Arsitektur dan Organisasi Komputer:
Mode Pengalamatan & Format

Ir. Heru Nurwarsito, M.Kom
Barlian Henryranu P, ST, MT
Eko Saksi Pramukantoro, S.Kom, M.Kom

JOIN UB BE THE BEST
BRAWIJAYA UNIVERSITY

11

SELF-PROPAGATING ENTREPRENEURIAL EDUCATION DEVELOPMENT

1. PENDAHULUAN

1.1 Pengantar

Dalam Bab ini, hal yang menjadi fokus pembahasan adalah apa yang dilakukan set instruksi. Dan membahas jenis-jenis operand dan operasi yang dispesifikasikan oleh instruksi mesin. Selain itu, pada bab ini juga membahas bagaimana cara menspesifikasikan operand dan operasi intruksi. Disini, terdapat dua buah masalah. Pertama, tentang cara menspesifikasikan alamat sebuah operand, dan kedua, tentang pengaturan bit-bit instruksi dalam menentukan alamat operand dan operasi intruksi tersebut.

1.2 Tujuan

Tujuan dari pembahasan bab ini adalah

- Mengetahui mode pengalamatan
- Mengetahui jenis-jenis operand
- Mengetahui cara mencari operand saat eksekusi opcode
- Implementasi mode pengalamatan yang digunakan dalam format istruksi PDP 8, PDP 10,dan PDP 11
- Mengetahui fungsi complier assembler
- Membuat symbolic program dan assembler program

1.3 Latar Belakang

Seperti yang telah kita ketahui, field alamat dalam format instruksi sangat tebatas. Kita ingin dapat mereferensikan lokasi dalam jumlah yang banyak dalam memori utama atau dalam memori virtual, pada mesin-mesin tertentu. Untuk mecapai keinginan itu telah ditemukan bermacam-macam teknik pengalamatan. Teknik-teknik ini memiliki untung-rugi yang terjadi antara range alamat dan / atau fleksibilitas pengalamatan disatu sisi, dengan jumlah referensi memori dan / atau kompleksitas kalkulasi alamat pada sisi lainnya.

Set Instruksi adalah kumpulan dari instruksi-instruksi yang berbeda yang dapat dijalankan oleh CPU disebut set instruksi (*Instruction Set*).

2. Mode Pengalamatan

Suatu variasi mode pengalamatan (addressing mode) dapatdigunakan untuk menentukan suatu alamat tempat untuk dimanaoperand akan di fetch. Beberapa teknik ini dapat meningkatkankecepatan pelaksanaan instruksi dengan menurunkan jumlahreferensi pada memori utama dan meningkatkan jumlah referensipada register kecepatan tinggi. Mode pengalamatan ini menjabarkansuatu aturan untuk menginterpretasikan atau memodifikasi fieldalamat dari instruksi sebelum operand direferensikan.

Teknik-teknik pengalamatan yang umum digunakan :

1) Immediate addressing
2) Direct addressing
3) Indirect addressing
4) Register addressing
5) Register indirect addressing
6) Displacement addressing
7) Stack addressing

1. Immediate addressing

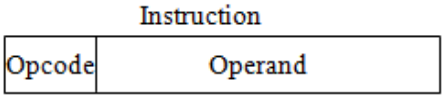
Bentuk pengalamatan yang paling sederhana, di mana operand langsung ada pada instruksi Format intruksi.

Bentuk pengalamatan ini yang paling sederhana :

- Operand benar-benar ada dalam instruksi atau bagian dari instruksi (operand sama dengan field alamat).
- Umumnya bilangan akan di simpan dalam bentuk komplemen dua.
- Bit paling kiri sebagai bit tanda.
- Ketika operand dimuatkan ke dalam register data, bit tanda akan di geser ke kiri hingga maksimum word data.
- Operand = address field
- Contoh : ADD 5

- Tambah nilai 5 ke isi akumulator
- 5 adalah operand

- Diagram *Immediate addressing*



Isi address field langsung berupa **operand**

- ***Keuntungan :***
  - Tidak ada referensi memori selain dari instruksi yang diperlukan untuk memperoleh operand.
  - Menghemat siklus instruksi sehingga proses keseluruhan akan cepat.
- ***Kerugian:***
  - Ukuran bilangan (operand) dibatasi oleh ukuran address field.

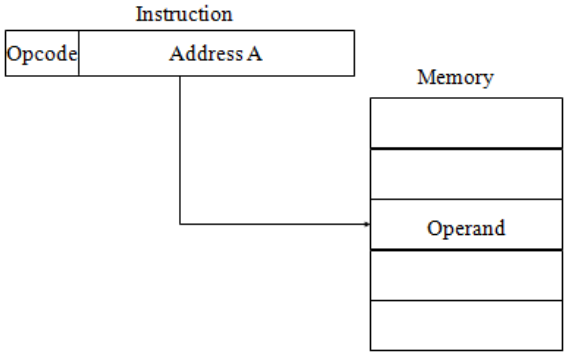
2. <u>Direct Adressing</u> (Pengalamatan langsung)

- **Kelebihan** :
  - Field alamat berisi efektif address sebuah operand
  - Teknik ini banyak digunakan pada komputer lama dan komputer kecil.
  - Hanya memerlukan sebuah referensi memory dan tidak memerlukan kalkulasi khusus.
- **Kelemahan** :
  - Keterbatasan field alamat karena panjang field alamat biasanya lebih kecil di bandingkan panjang word.
- Adress field berisi alamat dari operand
- Effective address (EA) = address field (A)

- A = Addressing → isi bit suatu field alamat dalam instruksi
- EA = Effective Addressing → alamat aktual sebuah lokasi yang berisi

- Contoh : ADD A
  - Cari di memory pada alamat A untuk operand
  - Tambahkan isi yang ada pada alamat A dengan nilai di register accumulator dan simpan hasilnya di register accumulator.

- Diagram *Direct Addressing*



3. Indirect addressing

Merupakan mode pengalamatan operand dimana area alamat (address field) berisi alamat dari suatu alamat yang akan menunjukkan alamat dari suatu nilai yang akan diproses. Field alamat mengacu pada alamat word didalam memory, yang pada gilirannya akan berisi alamat operand yang panjang.

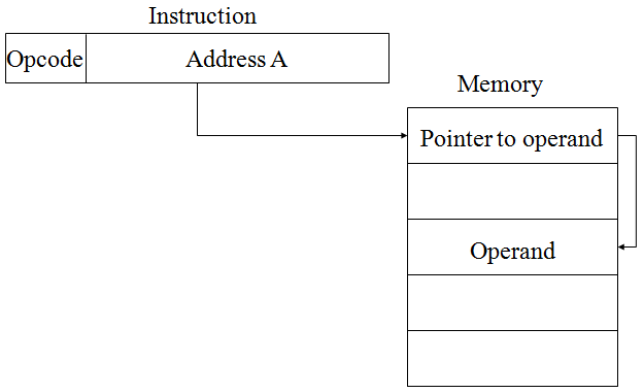
EA = (A)
Keterangan:
Cari di memory alamat A, cari alamat yang tertulis pada A untuk operand
Contoh:
ADD (A)
Keterangan:
Tambahkan isi dari cell yang alamatnya ditunjukkan oleh isi yang terdapat pada A dengan nilai yang ada di register accumulator dan simpan hasilnya di register accumulator.

- Karakteristik:
  - Memerlukan space address yang besar.
  - 2^n dimana n adalah panjang word
  - Dapat dibuat nested (bersarang), multilevel dan cascade (bertumpuk).

Contoh : EA = (((A)))

- Pengaksesan memory yang multiple untuk mendapatkan operand sehingga mengakibatkan proses mode ini agak lebih lambat.

- Diagram *Indirect Addressing*

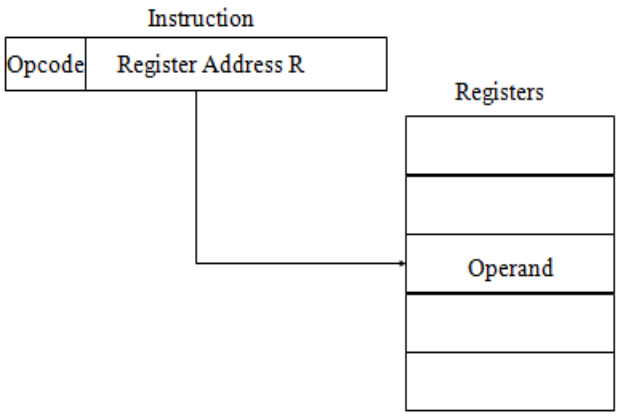


- ***Keuntungan:***
  - Jumlah lokasi memori yang dapat diacu lebih banyak.
  - Hanya membutuhkan satu referensi memori.
  - Tidak memerlukan kalkulasi alamat yang khusus.
  - Ruang bagi alamat menjadi besar sehingga semakin banyak alamat yang dapat referensi
- ***Kerugian:***
  - Memerlukan 2 referensi memori.
  - Jumlah lokasi yang diacu dibatasi oleh ukuran field alamat.
  - Diperlukan referensi memori ganda dalam satu fetch sehingga memperlambat proses operasi

4. Register Addresing

- Operand ada di penamaan register di address field
- EA = R
- Nomor dari register dibatasi
- Alamat yang sangat kecil membutuhkan sedikit instruksi dan instruksi fetch yang cepat
- Tidak ada memori akses
- Eksekusi yang sangat cepat
- Sangat sedikit tempat dari alamat yang disediakan
- Multiple register membutuhan performance

- Diagram *register addressing*



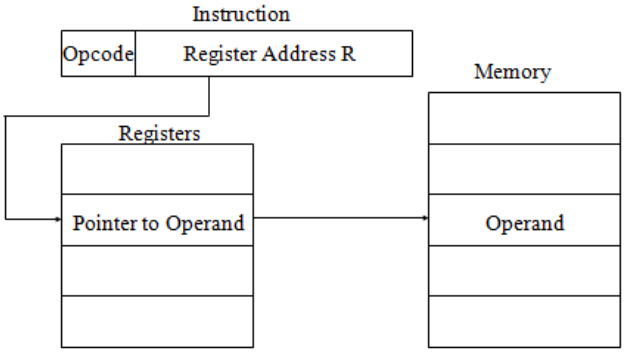
Operand terletak di register

- ***Keuntungan :***
  - Metode pengalamatan register mirip dengan mode pengalamatan langsung.
  - Perbedaannya terletak pada field alamat yang mengacu pada register, bukan pada memory utama.
  - Field yang mereferensi register memiliki panjang 3 atau 4 bit, sehingga dapat mereferensi 8 atau 16 register general
  - Pengambilan operand dari register lebih cepat dari pengambilan operand dari memori.
  - Jumlah register jauh lebih sedikit dibanding dengan jumlah lokasi memori, sehingga jumlah bit alamat yang dibutuhkan lebih sedikit akibatnya ukuran address bisa lebih kecil.
  - Diperlukan field alamat berukuran kecil dalam instruksi dan tidak diperlukan referensi memori
  - Akses ke regster lebih cepat daripada akses ke memori, sehingga proses eksekusi akan lebih cepat
- ***Kerugian :***
  - Jumlah register sangat terbatas, sehingga penggunaannya harus benar-benar efisien.
  - Ruang alamat menjadi terbatas

5. Register Indirect Addressing

- Metode pengalamatan register tidak langsung mirip dengan mode pengalamatan tidak langsung
- Perbedaannya adalah field alamat mengacu pada alamat register.
- Letak operand berada pada memori yang dituju oleh isi register
- Keuntungan dan keterbatasan pengalamatan register tidak langsung pada dasarnya sama dengan pengalamatan tidak langsung
- Keterbatasan field alamat diatasi dengan pengaksesan memori yang tidak langsung sehingga alamat yang dapat direferensi makin banyak
- Dalam satu siklus pengambilan dan penyimpanan, mode pengalamatan register tidak langsung hanya menggunakan satu referensi memori utama sehingga lebih cepat daripada mode pengalamatan tidak langsung

- EA = (R)
- Tempat pengalamatan luas (2^n)
- Diagram *register indirect addressing*



- ***Keuntungan:***
  Jumlah lokasi memori yang diacu lebih banyak, bergantung pada ukuran register.
- ***Kerugian:***
  Selain mengakses register, diperlukan satu referensi memori.

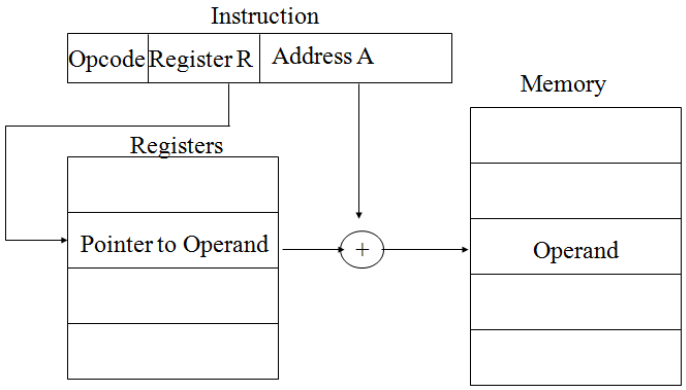
6. Displacement addressing

- Menggabungkan kemampuan pengalamatan langsung dan pengalamatan register tidak langsung
- Mode in mensyaratkan instruksi memiliki dua buah field alamat, sedikitnya sebuah field yang eksplisit
- Field eksplisit bernilai A dan field implisit mengarah pada register
- Operand berada pada alamat A ditambahkan isi register
- Tiga model displacement

- Relative addressing
- Base register addressing
- Indexing

- EA = A + (R)

  Berdasarkan formula di atas, address field menampung 2 nilai, yaitu:
  - A sebagai base value
  - R sebagai register yang menampung pertukaran sementara (that holds displacement)
  - Atau sebaliknya
- Diagram *displacement addressing*



Alamat efektif diperoleh dengan menjumlahkan isi register dengan A.

- ***Keuntungan:***

  Lebih fleksibel
- ***Kerugian:***

  Lebih kompleks

Relative Addressing

Register yang direferensi secara implisit adalah progra counter (PC)

- Alamat efektif didapatkan dari alamat instruksi saat itu ditambahkan ke field alamat
- Memanfaatkan konsep lokalitas memori untuk menyediakan operand-operand berikutnya
- Merupakan salah satu versi dari pengalamatan untuk pertukaran (displacement addressing)

R = Program counter, PC

EA = A + (PC)

Keterangan:

Ambil operand dari A dan dari lokasi yang ditunjukkan oleh program counter (PC)

Base-Register Addressing

Mode ini digunakan untuk relokasi program di dalam memori (dari satu area ke area lain). Pada mode pengalamatan pengalamatan base register, instruksi tidak berisi alamat. Dia memberikan perpindahan relatif terhadap area memori sekarang ke area memori yang lain, base register diisi dengan alamat base baru. Instruksi tidak perlu dimodifikasi/diubah. Dengan cara ini, keseluruhan program atau suatu segment dari program dapat dipindahkan dari satu area di memori ke yang lain tanpa mempengaruhi instruksi, dengan perubahan sederhana ini base register. Hal ini penting untuk sistem multiprogramming karena waktu yang berbeda (run), area berbeda dari memori tersedia untuk sebuah program. Sebuah CPU dapat mempunyai lebih dari satu base register.

Base register addresing, register yang direferensi berisi sebuah alamat memori, dan field alamat berisi perpindahan dari alamat itu

- Referensi register dapat eksplisit maupun implisit
- Memanfaatkan konsep lokalitas memori

Indexed Addressing

- Indexing dalah field alamat mereferensi alamat memori utama, dan register yang direferensikan berisi pemindahan positif dari alamat tersebut
- Merupakan kebalikan dari mode base register
- Field alamat dianggap sebagai alamat memori dalam indexing
- Manfaat penting dari indexing adalah untuk eksekusi program-program iteratif
- A = dasar
- R = displacement
- EA = A + R
- Baik untuk pengaksesan array

  EA = A + R

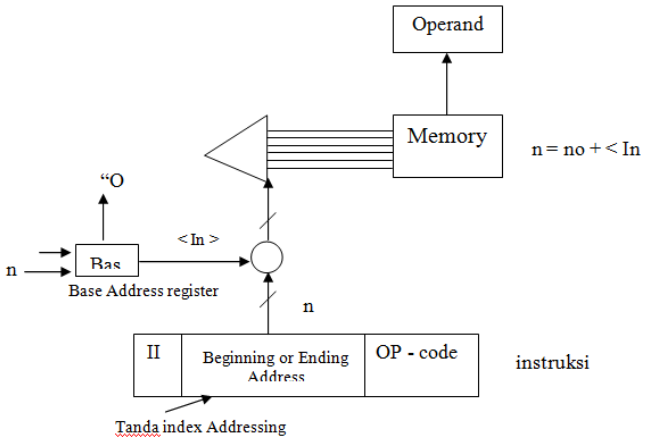
  R++

Untuk pelaksanaan perhitungan address di perlukan :

- Rangkaian addressing
- Index register

- Tanda untuk indexed addressing
- Instruksi tambahan:

  Load index register dengan n1, Rubah index register, Conditional jump



Combinations

- Postindex

  EA = (A) + (R)
- Preindex

  EA = (A+(R))

7. Stack Addressing

- Stack adalah array lokasi yang linear, yang merupakan blok lokasi yang terbalik, sehingga sering disebut juga Last In First Out Queue. Tidak ada referensi memori di dalam Stack Addressing di mana aplikasi memori yang dimilikinya terbatas. Stack addressing merupakan bentuk implied addressing. Instruksi-instruksi mesin tidak perlu memiliki referensi memori namun secara implisit beroperasi pada bagian paling atas stack. Dua elemen teratas stack dapat berada di dalam register CPU, yang dalam hal ini stack pointer mereferensi ke elemen ketiga stack. Stack pointer tetap berada dalam register. Dengan demikian, referensi-referensi ke lokasi stack di dalam memori pada dasarnya merupakan pengalamatan register tidak langsung
- Operand secara implisit berada pada bagian paling atas stack
- Contoh :

  - ADD *pop* dua elemen teratas stack dan jumlahkan

Format instruksi

Operand terletak di puncak stack

- ***Keuntungan:***
  Ukuran instruksi kecil
- ***Kerugian***:
  Pemakaiannya terbatas

Jika ada beberapa mode pengalamatan yang digunakan, bagaimana komputer tahu mode pengalamatan yang digunakan pada saat mengeksekusi suatu instruksi ?
Untuk mengkodekan mode pengalamatan pada format instruksi dapat ditambahkan sebuah field untuk mengkodekan mode pengalamatan :

Format instruksi

Opcode field	Mode field	Address field

Contoh :
Misalkan ukuran opcode field adalah 2 bit. Pada opcode field tersebut akan dikodekan 4 operasi (+,-,* , dan /). Untuk tiap operasi ada 4 mode pengalamatan yang mungkin. Untuk itu disediakan mode field yang ukurannya 2 bit.. Kode yang digunakan adalah sebagai berikut :

0000 : Penjumlahan dengan mode immediate
0001 : Penjumlahan dengan mode direct
0010 : Penjumlahan dengan mode indirect
0011 : Penjumlahan dengan mode register
0100 : Pengurangan dengan mode immediate
0101 : Pengurangan dengan mode direct
0110 : Pengurangan dengan mode indirect
0111 : Pengurangan dengan mode register
1000 : Perkalian dengan mode immediate
….
1100 : Pembagian dengan mode immediate

....

Tabel Basic Addressing Modes

Mode	**Algorithm**	**Example transfer ***	**Principal Advantage**	**Principal Disadvantage**
Immediate	Operand = A	OPR ← value	No memory reference	Limited operand magnitude
Direct	EA = A	OPR ← M[ad]	Simple	Limited address space
Indirect	EA = (A)	OPR ← M(M[ad])	Large address space	Multiple memory references
Register	EA = R	OPR ← (R1)	No memory Reference	Limited address space
Register Indirect	EA = (R)	OPR ← M[R1]	Large address space	Extra memory reference
Displacement	EA=A+(R)		flexibility	Complexity
Stack	EA=top of Stack		No memory Reference	Limited applicability

*) OPR mewakili sebuah register untuk menyimpan operand yangakan digunakan sewaktu instruksi dijalankan.

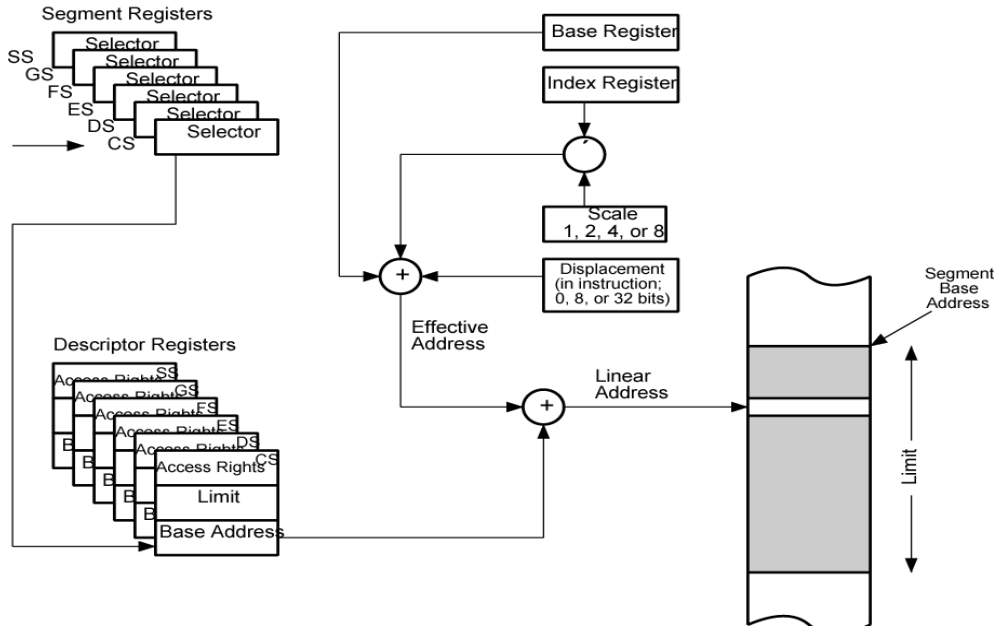
Mode pengalamatan Pentium

Virtual atau effective address diproses kedalam segmen.

12 addressing modes yang tersedia pada mode pengalamatan Pentium, diantaranya.

- Mode Immediate
  - ✓ Operand berada di dalam intruksi.
  - ✓ Operand dapat berupa data byte, word atau doubleword.
- Register operand
  - ✓ yaitu operand adalah isi register.
  - ✓ Register 8 bit (AH, BH, CH, DH, AL, BL, CL, DL)
  - ✓ Register 16 bit (AX, BX, CX, DX, SI, DI, SP, BP)
  - ✓ Register 32bit (EAX, EBX, ECX, ESI, EDI, ESP, EBP)
  - ✓ Register 64 bit yang dibentuk dari register 32 bit secara berpasangan.
  - ✓ Register 8, 16, 32 bit merupakan register untuk penggunaan umum (general purpose register)
  - ✓ Register 14 bit biasanyan untuk operasi floating point.
  - ✓ Register segmen (CS, DS, ES, SS, FS, GS)
- Mode Displacement
  - ✓ alamat efektif berisi bagian-bagian intruksin dengan displacement 8, 16, atu 32 bit.
  - ✓ dengan segmentasi, seluruh alamat dalam intruksi mengacu ke sebuah offset di dalam segmen.
  - ✓ dalam Pentium, mode ini digunakan untuk mereferensi variable-variabel global.
- Mode Base
  - ✓ pengalamatan indirect yang menspesifikasi saru register 8, 16 atau 32 bit berbasis alamat efektifnya.
- Mode Base with displacement
- Mode Scaled index with displacement
- Mode Base with index and displacement
- Mode Base scaled index with displacement
- Mode Relative Addressing
  - ✓ digunakan dalam intruksi-intruksi transfer control.
  - ✓ displacement di tambahkan ke program counter (PC), yang menunjuk ke intruksi berikutnya.

Proses perhitungan model pengalamatan Pentium



Mode pengalamatan PowerPC

- Load/store architecture
  - Indirect
    - ✓ Instruksimencakup 16 bit *displacement*untuk di tambahkan pada*base register* (may be GP register)
    - ✓ Dapatmenggantikan *base register content*dengan alamat baru
    - ✓ Algoritma : EA = (BR) + D
    - ➔ BR : base register
    - ➔ D : displacement
  - Indirect indexed
    - ✓ Referensi instruksi*base register* dan*index register* (both may be GP)
    - ✓ EA adalah jumlah content
    - ✓ Algoritma : EA = (BR) + (IR)
    - ➔ IR : index register
- Branch address
  - Absolute

    Algoritma : EA = 1
  - Relative

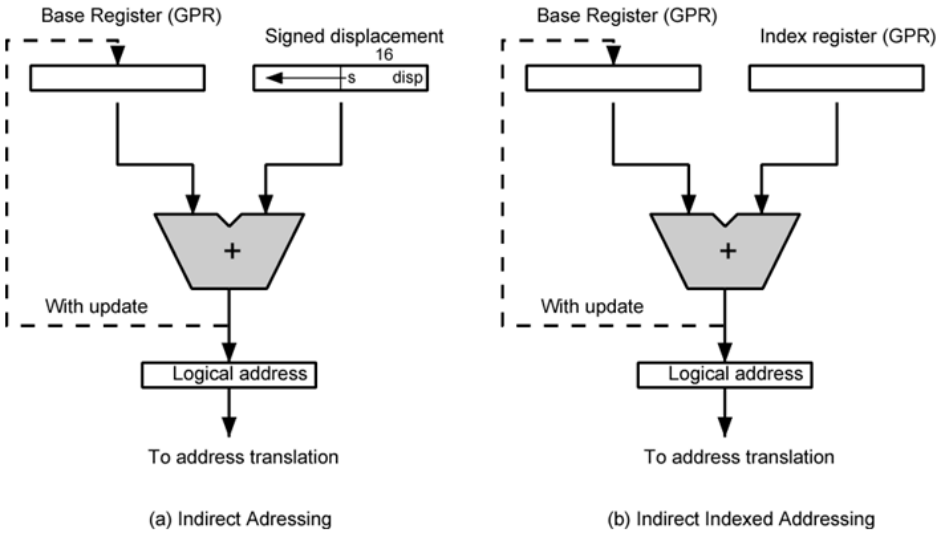
    Algoritma : EA = (PC) + 1
    - ➔ PC : program counter
  - Indirect

- ✓ Alamat operand ditunjukkan secara tidak langsung olehdata yang terkandung pada alamat yang ditunjuk
- ✓ Cara penulisan: LOAD **(Y)**
- ✓ Algoritma : EA = (L/CR)
- ➔ L/CR : link register atau count register

- Arithmetic
  - Operands di register atau bagian dari instruksi
  - Floating point is register only
  - Operasi set instruksi untuk arithmetic :

1. ADD : penjumlahan
2. SUBTRACT : pengurangan
3. MULTIPLY : perkalian
4. DIVIDE : pembagian
5. ABSOLUTE
6. NEGATIVE
7. DECREMENT
8. INCREMENT

Mode pengalamatan Operand memori PowerPC



Format Instruksi

Format instruksi menentukan layout bit di dalam suatu instruksi yang mencakup opcode, dan (implicit dan explicit) operand. Pada umumnya lebih dari satu format instruksi di sebuah set instruksi.

Panjang instruksi harus merupakan kelipatan panjang karakter, yang umumnya 8 bit dan kelipatan panjang bilangan fixed point.Untuk memahami hal ini, kita harus menggunakan istilah word. Umumnya ukuran word menentukan ukuran bilangan-bilangan fixed point

dan berhubungan dengan ukuran transfer memori.Panjang instruksi dipengaruhi dan mempengaruhi yaitu : ukuran memori, organisasi memori, struktur bus, kompleksitas CPU, kecepatan CPU. Trade off between powerful instruction repertoire and saving space.

Alokasi bit yang berkaitan dengan penggunaan bit-bit pengalamatan :

1. Jumlah mode pengalamatan dan operand
2. Jumlah set register
3. Register dan Memori
4. Jangkauan Alamat

Format instruksi PDP-8

Memory Reference Instructions

Opcode			D/I	Z/C	Displacement						
0		2	3	4	5						11

Input/Output Instructions

1	1	0	Device							Opcode	
0		2	3						8	9	11

Register Reference Instructions

Group 1 Microinstructions

1	1	1	0	CLA	CLL	CMA	CML	RAR	RAL	BSW	IAC
0	1	2	3	4	5	6	7	8	9	10	11

Group 2 Microinstructions

1	1	1	1	CLA	SMA	SZA	SNL	RSS	OSR	HLT	0
0	1	2	3	4	5	6	7	8	9	10	11

Group 3 Microinstructions

1	1	1	1	CLA	MQA	0	MQL	0	0	0	1
0	1	2	3	4	5	6	7	8	9	10	11

D/I = Direct/Indirect address
Z/C = Page 0 or Current page
CLA = Clear Accumulator
CLL = Clear Link
CMA = CoMplement Accumulator
CML = CoMplement Link
RAR = Rotate Accumultator Right
RAL = Rotate Accumulator Left
BSW = Byte SWap
IAC = Increment ACcumulator
SMA = Skip on Minus Accumulator
SZA = Skip on Zero Accumulator
SNL = Skip on Nonzero Link
RSS = Reverse Skip Sense
OSR = Or with Switch Register
HLT = HaLT
MQA = Multiplier Quotient into Accumulator
MQL = Multiplier Quotient Load

Format instruksi PDP-10

Opcode								Register				I	Index Register					Memory Address																	
0							8	9			12		14				17	18																		35

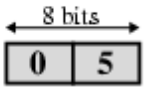
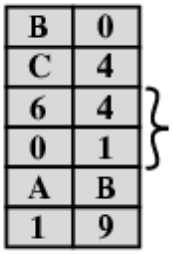
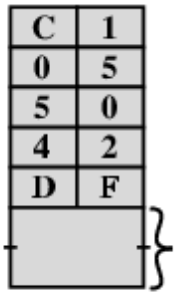
I = indirect bit

Format instruksi PDP-11

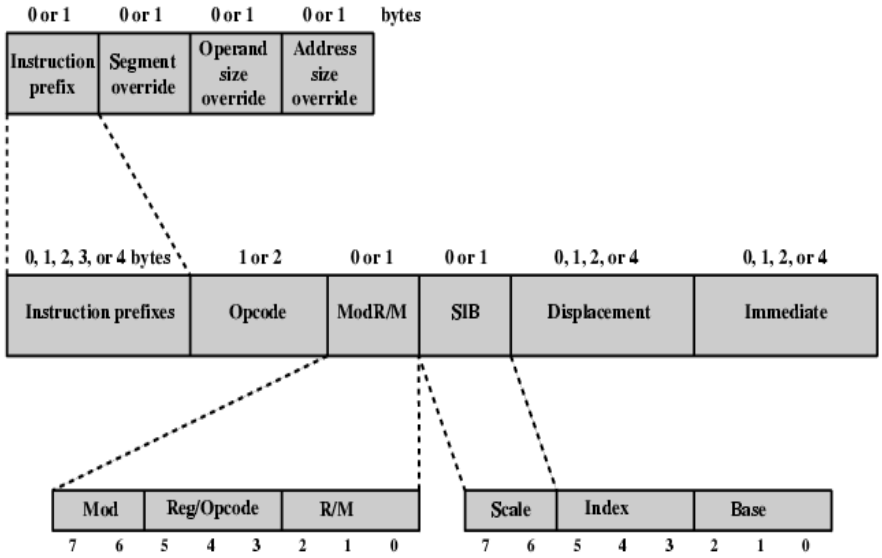
Opcode								Register				I	Index Register					Memory Address																	
0							8	9			12		14				17	18																		35

I = indirect bit

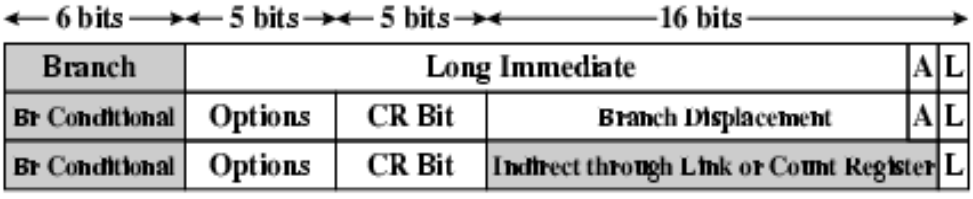
Contoh instruksi VAX

Hexadecimal Format	Explanation	Assembler Notation and Description
	Opcode for RSB	RSB Return from subroutine
	Opcode for CLRL Register R9	CLRL R9 Clear register R9
	Opcode for MOVW Word displacement mode, Register R4 356 in hexadecimal Byte displacement mode, Register R11 25 in hexadecimal	MOVW 356(R4), 25(R11) Move a word from address that is 356 plus contents of R4 to address that is 25 plus contents of R11
	Opcode for ADDL3 Short literal 5 Register mode R0 Index prefix R2 Indirect word relative (displacement from PC) Amount of displacement from PC relative to location A	ADDL3 #5, R0, @ A[R2] Add 5 to a 32-bit integer in R0 and store the result in location whose address is sum of A and 4 times the contents of R2

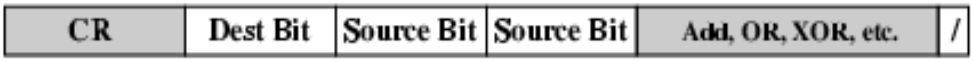
Format instruksi Pentium



Format instruksi PowerPC



(a) Branch instructions



(b) Condition register logical instructions

(c) Load/store instructions

Arithmetic	Dest Register	Src Register	Src Register	O	Add, Sub, etc.	R	
Add, Sub, etc.	Dest Register	Src Register	Signed Immediate Value				
Logical	Src Register	Dest Register	Src Register	ADD, OR, XOR, etc.		R	
AND, OR, etc.	Src Register	Dest Register	Unsigned Immediate Value				
Rotate	Src Register	Dest Register	Shift Amt	Mask Begin	Mask End	R	
Rotate or Shift	Src Register	Dest Register	Src Register	Shift Type or Mask		R	
Rotate	Src Register	Dest Register	Shift Amt	Mask	XO	S	R *
Rotate	Src Register	Dest Register	Src Register	Mask	XO		R *
Shift	Src Register	Dest Register	Shift Type or Mask			S	R *

(d) Integer arithmetic, logical, and shift/rotate instructions

Flt sgl/dbl	Dest Register	Src Register	Src Register	Src Register	Fadd, etc.	R

(e) Floating-point arithmetic instructions

A = Absolute or PC relative
L = Link or subroutine
O = Record overflow in XER
R = Record condition in CR1

XO = Opcode extension
S = Part of shift amount field
* = 64-bit implementation only

REFERENSI

Stalling, W. 2010. Computer Organization and Architecture design dan Performance eighth edition. Prentice Hall

PROPAGASI

A. Latihan dan Diskusi (Propagasi vertical dan Horizontal)

11.1 Briefly define immediate addressing.

11.2 Briefly define direct addressing.

11.3 Briefly define indirect addressing.

11.4 Briefly define register addressing.

11.5 Briefly define register indirect addressing.

B. Pertanyaan (Evaluasi mandiri)

11.1 Given the following memory values and a one-address machine with an accumulator,

what values do the following instructions load into the accumulator?

- Word 20 contains 40.
- Word 30 contains 50.
- Word 40 contains 60.
- Word 50 contains 70.

a. LOAD IMMEDIATE 20

b. LOAD DIRECT 20

c. LOAD INDIRECT 20

d. LOAD IMMEDIATE 30

e. LOAD DIRECT 30

f. LOAD INDIRECT 30

11.2 Let the address stored in the program counter be designated by the symbol X1. The instruction stored in X1 has an address part (operand reference) X2. The operand needed to execute the instruction is stored in the memory word with address X3.An index register contains the value X4.What is the relationship between these various quantities if the addressing mode of the instruction is (a) direct; (b) indirect; (c) PC relative; (d) indexed?

11.3 An address field in an instruction contains decimal value 14. Where is the corresponding operand located for

a. immediate addressing?

b. direct addressing?

c. indirect addressing?

d. register addressing?

e. register indirect addressing?

C. QUIZ -mutiple choice (Evaluasi)

D. PROYEK (Eksplorasi entrepreneurship, penerapan topic bahasan pada dunia nyata)