Pre-assignment Task

To prepare for this assignment enter the following code and run it on you FPGA board. Instead of typing the code, you can use the zipped file for this processor that is included in Module 3(remember to unzip it first).

This is the code for *a* custom single-purpose processor that counts up every time input button B is pressed. It always outputs the count as an unsigned number on an 8-bit output C, which is initially 0. A press is detected as a change from 1 to 0 the duration of that 0 does not matter. Figure 1 shows the inputs and outputs of the processor.

1. Figure 2 shows the FSMD state diagram for this system.
2. Figure 3 shows the Separation the FSMD into an FSM+D, this means it shows the controller design and datapath connections.
3. Figure 4 shows the FSM
4. State table and K-maps are used to come up with Boolean equations for controller design.
5. Figure 5 shows the controller design. Steps 2 – 5 must be used if we want to design the processor at the gate level, otherwise we can directly write the VHDL model from FSMD as shown in step 6.
6. The VHDL code for this custom single-purpose processor is given. This VHDL is directly written using the FSMD.

Input: B
Output: Count

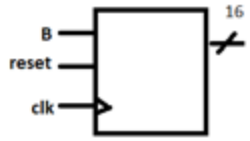

Figure 1

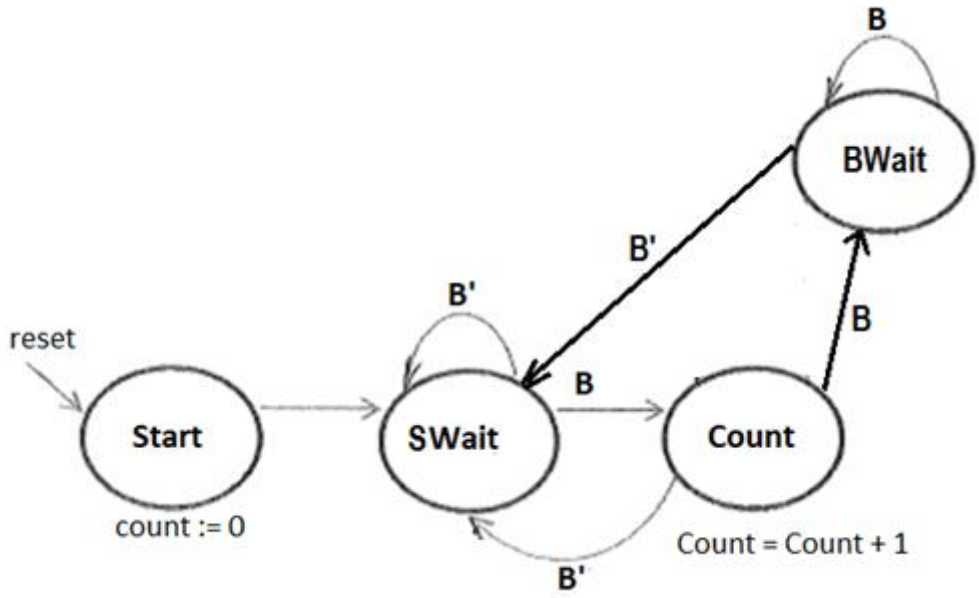

Figure 2 - FSMD

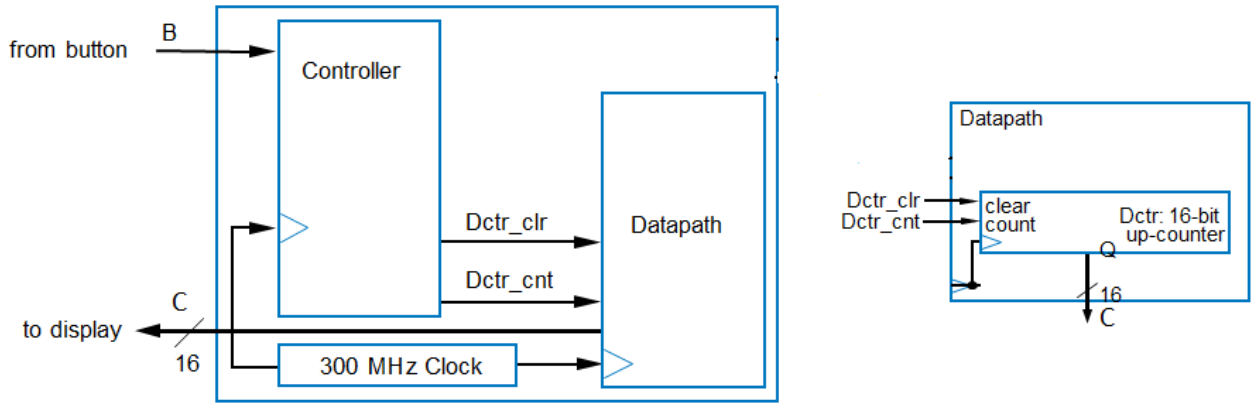

Figure 3 – Controller block diagram and datapath

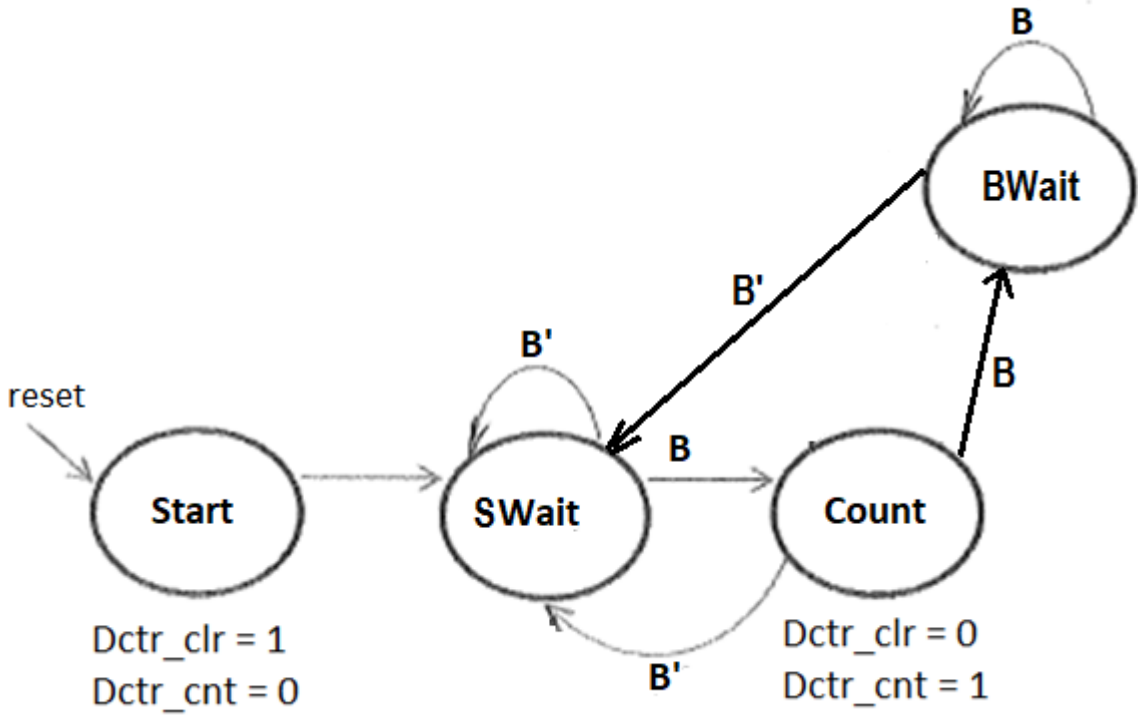

Figure 4 – FSM for controller

State	S1	S0	B	I1	I0	Dctr_clr	Dctr_cnt
Start	0	0	0	0	1	1	0
	0	0	1	0	1	1	0
SWait	0	1	0	0	1	0	0
	0	1	1	1	0	0	0
Count	1	0	0	0	1	0	1
	1	0	1	1	1	0	1
BWait	1	1	0	0	1	0	0
	1	1	1	1	1	0	0

State table

K- map used to minimize the logic fo r controller design.

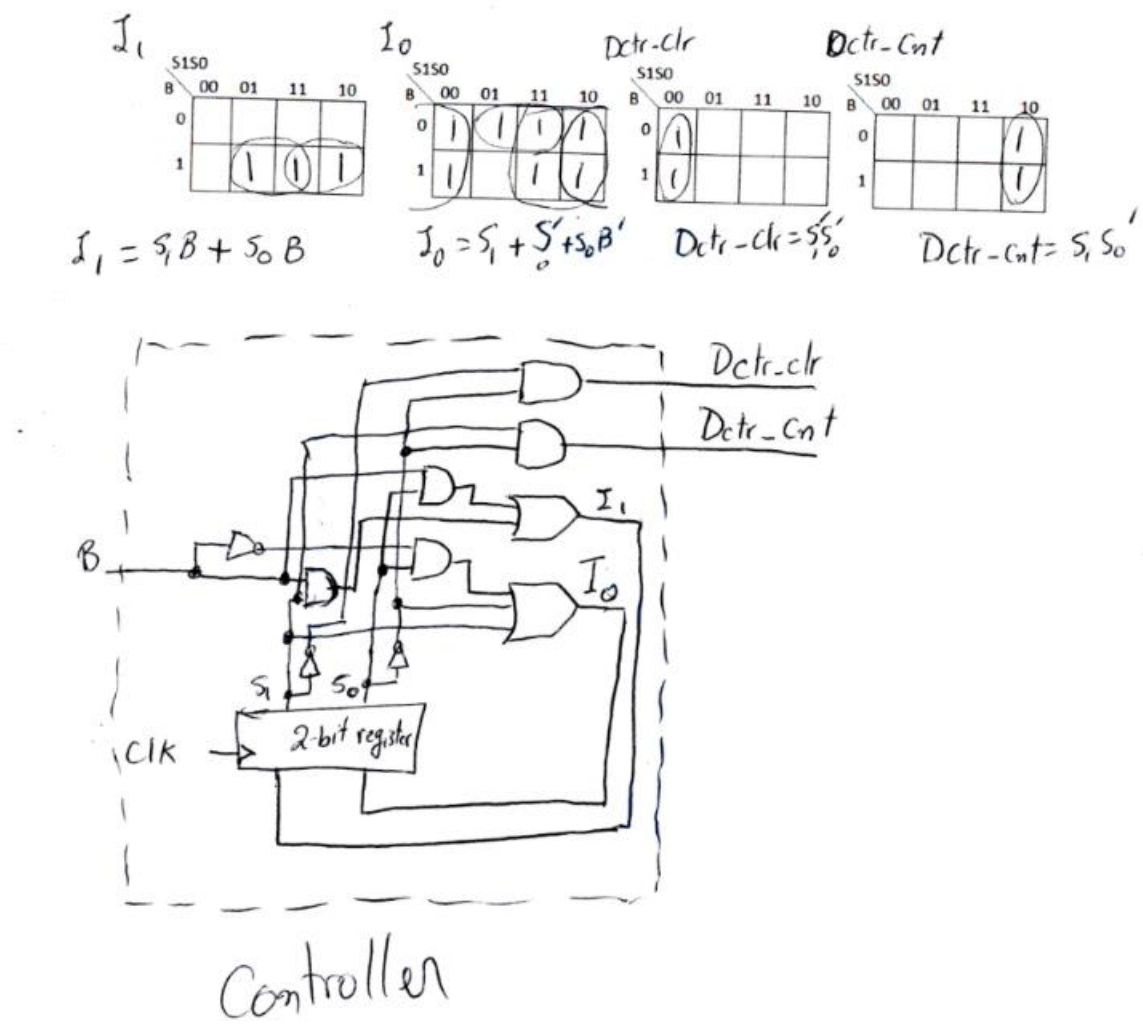

Figure 5 – Controller Design

6. VHDL Model for Counter processor

```
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22  --use IEEE.NUMERIC_STD.ALL;
23  use IEEE.STD_LOGIC_UNSIGNED.ALL;
24
25  entity Proc_Counter is
26      Port ( clk : in  STD_LOGIC;
27             reset : in  STD_LOGIC;
28             B : in  STD_LOGIC;
29             C : out  STD_LOGIC_VECTOR (7 downto 0));
30  end Proc_Counter;
31
32  architecture Behavioral of Proc_Counter is
33  type statetype is (start, Swait, count, Bwait);
34  signal state, nextstate: statetype;
35
36  signal counter, counternext: STD_LOGIC_VECTOR (7 downto 0);
37
38  begin
39  --state register
40  process(clk, reset)
41  begin
42  if(reset ='0') then
43  state <= start;
44  counter <= "00000000";
45  elsif (clk' event and clk='1') then
46  state <= nextstate;
47  counter <=counternext;
48  end if;
49  end process;
50
51  --combinational logic
52  process(state, B)
53  begin
54  case (state) is
55  when start =>
56  counternext <="00000000";
57  nextstate <= Swait;
58
59  when Swait =>
60  counternext <=counter;
61  if (B='0') then
62  nextstate <= count;
63  else
64  nextstate<= Swait;
65  end if;
66
67  when count =>
68  counternext <= counter + 1;
69  if (B='0') then
70  nextstate <= Bwait;
71  else
72  nextstate<= Swait;
73  end if;
74
75  when Bwait =>
```

```
counternext <=counter;
if (B='0') then
--c <= counter;
nextstate <= Bwait;
else
--c <= counter;
nextstate<= Swait;
end if;

end case;
end process;
--assign counter to c
c <= counter;

end Behavioral;
```

User constraint file

```
###Clock
   NET "clk"   LOC = P129 | IOSTANDARD = LVTTL | PERIOD = 12 MHz;
############Input push buttons
   NET "reset" LOC = P80 | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST | PULLUP;
   NET "B"  LOC = P79 | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST | PULLUP;

###########output LED
   NET "C[7]"  LOC = P46 | IOSTANDARD = LVTTL | DRIVE = 8;#| SLEW = FAST;
   NET "C[6]"  LOC = P47 | IOSTANDARD = LVTTL | DRIVE = 8;#| SLEW = FAST;
   NET "C[5]"  LOC = P48 | IOSTANDARD = LVTTL | DRIVE = 8;#| SLEW = FAST;
   NET "C[4]"  LOC = P49 | IOSTANDARD = LVTTL | DRIVE = 8;#| SLEW = FAST;
   NET "C[3]"  LOC = P50 | IOSTANDARD = LVTTL | DRIVE = 8;#| SLEW = FAST;
   NET "C[2]"  LOC = P51 | IOSTANDARD = LVTTL | DRIVE = 8;#| SLEW = FAST;
   NET "C[1]"  LOC = P54 | IOSTANDARD = LVTTL | DRIVE = 8;#| SLEW = FAST;
   NET "C[0]"  LOC = P55 | IOSTANDARD = LVTTL | DRIVE = 8;#| SLEW = FAST;

```

The zipped file for this code is provided to you in module 3.

EEL 4740 – Embedded Computing Systems Assignment 5

Points possible: 3 points

Submission Requirement: Submit your state diagram on paper and demonstrate the code running on your board on Thursday June 27th.

1. Design a custom single-purpose processor for a system that counts up every time input button B is pressed and counts down when button D is pressed. It always outputs the count as an unsigned number on an 8-bit output C, which is initially 0. A press is detected as a change from 1 to 0 the duration of that 0 does not matter. The system count must roll over when the maximum value of C is reached.
    7. Capture FSMD and draw the FSMD state diagram for this system.
    8. Write the VHDL code for this custom single-purpose processor.