Develop a single-cycle CPU that can do B and CBZ (only). Make it as simple as possible.

(OR)

Write the Truth Table for SUB Function with inputs: A, B & Cin for a full adder.
Given the above datapath and modified control signals, which of the following instructions are no longer supported?

1. ADD X2, X31, X4
2. SUB X1, X5, X2
3. LDUR X3, [X31, #9]
4. STUR X4, [X4, #5]
5. CBZ X7, LOOP
6. B LOOP
Use Assembly language to write a program that could recognize if the value of X1 is the multiple of 3; if so, store it in certain register. You can use other registers as temporary storage.
A, B, and C are arrays of length N containing integers. Implement the following (in assembly)

\[
\text{for } (i = 0; i < N; i++) \\
\quad C[i] = A[i] + B[i]
\]

// x0 = N, x1 = i, x2 = A, x3 = B, x4 = C
LOAD THIS VALUE INTO X14:
64h 00000000000AF2010
Draw the memory locations of the following code, Assume start at 1000:

```
Struct {int *a, *b, *c; char *d;} uw;
uw *obj;
obj = new UW;
obj -> a = new int[1];
obj -> b = new int[2];
obj -> c = new int[1];
obj -> d = new char[8];
obj -> a[0] = #10;
obj -> b[0] = #11;
obj -> b[1] = #12;
obj -> c[0] = #13;
obj -> d[7] = ‘j’;
```
Encode the following instructions:

1. B 12

2. STUR X8,[X11,#7]

3. SUBI X7,X1,#4
Turn the following machine code to assembly language and fill in the control setting for the instruction.

Machine code:
```
1111000010000101110010101100111
```

Assembly language:
```

```

Control setting:

<table>
<thead>
<tr>
<th>Reg2Loc</th>
<th>ALUSrc</th>
<th>MemToReg</th>
<th>RegWrite</th>
<th>MemWrite</th>
<th>BrTaken</th>
<th>UncondBr</th>
<th>ALUOp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. a) Design a 3-bits shifter.

b) Explain what a parallel Multiplier is?
USING A SIMILAR LOGIC FOR BUILDING A LEFT SHIFTER, CREATE A ROTATING LEFT SHIFTER FOR 8-BIT VALUES. IT TAKES IN AN 8-BIT VALUE AND SHIFTS OUT THE MOST SIGNIFICANT BIT (MSB) WHICH THEN BECOMES THE LEAST SIGNIFICANT BIT (LSB) OF THE 8-BIT VALUE. YOUR DESIGN SHOULD REQUIRE ONLY 2:1 MUXES.
Given the following signed 5-bit values (2's comp), compute the result.
Your answer should be a 9-bit signed value.
A: 01011
B: 11100
A*B
Handsome Eric got his new processor which runs at 180MHZ. For testing it, Eric used his own benchmark as shown in the table below with 100 instructions in it. (Assume this processor is so stupid that it executes one instruction at a time)

<table>
<thead>
<tr>
<th>Type</th>
<th>Type Cycle</th>
<th>Type Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load &amp; Store</td>
<td>6</td>
<td>30%</td>
</tr>
<tr>
<td>ALU</td>
<td>4</td>
<td>50%</td>
</tr>
<tr>
<td>All others</td>
<td>3</td>
<td>20%</td>
</tr>
</tbody>
</table>

To enhance the stupid processor's performance, Eric doubled the number of registers so that the clock period is increased by 20% and the number of Load & Store instructions is halved. Did smart Eric really enhance the processor's performance? By what percent the performance increased or decreased?
A RISC machine has the following properties: 1 million instructions, a CPI of 1, and a clock rate of 1.02MHz. A CISC machine has the following properties: 900,000 instructions, a CPI of 1.1, and a clock rate of 1.0MHz. Is the RISC machine better or worse than the CISC machine and by how much?
Suppose you want to run a program $P$ with $9 \times 10^7$ instructions on a 4 GHz machine with a CPI of 1.3. What is the expected CPU time?
Modify the datapath & specify control signals for the following instruction:

**BMEM [Rn, DAddr9]**

**RTL:**
- Instr = Mem[PC]
- PC = Mem[Reg[Rn] + SE(DAddr9)]
Q1: Write the 32-bit instruction that corresponds to the traced datapath and information provided below.

Given:
Rd = X15
Rn = X19
Rm = 5'b11010
DAaddr9 = 9'd420
RegWrite = 1
MemWrite = 0

Instruction:

References

<table>
<thead>
<tr>
<th>Opcode[31:26]</th>
<th>100010</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>110010</td>
</tr>
<tr>
<td></td>
<td>111110</td>
</tr>
<tr>
<td></td>
<td>111110</td>
</tr>
<tr>
<td></td>
<td>000101</td>
</tr>
<tr>
<td></td>
<td>101101</td>
</tr>
<tr>
<td>Opcode[25:21]</td>
<td>00000</td>
</tr>
<tr>
<td></td>
<td>00xxxx</td>
</tr>
<tr>
<td></td>
<td>00xxxx</td>
</tr>
</tbody>
</table>

Memory: LDUR, STUR

Branch: B

Conditional Branch: CBZ

Arithmetic: ADD, SUB