Assembly Language Lecture Series:

Why Assembly Language

Sensei RL Uy, College of Computer Studies,
De La Salle University, Manila, Philippines

Copyright Notice

Talking Points

01 Levels of Program Code

02 Code Challenge

03 Why Learn Assembly Language

04 Misconceptions about Assembly Language

Recap

Below your program is divided into three:

1 **Application Software**

Written in high-level language

3 **Hardware**

Processor, memory, I/O controllers

2 **System Software**

1. Compiler: translates HLL code to machine code
2. Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources

Levels of Program Code

1. High-level Language

- Level of abstraction closer to problem domain
- Provides for productivity and portability

2. Assembly Language

Textual representation of instructions

3. Hardware Representation

- Binary digits (bits)
- Encoded instructions and data

Levels of Program Code

High Level (C/java)	Assembly Language (x86-64)	Machine Code (hex)

int main()
{
int i;
i = 0;
for (int c=10; c>=0; c--)
i=i+c;
}
	xor rsi, rsi
	mov rcx, 0x0A
	L1: add rsi, rcx
	loop L1
		48 31 F6
		48 07 C1 0A 00 00 00
		48 01 CE
		E2 FB

Compiler	Assembler	

High Level (C/java)	Assembly Language (RISC-V)	Machine Code (hex)

int main()
{
int i;
i = 0;
for (int c=10; c>=0; c--)
i=i+c;
}
	add x10, x0, x0
	addi x11, x0, 0x0000000A
	L1: add x10, x10, x11
	addi x11, x11, -1
	bnez x11, L1
		00 00 00 53
		00 A0 05 93
		00 B5 05 33
		FF F5 85 93
		FE 05 9C E3

Compiler	Assembler	

Levels of Program Code

Assembly Language (x86-64)	Machine Code (hex)
xor rsi, rsi mov rcx, 0x0A L1: add rsi, rcx loop L1	48 31 F6 48 07 C1 0A 00 00 00 48 01 CE E2 FB

Address	Contents (binary)	Contents (hex)
4014EE	1111 1011	FB
4014ED	1110 0010	E2
4014EC	1100 1110	CE
4014EB	0000 0001	01
4014EA	0100 1000	48
4014E9	0000 0000	00
4014E8	0000 0000	00
4014E7	0000 0000	00
4014E6	0000 1010	0A
4014E5	1100 0001	C1
4014E4	0000 0111	07
4014E3	0100 1000	48
4014E2	1111 0110	F6
4014E1	0011 0001	31
4014E0	0100 1000	48

Operation Code (Opcode)

Why Learn Assembly Language?

Speed

Assembly language program is generally faster than other programs written in high-level language.

Space

Assembly language program consumes the smallest amount of memory.

Capability

Some programming techniques are difficult or impossible to implement in a high-level language (e.g. , pipelining, minimizing branch hazards, parallel computing, etc.)

Knowledge

Techniques learned in assembly language programs will help you write better programs, even when using high-level language.

Code Challenge

Code Challenge

Write a program to add a list of integers found in variable NUMB. The number of integers in the list is found in variable COUNT. Store the sum in variable ANS.

Code Challenge: C

```
int main()
{
	int i;
	int ANS = 0;
	int COUNT = 5;
	int NUMB[] = {12,22,32,42,52}
	for (i=0; i<=COUNT; i++)
		ANS += NUMB[i];
	printf(“Answer = %d”, ANS);
	return 0;
}
```

Code Challenge: Python

```
ANS = 0
COUNT = 5
NUMB = [12,22,32,42,52]
for i in NUMB:
        ANS += i
print (“Answer = “, ANS)
```

Code Challenge:
x86-64 Assembly Language

```
%include "io64.inc"
section .data
ANS dq 0
COUNT dq 05
NUMB dq 1,2,3,4,5
section .text
global CMAIN
CMAIN:    xor rbx, rbx
          mov rcx, [COUNT]
          lea rsi, [NUMB]
L1:       add rbx, [rsi]
          add esi, 8
          loop L1
          mov [ANS], rbx
          PRINT_DEC 8, ANS
          xor rax, rax
          ret
```

Code Challenge:
RISC-V Assembly program

```
.globl main
.data
  ANS: .word  0
  COUNT: .word 5
  NUMB: .word 1,2,3,4,5
.text
main:
    addi x5, x0, 0     # initialize answer
    lw x6, COUNT       # count
    la x7, NUMB        # x7 points to NUMB
L2: lw x8, (x7)
    add x5, x5, x8     # add
    addi x6, x6, -1    # decrement count
    beq x6, x0,  L1
    addi x7, x7, 4     # increment pointer to next number
    j L2
L1: la x10, ANS
    sw x5, (x10)       # store result to ANS
```

```
#print
  mv a0, x5
  li a7, 1
  ecall
#exit program
  li a7, 10
  ecall
```

Misconceptions about Assembly Language

- It is hard to learn.
- It is hard to read and understand.
- It is hard to debug.
- Assembly language programming is time consuming.
- Improved compiler has eliminated the need for assembly language programming.

Misconceptions about Assembly Language

- If you want the program to run faster, you should use better algorithm rather than switch to assembly language.
- Computers are very fast. Thus, no need for assembly language.
- Computers have so much memory; saving memory space using assembly language is not needed
- Assembly language is not portable.

Misconceptions about Assembly Language

- Who says so? Heresay?
- Pure lies, misconceptions, myths, half-truth
- Maybe they don’t know assembly language or they have only few lines of code experience in assembly language…
- What they don’t know scares them … (is assembly language programming a magic? a myth?)

Brief Rebuttal

- **It is hard to learn.**
  so is learning a new language … or learning a dissimilar language (e.g., JAVA/Python vs. PROLOG)

- **It is hard to read and understand.**
  yes, if you don't know assembly language programming … with experience, you should be able to read and understand assembly language quite easily … as with everything else, right?

Brief Rebuttal

- **It is hard to debug.**
  hmm… do you still remember debugging your first High-level program?

- **Assembly language programming is time consuming.**
  Codes are long, there is no library routines, but … you can develop your own subroutines!

Brief Rebuttal

- **Improved compiler has eliminated the need for assembly language programming**
  As of now, even the vastly improved compiler cannot beat hand-coded assembly language program. Technique in writing assembly language program is different from high level programming.

Brief Rebuttal

- **If you want the program to run faster, you should use better algorithm rather than switch to assembly language.**
  what if the algorithm that is implemented in high level is already the best option and it is still slow? Do you know that any algorithms that can be implemented in high-level can also be implemented in assembly language but not vice versa?

  Think: Parallel Computing

Brief Rebuttal

- **Computers are very fast, thus, no need for assembly language.**

  Why will people buy a computer that is slightly faster than their current computer but they won't spend some extra time writing their program in assembly so that it runs faster in their current computer?

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Cycle}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Program}} \times \frac{\text{Seconds}}{\text{Instruction}}$$

Brief Rebuttal

- **Computers have so much memory, saving memory space using assembly language is not needed.**
  “Humans are by nature greedy, if you give them an inch, they will take a mile.” It used to be, that computers have only 4MB of memory, and programs run okay. Right now, we have 16GB of memory, and the programs have become bloated (“fatware”) … note that embedded devices and mobile devices still have limited memory

Brief Rebuttal

- **Assembly Language is not portable.**
  This is an undeniable fact. But so is high-level languages. C programs written in Windows/DOS-based might not work in Apple IOS/Linux and vice versa. Just need to recompile!