PROGRAM STUDI
TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

Addressing Mode

- ✓ Immediate
- ✓ Direct
- ✓ Indirect
- ✓ Register
- ✓ Register Indirect
- ✓ Displacement (Indexed)
- ✓ Stack

Tim pengampu

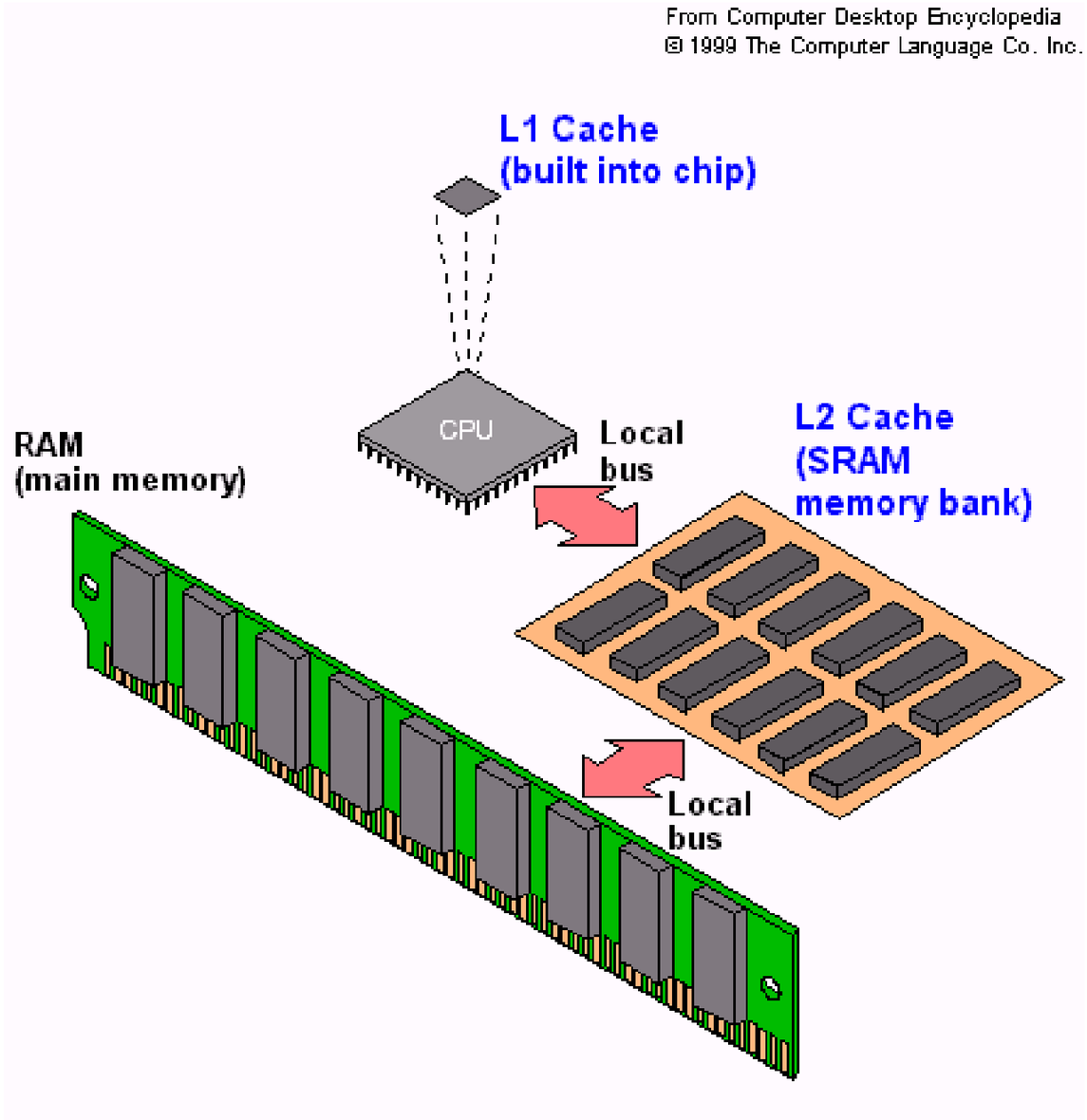
Sistem Komputer, Komunikasi dan Keamanan Data

T.A. 2020

Konsep Memory

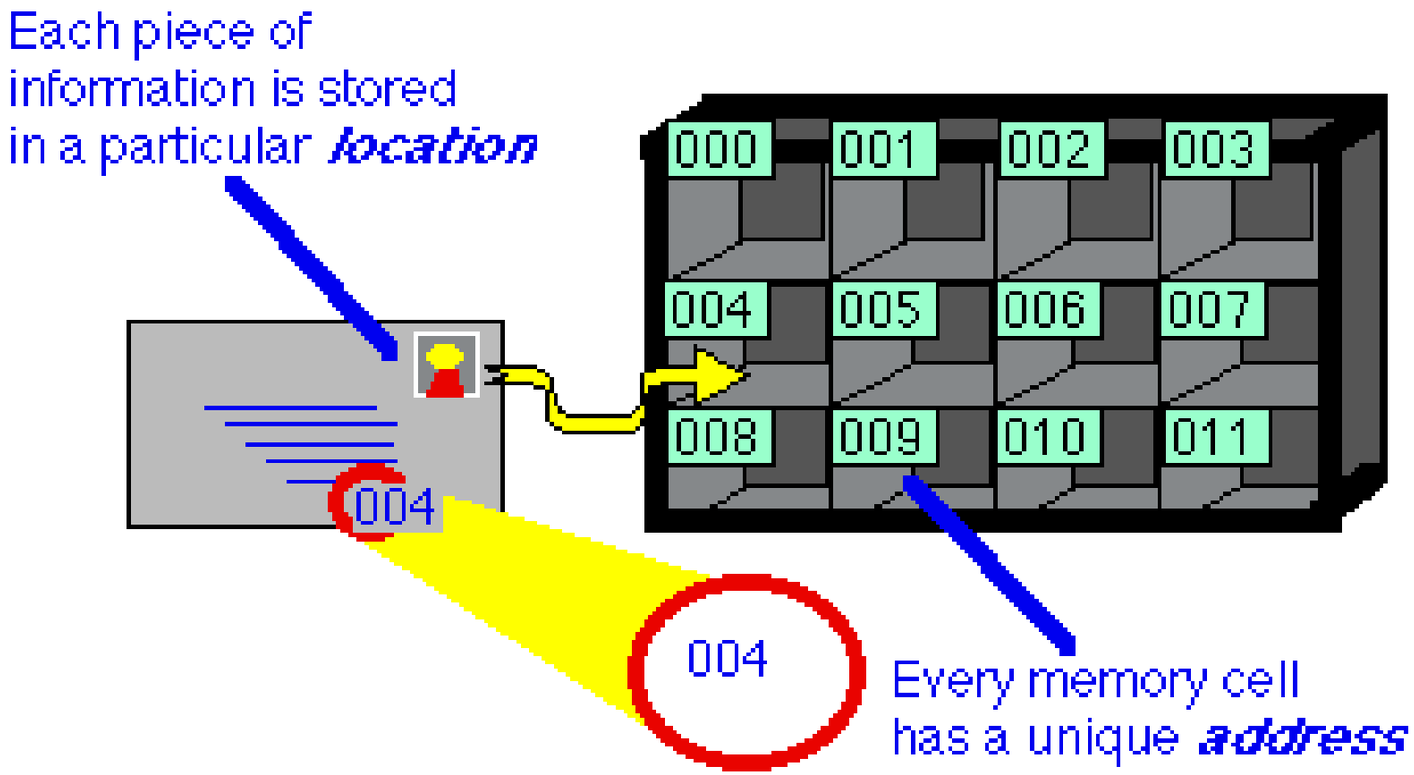
Menggambarkan bagaimana
Data (atau instruksi) disimpan
Sementara untuk dibawa ke CPU

Masing-masing tipe processor
Menggunakan mode pengalamatan
Yang **berbeda** satu dengan yang lain



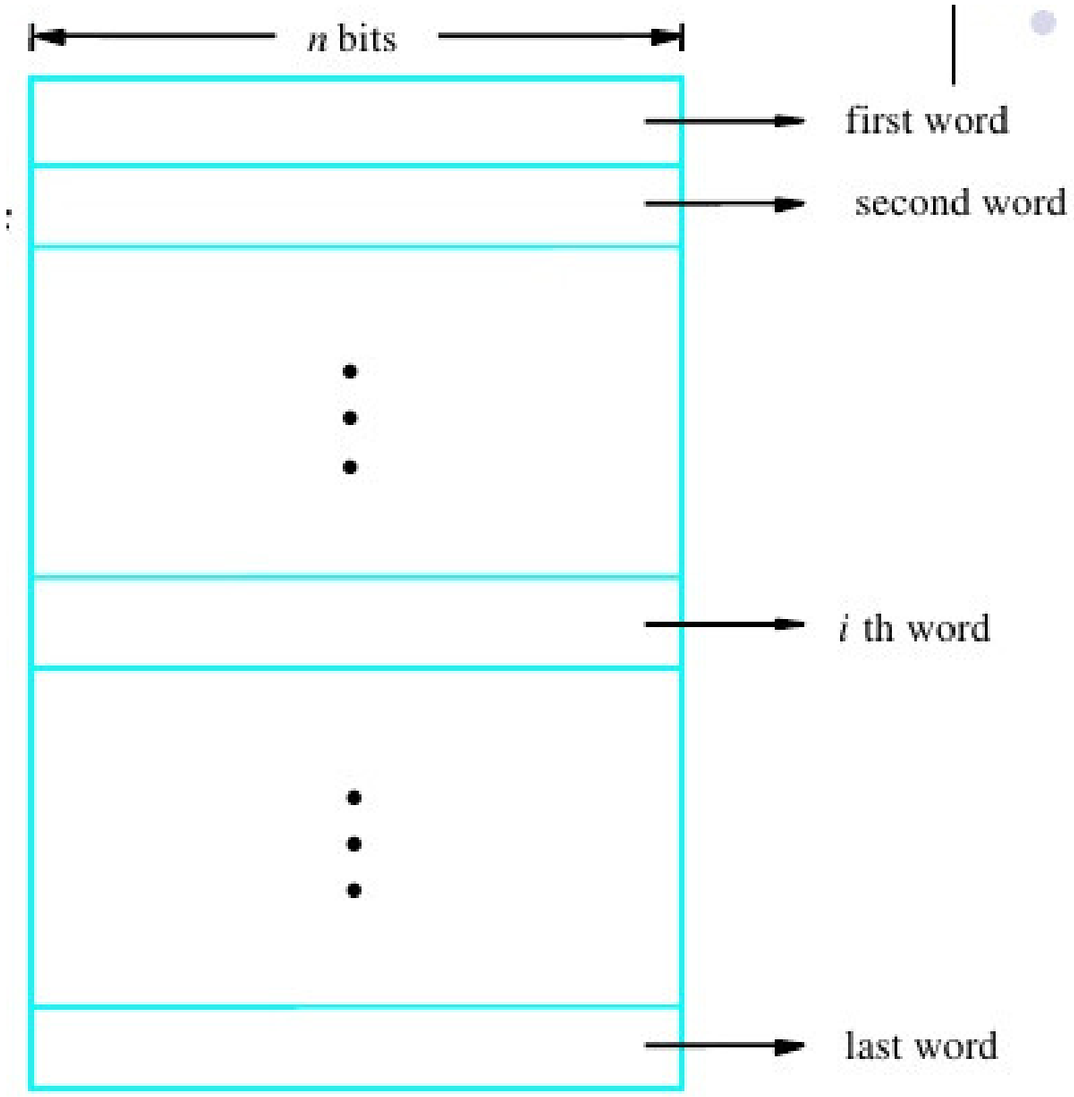
Mode Pengalamatan

Data atau instruksi disimpan di memory berdasarkan **alamat unik** tertentu agar memudahkan pengaksesan



Memory terdiri dari Jutaan ***cell*** penyimpanan
Setiap cell mampu menyimpan 1 bit data

Data biasanya diakses sejumlah n-bit, Setiap n-bit data disebut word



Komputer modern biasanya mempunyai panjang word 16-64bits

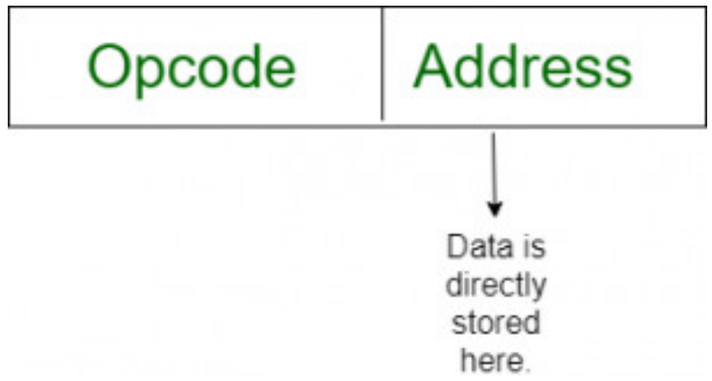
Satu unit 8 bit disebut **byte**

Mode Pengalamatan

- Cara menunjuk dan Mengarahkan alamat suatu lokasi register memori di mana operand akan diambil.
- Instruksi terdiri dari OpCode dan Alamat register.

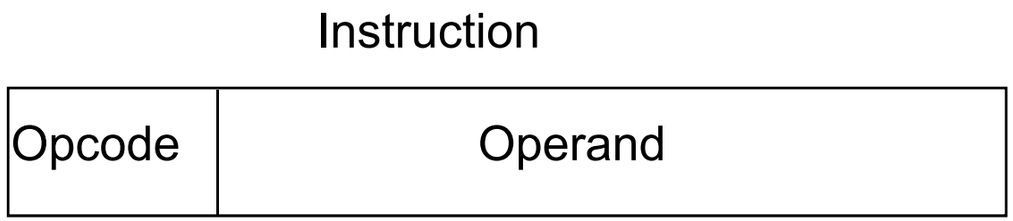
Macam Mode pengalamatan

- Immediate
- Direct
- Indirect
- Register
- Register Indirect
- Displacement (Indexed)
- Stack



Immediate Addressing

- Bentuk pengalamatan yg paling sederhana
- Operand merupakan bagian dari Instruksi
- Operand = address field



Contoh

ADD #5

- Langsung tambahkan (Add) 5 ke isi *accumulator*
- 5 adalah operand

Keuntungan

- Tidak butuh referensi memory untuk mengambil (fetch) data
- Cepat, Siklus instruksi pendek

Kerugian

- Jangkauan terbatas, dibatasi oleh ukuran field alamat

Direct Addressing

Pengalamatan Langsung, banyak digunakan pada komputer lama dan komputer kecil.

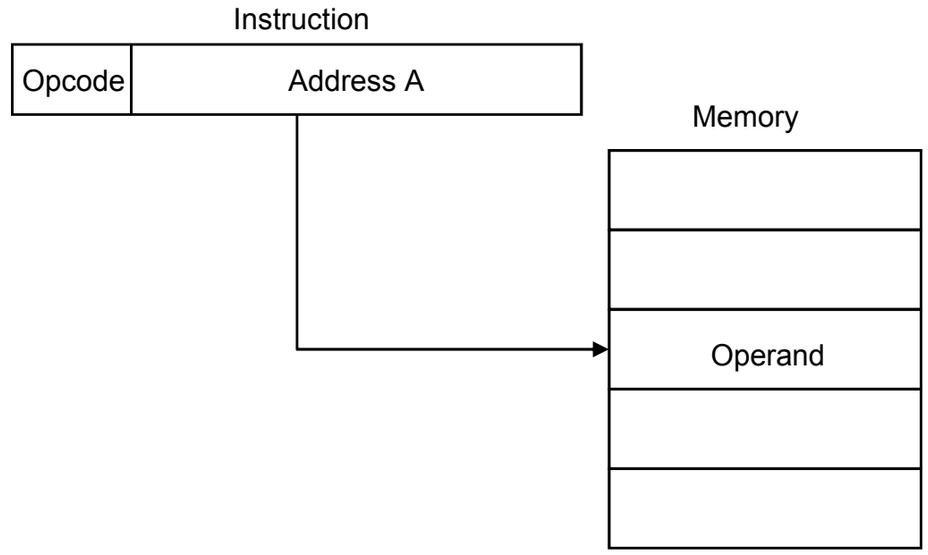
EA = M

Keuntungan

- Field alamat berisi *Efective Address (EA)* sebuah operand.
- Hanya memerlukan sebuah *alamat* referensi memori (A) untuk mengakses data
- dan tidak memerlukan kalkulasi khusus, untuk membaca alamat

Kerugian

- Keterbatasan field alamat karena panjang field alamat biasanya lebih kecil dibandingkan panjang word



Contoh

ADD A

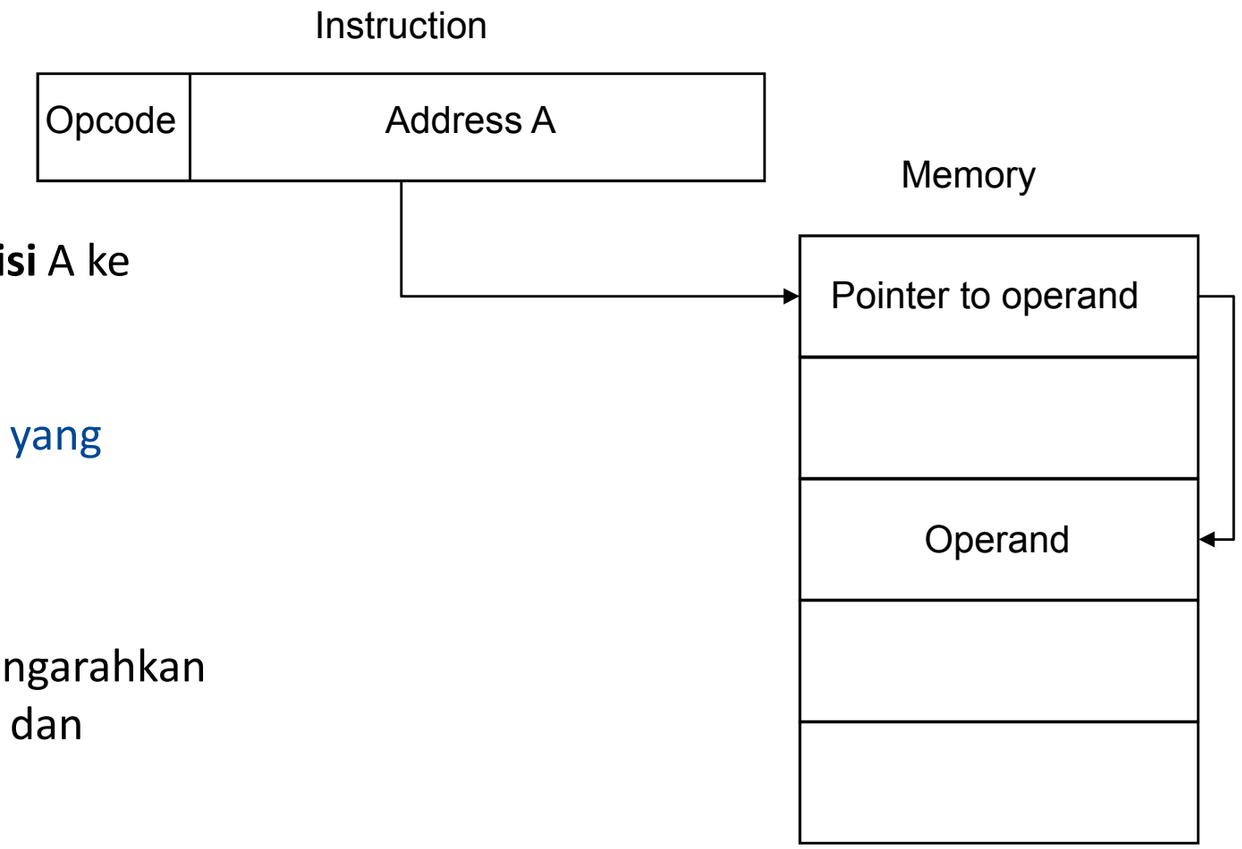
- Menambahkan (Add) isi Memory A ke *accumulator*
- *Cari* operand di memory Memory A

ADD 12

menambahkan nilai yang ada di alamat **memori** 12

InDirect Addressing

- Pengalamatan tidak langsung, Field alamat mengacu (pointer dari) pada alamat word di dalam memori, yang akan berisi alamat operand yang panjang



Contoh

ADD (A)

- Tambahkan (Add) isi dari *cell* yang **ditunjuk oleh isi** A ke *accumulator*

ADD (30)

Lihat alamat 30 di memory, tambahkan dengan nilai yang ditunjukkan isi alamat 30 di memory

Keuntungan

- Keterbatasan alamat Register dapat diatasi dengan mengarahkan ke memori utama, sehingga space menjadi lebih besar dan semakin banyak alamat yang dapat referensi.

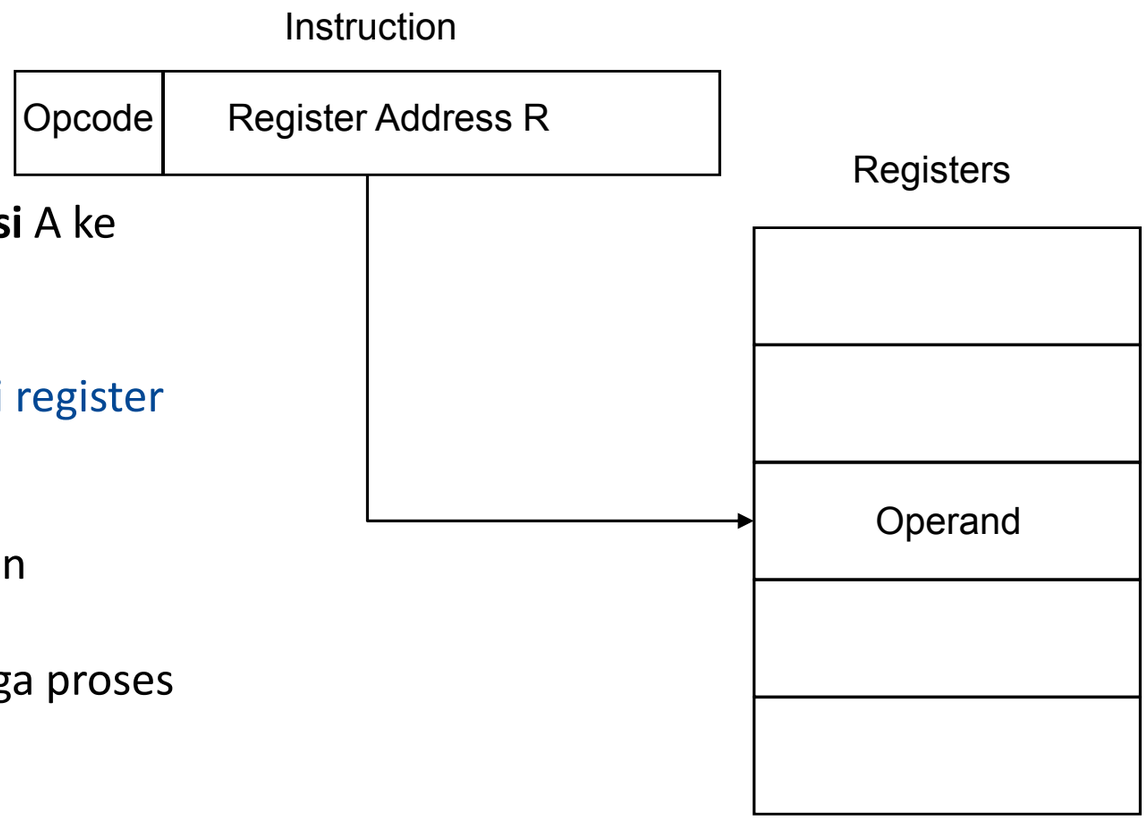
Kerugian

- Diperlukan referensi memori ganda dalam satu fetch sehingga memperlambat proses operasi

Register Addressing

- Mirip dengan pengalamatan langsung

  EA = R
- Bedanya pointer mengarah ke register bukan memory utama
- Field yang mereferensi register memiliki panjang 3 atau 4 bit, sehingga dapat mereferensi 8 atau 16 register general purpose.



ADD R

- Tambahkan (Add) isi dari *cell* register yang **ditunjuk oleh isi** A ke *accumulator*

ADD 75

menambahkan nilai dengan isi yang ditunjukkan alamat 75 di register

Keuntungan

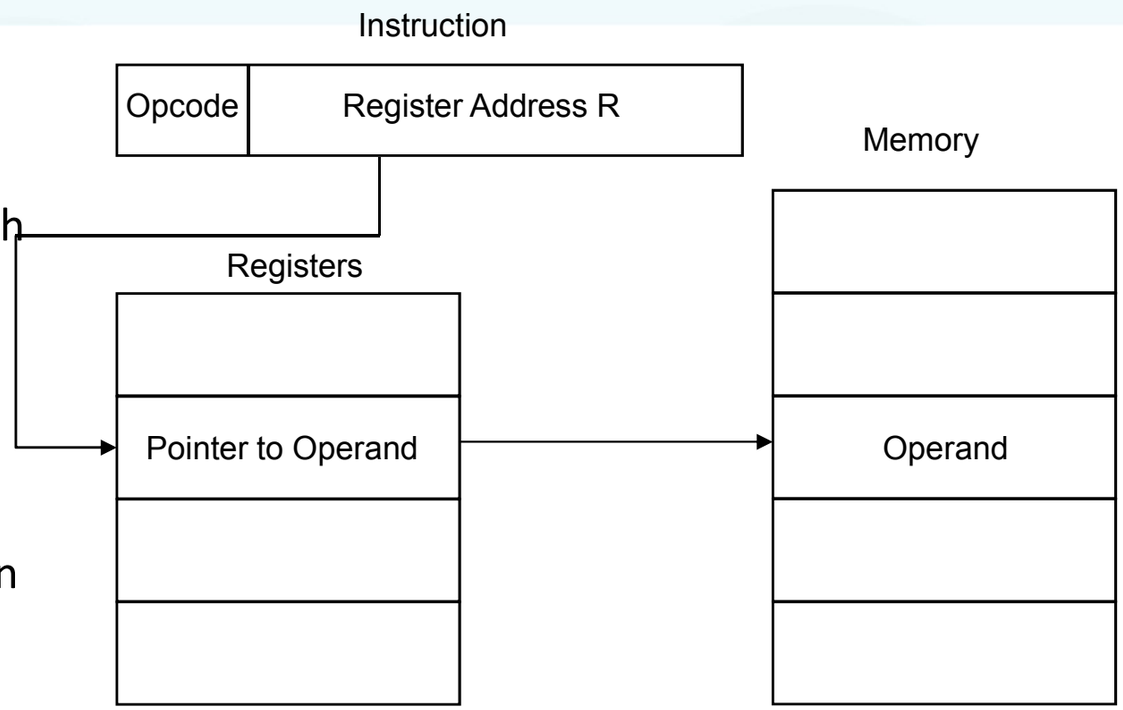
- field alamat berukuran kecil dalam instruksi dan tidak diperlukan referensi memori.
- Akses ke register lebih cepat daripada akses ke memori, sehingga proses eksekusi akan lebih cepat

Kerugian

- Ruang alamat menjadi terbatas (krn ukuran register kecil)

Register InDirect Addressing

- Mirip dengan pengalamatan Tidak Langsung
  EA = (R)
- Bedanya pointer mengacu alamat memory yang ditunjukkan oleh isi register R
- Field yang mereferensi register memiliki panjang 3 atau 4 bit, sehingga dapat mereferensi 8 atau 16 register general purpose.



Keuntungan (hampir sama dg Indirect Addressing)

- Keterbatasan alamat Register dapat diatasi dengan mengarahkan ke memori utama, sehingga space menjadi lebih besar dan semakin banyak alamat yang dapat referensi.
- Dalam satu siklus pengambilan dan penyimpanan, hanya menggunakan satu referensi memori utama sehingga lebih cepat daripada mode pengalamatan tidak langsung

Kerugian

- Diperlukan referensi memori ganda dalam satu fetch sehingga memperlambat proses operasi

Contoh

ADD (R)

Tambahkan (Add) isi dari *cell* register yang **ditunjuk oleh isi** R ke *accumulator*

ADD (45)

Lihat alamat 45 di **register**, tambahkan dengan nilai yang ditunjukkan isi alamat register 45 di **memory**

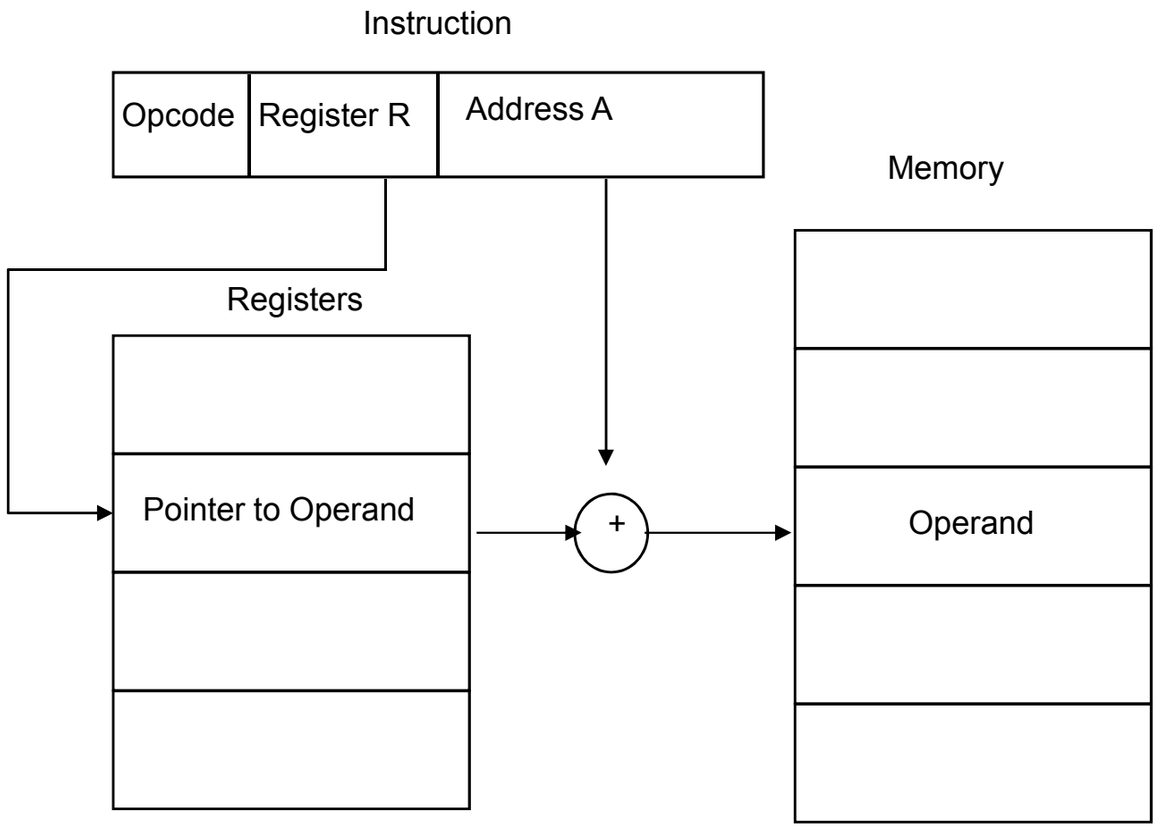
Displacement Addressing

- Gabungan *Direct Addressing* dan *Indirect Register Addressing*
- Operand berada pada alamat A ditambah isi register

EA = A + (R)

- Field alamat berisi 2 nilai
  - A = base value
  - R = register that holds displacement

  Atau sebaliknya



Contoh

ADD (23) + 5

- Lihat alamat 23 di **register** (misal berisi 10) , tambahkan dengan 5, 10+5=15
- Nilai mengacu ke alamat 15 di **memory**

Ada Tiga model displacement Addressing

- Relative Addressing
- Base Register Addressing
- Indexing

3 Model Displacement Addressing

- **Relative Addressing**
  Register yang direferensi secara implisit adalah program counter (PC).
  - Alamat efektif didapatkan dari alamat instruksi saat itu ditambahkan ke field alamat.
  - Memanfaatkan konsep lokalitas memori untuk menyediakan Operand-operand berikutnya.

- **Base Register Addressing**
  Register yang direferensikan berisi sebuah alamat memori, dan field alamat berisi perpindahan dari alamat itu.
  - Referensi register dapat eksplisit maupun implisit.
  - Memanfaatkan konsep lokalitas memori.

- **Indexing**
  Field alamat mereferensi alamat memori utama, dan register yang direferensikan berisi pemindahan positif dari alamat tersebut.
  - Merupakan kebalikan dari model base register.
  - Field alamat dianggap sebagai alamat memori dalam indexing.
  - Manfaat penting dari indexing adalah untuk eksekusi program-program iteratif

Stack Addressing

- Stack adalah array lokasi yang linier => pushdown, Antrian list = last-in-first-out (LIFO)
- Stack merupakan blok lokasi yang terbalik.
- Butir ditambahkan ke puncak stack sehingga setiap saat blok akan terisi secara parsial.
- Nilai Pointer stack merupakan alamat bagian paling atas stack
- Dua elemen teratas stack dapat berada di dalam register CPU, yang dalam hal ini stack ponter mereferensi ke elemen ketiga stack.
- Stack pointer tetap berada di dalam register.
- Dengan demikian, referensi referensi ke lokasi stack di dalam memori pada dasarnya merupakan *Indirect Register Addressing*

Contoh

ADD

Instruksi ini akan *POP* 2 item teratas dari *stack,* tambahkan(ADD), dan kemudian *PUSH* hasilnya ke *stack (tumpukan)* teratas

Perbandingan Mode Pengalamatan

Mode	Algoritma	Keuntungan Utama	Kerugian utama
Immediate	Operand = A	Tidak ada referensi memori	Besaran operand terbatas
Direct	EA = A	Sederhana	Ruang alamat terbatas
Indirect	EA = (A)	Ruang alamat besar	Referensi memori berganda
Register	EA = R	Tidak ada referensi memori	Ruang alamat terbatas
Register Indirect	EA = (R)	Ruang alamat besar	Referensi memori ekstra
Displacement	EA = A+ (R)	Fleksibelitas	Kompleksitas
Stack	EA = Puncak Stack	Tidak ada referensi	Aplikasi memori terbatas

Contoh Soal

```
LOAD 205
ADD #10
MPY ( 1 )
SUB 2
DIV ( B ) + 188
ADD (204)
STOR Y
```

REGISTER	
1	10
2	20
3	30
4	40
5	50
...	...
A	15
B	16
C	17
D	18
E	19
...	...
21	100
22	200
23	300
24	400
25	500
X	
Y	

MEMORI	
15	95
16	96
17	97
18	98
19	99
...	...
10	5
20	6
30	7
40	8
50	9
...	...
201	10
202	20
203	30
204	40
205	50

Contoh Penyelesaian

	TYPE	ADDR	OPR	BUFF
LOAD 205	DIR	205		50
ADD #10	IMM		+10	60
MPY (1)	REGI	10	*5	300
SUB 2	REG	2	-20	280
DIV (B) + 188	DISP	204	/40	7
ADD (204)	IND	40	+8	15
STOR Y				

...

Y = 15

KETERANGAN :

IMM	= IMMEDIATE	ADD	= TAMBAH
DIR	= DIRECT	SUB	= KURANG
IND	= INDIRECT	MPY	= KALI
REG	= REGISTER	DIV	= BAGI
REGI	= REGISTER INDIRECT		
DISP	= DISPLACEMENT		

LOAD 205
ADD #10
MPY (1)
SUB 2
DIV (B) + 188
ADD (204)
STOR Y

REGISTER	
1	10
2	20
3	30
4	40
5	50
...	...
A	15
B	16
C	17
D	18
E	19
...	...
21	100
22	200
23	300
24	400
25	500
X	
Y	

MEMORI	
15	95
16	96
17	97
18	98
19	99
...	...
10	5
20	6
30	7
40	8
50	9
...	...
201	10
202	20
203	30
204	40
205	50

Referensi

UTAMA

- William Stalling, Computer Organization and organization 8th edition, Pearson Education, Inc, Pearson Prentice Hall, 2010
- Andrew S. Tanenbaum, Structured Computer Organization 4th Edition Pearson Prentice Hall,2001
- Mostafa Abd-El-Barr- Hesham El-Rewini, Fundamentals Of Computer Organization And Architecture, John Wiley & Sons, Inc, 2005

TAMBAHAN

- http://www.computerhistory.org
- https://homepage.cs.uri.edu/faculty/wolfe/book/Readings/Reading04.htm
- https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/
- https://www.electronics-tutorials.ws/binary/bin_2.html
- http://www.ict.griffith.edu.au/~johnt/1004ICT/lectures/
- https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/

THANKS

ANY QUESTIONS?