COAL Lab # 07

## Designing Microarchitecture-II

## Name: Muhammad Usman

**Class:** BSCS 3C1

**Reg No:** cs211208

## 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 representations 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 Tasks

# **Task 1**

# Write Verilog code for the sign extend and adder block. Make sure to name the input and output ports correctly with the correct number of bits. Attach the code.

# **Verilog Code for Sign Extended Block**

module sign\_extension(Imm,ImmExt);

    input [11:0] Imm;

    output [31:0] ImmExt;

    assign ImmExt = {{20{Imm[11]}},Imm};

endmodule

# **Verilog Code for Adder Block**

module Adder(Inp1,Inp2,Sum);

    input [31:0] Inp1,Inp2;

    output [31:0]Sum;

    assign Sum = Inp1+Inp2;

endmodule

# **Task 2**

# Write a complete data path for Load instruction in Verilog by instantiating all the module blocks. Attach the code.