

10520 CS410001 - Computer Architecture 2017

Appendix B - Input Samples

An Example C program:

```
sum = 0;
for( i = 0; i < 3; i++ ) {
    sum += i;
}
```

Suppose that

1. The sizes of *sum* & *i* are words.
2. The address of *sum* is located at 0x00000000 in D memory, while the address of *i* is at 0x00000008 in D memory.
3. PC is initially 0, and \$sp is initially 0x400.

Translate into assembly:

```
andi $t0, $0, 0    # sum = $t0 = 0
andi $t1, $0, 0    # i = $t1 = 0
loop: slti $t2, $t1, 3    # $t2 = ( i < 3 )
      beq $t2, $0, end    # if (i >= 3), go to end
      add $t0, $t0, $t1   # sum = sum + i
      addi $t1, $t1, 1    # i++
      j loop              # jump to loop
end: sw $t0, 0($0)      # store sum
      halt
      halt
      halt
      halt
      halt
```

Then, **this program is compiled into binary assembly code, which is listed below in hexadecimal format for convenience of reading.** Note that the comments are here to help you understand the meaning of each line and your submitted input files shall allow no comments.

● **iimage.bin:**

- i. This file specifies the instruction image (in **big-endian** format, encoded in binary).
- ii. The first four bytes indicate the initial value of PC, i.e. the starting address to load the instruction image.
- iii. The next four bytes specify the number of words to be loaded into instruction memory.
- iv. The remaining ones are the program instructions to be loaded into I-memory.

10520 CS410001 - Computer Architecture 2017

iimage.bin:

```
0x00000000 # initial value of PC
0x0000000D # number of words to be loaded into I memory
0x30080000 # instructions in I memory begin from here
0x30090000
0x292A0003
0x11400003
0x01094020
0x21290001
0x08000002
0xAC080000
0xFFFFFFFF
0xFFFFFFFF
0xFFFFFFFF
0xFFFFFFFF
0xFFFFFFFF
```

● **dimage.bin:**

- i. This file specifies the data image (in **big-endian** format, encoded in binary).
- ii. The first four bytes indicate the initial value of \$sp.
- iii. **The next four bytes specify the number of words to be loaded into data memory, starting from address 0.**

dimage.bin:

```
0x00000400 # initial value of $sp
0x00000003 #number of words to be loaded into D memory
0x12345678 # content of D memory begins
0x9ABCDEF0
0x13572468
```

<NOTE> Here are a few friendly reminders for creating valid test cases.

- i. When initializing I memory or D memory, the address should be at most 1023 (1K size).
- ii. Address overflow or misaligned access in I or D memory is **not allowed**
- iii. The simulation should execute no more than 500,000 cycles.
- iv. In project 2, at the end of the simulation all pipeline stages should be filled with halt instructions, except when address overflow or misaligned access occurs in D memory.