IIT JODHPUR

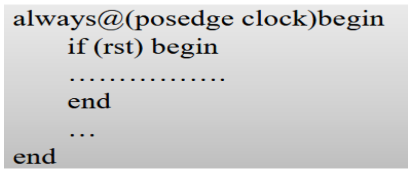
Minor Examination: EEL1570 Hardware Software Co-design (OPEN-BOOK)

Guidelines (Total time: 120 minutes, Maximum Marks: 25):

- Please read the question paper very carefully. **NO clarification is required in any question.** In case of any doubt, assume whatever you wish to and state that clearly in your answer.

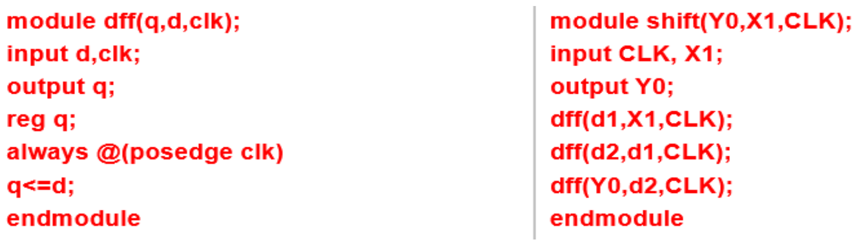
---

1. You are working as an intern in *BharatSemi* company. You encounter a bug where the reset to design is not working correctly. You suspect that even though the design is synchronous in nature, the reset needs to be asynchronous. How would you modify the below code segment so as to fix this design bug? [2]



```
always@(posedge clock)begin
      if (rst) begin
      ...............
      end
      ...
end
```

2. As an intern, your team leader gave the following code for understanding and analysis. If it is known that the clock to both modules is running at 60 MHz and the input to **shift** module is ready at 0 ns, find out at what time the output of **shift** module is available? Explain with a suitable diagram? [3]

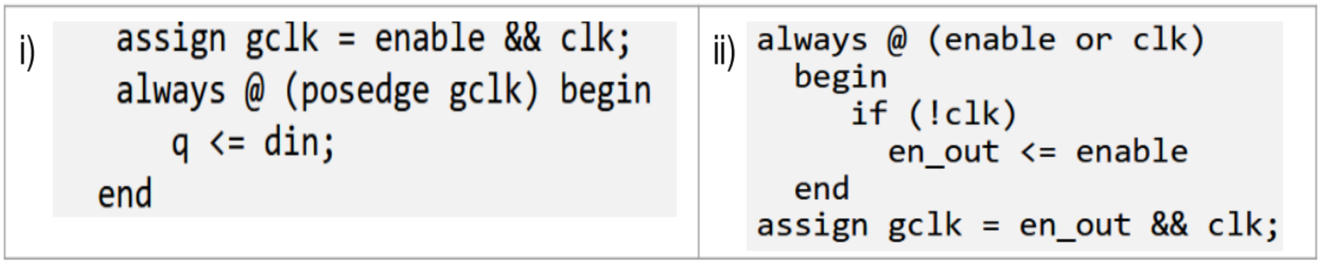


```
module dff(q,d,clk);
input d,clk;
output q;
reg q;
always @(posedge clk)
q<=d;
endmodule
```

```
module shift(Y0,X1,CLK);
input CLK, X1;
output Y0;
dff(d1,X1,CLK);
dff(d2,d1,CLK);
dff(Y0,d2,CLK);
endmodule
```

3. You are working at a deep-tech startup named as *MewarSemi Technologies* incubated at IIT Jodhpur. Suppose your team decides to design a system for which there are 3 types of CPUs available- CPU_1 has a cost of 5 units, CPU_2 has a cost of 6 units and CPU_3 has a cost of 4 units. Find out the number of CPU_1, CPU_2 and CPU_3 that we should choose if our goal is to minimize the cost of system given that a minimum of any two types of CPUs must be used while designing the above system? How? [3]

4. Clock gating (i.e., controlling system clock so that power dissipation of the design can be managed) is frequently utilized for the purpose of implementing low-power System-on-Chip (SoC) designs. Note that for the shown code, clk is the original clock signal, gclk is the gated clock signal and enable is the signal that turns ON the gating. Below are two methods for achieving clock gating. Which one out of (i) and (ii) you would utilize for an efficient implementation? Explain with a proper reason? [3]



i)
```
    assign gclk = enable && clk;
    always @ (posedge gclk) begin
       q <= din;
   end
```

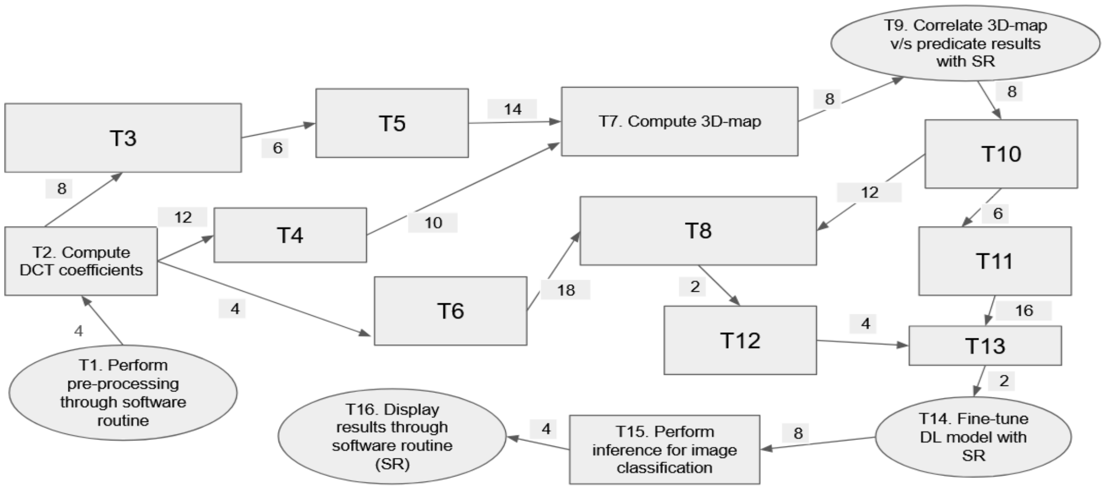
ii)
```
always @ (enable or clk)
  begin
    if (!clk)
       en_out <= enable
  end
assign gclk = en_out && clk;
```

5. Suppose your goal is to design a system that captures images with a camera and afterwards, the system displays the images on a separate screen. While displaying these images, the refresh is done at such a high rate that the captured images appear in the form of a video. The captured images are stored in the on-system memory in a compressed format due to limited size, however, for displaying these images on the screen, they must be decompressed. Furthermore, it is required to remove the background of night sky from the captured images and transform in daylight so that in the video, it appears that events are happening in day. Develop this system through either only-hardware/only-software/hardware software co-design approach? Out of these 3 approaches, which one is best? Why? [3.5 + 1.5]

6. Consider that you wish to design a system using a Moore-type FSM. A Moore machine is an FSM in which the outputs depend only on the current state, not on the inputs. There are two inputs (PB3, PB2) and two outputs (PB1, PB0). The FSM runs in the background with 1 ms clock period (that has duty cycle ~100%). Initially both outputs will be low, and you may also assume both inputs are initially low. If PB3 rises before PB2 rises, then set PB1 high. If PB2 rises before PB3 rises, then set PB0 high. If both rise during the same 1-ms window, set both PB1 and PB0 high. After either or both PB1 and/or PB0 become high, let the output remain fixed. Draw this FSM with all transitions and the states? [3]

7. Supported by your friends, you decide to incubate a startup at IIT Jodhpur instead of placements. The idea is to put a mini-camera in a smart-watch to enable image capture just by hand gestures and then images are to be analyzed so that Deep Learning algorithms can convert the image contents in text. For this purpose, you plan to design an intelligent image processing system (I2PS). Task graph of I2PS is shown as below where few task names are provided whereas other tasks are simply numbered. Computational resources available are: i) a dual-core processor (i.e., having 2 processor cores) and ii) two accelerators- one Deep Learning Unit (DLU) to execute DL algorithms such as DL-based 3D-map reconstruction etc., and one Signal Processing Unit (SPU) to execute tasks such as DCT coefficients finding etc.

   It is known that out of these **named tasks** (e.g., T1, T2, T7...), some must be completed on processor (generally shown by ellipse in below diagram) and some need to be executed on custom logic/accelerator (typically shown by rectangle in below diagram), whereas the **simply numbered tasks** (such as T3, T5, T4, T6 etc.) can either be performed on processor/accelerators (custom IPs). The edges in the below diagram denote the communication bandwidth (i.e., amount of data transferred) between the respective tasks (data transferred between tasks is equal to the number on the edge multiplied by 10 bytes). [**Hint:** Many tasks in the below task-graph/application-graph can be easily clustered/merged together.]



T9. Correlate 3D-map v/s predicate results with SR
T3
T5
T7. Compute 3D-map
T10
T4
T8
T11
T2. Compute DCT coefficients
T6
T12
T13
T1. Perform pre-processing through software routine
T16. Display results through software routine (SR)
T15. Perform inference for image classification
T14. Fine-tune DL model with SR
14
8
8
6
12
10
12
6
18
2
4
16
4
4
2
4
8
8

   Given the following parameters regarding the available computational and communication resources:

   - Each task when executed on the processor core has a latency between 10-90 cycles.
   - Each task executed on the hardware accelerators has a latency between 5-15 cycles.
   - Communication latency within the processor sub-system (i.e., between the two CPU cores) is zero, whereas the latency on the AHB (from standard AMBA architecture) shared bus (that connects processor IPs, on-chip memory and accelerators) has a fixed latency of 2 cycles.
   - Access of on-chip memory takes 8-12 cycles, off-chip memory access requires between 20-30 cycles.

   With the help of a proper modeling/estimation approach, develop the design as shown above through either only-hardware/only-software/hardware software co-design approach? Out of these 3 approaches, which one is best? Why? In case of co-design, kindly provide the details of system partitioning etc.? [6]