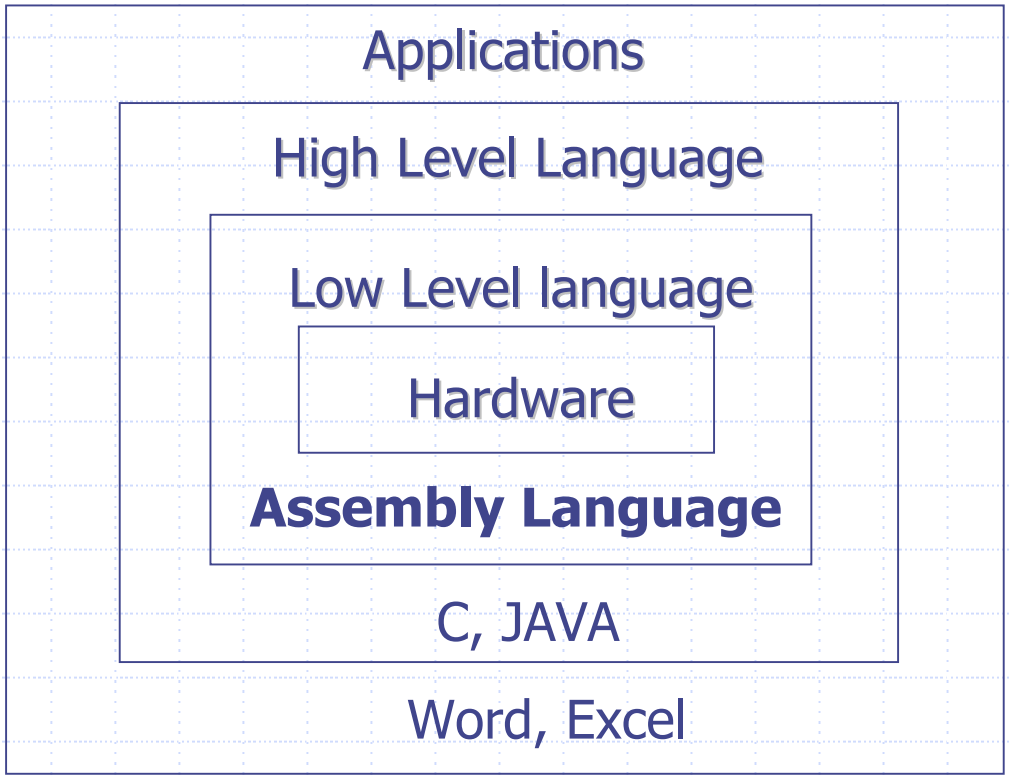
MK. PEMROGRAMAN SISTEM
Semester/SKS : 6/3

ASSEMBLER

Jurusan Sistem Komputer-S1
Universitas Gunadarma

Arsitektur Komputer
- dilihat dari segi Bahasa

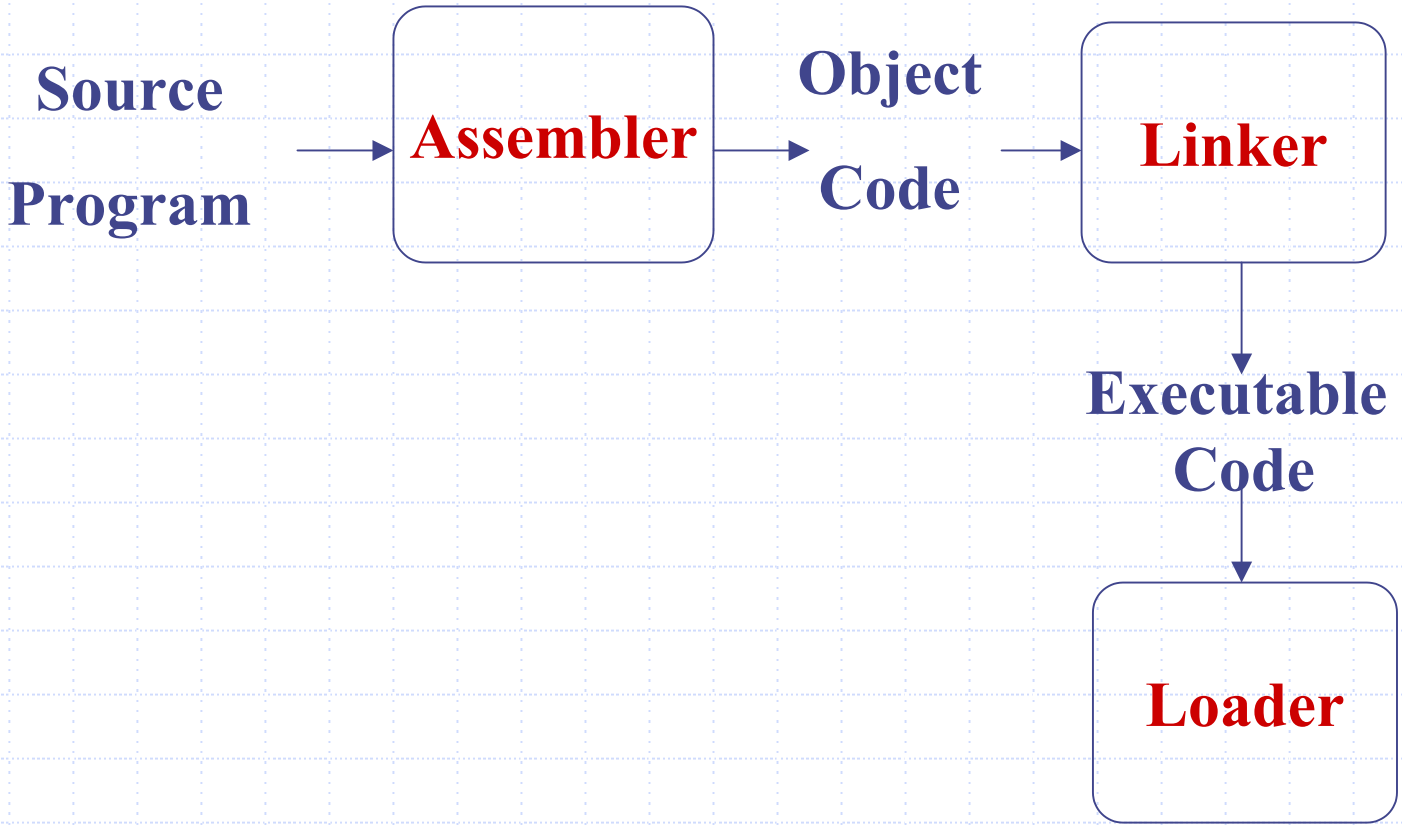


HHL
↓ Complier
Assembly Language
↓ Assembler
Machine Language

Assembly Language

- termasuk Low-Level Languange, dapat dibaca & dimengerti oleh manusia sebagai hasil representasi binary code yang dieksekusi oleh komputer
- Berorientasi pada mesin (machine dependent)
- bahasa yang dikendalikan oleh operasi primitive dalam data binari
- Operasi dasarnya meliputi data movement, addition, substraction, comparison, shifting dan branching

Peran Assembler



Karakteristik Dasar Assembly Languange

- Mnemonic Code
  - Lebih mudah dibandingkan operation codes (opcodes)
  - Terbagi menjadi bagian-bagian kecil yang memudahkan penulisan program
  - Mendukung pengindikasian kesalahan coding (mis. Kesalahan penulisan operation code.
  - Contoh :

Op code	Assembly mnemonic
00	STOP
01	ADD
02	SUB
03	MULT
04	LOAD

Karakteristik Dasar Assembly Languange (cont.)

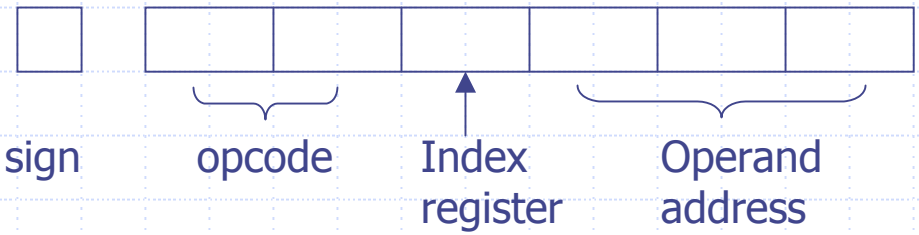
- Symbolic Operand Specification
  - Dihubungkan dengan data atau instruksi
  - Operand dapat ditentukan dalam bentuk symbolic reference
- Declaration of Data / Storage Area

Bentuk umum statement assembly languange

[Label]	**Mnemonic Op Code**	**Operand [operand]**
AGAIN	LOAD	NUMBER(4)

Statement Assembly Languange

- Imperative Statement
  - Menunjukkan aksi yang berlangsung selama eksekusi program assembly
  - Setiap imperative statement ditranslasikan ke dalam instruksi mesin
  - Format instruksi :



Statement Assembly Languange (cont.)

- Declarative Statement
  - Menunjukkan konstanta atau storage area pada program
  - Contoh :

```
A       DS      1           (DS = Declaration Storage)
ONE     DC      '1'         (DC = Declaration Constant)
```

- Assembler Directive Statement
  - Statement ini secara langsung mengarahkan assembler untuk mengambil alih aksi selama proses assembling program
  - Menunjukkan bagaimana input program assembly dibentuk
  - Contoh :

```
        START   100
        END
```

Proses Assembly

- Proses Translasi

{ Analysis of source text } + { Synthesis of target text } = { Translation from Source Text to Target Text }

Phase Analysis : mencari arti dari source text, dengan menggunakan struktur tatabahasa (literal, syntax, semantic)

Phase Synthesis : pemilihan machine operation code yang sesuai dengan mnemonic code

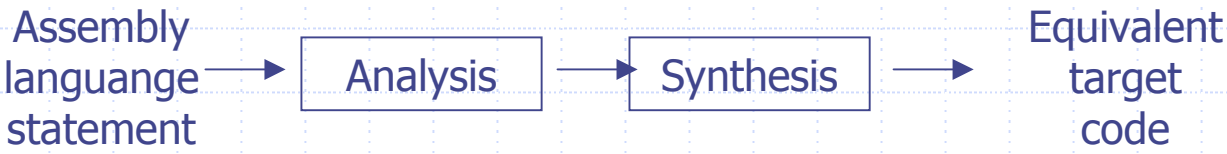
Skema Sederhana Assembly

- Phase Analysis
  - Mengisolasi/memisahkan label, mnemonic opcode dan operand field yang ada pada statement
  - Memasukkan simbol yang ditemukanpada label field dan alamat yang akan dituju machine word ke dalam Symbol Table
  - Melakukan validasi mnemonic opcode dengan melihat pada Mnemonic Table
  - Menentukan alamat yang dibutuhkan statement berdasar pada mnemonic opcode dan oerand field pada statement.
  - Proses perhitungan alamat awal machine word mengikuti target code yang dibangkitkan untuk statement tersebut (Location Counter-LC processing)

- Phase Syntesis
  - Menghasilkan machine operation code yang berkorespondensi dengan mnemonic opcode yang telah dicari pada Mnemonic Table
  - Menghasilkan alamat oerand dari Symbol Tabel
  - Melakukan sintesa instruksi machine

Pass Structure Assembly

Translasi statement demi statemet program assembly



Contoh :

```
                START   100
A               DS      1
B               DS      1
FIRST           READ    A
                READ    B
                LOAD    A
                SUB     B
                TRIM    LARGEB
                PRINT   A
                STOP
                PRINT   B
                STOP
                END
```

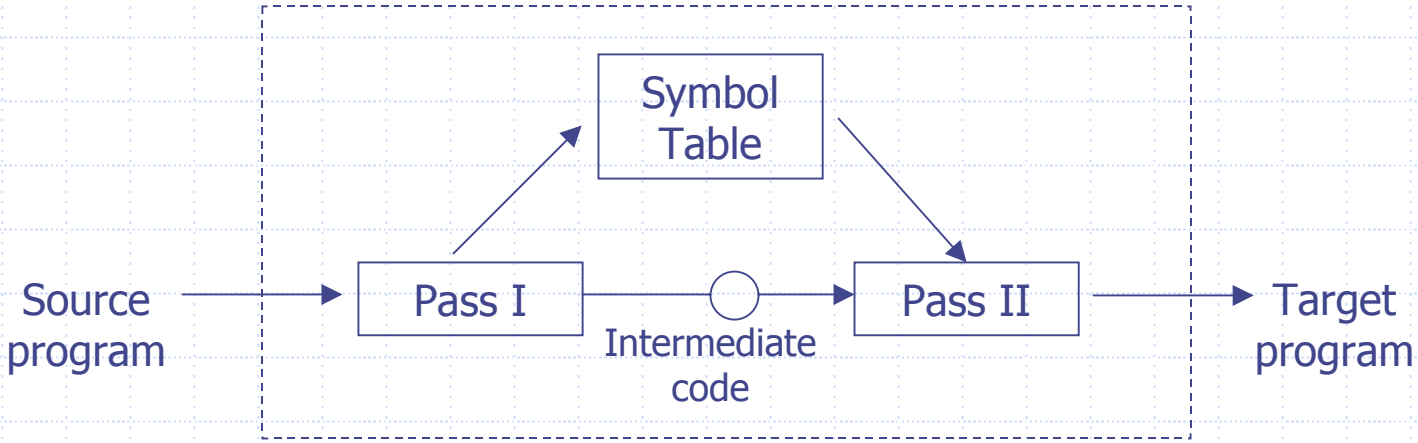
Forward refference

Pass Structure Assembly (cont.)

- Dalam pemrograman Assembly sering ditemui **Forward Refference**, yaitu symbol/label yang baru muncul setelah program tersebut menjadi acuan (lihat contoh pada slide sebelumnya…)
- Solusi untuk forward refference adalah melakukan proses terhadap source statement lebih dari satu kali secara beberapa tahap atau dikenal dengan konsep **translator pass**
- **Translator Pass** adalah penelusuran secara menyeluruh source program input oleh translator hingga mencapai equivalent representation
  - **Single Pass Translation**, translasi yang dilakukan statement demi statement
  - **Multi Pass Translation**, translasi yang dilakukan sekelompok statement yang membutuhkan banyak pass

Multi Pass Translation

Skema Multi Pass Translation



- unit source program mentranslasikan semua bagian program
- pada fase analisis (pass I), proses LC dikerjakan, simbol yang didefinisikan pada program dimasukkan ke dalam Symbol Table
- pass II, statement diproses untuk mensintesa target form
- cara lain, melakukan elaborasi pada hasil analisa source statement untuk mencegah duplikasi yang direpresentasikan dalam intermediate code

Single Pass Translation

Solusi forward refference dengan single pass translation adalah dengan cara:

- meninggalkan instruksi yang memuat forward refference dalam keadaan tidakselesaihingga alamat refference symbol diketahui.
- Alamat operand pada bagian akhir disimpan pada Table Incomplete Instruction (TII)
- Di akhir program, semua masukkan pada table diproses secara lengkap

Keuntungan : setiap source statement hanya diproses satu kali

Kekurangan : membutuhkan area storage yang besar karena fase analysis dan fase synthesis dijalankan bersamaan

Perancangan Two Pass Assembler

- Pass I
  - Memisahkan symbol, mnemonic code dan operand field
  - Menentukan kebutuhan storage untuk setiap assembly languange statement dan update location counter (LC)
  - Membangun table simbol
  - Merancang intermediate code untuk setiap assembly languange statement
- Pass II
  - Mensintesa target code dengan memproses intermediate code yang digenerate pada Pass I

Pass I Assembler

Dalam Pass I assembler digunakan beberapa table :

- OPTAB : table mnemonic opcode dan informasi lain yang terkait
- SYMTAB : symbol table
- LITTAB : literal table

Struktur data yang digunakan pada Pass I Assembler

Mnemonic opcode	Class	Machine opcode	Length
LOAD	1 (Imperative)	04	1
DS	2 (Declarative)	R#7	-
START	3 (Directive)	R#11	-
STORE	1 (Imperative)	05	1

OPTAB

Pass I Assembler (cont.)

Struktur data yang digunakan pada Pass I Assembler

Symbol	Address	Length	Other Information

SYMTAB