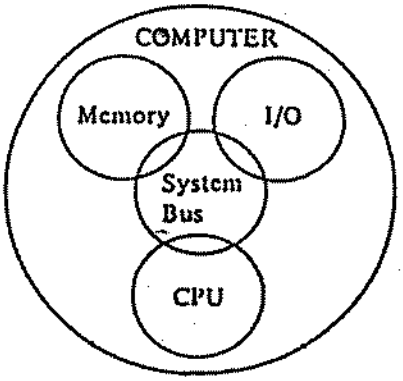
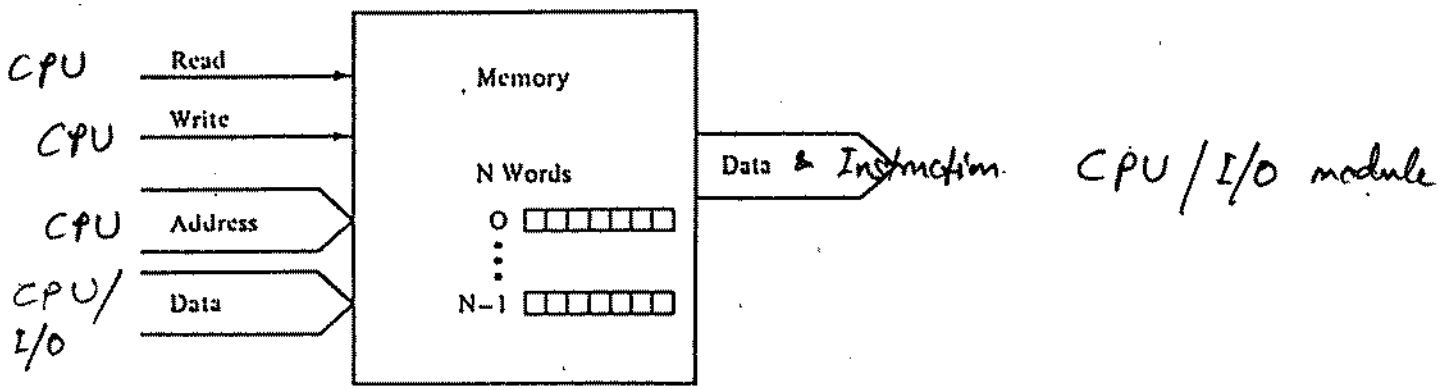
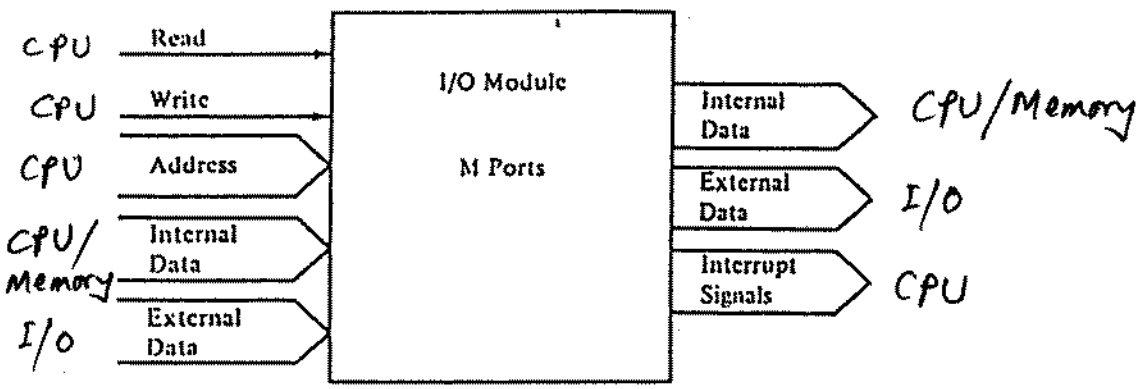
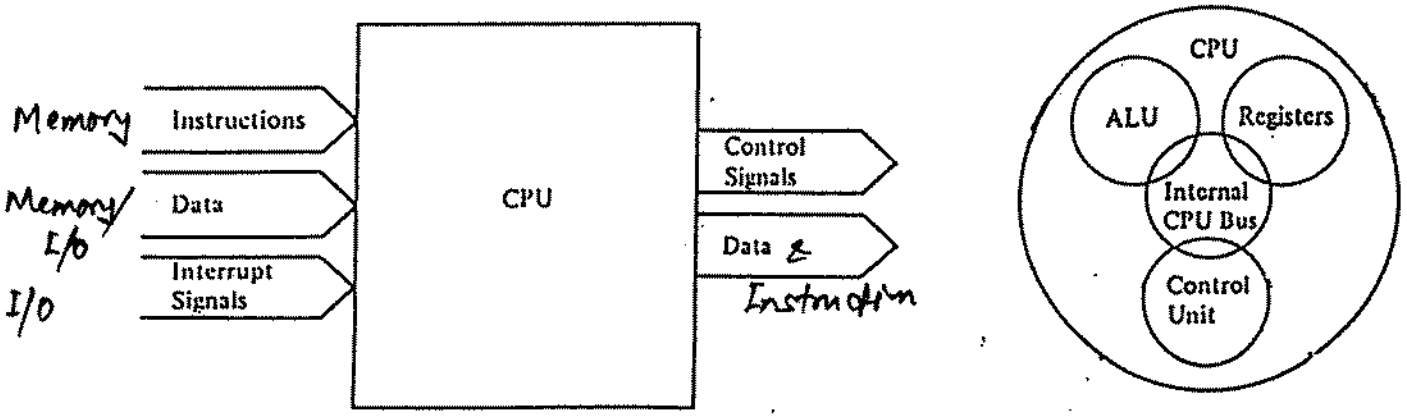
General Structure of a digital Computer









Computer modules

- *Operation Repertoire:* How many and which operations to provide, and how complex operations should be.
- *Data Types:* The various types of data upon which operations are performed.
- *Instruction Format:* Instruction length (in bits), number of addresses, size of various fields, and so on.
- *Registers:* Number of CPU registers that can be referenced by instructions, and their use.
- *Addressing:* The mode or modes by which the address of an operand is specified.

CPU Actions For Various Types of Operations

Data Transfer	Transfer data from one location to another
	If memory is involved:
	Determine memory address
	Perform virtual-to-actual-memory address transformation
	Check Cache
	Initiate Memory Read/Write
Arithmetic	May involve data transfer, before and/or after
	Perform function in ALU
	Set condition codes and flags
Logical	Same as arithmetic
Conversion	Similar to arithmetic and logical. May involve special logic to perform conversion
Transfer of Control	Update program counter. For subroutine call/return, manage parameter passing and linkage
I/O	Issue command to I/O module
	If memory-mapped I/O, determine memory-mapped address

TC2111

* SIMILAR TO TC1101 SEEN IN CSI1101

* **DESCRIPTION**
www.site.uottawa.ca/ftppub/courses/Fall/csi2111
where

tc2111.txt	:	info on TC2111
tc2111.pas	:	simulator for CPU of TC2111
min.dat	:	example program
quo.dat	:	example program
sum.dat	:	example program

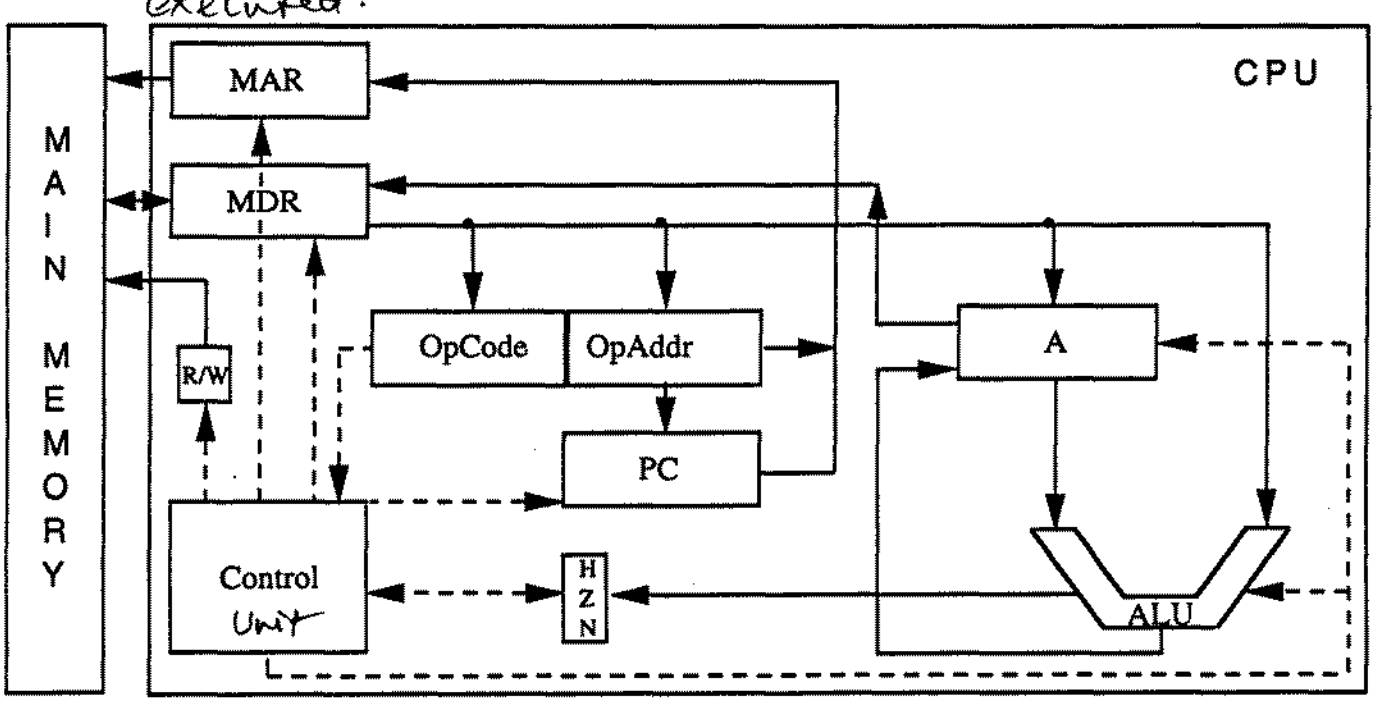
See course web page.

* **CHARACTERISTICS**
16-bit registers
Signed integers in 2's CF
Hexadecimal representation $HH where H stands for an hexadecimal digit
Instruction's opcode is 16 bits $HHHH
Instruction's opaddr is 16 bits $HHHH
IR has a length of 32 bits $HHHH HHHH

* **CPU ARCHITECTURE**

MAR	: MEMORY ADDRESS REGISTER	— 16 bits
MDR	: MEMORY DATA REGISTER	— 16 "
IR	: INSTRUCTION REGISTER	— 32 "
PC	: PROGRAM COUNTER	— 16 "
A	: ACCUMULATOR	— 16 "

PC holds the memory address of the next instruction to be executed.



INSTRUCTION CYCLE

Let's assume two types of Instructions

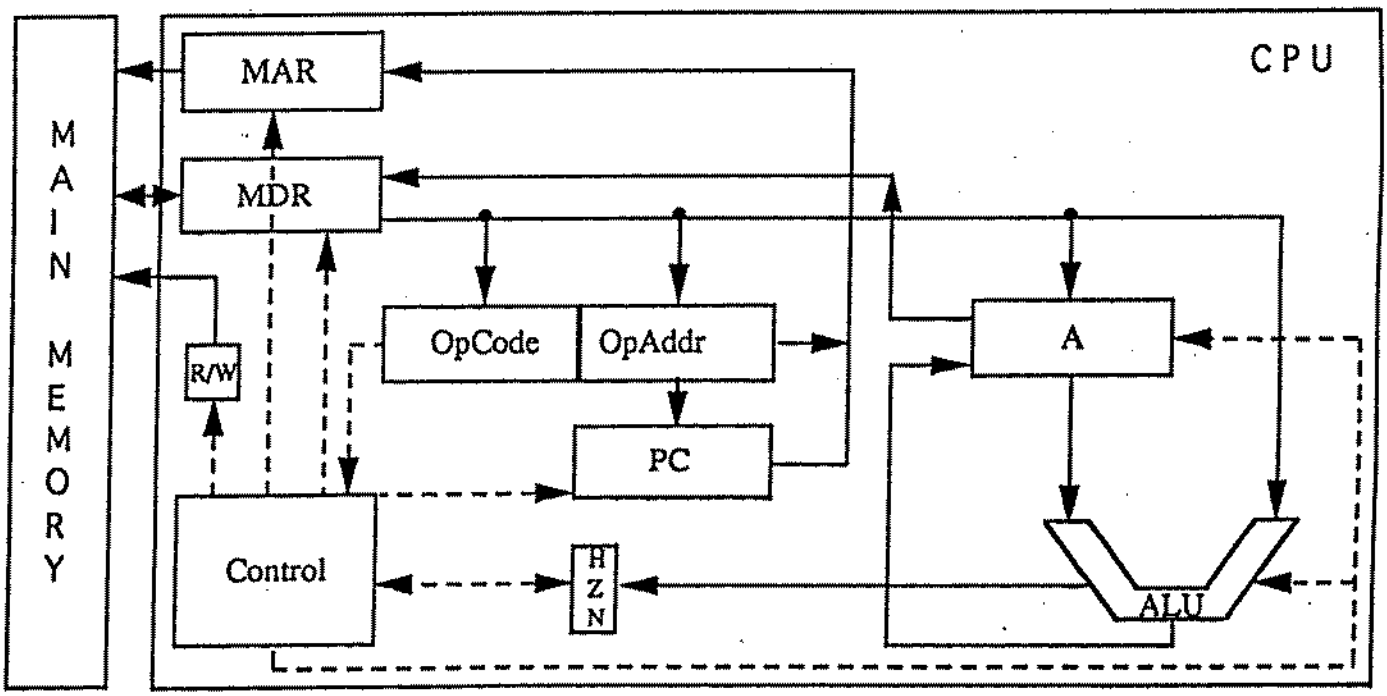
op code

op code op addr

Let's consider the following instruction set

```
(08) cla        (* Clear (set to zero)  the Accumulator *)
(10) inc        (* Increment (add 1 to) the Accumulator *)
     cp2        (* (2's) Complement the Accumulator *)
(64) hlt        (* Halt *)
(91) lda  X     (* Load  the Accumulator from memory *)
(39) sta  X     (* Store the Accumulator into memory *)
(99) add  X     (* Add to the Accumulator *)
(15) jmp  X     (* Jump ("go to ")  *)
(17) jz   X     (* Jump if the Zero     status bit is TRUE *)
(19) jn   X     (* Jump if the Negative status bit is TRUE *)
(01) dsp  X     (* Display (write on the screen) *)
```

Let's consider the following CPU

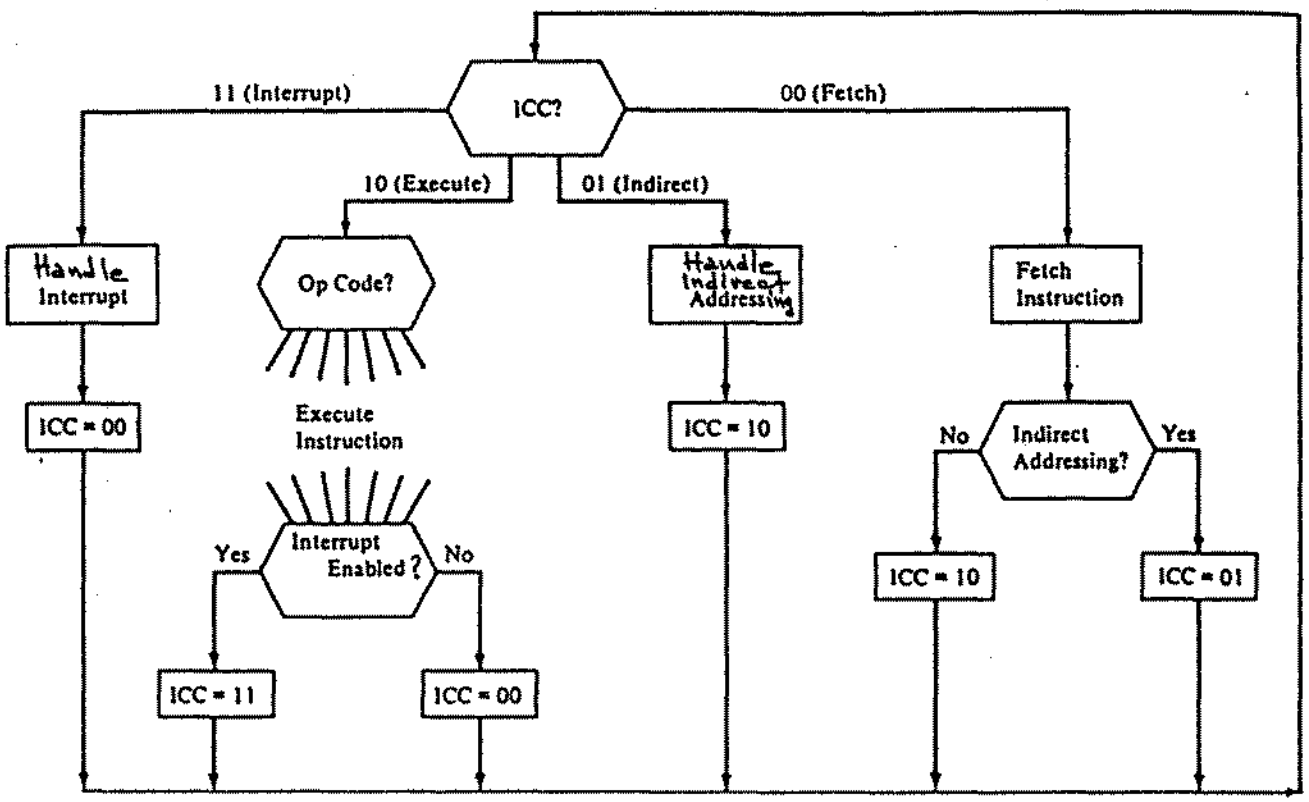


machine instruction.

```
/ Assembly Language          OPCODE\DATA OPERAND's ADDRESS/
/                            MSB LSB     MSB LSB/
       JMP [1]          /    $00 $15     $00 $08
         X              /    $14
         Y              /    $09
         Min            /    $00
     unused ($0007)     /    $00
 [ 1]  LDA X            /    $00 $91     $00 $04
       CP2              /    $00 $60
       ADD Y            /    $00 $99     $00 $05
       JN  [2]          /    $00 $19     $00 $1A
       JMP [3]          /    $00 $15     $00 $26
 [ 2]  LDA Y            /    $00 $91     $00 $05
       STA Min          /    $00 $39     $00 $06
       JMP [4]          /    $00 $15     $00 $2E
 [ 3]  LDA X            /    $00 $91     $00 $04
       STA Min          /    $00 $39     $00 $06
 [ 4]  DSP Min          /    $00 $01     $00 $06
       HLT              /    $00 $64
```

Instruction Cycle

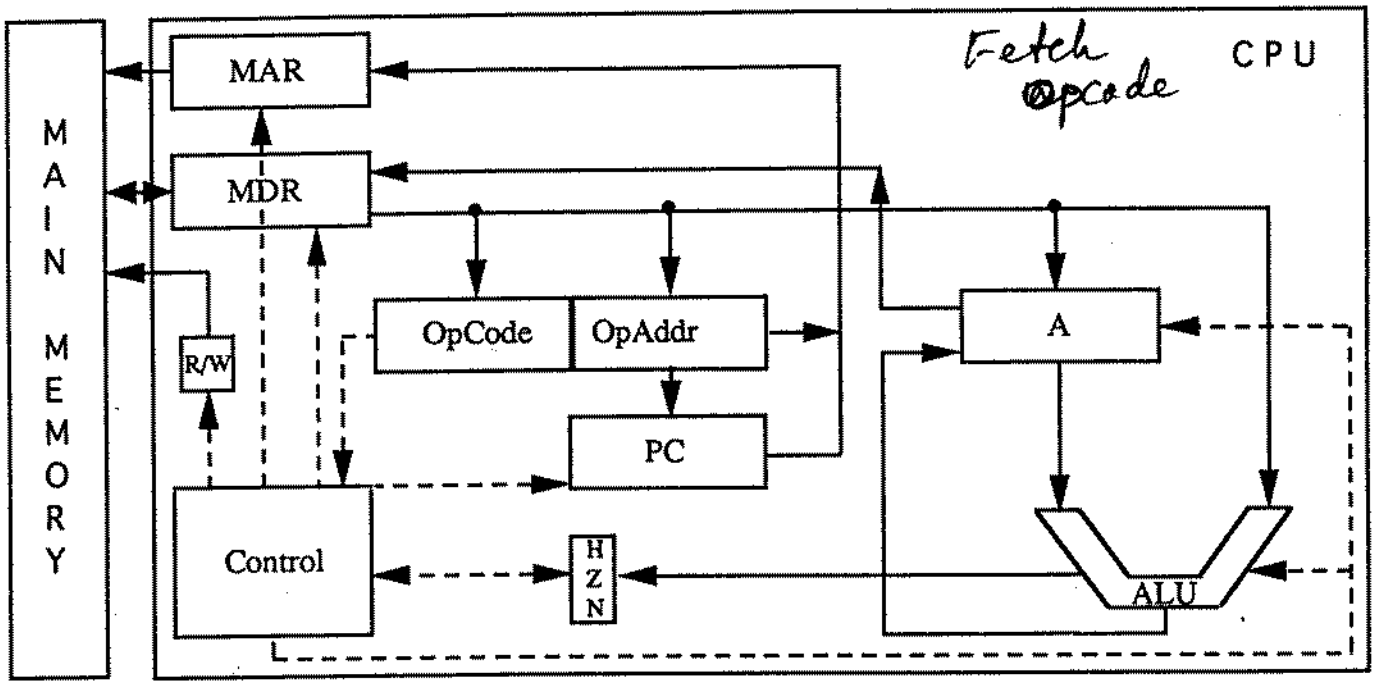
consists of fetch and execute.



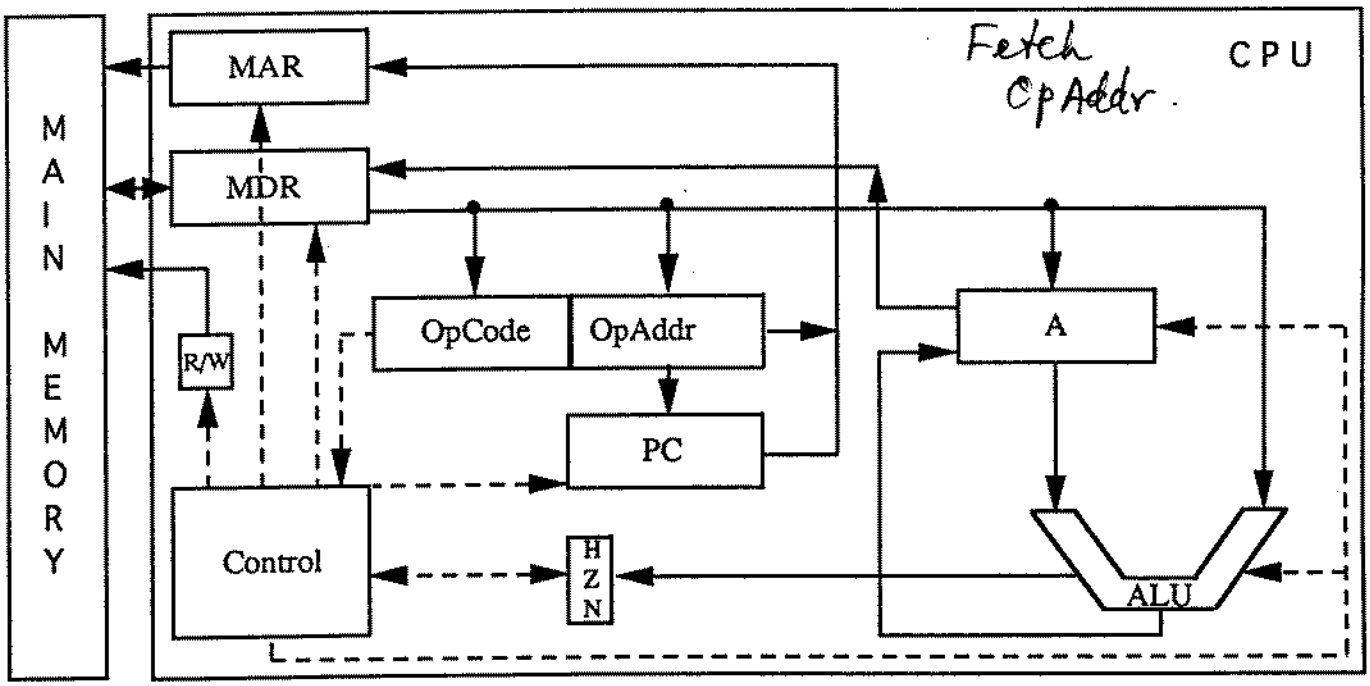
Flowchart for instruction cycle

RTL (Register Transfer Language)

←	Transfers a copy of the right hand side to the left hand side (right hand side replaces the left hand side)
comma	Separates simultaneous transfers
M[ ]	Specifies the memory location whose address is given within [ ]
R( )	Specifies the part of a register R which is identified within ( )

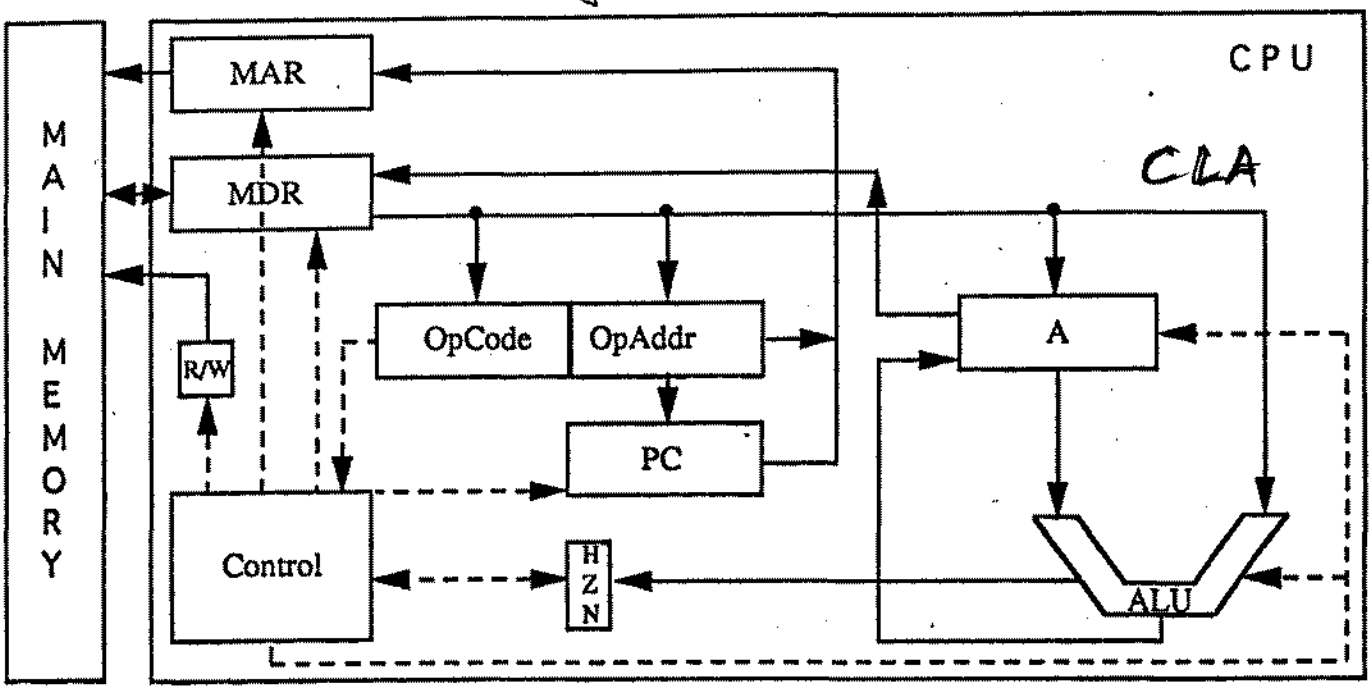


F_1 MAR ← PC
F_2 R/W ← True
F_3 MDR ← M[MAR]
F_4 PC ← PC + 2
OpCode ← MDR

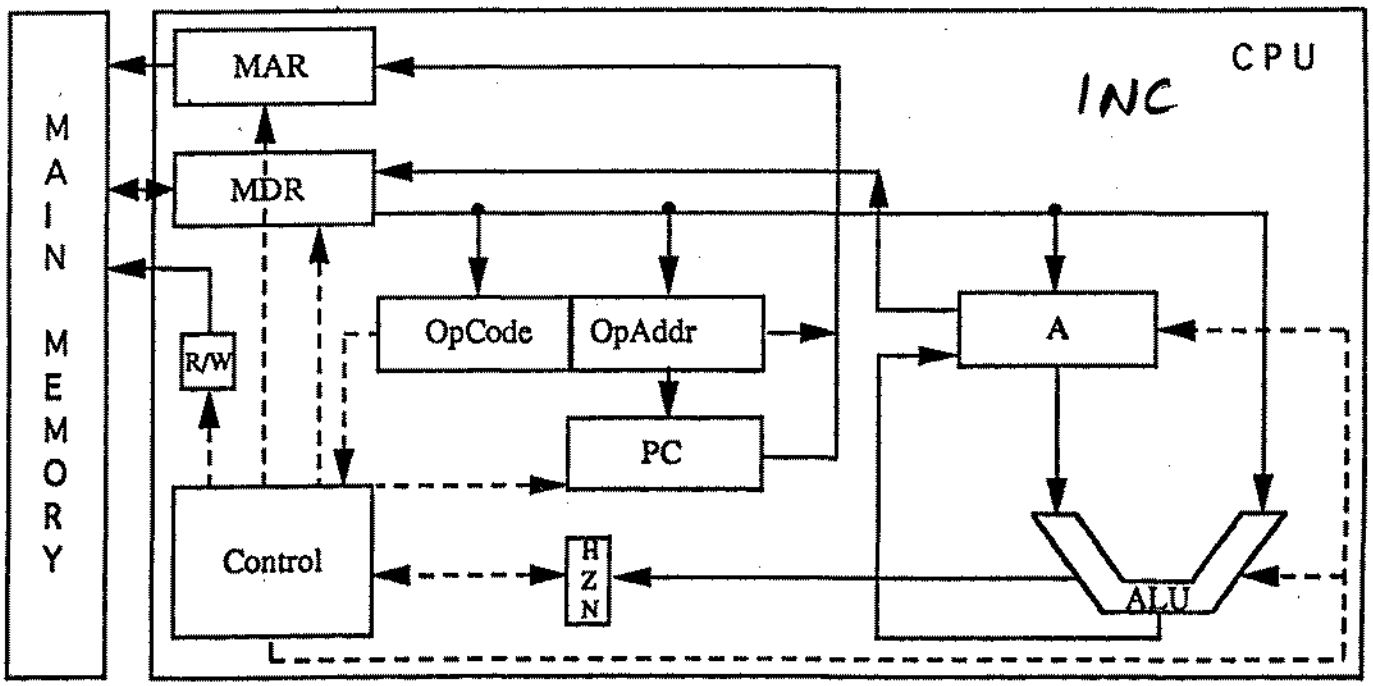


F_1
F_2
F_3
F_4
OpAddr ← MDR

Execute Cycle



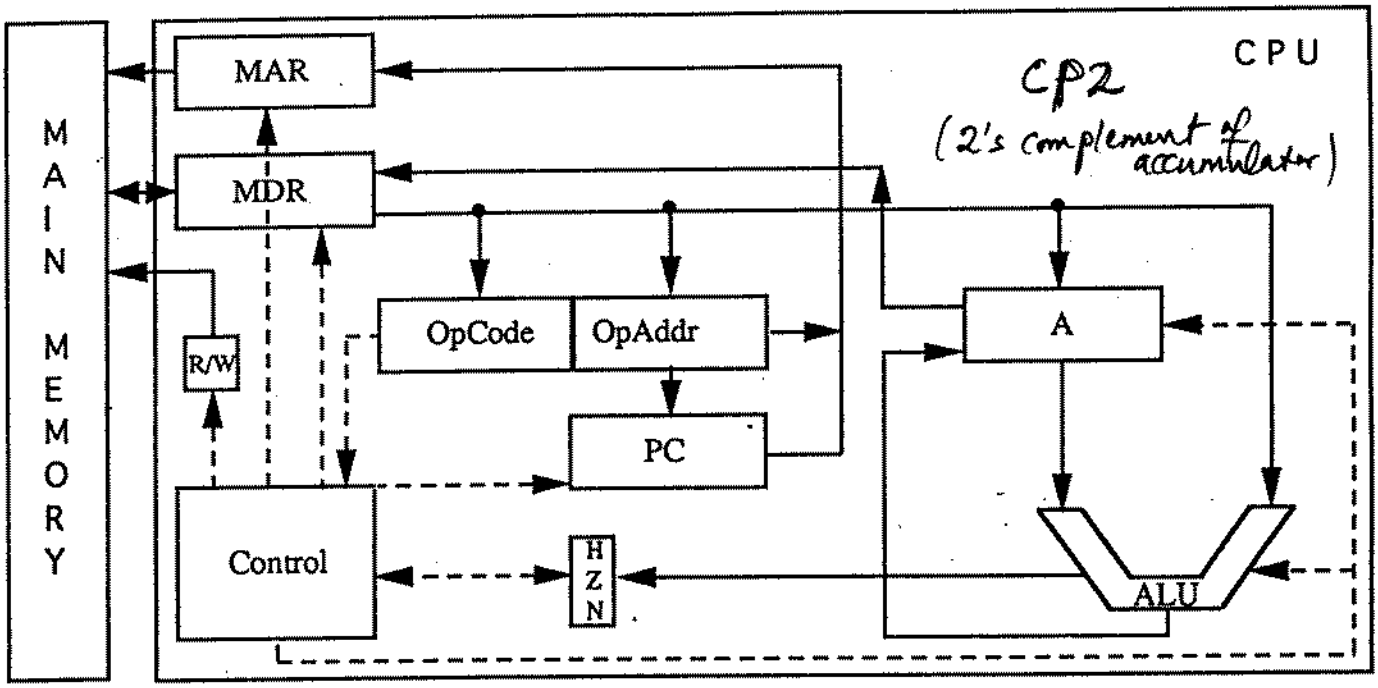
A ← 0
Z ← True
N ← False



A ← A + 1

c_1 Z ← True if A = 0

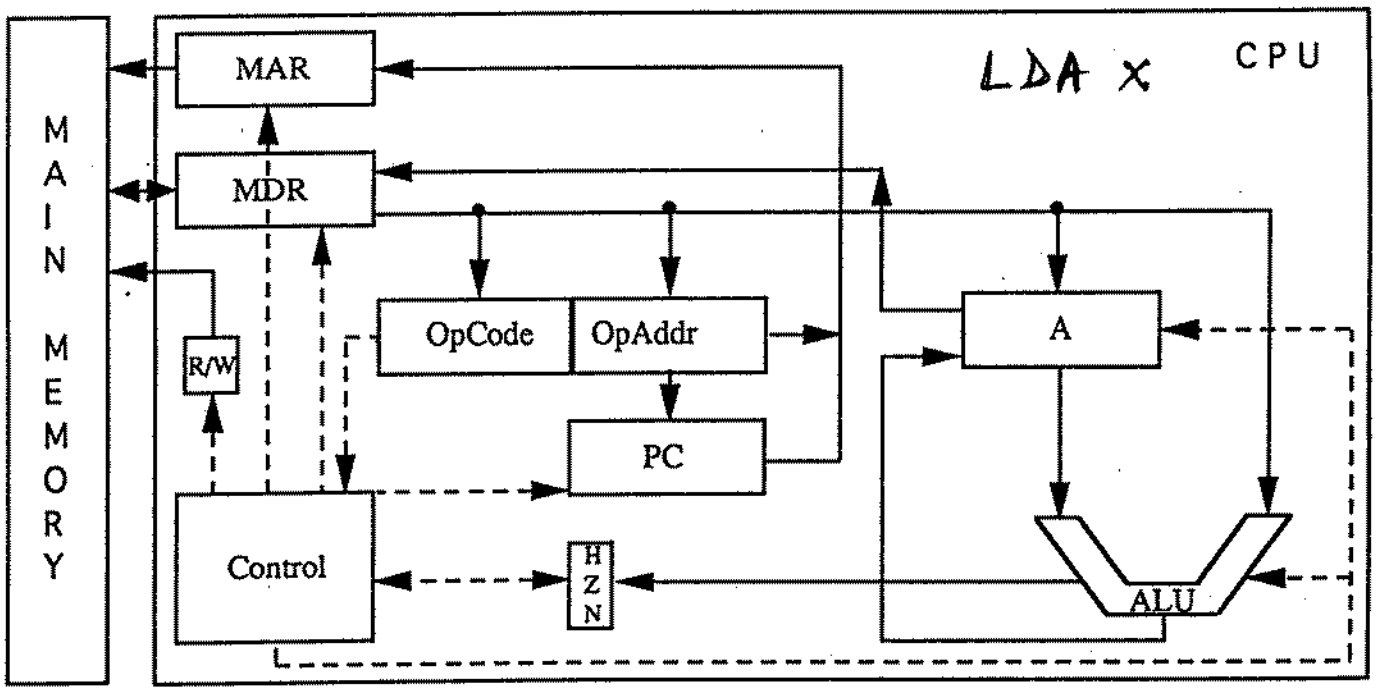
c_2 N ← True if A < 0



CP2
(2's complement of accumulator)

$A \leftarrow (2^{16}-1) \oplus A$

$A \leftarrow A+1$



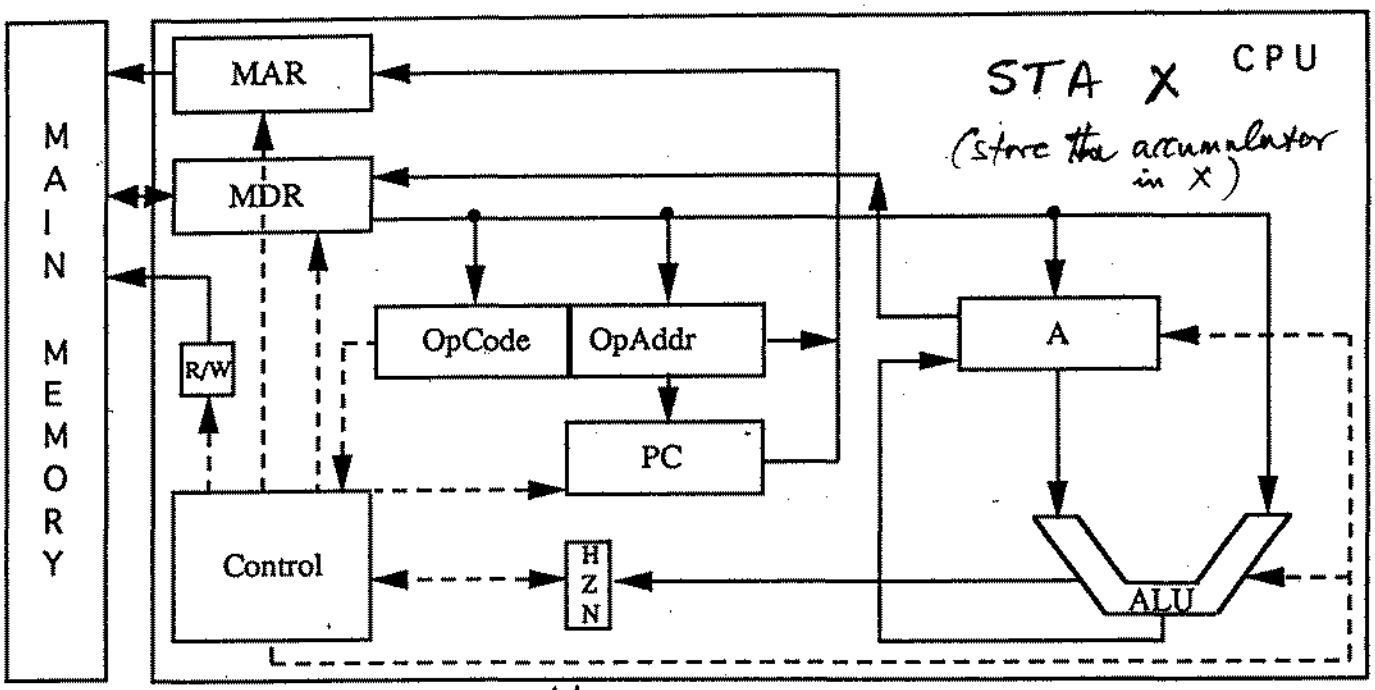
LDA x

G_1 MAR ← OpAddr

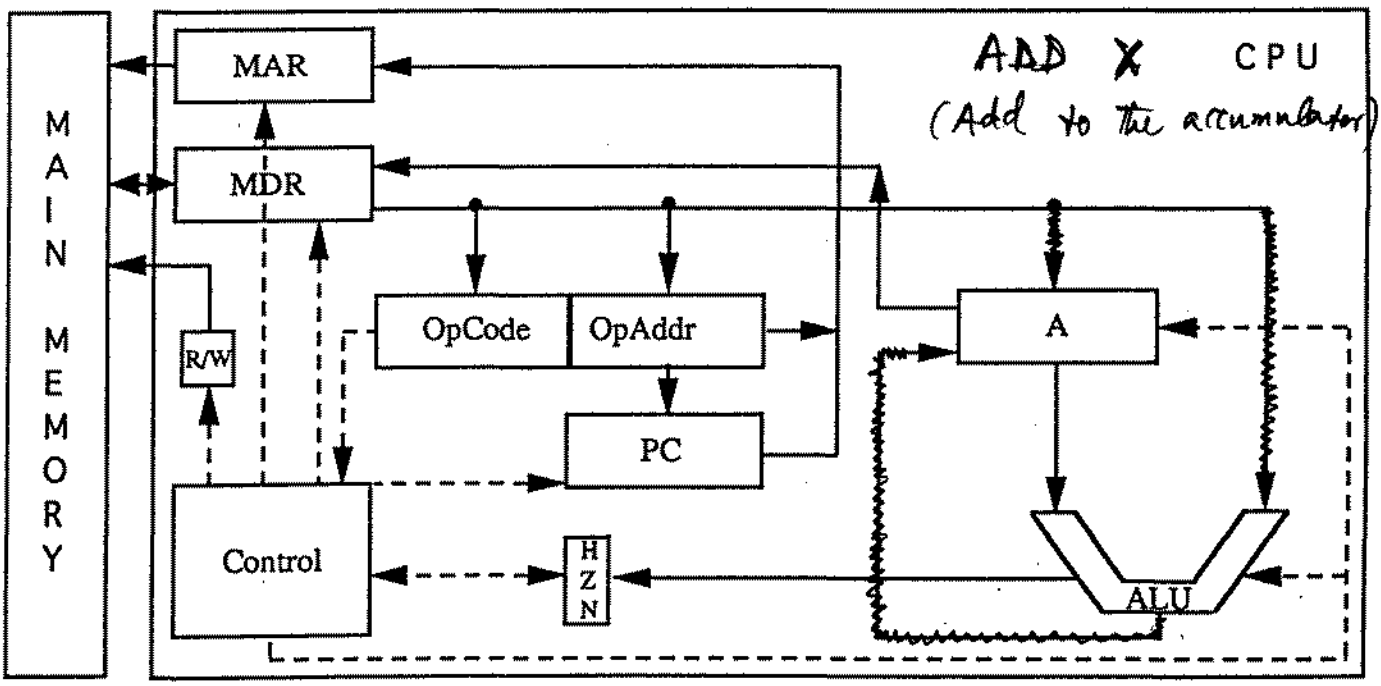
G_2 R/W ← True

G_3 MDR ← M[MAR]

A ← MDR

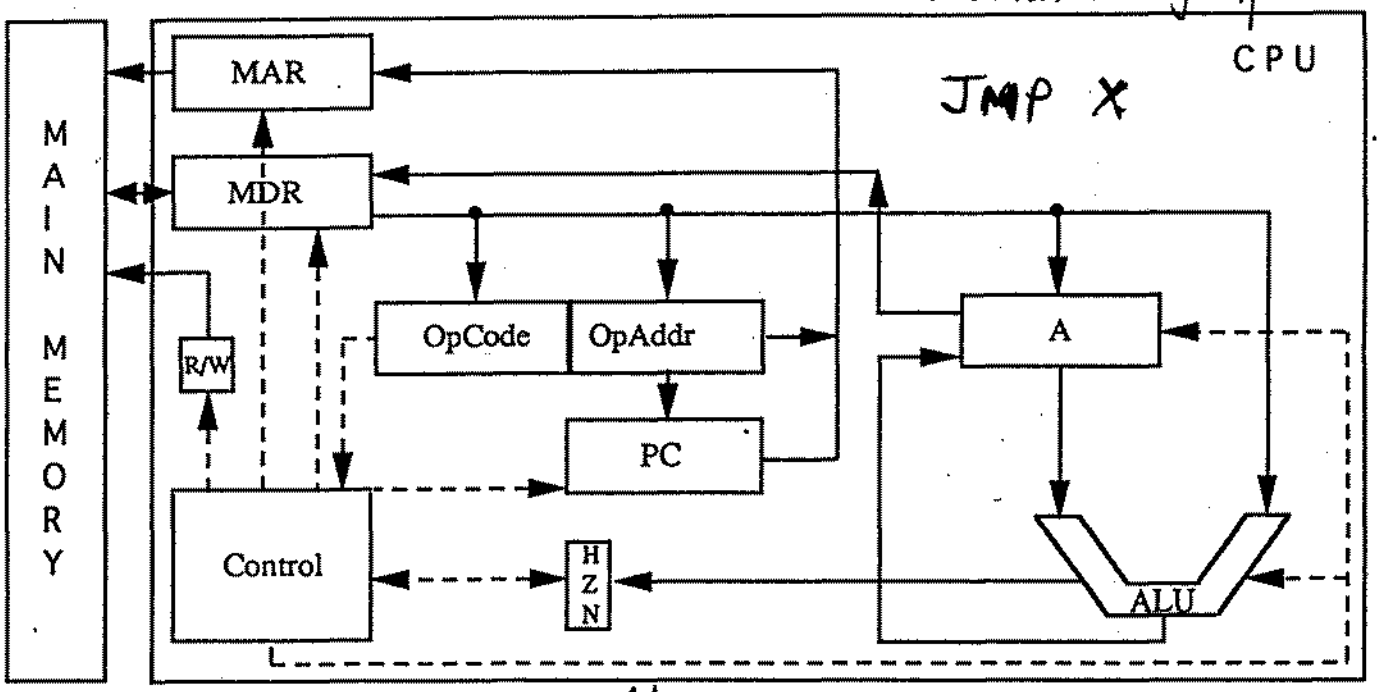


MAR ← OpAddr
MDR ← A
R/W ← False
M[MAR] ← MDR

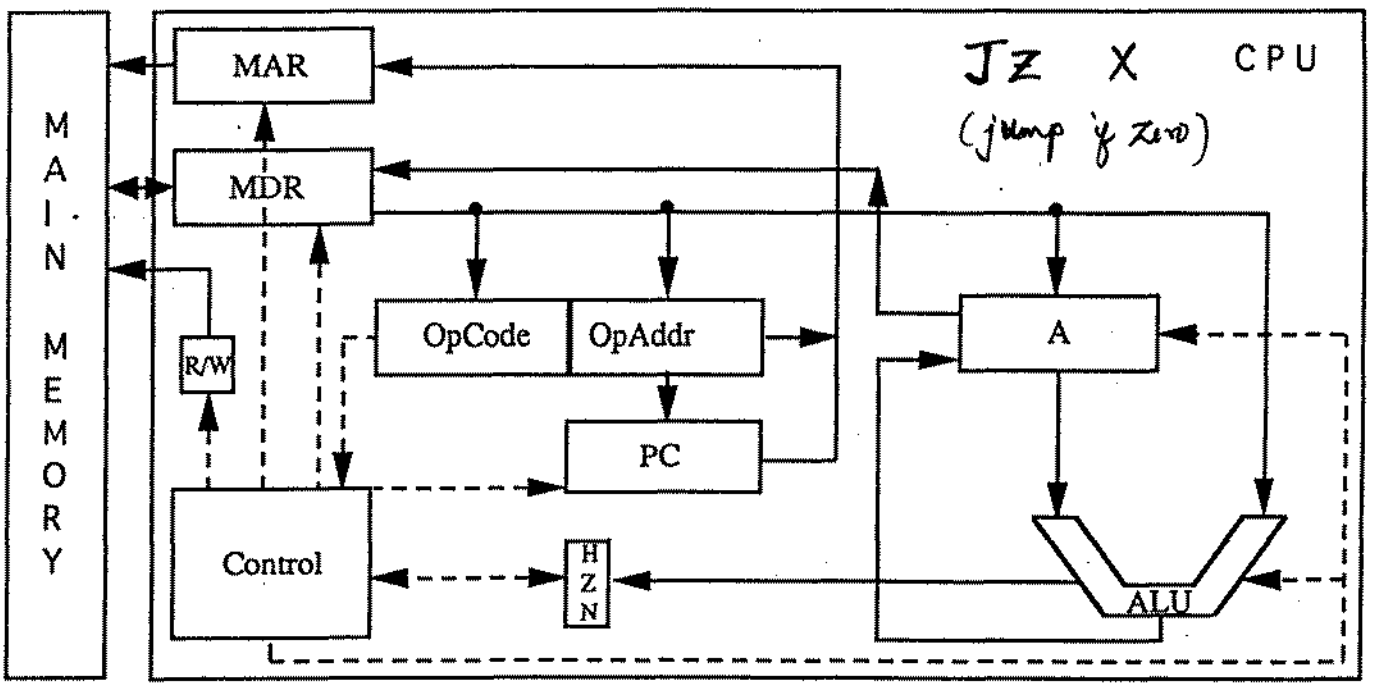


G_1 MAR ← OpAddr
G_2 R/W ← True
G_3 MDR ← M[MAR]
A ← A + MDR
C_1 Z ← True if A = 0
C_2 N ← True if A < 0

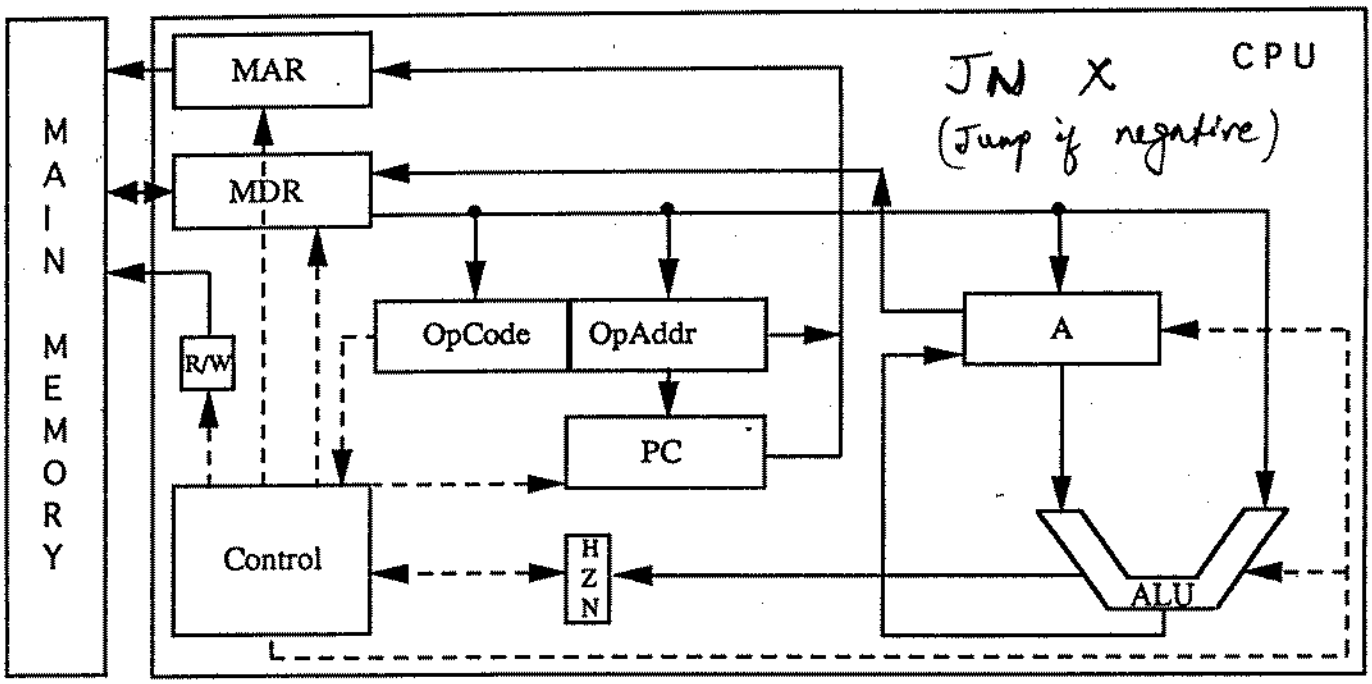
unconditional jump



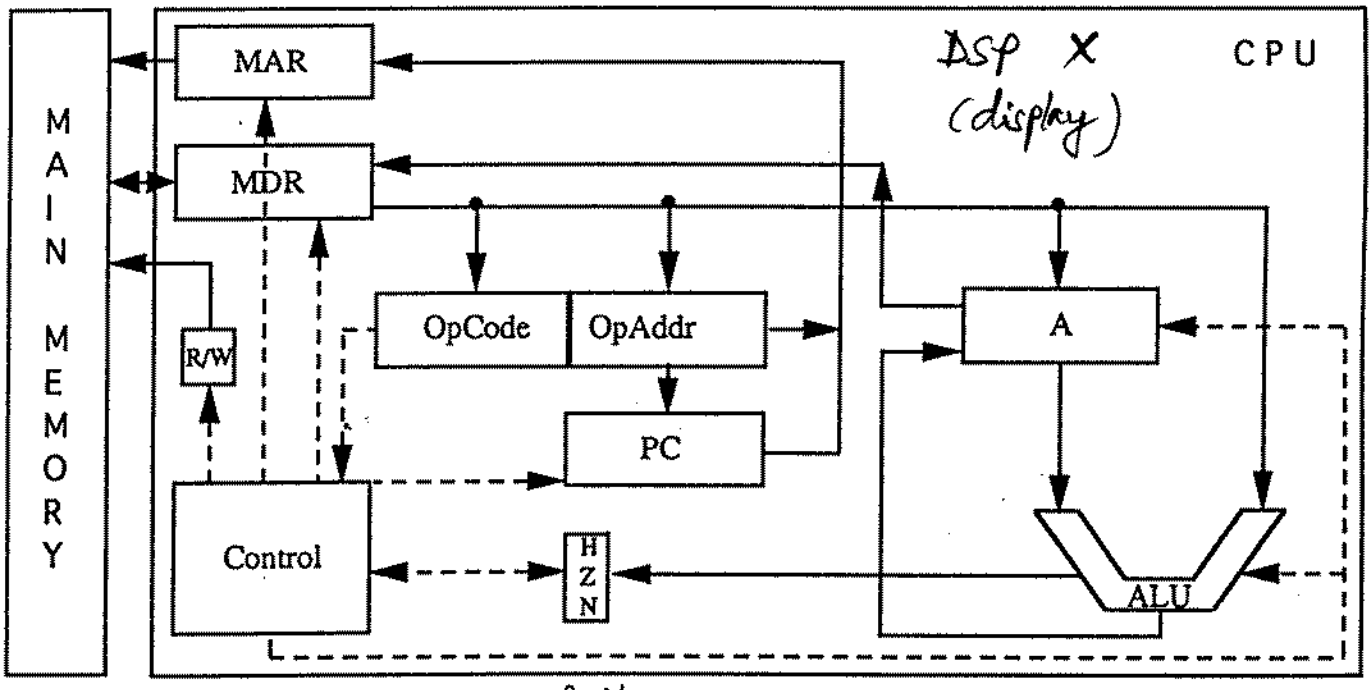
PC ← OpAddr



PC ← OpAddr if Z = true.



PC ← OpAddr if N = TRUE

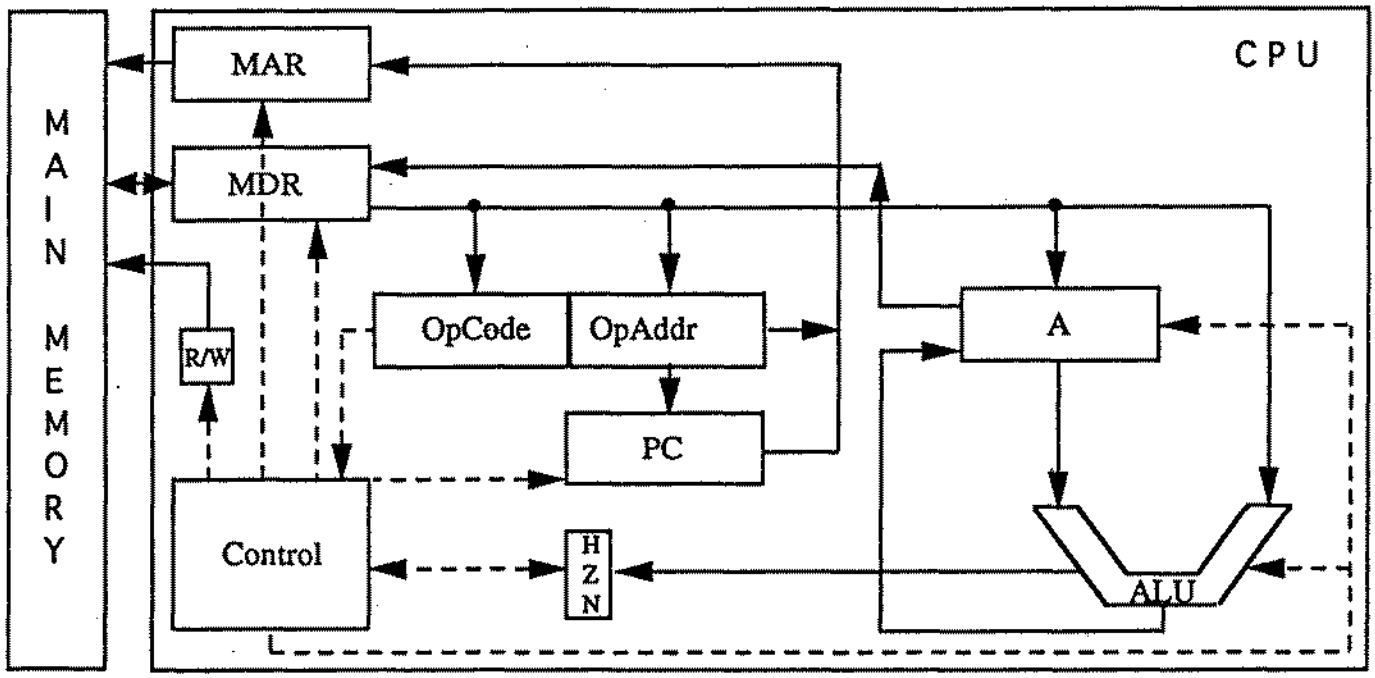


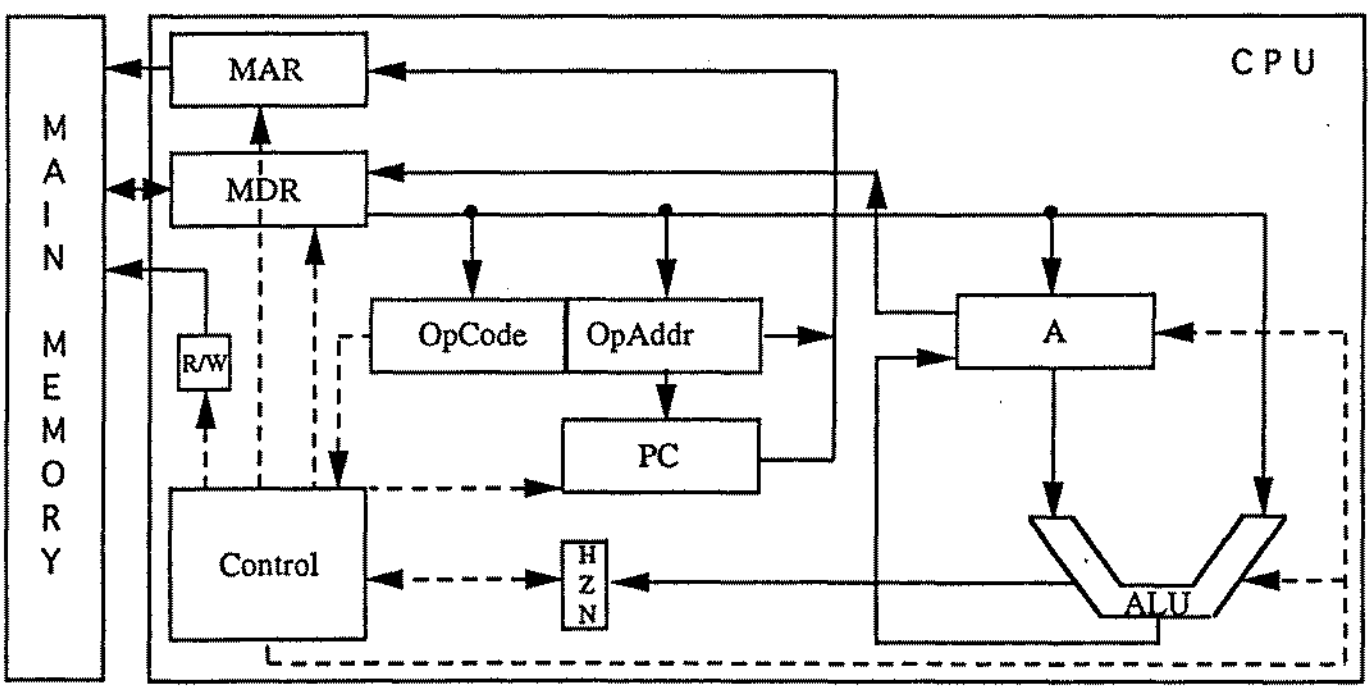
G_1 MAR ← OpAddr

G_2 R/W ← True

G_3 MDR ← M[MAR]

I/O ← MDR


MAIN MEMORY
CPU
MAR
MDR
R/W
OpCode
OpAddr
PC
A
Control
H
Z
N
ALU


MAIN MEMORY
CPU
MAR
MDR
R/W
OpCode
OpAddr
PC
A
Control
H
Z
N
ALU