香港中文大學
The Chinese University of Hong Kong

CSCI2510 Computer Organization

Tutorial 03: MASM Program Structure, Debugging, and Addressing Mode

Bentian Jiang

btjiang@cse.cuhk.edu.hk

Outline

- Program Structure (quick review)
  - Assembler Directives
  - Data Segment
  - Code Segment
- Basic Debugging Operations
  - Print
  - IDE built in
- Addressing mode

Program Structure Review

```
.386
.model flat, stdcall
option casemap:none
include windows.inc
include kernel32.inc
include user32.inc
```

Assembler Directives

```
.data
MsgCaption db "CSCI2510 Tutorial", 0
MsgBoxText db "Hello, World!", 0
```

Data Segment

```
.code
start:
    invoke MessageBox, NULL,addr MsgBoxText, addr MsgCaption, MB_OK
    invoke ExitProcess,NULL
end start
```

Code Segement

Program Structure Review

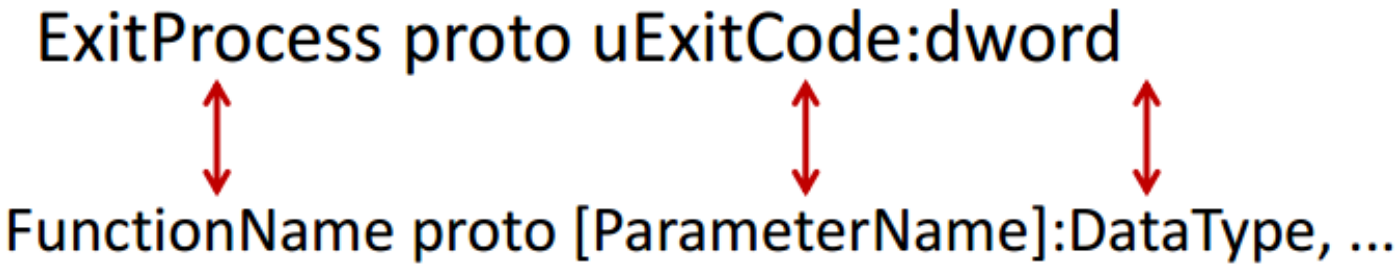
- Assembler Directives
  - Telling the assembler what to do:
    - Option, configuration, syntax etc…
- Data Segment
  - Declare and apply some memory space in primary memory (e.g. RAM)
  - Assign value to corresponding data object
- Code Segment
  - State the following segment is the program assembly code
  - Call function with arguments in data segment

Basic Debugging Operations

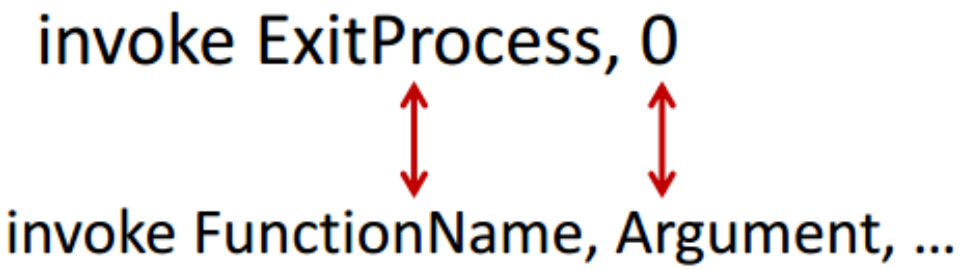
- Print the log
  - Function in MASM
  - Three Print Functions
    - crt_printf
    - StdOut
    - MessageBox
- Built-in debugging (local windows debugger)

Call a Function in MASM

- Prototype (Declaration):



- Usage (call the function)



Call a Function in MASM

- invoke MessageBox, NULL, addr szText, addr szCaption, MB_OK
  - push NULL
  - push addr szText
  - push addr szCaption
  - push MB_OK
  - call MessageBox

Three Printout Functions

- crt_printf (c style function)
  - Include “msvcrt.inc” and “msvcrt.lib” (for C functions)
  - Declaration: crt_printf proto format:dword
  - Usage:

```
.data
PrintFormat db "String: %s, Int: %d", 10, 0
String db "Tutorial", 0
Number db 2

.code
start:
    invoke crt_printf, addr PrintFormat, addr String, Number
    invoke ExitProcess, NULL
end start
```

- StdOut (standard print in MASM32)
- MessageBox (Win32 message box)

Three Printout Functions

- crt_printf (c style function)
- StdOut (standard print in MASM32)
  - Include “masm32.inc” and “masm32.lib”
  - Declaration: StdOut proto lpszText:dword
  - Usage:

```
.data
Message db "CSCI2510 Tutorial 2", 10, 0

.code
start:
    invoke StdOut, addr Message
    invoke ExitProcess, NULL
end start
```

- MessageBox (Win32 message box)

Three Printout Functions

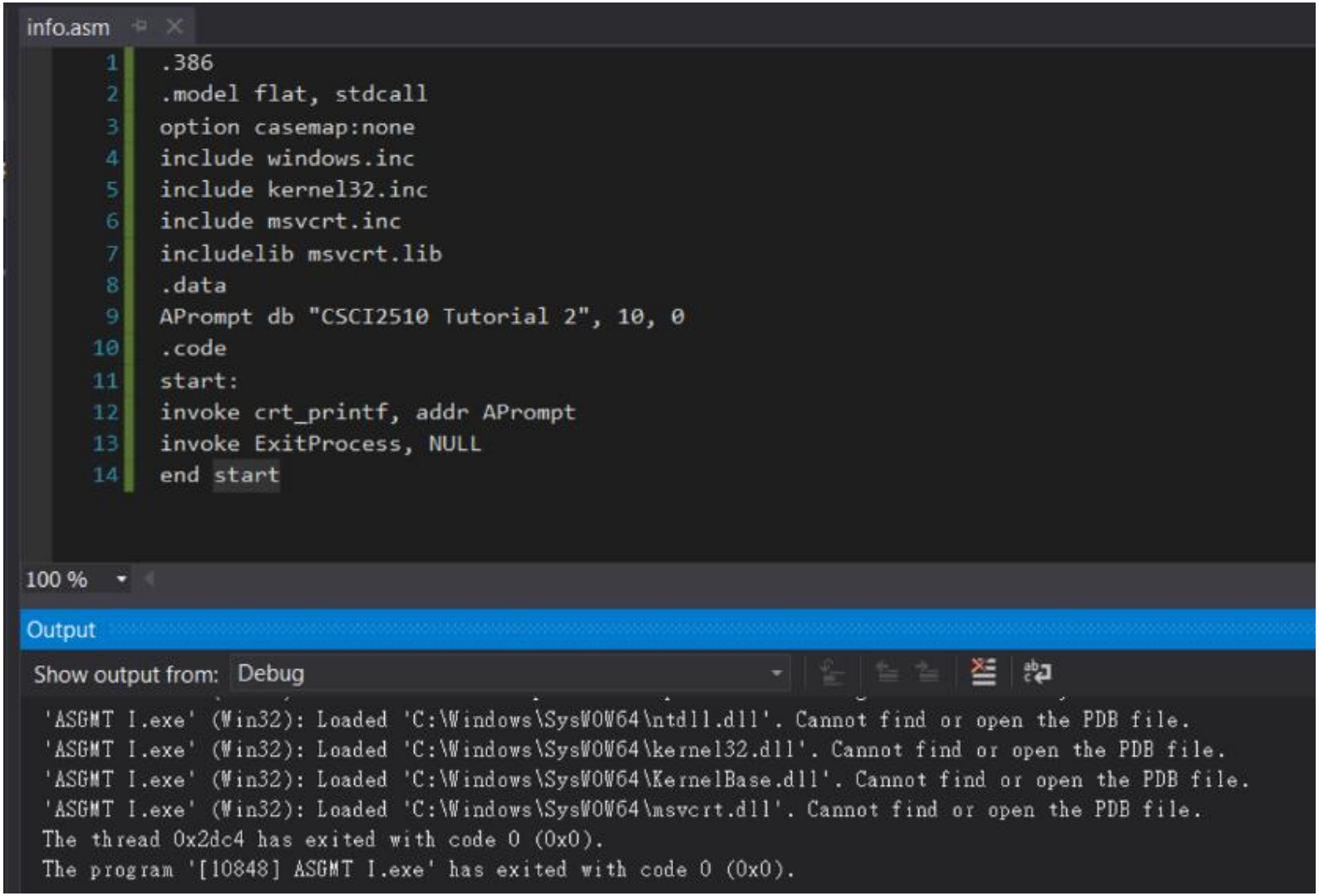
- crt_printf (c style function)
- StdOut (standard print in MASM32)
- MessageBox (Win32 message box)
  - Include “user32.inc”
  - Declaration:
    - MessageBox PROTO hwnd:DWORD, lpText:DWORD, lpCaption:DWORD, uType:DWORD
  - Usage:

```
.data
MsgCaption db "CSCI2510 Tutorial 1", 0
MsgBoxText db "Win 32 Assembly is Great!", 0

.code
start:
    invoke MessageBox, NULL, addr MsgBoxText, addr MsgCaption, MB_OK
    invoke ExitProcess, NULL
end start
```

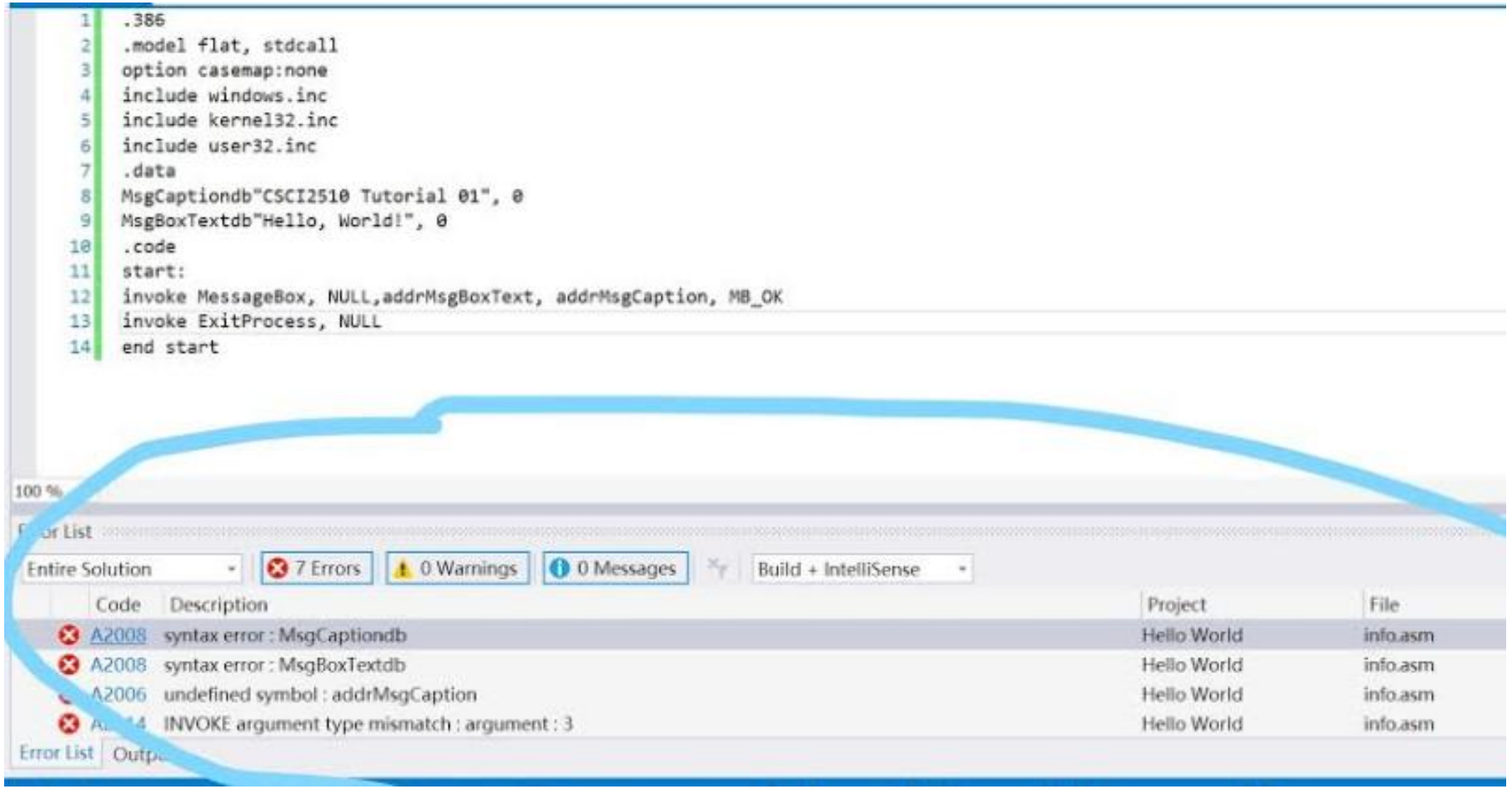
Basic Debugging Operations

- Built-in debugging (local windows debugger)



Basic Debugging Operations

- Built-in debugging (local windows debugger)



Check the register

- Steps:
  - Add breakpoint(s) at some instruction(s)
  - Start Debugging (F5)
  - Menu Debug --> Window --> Registers to watch the value of each register
  - Put the name of the register into the watch table

- Video Demonstration:
  - http://www.cse.cuhk.edu.hk/~mcyang/csci2510/Tut01%20Environment%20Setup%20for%20MASM.mp4
  - 2:40 ~ 4:26

Check the register

```
35      ; Addressing Mode Experiments
36
37      ; immediate
38      mov eax, 258147                 ; decimal
39      mov al, 11101111b               ; binary
40      mov eax, 0a34abcdfh             ; hexadecimal
41      mov eax, MONEY                  ; symbolic constant
42      mov eax, offset FOURBYTE        ; data address label
43      ;mov eax, FOURBYTE?
```

110 %

Watch 1

Name	Value	Type
eax	258147	unsigned long

Autos | Locals | Memory 1 | Threads | Modules | Watch 1

Addressing mode

- Immediate addressing
  - MOV EAX, 25
  - MOV EAX, 25H
- Direct addressing
  - MOV EAX, [012ABCD67H]
  - MOV EAX, [LOCAL];
    - optional [] if LOCAL is an address label
- Register addressing
  - MOV EAX, EDX
    - Difference between MOV EAX, [EBP] and MOV EAX, EDX?
- Comprehensive Addressing Mode
  - MOV EAX, [EBP+ESI*4+28]

Addressing mode

- Quick Reference for assembly addressing modes
  - http://www.cse.cuhk.edu.hk/~mcyang/csci2510/Lec04%20Machine%20Instructions.pdf
  - https://www.tutorialspoint.com/assembly_programming/assembly_addressing_modes.htm
- Addressing mode is a very important part, everyone is expected to understand it.

Summary

- Program Structure Review
- Basic Debugging Operations
- Addressing mode