcomputer system on a single PC board. Based on the MOS Technology 6502 microprocessor, the Apple also has a built-in video terminal and sockets for 8K bytes of on-board RAM memory. With the addition of a keyboard and video monitor, you'll have an extremely powerful computer system that can be used for anything from developing programs to playing games or running BASIC.

Combining the computer, video terminal and dynamic memory on a single board has resulted in a large reduction in chip count, which means more reliability and lowered cost. Since the Apple comes fully assembled, tested & burned-in and has a complete power supply on-board, initial set-up is essentially "hassle free" and you can be running within minutes. At $666.66 (including 4K bytes RAM!) it opens many new possibilities for users and systems manufacturers.

You Don't Need an Expensive Teletype.

Using the built-in video terminal and keyboard interface, you avoid all the expense, noise and maintenance associated with a teletype. And the Apple video terminal is six times faster than a teletype, which means more throughput and less waiting. The Apple connects directly to a video monitor (or home TV with an inexpensive RF modulator) and displays 960 easy to read characters in 24 rows of 40 characters per line with automatic scrolling. The video display section contains its own 1K bytes of memory, so all the RAM memory is available for user programs. And the Keyboard Interface lets you use almost any ASCII-encoded keyboard.

The Apple Computer makes it possible for many people with limited budgets to step up to a video terminal as an I/O device for their computer.

No More Switches, No More Lights.

Compared to switches and LED's, a video terminal can display vast amounts of information simultaneously. The Apple video terminal can display the contents of 192 memory locations at once on the screen. And the firmware in PROMS enables you to enter, display and debug programs (all in hex) from the keyboard, rendering a front panel unnecessary. The firmware also allows your programs to print characters on the display, and since you'll be looking at letters and numbers instead of just LED's, the door is open to all kinds of alphanumeric software (i.e., Games and BASIC).

8K Bytes RAM in 16 Chips!

The Apple Computer uses the new 16-pin 4K dynamic memory chips. They are faster and take ¼ the space and power of even the low power 2102's (the memory chip that everyone else uses). That means 8K bytes in sixteen chips. It also means no more 28 amp power supplies.

The system is fully expandable to 65K via an edge connector which carries both the address and data busses, power supplies and all timing signals. All dynamic memory refreshing for both on and off-board memory is done automatically. Also, the Apple Computer can be upgraded to use the 16K chips when they become available. That's 32K bytes on-board RAM in 16 IC's—the equivalent of 256 2102's!

A Little Cassette Board That Works!

Unlike many other cassette boards on the marketplace, ours works every time. It plugs directly into the upright connector on the main board and stands only 2" tall. And since it is very fast (1500 bits per second), you can read or write 4K bytes in about 20 seconds. All timing is done in software, which results in crystal-controlled accuracy and uniformity from unit to unit.

Unlike some other cassette interfaces which require an expensive tape recorder, the Apple Cassette Interface works reliably with almost any audio-grade cassette recorder.

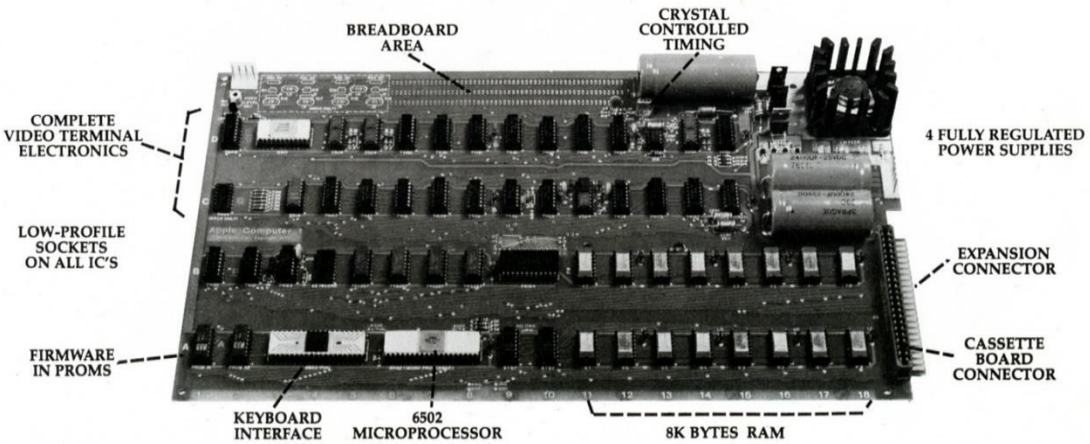
Software:

A tape of **APPLE BASIC** is included free with the Cassette Interface. Apple Basic features immediate error messages and fast execution, and lets you program in a higher level language immediately and without added cost. Also available **now** are a dis-assembler and many games, with many software packages, (including a macro assembler) in the works. And since our philosophy is to provide software for our machines free or at minimal cost, you won't be continually paying for access to this growing software library.

The Apple Computer is in stock at almost all major computer stores. (If your local computer store doesn't carry our products, encourage them or write us direct). **Dealer inquiries invited.**

Byte into an Apple $666.66*

* includes 4K bytes RAM



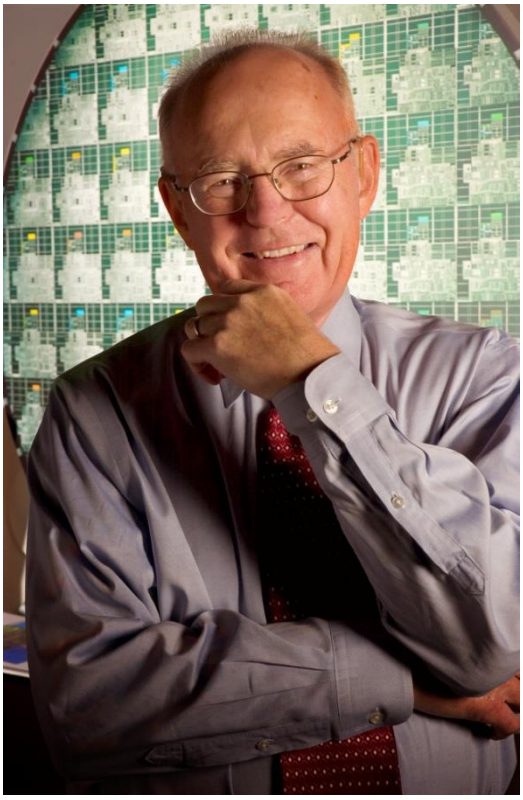
APPLE Computer Company • 770 Welch Rd., Palo Alto, CA 94304 • (415) 326-4248

OCTOBER 1976 CIRCLE NO. 7 ON INQUIRY CARD INTERFACE AGE 11

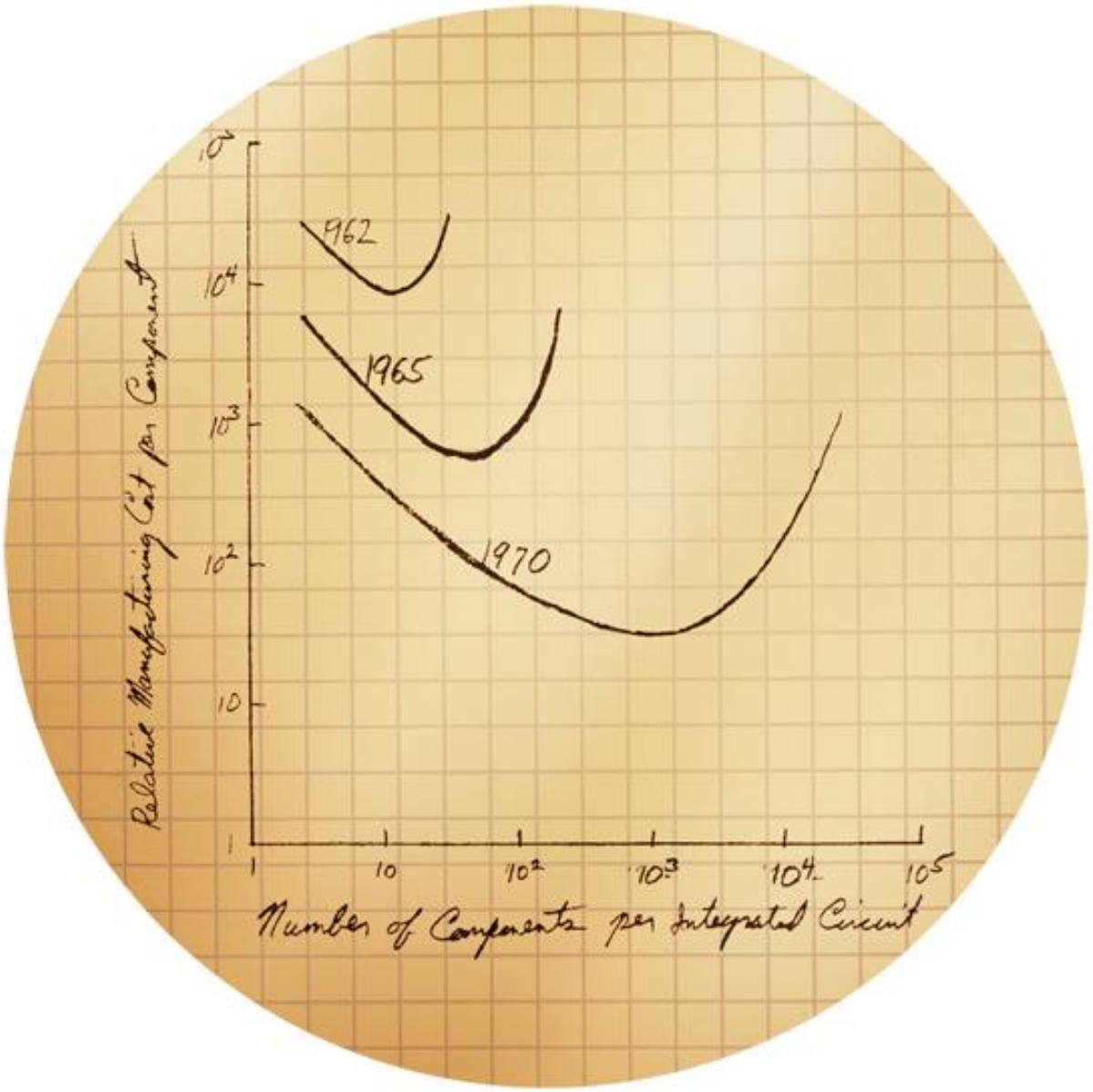
Belangrijkste microprocessor bouwers tot op heden

- Texas Instruments 9800 (1974)
- Motorola 6800 (agosto 1974) -> zal in Apple gebruikt worden
- RCA 1802 (1974)
- Fairchild F8 (1975)
- MOS Technology 6502 (1975)
- National Semiconductors SC/MP (aprile 1976)
- Zilog Z-80 (luglio 1976)
- Intel
- AMD
- Philips NXP
- Microchip (PIC ...)
- Atmel (AVR, Tiny ...)

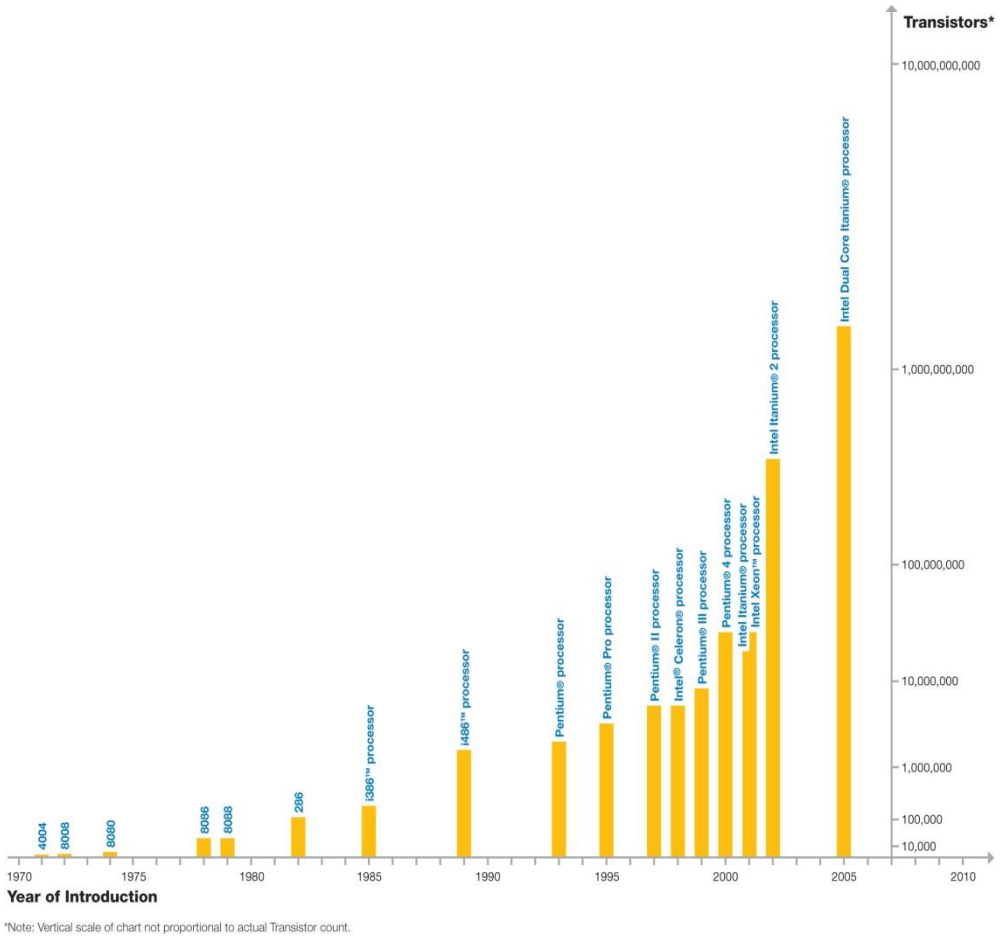
Moore's law: Het aantal transistoren verdubbelt elke 18 tot 24 maanden op dezelfde chip oppervlakte.



Gordon Moore, 2005 co-founder Intel



Originele grafiek van De wet van Moore



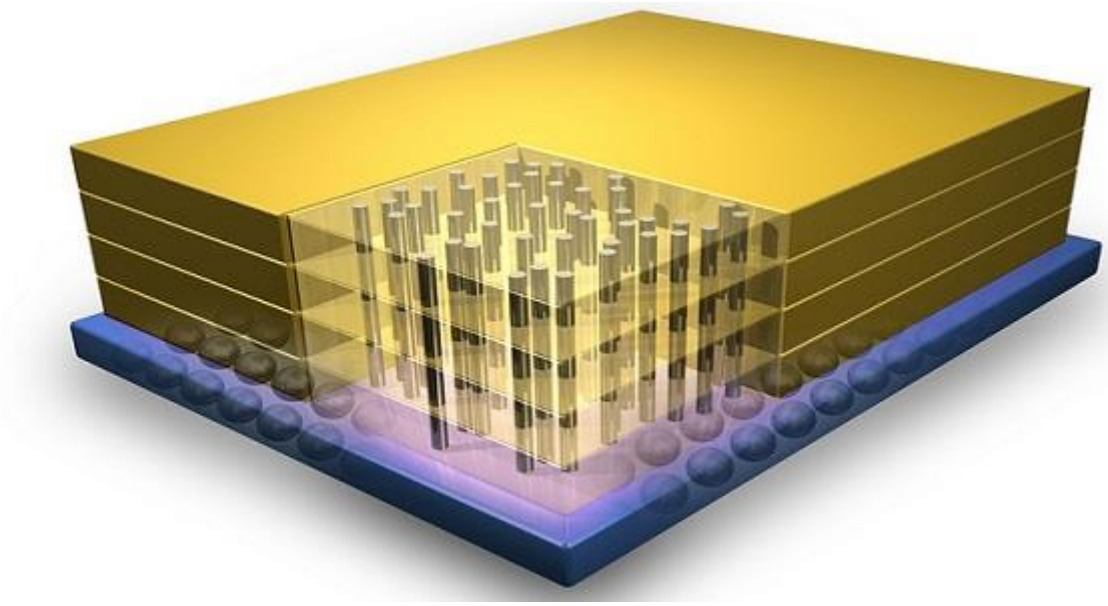
Actuele µC overzicht voor Intel volgens Moore's law

Microprocessor	Year of Introduction	Transistors
4004	1971	2,300
8008	1972	2,500
8080	1974	4,500
8086	1978	29,000
Intel286	1982	134,000
Intel386™ processor	1985	275,000
Intel486™ processor	1989	1,200,000
Intel® Pentium® processor	1993	3,100,000
Intel® Pentium® II processor	1997	7,500,000
Intel® Pentium® III processor	1999	9,500,000
Intel® Pentium® 4 processor	2000	42,000,000
Intel® Itanium® processor	2001	25,000,000
Intel® Itanium® 2 processor	2003	220,000,000
Intel® Itanium® 2 processor (9MB cache)	2004	592,000,000

Zie voor meer recentere info op http://en.wikipedia.org/wiki/List_of_Intel_chipsets

Extra: Laatste nieuwe snufjes op gebied van microcontrollers (artikels gevonden op internet):

3D geheugenchips:



3D geheugenchip

Eerste commerciële 3D-geheugenchip

Snelle geheugenkubus geproduceerd met nieuwe 3D-fabricagetechnologie

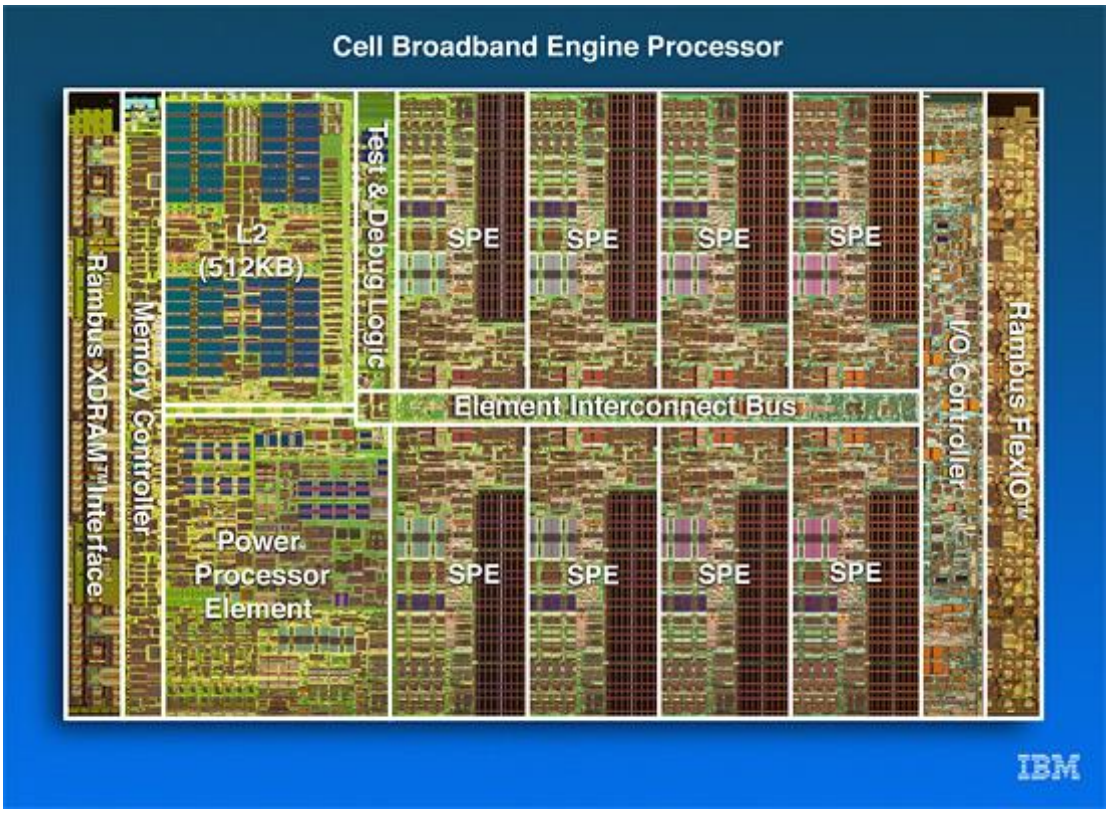
Publicatiedatum: 8 december 2011

Micron en IBM starten met de commerciële productie van een 3D-geheugenchip, de **Hybrid Memory Cube** (HMC). Hierbij wordt gebruik gemaakt van IBM's nieuwe 3D-chiptechnologie waarbij doorverbindingen in het silicium substraat worden aangebracht (Through-Silicon Vias, TSV). Met de geheugenkubus van Micron kunnen snelheden worden behaald die 15 keer hoger zijn dan wat met de huidige technologieën mogelijk is. Hoewel de kubus-chips in eerste instantie zullen worden toegepast in grote netwerken, snelle servers en industriële automatisering, zullen deze naar verwachting over enige tijd ook worden gebruikt voor consumentenproducten.

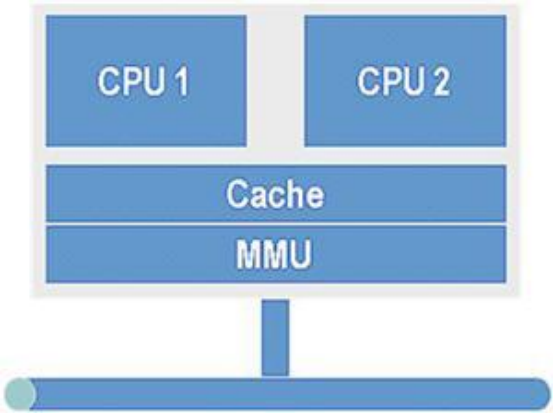
Bij de nieuwe geheugentechnologie worden doorverbindingen in het silicium gebruikt om de besturingslogica te verbinden met op elkaar gestapelde DRAM-lagen. Bij prototypes van de geheugenkubus werden doorvoersnelheden gemeten van 128 gigabytes per seconde, dat is tien keer sneller dan bij het snelste bestaande geheugen. Daarnaast heeft de kubus voor het datatransport 70% minder energie nodig dan conventionele geheugenchips, en neemt deze maar 10 % van de ruimte in. De Hybrid Memory Cube zal in het 32 nm proces van IBM worden geproduceerd in de halfgeleiderfabriek in East Fishkill (New York),

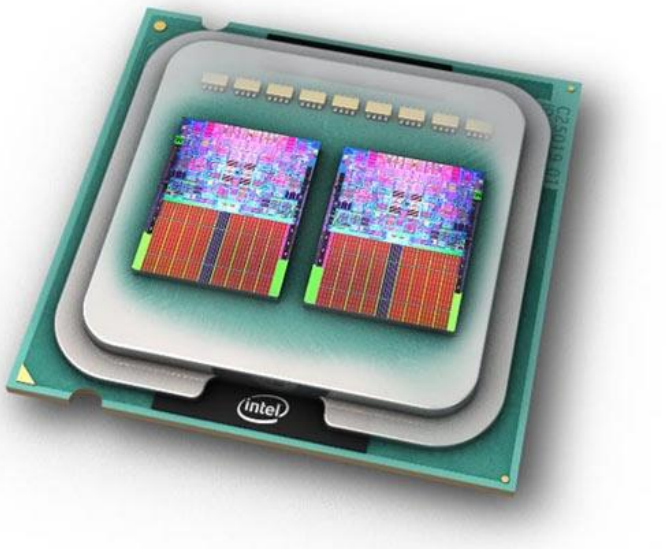
Meer info:
http://www-03.ibm.com/press/us/en/pressrelease/36125.wss

Multicore processors:



Multicore processor (8 stuks) van Intel





Het werk wordt verdeeld onder de verschillende processors.

Multikern(core)processor info

Uit Wikipedia, de vrije encyclopedie

Een **multikernprocessor** is een chip waarop meerdere processoren (Eng.: *cores*) zijn geïntegreerd.

Door het groeiende aantal processen dat op een gewone desktopcomputer draait, de vele dingen die mensen met hun computer doen terwijl ze met iets anders bezig zijn, de hoge prijs van echte multiprocessorsystemen en daarnaast de hoge effectiviteit van het werken met meerdere *threads*, is het tegenwoordig interessant om meerdere kernen op één processor te integreren. Bij een dubbelkernprocessor zitten er op één chip als het ware twee complete processors, die één enkele verbinding met het hoofdgeheugen delen — wat dan ook de flessenhals van multikernprocessoren is.

Inhoud

] Huidige situatie (2006)

IBM

IBM heeft met haar Power-serverprocessoren sinds de introductie van de Power5 CPU in 2005 haar tweede generatie multikernprocessoren in haar productlijn. Deze RISC-processoren worden hoofdzakelijk gebruikt in servertoepassingen voor IBM's System p (Besturingsystemen: AIX en Linux. Voorheen bekend als pSeries en daarvoor als RS6000) en System i (Besturingsystemen: i5/OS, AIX en Linux. Voorheen bekend als iSeries en daarvoor als AS/400). Tegenwoordig maakt Microsoft's Xbox 360 ook gebruik van drie IBM Power processoren.

AMD

AMD's nieuwste generatie processors was er al op ontwikkeld om ooit *dual-core* te worden, en nu heeft AMD met zijn Toledo ervoor gekozen om eerst de servermarkt in te stappen voordat ze de consumentenversies van hun *dual-core* processors lanceerden. Waarschijnlijk is deze zet bedoeld om Intel een stap voor te zijn in de markt waar de meeste winst te behalen valt, wat ook blijkt uit het feit dat de 8xx serie voor 8- en 4-way-servers het eerst uit komt, gevolgd door de 2xx serie voor twinprocessorsystemen en als laatste de 1xx serie, bedoeld voor singleprocessorsystemen. Nog iets later komen de desktopversies van deze chips uit.

De Desktop versie van de dual-core processors van AMD was de Athlon 64 X2. Dit omdat het eigenlijk gewoon twee aan elkaar geplakte Athlon 64's zijn. De quad-core versies heten "Athlon FX" en "Phenom X4". De FX series zijn al oud en al vrijwel uit de markt. De nieuwste quad-core processors zijn de Phenom X4 series.

Intel

Intel is precies de andere kant op aan het werken: ze begonnen met de introductie van de Intel Pentium D Extreme-Edition 840 (Smithfield). Deze processor draait op 3,20 Gigahertz, 600 Megahertz lager dan het oude, *single-core* topmodel. Intel introduceerde al eerder een technologie waardoor (virtueel) 2 threads tegelijk op 1 fysieke core kunnen draaien: Hyper-Threading (HT). Op een *dual-core* processor met HT zouden dan 4 threads tegelijk uitgevoerd worden.

Wanneer AMD zijn eerste dual-core processors uitbracht kon Intel in een relatief korte tijdspanne van iets langer dan een maand hun eigen dual-cores daar tegenover plaatsen. De reden hiervoor was dat Intel al voorop stond met deze kennis, maar deze nog niet commercieel ging uitbuiten. Deze dual-core processoren kregen de naam "Pentium D".

In 2006 kwam Intel met zijn nieuwste generatie dual-core processors: de Intel Core 2 Duo (C2D), welke gebaseerd was op hun eigen "Core micro architecture". Hun codenaam tijdens de ontwikkeling was "Conroe". Ook zijn er quad-core processoren van Intel, de Intel Core 2 Quad. Er zijn ook "Intel Core 2 Extreme" processoren. Dit zijn vaak de high-end processoren, vandaar de naam "extreme". Men kan aan de letters zien of het een dual-core is of een quad-core. Zo is een "Core 2 Extreme X...." een dual-core en een "Core 2 Extreme QX...." een quad-core. De X staat voor "extreme" en de Q uiteraard voor "quad".

Voordelen

Het plaatsen van meerdere kernen op een processor heeft als voordeel dat relatief veel snelheid gewonnen kan worden met een geringe investering; een computer met een dubbelkernprocessor is slechts een beetje trager dan een computer met twee losse processors, terwijl de dubbelkernprocessor geen speciale en vaak dure hardware zoals een moederbord met twee voetjes nodig heeft. Hoewel de aanschafprijs nu nog relatief hoog is door het productieproces (de twee cores worden tegelijk gemaakt: is er één kern kapot, dan is de tweede kern ook onbruikbaar), maar in de toekomst zullen ze goedkoper worden en omdat de moederborden met één processorvoet goedkoper zijn, zal er een aanzienlijke kostenbesparing mogelijk zijn.

Ook wordt gebruikers een gemakkelijke manier geboden om extra snelheid in hun systeem te krijgen, de gebruiker kan eenvoudig zijn oude processor vervangen door een nieuwe met meer kernen en daardoor de snelheid van zijn machine voor een zeer geringe investering flink doen toenemen.

Nadelen

Het grote nadeel van multikernprocessors is dat software veelal niet automatisch gebruik maakt van meerdere kernen; de software moet ofwel uit meerdere processen bestaan, ofwel specifiek van multithreading gebruikmaken. Klassieke applicaties met slechts een enkele thread profiteren nauwelijks van multikernprocessors.

Het productieproces van de processors vereist dat beide processor kernen tegelijk succesvol geproduceerd worden. Indien een kern defect is, is de andere ook niet meer bruikbaar. Vanuit fabrikageoogpunt zijn multikernprocessoren dan ook niet makkelijk hanteerbaar.

Ook betekent een verdubbeling van het aantal kernen vrijwel een verdubbeling van het energieverbruik van de processor. Dit levert warmteproblemen op en om dit te compenseren dient in veel gevallen de kloksnelheid van de processor verlaagd te worden. Applicaties die slechts één kern kunnen gebruiken zullen daarom trager werken dan in een enkelkernsysteem waar een hoger geklokte processor in zit.

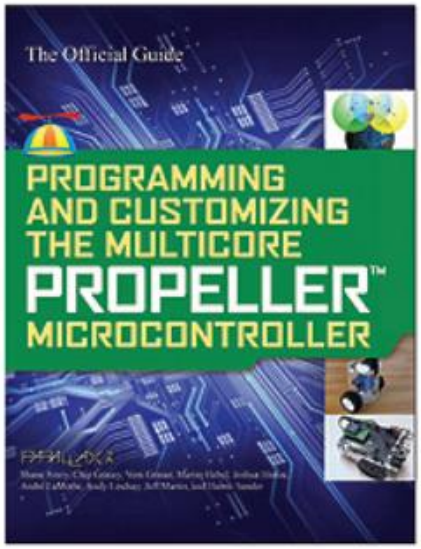
Toekomst

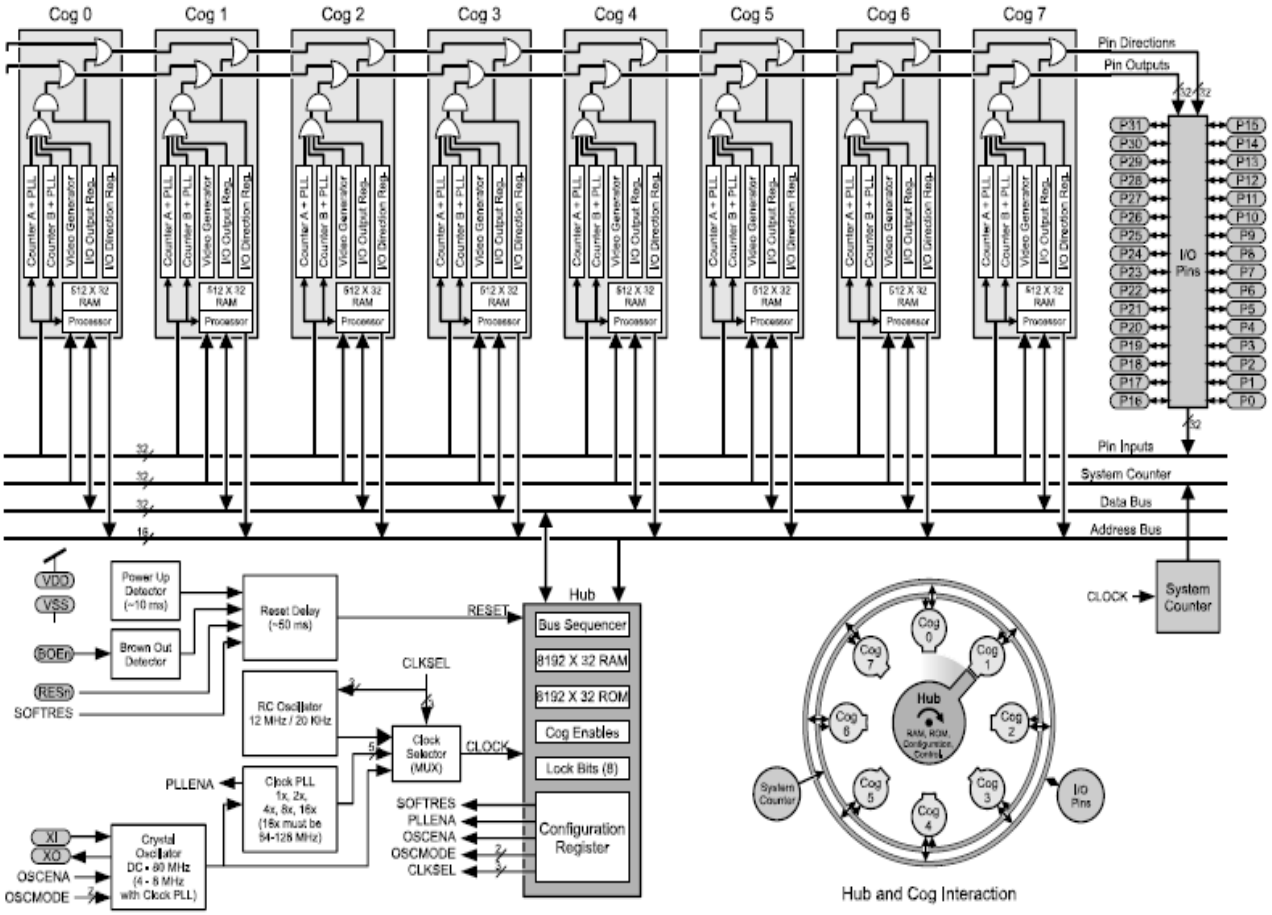
De op dit moment (april 2006) verkrijgbare processors beschikken over één of twee kernen. In de toekomst zullen dubbelkernprocessors niet meer voldoen, en zullen ook in het x86-gebied multikernprocessors opduiken. Bij andere architecturen is dat al het geval: Sun's Niagaraprocessor heeft 8 kernen die elk 4 threads kunnen verwerken, waardoor er in totaal 32 threads tegelijk kunnen worden uitgevoerd. Hoewel dit momenteel voor de consument nog niet zo praktisch is, zijn er servers die hier gebruik van kunnen maken en zal ook gemultithreadde consumentensoftware langzamerhand de overhand krijgen.

Intel ziet de toekomst voor multicore processoren rooskleurig in. Tegen september 2006 had Intel al een prototype ontwikkeld van een 80-core processor. Deze processor is in staat om een terabyte per seconde te verwerken en het kan meer dan een teraflop leveren. Verwacht wordt dat de chip rond 2012 op de markt gaat verschijnen. Het prototype beschikt over 80 floatingpoint eenheden, die elk op 3,16GHz geklokt zijn. Het bedrijf liet hiermee zien dat 45nm-productietechnologie stroomlekkage met factor vijf verminderde en wist ook de prestaties met twintig procent omhoog te schroeven. Dit bericht werd in september 2006 op het Intel Developer Forum geplaatst.

Ook in kleinere uitvoering

Tegenwoordig is er ook een Microcontroller met meerdere kernen, en wel de Parallax Propeller, welke acht 32-bit processor kernen bezit.





Propeller: P8X32A

Je kan heel mooi de 8 verschillende cores zien samenwerken.

Meer info hierover vind je op:

http://www.parallax.com/portals/0/help/P8X32A/QnaWeb/

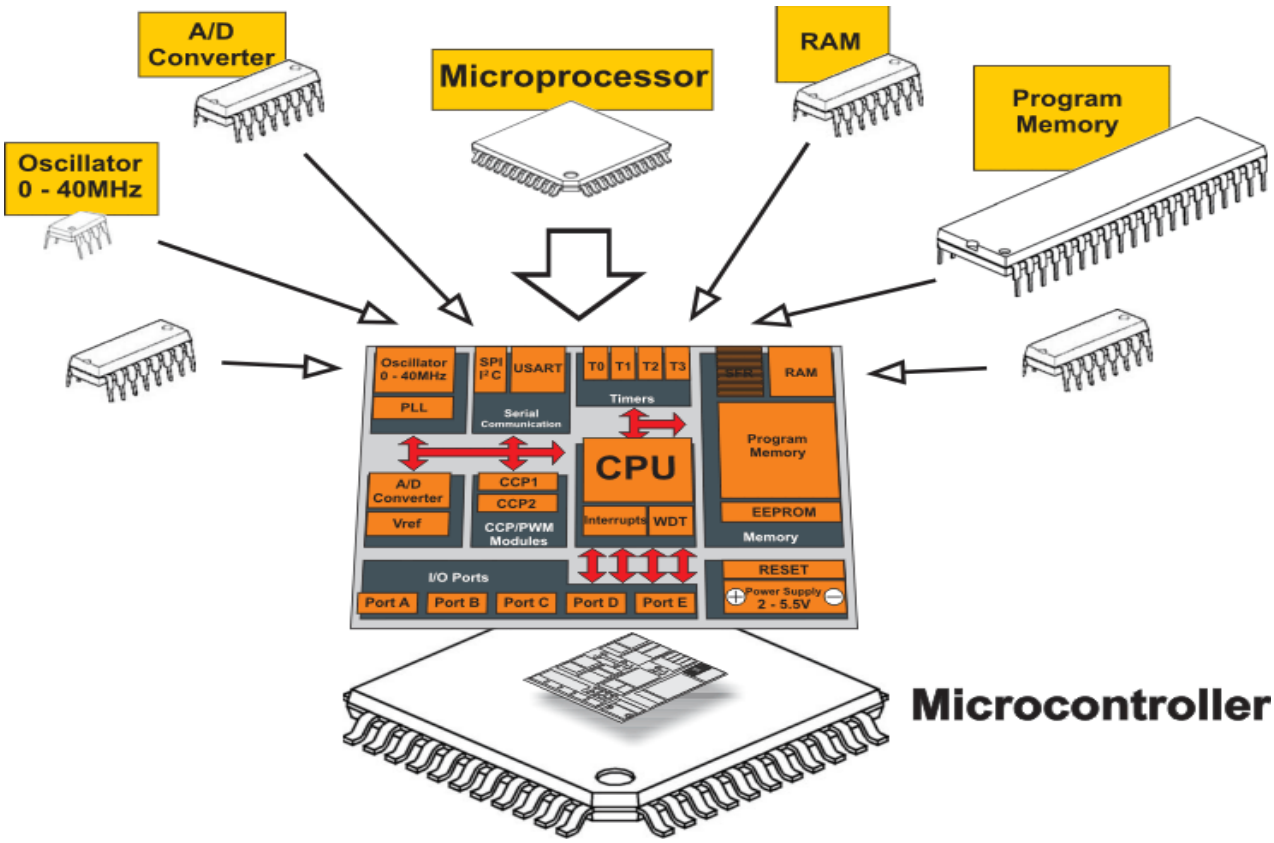
3. Inleiding tot de microcontrollers (zie voorafgaande figuren!)

Van 1969 tot 1971 bouwde Intel in opdracht van ontwerpers van de Amerikaanse **firma Datapoint** de **eerste microprocessor** (**4004**). Omdat de microprocessor ongeveer 10 keer trager werkte dan verwacht ging de koop niet door. Intel heeft toen in 1972 besloten om het product op eigen risico op de markt te brengen. Dit gebeurde onder de codenaam **8008**. Deze microprocessor werd enkele jaren later opgevolgd door de **8080**. Deze microprocessor werd gebruikt om in 1975 de **eerste microcomputer** te bouwen. Dit was de **Altair 8800** van de **firma MITS**. Naast de 8080 bezat deze microcomputer 256 bytes geheugen en dan toggleschakelaars en LED's op een frontplaat. Een werkend model kostte toen rond de $2000. MITS heeft van dit model ongeveer 2000 stuks verkocht. Voor deze computer werd een **BASIC-programmeertaal** gemaakt door twee jonge kerels (Bill Gates en Paul Allen) die een bedrijfje stichten met de naam **MICROSOFT.**
In ditzelfde jaar (1975) bracht **MOS-Technologies de 6502** microprocessor op de markt aan sterkt gereduceerde prijzen. $25 in plaats van $150 voor de Intel 8080. Het is de 6502 die later zal dienen als hart voor de **APPLE II**. APPLE werd gesticht door **Steve Jobs** en Steve Wozniak. Dit was de **eerste echte personal computer.**
Wanneer in 1981 **I.B.M.** besluit om ook een microcomputer op de markt te brengen is het startschot gegeven voor een waanzinnige evolutie die tot op dit moment nog steeds doorgaat.

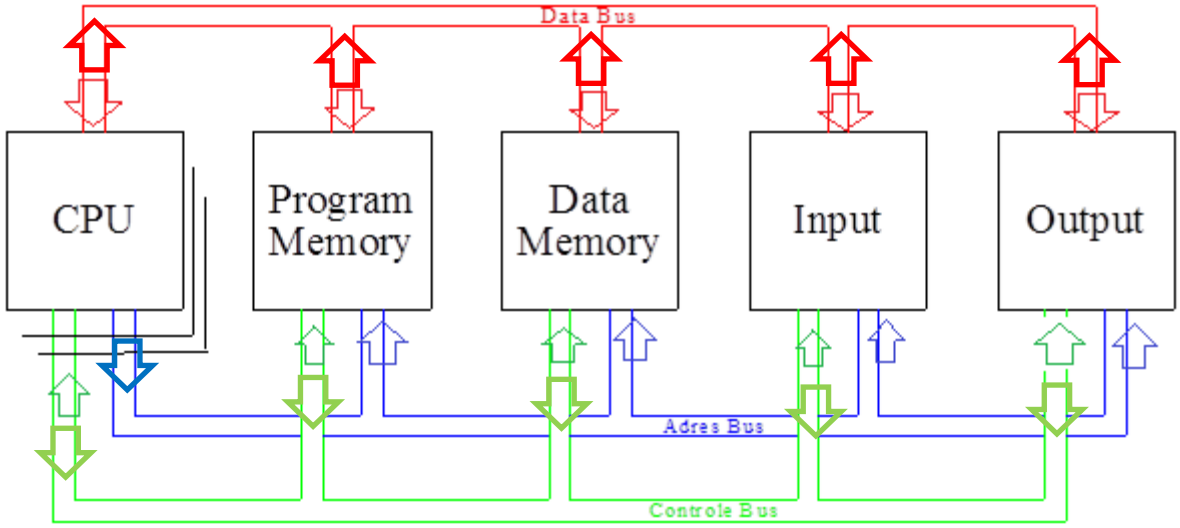
Een microcomputer is dus een kleine computer waarin een **microprocessor** + I/O + geheugen zorgt voor het verwerken van gegevens. Tegenwoordig vinden we kleine computertjes overal terug. In de weegschaal van de slager, in de kassa van de supermarkt, in de sturing van de verkeerlichten, in auto's, enz… Deze **microcomputers** zitten als het ware ingebakken in toestellen die we dagdagelijks gebruiken, we spreken van "**embedded computers**". Tegenwoordig zijn deze microcomputertjes zo klein dat **ze in één IC** zitten. We spreken dan van "**embedded controllers" of microcontrollers**. Het is over deze microcontrollers dat deze cursus handelt. Alvorens de microcontrollers zelf te behandelen bespreken we eerst het blokschema van een microcomputersysteem. Daarna maken we de overstap naar microcontrollers. Wat microcontrollers betreft zijn er heel veel fabrikanten die elk hun eigen (uitgebreid) reeks van microcontrollers verkopen. We denken dan onder andere aan Intel, Motorola, Atmel, Microchip, Hitachi, Zilog, Dallas, Philips, Siemens, National Semiconductor, enz… Wij gaan ons in deze cursus, na een algemene inleiding, beperken tot de **MEGA328P van Atmel.**
Het is niet de bedoeling van deze cursus om een diepgaande studie van een microcontroller te geven. Wel is het de bedoeling om een goed inzicht te krijgen in de werking van een microcontroller. Iemand die zich aangetrokken voelt tot deze materie moet de tijd nemen om tenminste één microcontroller volledig uit te

diepen. Welke microcontroller men neemt speelt eigenlijk weinig rol. De meeste microcontrollers werken op een vergelijkbare manier en dus is de overstap van één soort microcontroller naar een andere (migratie) meestal vrij makkelijk te maken. De verschillen zitten meestal in (soms zéér belangrijke) details. Neem dus minstens één keer de moeite om je in deze materie te verdiepen. De beste literatuur om dit te doen is meestal niet een cursus of een boek maar wel de datasheet van het component.

Bij deze cursus wordt er van uitgegaan dat de leerlingen een degelijke basis bezitten van digitale technieken. Bij de algemene inleiding gaan we uit van de opbouw van een **8-bit processor.**



4. Blokschema van een microcomputersysteem (single core)



Blokschematische voorstelling va een microcontroller

Uit bovenstaande figuur blijkt dat elk zinvol microcomputersysteem bestaat uit vijf blokken. De eerste blok is de **C.P.U. (central processing unit).** Dit is het deel van de computer waar de besturing van de computer in gebeurt. In deze unit vindt ook het verwerken van de gegevens plaats. Meestal is dit een microprocessor. De handelingen die de CPU uitvoert zijn **cyclisch.** Dit wil zeggen dat er een vast patroon zit in de handelingen die de processor verricht. De CPU haalt namelijk steeds een instructie op uit het geheugen, voert de in de instructie aangegeven bewerking uit, haalt de volgende instructie op, enz…

Zoals je kan merken zijn er **twee soorten geheugen** terug te vinden. Eén geheugen wordt gebruikt om het programma, dat moet worden uitgevoerd, te bewaren. Dit deel van het geheugen noemen we het pro**gram-mageheugen** (**program memory**). Het is noodzakelijk dat minstens één programmageheugen is dat niet vluchtig is (non-volatile). Dit wil zeggen dat de inhoud van het geheugen wordt bewaard ook als de voedingsspanning van het geheugen wordt verwijderd. Hierin moet zich het programma bevinden dat wordt uitgevoerd als de computer wordt opgestart (bootproces).

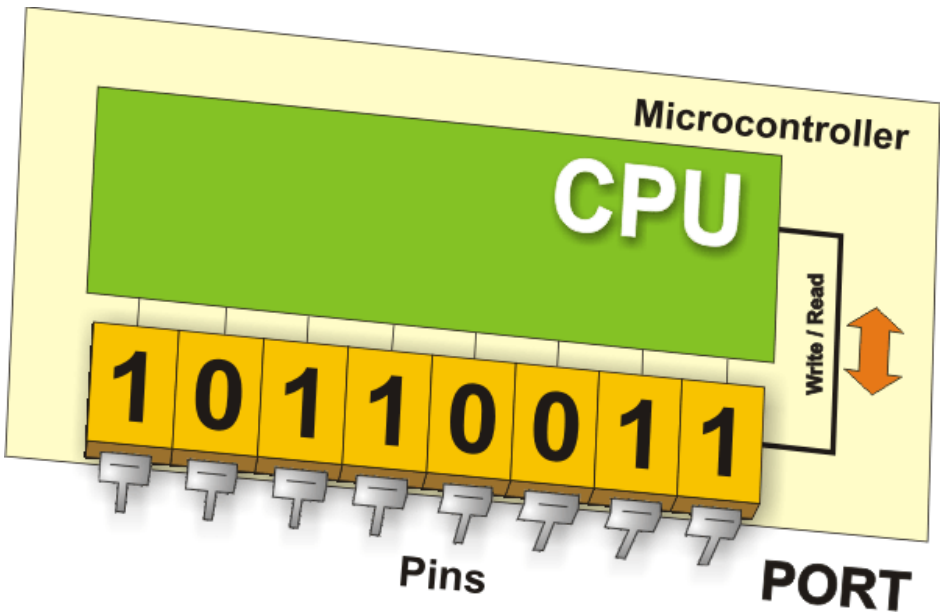
Een tweede deel geheugen is het **geheugen waar zich de gegevens** bevinden (**data memory**). Dit kunnen zowel de te verwerken gegevens zijn als de resultaten die de CPU uitkomt na het uitvoeren van het programma.

Een laatste deel vormt **de I/O** (Input – Output) of de randcomponenten (peripherical components). Via deze module komen gegevens de computer binnen en worden de gegevens van de computer terug naar buiten gestuurd. Als input kennen we onder andere het toetsenbord, de muis, een temperatuurmeter, een ingang voor een schakelaar, enz… Als output kennen we onder andere het scherm, een printer, een

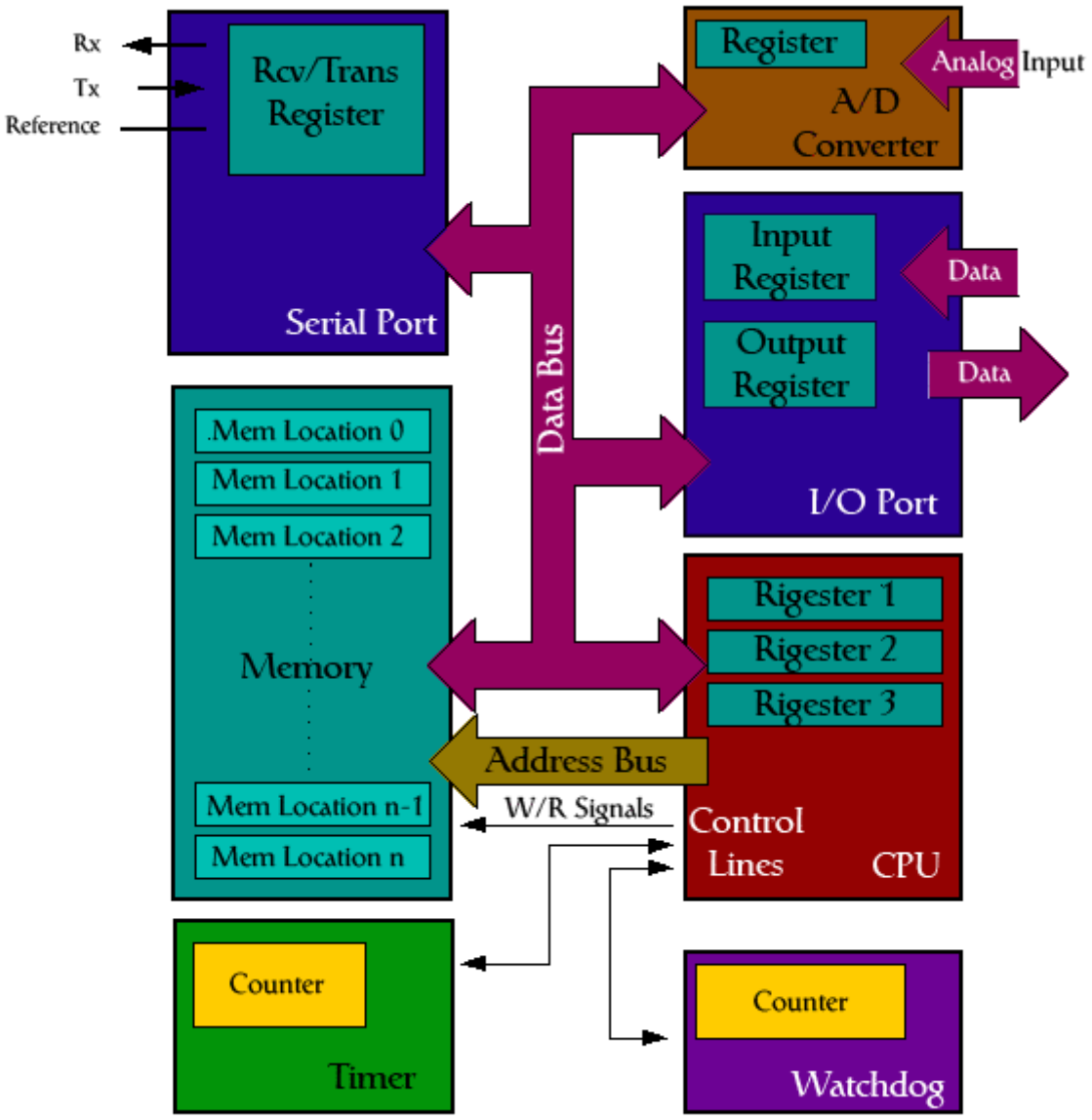
motorsturing, een lampje, enz… Er zijn ook randcomponenten die input en output combineren. Denk hierbij vooral aan een modem.

Het is natuurlijk noodzakelijk dat de **verschillende blokken met elkaar worden verbonden**. Zo moet de CPU de verschillende instructies uit het programmageheugen kunnen ophalen, resultaten naar de output-module sturen enz… De verbinding gebeurt door elektrische verbindingen (draadjes of printbanen). Afhankelijk van de grootte van de gegevens (aantal bits) heb je meerdere draadjes nodig. Bijvoorbeeld als je telkens een byte wil doorgeven dan moet je **acht draadjes** gebruiken. Zo een bundel van draadjes noemen we een **bus** en een draadje in deze bus noemen we een **lijn**. De bus verantwoordelijk voor het doorsturen van de **gegevens** (data) en de **instructiecodes** van het programma noemen we de **databus**. In het geval van microcontrollersystemen bedraagt de breedte van de databus meestal acht of zestien datalijnen. De bus die aanduidt waar in het geheugen gegevens moeten worden opgehaald of weggezet noemen we de **adresbus.** In de meeste 8-bit microcontrollersystemen is de adresbus 16 adreslijnen breed.

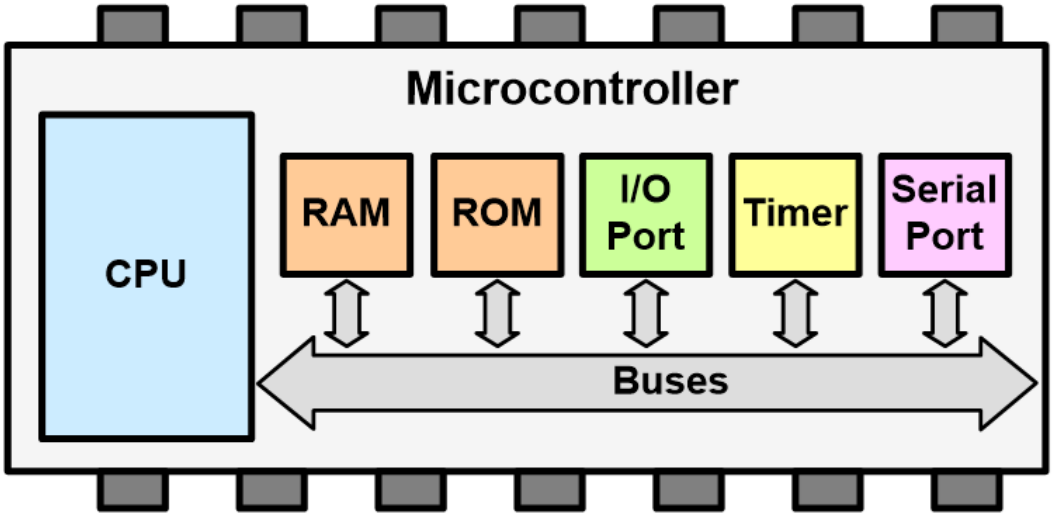
Naast de adresbus en de databus is er nog een derde bus. Deze bus wordt de **controlebus** genoemd. Hierin vinden we controlelijnen terug zoals de read/write-lijn, de interruptlijnen, de reset, enz… Deze lijnen controleren de richting waarin de gegevens over de databus gaan, bepalen de werking van de computer, enz… We komen op de verschillende controlelijnen later nog uitvoerig terug. Op een print zijn de busstructuren meestal makkelijk terug te vinden. Ze bestaan uit een aantal parallel lopende printbanen.



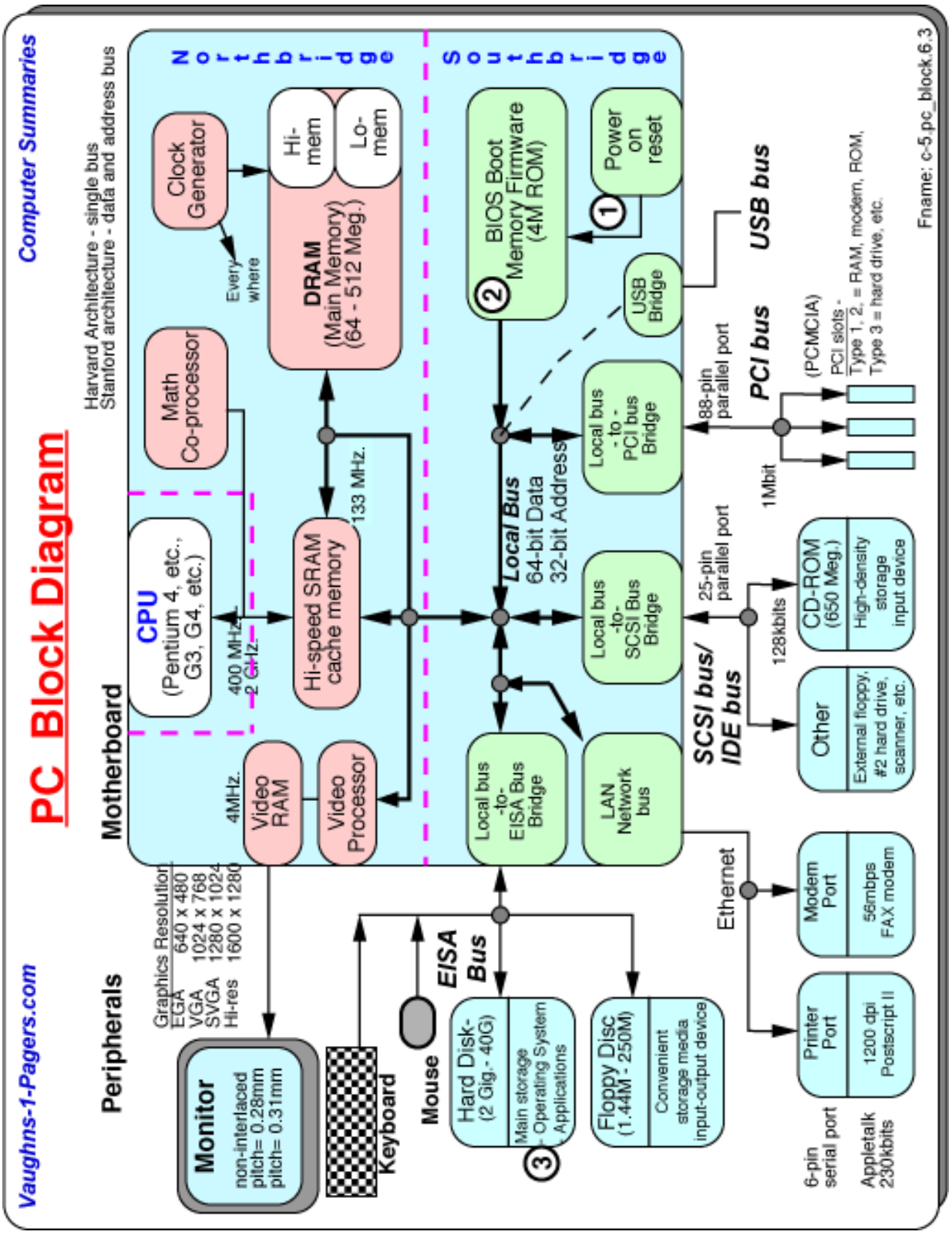
Voorbeeld van een I/O register (port) die via datalijnen (8 bits databus) en een controle lijn verbonden is met de CPU.



Andere voorstellingen van een microcontroller architecture



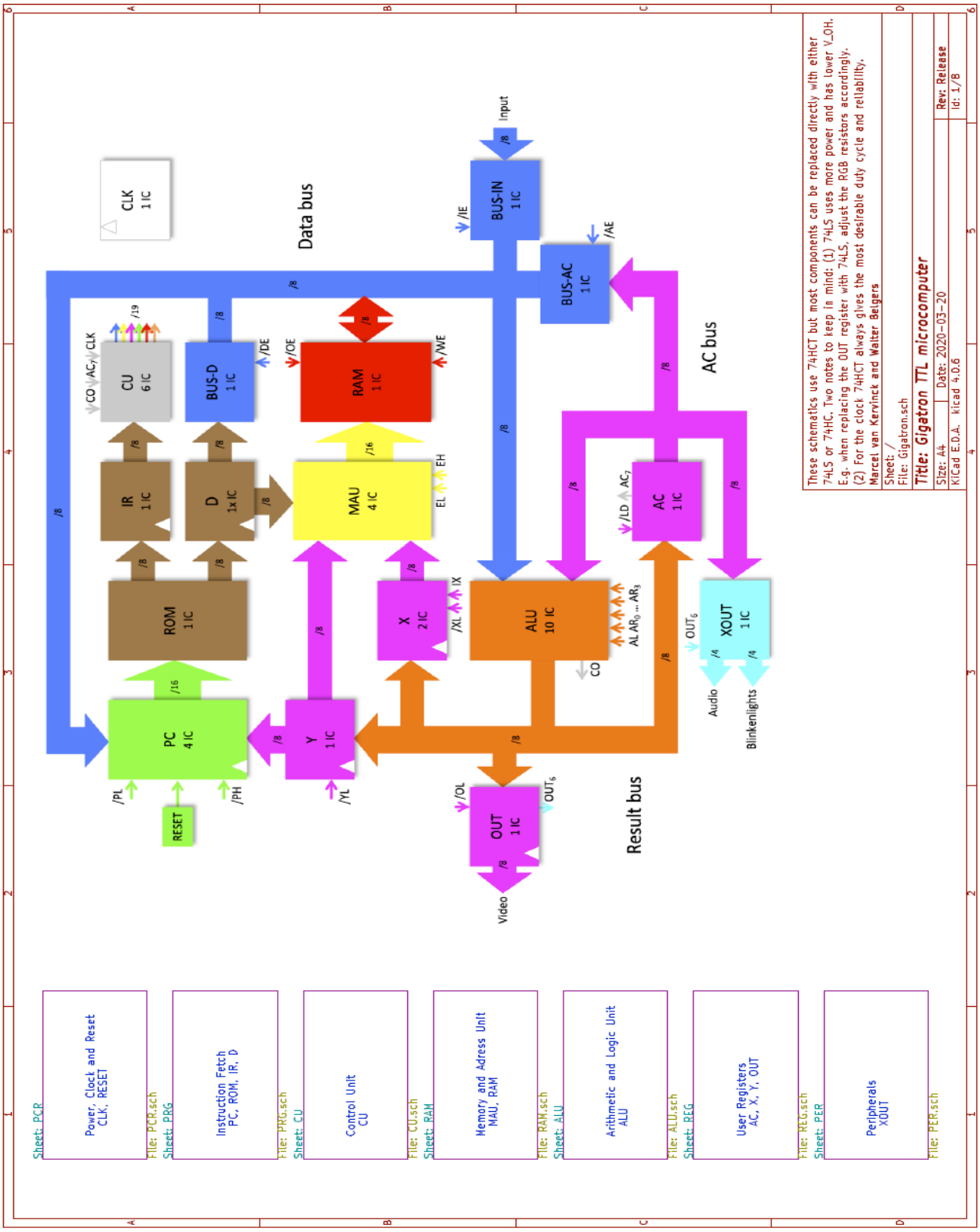
PC architectuur



➔ **Opdracht 1:** vergelijk deze architecture met de µC architecture op volgende pagina. Kan jij de gelijkenissen terugvinden?

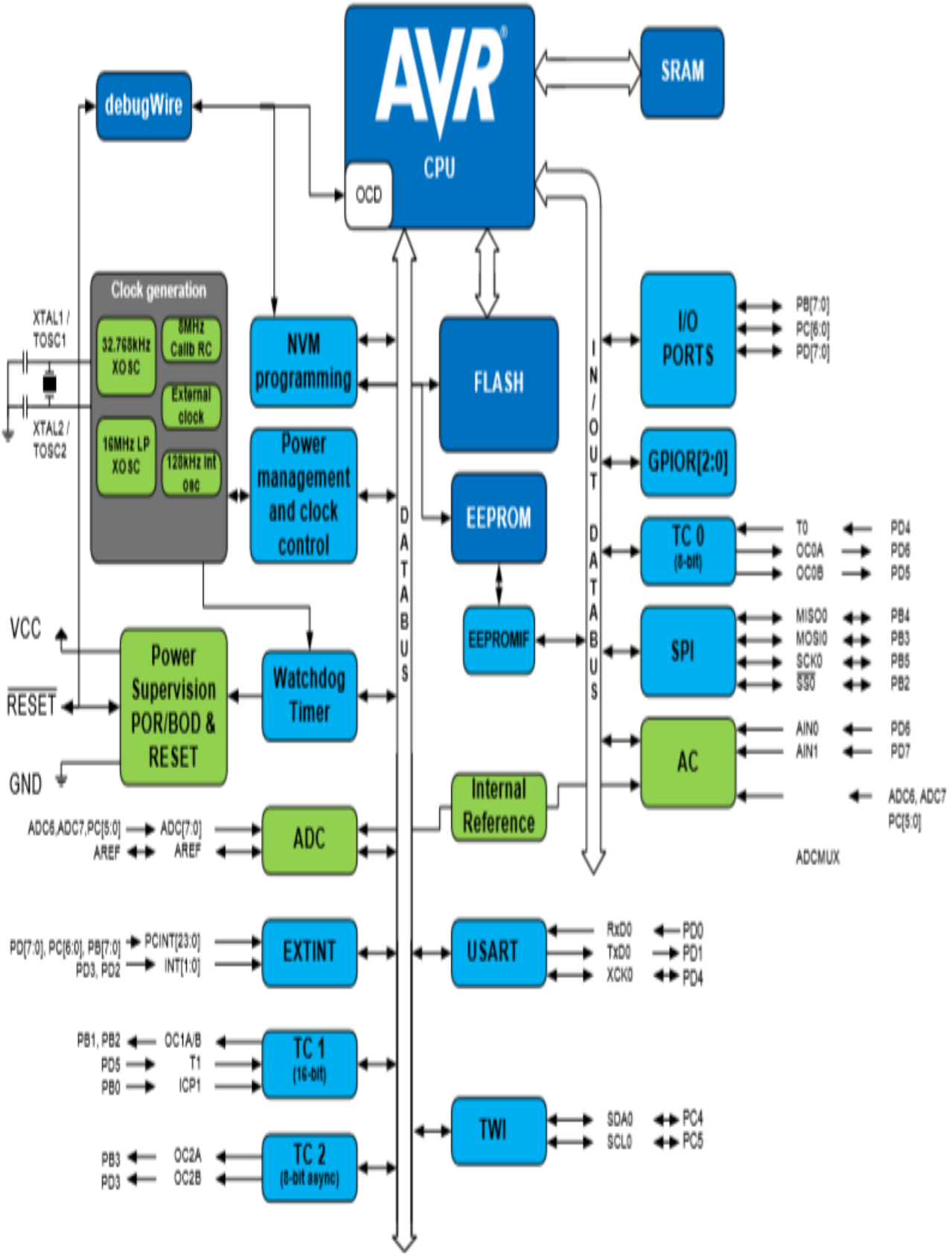
➔ **Opdracht 2:** bekijk met de lkr de PC opstelling op de plank. Ken jij alle onderdelen?

Gigatron architecture

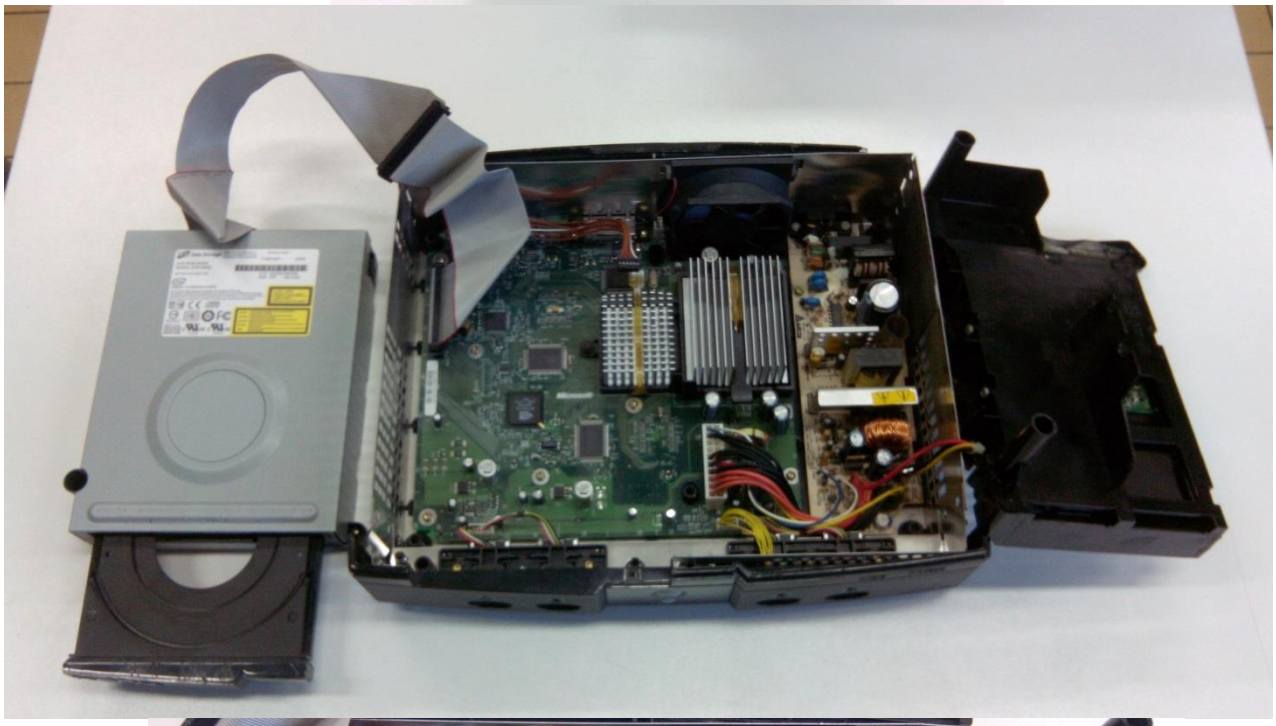


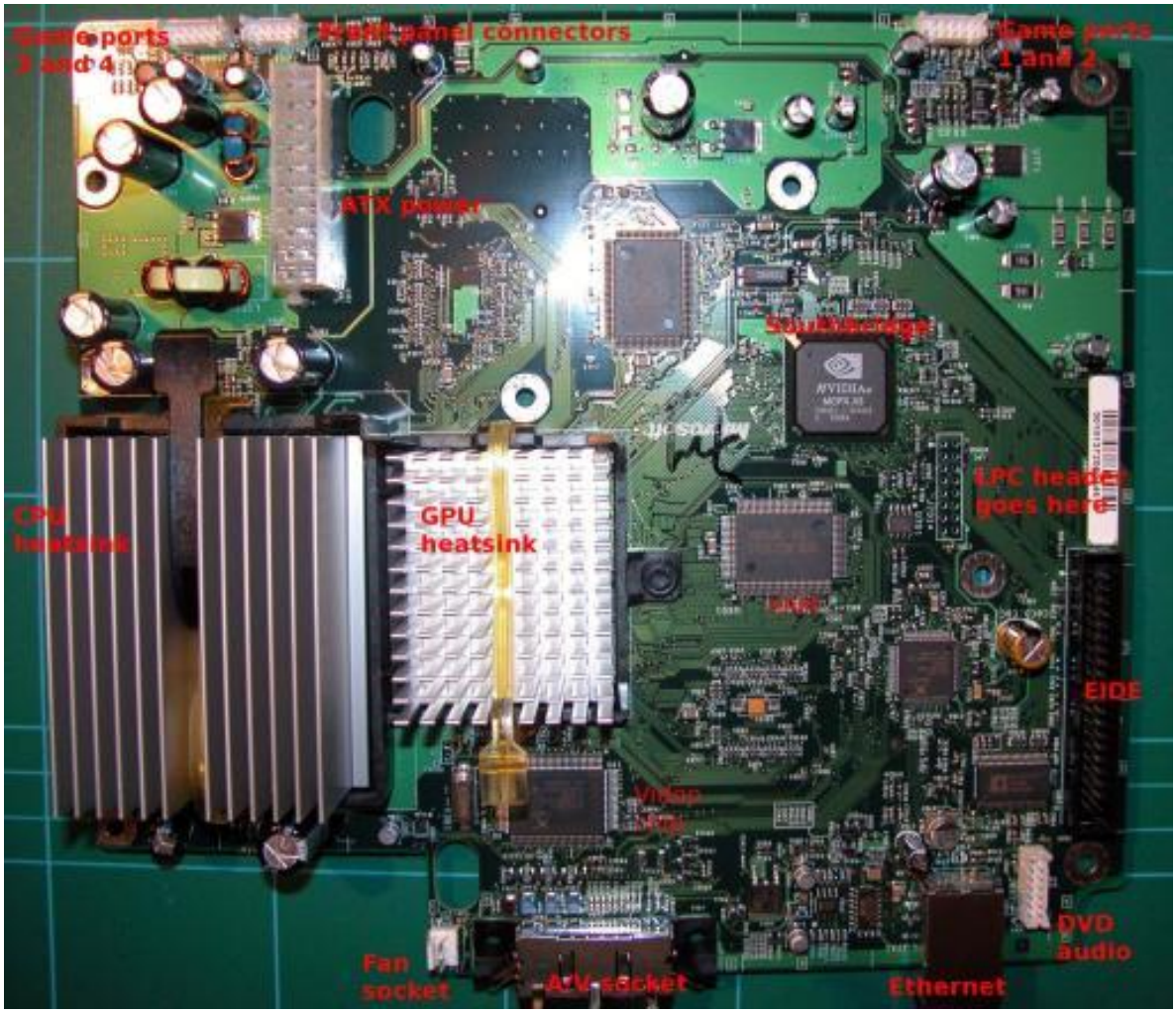
Merk op dat de gigatron (vóór de apple I) bestond uit losse IC's die samen een microcontroller vormen. Meer info vind je op https://gigatron.io/

Microcontroller architectuur MEGA328P



Praktische voorbeeld van een microcomputer systeem: XBOX





Kan jij op deze figure de hoofdblokken terugvinden?

Feitelijk is de Xbox een pc, maar omdat de software volledig aangepast is is hier weinig van te merken. De machine functioneert net als iedere andere spelcomputer. De specificaties zijn een aangepaste Pentium III, 64 MB geheugen, een grafische chip gebaseerd op de NVIDIA NV25, een harde schijf van 8 GB (in één bepaalde serie Xboxen 10 GB, maar waarvan maar 8 GB kon worden gebruikt), netwerkaansluiting en een dvd-speler. Het besturingssysteem van de Xbox is gebaseerd op de Windows NT-kernel en DirectX. Doordat de Xbox een (aangepaste) versie van DirectX gebruikt, is hij vooruit op de concurrentie. Doordat DirectX ook veel wordt gebruikt op pc, is het voor de programmeurs heel eenvoudig een spel over te zetten naar Xbox.

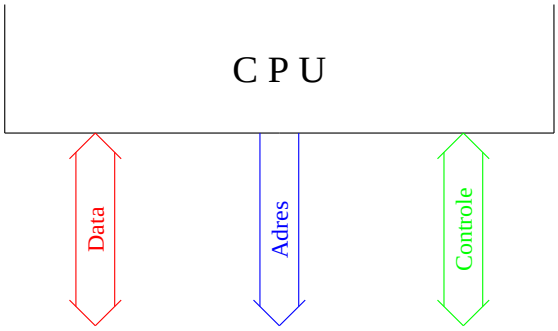
Verschil tussen de **AVR en de ARM** processor (op dit moment de meest voorkomende chips):

Most common microcontrollers

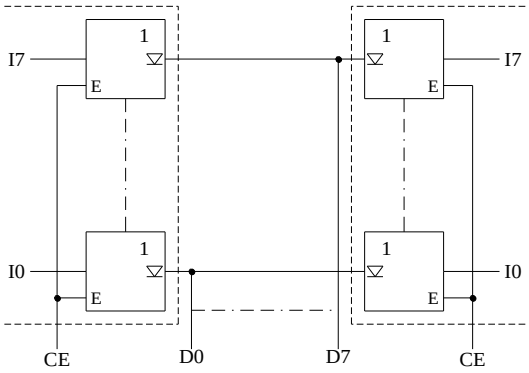
- 8-bit microcontrollers
  - AVR
  - PIC
  - HCS12
  - 8051
- 32-bit microcontrollers
  - ARM
  - AVR32
  - PIC32
  - CodeFire
  - PowerPC

5. Busstructuren

Wanneer we het over een bus hebben dan wordt ook de richting van de gegevenstransport op deze bus aangeduid. De richting wordt steeds bekeken vanuit de C.P.U. Zo zal in de meeste systemen de C.P.U. altijd bepalen waar er gegevens worden opgehaald of gelezen (read) of waar gegevens worden weg geplaatst of geschreven (write). Dit wil zeggen dat de C.P.U. de enige blok is die informatie op de adresbus plaatst. We zeggen dat de **adresbus unidirectioneel** is. Via de databus komen gegevens zowel de C.P.U. binnen als buiten. We zeggen dat de **databus bidirectioneel** is.



Op een bus mag op een bepaald moment slechts door één deel van de microcomputer informatie worden gezet. Ook moeten we blokken die op een bepaald moment geen gegevens willen uitwisselen via een bus tijdelijk van deze bus kunnen afschakelen. Hiervoor maken we gebruik van de **tri-state** technologie. Hierbij kan de toestand van een lijn naar hoog en laag ook een hoog-ohmige toestand aannemen (Z). Deze toestand wordt ook nog wel eens ‘floating’ genoemd.

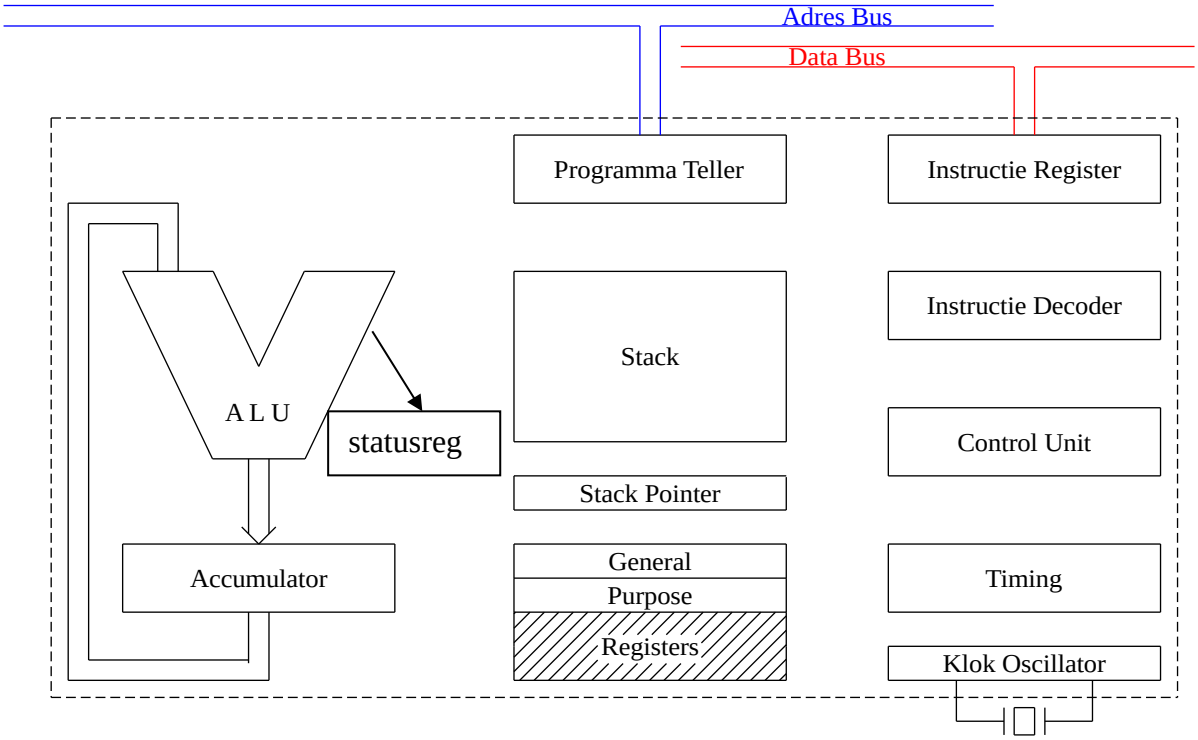


Diegenen die deze techniek wensen op te frissen kunnen misschien even de werking van de 74244 of 74245 bestuderen.

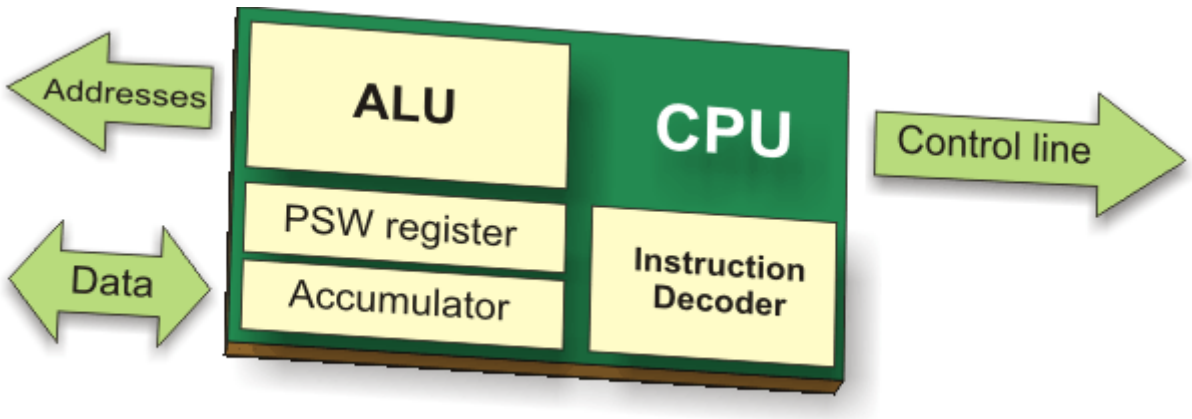
➔ **Taak:** neem de datasheet van de 2 vernoemde IC’s en zoek het **verschil** er tussen uit. Bekijk ook de WT.

6. CPU.

6.1. Blokschema van een CPU

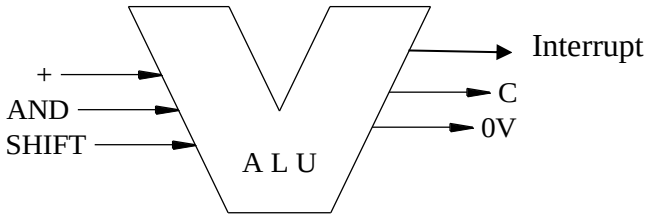


Schematische voorstellingen van een CPU

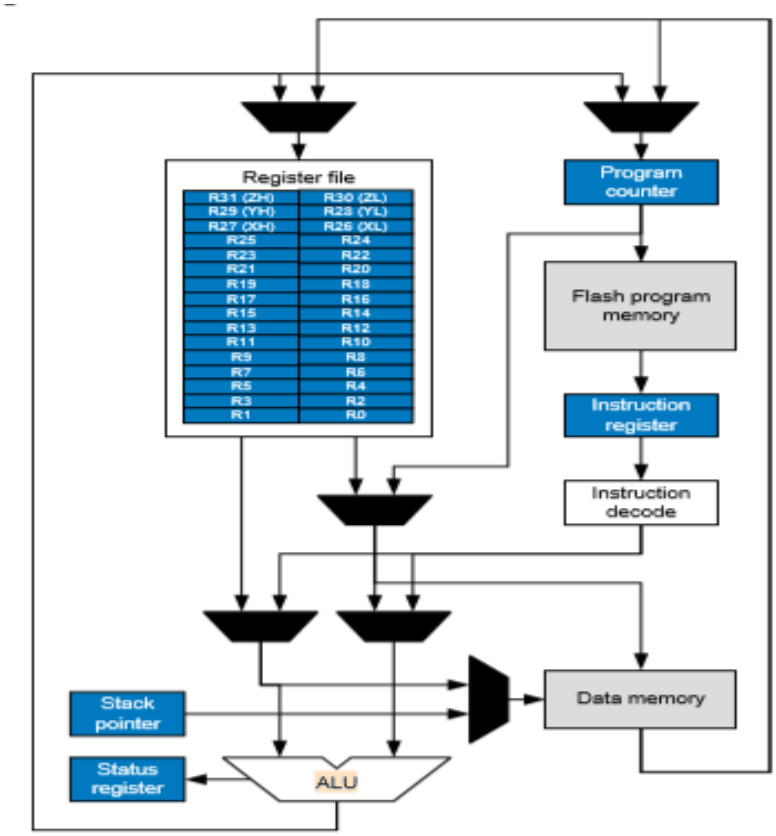


6.2. A.L.U.

De rekenkundige (+,-,x,/,…) en logische (and, or, not, exor,…) bewerkingen die een microprocessor moet uitvoeren, gebeuren in de A.L.U. (Arithmetic Logic Unit). Voor een groot deel van deze bewerkingen zijn twee gegevens nodig (bv. a+b). De gegevens die verwerkt worden bij de berekening worden **operanden** genoemd. De bewerking zelf noemen we de **opcode**. De operanden worden vanuit twee hulpregisters aan de ALU aangeboden en verwerkt. De ALU geeft ook **extra informatie** over de uitkomst van de bewerking die werd uitgevoerd. Hierbij denken we bijvoorbeeld aan een carrybit, zerobit (als de uitkomst nul is), overflowbit (tekenfout als we met negatieve getallen werken), enz . Deze 8 bits worden ook **flags of vlaggen** genoemd. Zij geven de status van de ALU weer in het **status register** (zie pg 25 en 28 in de MEGA328P datasheet).



Vb: opcode = IN, operanden = Rd en PORT, samen is dit de instructie: IN Rd, PORT



CPU architectuur uit de MEGA328P

➔ Noteer hier een **voorbeeld van de samenwerking** tussen de ALU en de geheugens, wanneer er 1 instructie van 1 byte wordt uitgevoerd:

Voorbeeld van een **logisch AND** commando in de AVR chipset van ATMEL:

AVR Instruction Set

AND – Logical AND

Description:

Performs the logical AND between the contents of register Rd and register Rr and places the result in the destination register Rd.

Operation:

(i) Rd ← Rd • Rr

	Syntax:	Operands:	Program Counter:
(i)	AND Rd,Rr	0 ≤ d ≤ 31, 0 ≤ r ≤ 31	PC ← PC + 1

16-bit Opcode:

0010	00rd	dddd	rrrr

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	⇔	0	⇔	⇔	–

S: N ⊕ V, For signed tests.

V: 0
Cleared

N: R7
Set if MSB of the result is set; cleared otherwise.

Z: $\overline{R7} \bullet \overline{R6} \bullet \overline{R5} \bullet \overline{R4} \bullet \overline{R3} \bullet \overline{R2} \bullet \overline{R1} \bullet \overline{R0}$
Set if the result is $00; cleared otherwise.

R (Result) equals Rd after the operation.

Example:

```
and   r2,r3     ; Bitwise and r2 and r3, result in r2
ldi   r16,1     ; Set bitmask 0000 0001 in r16
and   r2,r16    ; Isolate bit 0 in r2
```

Words: 1 (2 bytes)
Cycles: 1

➔ **Opdrachten:**

Duid in bovenstaande datasheet info de opcode, operator en instructie aan.

Teken er langs schematisch hoe de data flow tussen ALU en de registers/geheugen verloopt.

Duid ook aan welke vlaggen er belangrijk zijn voor deze instructie in het statusregister.

Hoeveel cycles vraagt het om deze instructie uit te voeren?

Zoek nu in de MEGA328P datasheet naar een gelijkaardige instructie. Noteer hier de pagina: ….

Meer info over het **Status register** van de MEGA328P:

11.3.1 Status Register

Name: SREG
Offset: 0x5F
Reset: 0x00
Property: When addressing as I/O Register: address offset is 0x3F

When addressing I/O registers as data space using LD and ST instructions, the provided offset must be used. When using the I/O specific commands IN and OUT, the offset is reduced by 0x20, resulting in an I/O address offset within 0x00 - 0x3F.

Bit	7	6	5	4	3	2	1	0
	I	T	H	S	V	N	Z	C
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – I Global Interrupt Enable
The global interrupt enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the Global Interrupt Enable register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the instruction set reference.

Bit 6 – T Copy Storage
The bit copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T-bit as source or destination for the operated bit. A bit from a register in the register file can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the register file by the BLD instruction.

Bit 5 – H Half Carry Flag
The half carry flag H indicates a half carry in some arithmetic operations. Half carry flag is useful in BCD arithmetic. See the *Instruction Set Description* for detailed information.

Bit 4 – S Sign Flag, S = N ⊕ V

The S-bit is always an exclusive or between the negative flag N and the two's complement overflow flag V. See the *Instruction Set Description* for detailed information.

Bit 3 – V Two's Complement Overflow Flag
The two's complement overflow flag V supports two's complement arithmetic. See the *Instruction Set Description* for detailed information.

Bit 2 – N Negative Flag
The negative flag N indicates a negative result in an arithmetic or logic operation. See the *Instruction Set Description* for detailed information.

Bit 1 – Z Zero Flag
The zero flag Z indicates a zero result in an arithmetic or logic operation. See the *Instruction Set Description* for detailed information.

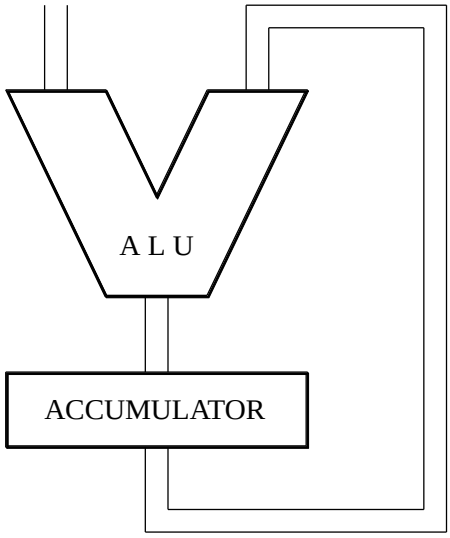
Meer info van een **deel van de instructieregisters** van de MEGA328P:

36. Instruction Set Summary

ARITHMETIC AND LOGIC INSTRUCTIONS					
Mnemonics	Operands	Description	Operation	Flags	#Clocks
ADD	Rd, Rr	Add two Registers without Carry	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add two Registers with Carry	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl,K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	Rd ← Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract two Registers with Carry	Rd ← Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract Constant from Reg with Carry.	Rd ← Rd - K - C	Z,C,N,V,H	1
SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	Rd ← Rd · Rr	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd · K	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	One's Complement	Rd ← 0xFF - Rd	Z,C,N,V	1
NEG	Rd	Two's Complement	Rd ← 0x00 - Rd	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd · (0xFF - K)	Z,N,V	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd ← Rd - 1	Z,N,V	1
TST	Rd	Test for Zero or Minus	Rd ← Rd · Rd	Z,N,V	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	Set Register	Rd ← 0xFF	None	1
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2

6.3. Accumulator

Het resultaat van de bewerkingen die gebeuren in de A.L.U. wordt opgeslagen in de accumulator (WREG = working register). Bij vele processors is de accumulator terug verbonden met één van de ingangen van de A.L.U. Dit is gedaan om het uitvoeren van kettingbewerkingen sneller te laten verlopen.

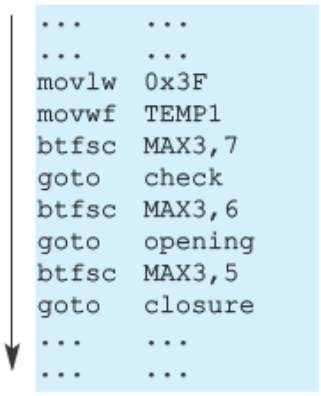


6.4. Programmateller

De microprocessor haalt de uit te voeren instructies op uit het programmageheugen. De instructies, die samen het programma vormen, zijn in opeenvolgend (sequentieel) in het programmageheugen opgeslagen. De CPU haalt deze instructies achtereenvolgens uit het programmageheugen om te bepalen welke acties hij moet ondernemen. Dit houdt echter wel in, dat de CPU moet weten op welke plaats de op te halen instructie staat. De plaats in het geheugen (adres) waar de eerst volgende uit te voeren instructie staat wordt bijgehouden in de programmateller. De programmateller (PC = program counter) wordt automatisch met één verhoogt (geïncrementeerd) nadat een instructie is opgehaald zodat de programmateller dan het adres van de volgende uit te voeren instructie bevat. De programmeur dient de instructies dan wel zo in het geheugen op te slaan, dat ze op opeenvolgende plaatsen in het geheugen staan.

6.5. Instructieregister

Nadat de microprocessor zijn instructie heeft opgehaald uit het geheugen komt de instructie terecht in het instructieregister.



Voorbeeld van instructies in een programma

Wat is **RISC**?

REDUCED INSTRUCTION SET COMPUTER

In dit geval maakt de processor enkel gebruik van **basis operaties.** Als hij moeilijker instructies moet uitvoeren zal hij gebruik maken van een **combinatie van de basisoperaties**. Zo wordt dan een vermenigvuldiging omgezet in een grote optelsom. Als gebruiker zie je echter enkel het resultaat en niet wat de processor intern allemaal uithaalt met de instructies. Deze instructieset is makkelijk aan te leren.

Wat is **CISC**?

COMPLEX INSTRUCTION SET COMPUTER

Dit is het omgekeerde van de RISC. In dit geval zijn er **meer dan 200 verschillende instructies** waarmee je kan gaan werken. Je moet dan natuurlijk meer kennis hebben van de set om alles goed te programmeren.

Welke SET kies ik dan wanneer?

Stel voordat je iets wil gaan maken met een processor de volgende vragen:

- Hoeveel I/O heb ik nodig?
- Heb ik slechts simpele instructies / bewerkingen nodig?
- Moet de processor enkele een relay of dergelijke aansturen?
- Wat mag het kosten? Hoeveel keer ga ik iets maken/verkopen?
- Is de snelheid kritisch?

6.6. Instructiedecoder

De instructiedecoder gaat de instructie die zich in het instructieregister bevindt decoderen en zien of alle informatie aanwezig is om de instructie uit te voeren. Indien dit nog niet het geval is, worden de volgende gegevens (operanden) uit het geheugen gehaald. Wanneer alle gegevens aanwezig zijn decodeert de instructiedecoder de volledig instructie. Het is dus nodig dat de instructiedecoder alle instructies kent die een microprocessor kan uitvoeren. De verzameling van alle instructies van een bepaalde microprocessor noemen we de instructieset (zie datasheet). De instructieset van een bepaalde processorfamilie kan je verkrijgen bij de fabrikant ervan. Het vervelende is echter dat de instructieset van processorfamilie tot processorfamilie sterk verschilt.

6.7. Control unit

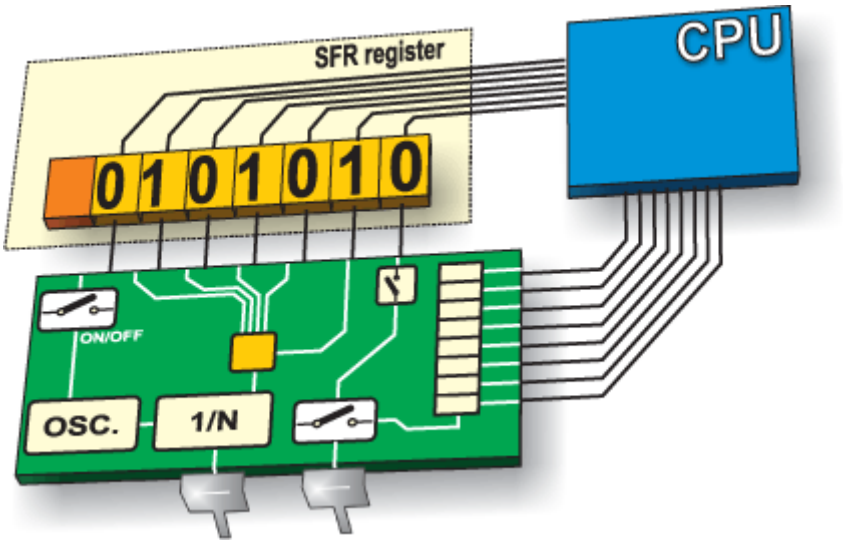
Vanaf het moment dat de gehele instructie is gedecodeerd neemt de control unit de taak over en zal alle nodige handelingen doen om de instructie uit te voeren (execution). Eénmaal de instructie volledig is uitgevoerd kan de processor beginnen met het ophalen van de eerstvolgende uit te voeren instructie.

6.8. Stack en stackpointer

In een processor is meestal een klein stukje geheugen aanwezig dat je kan gebruiken om kleine dingen tijdelijk te bewaren. Je kan het vergelijken met een klein notaboekje. Dit geheugen werkt echter volgens een speciaal principe dat we **LIFO** noemen. Dit staat voor **Last In First Out**. Dit wil zeggen dat het gegeven dat je er laatst instopte er ook eerst terug uit moet worden genomen. Dit geheugen noemen we de stack. De stackpointer bevat het adres van de eerstvolgende vrije geheugenplaats in de stack.
Merk op dat wanneer de stackpointer wijst naar adres 0 er niets in de stack zit.

6.9. General purpose registers

Dit zijn registers die vrij bruikbaar zijn **voor verschillende toepassingen**. Meestal hebben general purpose registers (GPR) een speciale functie voor sommige instructies. Bij onze PIC noemt dit het SFR (special function register). Deze SFR's worden zowel gebruikt door de "core" (ALU, reset, interrupt) als de peripheral functions (I/O).



Voorbeeld van een SFR: via het register wordt bijvoorbeeld een timer ingesteld.

11.4 General Purpose Register File

The register file is optimized for the AVR Enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the register file:

- One 8-bit output operand and one 8-bit result input
- Two 8-bit output operands and one 8-bit result input
- Two 8-bit output operands and one 16-bit result input
- One 16-bit output operand and one 16-bit result input

Figure 11-2. AVR CPU General Purpose Working Registers

	7 0	Addr.	
	R0	0x00	
	R1	0x01	
	R2	0x02	
	...		
	R13	0x0D	
General	R14	0x0E	
Purpose	R15	0x0F	
Working	R16	0x10	
Registers	R17	0x11	
	...		
	R26	0x1A	X-register Low Byte
	R27	0x1B	X-register High Byte
	R28	0x1C	Y-register Low Byte
	R29	0x1D	Y-register High Byte
	R30	0x1E	Z-register Low Byte
	R31	0x1F	Z-register High Byte

Vb van general purpose register in de MEGA328P

6.10. Timing en klok

De processor is een digitale schakeling die werkt volgens een vast ritme. In een microprocessor zit meestal een oscillator die een timingschakeling aanstuurt. Deze timingschakeling zorgt ervoor dat al de acties van de processor volgens een vast tijdschema verlopen. De frequentie waarop de oscillator oscilleert kan bepaalt worden door het kristal dat wordt aangesloten. In sommige gevallen kan ook een resonator worden aangesloten. Het ritme waarop de processor werkt is bij sommige processors een deeltal van aangesloten klokfrequentie.

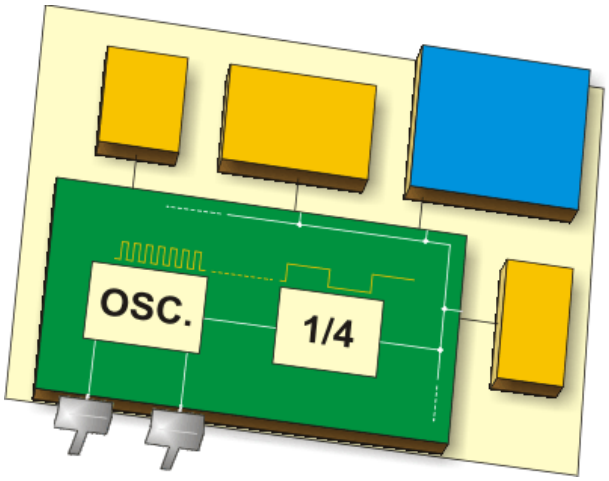
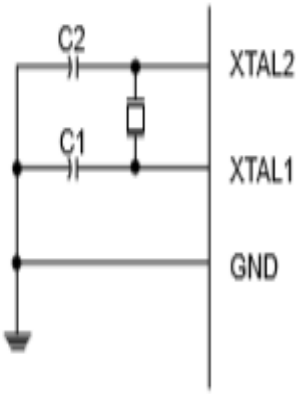
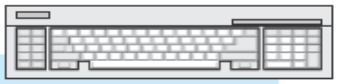


Figure 13-2. Crystal Oscillator Connections



Nog eens de stappen die er gebeuren tussen de c-code en de bits die uiteindelijk weggeschreven worden in de chip.



```
void main() {

   TRISB = 0;              // All port B pins are configured as
                           // outputs
   PORTB = 0b01010101;     // Logic state on port B pins
}
```

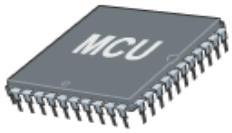
Program written in C

```
; ADDRESS     OPCODE        ASM
; ------------------------------------------
$0000         $2804         GOTO   _main
$0004         $       _main:
;Test.c,1 :: void main() {
;Test.c,3 :: TRISB = 0;              // All port B pins
$0004         $1303         BCF    STATUS, RP1
$0005         $1683         BSF    STATUS, RP0
$0006         $0186         CLRF   TRISB, 1
;Test.c,4 :: PORTB = 0b01010101; // Logic state
$0007         $3055         MOVLW  85
$0008         $1283         BCF    STATUS, RP0
$0009         $0086         MOVWF  PORTB
;Test.c,5 :: }
$000A         $280A         GOTO   $
```

Compiled Program

```
:100000000428FF3FFF3FFF3F03138316860155304F
:10001000831286000A28FF3FFF3FFF3FFF3FFF3F5D
:04400E00F22FFFFF8F
:00000001FF
```

Executable Code of the program (HEX code)



Voorbeeld van een instructie dewelke na de c-code (na flowcode dus) omgezet wordt in assembler en daarna nog eens naar een HEX file code. Deze laatste wordt dan in het geheugen van een processor geladen en bevat alle nodige instructies.

Meer info over **assembler** nodig?

Kijk naar de assembler **basiscursus** in de **elektor** van juli –aug, sept-okt 2015.

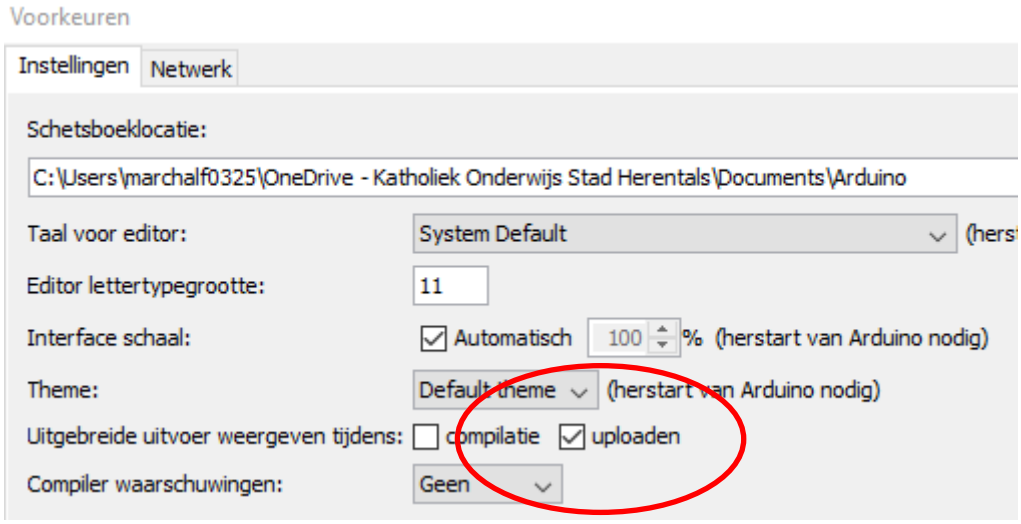
➔ **Taak:** installeer het **AVR simulatie IDE programma van Oshon.**

Laad een **hex file** van een eenvoudig Arduino programma in (vb stuur een LED aan) en test dit stap per stap uit op deze simulator, ingesteld als een **ATMEGA328P. Volg hiervoor de volgende stappen:**

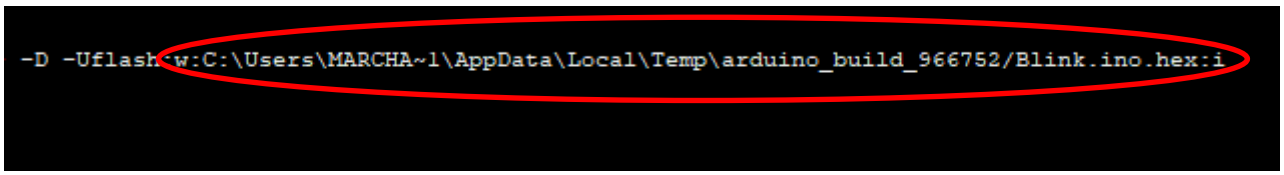
Waar staat de hex file?

Haal bijvoorbeeld het "blink" voorbeeld op uit arduino. Pas de delay aan naar 10 us ipv 1000 ms. Type hiervoor delayMicroseconds(10)

Stel bij **File -> Voorkeuren** in de Arduino IDE de setting "uitgebreide uitvoer weergeven tijdens" "**uploaden**" in (vinkje zetten).

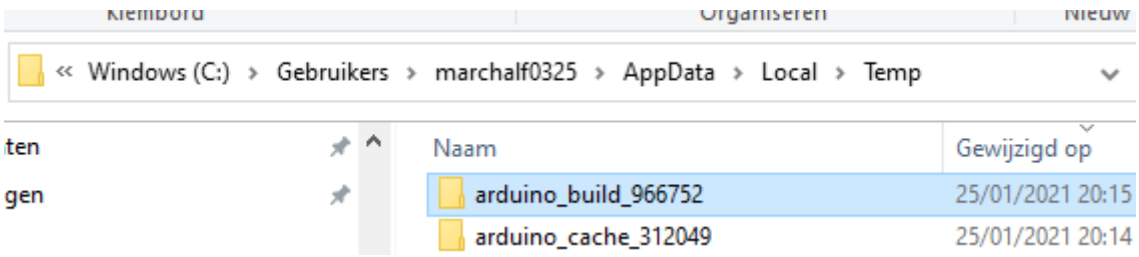


Druk nu op **uploaden** (mag zonder uno). Doe dan het debug venster open onder de code en ga op zoek naar de plaats waar de **hex file** wordt getoond.

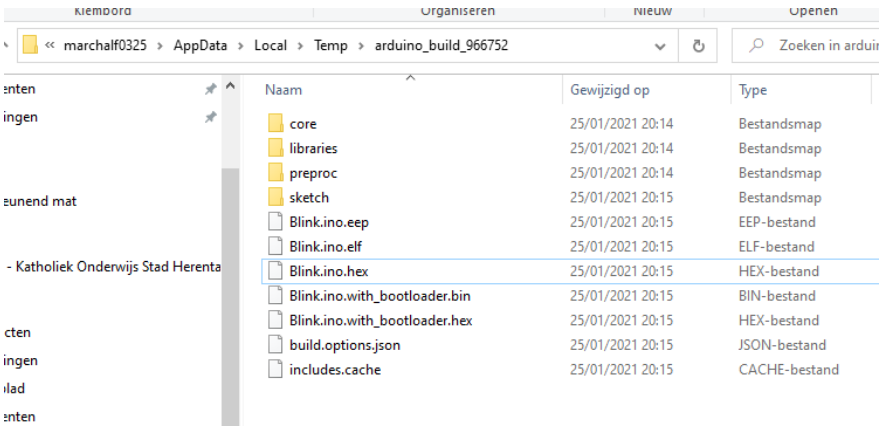


Ga nu op jouw PC op zoek naar deze hex file:

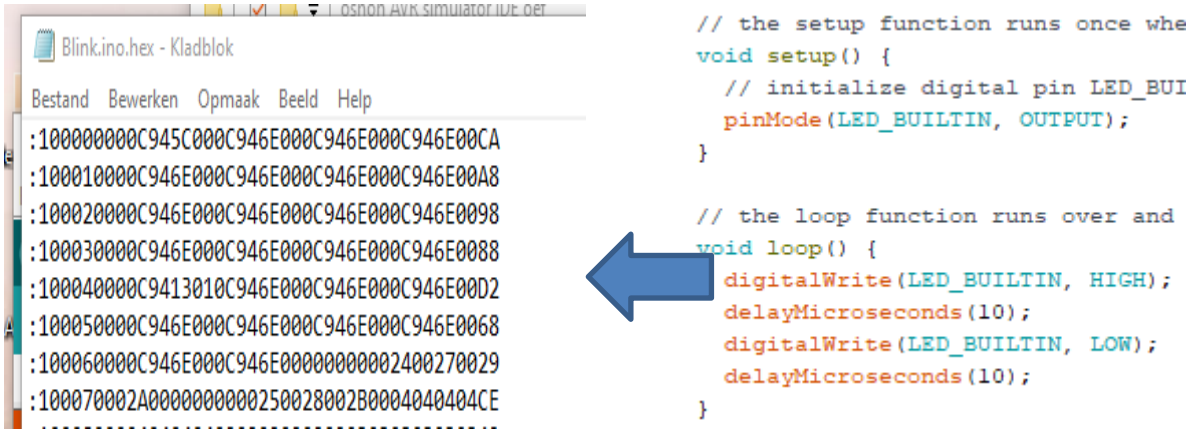
C:\Users\MARCHA~1\AppData\Local\Temp\arduino_build_966752/Blink.ino.**hex**



Maak een kopie van de hex file en plak deze in een mapje op jouw werkplek waar je met de simulator aan de slag wil gaan:



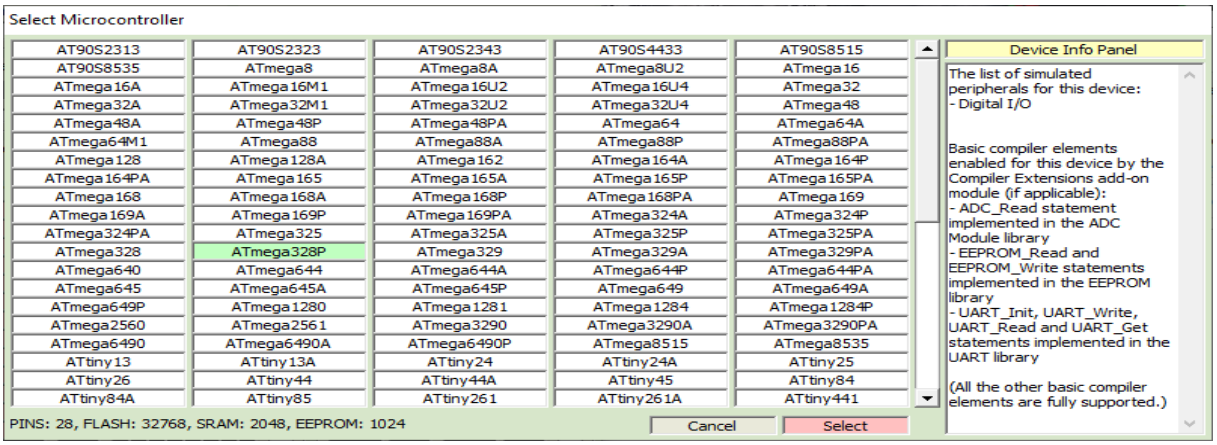
Open de hex file eens met een kladblok programma:



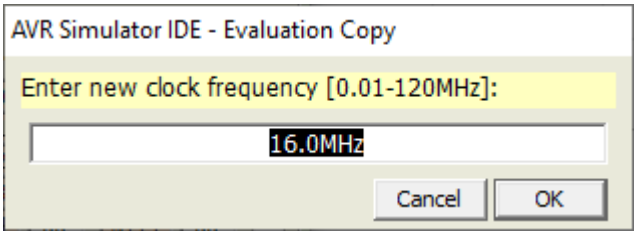
Enkel nog hexadecimale getallen blijven er over van jouw programma.

Open de simulator en stel de volgende settings in:

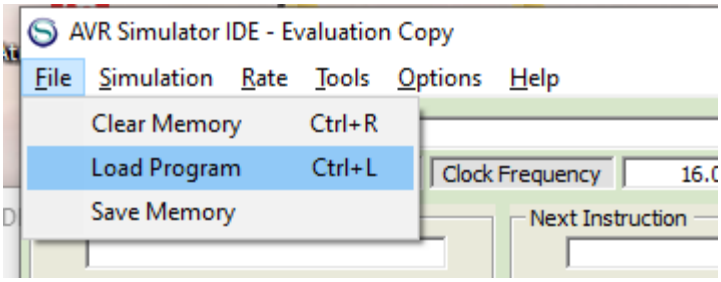
- Microcontroller : MEGA 328P (merk op dat de tool ook de verschillende geheugens toont)



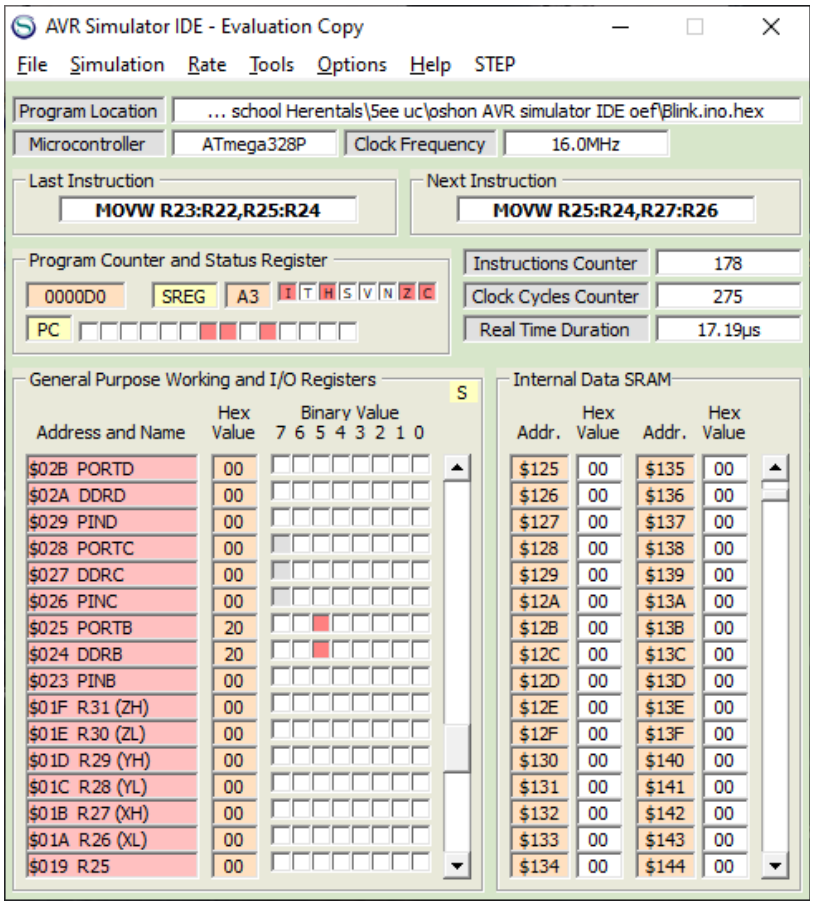
- Stel de frequentie in: 16MHz



- Laad de hex file in het flash geheugen via de File -> Load Program menu.



- Start nu de simulator in “Normal” mode.
- Hou het PORTB register op bit 5 (PB5) in het oog. Dit is de pin van onze built in LED (D13).



Na ongeveer elke 10 us gaat de LED op PB5 aan en uit.

-Houd ook de instruction counter (PC) en de statusregister (SREG) in het oog.
Na elke instructie verandert bij deze laatste de inhoud constant, afhankelijk van de bewerking (operand).

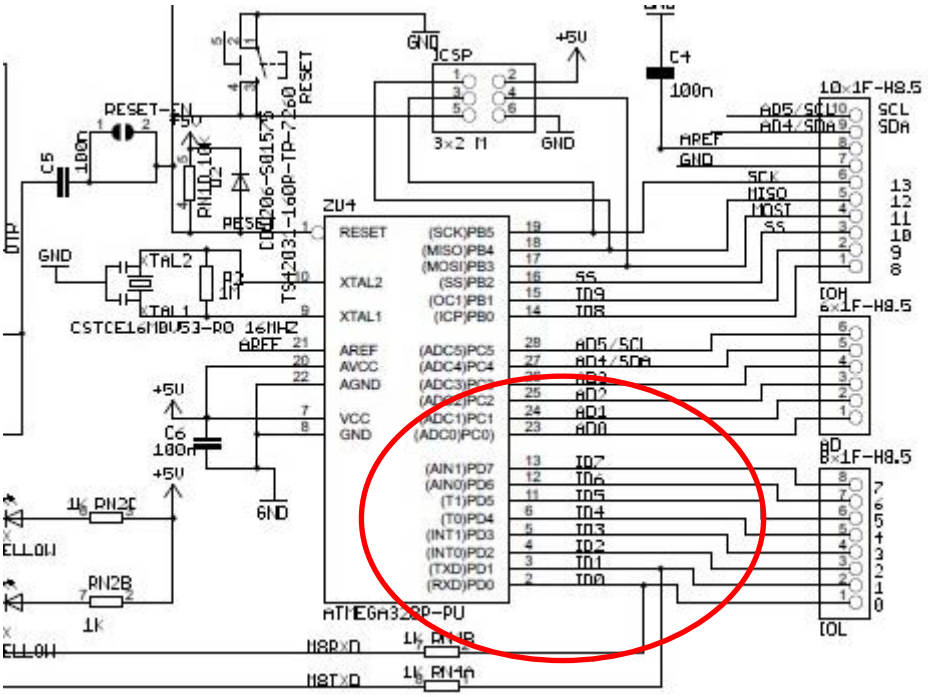
- Merk op dat het SRAM geheugen bij deze oefening niet wordt aangesproken.
- Doordat de controller 10 us moet wachten is deze gewoon heel veel niets aan het doen. Stel u voor als je daar 1000 ms = 1 seconde in geeft in de code.

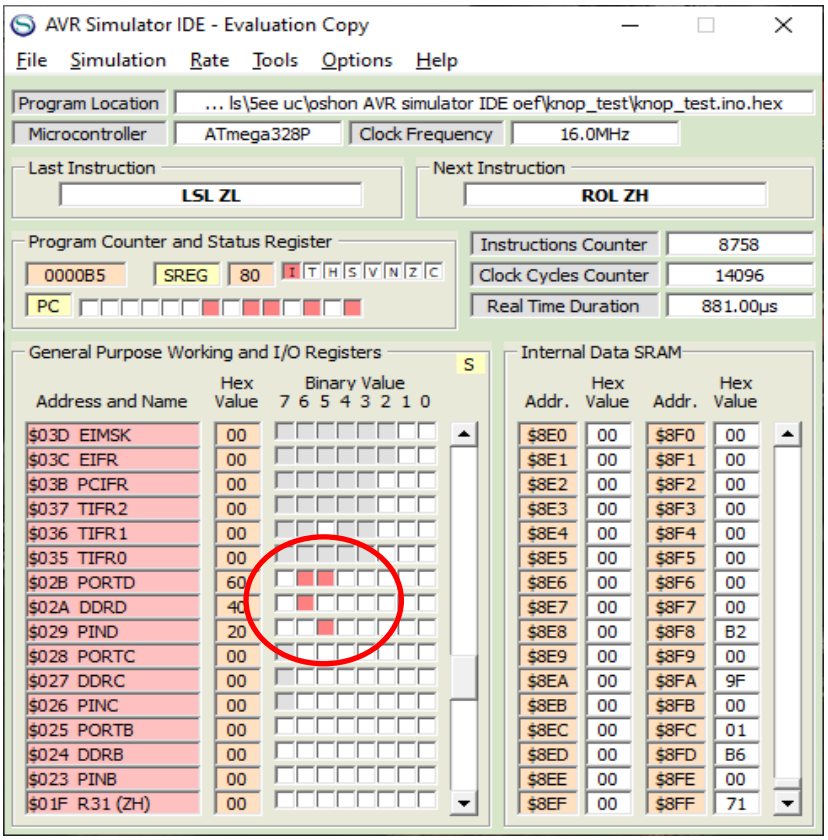
- "Clear" nu het geheugen nadat je de simulator hebt gestopt.

- Je kan nu ook een programma maken met een variabele en een knop.

Upload opnieuw deze oefening, zoek de hex file op jouw PC en load deze in de simulator.

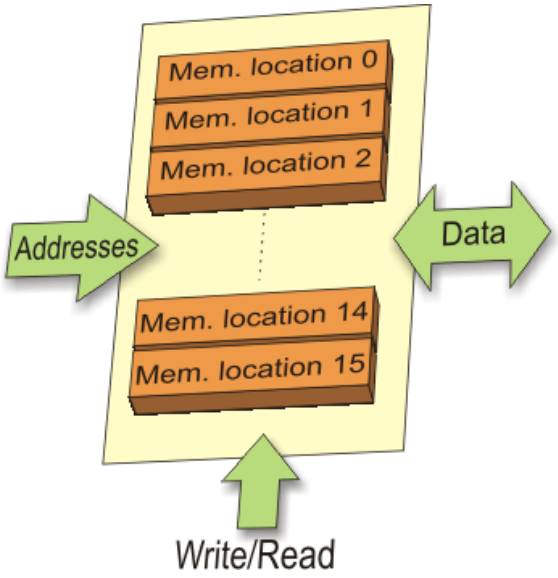
- In onze oefening is Input 5 (D5) in de werkelijke wereld van arduino ook PD5.
- Voor D6, waar de knop aan zit, is dit ook PD6.





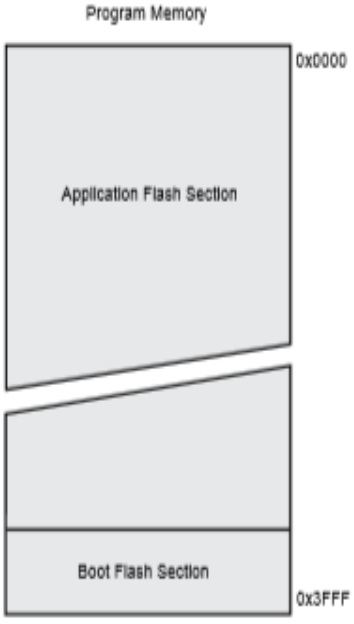
Run de simulator FAST om iets te kunnen zien. Klik op de bit PORTD5 om de knop te simuleren.

7.Geheugenstructuur



In het eerste schema op de volgende pagina hebben we een onderscheid gemaakt tussen het programma geheugen en het data geheugen. Het zijn **twee aparte blokken** in onze computer. Er zijn nu twee manieren om deze twee blokken in het geheugen te zetten. Stel dat we beschikken over een 16-bit adresbus. Deze laat ons toe om 2^{16} = 65.536 verschillende geheugenplaatsen te adresseren. Hexadecimaal gaat het bereik van 0000H tot FFFFH. Merk op dat het **programma geheugen** ook een **stukje bootloader** bevat, terwijl het **werk/data geheugen** ook bestaat uit een **deel SRAM en een deel EEPROM**.

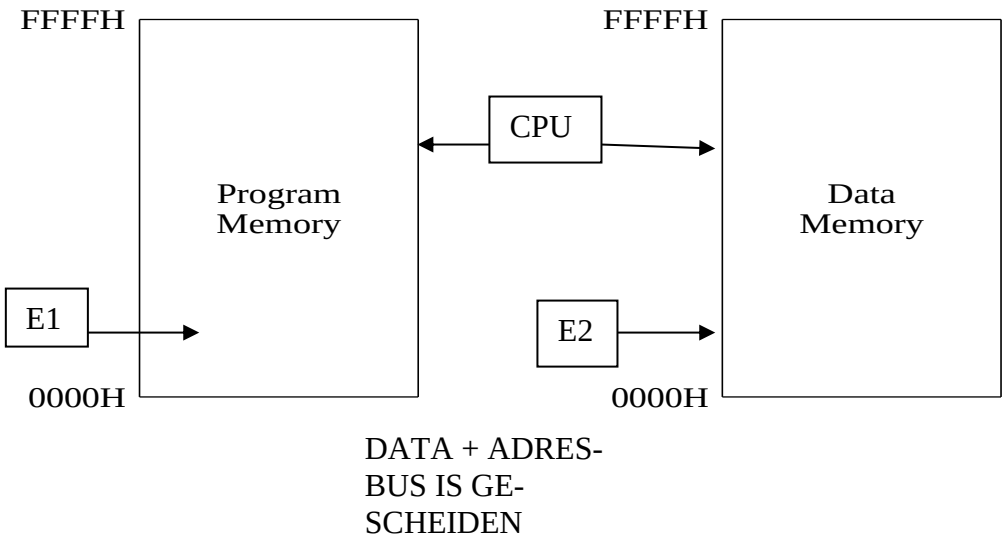
Figure 12-1. Program Memory Map ATmega328/P

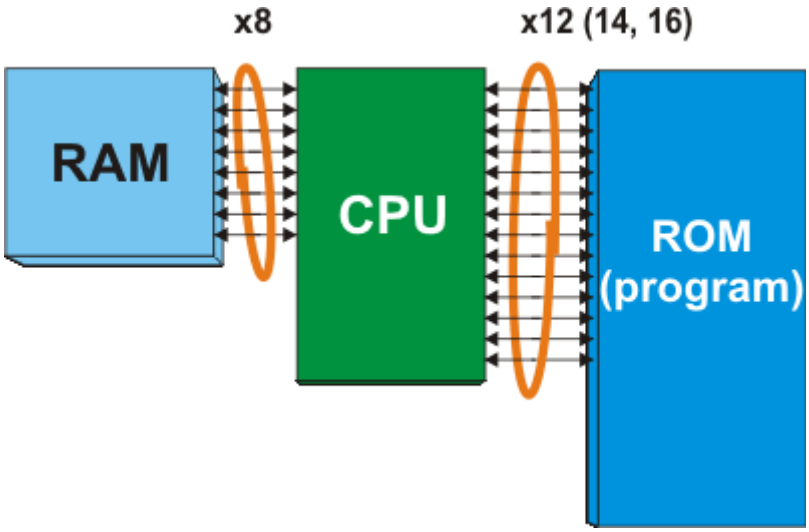


7.1. Harvardstructuur

Sommige processors zijn uitgerust om **zowel voor** het **datageheugen** als voor het **programmageheugen** evenveel **geheugenplaatsen** te voorzien (dit hoeft niet altijd gelijk te zijn). Zij gebruiken een **enablelijn** voor het datageheugen en een andere enablelijn voor het programmageheugen. Deze geheugenstructuur noemen we de Harvard-geheugenstructuur. Vermits beide bussen gescheiden zijn kunnen we ze beiden ook **tegelijk aanspreken**.

Dit geeft tijdwinst tijdens het verwerken van instructies (haal instructie op (FETCH) terwijl de vorige reeds wordt verwerkt (EXECUTE).



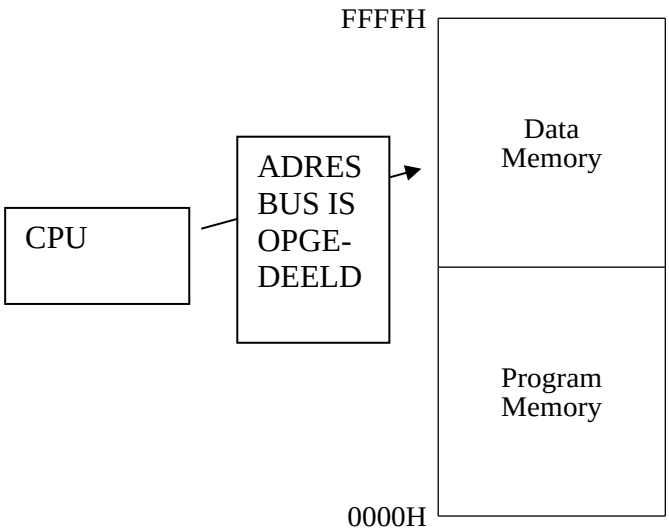


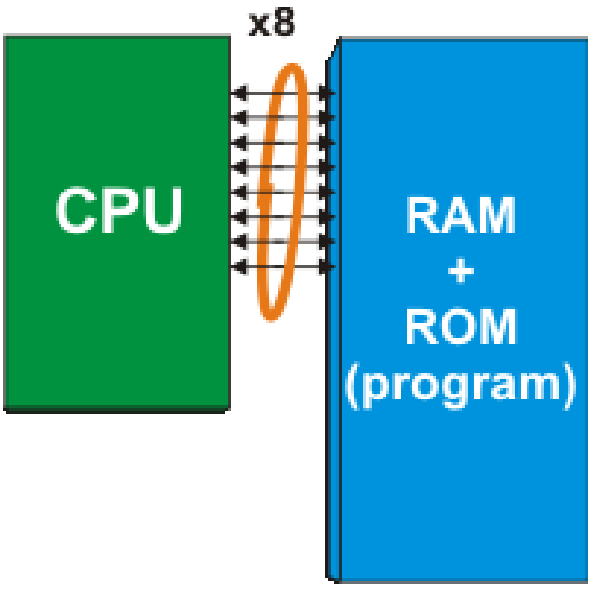
Voorstellingen van een HARVARD structuur

7.2. Von Neumannstructuur

Processores die deze voorzieningen niet hebben en het **totale geheugenbereik** van geheugenplaatsen moeten **verdelen** over zowel data als programmageheugen werken volgens de Von Neumann geheugen-structuur. Dit is de vroegere methode maar is **kwa snelheid dus minder interessant**.

Voorstellingen van een Von Neumann structuur:





7.3. Aangepaste (Modified) Harvard structuur

Neem hier nota van de Modified Harvard structuur.

Wat is er zo speciaal aan deze structuur? …….

Zie https://en.wikipedia.org/wiki/Modified_Harvard_architecture voor meer info

Instruction-memory-as-data architecture [edit]

Another change preserves the "separate address space" nature of a Harvard machine, but provides special machine operations to access the contents of the instruction memory as data. Because data is not directly executable as instructions, such machines are not always viewed as "modified" Harvard architecture:

- Read access: initial data values can be copied from the instruction memory into the data memory when the program starts. Or, if the data is not to be modified (it might be a constant value, such as pi, or a text string), it can be accessed by the running program directly from instruction memory without taking up space in data memory (which is often at a premium).
- Write access: a capability for reprogramming is generally required; few computers are purely ROM-based. For example, a microcontroller usually has operations to write to the flash memory used to hold its instructions.[2] This capability may be used for purposes including software updates. EEPROM/PROM replacement is an alternative method.

Taak:

Ga op zoek op het internet info over het **FLASH, EEPROM en SRAM/DRAM** geheugen.
Maak hier een kort verslag van en noteer erbij hoe de **werking** is van elk soort geheugen. Ook een **toepassing, verschillen, voor/nadelen** en een **foto** per geheugen moet je toevoegen.

Opdracht:

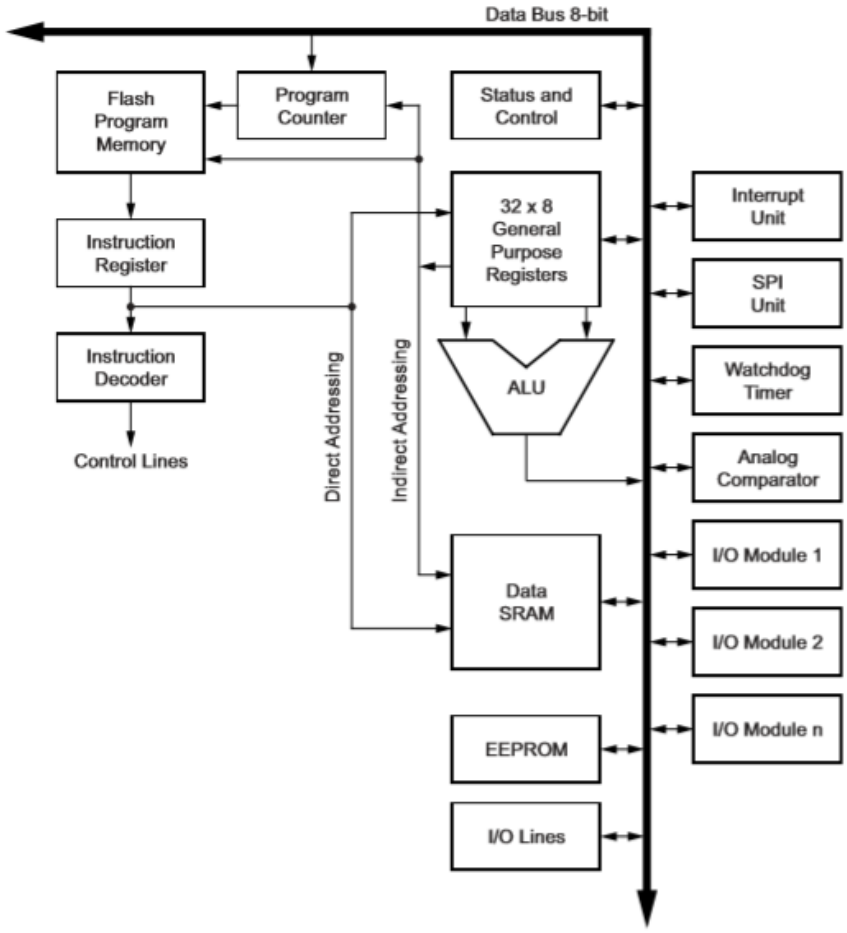
Kan je deze 3 geheugens ook terugvinden in de MEGA328P? Zie datasheet!

Flash (Bytes)	32K
SRAM (Bytes)	2K
EEPROM (Bytes)	1K

Welke geheugen structuur zit er dan in de MEGA328P?

Noteer hier de pg van de datasheet waar je de oplossing kan vinden: …….

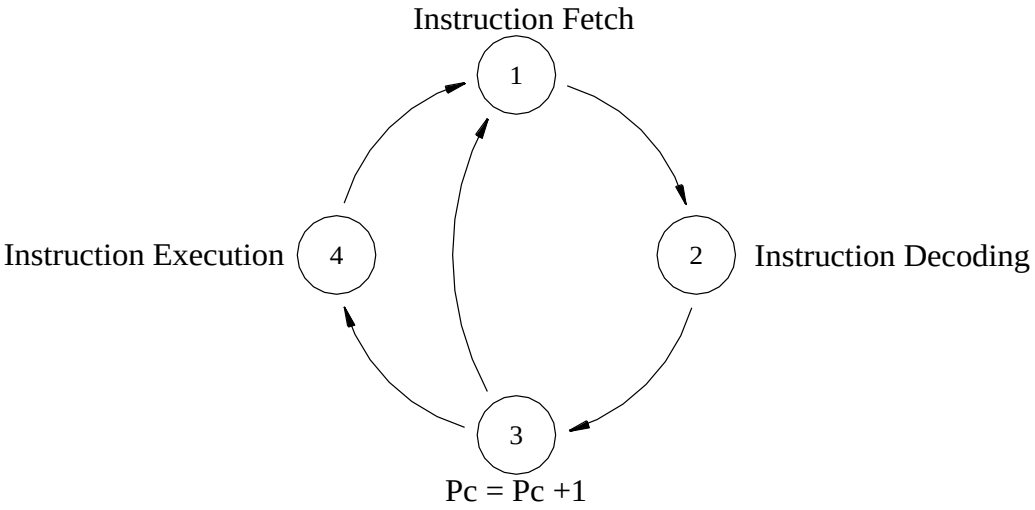
Figure 6-1. Block Diagram of the AVR Architecture



In order to maximize performance and parallelism, the AVR uses a harvard architecture – with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is in-system reprogrammable flash memory.

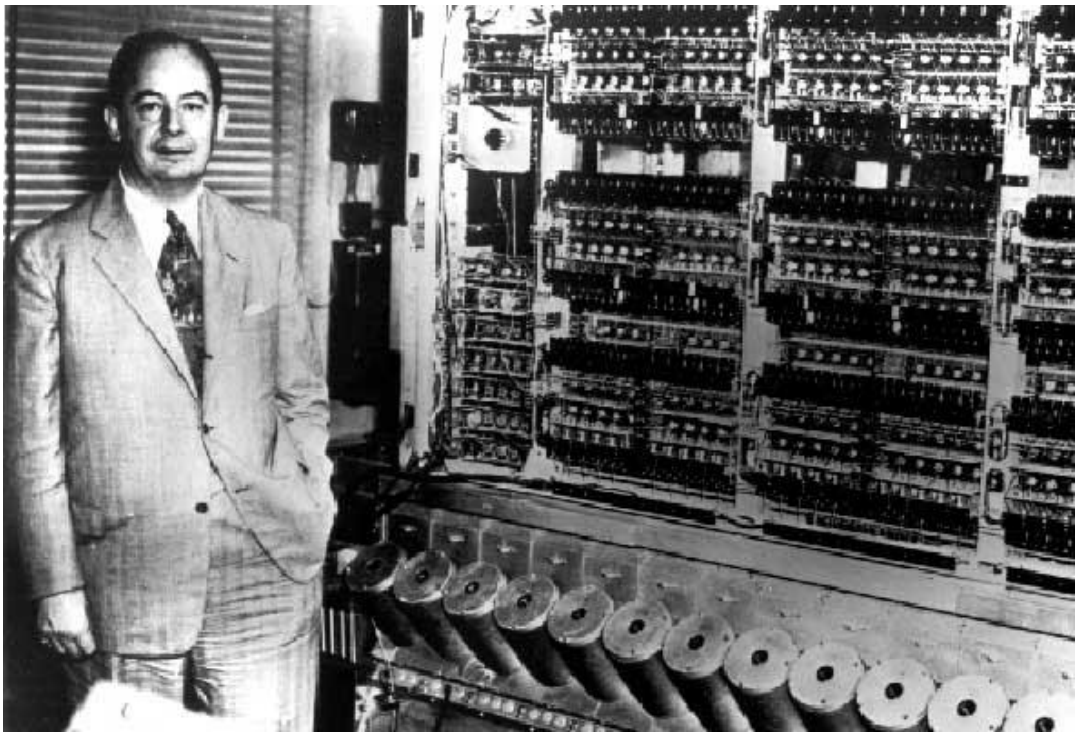
8.Von Neumanncyclus

De activiteiten van een CPU zijn cyclisch. Dit wil zeggen dat er een vast patroon zit in de handelingen die de processor verricht. De CPU haalt namelijk steeds een instructie op het geheugen, voert die uit, haalt de volgende instructie enz… We gaan eens van naderbij bekijken hoe dit juist in zijn werk gaat.

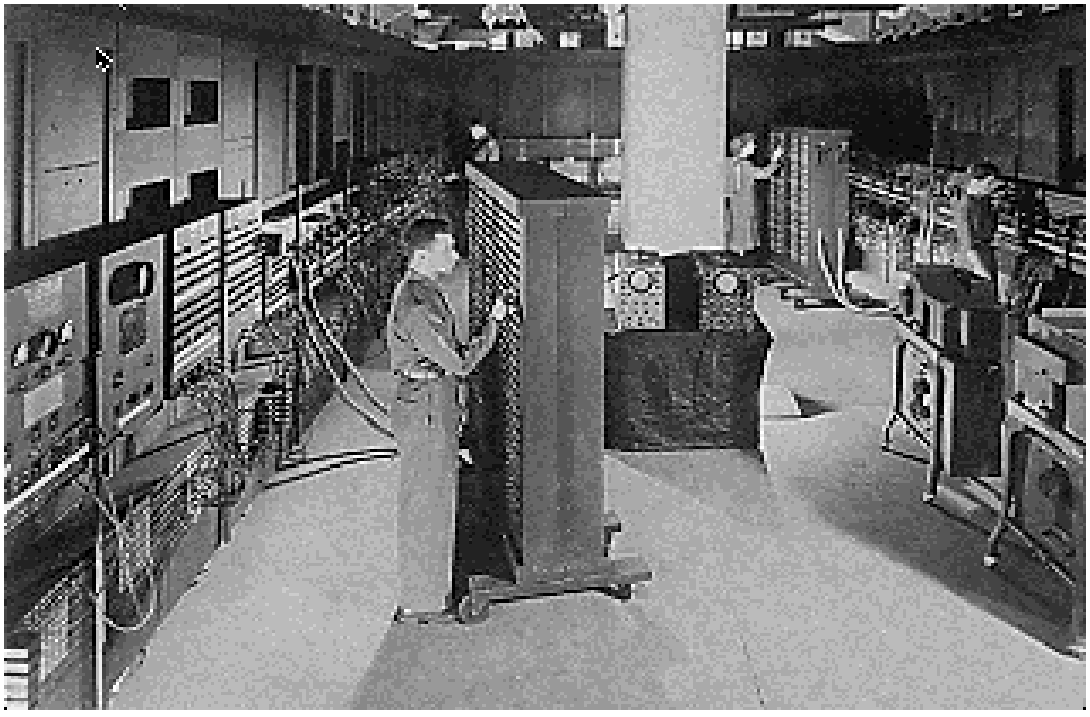


De **eerste fase** in het proces is de **instruction fetch.** Het ophalen van de instructie (instruction fetch) uit het programmageheugen gebeurt in twee stappen. In de eerste stap stuurt de CPU het adres, dat door de programmateller wordt aangegeven, via de adresbus naar het programmageheugen. Het bij dit adres horende geheugenwoord wordt geselecteerd. In de tweede stap geeft het programmageheugen de inhoud van het geadresseerde geheugenwoord af aan de CPU. De CPU bergt deze inhoud op in het instructieregister. In een **tweede fase** van het proces gaat de **instructiedecoder** de instructie in het register bekijken. Voor sommige instructies heeft de CPU aan dit ene instructiewoord voldoende informatie om deze instructie uit te voeren. Men spreekt dan van een één-byte-instructie. Er zijn echter ook instructies die een **tweede of soms nog een derde byte nodig** hebben om de uitvoering van de instructie te kunnen voltooien
In de **derde fase** van het proces wordt de **programmateller** met één verhoogt. Indien we te maken hebben met een twee-byte-instructie of een drie-byte-instructie zal de CPU nog één of twee bytes uit het geheugen ophalen alvorens de uit te voeren instructie volledig is. Eénmaal alle bytes binnen zijn kan de CPU aan de **vierde fase** van het proces beginnen namelijk het uitvoeren van de volledige instructie. Deze fase noemen we de "**instruction execution**". Deze manier van werken staat bekend als de Von Neumann-cyclus.

Johann Von Neumann was een Hongaars wiskundige die leefde van 1903 tot 1957. Hij creëerde in de veertiger jaren een computer die volgens dit principe werkte. Tegenwoordig werken de meeste computer **nog steeds volgens dit principe**. Ook het principe van parallel processing is een idee van Von Neumann.



John Von Neumann aan het werk aan de IAS

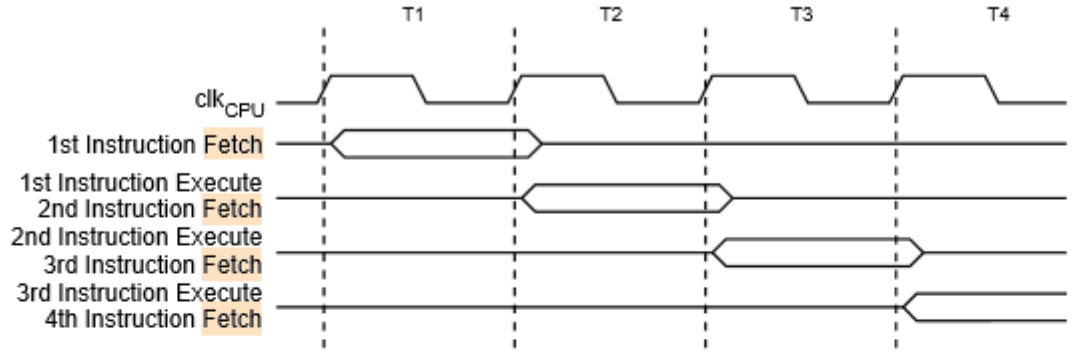


De ENIAC, ook een computer met de Von Neumann architecture (zo groot als een kamer)

Instruction Execution Timing

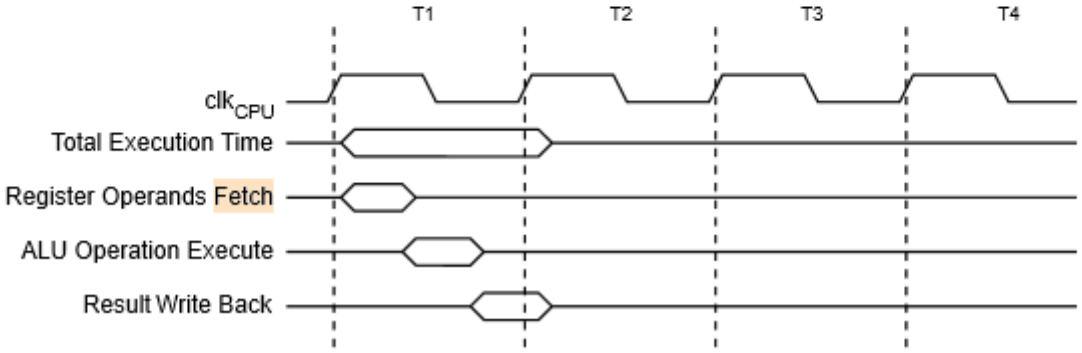
This section describes the general access timing concepts for instruction execution. The AVR CPU is driven by the CPU clock clk_{CPU}, directly generated from the selected clock source for the chip. No internal clock division is used. The figure below shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access register file concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power unit.

Figure 11-4. The Parallel Instruction Fetches and Instruction Executions

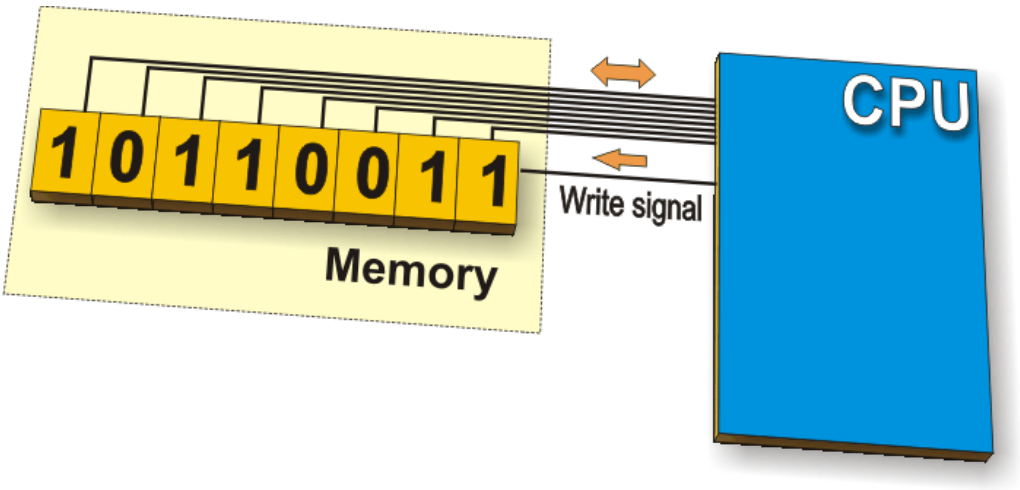


The following figure shows the internal timing concept for the register file. In a single clock cycle, an ALU operation using two register operands is executed and the result is stored back to the destination register.

Figure 11-5. Single Cycle ALU Operation



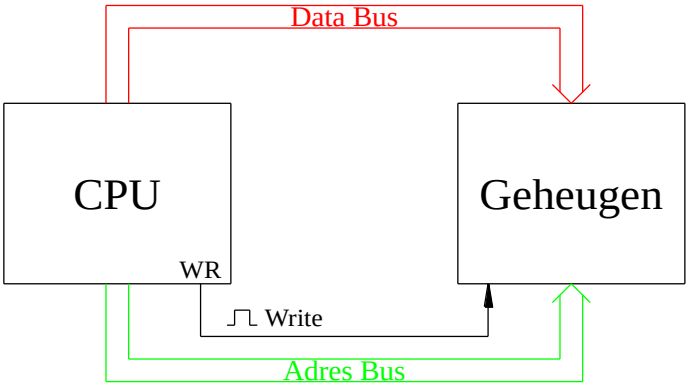
9.Gegevensstroom in een microcomputer



9.1. Van CPU naar geheugen.

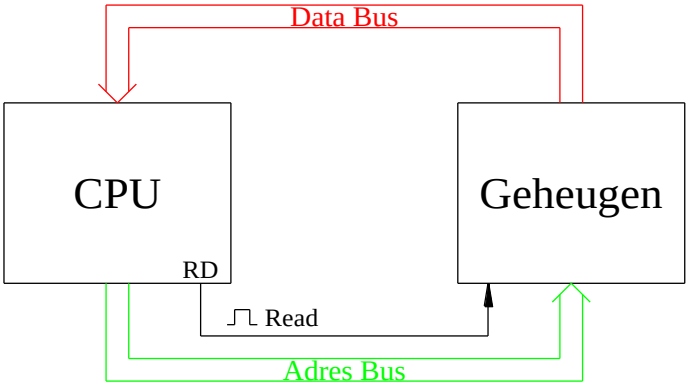
De memory write operatie = schrijven in het geheugen, wordt uitgevoerd om data vanuit de CPU op een geselecteerde geheugenplaats te schrijven. Dit gebeurt in volgende stappen.

1. Het adres van de geheugenplaats, waar de data in moet worden weg geschreven, gaat via de **adres-bus** van de CPU naar het geheugen.
2. Het adres wordt door de **adresselectie** van het geheugen gedecodeerd.
3. De **CPU zendt** de data, via de databus naar het geheugen.
4. Wanneer deze data stabiel is zal de CPU een **writepuls** geven, dat een schrijfoperatie aangeeft.
5. Het geselecteerde geheugen wordt **gevuld met de data**.



9.2. Van geheugen naar CPU

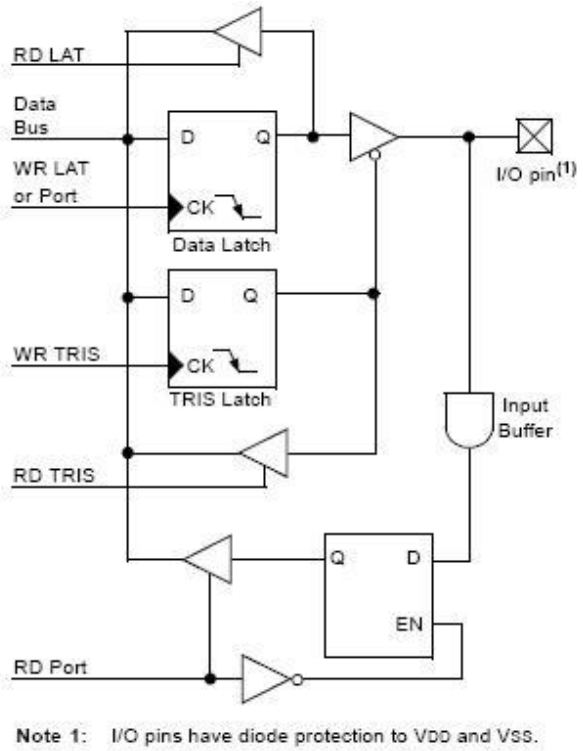
1. Het adres van de geheugenplaats, waar de data moet worden gelezen, gaat via de **adresbus** van de CPU naar het geheugen.
2. Het adres wordt door de **adresselectie** van het geheugen gedecodeerd.
3. De CPU zal een **readpuls** aan het geheugen geven, dat een leesoperatie aangeeft.
4. De inhoud van het geselecteerde geheugen wordt via de **databus** naar de CPU gezonden.



Op eenzelfde manier kan met alle randcomponenten worden gecommuniceerd.

10. Werking van een GPIO pin van de processor

We hebben reeds gesproken over de I/O pinnen die meer als 1 functie of feature kunnen ondersteunen. Onderstaande tekening geeft aan hoe een **algemene I/O pin** ingebouwd is **in de MEGE328P**. Met "algemene" wordt hier bedoeld dat er geen extra features zijn voorzien op deze pin (zoals PWM, INT, I2C,…). Je kan dus enkel de pin als **DIGITAAL IN of OUT** schakelen.



Teken hier de vereenvoudigde versie:

Voor het **schrijven (WR)** van bijvoorbeeld een '1' op de uitgang zal er eerst een "1" op de **Data Bus** worden geplaatst. Na het **latchen** van de data, waarbij een klok puls wordt gegeven aan de 'CK' van de Data Latch, zal de data op de Q uitgang komen te staan. Wanneer nu de **TRISTATE enabled** is zal de '1' op de I/O pin verschijnen.

Wanneer **de I/O pin als input** is gedefinieerd wordt bijvoorbeeld een '1' via de **input buffer** aangevoerd naar de onderste data latch in de tekening. Ondertussen is de **TRISTATE** van het output circuit **disabled**. Nadat de **RD port actief** is gemaakt zal de data verschijnen op de Q van de data latch. Uiteindelijk kan dan de data via de onderste tristate component op de **Data Bus** geplaatst worden. Zo kan de microcontroller de '1' nu verder verwerken.

Natuurlijk wordt het schema van een I/O port alleen maar complexer wanneer het meerdere features ondersteunt. Al deze settings **kunnen ingesteld worden via de SFR's** (Special Function Registers). Hierover kan je meer lezen in de datasheet.
Ten gepaste tijde zullen we deze schema's bespreken in de verdere hoofdstukken wanneer zulk speciaal feature aanbod komt.

Wat is de werking van een DFF? ……

11 Hoe werkt de klok precies van de microcontroller?

Waar staat MIPS voor? …….

Op hoeveel MIPS draait de MEGA328P ? ……..

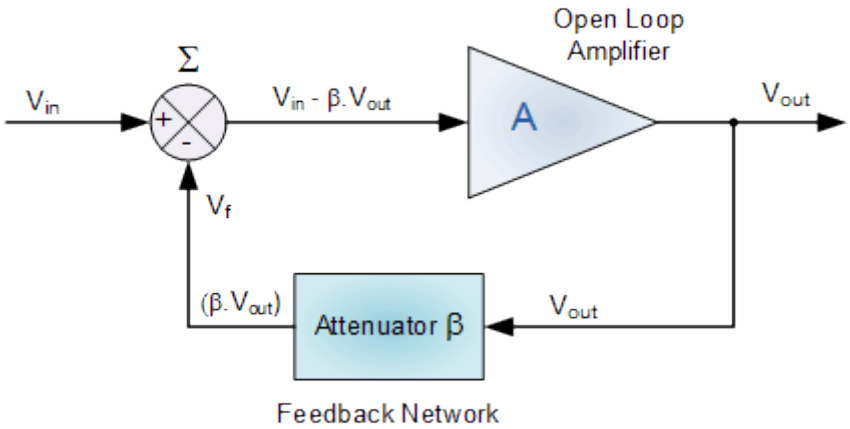
Zie ook https://www.electronics-tutorials.ws/oscillator/oscillators.html

De werking van de oscillator staat hieronder ter info. We bespreken deze later in de cursus Arduino gevorderden 2.

Wat zijn de voorwaarden om een oscillator te maken?

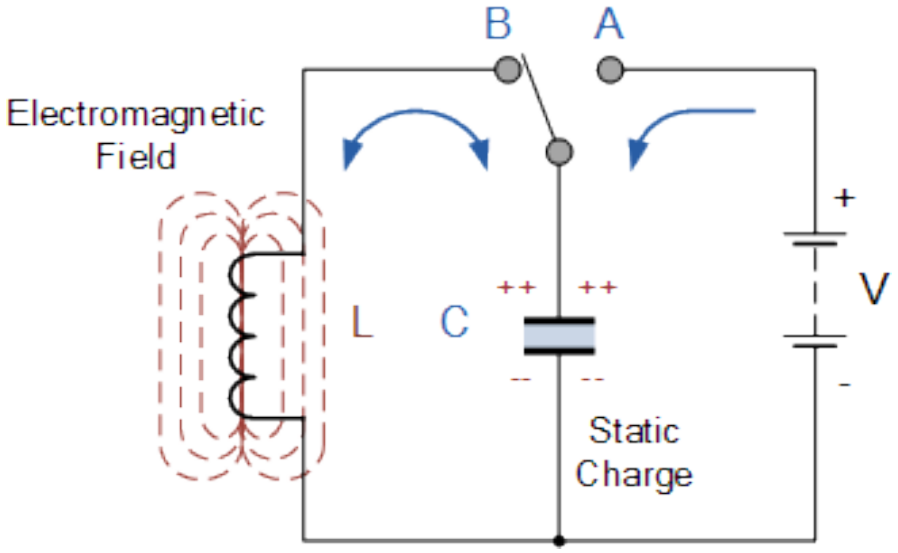
Wanneer kan een schakeling gaan oscilleren? Daarvoor moet er aan enkele voorwaardes voldaan worden:

1. De totale versterking A . B van gans de schakeling moet 1 zijn.
2. De rondgaande faseverschuiving moet 0 zijn.

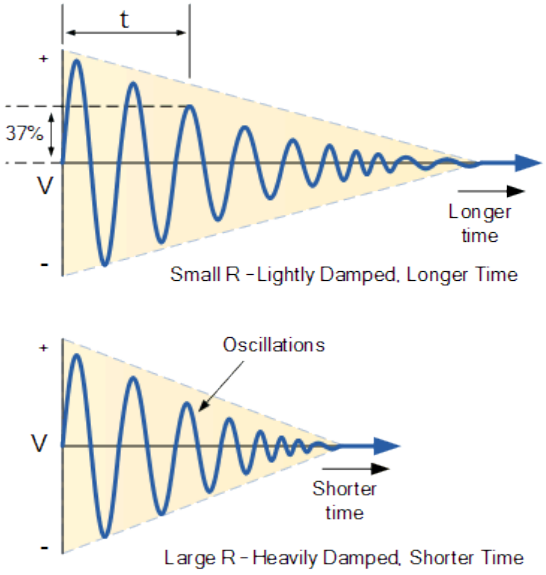


A = de versterking van de schakeling
B = de verzwakking door de frequentie selectieve LC tank

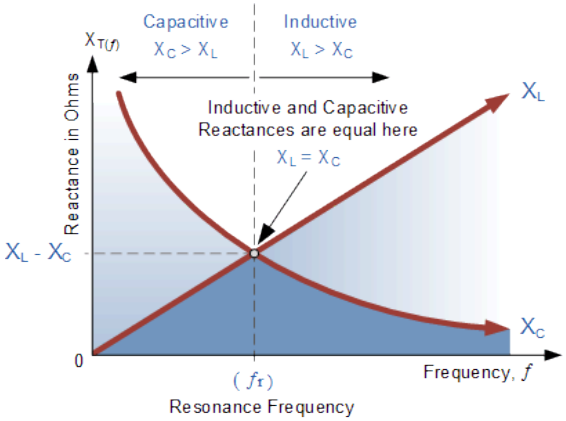
Hoe werkt een LC tank?



De schakeling bevat een spoel en een condensator.
De condensator slaat energie op in de vorm van een elektrostatisch veld. De spoel slaat energie op in de vorm van elektromagnetisch veld.
Als we de knop in stand A zetten wordt de condensator geladen tot de spanning V. Als de condensator volledig is opgeladen zetten we de knop in stand B.
De condensator ontlaadt nu in de spoel. De spanning van de condensator zakt en de stroom van de spoel stijgt. De volledig ontladen spanning van de condensator wordt nu als een elektromagnetisch veld opgeslagen in de spoel.
Als er nu geen externe spanning zit in het circuit, dan gaat het magnetisch veld van de spoel terug afnemen. Een tegen emk wordt geïntroduceerd en houdt de stroom in de oorspronkelijke richting.
De stroom laadt nu de condensator weer op totdat het magnetisch veld van de spoel helemaal verdwenen is. De energie is dus nu teruggekeerd naar de condensator. De polariteit van de condensator is echter omgekeerd.
Nu begint de condensator terug met ontladen in de spoel en het hele proces herhaald zich opnieuw. De polariteit van de spanning verandert constant en de energie wordt in een sinusvormige spanning geproduceerd.
Natuurlijk zijn er energieverliezen in de componenten en zal de energie na verloop van tijd nul zijn.
De LC tank zorgt dus voor demping van het signaal.

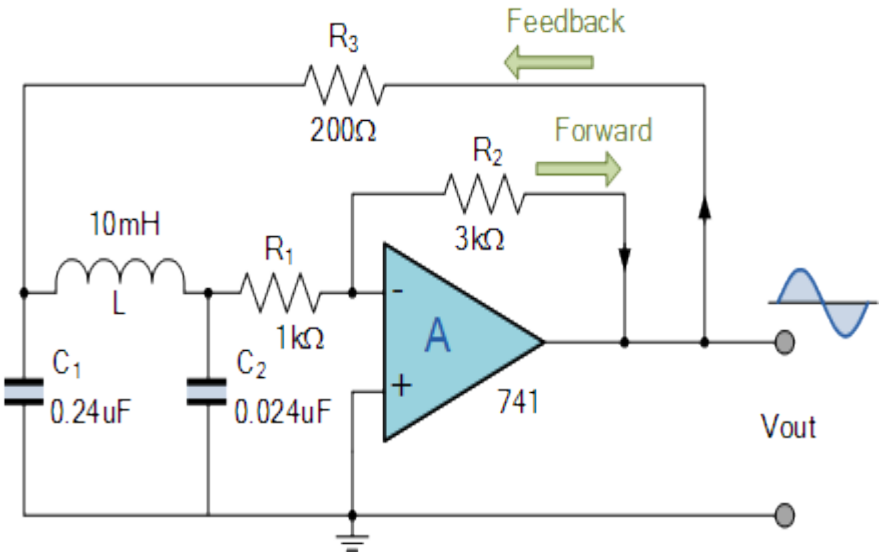


De frequentie hangt af van de inductantie XL en de capacitive reactantie XC van in de LC tank.
Er is 1 punt waar XC en XL gelijk zijn, en dat is de resonantiefrequentie fr.



$$f_r = \frac{1}{2\pi\sqrt{LC}}$$

Via onze versterker, gekoppeld aan de LC tank gaan we telkens weer wat energie toevoegen, zodat de LC tank kan blijven uitwisselen.



De opamp versterker zorgt voor een -180 graden versterking. De LC tank zorgt ook voor een selectieve filtering van de frequentie en een verschuiving van 180 graden. Dus de totale faseverschuiving is 0 graden. Als je R2 regelbaar maakt kan je A . B = 1 afregelen en gaat de oscillator constant dezelfde frequentie produceren.

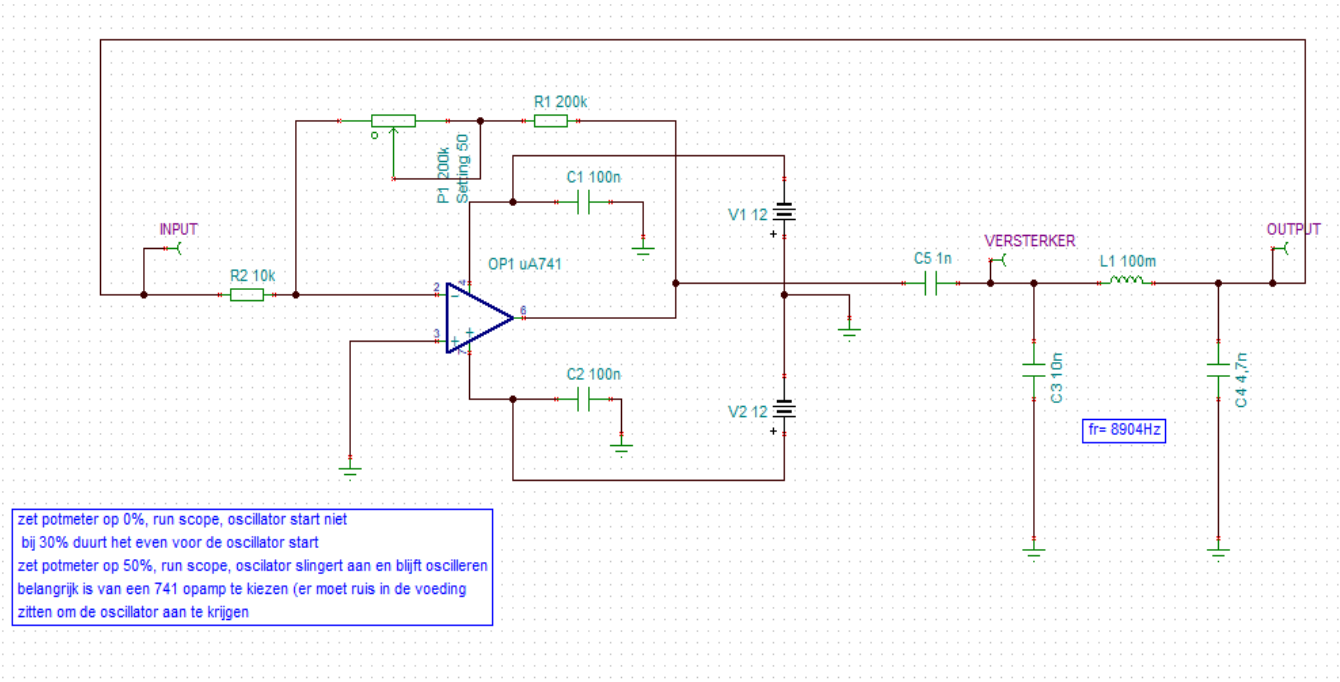
Hoe start dan de oscillator?

Eerst is A . B > 1 en zal uit de ruis van de voeding de frequentie oppikken die is ingesteld.

Stillaan wordt het signaal groter en gaat A . B steeds meer naar 1 toewerken.

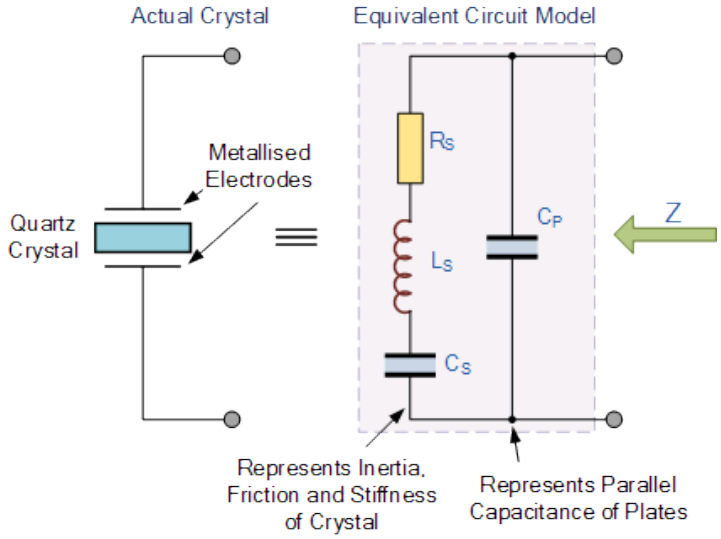
Op een bepaald moment is dan A . B = 1 en blijft de frequentie constant.

➔ **Taak: teken de volgende Colpitts oscillator in MS of TINA TI en run de simulator.**



Hoe werkt een X-tal oscillator?

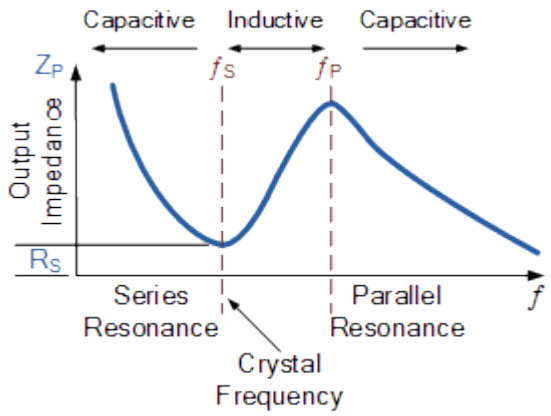
Als je het kwartskristal als een zeer dun schijfje van een spanningsbron voorziet, dan treed **het piëzo-elektrisch effect** op. Door een elektrische lading wordt een mechanische kracht geproduceerd op het kristal en verandert het van vorm. Andersom levert een mechanische kracht ook een elektrische lading op.
De grote en vorm, hoe het kristal is gesneden, bepaald de oscillatiefrequentie.



Typische quartz crystal in behuizing, en hoe je deze in de natuur kan terugvinden.

Een quartz kristal kan je ook tekenen als een elektrische kring van een condensator in parallel met een serieketen van weerstand, spoel en condensator.
De RLC keten stelt de mechanische trilling van het kristal voor, terwijl de Cp de elektrische aansluitingen zijn van het kristal met de buitenwereld.

Een kristal heeft 2 resonantiefrequenties: serie en parallelresonantie.



$$R = R \quad \text{and} \quad X_{LS} = 2\pi f L_S$$

$$X_{CS} = \frac{1}{2\pi f C_S} \quad \text{and} \quad X_{CP} = \frac{1}{2\pi f C_P}$$

$$Z_S = \sqrt{R_S^2 + \left(X_{LS} - X_{CS}\right)^2}$$

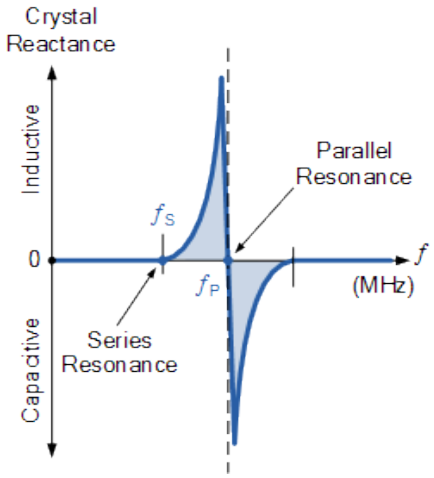
$$\therefore Z_P = \frac{Z_S \times X_{CP}}{Z_S + X_{CP}}$$

Bovenstaande curve geeft mooi aan hoe het kristal zich gedraagt. Bij een frequentie lager dan fs gedraagt het kristal zich als een capaciteit, net zoals wanneer de frequentie groter is als fp.
Tussen beide resonantie frequenties lijkt het kristal spoel gedrag te vertonen.
Op het moment van fp bereikt het kristal zijn maximum impedantie waarde Zp. Nu wordt er een **afgestemde LC tank** gecreëerd. Deze kunnen we verder in een LC oscillator inschakelen.

We moeten alleszins ons kristal afstemmen op 1 van de 2 frequenties, want op allebei tegelijk kan niet.
Afhankelijk van de eigenschappen van de schakeling rondom het het kristal gedraagt deze zich als een spoel, condensator, serieresonantiekring of parallelresonantiekring.

Dit kan voorgesteld worden door volgende cure.

Kristal reactantie t.o.v. de frequentie:



$$X_S = R^2 + \left(X_{LS} - X_{CS}\right)^2$$

$$X_{CP} = \frac{-1}{2\pi f C_P}$$

$$X_P = \frac{X_S \times X_{CP}}{X_S + X_{CP}}$$

$$f_S = \frac{1}{2\pi\sqrt{L_S C_S}}$$

Serie resonantie frequentie

$$f_P = \frac{1}{2\pi\sqrt{L_S\left(\frac{C_P C_S}{C_P + C_S}\right)}}$$

Parallelle resonantie frequentie

Oefening op berekenen van fs en fp:

Een kwartskristal heeft de volgende waarden: Rs = 6,4Ω , Cs = 0,09972pF en Ls = 2,546mH . Als de capaciteit over zijn terminal, Cp wordt gemeten bij 28,68 pF.

Bereken dan de fundamentele oscillatiefrequentie van het kristal en zijn secundaire resonantiefrequentie.

De serie resonantiefrequentie van de kristallenreeks, f_S

$$f_S = \frac{1}{2\pi\sqrt{L_S C_S}} = \frac{1}{2\pi\sqrt{2.546\text{mH} \times 0.09972\text{pF}}}$$

$$f_S = \frac{1}{2\pi\sqrt{0.002546 \times 99.72 \times 10^{-15}}} = 9.987\text{MHz}$$

De parallelle resonantiefrequentie van het kristal, f_P

$$f_P = \frac{1}{2\pi\sqrt{L_S\left(\frac{C_P C_S}{C_P + C_S}\right)}}$$

$$f_P = \frac{1}{2\pi\sqrt{2.546\text{mH}\left(\frac{28.68\text{pF} \times 0.09972\text{pF}}{28.68\text{pF} + 0.09972\text{pF}}\right)}}$$

$$f_P = 10{,}004{,}996\text{Hz or } 10.005\text{MHz}$$

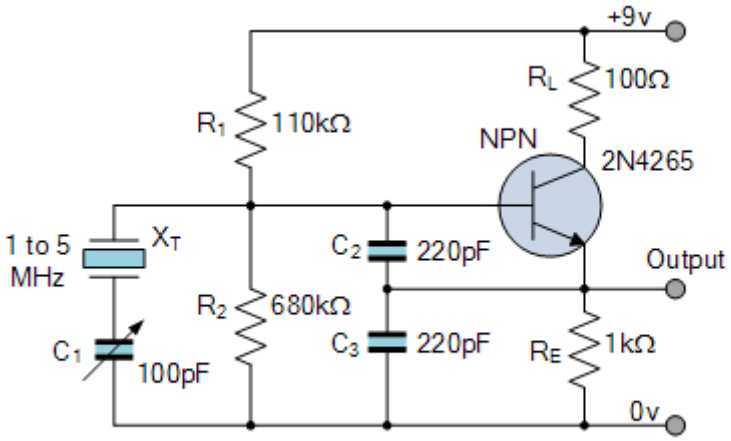
De **kwaliteitsfactor van het kristal** wordt gegeven door het berekenen van **de Q-factor**. Bij de seriere-sonantie toegepast op de vorige oefening is deze :

$$Q = \frac{X_L}{R} = \frac{2\pi f L}{R} = \frac{2\pi \times 9.987 \times 10^6 \times 0.002546}{6.4}$$

$$Q = 24966 \text{ or } 25{,}000$$

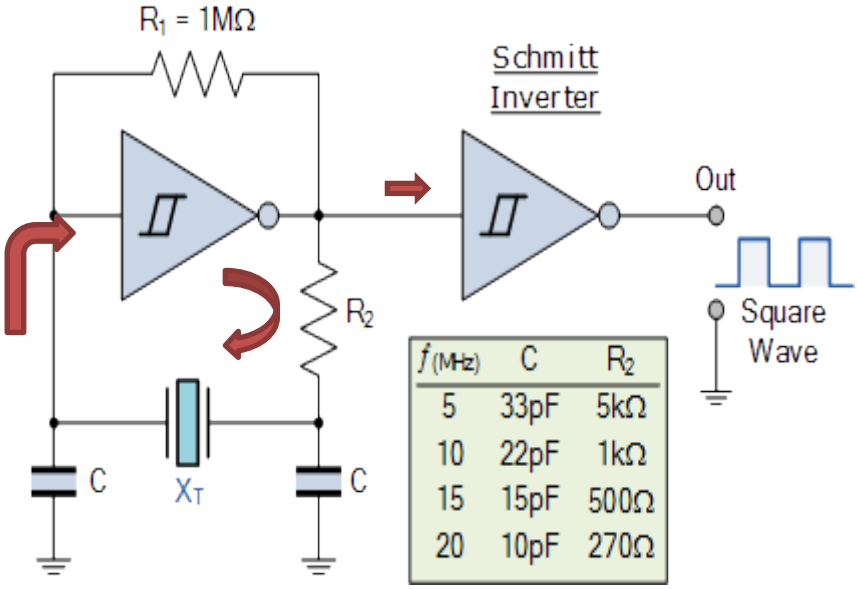
Dit is behoorlijk hoog t.o.v. een gewone LC oscillator waar we nog niet aan 1000 komen.
In een kwarts kristal oscillator wil de kristal altijd **oscilleren op de parallel frequentie** wanneer je deze aansluit op een spanningsbron.

Onderstaand schema is een voorbeeld van een **Colpitts kristal oscillator.**



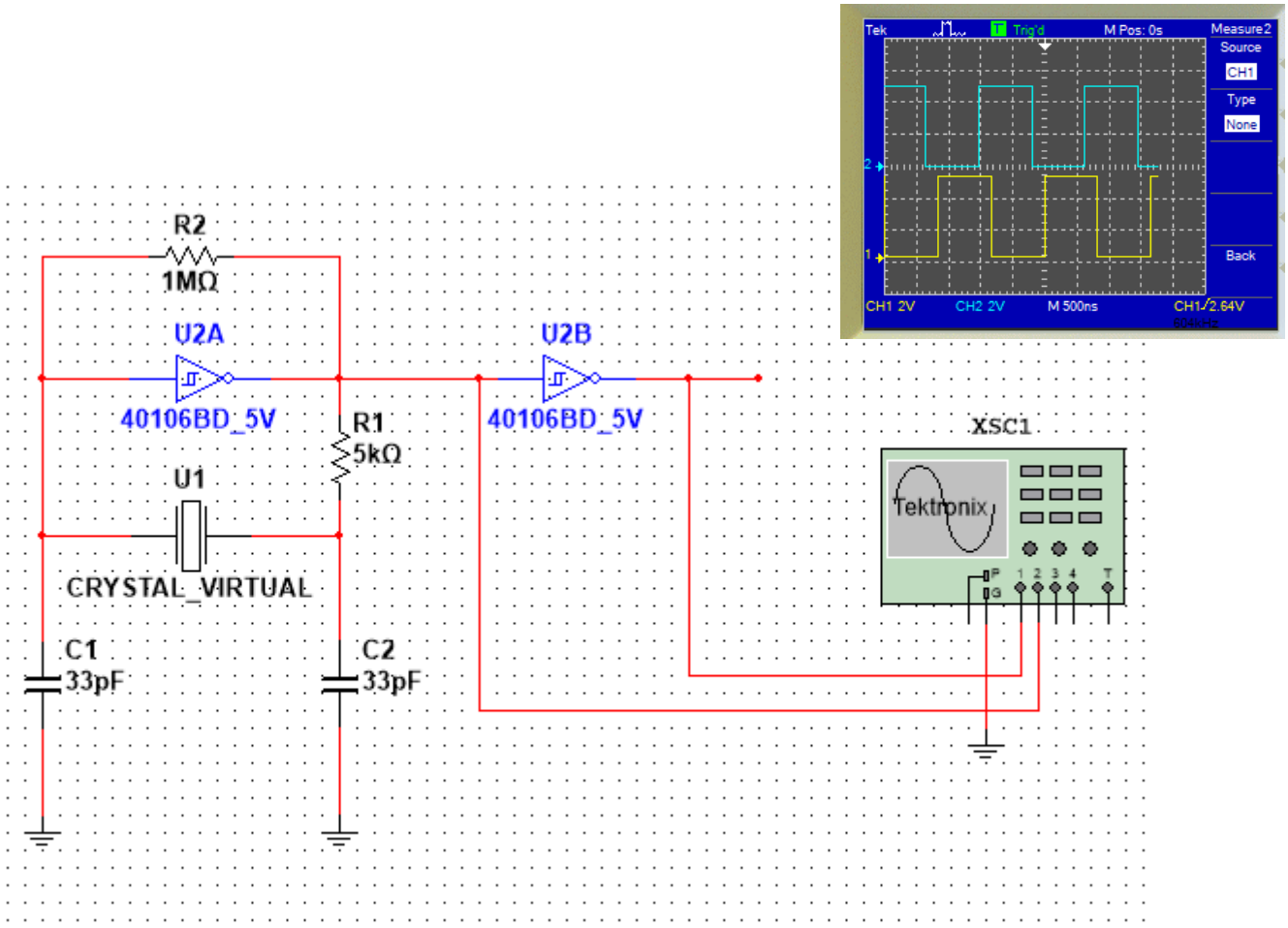
De oscillator is ontworpen rond een gemeenschappelijke collector versterker.
R1 en R2 regelen de DC waarde aan de basis, terwijl RE de uitgangsspanning instelt.

Dikwijls wordt in een microcontroller de volgende opstelling gebruikt van een **CMOS kristal oscillator**:



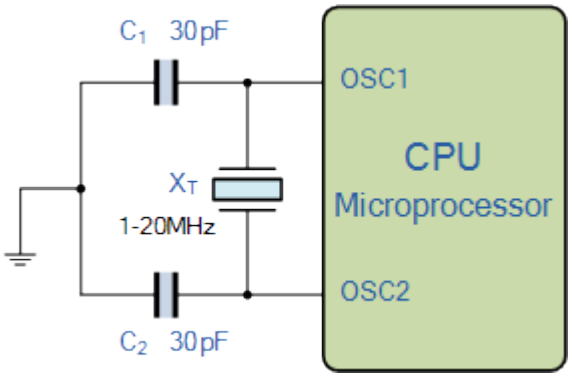
f (MHz)	C	R_2
5	33pF	5kΩ
10	22pF	1kΩ
15	15pF	500Ω
20	10pF	270Ω

Deze **Pierce** oscillator is zodanig ingesteld dat hij werkt bij de serie resonantie frequentie fs.
De 1M ohm weerstand zorgt ervoor dat de oscillator ingesteld in een gebied waar de versterking hoog is en lineair.
De rechtse Schmitt-trigger zet het sinus signaal mooi gebufferd voor een belasting om in een bloksignaal.
De **linkse inverter levert een 180 graden** faseverschuiving.
Het kristal circuit levert ook ongeveer **120 graden** verschuiving in combinatie met de **LC tank**.
Het R_2C circuit dat een **LDF** vormt zorgt nog eens voor **ongeveer 60 graden verschuiving.**
Je kan dit berekenen met de formule: $\varphi = - \arctan 2 \pi f RC$ (waarbij $R = R_2$).
Zodoende is de **rondgaande faseverschuiving 360 graden**, wat nodig is om een oscillator te bekomen.



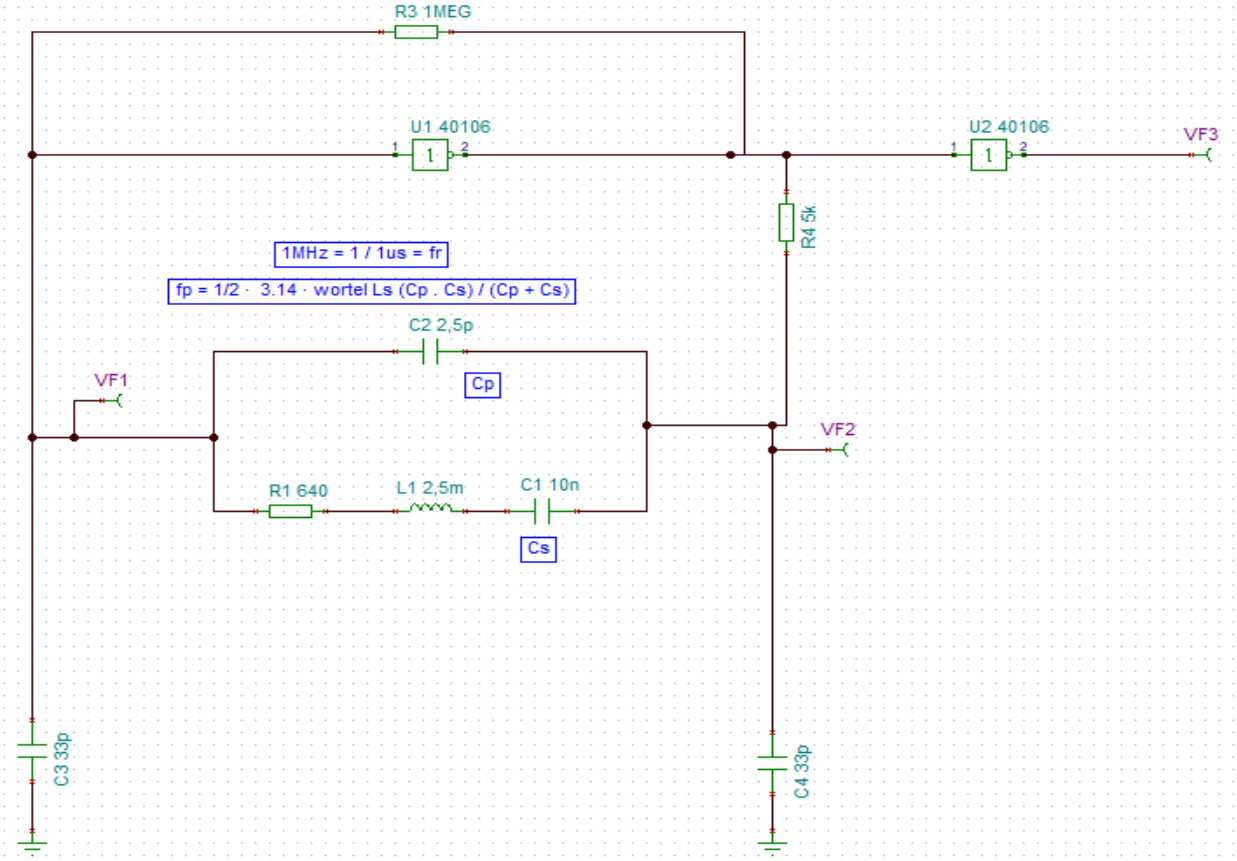
De microcontroller oscillator (**Pierce** oscillator):

Uiteindelijk gaan we een **kristal samen met 2 ceramische condensatoren** aansluiten op de oscillator-pinnen van de CPU. Tijdens het solderen van je eigen UNO gaan we dit er zeker over hebben.

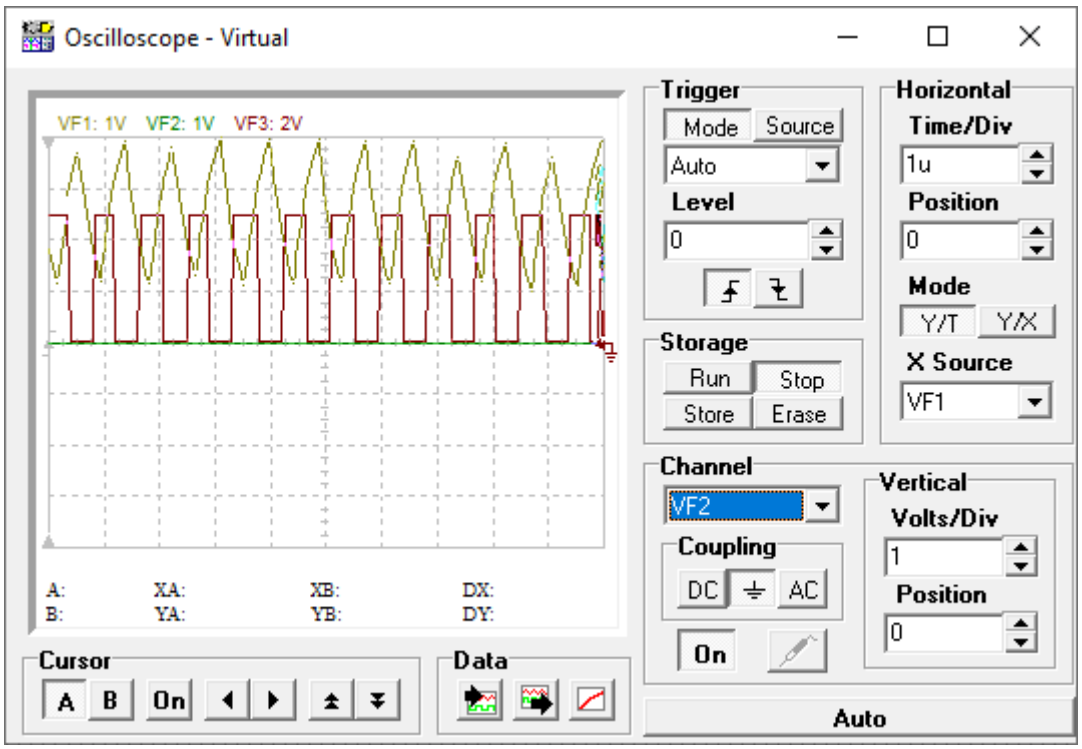


In de datasheet geeft de fabrikant aan wat de verschillende waardes moeten zijn om een bepaalde frequentie te bekomen van onze kwarts kristal oscillator. De rest van de schakeling zit in de chip.
Hier kan je opnieuw de voorgaande fs en fp formules op loslaten.

➔ **Taak: teken de volgende X-tal oscillator in MS of TINA TI en run de simulator.**



Dit is de TINA TI versie van de X-tal oscillator.



Planner : algemene kennis µC				
Onderwerp : Inleiding tot de µC				
Wat ?	**Hoe?**	**Werk-vorm ?**	**Doen ?** ☑	**Check!**
1. Zoek uit a.d.h.v. de foto's en tekst in de cursus hoe de **IC geschiedenis** in elkaar zit. **Welke componenten** werden hiervoor uitgevonden en wat zijn hun **voor/nadelen**.	Zie cursus "Microcontrollers Inleiding"	A,I		Test
2. Zoek uit a.d.h.v. de foto's en tekst in de cursus hoe de **µC history** in elkaar zit. Welke chips volgden elkaar op en wat zijn telkens de verschillen. Welke **belangrijke stappen** werden er gezet in deze geschiedenis om te komen tot de **µC van vandaag**?		A,I		Test
3. Zoek uit wat het **verschil** is tussen een µprocessor, µcontroller en µcomputer		A,I		test
4. Vertel kort wat **de wet van Moore** inhoud. Kan je dit ook tekenen op een grafiek?		A		test
5. Als we spreken over een **8-bit processor**, wat bedoelen we daar dan mee?		A		Test
6. Teken het **blokschema** van een **µcomputersysteem** en benoem alle onderdelen. Duid het nut van elke blok.	Zie PPP	A,		Test
7. Verbind alle blokken van het blokschema uit punt 5 en benoem de **bussen**. Geef ook duidelijk met pijlen aan in welke **richting** de data gaat.		A,		Test
8. Weet jij wat een **multicore** processor is? Wat is een **thread**?	Internet	Co		test
9. Welke bus van jouw blokschema is		A,		test

uni/bidirectioneel? Wat betekent dit? Waarom is dit zo denk je. Geef bij elke bus een praktische toepassing.				
10. **Vergelijk** de tekening van de PC architecture met deze van de µC architecture.Kan jij de **gelijkenissen** terugvinden t.o.v. het µC blokschema?	PC op houten plank bestuderen, 2 tekeningen in cursus	A, I 		Praktisch
11. Hoe zit de **busstructuur** in elkaar bij een uc? Hoe komt het dat we **meerdere blokken (geheugen, I/O) kunnen aansluiten op 1** en dezelfde bus? Welke techniek hebben we hiervoor nodig?		A, 		Test
12. Taak: zoek de werking uit van de 74LS244 en 74LS245. Zoek beide op in een datasheet en bekijk de verschillen.	Internet	A		Als verslag deze info en verschillen afgeven (enkel de interne werking)
13. Hoe zit de **CPU** van de uc in elkaar? Teken het **blokschema**, benoem alle **blokken en leg uit** waarvoor elke blok dient.	Zie PPP	A		Test
14. Pas de **werking** van het ALU toe op een toepassing, vb: A AND B. Teken ook de samenwerking tussen CPU en geheugen.		I		Test
15. Wat is een **opcode, operand?** Geef een toepassing van deze termen bij een assembler code commando naar keuze.		A,I 		Test
16. Wat is een **accumulator**? Wat is het nut/werking hiervan in een CPU? Pas deze toe op onze FFrobot uc		A 		Test
17. Wat is een **statusregister**? Geef hier een praktisch voorbeeld van.		A,I 		Test
18. Wat is het nut van een **programma teller** in een		A 		Test

CPU? Hoe groot is deze bij de 18F4455?				
19. Wat is een **instructieregister?** Wat is het verschil tussen een RISC en CISC instructieset?		A		Test
20. Wat is het nut van een **instructiedecoder** en een **control unit** in de CPU?		A		Test
21. Hoe werkt de **stack** en stack pointer bij een CPU? Hoe groot is deze bij een 18F4455? Waar kan je een stack makkelijk mee vergelijken in het dagelijks leven?		A,I		Test
22. **GPR en SFR**? Wat zijn dat voor registers?		A,		TEST
23. Wat is het nut van een **timing en klok** bij een CPU?		A		TEST
24. Maak gebruik van de **simulator** en laat een arduino programma -> hex file runnen in deze simulator. Volg in vertraging de verschillende stappen dewelke de CPU doorloopt. Kan jij alle registers terugvinden die we besproken hebben?	Oshon AVR simulator.	A,I		praktisch
25. Onderzoek het verschil tussen **een harvard en Von Neumanngeheugen structuur**. Maak van elk een tekening en leg de verschillen uit.		A,		Test
26. Wat is het verschil tussen de harvard en **modified harvard structuur**?		A,I		Test
27. Zoek op internet info over de volgende geheugens: **FLASH, EEPROM en SRAM/DRAM**. Los de vragen in de taak op.	Taak zie cursus, internet	A		Taak als kort verslag afgeven (niet enkel copy paste, beantwoord kort de vragen)
28. Leg a.d.h.v. een tekening uit hoe de **Von Neumanncyclus** werkt. Pas		A,I		Test

deze toe op een commando van de PIC18F4455				
29. Leg a.d.h.v. een tekening uit hoe **de gegevensstroom** in een CPU werkt van CPU naar geheugen en omgekeerd. Leg alle tussenstappen uit.		A, 		Test
30. Leg a.d.h.v. een tekening uit hoe 1 pin werkt van de **GPIO.** Hoe is de gegevens stroom bij een INPUT en OUTPUT?		A, 		Test
31. Zorg dat je weet hoe een **LC tank** en **X-tal** werkt, wat de **voorwaardes** van een oscillator zijn, hoe je een **Collpits oscillator en een Pierce oscillator** kan bouwen. Bespreek de werking van de verschillende onderdelen.		A, 		Test
32. **Simuleer** de Collpits en Pierce oscillator in Tina TI / MS. Meet na of we frequentie klopt met de ingestelde en berekende componenten.	Tina TI / Multisim	A, 		Verslag inleveren