# Stichting NIOC en de NIOC kennisbank

Stichting NIOC (www.nioc.nl) stelt zich conform zijn statuten tot doel: het realiseren van congressen over informatica onderwijs en voorts al hetgeen met een en ander rechtstreeks of zijdelings verband houdt of daartoe bevorderlijk kan zijn, alles in de ruimste zin des woords.

De stichting NIOC neemt de archivering van de resultaten van de congressen voor zijn rekening. De website www.nioc.nl ontsluit onder "Eerdere congressen" de gearchiveerde websites van eerdere congressen. De vele afzonderlijke congresbijdragen zijn opgenomen in een kennisbank die via dezelfde website onder "NIOC kennisbank" ontsloten wordt.

Op dit moment bevat de NIOC kennisbank alle bijdragen, incl. die van het laatste congres (NIOC2023, gehouden op donderdag 30 maart 2023 jl. en georganiseerd door NHL Stenden Hogeschool). Bij elkaar bijna 1500 bijdragen!

We roepen je op, na het lezen van het document dat door jou is gedownload, de auteur(s) feedback te geven. Dit kan door je te registreren als gebruiker van de NIOC kennisbank. Na registratie krijg je bericht hoe in te loggen op de NIOC kennisbank.

Het eerstvolgende NIOC vindt plaats op donderdag 27 maart 2025 in Zwolle en wordt dan georganiseerd door Hogeschool Windesheim. Kijk op www.nioc2025.nl voor meer informatie.

Wil je op de hoogte blijven van de ontwikkeling rond Stichting NIOC en de NIOC kennisbank, schrijf je dan in op de nieuwsbrief via

www.nioc.nl/nioc-kennisbank/aanmelden nieuwsbrief

Reacties over de NIOC kennisbank en de inhoud daarvan kun je richten aan de beheerder:

R. Smedinga kennisbank@nioc.nl.

Vermeld bij reacties jouw naam en telefoonnummer voor nader contact.

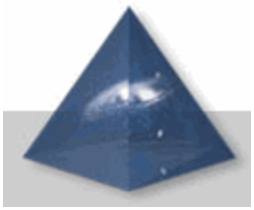
DE UNIVERSITEIT VAN AMSTERDAM UvA

# SIM-PL

## Auteursomgeving voor Digitale Componenten

Ben Bruidegom

Wouter Koolen-Wijkstra



AMSTEL INSTITUUT
FACULTEIT DER NATUURWETENSCHAPPEN,
WISKUNDE EN INFORMATICA
UNIVERSITEIT VAN AMSTERDAM

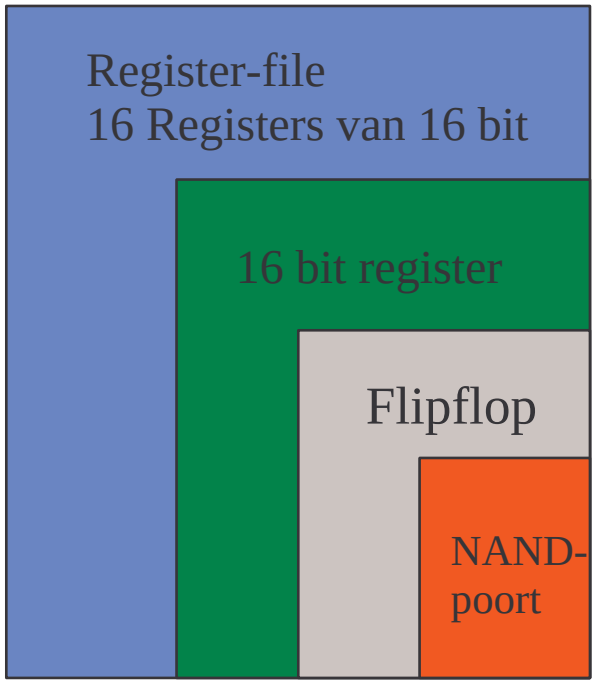
# Inhoud

- Digitale Componenten
- Demo Editor
- Simulator
- Harvard Processor
- Toepassing
- Nabeschouwing

# Objecten: Digitale Componenten

## Hardware met alle in/uitgangen 0 of 1

- Complexiteit
  - Van poort tot processor
  - Hiërarchische compositie
- Timing
  - Synchroon v.s. asynchroon
- Functionaliteit
  - Combinatorisch v.s. geheugenfunctie



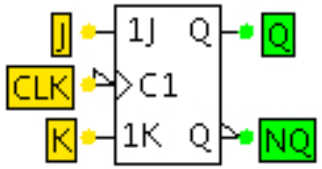
# Ontwerp: Componenten Schakelingen

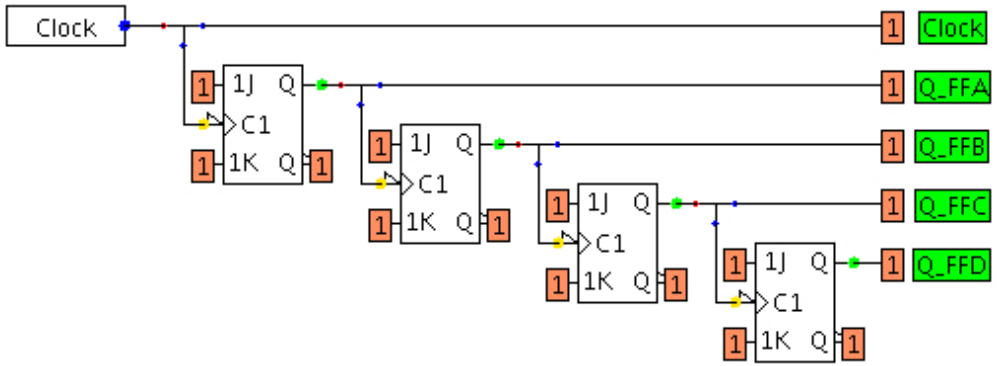
Simple (basis)

- Basisfiguren
- Input/Output
- Programmering
- Delay

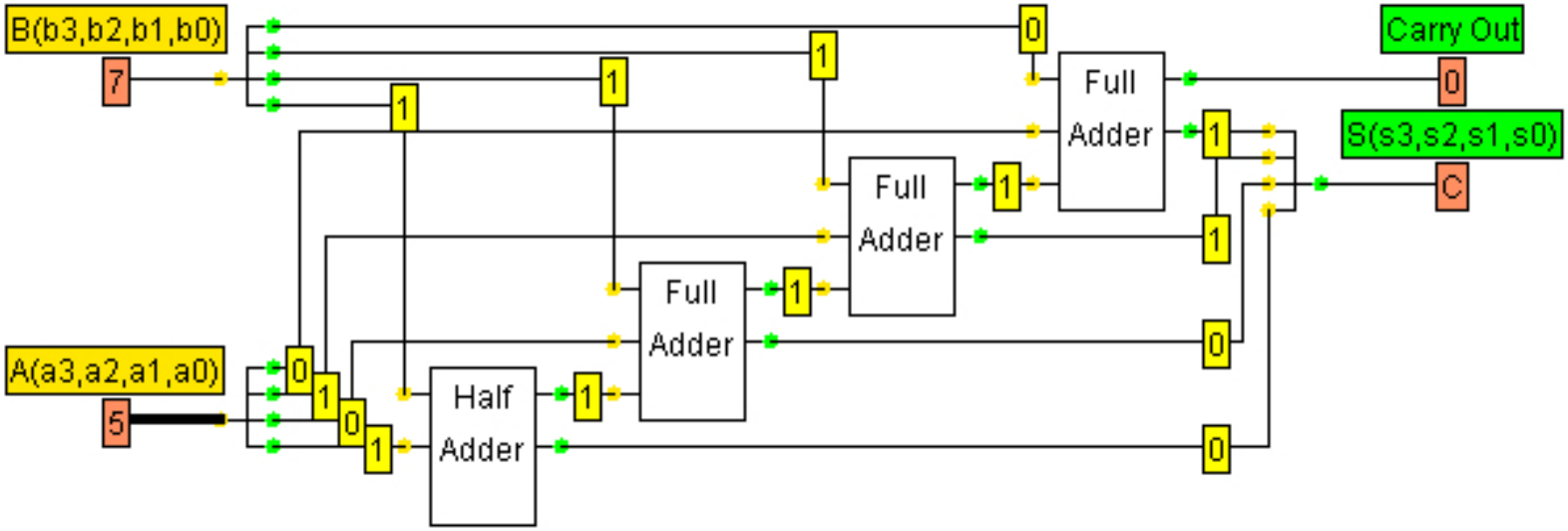
Complex (hiërarchie)

- Subcomponenten
- Input/Output
- Connecties



# Voorbeeld: 4 bits opteller



# Demo SIM-PL Editor

- Simple Component
  - Bouwen AND poort
- Complex Component
  - XOR poort bouwen uit poorten:
    - NOT
    - AND
    - OR

$$a \otimes b = (a \cdot \bar{b}) + (\bar{b} \cdot a)$$

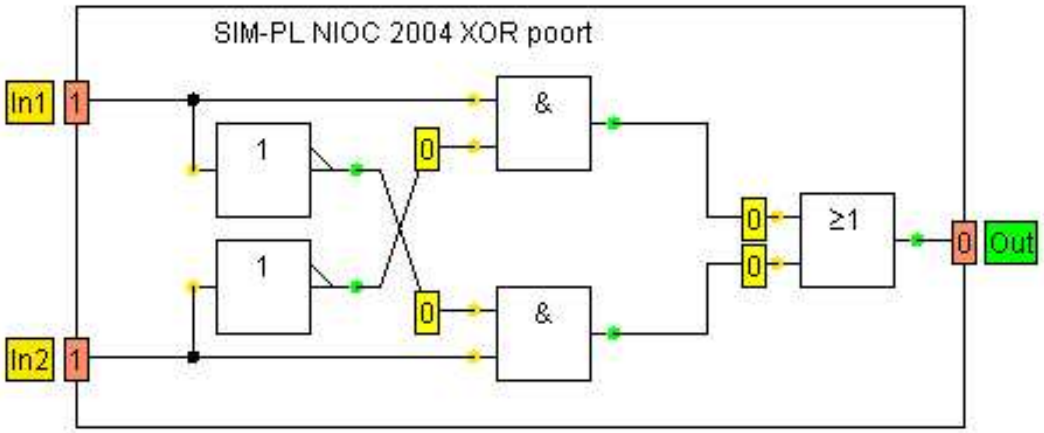
# Interne programmeertaal: nBit

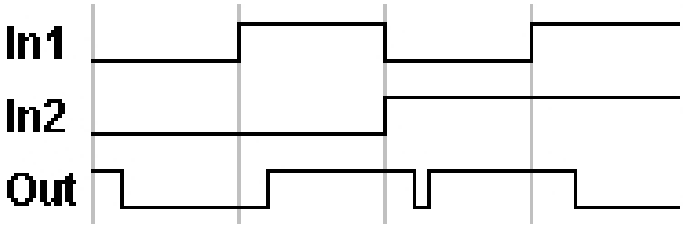
- Syntax
  - C/C++/Java
- Basisdatatype
  - getallen van ***n*** bits
- Voorbeeld: ALU
  - A, B: 55 bit ingang
  - opcode: 2 bit ingang
  - R: 55 bit uitgang
  - {

```
switch (opcode)
{
    case  0:  R = A + B;  break;
    case  1:  R = A – B;  break;
    case  2:  R = A |  B;  break;
    case  3:  R = A & B;  break;
}
}
```

# Samenvatting: Discrete Event Simulation

- Beginconfiguratie
  - Gebruiker
  - Compilers
- Signaalveranderingen als boodschappen
- Propagation Delay



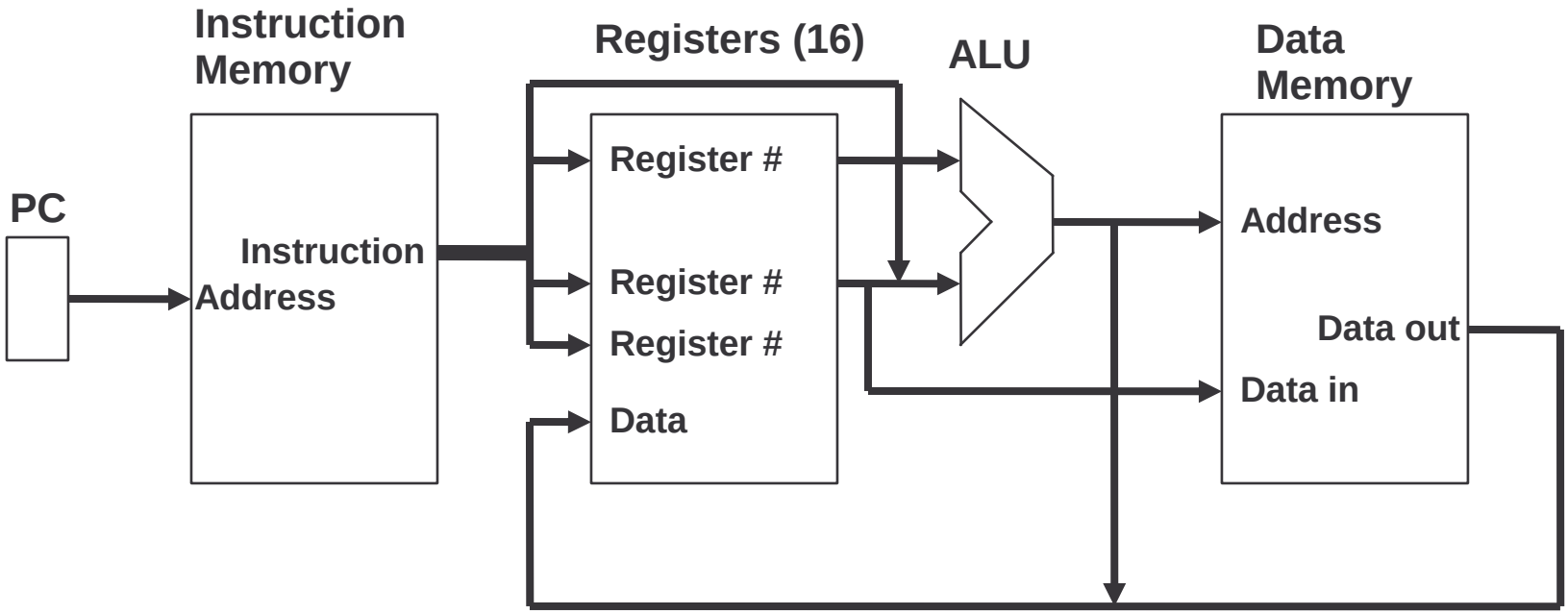




B. Bruidegom en W. Koolen-Wijkstra

# Een eenvoudige Harvard processor

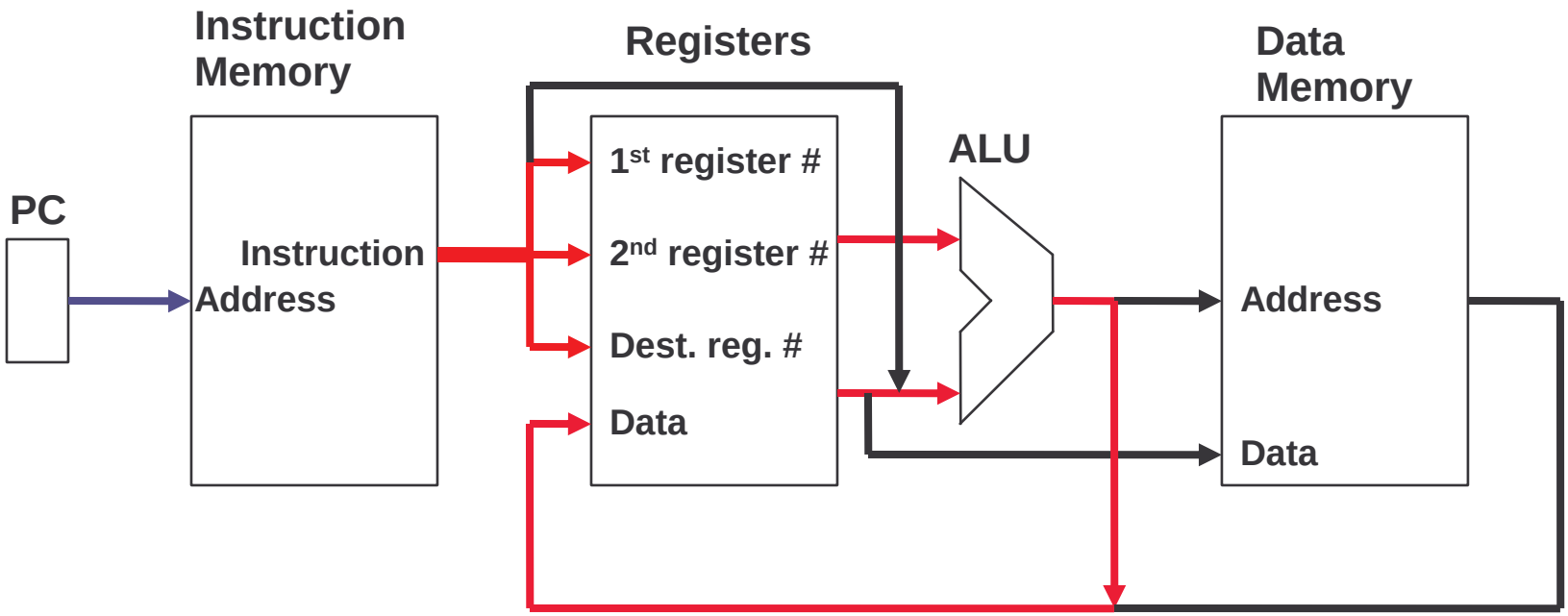
- Simpele architectuur om de werking van een computer tot in detail te kunnen begrijpen.
- Bekeken vanuit de “hardware”
- Ook geschikt voor VWO scholieren

# Simplified View of a Harvard Architecture*




* Bij een Harvard architectuur is het geheugen gescheiden in twee delen: instructiegeheugen en datageheugen

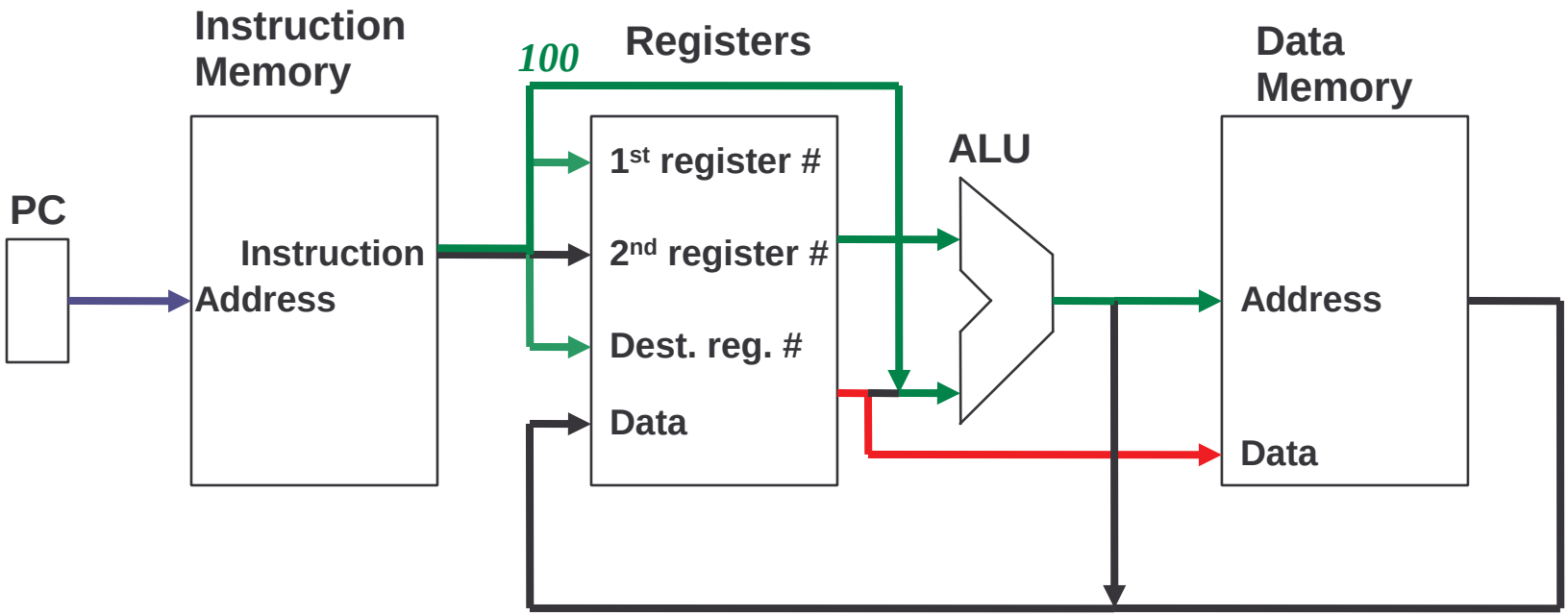
# Voorbeeld van een instructie: ADD




*Assembly Language*

***ADD $r0, $r1, $r2***
***$r0 = $r1 + $r2***

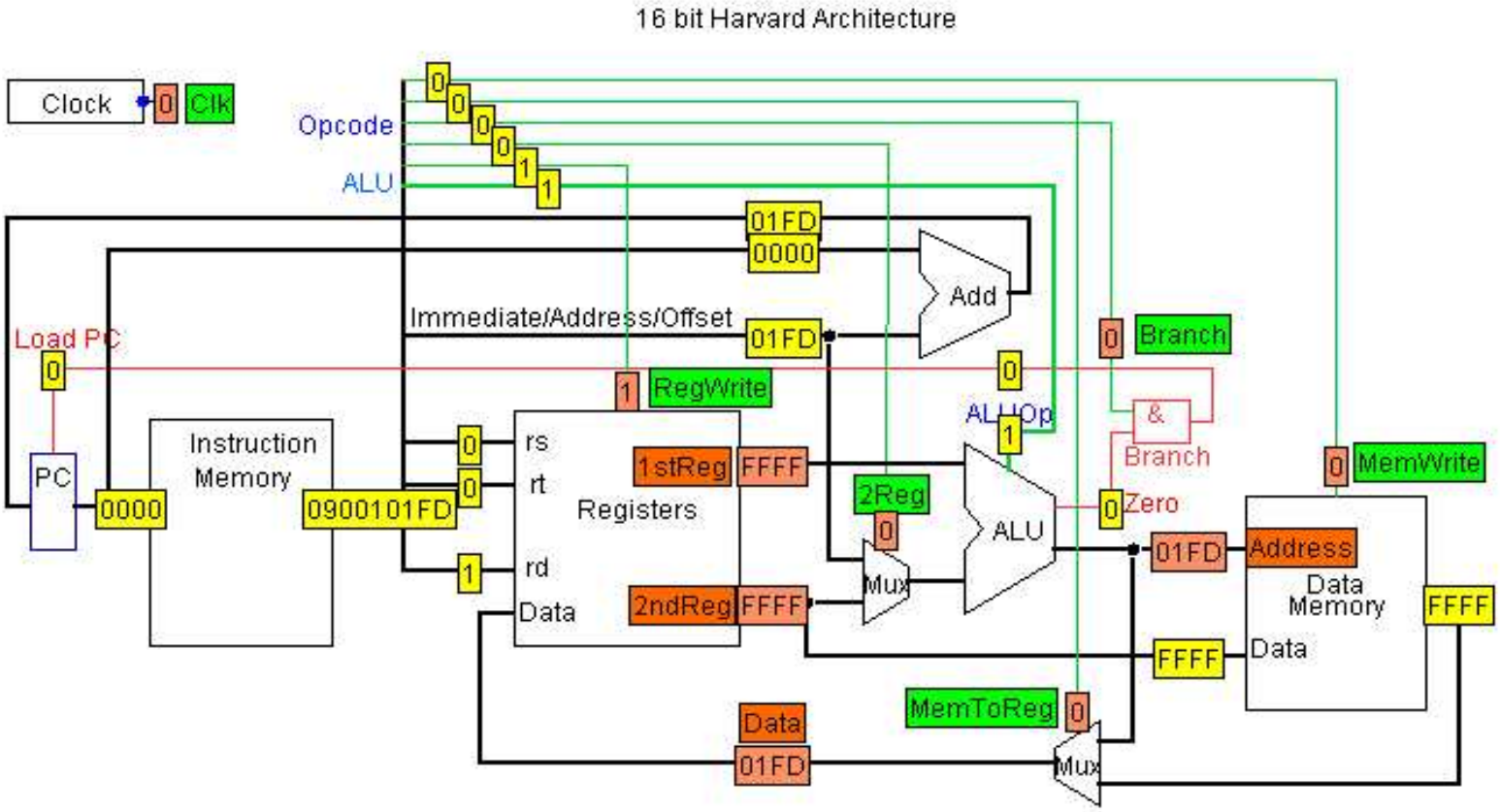
# De store-instructie SW: Register → Memory




***SW $r0, 100($r1)***

***Memory[$r1 + 100] = $r0***

# De status van de Harvard machine na het uitvoeren van de instructie Load Immediate 0x01FD



# Waar is/wordt SIM-PL toegepast?

- Cursus Architectuur en Computerorganisatie voor 1e jaars Informaticastudenten UvA
- Cursus Digitale techniek/Architectuur voor voor 1e jaars AI-studenten UvA
- VWO-scholieren
  - Beta-festival
  - Diverse schoolklassen
- Mastercourse voor VWO-docenten

# Verbeteringen, uitbreidingen

- Gebruikersvriendelijker Editor
- Implementatie Micro-programmeren
- C Compiler om aansluiting te maken met Operating Systems
- Toevoegen Componenten en Architecturen
- Geschikt maken voor het middelbaar onderwijs

# Visie voor realisatie

- Voor HBO en WO:
  - Partnerschap in Digitale Universiteit-verband
    - Universiteit van Amsterdam
    - Vrije Universiteit
    - Open Universiteit Nederland
    - Universiteit Twente
    - Hogeschool van Amsterdam
    - Hogeschool Rotterdam
    - Hogeschool van Utrecht
    - Hogeschool INHOLLAND
    - Fontys Hogescholen
    - Saxion Hogescholen
- Voor MO?

# Waarom SIM-PL?

- Geeft inzicht in de werking van ingewikkelde digitale schakelingen op alle niveau's
- Dicht het "gat" tussen Digitale techniek (poorten, flipflops etc.) en complete pipeline processoren
- Auteursomgeving voor docenten, studenten/scholieren
- Beschikbaar onder GPL licentie (Free Software) http://staff.science.uva.nl/~benb/SIM-PL