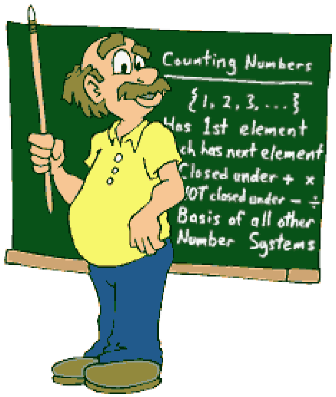
AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

MSYS

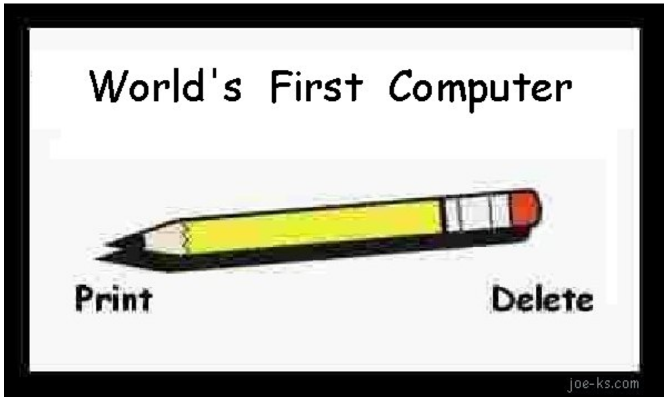
Microcontroller Systems

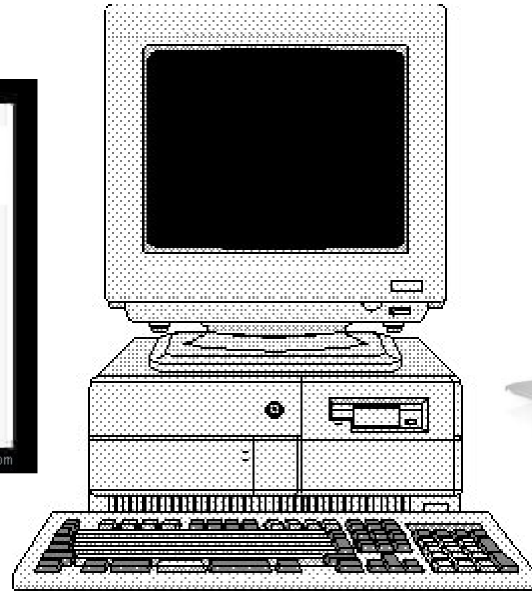
Lektion 2: Introduktion til computere



Version: 8-8-2017, Henning Hargaard

Hvad er en computer ?



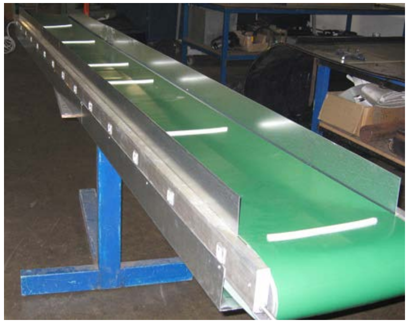


- Grundlæggende er en ”computer” blot en elektronisk enhed, der er i stand til at foretage beregninger.

- En PC er en ”general purpose” computer.
Mange apparater indeholder en ”special purpose” computer (typisk en microcontroller).

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Microcontroller = ”Lille computer”



AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Hvad er computer-arkitektur ?

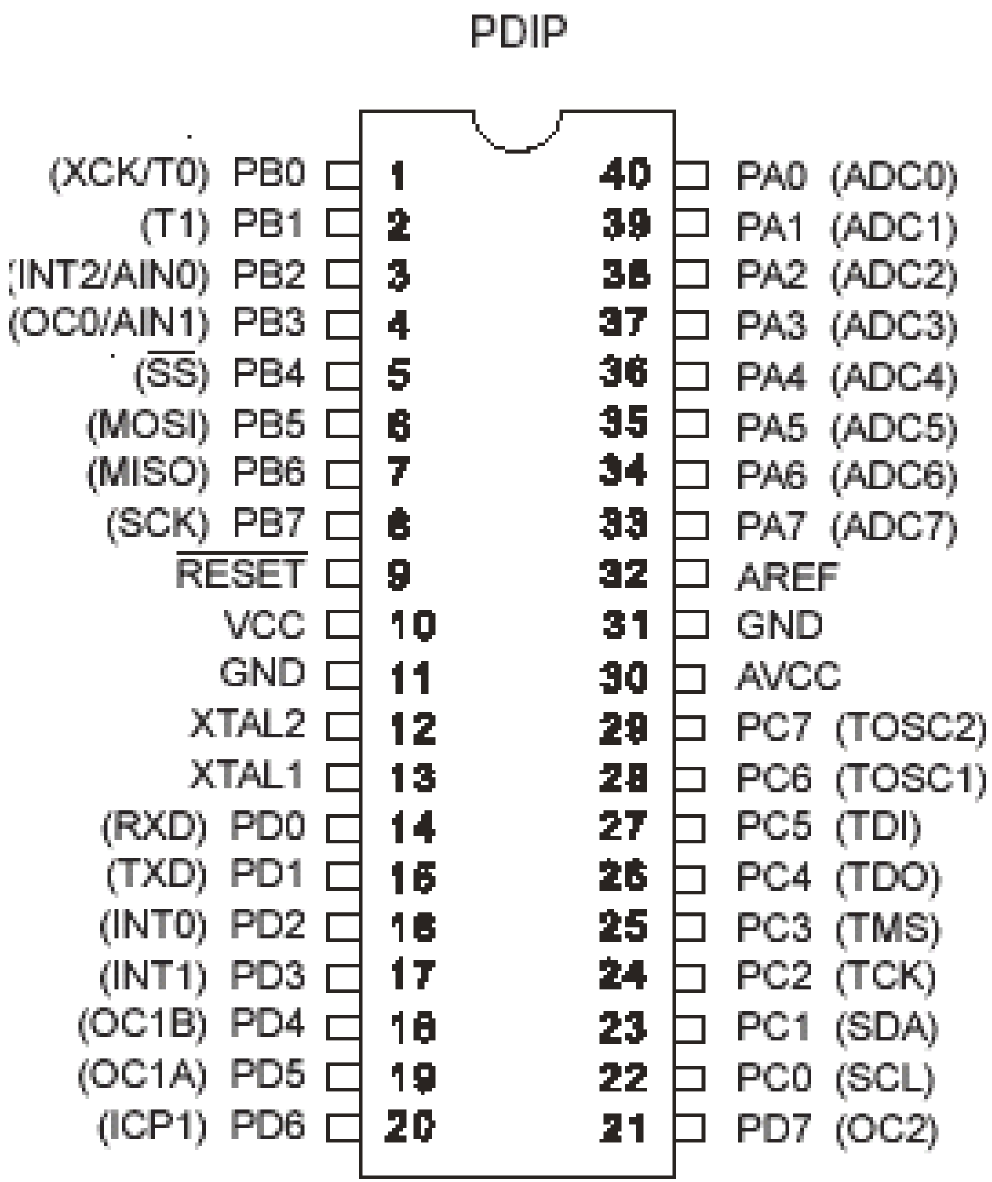
Måden, hvorpå en computer internt er opbygget (HW).

Hvad skal en computer indeholde ?

- Regne-enhed (ALU) og status-register.
- Hukommelse (program og data).
- Program-tæller.
- Instruktions-dekoder.
- Input og output – enheder.

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Atmel AVR: Single chip microcontroller

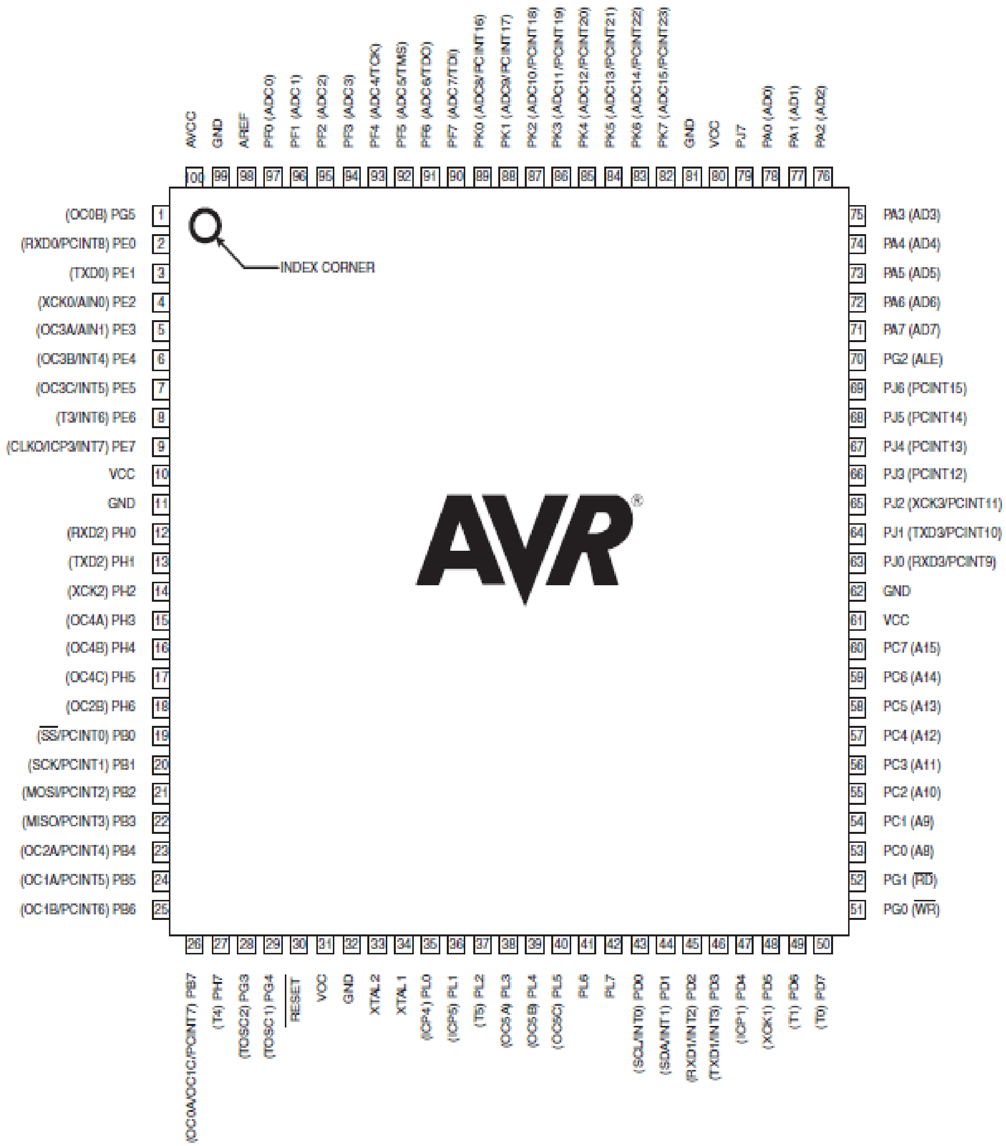


Pinout for Mega32

(den fra lærebogen)

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Atmel AVR: Single chip microcontroller

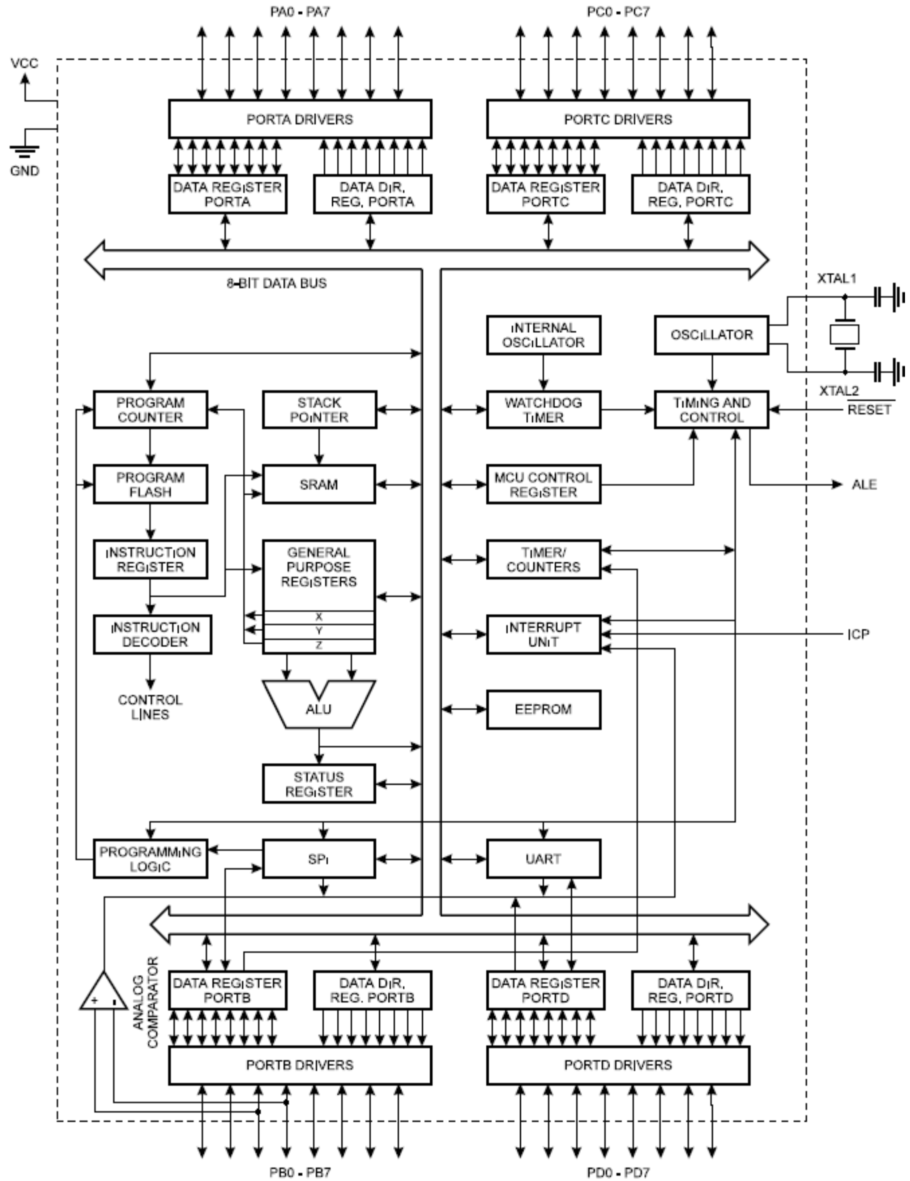


Pinout for Mega2560

(bruges i LAB-øvelserne)

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

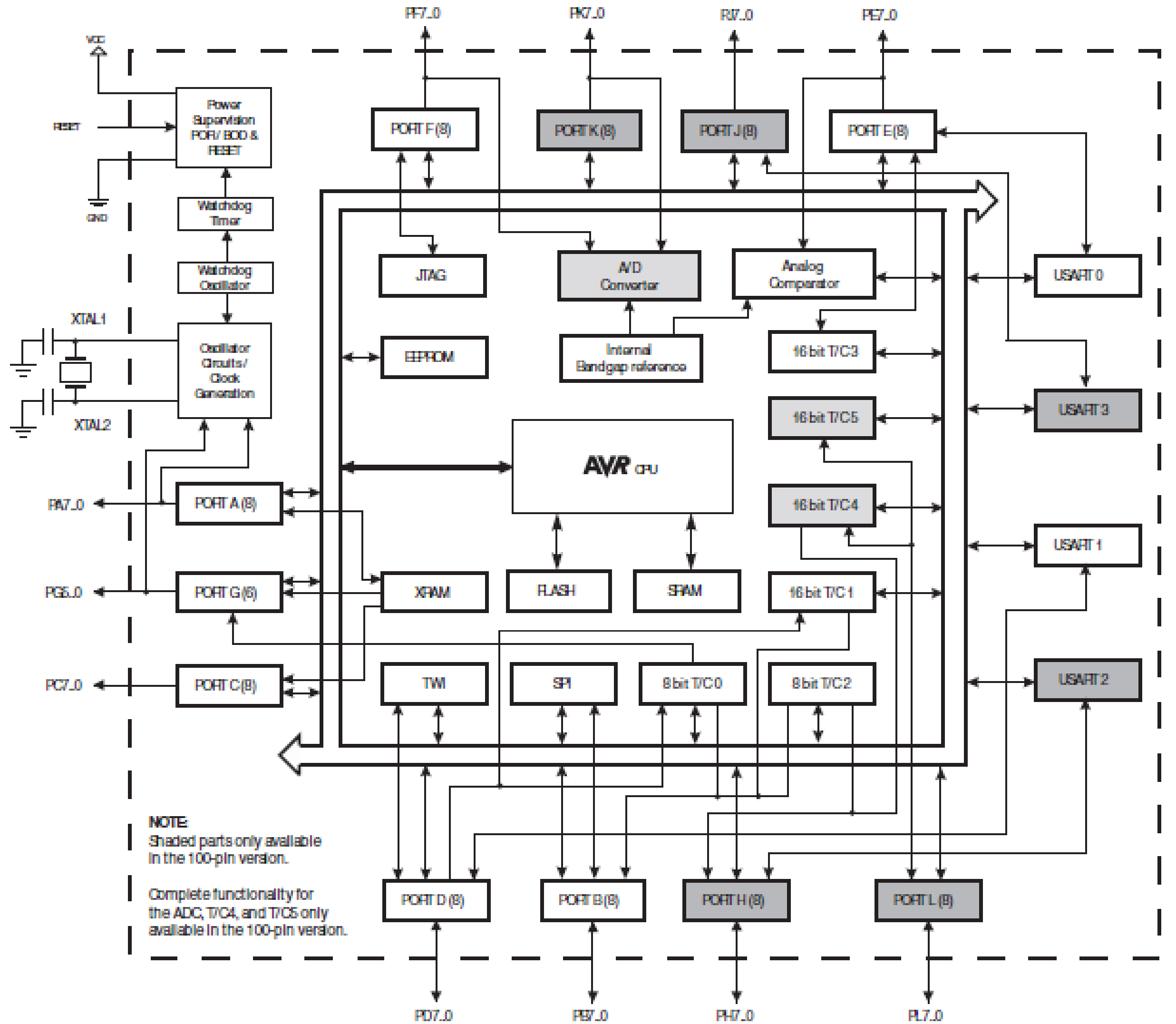
Mega32: Intern Arkitektur



AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Mega2560: Intern Arkitektur

Figure 2-1. Block Diagram



AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Program (SW)

- Den sekvens af instruktioner, som beskriver, hvad en computer skal udføre (funktionaliteten).
- Programmet skal overføres (”downloades” eller ”installeres”) på computeren, før det kan afvikles.
- En PC er ”general purpose”, og udfører typisk mange forskellige programmer til forskellige tidspunkter (afhængig af formålet).
- En microcontroller vil typisk styre et apparat med en bestemt funktionalitet (”special purpose”), og programmet installeres derfor ofte en gang for alle.

Maskinkode

- Grundlæggende kan en microcontroller eller microprocessor kun forstå og afvikle maskinkoder, der er binære talkoder.

- Koderne afvikles sekventielt (”efter tur”), og styrer på primitiv vis computeres interne hardware.

- Man anvender ofte et symbolsk maskinkode – sprog (ASSEMBLY language), når programmet skrives.

- Symbols maskinkode oversættes til maskinkode med en assembler (et program, der kører på en PC).

Eksempel:

```
LDS  R26,_led_status
CPI  R26,LOW(0xFF)
BRNE _0x4
```

Højniveau - sprog

• Det er besværligt, og kræver dyb indsigt i computerens interne arkitektur at kunne skrive assembly-kode.

• Langt overvejende anvendes derfor programmering på et højere abstraktions-niveau (høj-niveau sprog).

• Eksempler er: Pascal, Basic, Java, C++, C, C#.

• Al højniveau-sprog skal oversættes til maskinkoder for at kunne forstås og afvikles af computeren.
Hertil anvendes et program, der kaldes en COMPILER.

Eksempel på højnivieau -sprog :

```
if ( der_er_trykket_paa_knappen() )
   start_motoren();
else
   stop_motoren();
```

C kontra Assembly

- C :

```
if (led_status==0xff)
```

- Assembly :

```
LDS  R26,_led_status
CPI  R26,LOW(0xFF)
BRNE _0x4
LDI  R30,LOW(254)
STS  _led_status,R30
```

Fordele / ulemper ?

Et C program

- C er ”forgængeren” for C++ (simplere end C++).
  C anvendes oftest til programmering af microcontrollers.

Eksempel:

```
#include <avr/io.h>

int main()
{
unsigned char i = 0;
  DDRA = 0xFF; //port A as output
  DDRB = 0xFF; //port B as output
  DDRC = 0xFF; //port C as output
  PORTA = 0xAA;
  while (1)
  {
    PORTC = PORTC ^ 0x01; //toggle PORTC.0
    PORTB = i;
    i++;
  }
  return 0;
}
```

Compiler

• Compileren ”oversætter” vores C-code til en fil, der indeholder alle maskinkoderne for programmet.

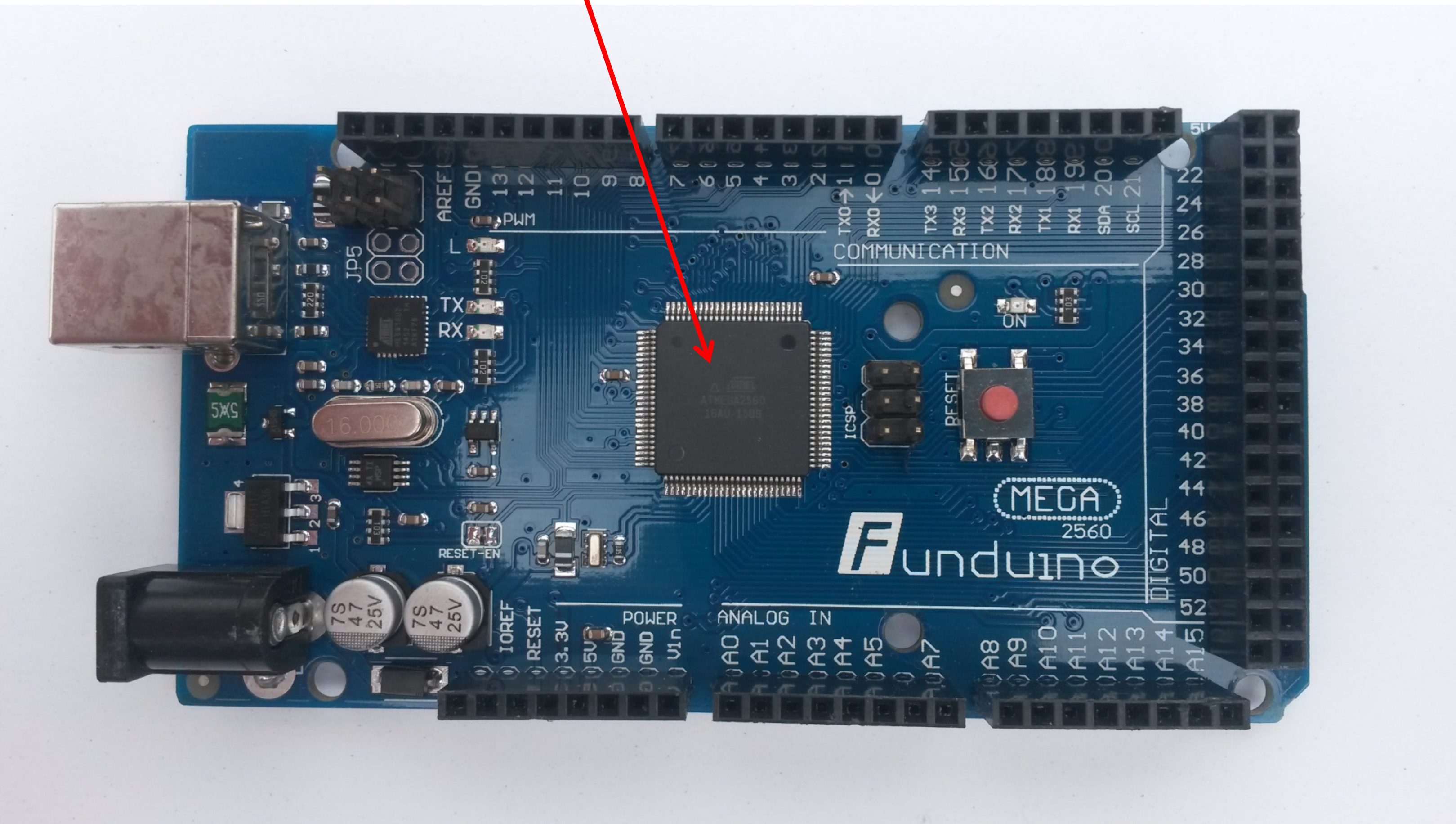
<u>Forskellige computere har forskellige maskinkoder</u>.

• Compileren selv er et program, der typisk kører på en PC.

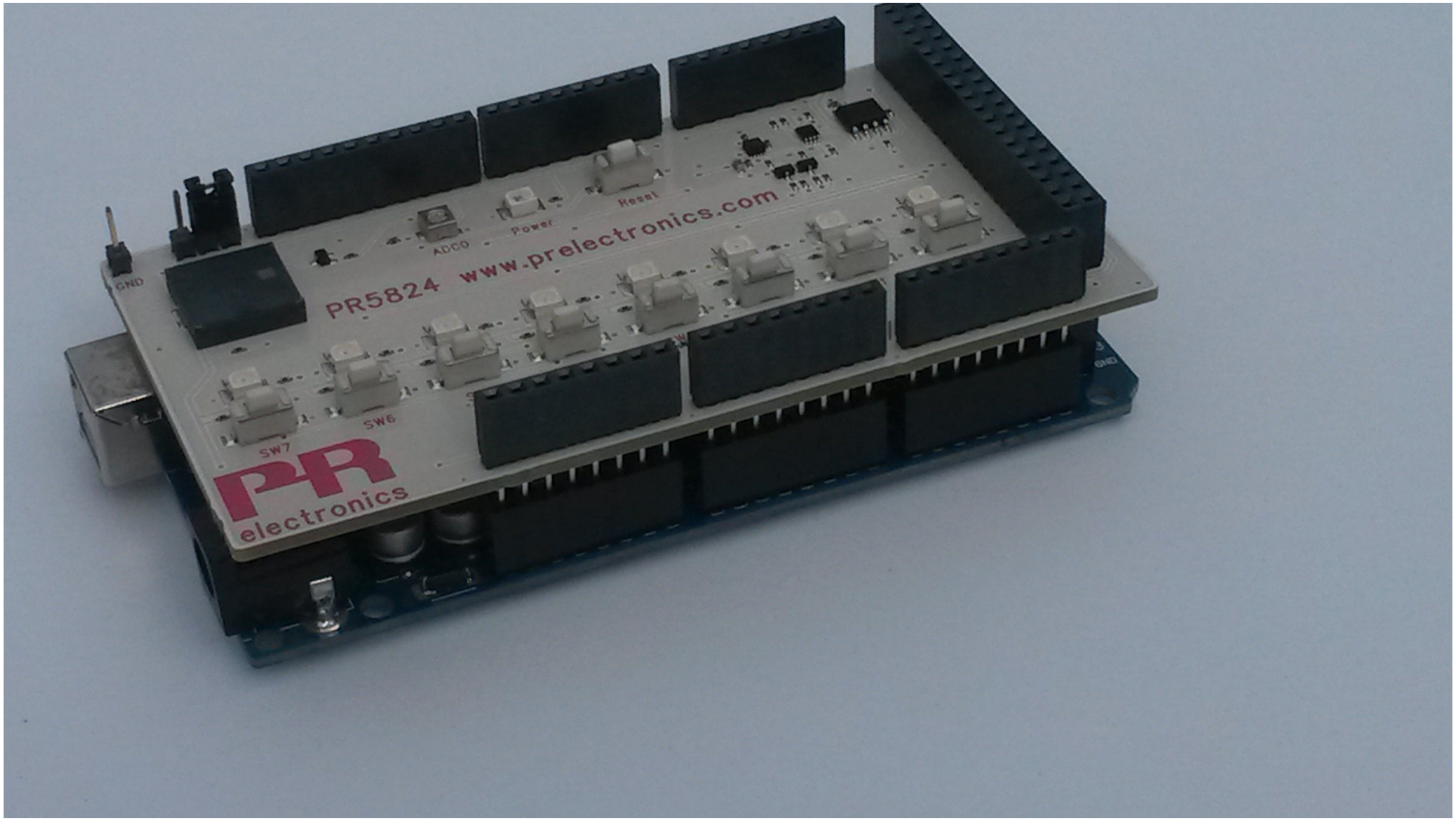
AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

"Arduino/Funduino Mega2560"

Microcontroller = Mega2560

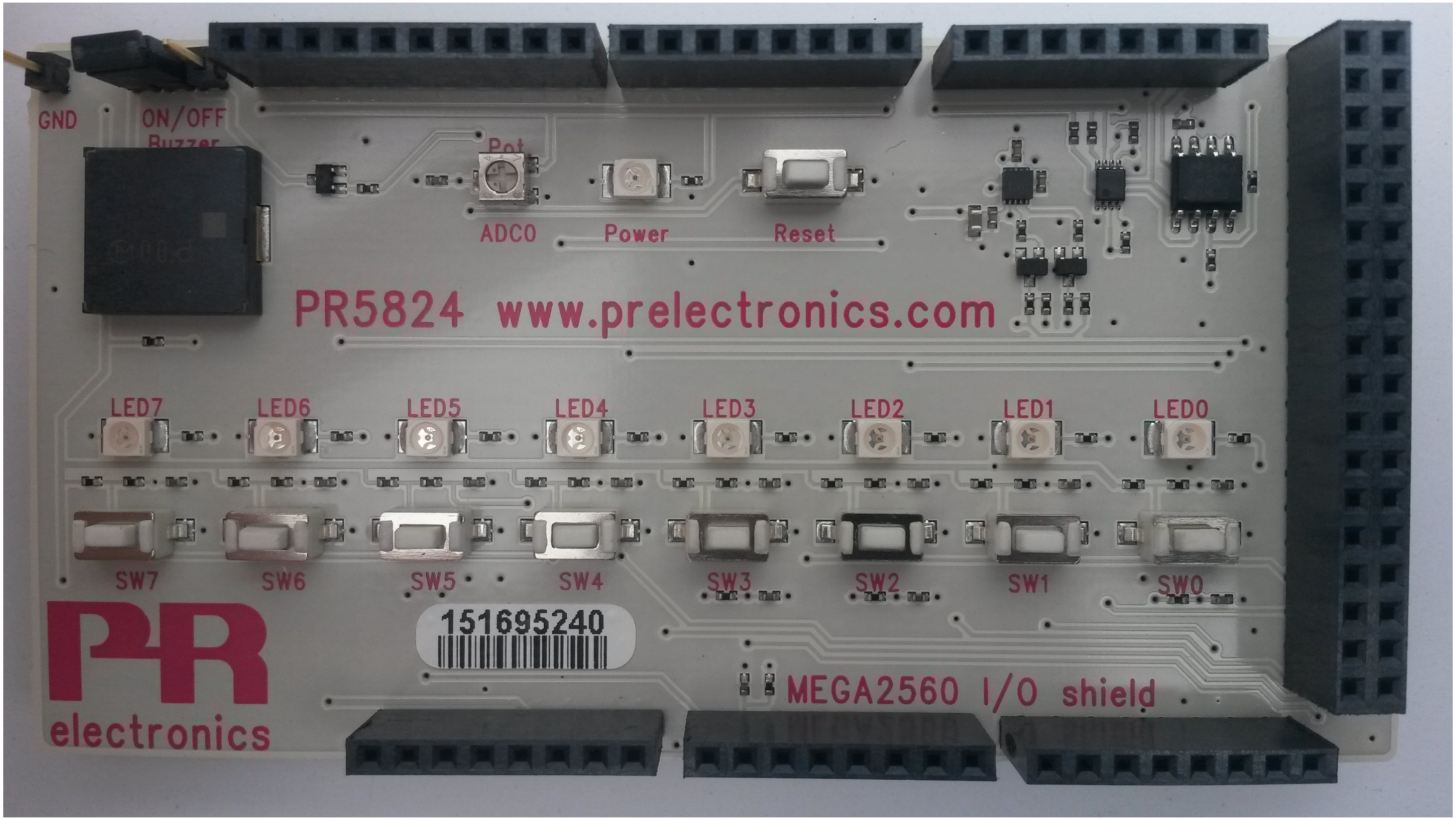


Monteret med "Mega2560 I/O Shield"



AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

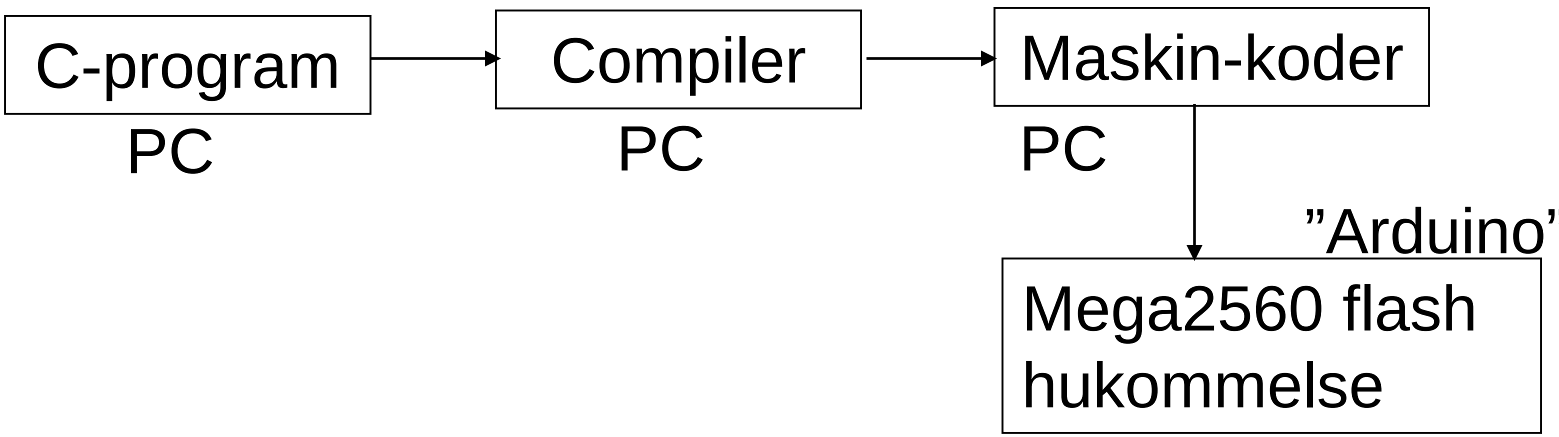
Mega2560 I/O Shield



AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Program - download

- Når C-programmet er oversat af compileren (på PC'en), skal programmet overføres til "vores computer" (= target).
- Dette kan foregå simpelt via et USB kabel.
- I LAB er vores "target" microcontrolleren Mega2560. Den har en speciel hukommelse (Flash), der husker programmet, selv om vi slukker for strømmen.



Test ("socrative.com": Room = MSYS)

- Hvad er en af fordelene ved at skrive kode i assembly kode (frem for f.eks. C) ?

A:

Koden passer til alle computere.

B:

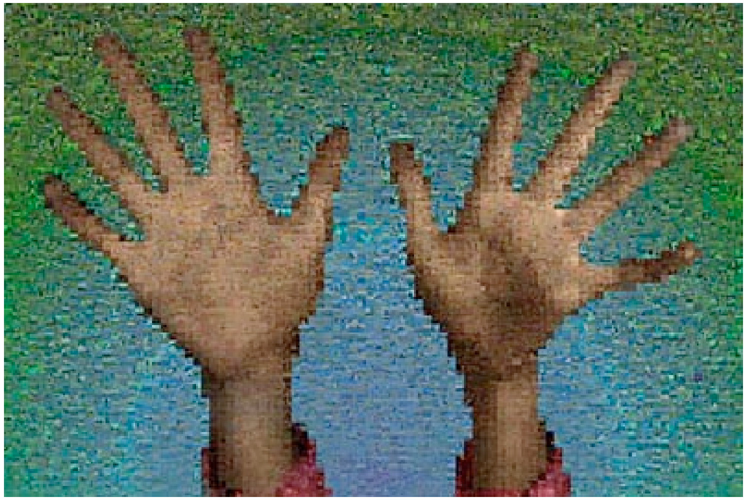
Man har fuld kontrol over, hvad der sker.

C:

Der er meget nemt at læse et Assembly program.

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Decimal / binær



- Hvis vi kun havde haft 2 fingre, ville vi foretrække at regne binært (som en computer) !

Decimale og binære tal

Converting from binary to decimal

To convert from binary to decimal, it is important to understand the concept of weight associated with each digit position. First, as an analogy, recall the weight of numbers in the base 10 system, as shown in the diagram. By the same token, each digit position of a number in base 2 has a weight associated with it:

$$740683_{10} =$$

3×10^0	=	3
8×10^1	=	80
6×10^2	=	600
0×10^3	=	0000
4×10^4	=	40000
7×10^5	=	700000
		740683

$$110101_2 =$$

				Decimal	Binary
1×2^0	=	1×1	=	1	1
0×2^1	=	0×2	=	0	00
1×2^2	=	1×4	=	4	100
0×2^3	=	0×8	=	0	0000
1×2^4	=	1×16	=	16	10000
1×2^5	=	1×32	=	32	100000
				53	110101

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Test (”socrative.com”: Room = MSYS)

- Hvordan skrives det binære tal 00010100 som decimalt tal ?

A: 40

B: 10

C: 20

D: 5

Fra decimal til binær

Example 0-1

Convert 25_{10} to binary.

Solution:

```
                Quotient   Remainder
25/2  =         12         1      LSB (least significant bit)
12/2  =         6          0
6/2   =         3          0
3/2   =         1          1
1/2   =         0          1      MSB (most significant bit)
```

Therefore, $25_{10} = 11001_2$.

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Test (”socrative.com”: Room = MSYS)

- Hvordan skrives det decimale tal 217 som binært tal ?

A: 00101100

B: 01101100

C: 11000001

D: 11011001

AARHUS UNIVERSITY SCHOOL OF ENGINEERING

Hexa-decimale tal

Table 0-1: Base 16 Number System

Decimal	Binary	Hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Bemærk:
A = 10
B = 11
C = 12
D = 13
E = 14
F = 15

Eksempler med HEX-tal

Example 0-4

Represent binary 100111110101 in hex.

Solution:

First the number is grouped into sets of 4 bits: 1001 1111 0101.
Then each group of 4 bits is replaced with its hex equivalent:

1001	1111	0101
9	F	5

Therefore, 100111110101_2 = 9F5 hexadecimal.

Example 0-5

Convert hex 29B to binary.

Solution:

		2	9	B
29B	=	0010	1001	1011

Dropping the leading zeros gives 1010011011.

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Test (”socrative.com”: Room = MSYS)

- Hvordan skrives det decimale tal 217 som ”hex” tal ?

A: 17

B: 6C

C: C1

D: D9

Binær addition

Table 0-3: Binary Addition

A + B	Carry	Sum
0 + 0	0	0
0 + 1	0	1
1 + 0	0	1
1 + 1	1	0

Example 0-8

Add the following binary numbers. Check against their decimal equivalents.

Solution:

	Binary	*Decimal*
	1101	13
+	1001	9
	10110	22

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Binær addition

$$
\begin{array}{rr}
 & \\
X & 190 \\
Y & +\,141 \\
\hline
X+Y & 331
\end{array}
\qquad
\begin{array}{cccccccccc}
 & & & 1 & 1 & 1 & 1 & & & \\
 & & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
+ & & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
\hline
 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1
\end{array}
\qquad\qquad
\begin{array}{rr}
 & \\
X & 173 \\
Y & +\,44 \\
\hline
X+Y & 217
\end{array}
\qquad
\begin{array}{ccccccccc}
 & & 1 & & 1 & 1 & & & \\
 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\
+ & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
\hline
 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1
\end{array}
$$

Figure 2–1 Examples of decimal and corresponding binary additions.

Hexa-decimal addition

Example 0-10

Perform hex addition: 23D9 + 94BE.

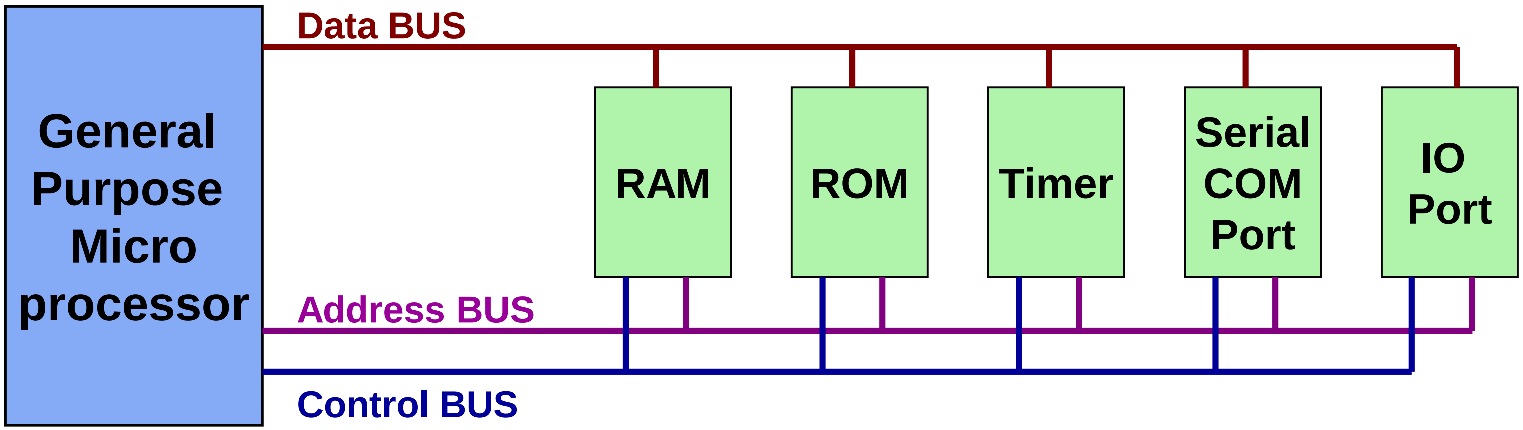
Solution:

23D9	LSD:	9 + 14 = 23	23 – 16 = 7 with a carry
+ 94BE		1 + 13 + 11 = 25	25 – 16 = 9 with a carry
B897		1 + 3 + 4 = 8	
	MSD:	2 + 9 = B	

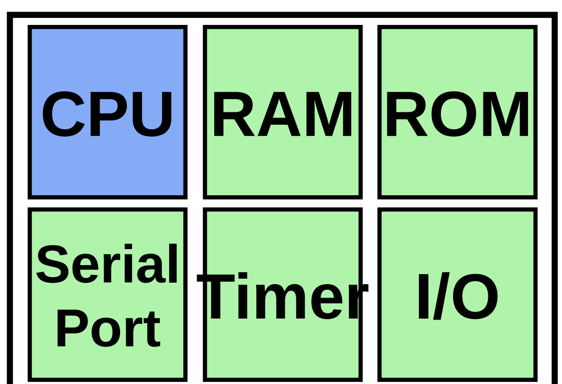
AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

General Purpose Microprocessors vs. Microcontrollers

- General Purpose Microprocessors



- Microcontrollers



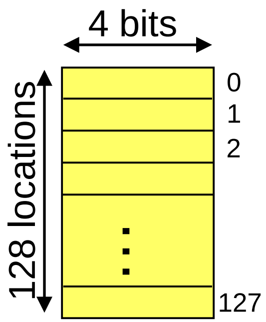
AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Intern organisering af computere

- CPU
- Memory
- I/O
  - Input
    - F.eks. Keyboard, Mus, Sensor
  - Output
    - F.eks, LCD, printer, robot-hænder

Memory characteristics

- Capacity
  - The number of bits that a memory can store.
    - E.g. 128 Kbits, 256 Mbits
- Organization
  - How the locations are organized
    - E.g. a 128 x 4 memory has 128 locations, 4 bits each



- Access time
  - How long it takes to get data from memory

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Semiconductor memories

- **ROM**
  - Mask ROM
  - PROM (Programmable ROM)
  - **EPROM** (Erasable PROM)
  - **EEPROM** (Electronic Erasable PROM)
  - **Flash Memory**

- RAM
  - **SRAM** (Static RAM)
  - **DRAM** (Dynamic RAM)
  - NV-RAM (Nonvolatile RAM)

Memory\ROM\EPROM (Erasable Programmable ROM)

- UV-EPROM
  - You can shine ultraviolet (UV) radiation to erase it
  - Erasing takes up to 20 minutes
  - The entire contents of ROM are erased

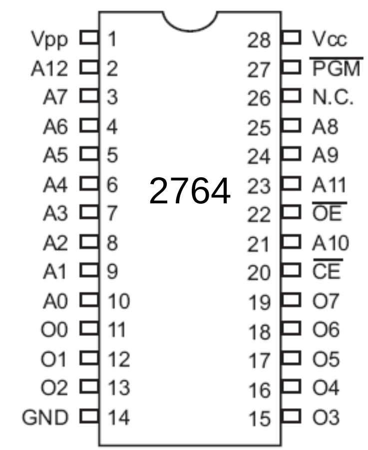
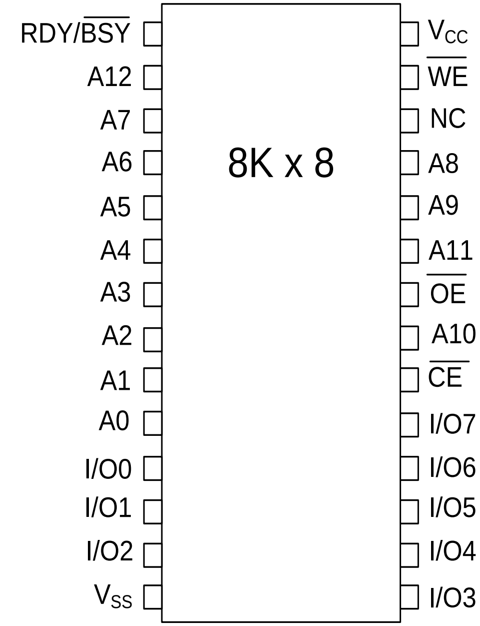


Table 0-5: Some UV-EPROM Chips

Part #	Capacity	Org.	Access	Pins	V_{PP}
2716	16K	2K × 8	450 ns	24	25 V
2732	32K	4K × 8	450 ns	24	25 V
2732A-20	32K	4K × 8	200 ns	24	21 V
27C32-1	32K	4K × 8	450 ns	24	12.5 V CMOS
2764-20	64K	8K × 8	200 ns	28	21 V
2764A-20	64K	8K × 8	200 ns	28	12.5 V
27C64-12	64K	8K × 8	120 ns	28	12.5 V CMOS

Memory\ROM\EEPROM (Electrically Erasable Programmable ROM)

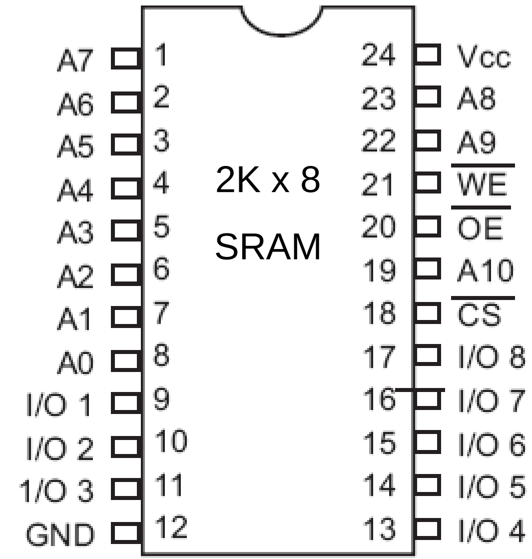
- Erased Electrically
  - Erased instantly
  - Each byte can be erased separately



Part No.	Capacity	Org.	Speed	Pins	V_{PP}
2816A-25	16K	2K × 8	250 ns	24	5 V
2864A	64K	8K × 8	250 ns	28	5 V
28C64A-25	64K	8K × 8	250 ns	28	5 V CMOS
28C256-15	256K	32K × 8	150 ns	28	5 V
28C256-25	256K	32K × 8	250 ns	28	5 V CMOS

SRAM (Static RAM)

- Made of flip-flops (Transistors)
- Advantages:
  - Faster
  - No need for refreshing
- Disadvantages:
  - High power consumption
  - Expensive

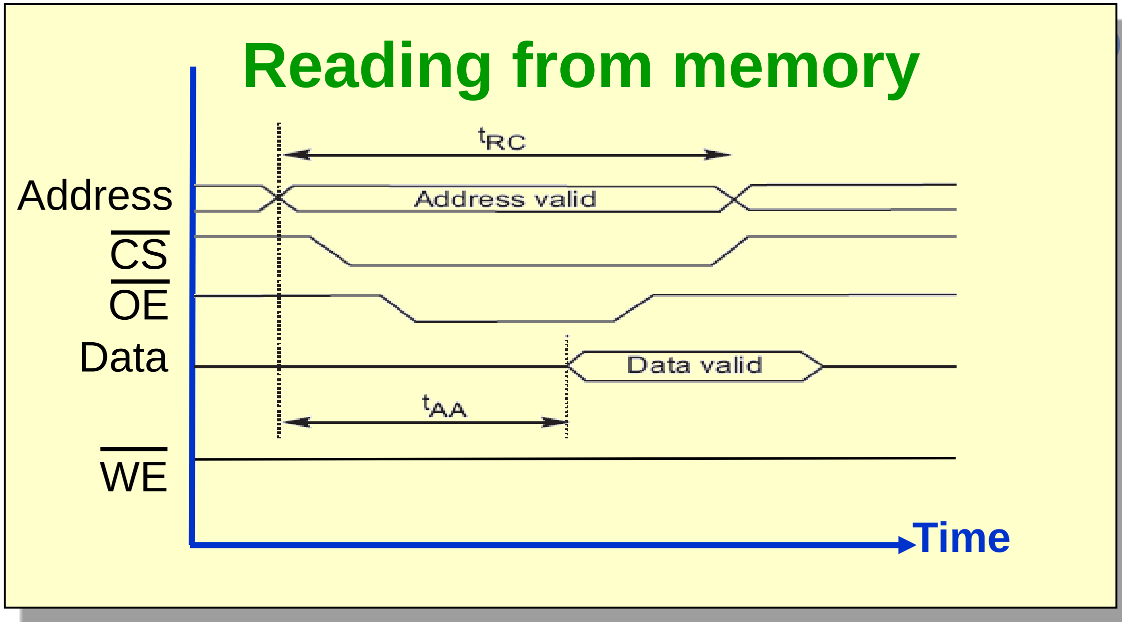


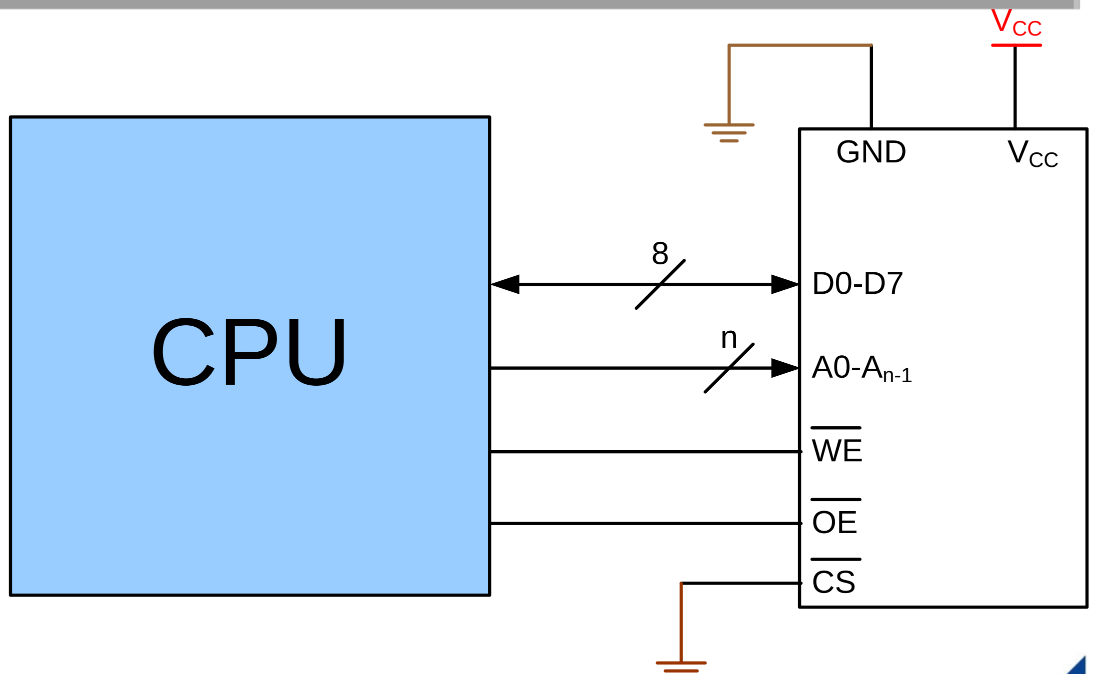
Memory\RAM\DRAM (Dynamic RAM)

- Made of capacitors
- Advantages:
  - Less power consumption
  - Cheaper
  - High capacity
- Disadvantages:
  - Slower
  - Refresh needed

CPU

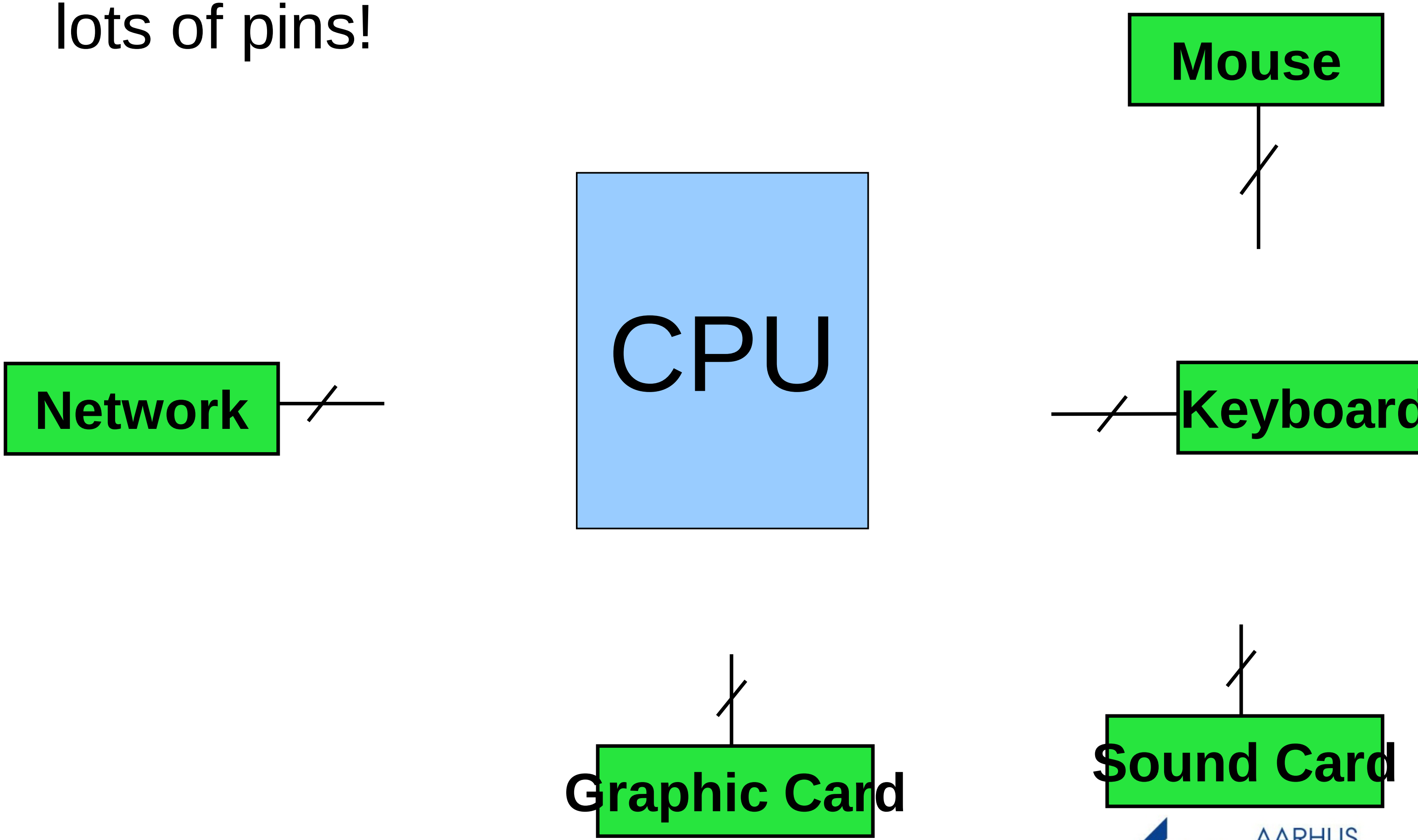
- Tasks:
  - It should execute instructions
    - It should recall the instructions one after another and execute them


Reading from memory
t_{RC}
Address
Address valid
$\overline{CS}$
$\overline{OE}$
Data
Data valid
t_{AA}
$\overline{WE}$
Time

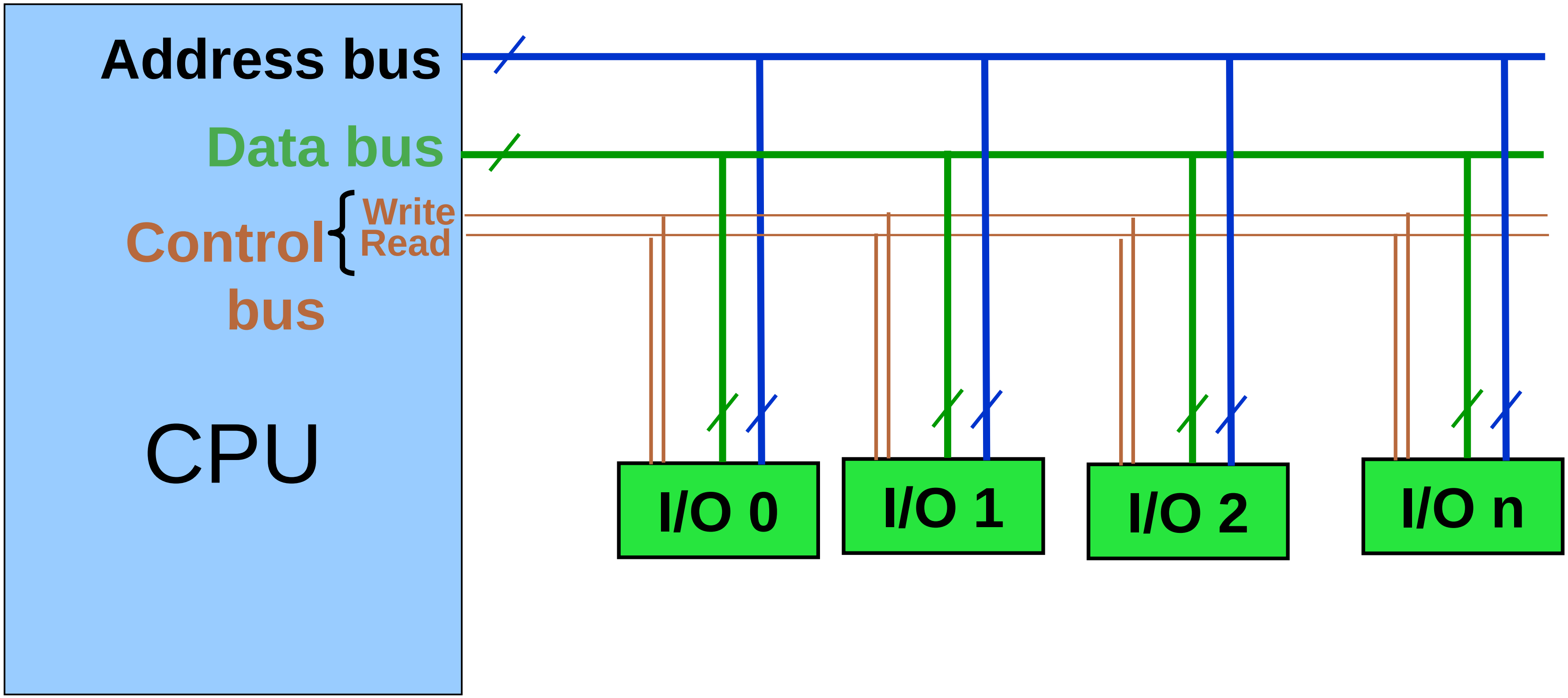

V_{CC}
GND
V_{CC}
CPU
8
D0-D7
n
A0-A_{n-1}
$\overline{WE}$
$\overline{OE}$
$\overline{CS}$

Connecting I/Os to CPU

- CPU should have lots of pins!

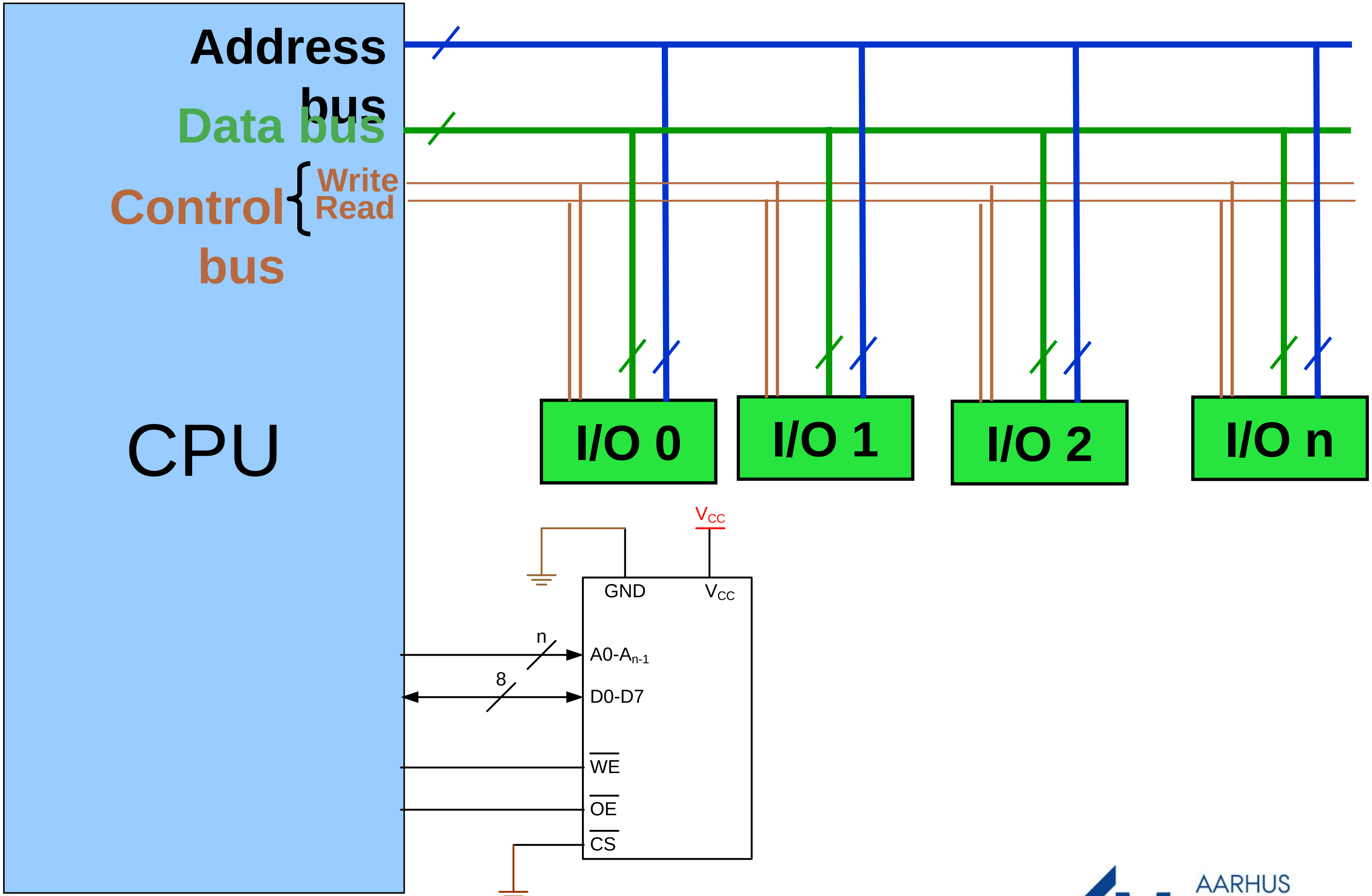


Connecting I/Os to CPU using bus



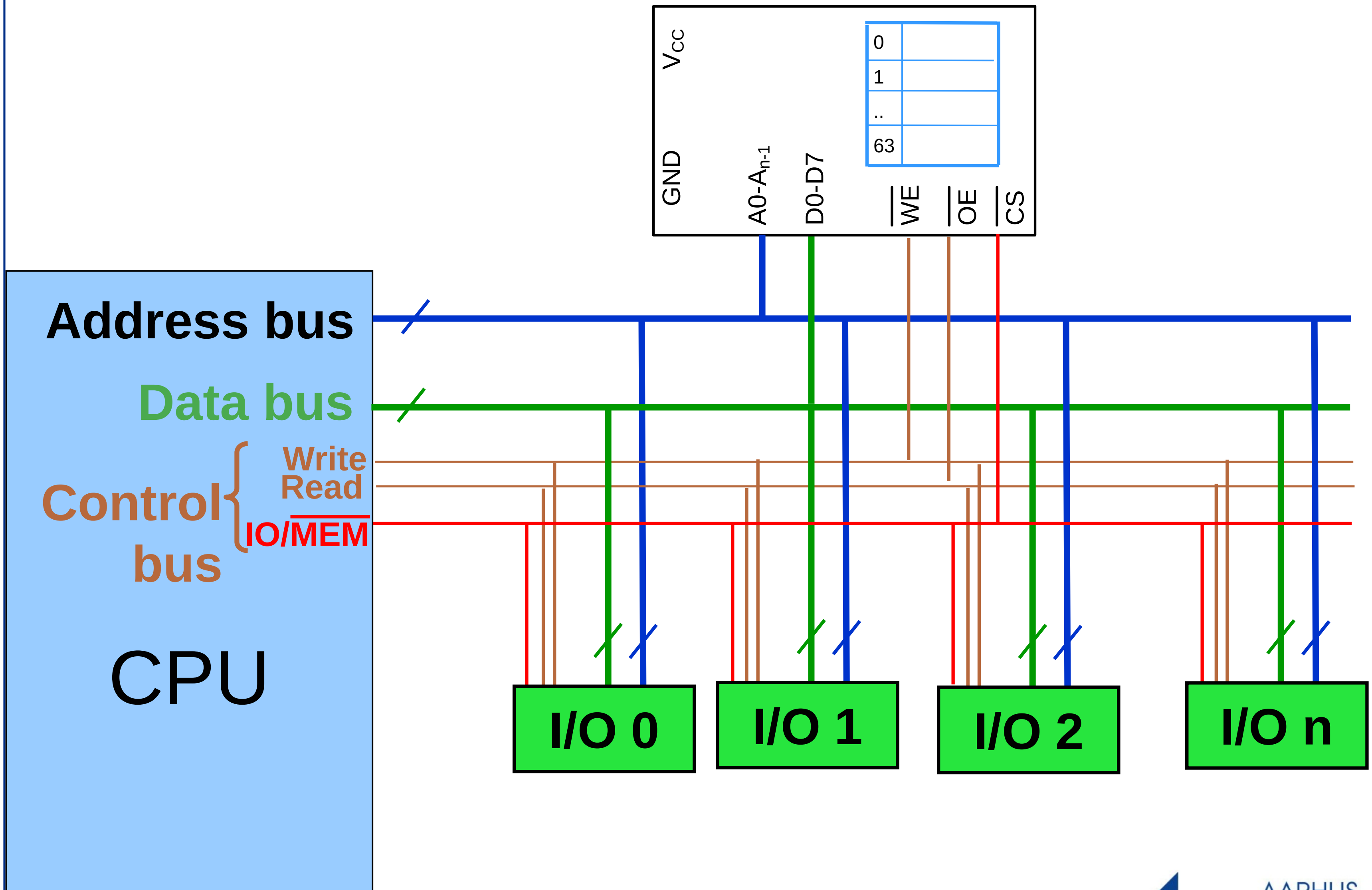
AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Connecting I/Os and Memory to CPU



AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

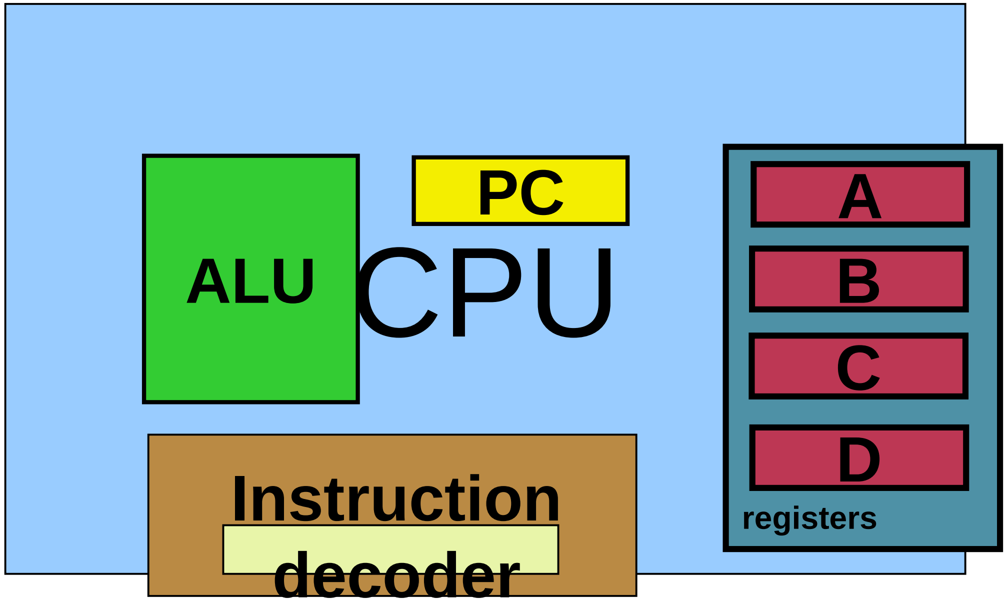
Connecting I/Os and Memory to CPU



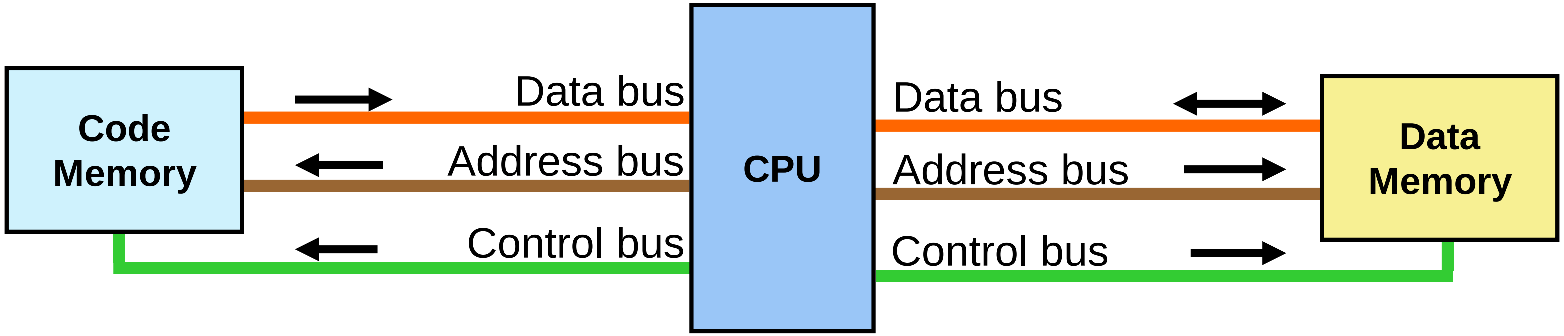
AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Inside the CPU

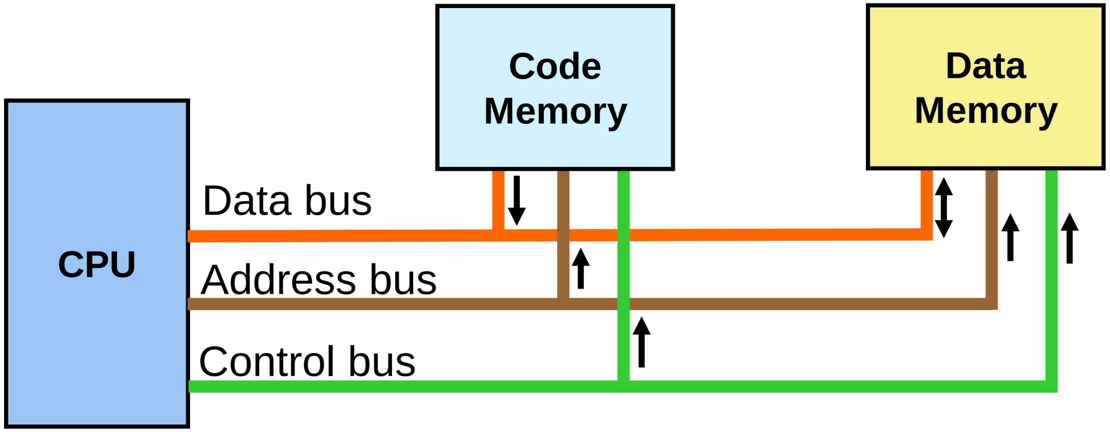
- PC (Program Counter)
- Instruction decoder
- ALU (Arithmetic Logic Unit)
- Registers



Von Neumann vs. Harvard architecture



- Harvard architecture



- Von Neumann architecture

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Slut på MSYS lektion 2

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING