Assembly Language Programming

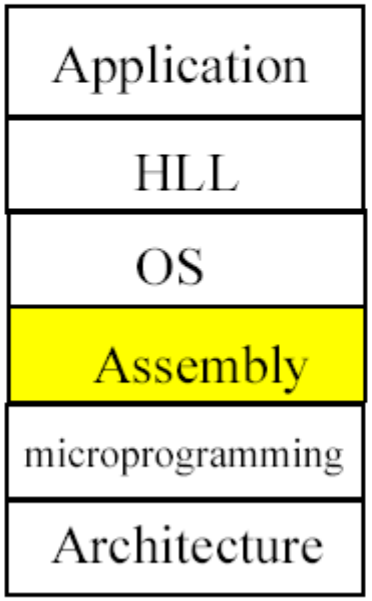
Assembly Programming

- Machine Language
  - binary
  - hexadecimal
  - machine code or object code
- Assembly Language
  - mnemonics
  - assembler
- High-Level Language
  - Pascal, Basic, C
  - compiler

Assembly Language Programming

Motivations

- Why do you learn assembly language?

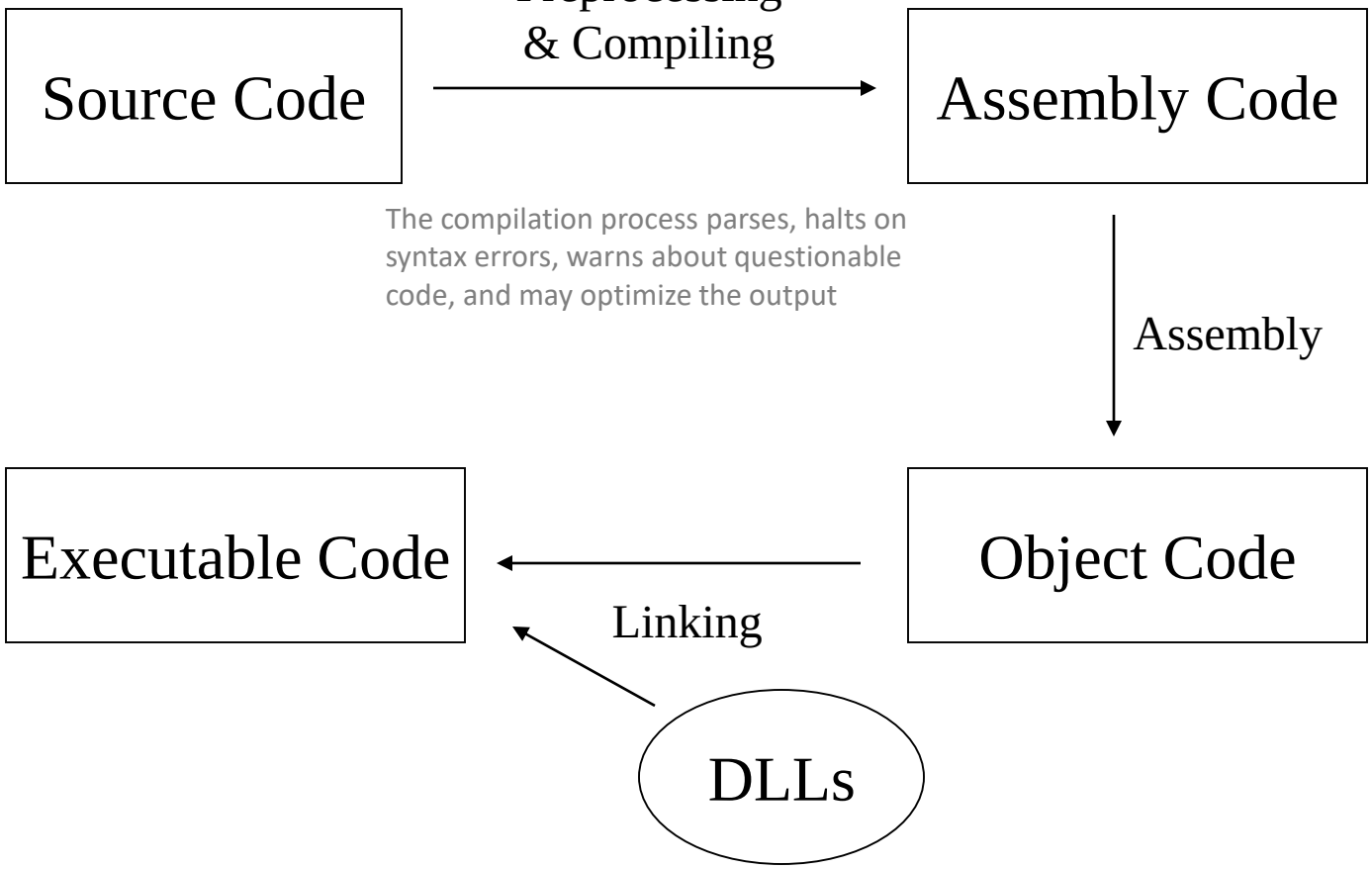


What Does It Mean to Disassemble Code?

Source code is written in a high level language such as C, Pascal, or C++. High level languages are human readable.

Preprocessing eliminates things that will not be included in the executable program (e.g., comments).

After preprocessing is complete, compiling turns source code into assembly code.

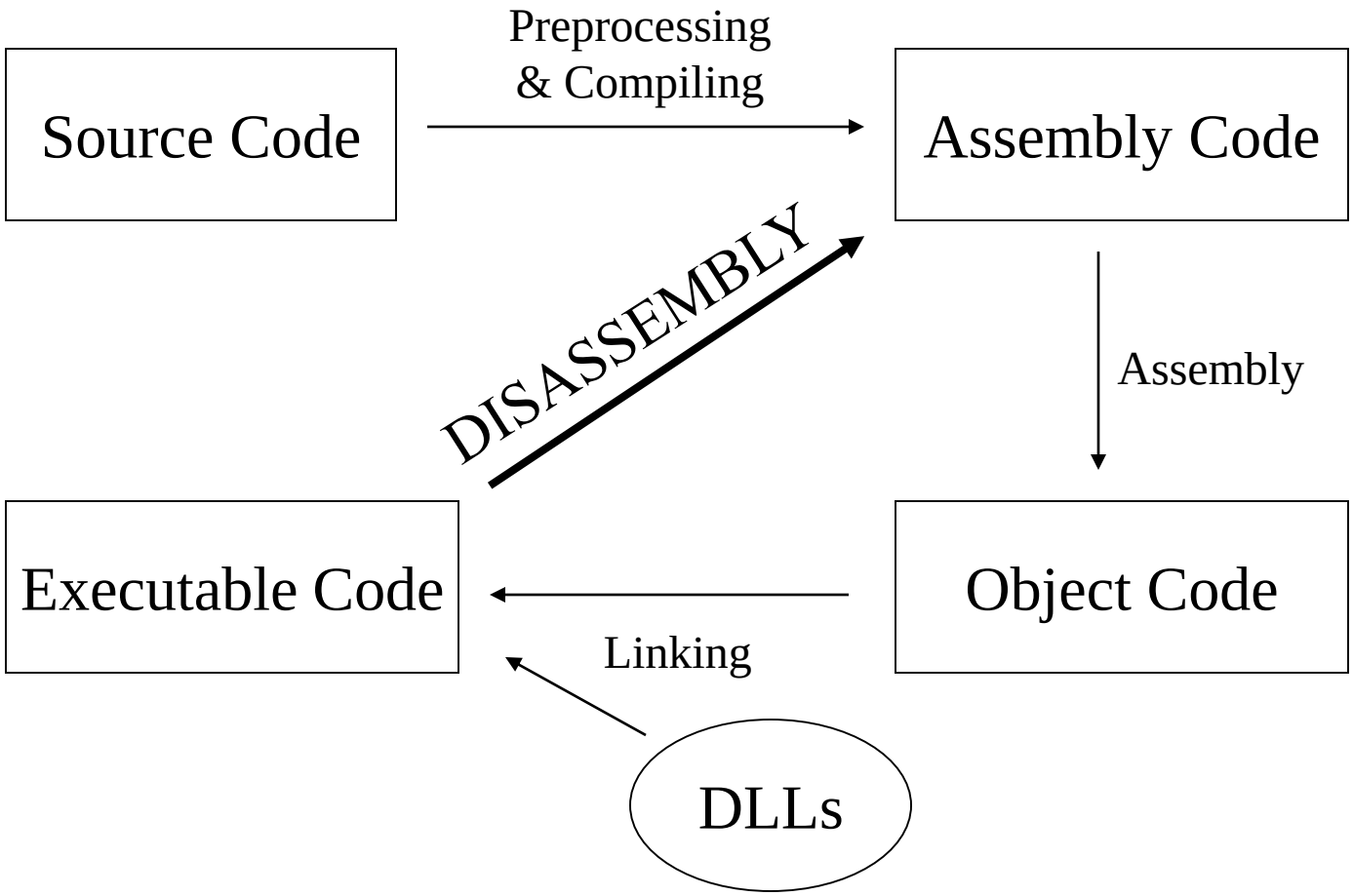


The compilation process parses, halts on syntax errors, warns about questionable code, and may optimize the output

The process of assembly turns assembly code into object code. Some optimization is performed

Linking combines all required object code, handles call addresses, etc. Programs almost universally make use of code libraries. Many "standard" functions are available through libraries: sockets, math functions, graphics functions, database functions, specific protocols, etc. The functions provided this way must be linked into the program

What Does It Mean to Disassemble Code?



- Kaynak kodu (**Source Code**), C, Pascal veya C ++ gibi yüksek seviyeli bir dilde yazılmıştır.
- Ön işleme (**Preprocessing**), çalıştırılabilir programa dahil edilmeyecek şeyleri ortadan kaldırır (Örn. Yorumlar). Aynı zamanda önişlemci direktiflerini (örneğin, # ifdef / # ifndef, #include, #define) ve temel parsing'i de işler. Genellikle, önişleme çıktısı doğrudan bir sonraki adıma beslenir: önişleme dosyaları genellikle kaydedilmez.
- Ön işleme tamamlandıktan sonra, derleme (**Compiling**) kaynak kodunu derleme koduna dönüştürür. Bu süreçteki en önemli adımdır. Derleme işlemi ayrıştırır, sözdizimi hatalarında durur, şüpheli kod hakkında uyarır ve çıktıyı optimize edebilir. Çıktı doğrudan assembler'a beslenir: sonuç genellikle bir dosya olarak kaydedilmez.
- Assembly işlemi, derleme kodunu **object** koda (amaç kod) dönüştürür. Bazı optimizasyonlar yapılır.
- **Linking**, gerekli tüm amaç kodunu birleştirir, çağrı adreslerini işler, vb. Programlar çoğu zaman evrensel kod kütüphanelerinden (*code libraries*) yararlanır. Kütüphaneler aracılığıyla birçok "standart" fonksiyon kullanılabilir: soketler, matematik fonksiyonları, grafik fonksiyonları, veritabanı fonksiyonları, özel protokoller, vb. Bu şekilde sağlanan fonksiyonlar programa bağlanmalıdır.

Computer Programming

- Machine Language vs Assembly Language
  - Makine dili veya amaç kodu (*object code*), bir bilgisayarın çalıştırabileceği tek koddur, ancak bir insan için üzerinde çalışmak neredeyse imkansızdır.
  - E4 27 88 C3 E4 27 00 D8 E6 30 F4 klavyeden girilen iki sayı toplamak için amaç kod
- Bir mikroişlemciyi programlarken, programcılar genellikle assembly dili kullanır
  - Bu, ikili veya onaltılık amaç kodları yerine komut kodları (anımsatıcılar) için 3-5 harfli kısaltmaları içerir.

Address	Hex Object Code				Mnemonics		Comment
					Op-Code	Operand	
0100	E4	27			IN	AL,27H	Input first number from port 27H and store in AL
0102	88	C3			MOV	BL,AL	Save a copy of register AL in register BL
0104	E4	27			IN	AL,27H	Input second number to AL
0106	00	D8			ADD	AL,BL	Add contents of BL to AL and store the sum in AL
0107	E6	30			OUT	30H,AL	Output AL to port 30H
0109	F4				HLT		Halt the computer

Edit, Assemble, Test, and Debug Cycle

- Bir ***editor*** kullanılarak programın kaynak kodu oluşturulur. Bu, görevi gerçekleştirmek için uygun komut anımsatıcılarını seçmek anlamına gelir.

- Editör tarafından oluşturulan kaynak kodu dosyasını inceleyen ve programdaki her komut için amaç kodunu belirleyen bir derleyici programı çalıştırılır. Assembly dili programlamasında buna *assembler* denir (MASM, DEBUG, vb.)

- Bilgisayar tarafından üretilen amaç kodu, hedef (*target*) bilgisayarın belleğine yüklenir ve ardından çalıştırılır (*run*).

- ***Debugging:*** hata kaynağını bulmak ve düzeltmek

- High-level programming Languages
  - Basic, Pascal, C, C++ vb. diller

Assembly Language

- Assembly language veya Assembler language
  - Bir bilgisayar veya diğer programlanabilen cihazlara yönelik bir alt-seviye programlama dilidir. **(low-level programming language)**
  - bu dilde genellikle mimarinin makine kod emirleri ile arasında güçlü bir (genellikle bire-bir) benzeşme vardır **(architecture's machine code instructions)**
  - (*Yüksek seviyeli programlama dillerinin tersine*) Her assembly dili, belirli bir bilgisayar mimarisine özgüdür.

High-level programming language

- High-level programming language
  - Genellikle çeşitli mimariler arasında taşınabilir **(portable)**
  - Fakat yorumlamaya veya derlemeye ihtiyaç duyar **(interpreting** or **compiling)**

Assembly Language

- Assembly language aynı zamanda
  - *Assembly*
  - *Assembler*
  - *ASM*
  - *Symbolic machine code*
  - *Assembly program*

  olarak da anılır.

Assembly Language

- Assembly dili, '*assembler*' olarak anılan bir utility program tarafından 'executable machine code' a dönüştürülür.

- Bu dönüşüm işlemi 'assembly' olarak anılır
  - *assembling the source code*

Assembly Language

- Semboller (*symbols*) tanımlayarak ve kullanarak donanımdaki bellek adreslerini (*memory adresses*) temsil etmeye imkan tanır
- ‘mnemonic’ olarak anılan özel sembol kullanılarak her bir alt-seviye makine emiri veya operasyon belirtilmiş olur.
- Tipik operasyonlar bir veya birden fazla operand gerektirir.

Assembly Programming

- Assembly dil emirleri 4 alandan oluşur

[label:] mnemonic [operands] [;comment]

- Label
- mnemonic, operands
  - MOV AX, 6764
- comment
  - ; this is a sample program

Model Definition

MODEL directive

– memory modelinin boyutunu seçer

- **MODEL MEDIUM**
  - Data must fit into 64KB
  - Code can exceed 64KB
- **MODEL COMPACT**
  - Data can exceed 64KB
  - Code cannot exceed 64KB
- **MODEL LARGE**
  - Data can exceed 64KB (but no single set of data should exceed 64KB)
  - Code can exceed 64KB
- **MODEL HUGE**
  - Data can exceed 64KB (data items i.e. arrays can exceed 64KB)
  - Code can exceed 64KB
- **MODEL TINY**
  - Data must fit into 64KB
  - Code must fit into 64KB
  - Used with COM files

Segments

- Segment definition:

The 80x86 CPU 4 tane segment registere sahiptir: CS, DS, SS, ES

- Segments of a program:

.STACK ; stack segmentin başlangıcına işaret eder

example:

```
.STACK    64    ; 64Byte'lık  yığın bellek alanı rezerve eder
```

.DATA ; data segmentin başlangıcına işaret eder

example:

```
.DATA1  DB    52H    ; DB direktifi bayt boyutunda parçalar halinde
                     bellek ayırır
```

.CODE ; Code segmentin başlangıcına işaret eder.

Assemble, Link, and Run Program

STEP	INPUT	PROGRAM	OUTPUT
1. Edit the program	keyboard	editor	myfile.asm
2. Assemble the program	myfile.asm	MASM or TASM	myfile.obj myfile.lst myfile.crf
3. Link the program	myfile.obj	LINK or TLINK	myfile.exe myfile.map

Assemble, Link, Run Files

STEP	INPUT	PROGRAM	OUTPUT

.asm – source file

.obj – machine language file

.lst – list file

- it lists all the Opcodes, Offset addresses, and errors that MASM detected

.crf – cross-reference file

- programda kullanılan tüm sembollerin ve etiketlerin alfabetik bir listesi ve bunların referans alındığı program satır numaraları

.map – map file

- kod veya veri için birçok segment olduğunda kullanılan baytların konumunu ve sayısını görmek için

Control Transfer Instructions

- NEAR – Kontrol, aktif code segment içerisindeki bir bellek lokasyonuna transfer edildiğinde (±32K bytes)

- FAR – Kontrol, aktif code segment dışında bir yere transfer edildiğinde

CS:IP – Bu register çifti her zaman sıradaki icra edilecek emirin adresine işaret eder

- NEAR Jump: IP güncellenir, CS aynı kalır
  (In a NEAR jump, IP is updated, CS remains the same)
- FAR Jump : hem CS hem de IP güncellenir
  (In a FAR jump, both CS and IP are updated)

Control Transfer Instructions

- Conditional Jumps
- Short Jump
  - Tüm koşullu dallanmalar kısa dallanmadır
  - Hedefin adresi, IP'nin –128 ile +127 byte aralığında olmalıdır
  - Koşullu dallanma 2-byte lık emirdir
    - Bir byte'ı dallanma şartının opcode'udur
    - 2.byte ise 00 ile FF arasında bir sayıdır
      - 256 olası addres:
        - forward jump to +127
        - backward jump to –128

Data Types and Data Definition

- Assembler data directives
  - **ORG** (origin) – to indicate the beginning of the offset address
    - *example:*

```
        ORG     0010H
```

  - **DB** (define byte) – allocation of memory in byte-sized chunks
    - *example:*

```
DATA1   DB      25          ;decimal
DATA2   DB      10001001B   ;binary
DATA3   DB      12H         ;hex
DATA4   DB      '2591'      ;ASCII numbers
DATA5   DB      ?           ;set aside a byte
DATA6   DB      'Hello'     ;ASCII characters
DATA7   DB      "O' Hi"     ;ASCII characters
```

Data Types and Data Definition

- Assembler data directives

  - **DUP** (duplicate) – to duplicate a given number of characters
    - *example:*

```
DATA1    DB     0FFH, 0FFH, 0FFH, 0FFH        ;fill 4 bytes with FF
```

Can be replaced with:

```
DATA2    DB     4 DUP(0FFH)                   ;fill 4 bytes with FF

DATA3    DB     30 DUP(?)                     ;set aside 30 bytes

DATA4    DB     5 DUP (2 DUP (99))            ;fill 10 bytes with 99
```

Data Types and Data Definition

- Assembler data directives

  - **DW** (define word) – allocate memory 2 bytes (one word) at a time
    - *example:*

```
DATA1    DW    342                ;decimal
DATA2    DW    01010001001B       ;binary
DATA3    DW    123FH              ;hex
DATA4    DW    9,6,0CH, 0111B,'Hi'  ;Data numbers
DATA5    DW    8 DUP (?)          ;set aside 8 words
```

  - **EQU** (equate) – define a constant without occupying a memory location
    - *example:*

```
COUNT    EQU    25

;COUNT can be used in many places in the program
```

EXE vs. COM

- COM files
  - Smaller in size (max of 64KB)
  - Does not have header block

- EXE files
  - Unlimited size
  - Do have header block (512 bytes of memory, contains information such as size, address location in memory, stack address)