

COAL Lab-06

Load & Store Instruction Assembly Code

Name: Saad Nisar Butt

Reg. no: cs211246

Class: BSCS-3C-1

Literature Review:

Load Instructions:

Load instructions are used to move data from memory to registers (before operation). Loads are encoded in the I-type format. The effective byte address is obtained by adding register rs1 to the sign-extended 12-bit offset. Loads copy a value from memory to register rd. The assembly representation for load instructions are:

lw (destination_register), (offset)(source_register)

or

lw (rd), offset(rs1)

The LW instruction loads a 32-bit value from memory into rd. LH loads a 16-bit value from memory, then sign-extends to 32-bits before storing in rd. LHU loads a 16-bit value from memory but then zero extends to 32-bits before storing in rd. LB and LBU are defined analogously for 8-bit values.

Store Instructions:

Store instructions are used to move data from registers to memory (after operation). Stores are encoded in the S-type format. The effective byte address is obtained by adding register rs1 to the sign-extended 12-bit offset. Stores copy the value in register rs2 to memory.. The assembly representation for store instructions are:

sw (source_register_2), (offset)(source_register_1)

or

sw (rs2), offset(rs1)

The SW instruction stores a 32-bit value from the low bits of register rs2 to memory. SH stores a 16-bit value from the low bits of register rs2 to memory. SB stores a 8-bit value from the low bits of register rs2 to memory.

Lab Exercise 1:

Task:

Run the below assembly code on Venus Simulator

li s0, 0x12345678 # Data to be store

li s1, 0x00000020 # memory address

sb s0, 0x0(s1)

sh s0, 0x4(s1)

sw s0, 0x8(s1)

sb s0, 0x0(s1)

| | | | | | |
|---------|-------|-------|-----|-------|---------|
| 0000000 | 01000 | 01001 | 000 | 00000 | 0100011 |
|---------|-------|-------|-----|-------|---------|

Machine Code: 0000 0000 1000 0100 1000 0000 0010 0011

Hexadecimal Code: 00848023

sh s0, 0x4(s1)

| | | | | | |
|---------|-------|-------|-----|-------|---------|
| 0000000 | 01000 | 01001 | 001 | 00100 | 0100011 |
|---------|-------|-------|-----|-------|---------|

Machine Code: 0000 0000 1000 0100 1001 0010 0010 0011

Hexadecimal Code: 00849223

sw s0, 0x8(s1)

| | | | | | |
|---------|-------|-------|-----|-------|---------|
| 0000000 | 01000 | 01001 | 010 | 01000 | 0100011 |
|---------|-------|-------|-----|-------|---------|

Machine Code: 0000 0000 1000 0100 1010 0100 0010 0011

Hexadecimal Code: 0084A423

Venus Simulation:

The screenshot shows the Venus simulator interface. The main window displays a table of assembly code with columns for PC, Machine Code, Basic Code, and Original Code. The code is as follows:

| PC | Machine Code | Basic Code | Original Code |
|------|--------------|-----------------|-------------------|
| 0x0 | 0x12345437 | lui x8 74565 | li s0, 0x12345678 |
| 0x4 | 0x07890413 | addi x8 x8 1656 | li s0, 0x12345678 |
| 0x8 | 0x82000493 | addi x9 x0 32 | li s1, 0x00000020 |
| 0xc | 0x00848023 | sb x8 0(x9) | sb s0, 0x0(s1) |
| 0x10 | 0x00049223 | sh x8 4(x9) | sh s0, 0x4(s1) |
| 0x14 | 0x0084A423 | sw x8 8(x9) | sw s0, 0x8(s1) |

Below the table is a console output area. On the right side, there is a panel for registers, memory, cache, and VCR. The registers panel shows the following values:

| Register | Value |
|----------|------------|
| zfp0 | 0x00000000 |
| ra (x1) | 0x00000000 |
| sp (x2) | 0x00000000 |
| gp (x3) | 0x00000000 |
| tp (x4) | 0x00000000 |
| t0 (x5) | 0x00000000 |
| t1 (x6) | 0x00000000 |
| t2 (x7) | 0x00000000 |
| s0 (x8) | 0x12345678 |
| s1 (x9) | 0x00000020 |
| a0 (x10) | 0x00000000 |

The screenshot shows the Venus simulator interface. The main window displays a table of assembly code with columns for PC, Machine Code, Basic Code, and Original Code. The code is as follows:

| PC | Machine Code | Basic Code | Original Code |
|------|--------------|-----------------|-------------------|
| 0x0 | 0x12345437 | lui x8 74565 | li s0, 0x12345678 |
| 0x4 | 0x07890413 | addi x8 x8 1656 | li s0, 0x12345678 |
| 0x8 | 0x82000493 | addi x9 x0 32 | li s1, 0x00000020 |
| 0xc | 0x00848023 | sb x8 0(x9) | sb s0, 0x0(s1) |
| 0x10 | 0x00049223 | sh x8 4(x9) | sh s0, 0x4(s1) |
| 0x14 | 0x0084A423 | sw x8 8(x9) | sw s0, 0x8(s1) |

Below the table is a console output area. On the right side, there is a panel for registers, memory, cache, and VCR. The memory panel shows a dump of memory addresses and their corresponding values:

| Address | x3 | x2 | x1 | x0 |
|------------|----|----|----|----|
| 0x00000020 | 00 | 00 | 00 | 78 |
| 0x0000002C | 30 | 00 | 00 | 00 |
| 0x00000018 | 30 | 00 | 00 | 00 |
| 0x00000014 | 00 | 04 | A4 | 23 |
| 0x00000010 | 00 | 04 | 00 | 23 |
| 0x0000000C | 00 | 04 | 00 | 23 |
| 0x00000008 | 00 | 00 | 04 | 43 |
| 0x00000004 | 07 | 04 | 04 | 13 |
| 0x00000000 | 12 | 34 | 54 | 27 |
| ----- | 00 | 00 | 00 | 00 |
| ----- | 00 | 00 | 00 | 00 |

Lab Exercise 2:

Task:

Run the below assembly code on Venus Simulator

lb t0, 0x0(x0)

lbu t1, 0x4(x0)

lh t2, 0x8(x0)

lhu s0, 0xC(x0)

lw s1, 0x10(x0)

lb t0, 0x0(x0)

| | | | | |
|--------------|-------|-----|-------|---------|
| 000000000000 | 00000 | 000 | 00101 | 0000011 |
|--------------|-------|-----|-------|---------|

Machine Code: 0000 0000 0000 0000 0000 0010 1000 0011

Hexadecimal Code: 00000283

lbu t1, 0x4(x0)

| | | | | |
|--------------|-------|-----|-------|---------|
| 000000000100 | 00000 | 100 | 00110 | 0000011 |
|--------------|-------|-----|-------|---------|

Machine Code: 0000 0000 0100 0000 0100 0011 0000 0011

Hexadecimal Code: 00404303

lh t2, 0x8(x0)

| | | | | |
|--------------|-------|-----|-------|---------|
| 000000001000 | 00000 | 001 | 00111 | 0000011 |
|--------------|-------|-----|-------|---------|

Machine Code: 0000 0000 1000 0000 0001 0011 1000 0011

Hexadecimal Code: 00801383

lhu s0, 0xC(x0)

| | | | | |
|--------------|-------|-----|-------|---------|
| 000000001100 | 00000 | 101 | 01000 | 0000011 |
|--------------|-------|-----|-------|---------|

Machine Code: 0000 0000 1100 0000 0101 0100 0000 0011

Hexadecimal Code: 00C05403

lw s1, 0x10(x0)

| | | | | |
|--------------|-------|-----|-------|---------|
| 000000010000 | 00000 | 010 | 01001 | 0000011 |
|--------------|-------|-----|-------|---------|

Machine Code: 0000 0001 0000 0000 0010 0100 1000 0011

Hexadecimal Code: 01002483

Venus Simulation

The screenshot displays the Venus CPU simulator interface. The main window is divided into several sections:

- Assembly Code Table:** A table with four columns: PC, Machine Code, Basic Code, and Original Code. It lists several instructions, including `lb x5, 0(x0)`, `lbu x6, 4(x0)`, `lh x7, 8(x0)`, `lhu x8, 12(x0)`, and `lw x9, 16(x0)`.
- Registers, Memory, Cache, VCB:** A section on the right showing the state of registers and memory. It includes a table with columns for Address, +3, +2, +1, and +0, and rows for various memory locations.
- Console Output:** A text area at the bottom left labeled "console output".
- Buttons:** A row of buttons at the top includes Run, Stop, Free, Reset, Dump, Trace, and An assembly from Editor.

venustools.org

View Editor Simulator Chocopy

Run Stop Prev Next Dump Load Save assembly from Editor

| PC | Machine Code | Basic Code | Original Code |
|------|--------------|---------------|-----------------|
| 0x0 | 0x0000283 | lb x5 0(x0) | lb t0, 0x0(x0) |
| 0x4 | 0x0040400 | lbu x6 4(x0) | lbu t1, 0x4(x0) |
| 0x8 | 0x008013E3 | lh x7 8(x0) | lh t2, 0x8(x0) |
| 0xC | 0x00C05403 | lhw x8 12(x0) | lhw s0, 0xC(x0) |
| 0x10 | 0x010024E3 | lw x9 16(x0) | lw s1, 0x10(x0) |

Registers Memory Cache VCB

Integer (R) Floating (F)

zero 0x00000000

ra (x1) 0x00000000

sp (x2) 0x00000000

gp (x3) 0x00000000

tp (x4) 0x00000000

t0 (x5) 0x00000000

t1 (x6) 0x00000000

t2 (x7) 0x00000000

s0 (x8) 0x00000000

s1 (x9) 0x00000000

a0 (x10) 0x00000000

Display Settings

console output

Copy Download Clear

Activate Windows
Go to Settings to activate Windows.

InLab Tasks:

Task 1:

Write down a simple assembly program to add, and subtract two integer numbers and store their result into different memory locations. Stimulate the code on Venus.

RISC-V Assembly Code:

```
addi x5, x0, 4 # storing 4 in register x5
```

```
addi x6, x0, 6 # storing 6 in register x6
```

```
add s0, x5, x6 # addition
```

```
sub s1, x6, x5 # subtraction
```

```
addi x20, x20, 0x20 # setting memory's base address
```

```
sw s0, 0x0(x20) # storing s0's added value in memory
```

```
sw s1, 0x4(x20) # storing s1's subtracted value in memory
```

Venus Simulation:

The screenshot shows the Venus simulator interface. At the top, there are tabs for 'Venus', 'Editor', 'Simulator', and 'Chocopy'. Below the tabs, there are buttons for 'Run', 'Step', 'Prev', 'Reset', 'Dump', 'Trace', and 'Load assembly from Editor'. The main area displays a table of assembly code with columns for PC, Machine Code, Basic Code, and Original Code. The table contains 7 rows of code. To the right of the table, there is a 'Registers' window showing the values of various registers (x0 to x10) and a 'Memory' window showing the values of memory locations (0x00000000 to 0x00000010). The 'console output' area is empty.

| PC | Machine Code | Basic Code | Original Code |
|------|--------------|-------------------|---------------------|
| 0x0 | 0x00400293 | addi x5, x0, 4 | addi x5, x0, 4 |
| 0x4 | 0x00600513 | addi x5, x0, 6 | addi x5, x0, 6 |
| 0x8 | 0x00628433 | add x8, x5, x6 | add x8, x5, x6 |
| 0xc | 0x00300483 | sub x9, x6, x5 | sub x2, x6, x5 |
| 0x10 | 0x02800413 | addi x20, x20, 32 | addi x20, x20, 0x20 |
| 0x14 | 0x008A2023 | sw x8, 0(x20) | sw x8, 0x0(x20) |
| 0x18 | 0x009A2223 | sw x9, 4(x20) | sw x1, 0x4(x20) |

The screenshot shows the Venus simulator interface. At the top, there are tabs for 'Venus', 'Editor', 'Simulator', and 'Chocopy'. Below the tabs, there are buttons for 'Run', 'Step', 'Prev', 'Reset', 'Dump', 'Trace', and 'Load assembly from Editor'. The main area displays a table of assembly code with columns for PC, Machine Code, Basic Code, and Original Code. The table contains 7 rows of code. To the right of the table, there is a 'Registers' window showing the values of various registers (x0 to x10) and a 'Memory' window showing the values of memory locations (0x00000000 to 0x00000010). The 'console output' area is empty.

| PC | Machine Code | Basic Code | Original Code |
|------|--------------|-------------------|---------------------|
| 0x0 | 0x00400293 | addi x5, x0, 4 | addi x5, x0, 4 |
| 0x4 | 0x00600513 | addi x5, x0, 6 | addi x5, x0, 6 |
| 0x8 | 0x00628433 | add x8, x5, x6 | add x8, x5, x6 |
| 0xc | 0x00300483 | sub x9, x6, x5 | sub x2, x6, x5 |
| 0x10 | 0x02800413 | addi x20, x20, 32 | addi x20, x20, 0x20 |
| 0x14 | 0x008A2023 | sw x8, 0(x20) | sw x8, 0x0(x20) |
| 0x18 | 0x009A2223 | sw x9, 4(x20) | sw x1, 0x4(x20) |

Task 2:

Write down a simple assembly program to load the contents from memory into registers and perform the logical operations on them. Stimulate the code on Venus.

RISC-V Assembly Code:

```
addi x20, x20, 0x20 # setting memory's base address
```

```
lw s2, 0x0(x20) # loading data from memory into register
```

```
lw s3, 0x4(x20) # loading data from memory into register
```

```
and s5, s2, s3 # ANDing data loaded from memory
```

```
or s6, s3, s2 # Oring data loaded from memory
```

Venus Simulation:

The top screenshot shows the Venus simulator interface with the 'Registers' panel on the right. The panel displays memory addresses and values for registers x0 through x24. The bottom screenshot shows the same interface but with the 'Registers' panel displaying register names and values.

Registers Panel (Top Screenshot):

| Address | x3 | x2 | x1 | x0 |
|------------|----|----|----|----|
| 0x00000038 | 00 | 00 | 00 | 00 |
| 0x00000034 | 00 | 00 | 00 | 00 |
| 0x00000030 | 00 | 00 | 00 | 00 |
| 0x0000002C | 00 | 00 | 00 | 00 |
| 0x00000028 | 00 | 00 | 00 | 00 |
| 0x00000024 | 00 | 00 | 00 | 00 |
| 0x00000019 | 00 | 00 | 00 | 00 |
| 0x0000001C | 00 | 00 | 00 | 00 |
| 0x00000018 | 00 | 00 | 00 | 00 |
| 0x00000014 | 00 | 00 | 00 | 00 |
| 0x00000010 | 00 | 00 | 00 | 00 |

Registers Panel (Bottom Screenshot):

| Register | Value |
|----------|------------|
| x0 (x16) | 0x00000000 |
| x1 (x17) | 0x00000000 |
| x2 (x18) | 0x00000001 |
| x3 (x19) | 0x00000000 |
| x4 (x20) | 0x00000000 |
| x5 (x21) | 0x00000000 |
| x6 (x22) | 0x00000001 |
| x7 (x23) | 0x00000000 |
| x8 (x24) | 0x00000000 |

* ----- *