Embedded systems, an introduction

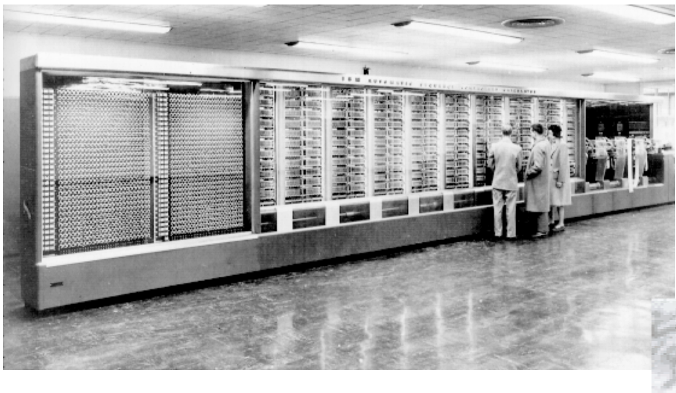
Roger Johansson
Associate Professor
Sven Knutsson
Lecturer

Dept. of Computer Science and Engineering

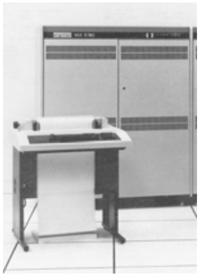
Computer organisation and design

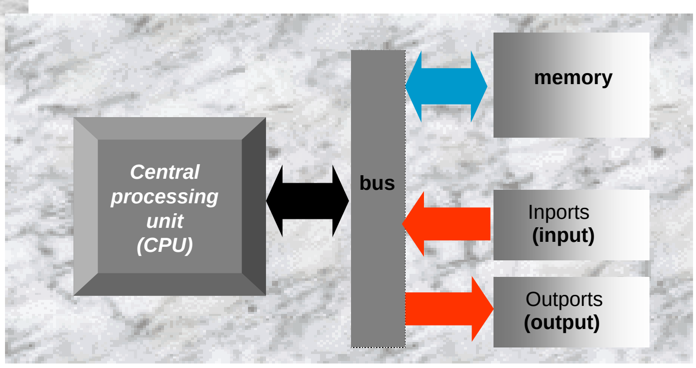
- Introduction to microcomputers
- Number systems
- Computer structures...
- .. and how computers work
- Programming computers

Computers now and then...

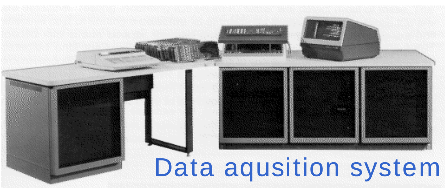


..they share the basic structures...

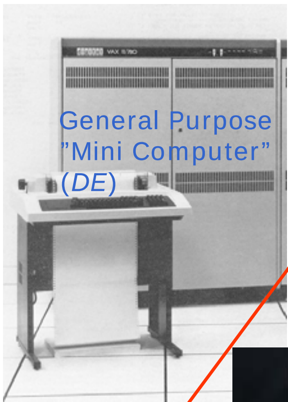




Dedicated computers



Data aquisition system (*HP*)



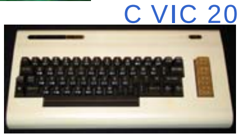
General Purpose "Mini Computer" (*DE*)

Mainframe (*IBM*)

Supercomputers (*Cray*)

Altair 8800

Apple II-e



C VIC 20

SOL

IBM PC-AT

IBM "portable"

Number systems

A decimal number is coded as a sequense of digits, e.g. 435,72.
This is interpreted as

4 is worth four times hundred	$(4 \cdot 10^2)$
3 is worth three times ten	$(3 \cdot 10^1)$
5 is worth five times one	$(5 \cdot 10^0)$
7 is worth seven divided by ten	$(7 \cdot 10^{-1})$
2 is worth two divided by hundred	$(2 \cdot 10^{-2})$

So the number can be expressed as the sum of:

$$4 \cdot 10^2 + 3 \cdot 10^1 + 5 \cdot 10^0 + 7 \cdot 10^{-1} + 2 \cdot 10^{-2}$$

Number systems, *Notation*

Notation for numbers with different number base (*radix*):

$(1001)_2$	radix 2	[0,1]
$(1072)_8$	radix 8	[0-7]
$(2934)_{10}$	radix 10	[0-9]
$(3AF5)_{16}$	radix 16	[0-9,A-F]

Prefix for different radices in the 'C' programming language:

01072	First digit is 0 (zero) denotes an octal number	(radix 8)
2934	A non-zero first (decimal) digit denotes a decimal number	(radix10)
0x3AF5	The prefix "0x" (zero-x) denotes a hexadecimal number	(radix16)

Prefix for different radices in assembler code:

%1001	The prefix '**%**' denotes a binary number	(radix 2)
@1072	The prefix '**@**' denotes an octal number	(radix 8)
2934	No prefix denotes a decimal number	(radix 10)
$3AF5	The prefix '**$**' denotes a hexadecimal number	(radix16)

Number systems, radix conversions, binary to decimal...

$$N_{10} = d_{n-1} * r^{n-1} + d_{n-2} * r^{n-2} + ... + d_0$$

The equation provides simple methods to convert numbers between different number systems, E.g....

Convert binary number $(101110)_2$ to decimal form.

Apply the equation above. The number of binary digits (n) is 6, so

$(N)_{10} = 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 =$

$32 + 0 + 8 + 4 + 2 + 0 = 52$

I.e. $(101110)_2 = (52)_{10}$

Number systems, radix conversions, decimal to binary ...

A rewrite of the equation provides this method...

Convert decimal number 211 to it's binary representation:

Q / r		**quotient**			**remainder**
211/2	=	105	+	1/2	$d_0 = 1$
105/2	=	52	+	1/2	$d_1 = 1$
52/2	=	26	+	0	$d_2 = 0$
26/2	=	13	+	0	$d_3 = 0$
13/2	=	6	+	1/2	$d_4 = 1$
6/2	=	3	+	0	$d_5 = 0$
3/2	=	1	+	1/2	$d_6 = 1$
1/2	=	0	+	1/2	$d_7 = 1$

I.e. : $(211)_{10} = (11010011)_2$

Number systems, radix conversions, decimal to hexadecimal ...

Another rewrite of the equation provides this method...

Convert decimal number 211 to it's hexadecimal representation:

Q / r	**quotient**		**remainder**
211/16 =	$(13)_{10} = (D)_{16}$	3/16	$d_0=3$
13/16 =	0	13/16	$d_1=D$

I.e. $(211)_{10} = (D3)_{16}$

Number systems, natural binary coded decimal numbers , (NBCD code) ...

A group of four binary digits corresponds to *one* decimal digit...

The decimal number 9756 is expressed in NBCD-code as:

9756 =	1001	0111	0101	0110
	9	7	5	6

Number systems, ASCII ...

ASCII = American Standard Code for Information Interchange

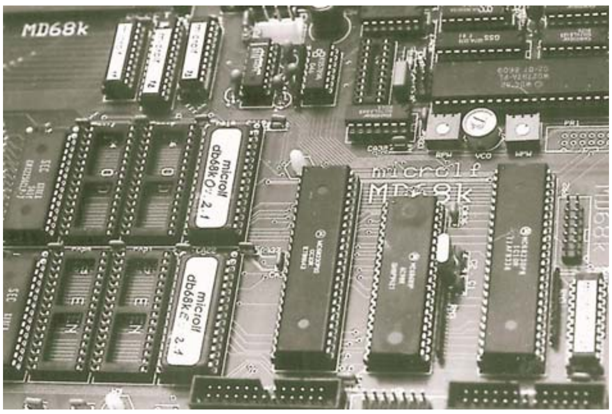
Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII
0	NUL	20		40	@	60	`
1	SOH	21	!	41	A	61	a
2	STX	22	"	42	B	62	b
3	ETX	23	#	43	C	63	c
4	EOT	24	$	44	D	64	d
5	ENQ	25	%	45	E	65	e
6	ACK	26	&	46	F	66	f
7	BEL	27	'	47	G	67	g
8	BS	28	(	48	H	68	h
9	HT	29	)	49	I	69	i
A	LF	2A	*	4A	J	6A	j
B	VT	2B	+	4B	K	6B	k
C	FF	2C	,	4C	L	6C	l
D	CR	2D	-	4D	M	6D	m
E	SO	2E	.	4E	N	6E	n
F	S1	2F	/	4F	O	6F	o
10	DLE	30	0	50	P	70	p
11	DC1	31	1	51	Q	71	q
12	DC2	32	2	52	R	72	r
13	DC3	33	3	53	S	73	s
14	DC4	34	4	54	T	74	t
15	NAK	35	5	55	U	75	u
16	SYN	36	6	56	V	76	v
17	ETB	37	7	57	W	77	w
18	CAN	38	8	58	X	78	x
19	EM	39	9	59	Y	79	y
1A	SUB	3A	:	5A	Z	7A	z
1B	ESC	3B	;	5B	[Ä	7B	{ ä
1C	FS	3C	<	5C	\ Ö	7C	| ö
1D	GS	3D	=	5D	] Å	7D	} å
1E	RS	3E	>	5E	^	7E	~
1F	US	3F	?	5F	_	7F	DEL

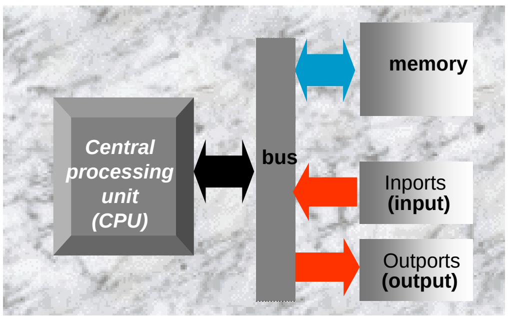
Provides an interpretation of 8-bit digital numbers into alphabetical and numerical characters.

There are also some control characters included. Most of these are mainly of historical interest.

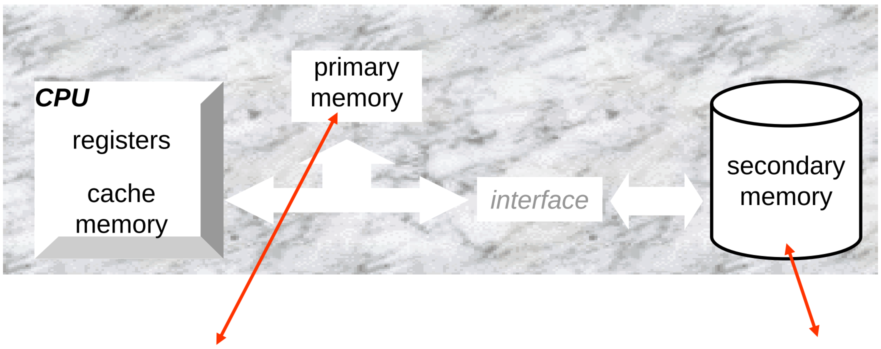
Computer structures

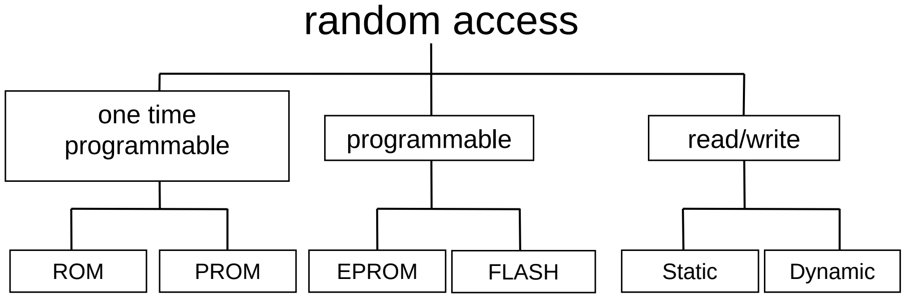
basically…





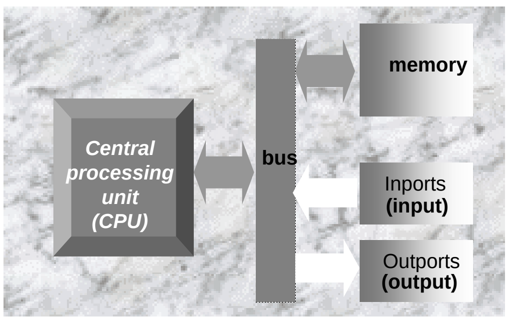
Memory hierarchy

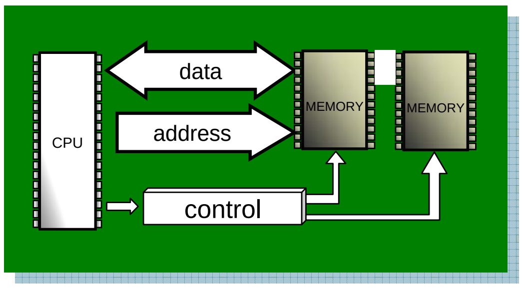


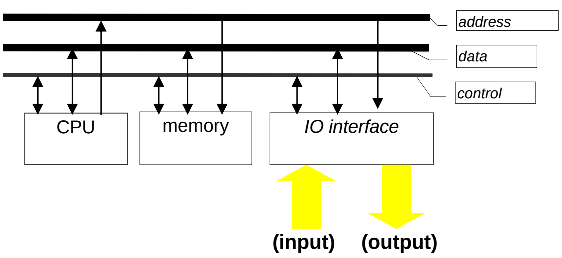


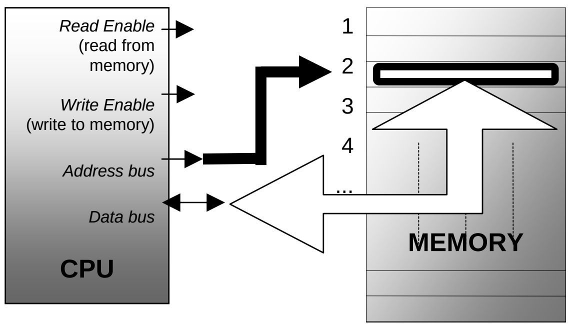
block access
- hard discs
- CD/DVD
- Memory sticks
- Smartcards
- etc...

Primary memory access

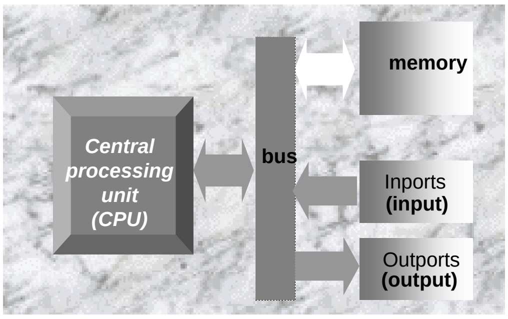


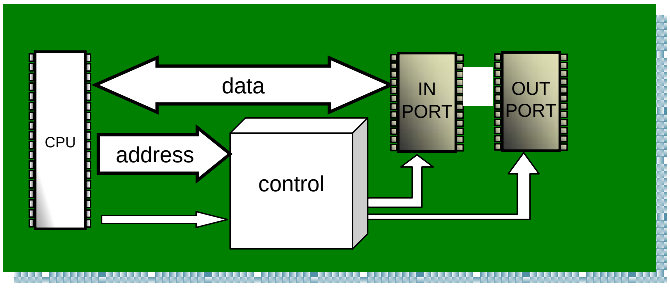


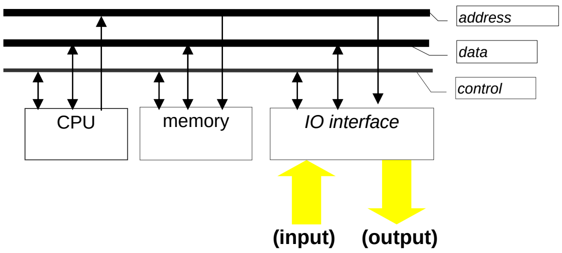


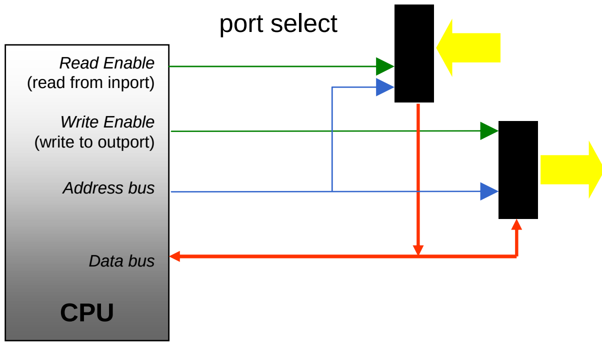


Peripheral access

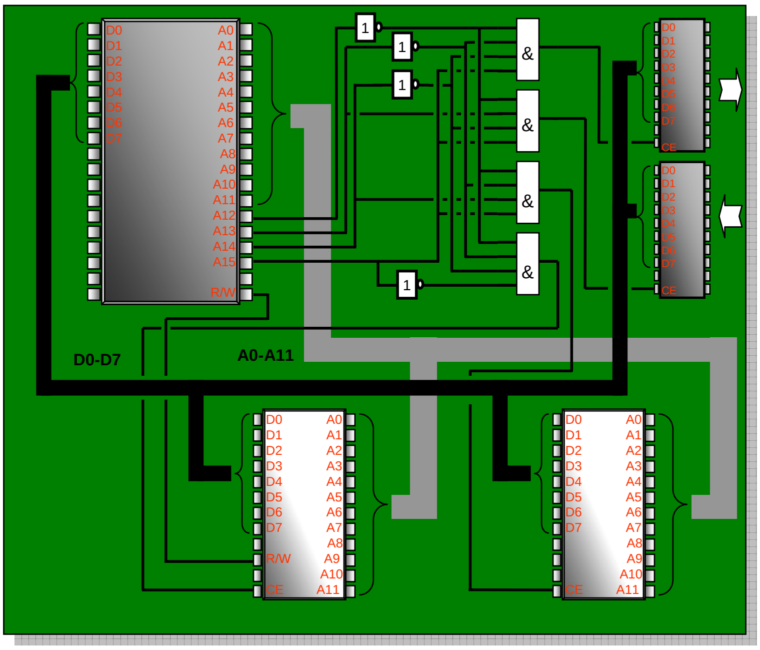


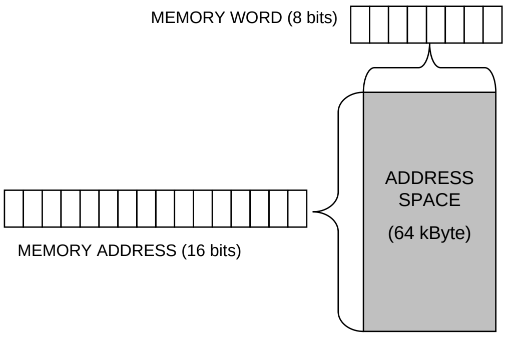




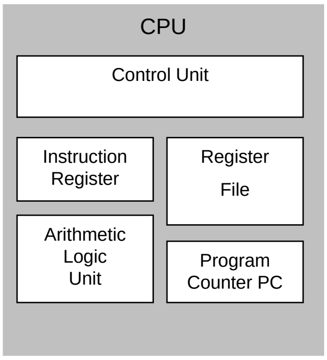


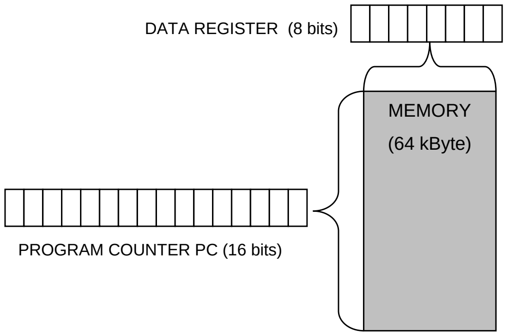
The design





Central Processing Unit



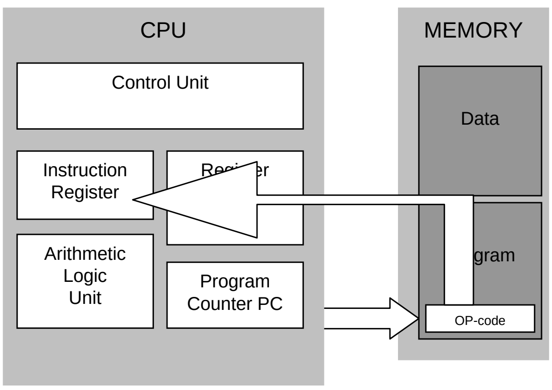


Width of PC determines adressable space, i.e. 16 bits

Width of registers in ”Register File” determines data word size, i.e. 8 bits.

Instruction Fetch

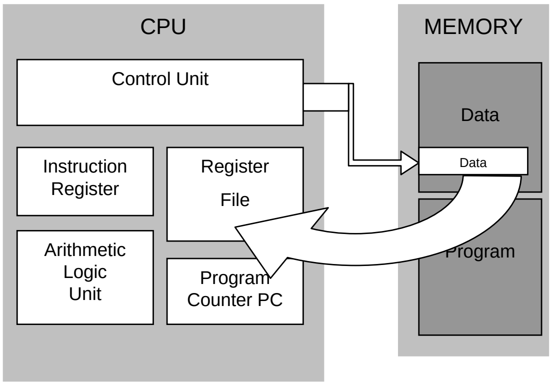
When powered on, the CPU starts fetching and executing instructions



The Operation Code (OP-code) details the instruction...

Instruction Execute example: Load from Memory

After decoding this instruction the control unit generates the specified memory address.



Memory contents is copied to a CPU register.

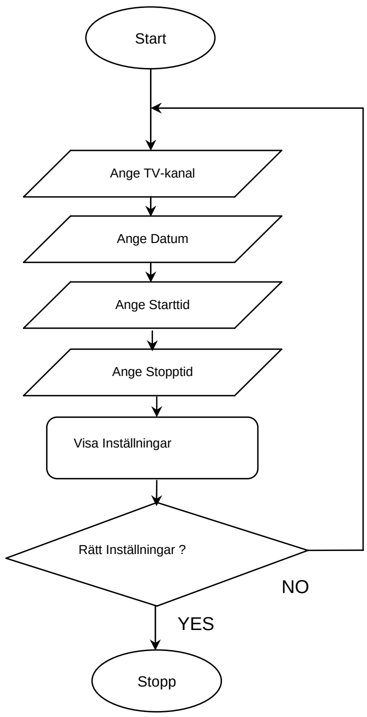
Programming computers

A computer only do what it is ***instructed*** to do.

Such instructions are introduced by a ***program***

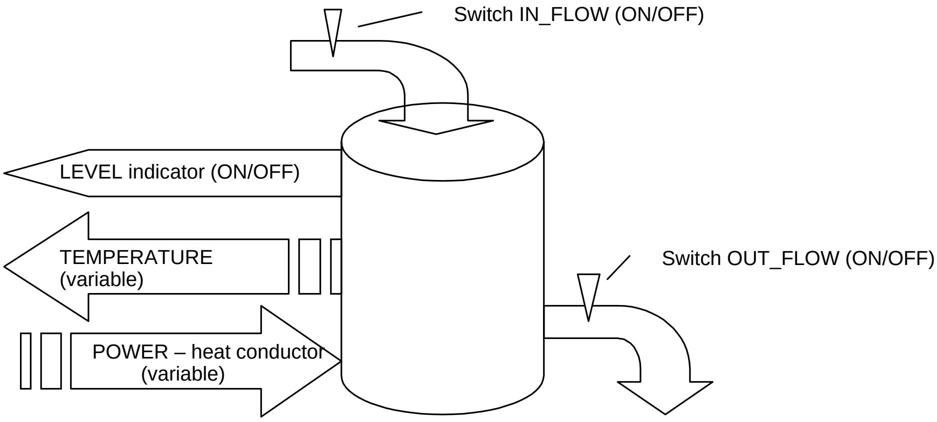
A description of what a computer program is intended to do is called an ***algoritm***.

A ***flow chart*** is a detailed interpretation of an algoritm.



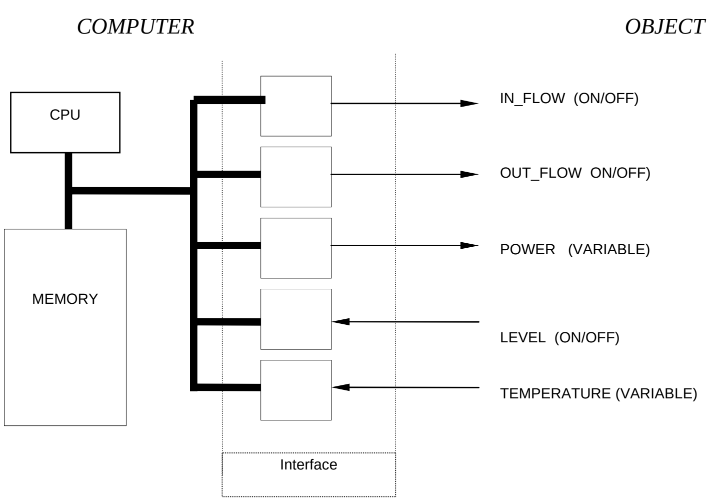
The Water Heater

Consider a water heater with inflow (cold water) and outflow which is supposed to be appropriate for a shower...



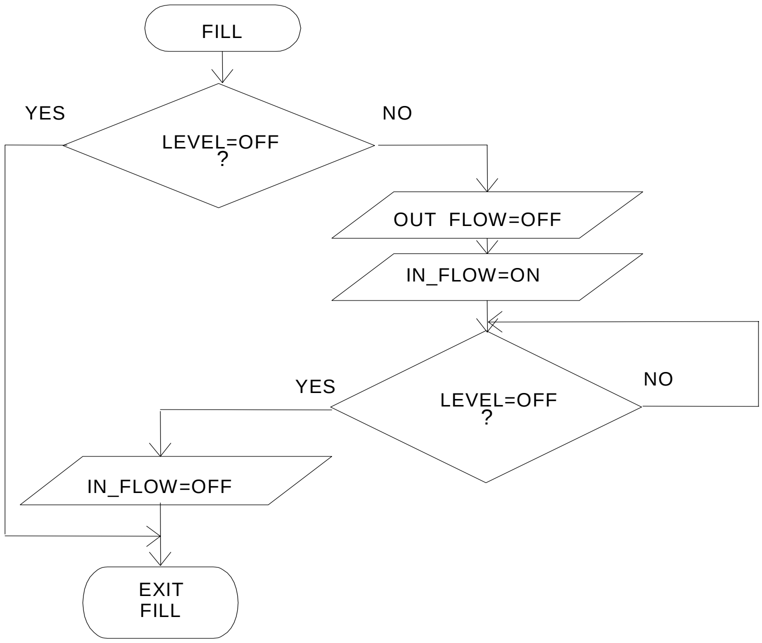
The Water Heater

We apply the water heater to a computer structure and sees what we get .



The Water Heater

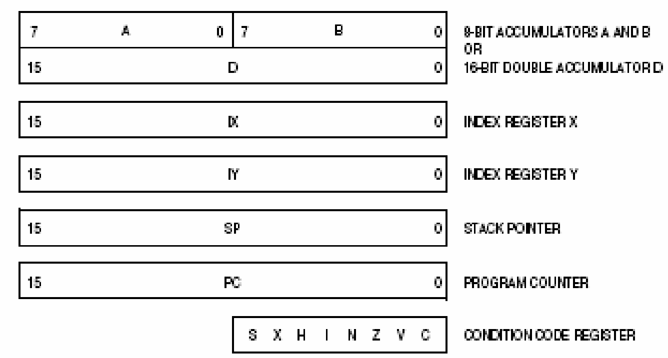
We now want to fill up the tank with water.



CPU12 – Assembly programmers model

The CPU12 programming model, has two 8-bit general-purpose accumulators (A and B) that can be concatenated into a single 16-bit accumulator (D) for certain instructions. It also has:

- Two index registers (X and Y)
- 16-bit stack pointer (SP)
- 16-bit program counter (PC)
- 8-bit condition code register (CCR)



CPU12 – Load and Store Instructions

Mnemonic	Function	Operation
Load Instructions		
LDAA	Load A	(M) ⇒ A
LDAB	Load B	(M) ⇒ B
LDD	Load D	(M : M + 1) ⇒ (A:B)
LDS	Load SP	(M : M + 1) ⇒ SP_H:SP_L
LDX	Load index register X	(M : M + 1) ⇒ X_H:X_L
LDY	Load index register Y	(M : M + 1) ⇒ Y_H:Y_L
LEAS	Load effective address into SP	Effective address ⇒ SP
LEAX	Load effective address into X	Effective address ⇒ X
LEAY	Load effective address into Y	Effective address ⇒ Y

Store Instructions		
STAA	Store A	(A) ⇒ M
STAB	Store B	(B) ⇒ M
STD	Store D	(A) ⇒ M, (B) ⇒ M + 1
STS	Store SP	(SP_H:SP_L) ⇒ M : M + 1
STX	Store X	(X_H:X_L) ⇒ M : M + 1
STY	Store Y	(Y_H:Y_L) ⇒ M : M + 1

EXAMPLE, copy byte data from memory location $600 to $400...

```
LDAA   $600
STAA   $400
```

or..

```
LDAB   $600
STAB   $400
```


CPU12 – Addition and Subtraction Instructions

Mnemonic	Function	Operation
Addition Instructions		
ABA	Add B to A	(A) + (B) ⇒ A
ABX	Add B to X	(B) + (X) ⇒ X
ABY	Add B to Y	(B) + (Y) ⇒ Y
ADCA	Add with carry to A	(A) + (M) + C ⇒ A
ADCB	Add with carry to B	(B) + (M) + C ⇒ B
ADDA	Add without carry to A	(A) + (M) ⇒ A
ADDB	Add without carry to B	(B) + (M) ⇒ B
ADDD	Add to D	(A:B) + (M : M + 1) ⇒ A : B
Subtraction Instructions		
SBA	Subtract B from A	(A) – (B) ⇒ A
SBCA	Subtract with borrow from A	(A) – (M) – C ⇒ A
SBCB	Subtract with borrow from B	(B) – (M) – C ⇒ B
SUBA	Subtract memory from A	(A) – (M) ⇒ A
SUBB	Subtract memory from B	(B) – (M) ⇒ B
SUBD	Subtract memory from D (A:B)	(D) – (M : M + 1) ⇒ D

EXAMPLE, subtract 10 from value at $600, store result at $400...

```
LDAA   $600
SUBA   #10
STAA   $400
```

NOTE:
means ”immediate” i.e the actual value rather than the value at location 10...

CPU12 – Assembler programs

```
InPort        EQU   $600          substitute address with symbol
OutPort       EQU   $400
* Line is treated as a comment
              ORG   $1000         code starts at this address
              LDAA  InPort
              SUBA  #10
              STAA  OutPort
              STAA  OutPortCopy   save copy in variable

              ORG   $2000
OutPortCopy   RMB   1             reserve space for variable
```


Same program in ‘C’...

```
unsigned char OutPortCopy;

#define      InPort       ((char *) 0x600)
#define      OutPort      ((char *) 0x400)

...
       OutPortCopy = *Inport - 10;
       *OutPort = OutPortCopy ;
...
```

Conclusion

we have got a brief introduction to

- microcomputers
- number systems
- computer structures...
- .. and how computers work
- programming computers

which finishes today's lecture ...