Experiment4: Design and synthesize a 32-bit processor using Verilog

Design a 32-bit RISC-like processor in Verilog with the following specifications:

- Sixteen 32-bit general-purpose registers R0..R15, organized as a register bank with two read ports and one write port.
    - R0 is a special read-only register that is assumed to contain the fized value of 0.
- Memory is byte addressable with a 32-bit memory address. For simplicity, in this design we shall assume that all operations are on 32-bits data.
- A 32-bit program counter PC
- A 32-bit stack pointer SP
- Addressing modes to be supported:

    a) Register addressing

    b) Immediate addressing

    c) Register indexed addressing for accessing memory
- The following instruction set has to be implemented:

    a) Arithmetic and logic instructions: ADD, SUB, AND, OR, XOR, NOT, SLA, SRA, SRL. Some example uses are as follows:

```
ADDI    R3,#25      // R3 = R3 + 25
ADDI    R5,#-1          // R5 = R5 - 1
ADD     R1,R2,R3    // R1 = R2 + R3
SLA     R5,#2       // R5 =  R5 << 2
```

b) Load and store instructions: LD, ST (all load and stores are 32-bits) and use register indexed addressing. Any of the registers R1..R15 or SP can be used. Some example uses are as follows:

```
LD      R2,10(R6)    // R2 = Mem[R6+10]

ST      R2,-2(R11)   // Mem[R11-2] = R2

LDSP    SP,0(R2)     // SP = Mem[R2+0]
```

c) Branch instructions: BR, BMI, BPL, BZ. Some example uses are as follows:

```
BR      10(R2)              // PC = Mem[R2+10]

BMI     R5,10(R2)    // PC = Mem[R2+10] if R5<0

BPL     R5,10(R2)    // PC = Mem[R2+10] if R5>0

BZ      R5,10(R2)    // PC = Mem[R2+10] if R5=0
```

d) Stack instructions: PUSH, POP. Any registers R1..R15 can be used. Some example uses are as follows:

```
PUSH    R6          // Push R6 in the stack

POP     R10         // Pop from stack and store in R10
```

e) Register to register transfer: MOVE. Some example uses are as follows:

```
MOVE    R10,R5      // R10 = R5

MOVE    R2,R0       // R2 = R0
```

f) Program control: HALT.

Steps of implementation:

1. Design a 32-bit ALU corresponding to the 32-bit Processor specification as given above. (All shift operations are restricted to 1-bit shift only.)