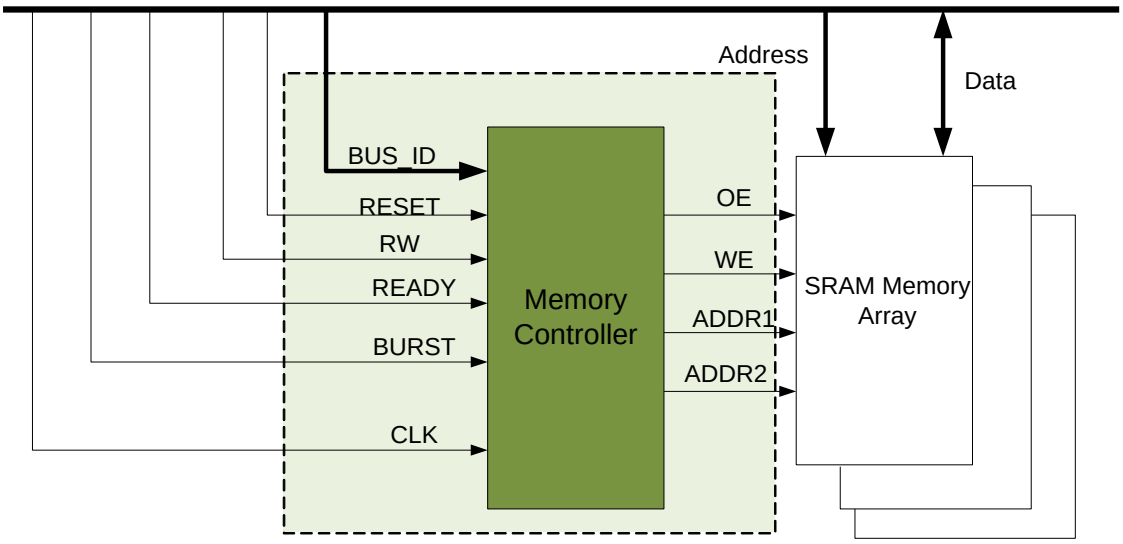
This is a warm-up lab assignment, meant to help you understand and get familiar with the process. You need to show the results to the TA or the instructor within **Friday, 18th October 2013**.

****

Design a **memory controller** whose block diagram is shown below.



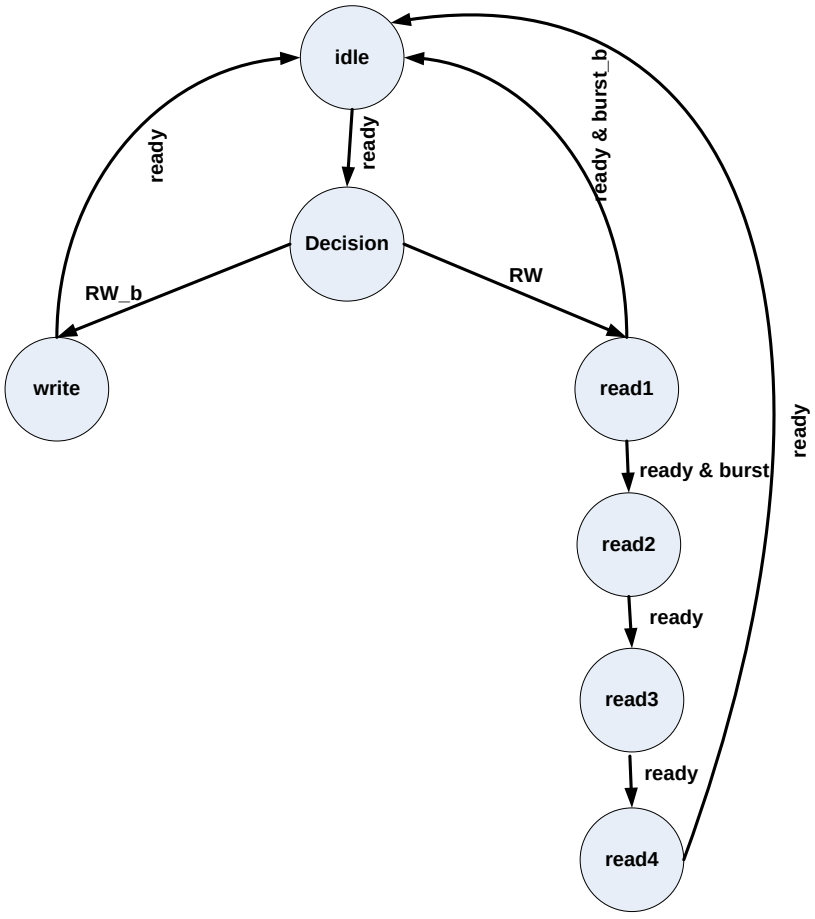
The **functional description of the state machine** is as follows.

- First, a synchronous reset places the state machine in the ***idle*** state. When the memory buffer is not being accessed, the controller remains in the ***idle*** state.
- If the **BUS_ID** is asserted as **F3** (hex) while the controller is in ***idle***, then the machine transitions to the ***decision*** state. (F3 is just an example; feel free to choose your bus id)
- On the next clock cycle, the controller transitions to either ***read1*** or ***write*** state, depending on the value of ***RW*** signal.
- If the access is a read, the controller branches to the read portion of the state machine. A single-word read is indicated by the assertion of ***ready***

without the assertion of *burst* while in the *read1* state. In this case, the controller returns to the *idle* state.

- A burst read is indicated by the assertion of both *ready* and *burst* while in the *read1* state. In this case the machine transitions through each of the read states, advancing on *ready*. *OE (Output Enable)* is asserted during each of the read cycles. *Addr* is incremented in successive read cycles following the first.
- If the access is a write, it can only be single-word write.
- Therefore, after determining that the access is a write in the decision state, the controller branches to the write portion of the state machine.
- It asserts *WE* to the memory buffer, waits for the *ready* signal from the bus, and then returns directly to the idle state.

The **state machine flow diagram** is shown below.



What you have to do:

1. Associate **outputs** with appropriate states based on the FSM functional description provided above.
2. Write **VHDL** or **Verilog** code to implement the state machine and the outputs.
3. Code a **testbench** to show that your design works as expected. In particular, you need to show that each of the 9 transitions (9 arrows) occur as expected. In addition, you have to add two test statements that will test for input combinations, such as **RW_b & ready_b** to verify that unintended transitions do not occur.
4. Run **VCS** to **simulate** your design with your testbench and show the output **waveforms** for each of the signals of interest.