Set Intruksi
(*Intruction Sets*)

Karakteristik Intruksi Mesin

- Intruksi mesin (*machine instruction*) yang dieksekusi membentuk suatu operasi dan berbagai macam fungsi CPU.
- Kumpulan fungsi yang dapat dieksekusi CPU disebut set intruksi (*instruction set*) CPU.
- Beberapa hal terkait karakteristik intruksi mesin:
  - Elemen-elemen intruksi mesin
  - Representasi intruksinya
  - Jenis-jenis intruksi
  - Penggunaan Alamat
  - Rancangan set intruksi

Elemen Intruksi mesin

- Operation Code (Op Code): menspesifikasikan operasi yang akan dilakukan. Operasi berbentuk kode biner.
- Source Operand Reference: operasi dapat berasal dari lebih satu sumber. Operand adalah input operasi.
- Result operand reference: merupakan hasil atau keluaran operasi.
- Next Intruction Reference: elemen ini menginformasikan CPU posisi intruksi berikutnya yang harus diambil dan dieksekusi.

Operand dan Operasi

- Operand suatu operasi dapat berada disalah satu dari ketiga daerah ini:
  - Memori utama atau memori virtual (kapasitas besar namun lambat)
  - Register CPU (kapasitas kecil tapi cepat)
  - Perangkat I/O: dari mouse, keyboard, monitor, dll

Representasi Intruksi

- Intruksi komputer direpresentasikan oleh sekumpulan bit. Intruksi dibagi menjadi field (bagian-bagian).
- Field-field ini diisi oleh elemen-elemen intruksi yang membawa informasi bagi operasi CPU.
- Layout intruksi dikenal dengan format intruksi

Format Intruksi

Opcode	Alamat

- Kode operasi (opcode) direpresentasikan dengan singkatan-singkatan, yg disebut ‘MNEMONIC’
- Contoh:
  - ADD= penambahan,
  - SUB=subtract (pengurangan), dan
  - LOAD=muatkan data ke memori
- Contoh representasi operand secara simbolik:
  - ADD X, Y artinya tambahkan nilai yang berada pada lokasi Y ke register X, dan simpan di Register X.

Format Intruksi

- Contoh representasi operand secara simbolik:
  - ADD X, Y artinya tambahkan nilai yang berada pada lokasi Y ke register X, dan simpan di Register X.
- Programer dapat menuliskan program Bahasa mesin dalam bentuk simbolik
- Setiap opcode simbolik memiliki representasi biner yang tetap dan programmer dapat menetapkan lokasi masing-masing operand.

Jenis-jenis Intruksi

- Contoh suatu ekspresi bilangan:
  - X=X+Y
    - X dan Y berkorespondensi dengan lokasi 513 dan 514
- Pernyataan dalam Bahasa Tingkat tinggi tersebut mengintruksikan komputer untuk melakukan Langkah-Langkah berikut:
  - Muatkan sebuah register dengan isi lokasi memori 513
  - Tambahkan isi lokasi memori 514 ke register
  - Simpan register kelokasi memori 513

Korelasi

- Terlihat hubungan antara ekspresi bahasa tingkat tinggi dengan bahasa mesin.
- Dalam bahasa tingkat tinggi, operasi dinyatakan dalam bentuk aljabar singkat menggunakan variabel.
- Dalam bahasa mesin hal tersebut diekpresikan dalam operasi perpindahan antar register

Jenis-jenis Kumpulan Unik Set Intruksi

- Pengolahan data (data processing)
  - Meliputi operasi-operasi aritmatika dan logika.
    - Op. aritmatika: kemampuan komputasi pengolahan data numerik
    - Intruksi Logika: pengolahan data lain. (bit-bit word sebagai bit, bukan sebagai bilangan)
- Perpindahan data (data movement)
  - Berisi intruksi perpindahan data antar register maupun modul I/O.
- Penyimpanan data (data storage)
  - Berisi intruksi-intruksi penyimpanan ke mempri
  - Data ini akan digunakan untuk operasi berikutnya, minimal untuk ditampilkan pada layer.
- Kontrol aliran program (program flow control)
  - Berisi intruksi pengontrolan operasi dan pencabangan.
  - Intruksi ini untuk pengontrolan status dan mengoperasikan pencabangan ke set intruksi lain.

Jumlah Alamat

- Jumlah register atau alamat yang digunakan dalam operasi CPU tergantung format operasi masing – masing CPU.
- Ada format operasi yang menggunakan 3, 2, 1 dan 0 register.
- Umumnya yang digunakan adalah 2 register dalam suatu operasi.
- Desain CPU saat ini telah menggunakan 3 alamat dalam suatu operasi, terutama dalam MIPS (*million instruction per secon*).

Jumlah Alamat

- Alamat per instruksi yang lebih sedikit akan membuat instruksi lebih sederhana dan pendek, tetapi lebih sulit mengimplementasikan fungsi-fungsi yang kita inginkan.
- Karena instruksi CPU sederhana maka rancangan CPU juga lebih sederhana.
- Jumlah bit dan referensi per instruksi lebih sedikit sehingga fetch (Langkah awal eksekusi intruksi) dan eksekusi lebih cepat.
- Jumlah instruksi per program biasanya jauh lebih banyak

Jumlah Alamat

- Pada jumlah alamat per instruksi banyak, jumlah bit dan referensi instruksi lebih banyak sehingga waktu eksekusi lebih lama.
- Diperlukan register CPU yang banyak, namun operasi antar register lebih cepat.
- Lebih mudah mengimplementasikan fungsi – fungsi yang kita inginkan.
- Jumlah instruksi per program jauh lebih sedikit.
- Untuk lebih jelas perhatikan contoh instruksi – instruksi dengan jumlah register berbeda untuk menyelesaikan persoalan yang sama

Intruksi dengan 3 Alamat

Contoh penggunaan set instruksi dengan alamat 3 untuk menyelesaikan operasi berikut: **Y = (A – B) ÷ (C + D* E)**

Instruksi		Komentar
SUB	Y, A, B	Y =A – B
MPY (Multiply)	T, D, E	T= D x E
ADD	T, T, C	T=T + C
DIV	Y, Y, T	Y= Y ÷ T

"Spesifikasi Instruksi 3 Alamat" :

- Simbolik : a = b + c
- Format alamat : hasil, operand1, operand2
- Digunakan dalam arsitektur MIPS (Microprocessor without Interlocked Pipeline Stages)
- Memerlukan word pandang dalam satu instruksi

Intruksi dengan 2 Alamat

Y = (A – B) ÷ (C + D* E)

Instruksi		Komentar
MOVE	Y, A	Y = A
SUB	Y, B	Y = Y – B
MOVE	T, D	T =D
MPY	T, E	T = T E
ADD	T, C	T=T + C
DIV	Y, T	Y = Y ÷ T

"Spesifikasi Instruksi 2 Alamat" :

- Simbolik : a = a + b
- Satu alamat diisi operand terlebih dahulu, kemudian digunakan untuk menyimpan hasilnya
- Tidak memerlukan instruksi yang panjang
- Jumlah instruksi per program akan lebih banyak daripada instruksi 3 alamat
- Diperlukan penyimpanan sementara untuk menyimpan hasil

Intruksi dengan 1 Alamat

$$Y = (A - B) \div (C + D^* E)$$

Instruksi		Komentar
LOAD	D	AC =D
MPY	E	AC = AC E
ADD	C	AC= AC + C
STOR	Y	Y = AC
LOAD	A	AC = A
SUB	B	AC = AC – B
DIV	Y	AC =AC ÷ Y
STOR	Y	Y= AC

"Spesifikasi Instruksi 1 Alamat" :

- Memerlukan alamat implisit untuk operasi
- Menggunakan register akumulator (AC) dan digunakan pada mesin lama

Intruksi dengan 0 Alamat

Y = (A – B) ÷ (C + D* E)

Instruksi	Keterangan	isi stack
PUSH	B	B
PUSH	A	B,A
SUB	A-B	(A-B)
PUSH	E	(A-B),E
PUSH	D	(A-B),E,D
MUL	D*E	(A-B),(D*E)
PUSH	C	(A-B),(D*E),C
ADD	C+(D*E)	(A-B),(C+D*E)
DIV	(A-B)/(C+(D*E))	(A-B)/(C+(D*E))

"Spesifikasi Instruksi 0 Alamat" :

- Seluruh alamat yang digunakan adalah implisit
- Digunakan pada organisasi memori, terutama operasi stack

Rancangan Set Instruksi

- Aspek paling menarik dalam arsitektur komputer adalah perancangan set instruksi, karena rancangan ini berpengaruh banyak pada aspek lainnya.
- Set instruksi menentukan banyak fungsi yang harus dilakukan CPU.
- Set instruksi merupakan alat bagi para pemrogram untuk mengontrol kerja CPU.
- Pertimbangan: Kebutuhan pemrogram menjadi bahan pertimbangan dalam merancang set instruksi

Masalah rancangan yang fundamental meliputi :

- Operation repertoire :
  - Berapa banyak dan operasi – operasi apa yang harus tersedia
  - Sekompleks apakah operasi itu seharusnya
- Data types :
  - Jenis data
  - Format data
- Instruction format
  - Panjang instruksi,
  - Jumlah alamat,
  - Ukuran field
- Registers
  - Jumlah register CPU yang dapat direferensikan oleh instruksi, dan fungsinya
- Addressing
  - mode untuk menspesifikasi alamat suatu operand

Jenis Operasi

- Transfer data
- Aritmetika
- Logika
- Input/Output
- Kontrol sistem dan transfer kontrol
- Konversi

Operasi Set Intruksi secara umum

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
Logical	AND	Perform logical AND
	OR	Perform logical OR
	NOT	(complement) Perform logical NOT
	Exclusive-OR	Perform logical XOR
	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

Operasi Set Intruksi secara umum

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
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)

Transfer Data

- Intruksi transfer data harus menetapkan
  - Lokasi operand sumber
  - Lokasi operand tujuan
  - Panjang data yang akan dipindahkan
  - Mode pengalamatanya: metode yang digunakan oleh instruksi komputer untuk menentukan alamat memori atau lokasi operand dalam memori.

Mode Pengalamatan

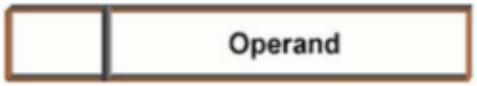
Mengatasi keterbatasan format instruksi

- Dapat mereferensi lokasi memori yang besar
- Mode pengalamatan yang mampu menangani keterbatasan tersebut
- Masing – masing prosesor menggunakan mode pengalamatan yang berbeda – beda.
- Memiliki pertimbangan dalam penggunaannya.
- Ada beberapa teknik pengalamatan
    a. Immediate Addressing
    b. Direct Addressing
    c. Indirect Addressing
    d. Register Addressing
    e. Register Indirect Addressing
    f. Displacement Addressing
    g. Stack Addressing

Immediate Addressing

- Operand benar – benar ada dalam instruksi atau bagian dari instruksi = Operand sama dengan field alamat.
- Umumnya bilangan akan disimpan dalam bentuk komplemen dua.
- Bit paling kiri sebagai bit tanda.
- Ketika operand dimuatkan ke dalam register data, bit tanda akan digeser ke kiri hingga maksimum word data.
- Contoh:

  ADD 5 ; tambahkan 5 pada akumulator



Direct Addressing

Pengalamatan langsung

- **Kelebihan:**

•Field alamat berisi *efektif address* sebuah operand.

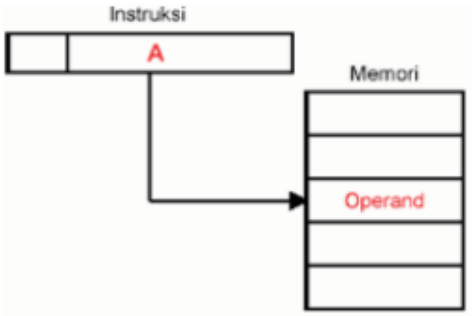
- Teknik ini banyak digunakan pada komputer lama dan komputer kecil.
- Hanya memerlukan sebuah referensi memori dan tidak memerlukan kalkulasi khusus.

- **Kelemahan:**

•Keterbatasan field alamat karena panjang field alamat biasanya lebih kecil dibandingkan panjang word.

- **Contoh:**

•ADD A ; tambahkan isi pada lokasi alamat A ke akumulator



Indirect Addressing

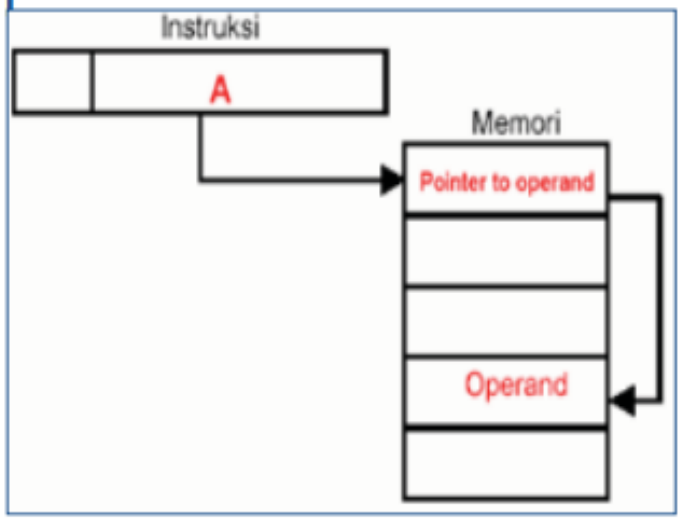
Mode pengalamatan tak langsung

- Field alamat mengacu pada alamat word di dalam memori, yang pada gilirannya akan berisi alamat operand yang panjang
- Contoh :

  ADD (A) ; tambahkan isi memori

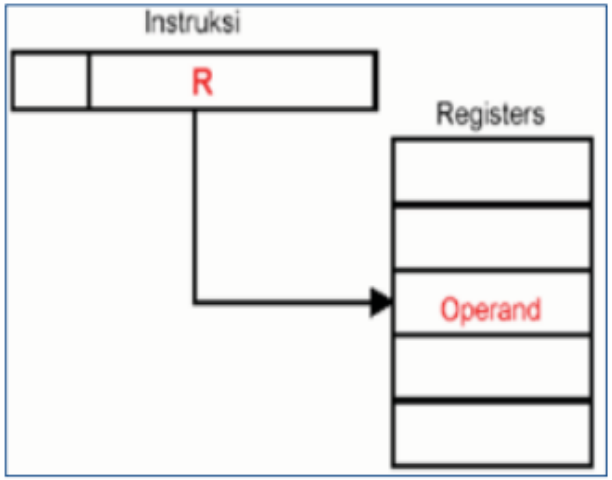
  yang ditunjuk oleh

  isi alamat A ke akumulator



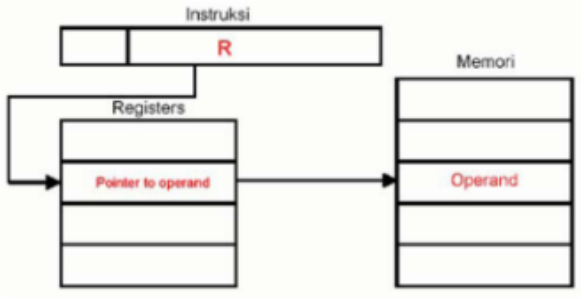
Register Addressing

- Metode pengalamatan register mirip dengan mode pengalamatan langsung.
- Perbedaannya terletak pada field alamat yang mengacu pada register, bukan pada memori utama.
- Field yang mereferensi register memiliki panjang 3 atau 4 bit, sehingga dapat mereferensi 8 atau 16 register general purpose.



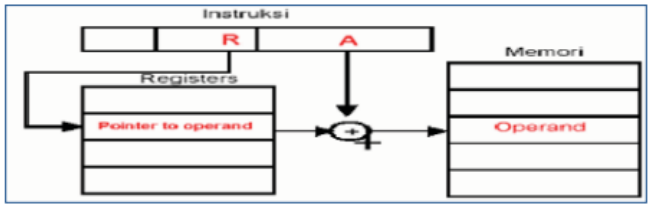
Register Indirect Addressing

- 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 ditunjuk 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



Displacement Addressing

- Menggabungkan kemampuan pengalamatan langsung dan pengalamatan register tidak langsung.
- Mode ini 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 ditambah isi register.

Stack Addressing

- Stack adalah array lokasi yang linier = pushdown list = last-in-first-out-queue.
- Stack merupakan blok lokasi yang terbalik.
  - Butir ditambahkan ke puncak stack sehingga setiap saat blok akan terisi secara parsial.
- Yang berkaitan dengan stack adalah pointer yang nilainya 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 pengalamatan register tidak langsung

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

Keterangan :

- A = isi suatu field alamat dalam instruksi
- EA = alamat aktual (efektif) sebuah lokas yang berisi operasi yang di referensikan
- (X) = isi lokasi X

Format-Format Intruksi

- Format instruksi menentukan layout bit suatu instruksi.
- Format instruksi harus mencakup opcode dan secara implisit atau eksplisit, nol operand atau lebih.
- Seluruh operand eksplisit direferensikan dengan menggunakan salah satu mode pengalamatan yang ada.
- Secara implisit atau eksplisit format harus dapat mengindikasikan mode pengalamatan seluruh operandnya.
- Pada sebagian besar set instruksi digunakan lebih dari satu format instruksi.

Panjang Instruksi

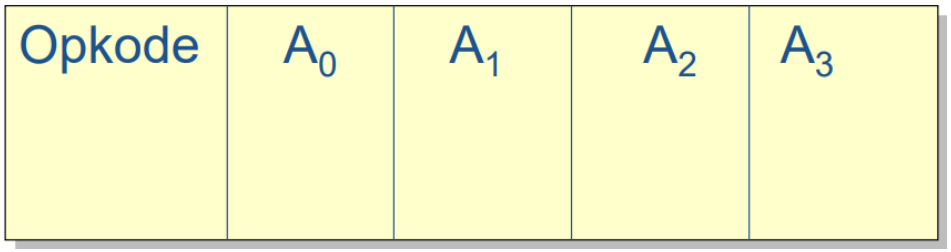
- Umumnya pemograman menginginkan opcode, operand, dan mode pengalamatan yang lebih banyak serta range alamat yang lebih besar
- Mode pengalamatan yang lebih banyak akan memberikan fleksibilitas yang lebih besar terhadap pemogram dalam mengimplementasikan fungsi-fungsi tertentu, seperti manipulasi table dan pencabangan yang berjumlah banyak.
- Dengan bertambahnya ukuran memori utama dan semakin banyaknya pemakaian memori virtual, pemogram akan dapat mengalamati jangkauan memori yang lebih besar.
- Kecepatan perpindahan memori tidak dapat diatasi dengan penambahan kecepatan processor.
- Karena memori akan dapat menjadi sebuah bottleneck apabila prosessor dapat mengeksekusi instruksi lebih cepat dari pada kecepatan untuk mengambil instruksi itu.
- Salah satu cara mengatasi masalah ini adalah dengan menggunakan cache memori atau dengan menggunakan instruksi-instruksi yang lebih pendek
- Instruksi 16 bit akan dapat diambil dua kali lebih cepat di bandingkan instruksi 32 bit namun mungkin akan dieksekusi dua kali lebih lambat

Fomat Intruksi

- Organisasi internal pada komputer dinyatakan oleh instruksi-instruksi yang dapat dijalankannya.
- Suatu instruksi merupakan suatu tata cara yang digunakan oleh komputer untuk menyatakan operasi-operasi seperti ADD, STORE, LOAD, MOVE dan BRANCH beserta untuk menentukan lokasi data di mana suatu operasi akan dikerjakan.

Format Alamat

- Pengkodean biner pada setiap komputer memiliki format kode instruksi tersendiri.
- Pada komputer terdahulu, setiap instruksi terdiri atas sebuah upcode dan empat field alamat.
  - A0 = alamat operand pertama
  - A1 = alamat operand kedua
  - A2 = alamat di mana hasil operand disimpan
  - A3 = Alamat dari instruksi berikutnya



Mode Pengalamat

- Suatu mode pengalamatan dapat digunakan untuk menentukan suatu alamat tempat untuk dimana operand akan di fetch.
- Beberapa teknik semacam ini dapat meningkatkan kecepatan pelaksanaan instruksi dengan menurunkan jumlah referensi pada memori utama dan meningkatkan jumlah referensi pada register kecepatan tinggi
- Mode pengalamatan ini menjabarkan suatu aturan untuk menginterpresetasikan atau memodifikasi field alamat dari instruksi sebelum operand di referensikan
- Pada semua mode pengalamatan lainnya, operand yang sesungguhnya tidak disimpan pada field alamat tetapi beberapa nilai di jabarkan dan di gunakan untuk menentukan operasi operand.

Kode Intruksi

- Selain dari representasi data, kode biner juga digunakan untuk membuat instruksi kontrol dalam komputer, yang disebut kode instruksi.
- Kode instruksi merupakan kelompok bit yang memberitahukan kepada komputer untuk menunjukan suatu operasi tertentu.
- Kode Instruksi dibagi dalam bagian-bagian, yang masing-masing bagian mempunyai interpretasi sendiri
- Bagian yang paling pokok adalah kode operasi (Operation Code / Opcode)
- Opcode adalah sekelompok bit yang menunjukan operasi seperti ADD, SUBTRACT, SHIFT, dan COMPLEMENT
- Bagian lain dari instruksi mencakup satu operasi (operand) atau lebih
- Operand adalah suatu nama yang digunakan untuk obyek instruksi dan mungkin data atau alamat yang mengatakan dimana data tersebut
- Untuk membuat kode instruksi dalam komputer harus kode biner. (seperti operasi LOAD dan Store)
- Load adalah meng-copy bilangan dari lokasi memori kedalam register
- Strore adalah meng-copy bilangan dari register kedalam lokasi memori

Soal

- Tulislah set intruksi (dan penjelasan) dengan jumlah Alamat 3, 2, 1, dan 0 untuk menyelesaikan operasi berikut:

Y=(A*B-C)+(D/E)

TERIMAKASIH

UNDIP | UNIVERSITAS DIPONEGORO
becomes an excellent research university