Problem 1. Finite State Machines (6 points)

Consider the sequential logic circuit to the right, which implements an FSM with a single data input IN and single data output OUT. Assume that all signal transitions are timed so that the dynamic discipline is satisfied at each register.

Please describe the operation of the FSM by filling in both the state transition diagram and the truth table shown below. The two-digit state names in the state transition diagram are S0, S1, the logic values present at the outputs of REG0 and REG1 after the rising edge of the clock. In the truth table, S0’ and S1’ are the values that will loaded into REG0 and REG1 at the next rising clock edge.

Fill in state transition diagram and truth table

<table>
<thead>
<tr>
<th>S0</th>
<th>S1</th>
<th>IN</th>
<th>S0’</th>
<th>S1’</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Problem 2. Pipelining (6 points)

The following 1-stage pipelined circuit computes Z from the four inputs A, B, C, and D. Each component is annotated with its propagation delay in ns.

Extra copies of the diagrams can be found on the back of the first page.

(A) (3 points) Please pipeline the circuit above for maximum throughput with the minimum possible latency using ideal pipeline registers ($t_{pd} = 0, t_{setup} = 0$). Show the location of pipeline registers in the diagram above using filled-in circles, like the one shown on the Z output. Please give the latency and throughput of the resulting pipelined circuit.

Latency (ns)? __________
Throughput (1/ns)? __________

(B) (3 points) Now suppose the “3” component is replaced by a two-way interleaved component with a minimum $t_{CLK}$ of 1.5ns. Recall that a two-way interleaved component behaves like a 2-stage pipelined component. Again please pipeline the circuit below for maximum throughput with the minimum possible latency using ideal pipeline registers. Show the location of pipeline registers in the diagram below using filled-in circles, like the one shown on the Z output. Please give the latency and throughput of the resulting pipelined circuit.

Latency (ns)? __________
Throughput (1/ns)? __________
Problem 3. Beta Assembly (6 Points)

Each of the following programs is loaded into a Beta’s main memory starting at location 0 and execution is started with the Beta’s PC set to 0. Assume that all registers have been initialized to 0 before execution begins. Please determine the specified values after execution reaches the HALT() instruction and the Beta stops. Write “CAN’T TELL” if the value cannot be determined. Please write all values in hex.

(A) (2 points)

```
. = 0
LD(R31,X+4,R1)
SHLC(R1,2,R1)
LD(R1,X,R2)
HALT()

X: LONG(4)
   LONG(3)
   LONG(2)
   LONG(1)
   LONG(0)
```

Value left in R1: 0x_______________
Value left in R2: 0x_______________

(B) (4 points)

```
. = 0
LD(R31,X,R0)
CMOVE(0,R1)

L:  CMPLTC(R0,0,R2)
   BNE(R2,DONE)
   ADDC(R1,1,R1)
   SHLC(R0,1,R0)
   BR(L)

DONE:  HALT()

X: LONG(0x08306352)
```

Value left in R0: 0x_______________
Value left in R1: 0x_______________
Value left in R2: 0x_______________
Value assembler assigns to symbol X: 0x_______________
Problem 4. Stack Detective (12 points)

The following C program implements a function \(H(x,y)\) of two arguments, which returns an integer result. The assembly code for the procedure is shown on the right.

\[
\text{int } H(\text{int } x, \text{int } y) \{ \\
\text{int } a = x - y; \\
\text{if } (a < 0) \text{ return } x; \\
\text{else return } ???; \\
\}
\]

The execution of the procedure call \(H(0x68, 0x20)\) has been suspended just as the Beta is about to execute the instruction labeled “rtn:” during one of the recursive calls to \(H\). A partial trace of the stack at the time execution was suspended is shown to the right below.

(A) (2 Points) Examining the assembly language for \(H\), what is the appropriate C code for ??? in the C representation for \(H\)?

**C code for ???:** _____________________________________

(B) (5 Points) Please fill in the values for the blank locations in the stack dump shown on the right. Express the values in hex or write “---” if value can’t be determined. Hint: Figure out the layout of \(H\)’s activation record and use it to identify and label the stack frames in the stack dump.

**Fill in the blank locations with values (in hex!) or “---“**

(C) (5 Points) Determine the specified values at the time execution was suspended. Please express each value in hex or write “CAN’T TELL” if the value cannot be determined.

Value in R0 or “CANT TELL”: 0x___________

Value in R1 or “CANT TELL”: 0x___________

Value in BP or “CANT TELL”: 0x___________

Value in LP or “CANT TELL”: 0x___________

Value in SP or “CANT TELL”: 0x___________

END OF QUIZ 2!

Extra copies of the code and stack frame can be found on the back of the third page.