

# SPIM16 EECS Project 1 Specification

## ASSEMBLY APPLICATION FORMAT

### Assembly File

- asm File Extension
- Program Lines
  - Zero or One Label, plus
  - Zero or One Instruction, plus
  - Zero or One Comment
- Labels – Symbol Definitions
  - Starts with a letter, plus
  - Zero or more symbol characters, plus
  - A single colon ':'
- Symbol Characters
  - Letter
  - Digit
  - Underscore
- Instructions
  - Starts with a name, plus
  - One or more operands separated by whitespace
- Operands (depends on instruction format)
  - A Register, OR
  - An Immediate Value, OR
  - A Label
- Comments
  - Begin with pound sign '#'
  - Continues to end of line
- Load/Store
  - Source/destination register first operand
  - Immediate value second operand
  - Base address register third operand surrounded by parenthesis

All assembler input is to be considered valid and well formed.

## SAMPLE APPLICATION

```
# Sample Application
Main:
    and $s1, $s1, $zero # clear contents
    addi $s1, $zero, Count # handle appropriately
    lw $s0, 0($s1) # data at Counter address
Loop:
    bz $s0, Done # test
    or $a0, $s1, $zero # move
    addi $a0, $a0, 2 # next instruction
    addi $ra, $zero, Loop
    addi $ra, $ra, 12 # return address
    jr $a0 # jump to Dummy
    addi $s0, $s0, -1 # decrement count
    sw $s0, 0($s1) # store value
    j Loop
Done:
    j Done
    addi $s0, $zero, 43
    addi $s1, $zero, 34
Count:
    .data 17 # data value
Dummy:
    jr $ra # return
    j Foobar
    .data -7
```

## BINARY INSTRUCTION FORMATS

### Register Format

```
# Assembly Code
operation rd, rs, rt
```

```
# Machine Code
oooo ssss tttt dddd
```

### Register-Immediate Format

```
# Assembly Code
operation rt, rs, immediate
operation rt, immediate(rs)
```

```
# Machine Code
oooo ssss tttt iiii
```

### Immediate Format

```
# Assembly Code
operation rs, immediate
operation rs, label
```

```
# Machine Code
oooo ssss iiii iiii
```

### Jump Format

```
# Assembly Code
operation immediate
operation label
```

```
# Machine Code
oooo iiii iiii iiii
```

## LABEL HANDLING

### Label Addressing

- Label Address is Address of Next Instruction
- Only Instruction Lines Increment Addresses
- Addresses Increment by Instruction Width in Bytes (2)
- Store All Label Definitions in a Symbol Table

### Symbol Table

- Table Entry
  - Symbol Label
  - Defined?
  - Address Value
  - Usage List
- Usage Entry
  - Instruction
  - Usage Address
- Branch instructions are ignored

## FILE OUTPUT

- Symbols File (.syms)
- Object File (.o)
- All addresses should be hexadecimal and 4 characters wide
- Object file output should be printed as hexadecimal text