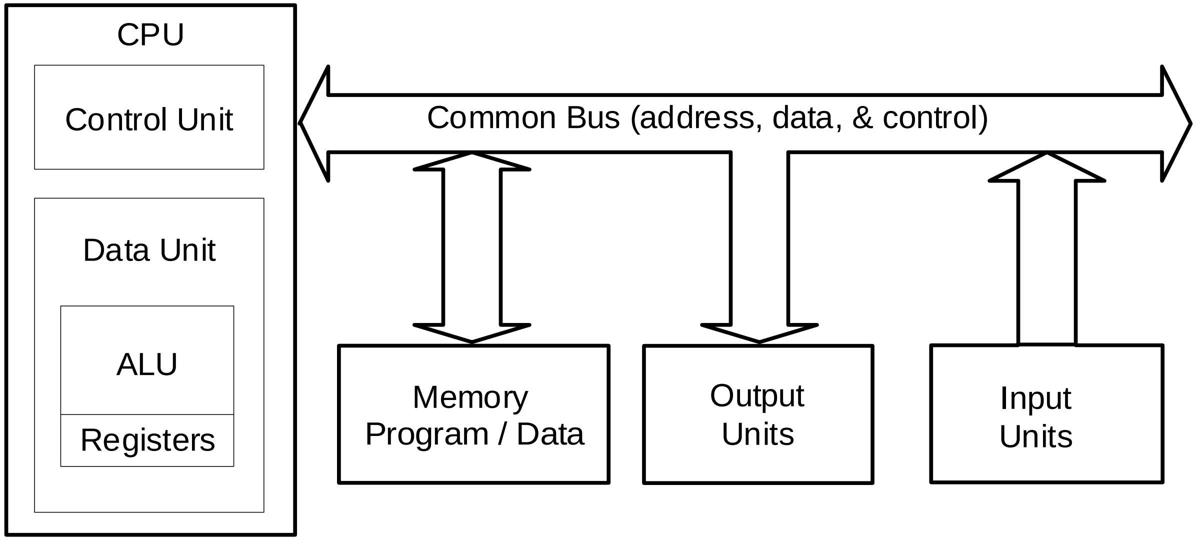
Computer

What is a computer?

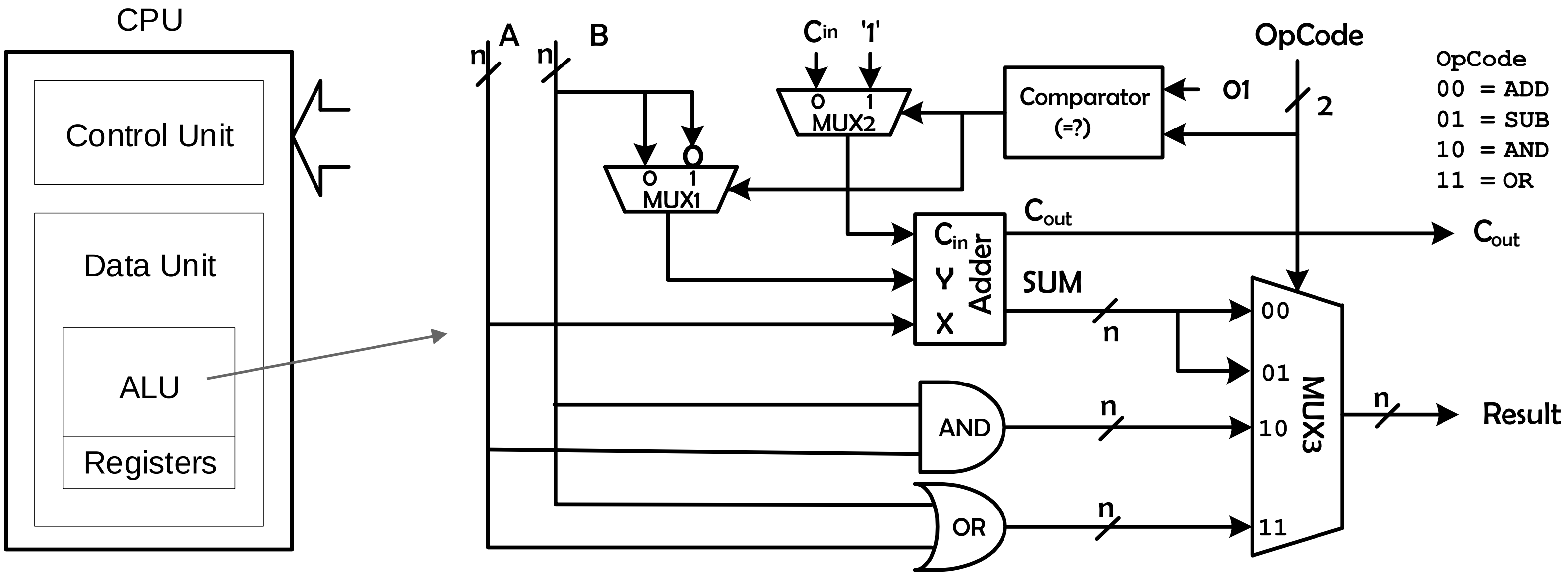


HW + SW

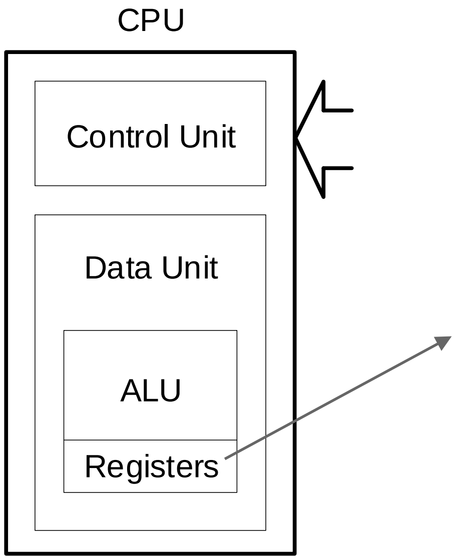
Numbers: %, @, $

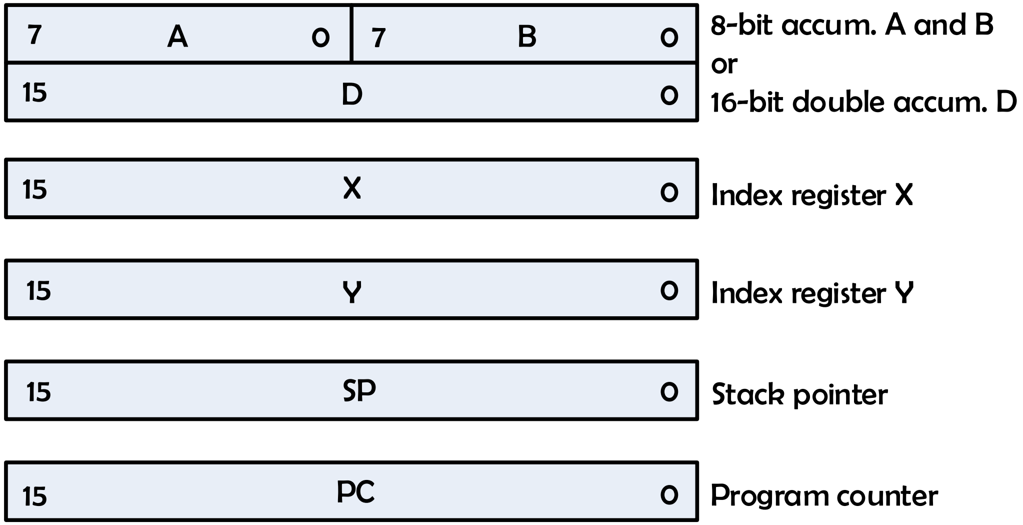
Reproduced with permission from Delmar Cengage Learning

ALU


CPU
Control Unit
Data Unit
ALU
Registers
A
B
n
n
Cin
'1'
0
1
MUX2
Comparator
(=?)
01
OpCode
2
0
1
MUX1
Cout
Cin
Y
X
Adder
SUM
n
AND
OR
n
n
00
01
10
11
MUX3
n
Result
Cout
OpCode
00 = ADD
01 = SUB
10 = AND
11 = OR

CPU Registers





7 A 0	7 B 0

15 D 0

8-bit accum. A and B
or
16-bit double accum. D

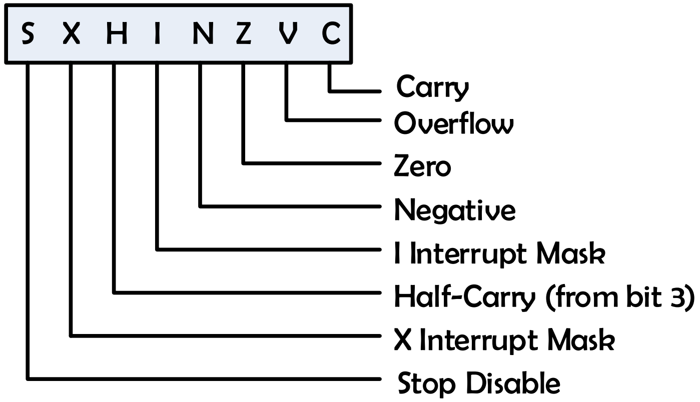
15 X 0 — Index register X

15 Y 0 — Index register Y

15 SP 0 — Stack pointer

15 PC 0 — Program counter

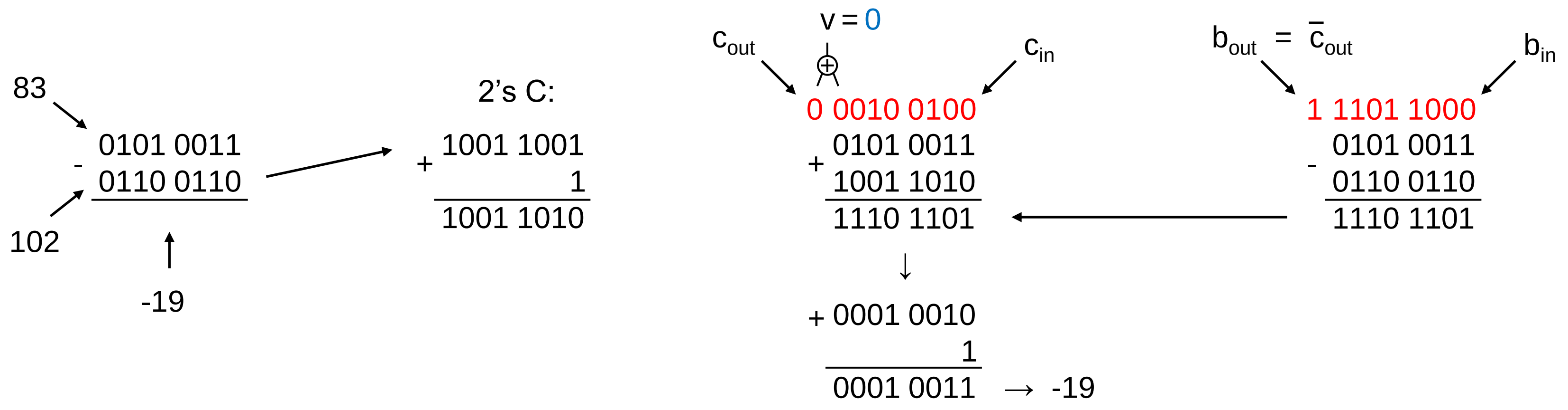
Condition code register (CCR)



S	X	H	I	N	Z	V	C

Carry
Overflow
Zero
Negative
I Interrupt Mask
Half-Carry (from bit 3)
X Interrupt Mask
Stop Disable

Overflow



83

102

- 0101 0011
- 0110 0110

-19

2's C:

+ 1001 1001
+ 1
= 1001 1010

$v = 0$

c_{out} c_{in}

0 0010 0100

+ 0101 0011
+ 1001 1010
= 1110 1101

+ 0001 0010
+ 1
= 0001 0011 → -19

$b_{out} = \bar{c}_{out}$ b_{in}

1 1101 1000

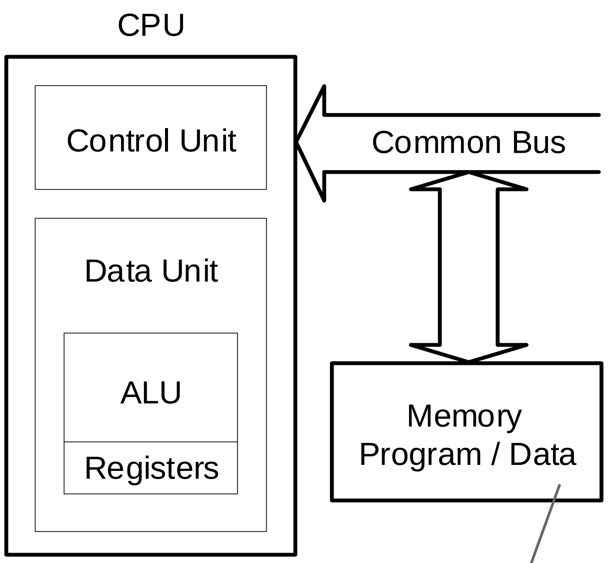
- 0101 0011
- 0110 0110
= 1110 1101

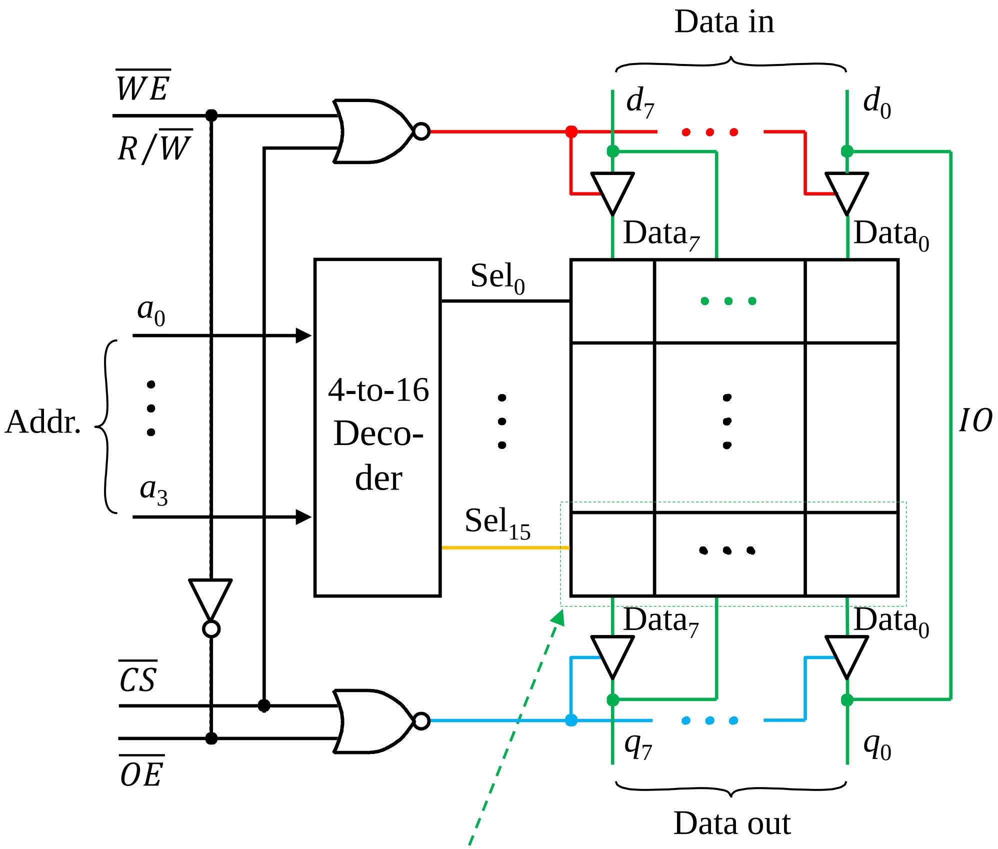
Signed numbers extension

Unsigned:			Signed:		
0000 0001	→	0000 0000 0000 0001	0000 0001	→	0000 0000 0000 0001
\$01	→	\$0001	\$01	→	\$0001
1111 1111	→	0000 0000 1111 1111	1111 1111	→	1111 1111 1111 1111
\$FF	→	\$00FF	\$FF	→	\$FFFF

-1 -1

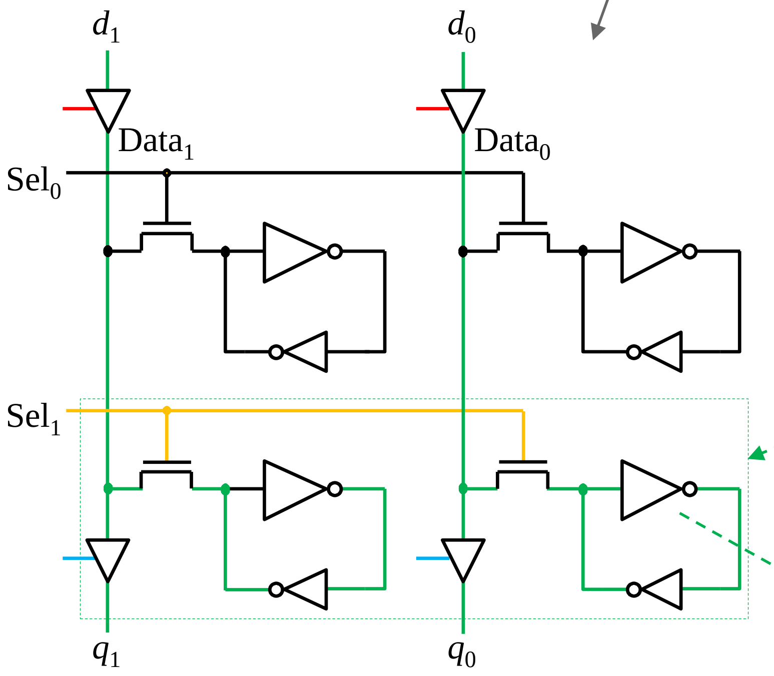
Memory System

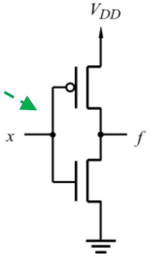




Memory cell
(word, address, location)

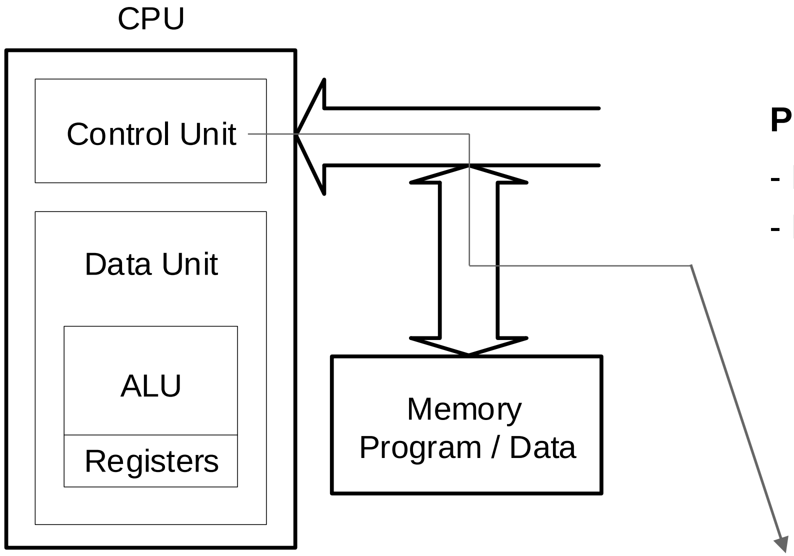
A 16 × 8 array of SRAM cells





A 2 × 2 array of SRAM cells

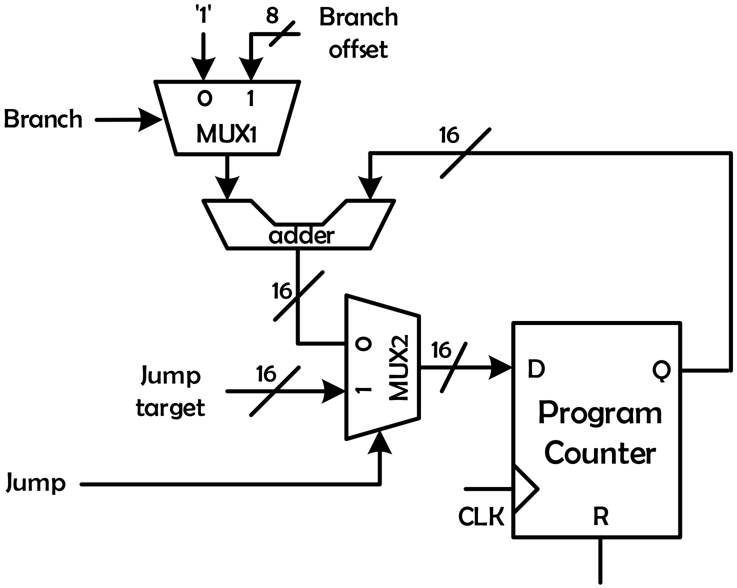
Program Execution and Program Counter (PC)



Program Execution:

- PC starts from fixed location: PIC18, AVR, 8051
- PC fetches the start address from fixed location: HCS12

Program Counter



Control Unit: Machine instruction: opcode + operand fields. Program Counter. Instruction fetch -> Instruction Register. Sequential locations; conditional & unconditional branch / jump.

Machine instruction:	01000011	A ← A + 1
Assembly language:	INCA	A ← A + 1
	10000110 00001001	A ← 9 (Load ACCA with number 9)
	LDAA #9	

Native assembler (compiler) and **cross** assembler (compiler). **Source / object** code.

Registers: Data or address - fast access. Dedicated register (Acc) – always an operand (HCS12, PIC18, 8051). Or, many general-purpose registers as operands (AVR, PIC32, Coldfire).

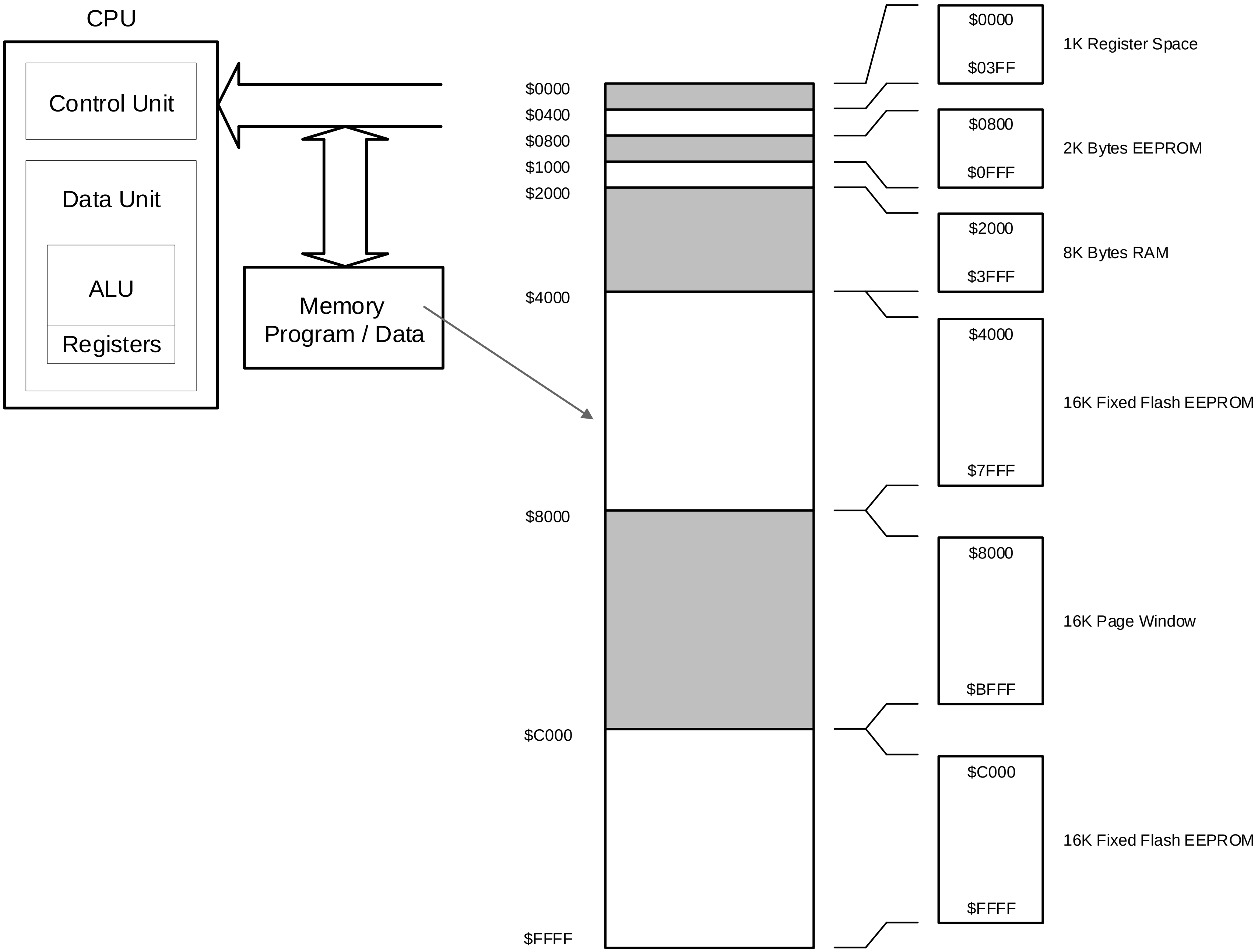
µP: on a single chip. Intel 4004 (1968), 8008 (1972). No peripherals, no timer, no ADC or DAC, no memory.

µC: all of the above + parallel & serial communication interface (USART, I2C, SPI, CAN, Ethernet), DMA, SW debug support. Used everywhere.

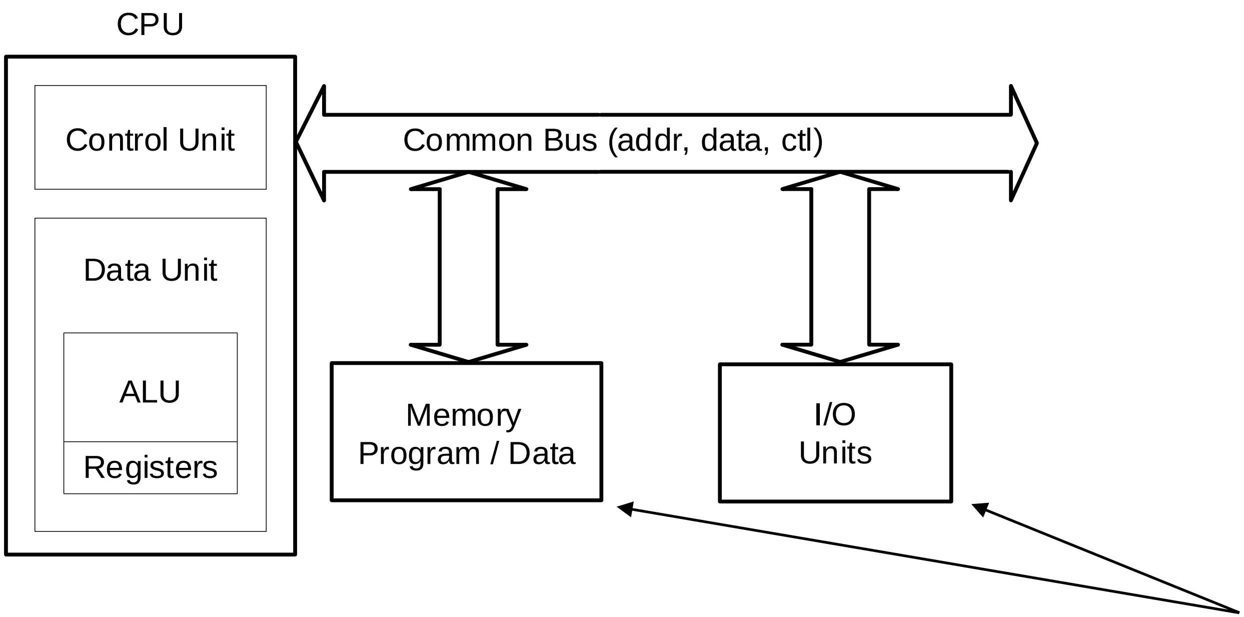
Embedded Systems: A computer designed to perform dedicated function. SW called FW.

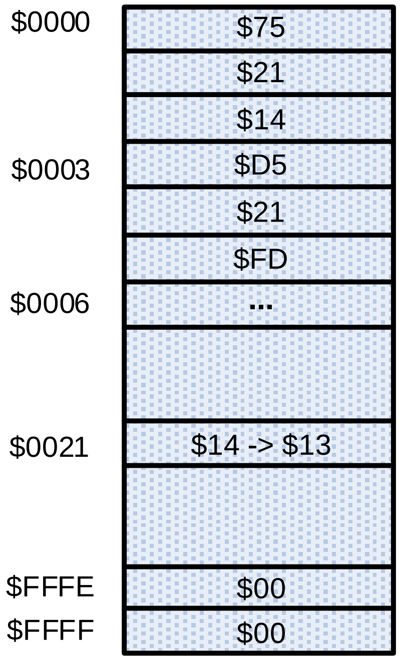
Memory: Magnetic (tape, drum, disk), Optical (CD, DVD), Semiconductor (volatile, nonvolatile, RAM, ROM, DRAM, SRAM, Magnetoresistive (35ns), Ferroelectric (55ns), Mask-programmable (MROM – when manufactured), Programmable ROM (PROM) – blowable fuses, EPROM, EEPROM, FLASH.

9S12 Memory Map in Normal Single Chip Mode



Example





	Assembly Instr		Comment	Addr	Mach Code
	ld	$21,#20	[21]=$14	$0000	75 21 14
loop:	dbnz	$21,loop	decr.[21]; br.if ≠0	$0003	D5 21 FD
	…			$0006	…

$0006+$FFFD=$0003 ($FD comes from instruction, $FFFD = -$3)

HCS12: 16-bit CPU, 64 KB mem. (expand. to 1 MB), 0-4KB EEPROM, 2-14KB SRAM, 32-512KB FLASH, Timer, SCI, SPI, 10-bit ADC.

Addressing Modes

Machine Instruction: 1-2 byte **opcode** + 0-5 byte **operand** addressing information.

1) Inherent Mode: Does not specify operands: NOP; INCA; INX; DECA

2) Immediate Mode: Operands included in the instruction: LDAA #55; LDX #1000

3) Direct Mode: Uses 1 byte to specify address: LDAA 55; LDAB 100

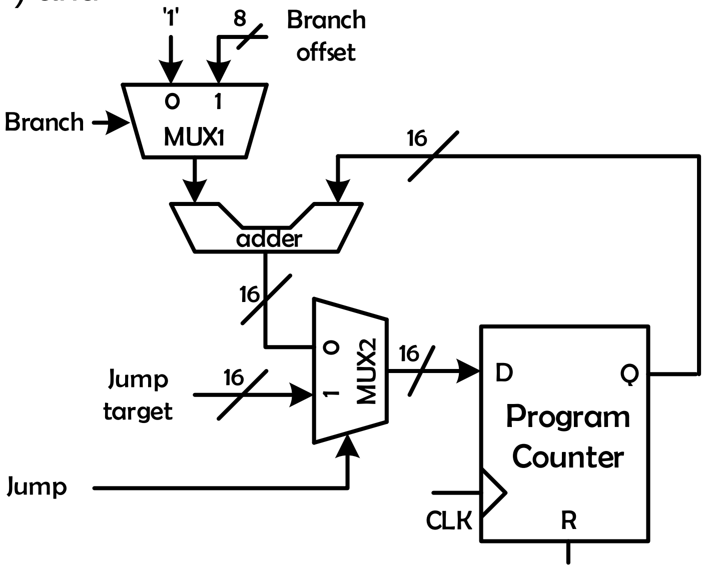
4) Extended Mode: Uses 2 bytes to specify address: LDAA 5555; LDX 6034

5) Relative Mode: Used in branch instructions; short 8-bit (-128~+127), long 16-bit (-32768~+32767)

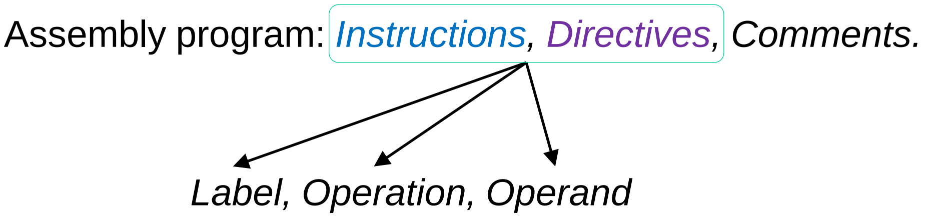
6) Indexed Mode Uses sum of index (base) register (X, Y, PC, or SP) and offset, which can be up to 16-bit signed value or contents of A, B, D:

a) LDAA 2000,X; LDAA D,X

b) Pre/Post Decrement/Increment: LDAA 2,-X; STAA 4,SP+



HCS12 Instructions



Assembly program: *Instructions*, *Directives*, *Comments.*

Label, Operation, Operand

Instructions

Load / Store (mem <-> reg):

LDAA $20

STX $8000

Transfer / Exchange (reg <-> reg):

TFR A,B ; [A] ⇒ B

EXG D,X

TFR A,X ; sign-extended [A] ⇒ X

TFR X,A ; X[7:0] ⇒ A

EXG A,X ; $00:[A] ⇒ X ; X[7:0] ⇒ A

Move (mem <-> mem):

MOVB $100,$800

MOVW 0,X, 0,Y

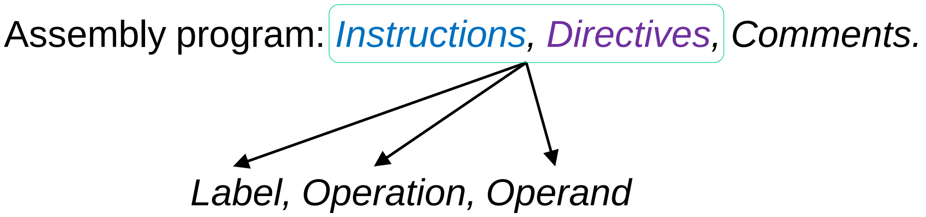
Add / Subtract

ADDA $1000; ADCA $1000 (A ⇐ [A] + [$1000] + C)

SUBA $1000; SBCA $1000 (A ⇐ [A] – [$1000] – C)

ABA (B + A ⇒ A); ABX (B + X ⇒ X)

HCS12 Assembly Programming



Assembly program: *Instructions*, *Directives*, *Comments.*

Label, Operation, Operand

Directives

```
lp      equ     20
        org     $1000
arry    dc.b    $11,$22             ; %00010001, …
        dc.w    $1234, $3355        ; %0001001000110100, …
        fcc     ‘AB’                ; $41, $42
buf     ds.b    100
dbuf    ds.w    20
        dc.b    ‘AB’,0              ; $41, $42, $00
```