IIT JODHPUR

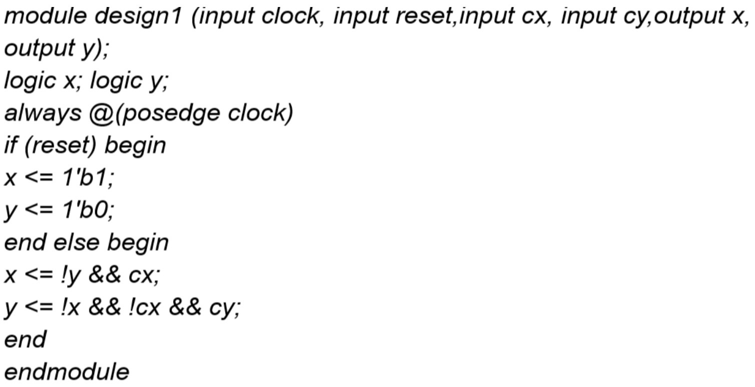
Major Examination: Formal Verification (Nov'24) [OPEN-BOOK]

Guidelines (Total time: 180 minutes, Maximum Marks: 50):

- Answer **to-the-point** ONLY. Writing unnecessary statements/lengthy answers may attract penalty.
- Use mathematical reasoning and symbols as far as possible in all your answers.
- Any answer remotely similar to AI/LLM tool-generated text would lead to **NEGATIVE** marks.

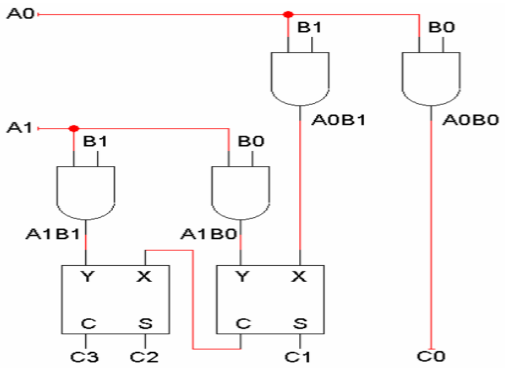
---

1. Consider the design shown below. Perform symbolic model checking to check if this design is correct given that the below design implements a mutual exclusivity principle (i.e., two grant/output signals are not active simultaneously) ? [6 marks]

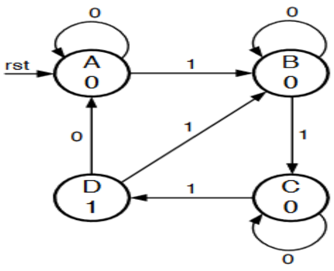


```
module design1 (input clock, input reset,input cx, input cy,output x,
output y);
logic x; logic y;
always @(posedge clock)
if (reset) begin
x <= 1'b1;
y <= 1'b0;
end else begin
x <= !y && cx;
y <= !x && !cx && cy;
end
endmodule
```

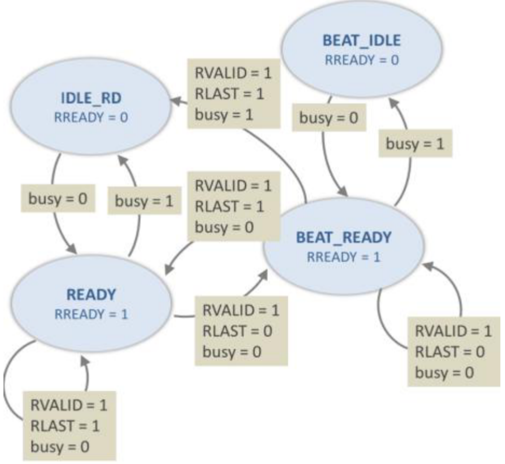
2. Consider the netlist shown below (containing 2 half-adders and logic gates). Use a formal verification method to check if this netlist correctly implements a 2-bit addition circuitry? [4 marks]



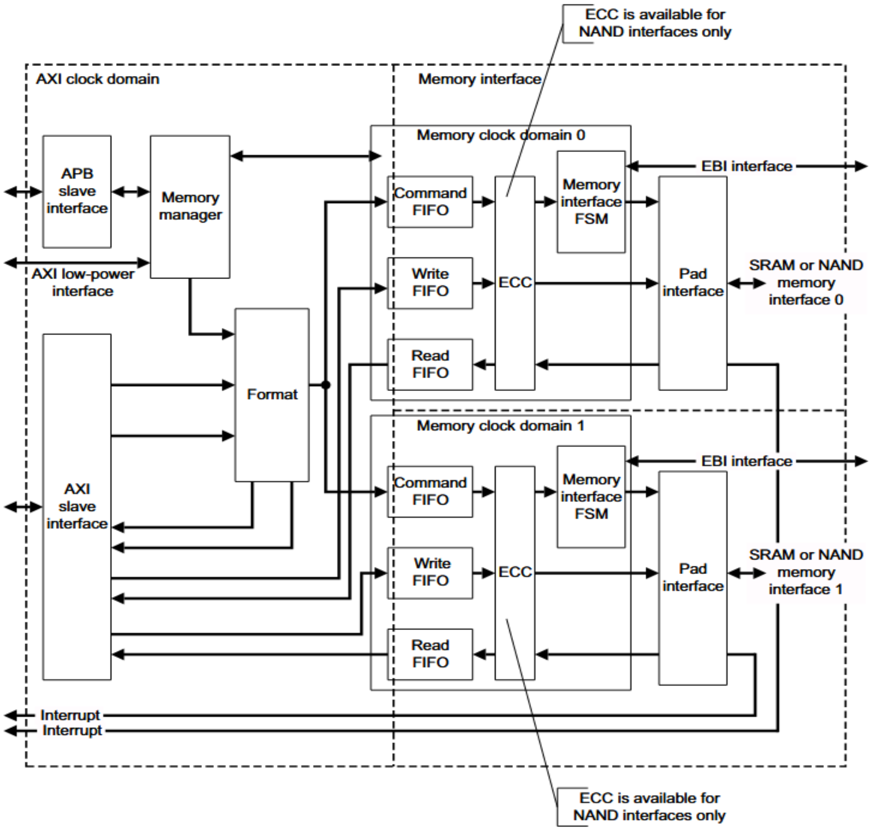
3. Consider the design model shown below. Derive any functionality of the design that can be related to the below model? [4 marks]



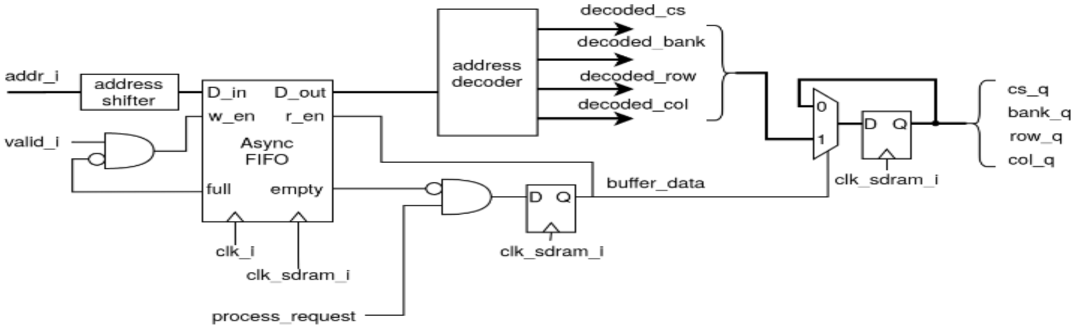
4. Below is the FSM description of Master Read Address Channel of AXI protocol (which is an industrial-level communication protocol for connecting multiple slaves and multiple masters and this protocol has a total of 5 channels for the interconnection). The triggering inputs are RVALID and RLAST from Slave and "busy" which is internal signal to denote master is not ready to receive data. The different states in the FSM are IDLE_RD, READY, BEAT_IDLE and BEAT_READY. As AXI is burst based if ARLEN is greater than zero more beats (beat means individual data transfer) need to be transferred and it moves on to the BEAT_IDLE and BEAT_READY states. **Write down properties of ensure logical correctness of this FSM?** [6 marks]



5. Consider the diagram of AXI Static Memory Controller (SMC) shown below (from the CoreLink Interconnection Suite by ARM). **Derive any abstraction of this original design and explain the impact of your abstraction on the formal verification coverage (when FV technique such as model checking is applied on the abstracted design)**? [5 + 2 marks]

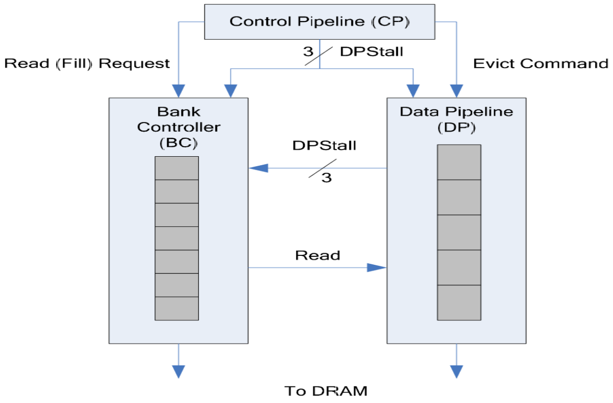


6. Consider the below design of address processing logic in a SDRAM controller of a modern system-on-chip that supports multiple clock domains for high performance. Since the address (addr_i) can come from a different clock domain, an asynchronous FIFO is used to syncrhonize the address from the external clock (clk_i) to the clock of the controller (clk_sdram_clk_i), which operates at the same frequency as the SDRAM. When the input address is valid, it is indicated with the valid_i signal, and the FIFO is not full the address is written into the FIFO. **With some proper assumptions, demonstrate how can you utilize any formal technique (such as FPV) for the verification of this design?** [6 marks]



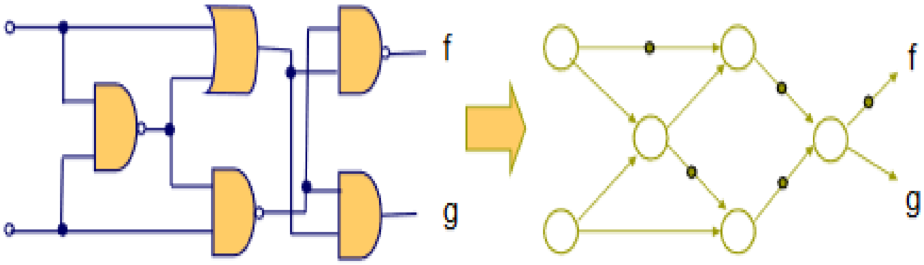
7. Processors have caches (small fast memories) to achieve high performance by cutting down the latency of memory accesses. These cache operations are managed by a memory controller. As caches have smaller size, it is continuously required to perform miss/evict operations in order to create room for the next set of data blocks.

Below is the figure of a memory controller. To process the miss/evict operation, the control pipeline (CP) issues a miss/evict command to the data pipeline (DP) to save (back to DRAM) the current contents of the cache line that will be used for the operation. Simultaneously, it issues a read request to the bank controller (BC) to fetch the targeted memory line. To prevent a race condition between the miss/evict and the read operations, the CP issues a 3-bit field (dpstall) with both operations. The FIFO to queue up the miss/evict operations is five levels deep, so three bits to hold dpstall is sufficient to prevent duplicate values. The read operation (in the BC) must wait for the corresponding dpstall value to be sent from the DP before it issues the read operation. For its part, the DP should only issue the corresponding dpstall value when it is complete with the miss/evict operation. **Suppose it is observed that the processor hangs at a particular memory address. Derive a semi-formal methodology to systematically debug this issue given the above design description and the below schematic.** [6 marks]



8. AIG (AND-Inverter-Graph) representations are utilized in industrial-level model checkers (and other FV tools) for understanding the input design descriptions. Basically, these representations

are directed acyclic graphs (DAG) in which nodes are the AND gates/NOT gates and the edges depict the wires/nets. So, we can assume that AIG graph is a model for representing a digital circuit. An AIG graph is shown as below (Note that the graph on the right denote AIG of the circuit shown in left. Left-side contains AND/OR/NOT whereas right-side denote only AND/NOT gates – the AND gates are shown by bigger circles and the inverter/NOT gates are shown by smaller circles). Write down an algorithm (pseudo-code/systematic procedure) for converting a generic sequential design (flip-flops/latches, logic gates etc.) into respective AIG representation? [5 marks]



9. Consider a multiprocessor design that has multiple cores (processors) connected to a shared main memory. The interface between the processor and the main memory is managed by a smart controller that has connections arranged in a group-wise manner.

   Specifically, this memory controller is connected to two groups of incoming cores (total eight cores) that are interfaced to 4 groups of memory banks (total eight memory banks of the main memory) organized as 2 banks in each group. Let's represent the incoming memory requests from cores as C0 to C7 and the memory banks are represented as M0 to M7. The core C0 to C3 are in one group; C4 to C7 in another group. The smart controller essentially performs some kind of arbitration for processing of these memory requests initiated by the cores. Consider these three simple behavioral rules that must be followed by this smart controller in order to ensure correct operation of the overall design. **Devise a formal verification test plan (or, write a suitable formal verification testbench) to verify the above three behaviors for this multiprocessor design? Comment on the completeness of the above three behavioral rules from the viewpoint of the full/exhaustive verification of this design. What is the scope of abstraction in your FV testbench?** [4 + 1 + 1 marks]

Behavior Rule Name	Meaning
First-come-first-serve	Memory requests originating from any core from any group sent to any of the memory banks must arrive at the memory bank in the same order as they were received at the controller.
Out-of-order	If memory requests from different requestor (i.e., core) groups compete for the same memory bank then any of the accepted memory requests can arrive at the given memory bank in any order.
Priority order	Memory requests originating from the core of the same group competing for the same memory bank must have a priority order. For example, if multiple instances from the group C0 to C3 sent requests in the same cycle to a bank Mj where j ranges from 0 to 7; then requests from C0 should be accepted before requests from C1, requests from C1 before C2, and C2 before C3. The same rule applies to requests from the second core group (C4 to C7).

   Hints: 1. Formal verification plan can consist of any one approach (out of theorem proving/equivalence checking/model checking) or a proper combination of them. 2. It is advisable to draw the system to help in visualization of the design internals and the associated signals.