

# Lab 2b

CSE141L

5/7/2007

# Assembler

- Inputs
  - src \*.s (benchmark file)
  - text\_addr <addr>
  - data\_addr <addr>
- Outputs
  - \$name\_i.coe
  - \$name\_d.coe
- Example
  - *prompt> asm -src bench.s -text\_addr 0x100 -data\_addr 0x200*
  - Outputs: bench\_i.coe, bench\_d.coe

# A Sample Assembly File

`.text`

```
la $1, table0 // load the address of a label table0 (pseudo instruction)
lw $2, $1 // $2 <= 0x000COFFEE
lw $3, table0 // load the value at label table0 (pseudo instruction), $3 <= 0x000COFFEE
lw $4, 3(table0) // $4 <= table1
lw $5, 1($4) // $5 <= 0x1DEADBEEF
sw $5, 1($1) // 0x001COFFEE is overwritten to 0x1DEADBEEF
li $6, 0xCOFFEE // load immediate (pseudo instruction), $6 <= 0xCOFFEE
```

`.data`

```
table0:
.word 0x000COFFEE
.word 0x001COFFEE
.word 0x002COFFEE
.word table1

table1:
.word 0x0DEADBEEF
.word 0x1DEADBEEF
.word 0x2DEADBEEF
.word 0x3DEADBEEF
```

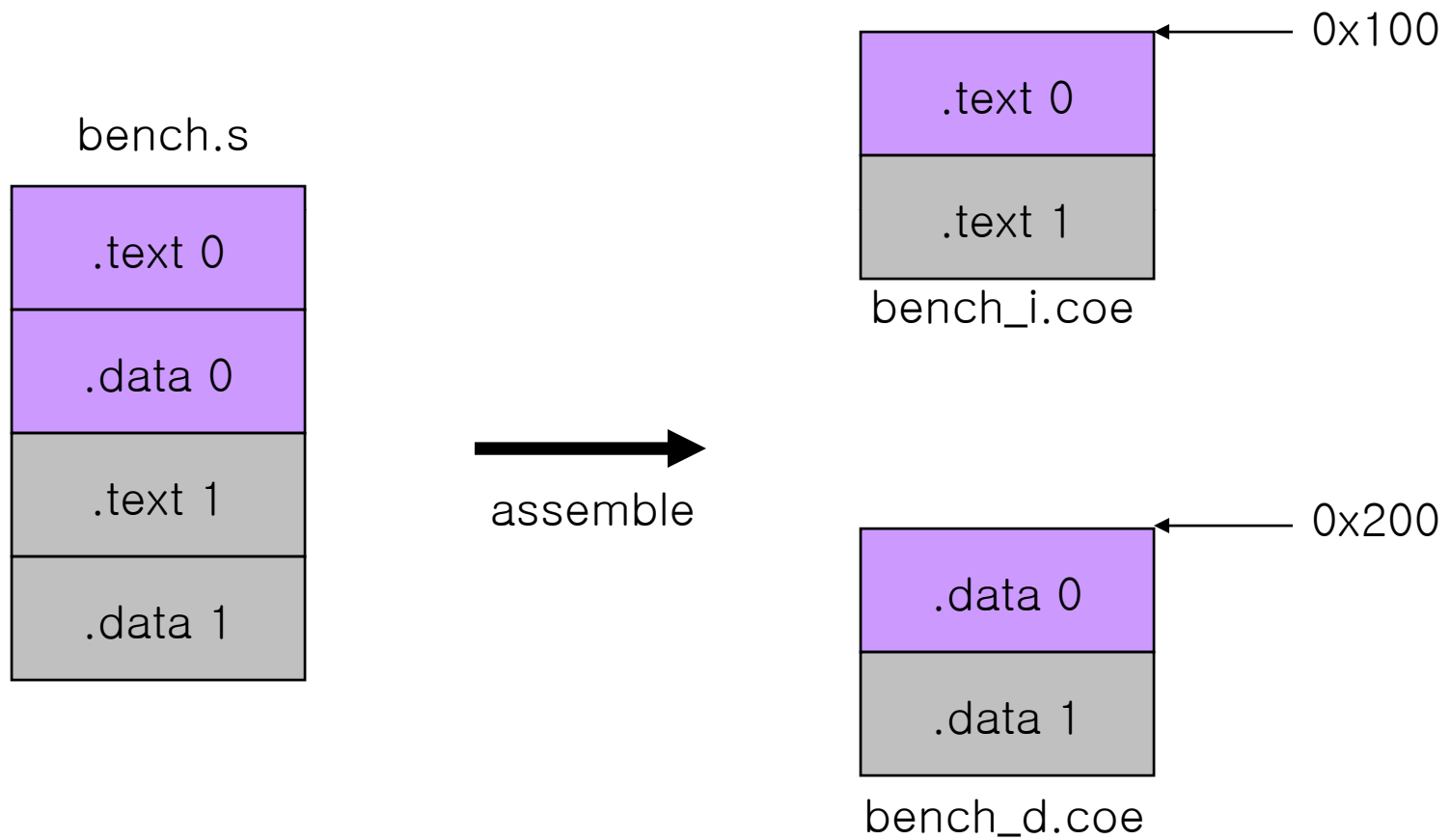
`.code?` `.data?`

why use labels?

# Pseudo Instructions

- *la \$rd, offset (label)*  
loads the address of (label + offset) into \$rd.  
 $\$rd = \text{label} + \text{offset};$
- *lw \$rd, offset (label)*  
loads the data located at (label + offset) into \$rd.  
 $\$rd = *(\text{label} + \text{offset});$
- *sw \$rs, offset (label)*  
stores the value of \$rs to the address (label + offset).  
 $*(\text{label} + \text{offset}) = \$rs$
- *li \$rd, immediate*  
loads an immediate value into \$rd.

# Example



# bench\_i.coe

```
MEMORY_INITIALIZATION_RADIX=16;  
MEMORY_INITIALIZATION_VECTOR=  
00000, ← addr 0x0  
00000,  
00000,  
....  
....  
....  
10011 ← addr 0x100  
110DE  
....  
11110 ← end of the text  
<EOF>
```

- Start at 0x0
- Not necessarily specify data for every memory location
- Unspecified locations will have arbitrary values

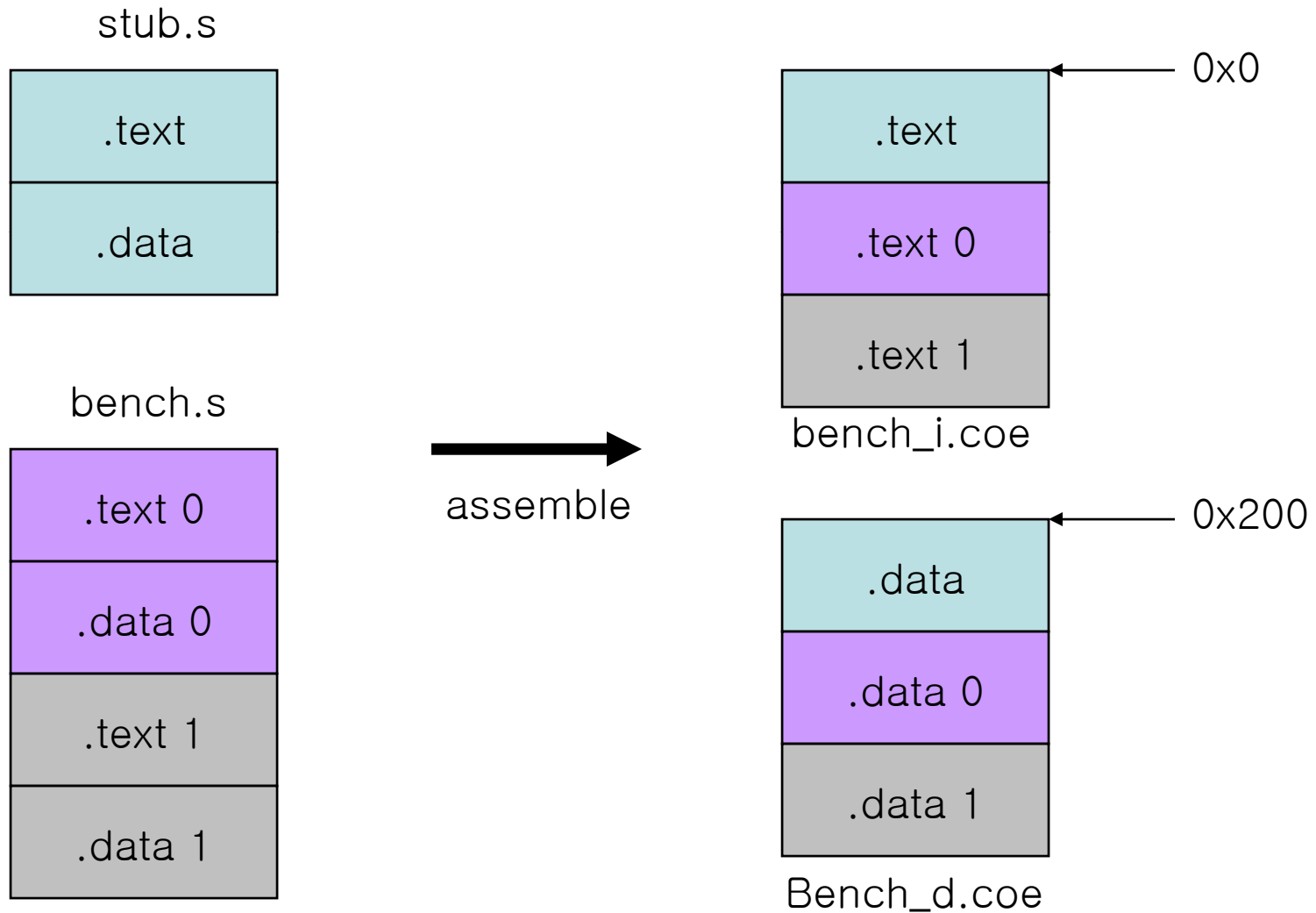
# Code Stub

- What are required to properly run a function?
  - Stack pointer
  - Global pointer
  - Function arguments
- Code stub
  - Setup required environments for a function call
    - stack pointer
    - global pointer
    - function arguments
  - Should be located at 0x0 of the i-memory

# Assembler with a Code Stub

- Inputs
  - src \*.s (a benchmark file)
  - stub \*.s (a stub file)
  - data\_addr <addr>
- Outputs
  - \$name\_i.coe
  - \$name\_d.coe
- Example
  - *prompt> asm -stub stub.s -src bench.s -data\_addr 0x200*
  - Outputs: bench\_i.coe, bench\_d.coe

# Assemble with a Code Stub



Any Questions?