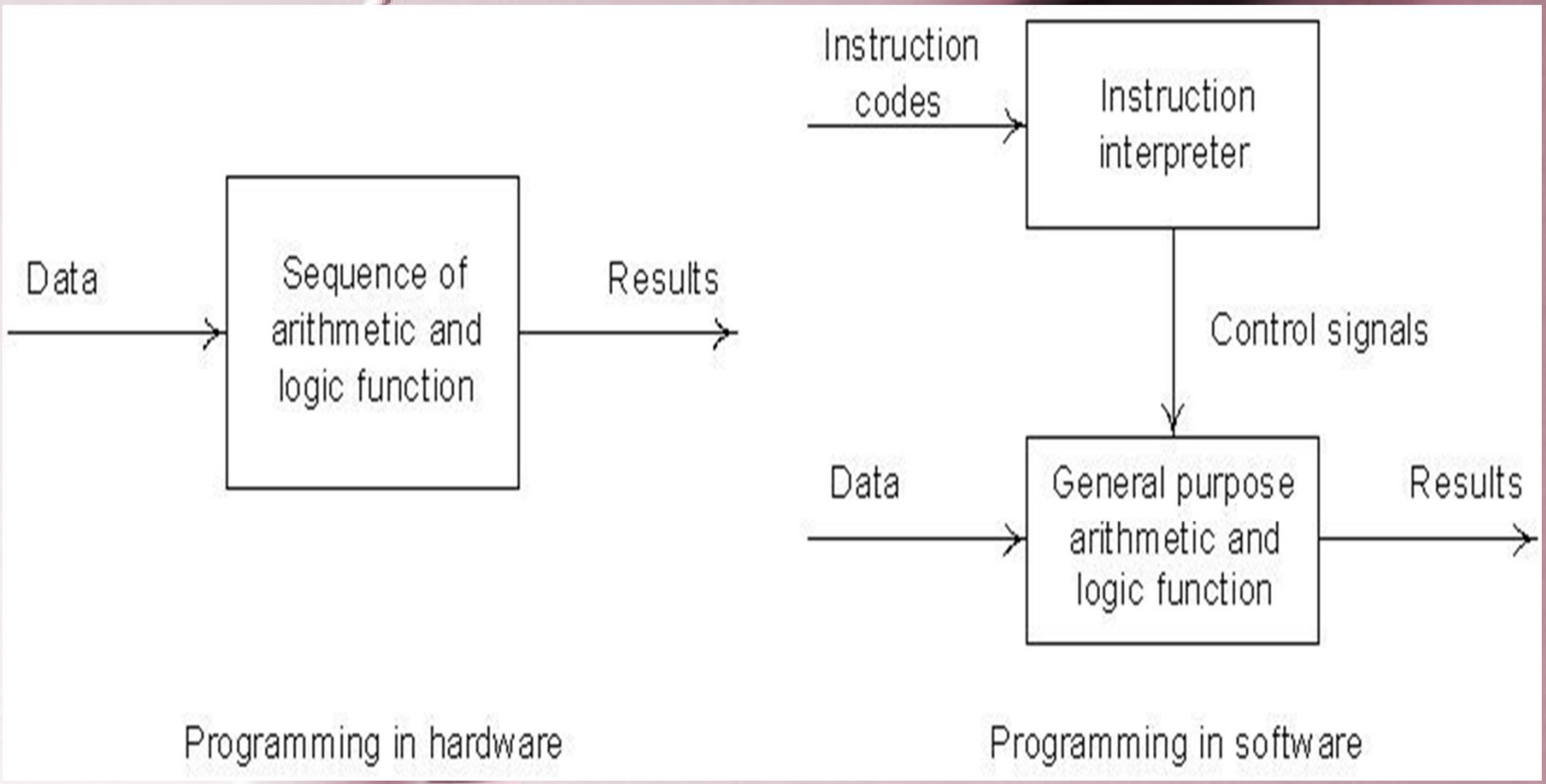

Pertemuan
III
Mode Pengalamatan

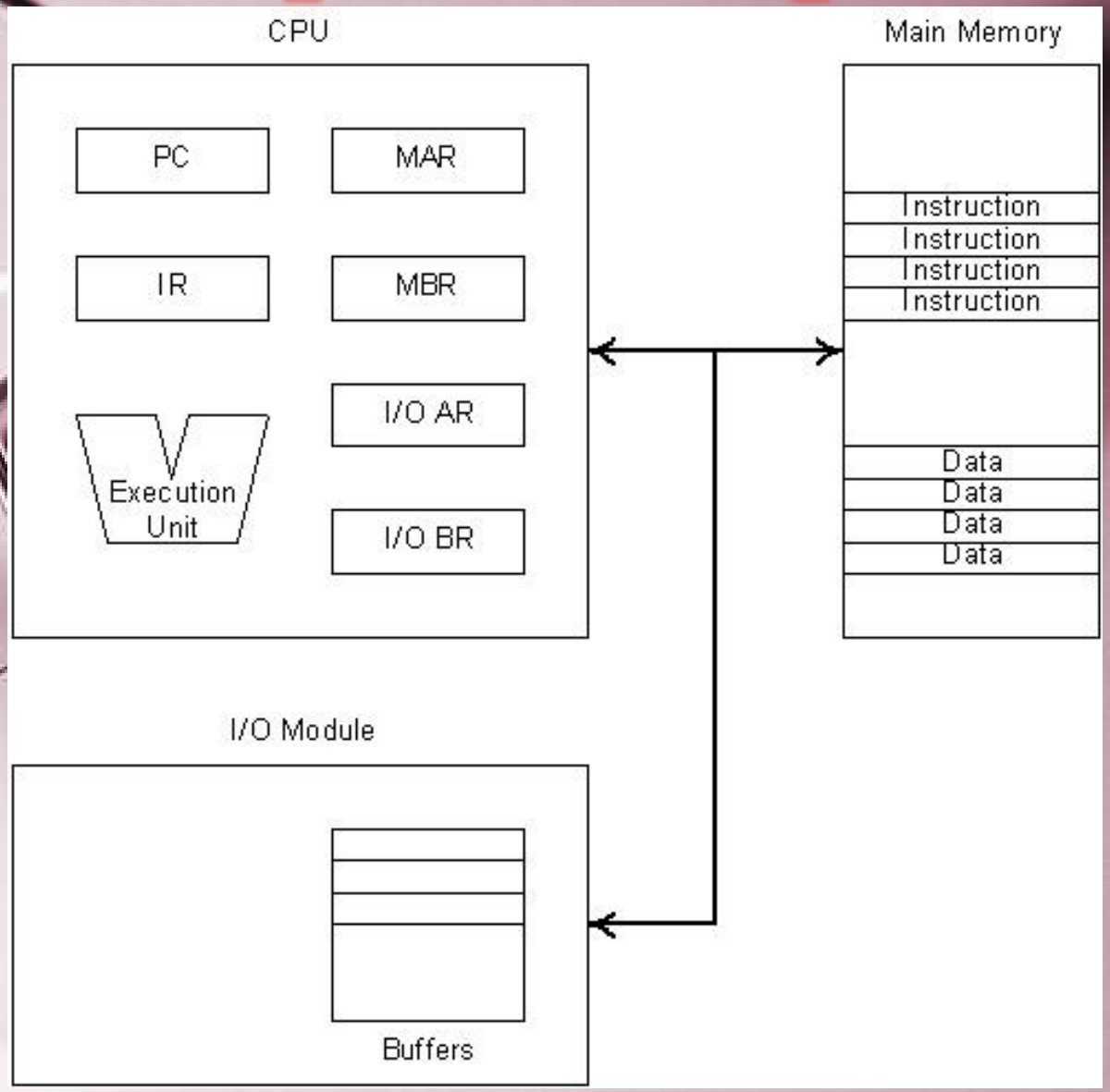
Konsep Arsitektur von Neumann

- Data dan instruksi disimpan dalam satu memori
- Isi dari memori ini dapat dialamatkan dengan lokasi tanpa memperhatikan tipe datanya
- Eksekusi terjadi secara sekuensial dari satu instruksi ke instruksi selanjutnya

Hardware and Software Approach



Komponen Komputer



Tipe-Tipe Instuksi

- Pada bahasa pemrograman tingkat tinggi, misal C, contoh : x = x +y;
- Statemen tersebut memberi instruksi kepada komputer untuk menambah nilai yang tersimpan di x dengan nilai yang tersimpan di y dan meletakkan hasilnya di x

Tipe-Tipe Instuksi

- Pada bahasa mesin, operasi tersebut membutuhkan tiga instruksi (misalnya variabel x dan y terletak di lokasi 513 dan 514) :
  - Load register berisi lokasi memori 513
  - Tambahkan isi lokasi memori 513 ke register
  - Simpan isi register di lokasi memori 513
- Bahasa pemrograman tingkat tinggi mengekspresikan operasi dalam bentuk aljabar ringkas, menggunakan variabel
- Bahasa mesin mengekspresikan operasi dalam bentuk dasar melibatkan perpindahan data dari dan ke register

Tipe-Tipe Instuksi

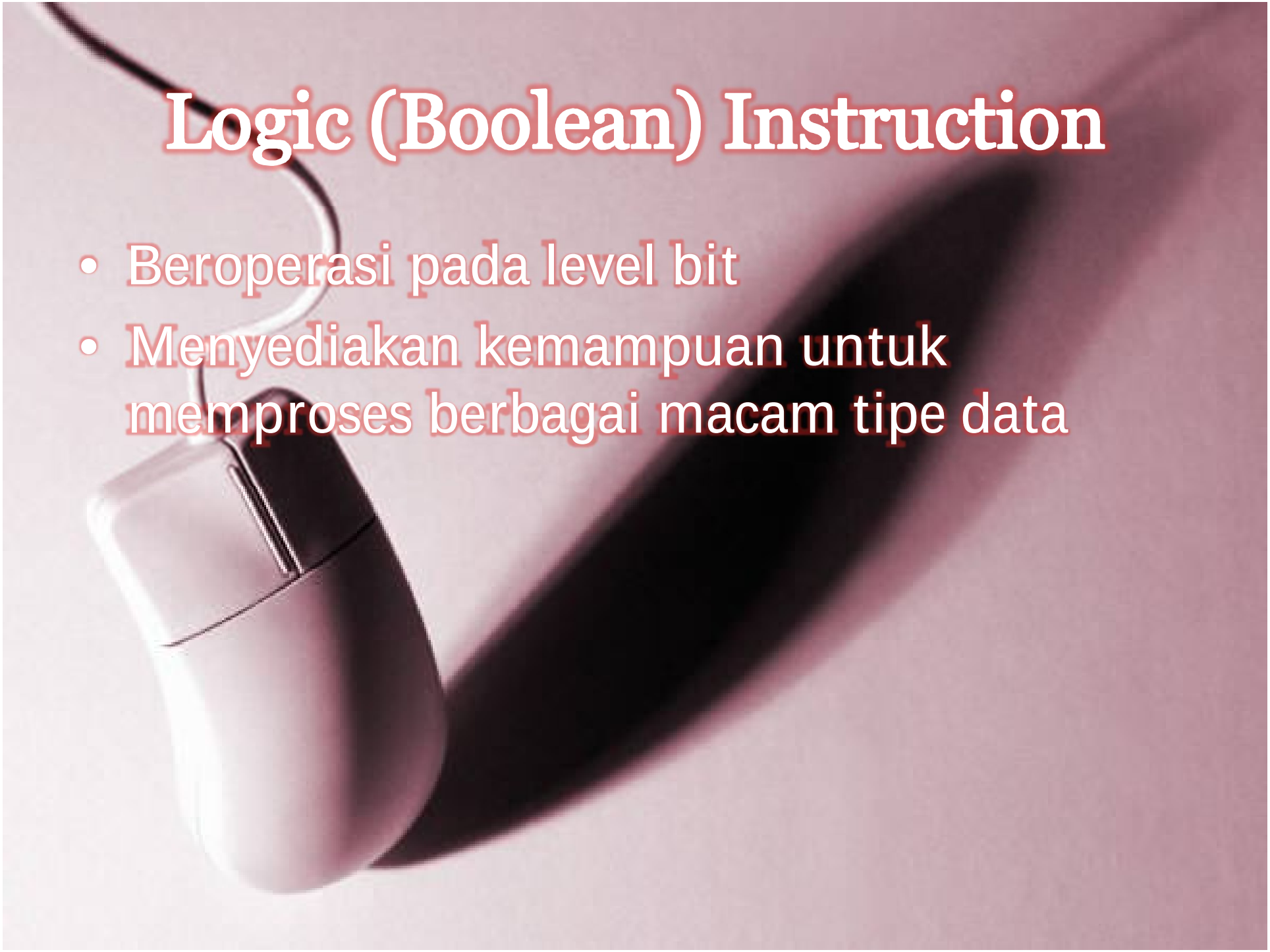
- Komputer harus memiliki suatu set instruksi supaya user dapat memformulasikan pemrosesan data
- Program yang ditulis dalam bahasa pemrograman tingkat tinggi harus diterjemahkan ke bahasa mesin untuk dijalankan/dieksekusi
- Jadi, set instruksi harus mencukupi untuk menjalankan instruksi dari bahasa tingkat tinggi

Tipe-Tipe Instuksi

- Instruksi Aritmatika dan Logika (Arithmetic and Logic Instruction)
  - Data Processing, contoh : ADD, SUB, MPY, DIV, OR, AND
- Instruksi Memory (Memory Instruction)
  - Data Storage, contoh : LOAD, STOR
- Instruksi I/O (I/O Instruction)
  - Data Movement
- Instruksi Test dan Branch (Test And Brach Instruction)
  - Control

Arithmetic Instruction

- Menyediakan kemampuan komputasional untuk pemrosesan data numerik



Logic (Boolean) Instruction

- Beroperasi pada level bit
- Menyediakan kemampuan untuk memproses berbagai macam tipe data

Memory Instruction

- Perpindahan data antara memori dan register

I/O Instruction

- Transfer program dan data ke memori dan hasil perhitungan ke user

Test Instruction

- Digunakan untuk menguji nilai data atau status dari perhitungan

Branch Instruction

- Digunakan untuk percabangan pada set instruksi yang lain tergantung dari keputusan yang dibuat

Common Instruction Set Operations

Type	Operation Name	Description
Data transfer	Move (transfer)	Transfer word or block from source to destination
	Store	Transfer word from processor to memory
	Load (fetch)	Transfer word from memory to processor
	Exchange	Swap contents of source and destination
	Clear (reset)	Transfer word of 0s to destination
	Set	Transfer word of 1s to destination
	Push	Transfer word from source to top of stack
	Pop	Transfer word from top of stack to destination
Arithmetic	Add	Compute sum of two operands
	Subtract	Compute difference of two operands
	Multiply	Compute product of two operands
	Divide	Compute quotient of two operands
	Absolute	Replace operand by its absolute value
	Negate	Change sign of operand
	Increment	Add 1 to operand
	Decrement	Subtract 1 from operand

Common Instruction Set Operations

Logical	AND OR NOT (Complement) Exclusive-OR	Perform the specified logical operation bitwise
	Test	Test specified condition; set flag(s) based on outcome
	Compare	Make logical or arithmetic comparison of two or more operands; set flag(s) based on outcome
	Set control variables	Class of instructions to set controls for protection purposes, interrupt handling, timer control, etc.
	Shift	Left (right) shift operand, introducing constants at end
	Rotate	Left (right) shift operand, with wraparound end
Transfer of control	Jump (branch)	Unconditional transfer; load PC with specified address
	Jump conditional	Test specified condition; either load PC with specified address or do nothing, based on condition
	Jump to subroutine	Place current program control information in known location; jump to specified address
	Return	Replace contents of PC and other register from known location
	Execute	Fetch operand from specified location and execute as instruction; do not modify PC
	Skip	Increment PC to skip next instruction
	Skip conditional	Test specified condition; either skip or do nothing based on condition
	Halt	Stop program execution
	Wait (hold)	Stop program execution; test specified condition repeatedly; resume execution when condition is satisfied
	No operation	No operation is performed, but program execution is continued

Common Instruction Set Operations

Input/output	Input (read)	Transfer data from specified I/O port or device to destination (e.g., main memory or processor register)
	Output (write)	Transfer data from specified source to I/O port or device
	Start I/O	Transfer instructions to I/O processor to initiate I/O operation
	Test I/O	Transfer status information from I/O system to specified destination
Conversion	Translate	Translate values in a section of memory based on a table of correspondences
	Convert	Convert the contents of a word from one form to another (e.g., packed decimal to binary)

Aksi Yang Dilakukan CPU Terhadap Berbagai Operasi

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

Jenis Mode Pengalamatan

- Direct
  - Alamat operand ditunjuk secara langsung pada instruksi
  - Contoh : instruksi LOAD, cara penulisan : LOAD Y
- Indirect
  - Alamat operand ditunjukkan secara tidak langsung oleh data yang terkandung pada alamat yang ditunjuk
  - Cara penulisan : LOAD (Y)
- Displacement
  - Merupakan alamat relatif, artinya alamat operand yang dituju berjarak n alamat dari sebuah alamat yang diekspresikan secara indirect
  - Cara penulisan : LOAD (Y) + 100
- Immediate
  - Alamat operand tidak berisi sebuah alamat, tetapi langsung operand yang akan diproses
  - Cara penulisan : LOAD #9

Penerapan Mode Pengalamatan

Misalkan kita memiliki contoh register dan memory sebagai berikut :

Register	
A	
B	
C	50
D	51
...	
Y	
Z	
100	5
101	6
102	15
...	

Memory	
0	52
1	1
2	12
3	43
...	
50	3
51	7
52	10
...	
100	5
101	6
102	15

Direct

- Untuk pemanggilan alamat operand berupa register disebut Register Addressing
  - Contoh : LOAD C, akan menghasilkan

    ACC ← 50, karena alamat register C berisi 50
- Untuk pemanggilan alamat operand berupa memory disebut Direct Addressing
  - Contoh : LOAD 3, akan menghasilkan

    ACC ← 43, karena alamat memory 3 berisi bilangan 43

Indirect

- Untuk pemanggilan alamat operand berupa register namanya Register Indirect Addressing
  - Contoh : LOAD (D), akan menghasilkan ACC ← 7, karena alamat register D berisi alamat memory 51, sedangkan alamat memory 51 berisi bilangan 7
- Untuk pemanggilan alamat operand berupa memory namanya Indirect Addressing
  - Contoh : LOAD (50), akan menghasilkan ACC ← 43, karena alamat memory 50 berisi alamat memory 3, sedangkan alamat memory 3 berisi bilangan 43

Displacement

- Merupakan alamat relatif, artinya alamat operand yang dituju berjarak n alamat dari sebuah alamat yang diekspresikan secara indirect
- Contoh : LOAD (C) + 50

  akan menghasilkan ACC ← 5, karena alamat register C berisi alamat 50, sedangkan alamat yang dituju berjarak +50 darinya. 50 + 50 = 100. Alamat 100 berisi bilangan 5

Immediate

- Immediate tidak membutuhkan alamat memory/register, karena tempat untuk alamat operand diisi langsung oleh bilangan operand-nya
- Contoh 1 : LOAD #9, akan menghasilkan ACC ← 9, nilai di belakang tanda # dianggap sebagai operand-nya
- Contoh 2 : ADD Y, #2, #3, akan menghasilkan register Y ← 5, kedua operand adalah 2 dan 3, dengan demikian 2+3 = 5

Contoh Soal

- Berapakah isi register Y?
- Gunakan kondisi register dan memori yang ada!

LOAD (C)

ADD 3

SUB (0)

MPY (C) + 50

DIV #9

STOR Y

Contoh Soal

- Berapakah isi register Y?
- Gunakan kondisi register dan memori yang ada!

INSTRUKSI	ALGORITMIK	ISI ACC
LOAD (C)	AC ← 3	3
ADD 3	AC ← AC + 43	46
SUB (0)	AC ← AC - 10	36
MPY (C) + 50	AC ← AC X 5	180
DIV #9	AC ← AC/9	20
STOR Y	Y ← 20	

Register Y berisi **20**

Latihan

Berapakah Nilai Y Dari Tabel Dibawah ini, Jika Diketahui Soal Sebagai Berikut

LOAD (C)
ADD 2
SUB (2)
ADD (0)
SUB 0
MPY (C) + 50
ADD #10
STOR Y

Register	
A	
B	
C	50
D	51
...	
Y	
Z	
100	54
101	66
102	151
...	

Memory	
0	51
1	1
2	3
3	102
...	
50	2
51	75
52	107
...	
100	54
101	66
102	151