Interrupt Controller

Overview:

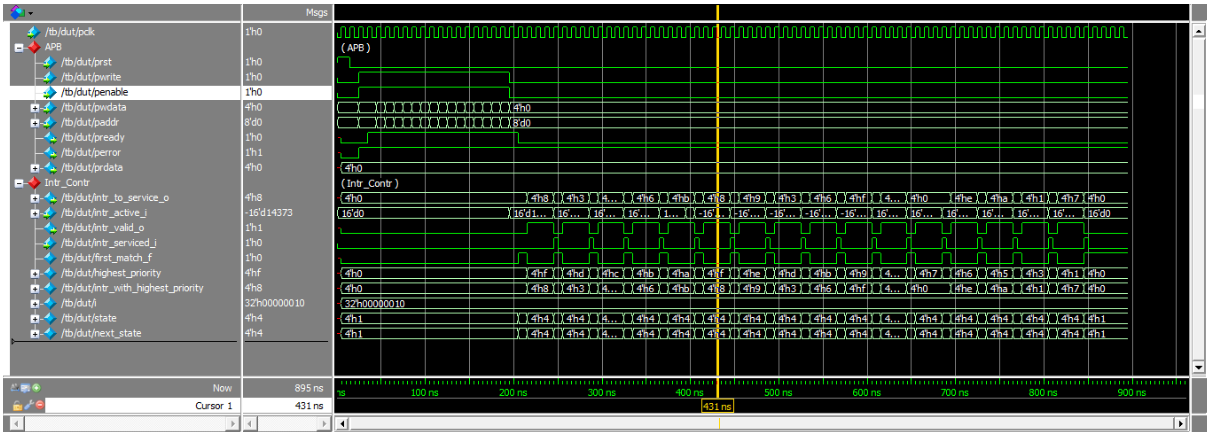
In any system, Processor acts as the **MASTER** and the other peripheral controller act as the **SLAVE**. Slaves can't directly get into the processor as the processor will be working on some other interrupts at the same time. Thus, we are in need of Interrupt Controller to manage the interrupts.

*Working***:**

Processor will have numerous peripherals (KYB, USB, etc.,) via AMBA APB protocol and they all want to use the processor, therefore it is the responsibility of the Interrupt Controller to prioritise the interrupts based on the necessity.
Assume that we have three scenarios;

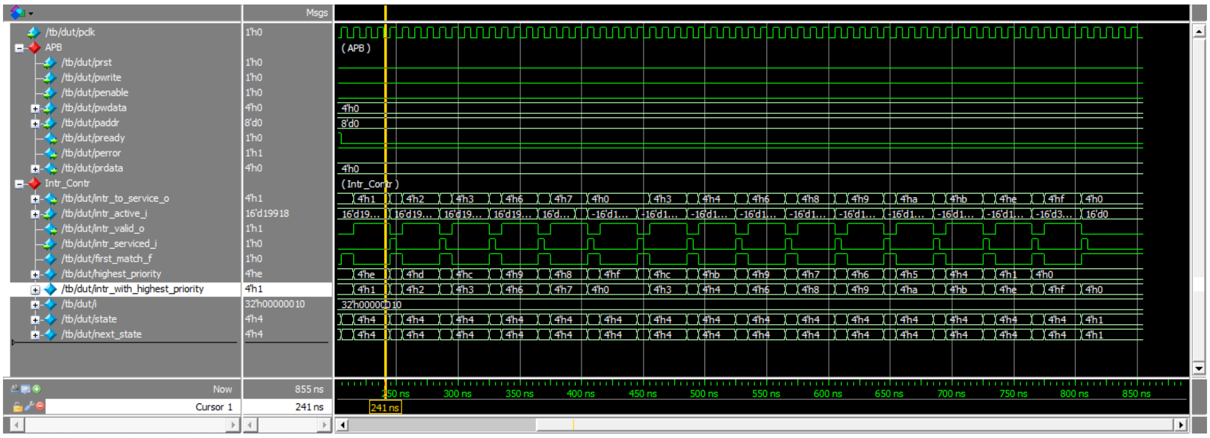
- Lowest peripheral has the lowest priority
- Lowest peripheral has the highest priority
- Random peripheral has the random priority

Design a Verilog code for Interrupt Controller and write a testbench to verify it.



```
                                              Msgs
/tb/dut/pclk                                  1'h0
APB
  /tb/dut/prst                                1'h0
  /tb/dut/pwrite                              1'h0
  /tb/dut/penable                             1'h0
  /tb/dut/pwdata                              4'h0
  /tb/dut/paddr                               8'd0
  /tb/dut/pready                              1'h0
  /tb/dut/perror                              1'h1
  /tb/dut/prdata                              4'h0
Intr_Contr
  /tb/dut/intr_to_service_o                   4'h8
  /tb/dut/intr_active_i                       -16'd14373
  /tb/dut/intr_valid_o                        1'h1
  /tb/dut/intr_serviced_i                     1'h0
  /tb/dut/first_match_f                       1'h0
  /tb/dut/highest_priority                    4'hf
  /tb/dut/intr_with_highest_priority          4'h8
  /tb/dut/i                                   32'h00000010
  /tb/dut/state                               4'h4
  /tb/dut/next_state                          4'h4
Now                                           895 ns
Cursor 1                                      431 ns
```

Timing diagram of random peripheral has the random priority



```
                                              Msgs
/tb/dut/pclk                                  1'h0
APB
  /tb/dut/prst                                1'h0
  /tb/dut/pwrite                              1'h0
  /tb/dut/penable                             1'h0
  /tb/dut/pwdata                              4'h0
  /tb/dut/paddr                               8'd0
  /tb/dut/pready                              1'h0
  /tb/dut/perror                              1'h1
  /tb/dut/prdata                              4'h0
Intr_Contr
  /tb/dut/intr_to_service_o                   4'h1
  /tb/dut/intr_active_i                       16'd19918
  /tb/dut/intr_valid_o                        1'h1
  /tb/dut/intr_serviced_i                     1'h0
  /tb/dut/first_match_f                       1'h0
  /tb/dut/highest_priority                    4'he
  /tb/dut/intr_with_highest_priority          4'h1
  /tb/dut/i                                   32'h00000010
  /tb/dut/state                               4'h4
  /tb/dut/next_state                          4'h4
Now                                           855 ns
Cursor 1                                      241 ns
```

Timing diagram of lowest peripheral has the highest priority