Assignment 6

Due March 23, 1998

Design a DRAM controller to interface a bank of 4 Mbit DRAMs to a 386SX CPU. Your controller need only deal with read and refresh cycles.

The controller's inputs are:

- CLK2: The CPU's clock input. The clock has a 50% duty cycle and each bus cycle starts on the rising edge of the clock. There are 4 clock cycles per bus cycle. Each read or refresh cycle should take exactly 4 clock cycles.
- ADS*: An 'address valid' signal from the CPU. This signal is only asserted at the start of the second and third clock cycles of a bus cycle.
- A: A 22-bit address bus from the CPU. This address is valid only during the first two clock cycles of a bus cycle (while ADS* is valid).
- RFRSH: A refresh request input from a external circuit. This signal is synchronized so that it will be active at the start of the second clock cycle of a bus cycle (like ADS*) if a memory refresh needs to be generated. It will stay active for two clock periods.

The controller's outputs are:

- READY*: A signal to the CPU indicating the bus cycle is complete. It is asserted at the start of the last clock cycle of a bus cycle if the CPU can continue with the next bus cycle.
- RAMA: The multiplexed 10-bit address supplied to the DRAMs. The DRAMs have zero ns setup times and a hold times of less than one clock period for both row and column addresses.
- CAS and RAS: CAS* and RAS* strobes. Their behaviour must agree with the timing diagram shown below. Note that these signals are active-low and that the refresh bus cycle uses CAS*-before-RAS* refresh.

The timing diagram in Figure 1 shows the desired behaviour of the DRAM controller signals for RAM read and refresh cycles.

Write a synthesizeable VHDL description of a circuit that implements this DRAM controller. Test your design with the supplied test bench. Synthesize your code using the 'class' target library. Submit your VHDL code, the testbench output and a schematic.

Use the following VHDL entity (signals ending in "_n" are active-low):

```
entity dramc is
   port ( clk2, rfrsh, ads_n : in std_logic ;
      a : in std_logic_vector (21 downto 0) ;
      ready_n, ras_n, cas_n : out std_logic ;
      rama : out std_logic_vector (10 downto 0) ) ;
end ;
```

Hint: Use a Moore state machine for your controller.

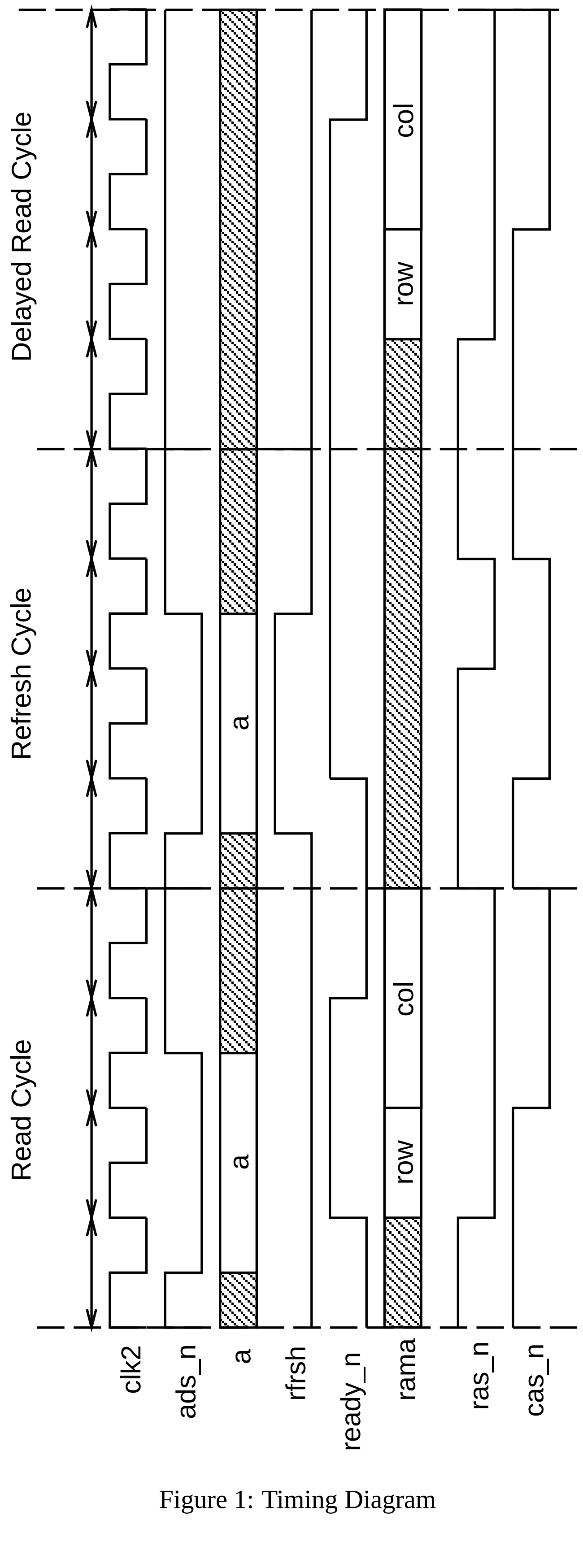

Figure 1: Timing Diagram