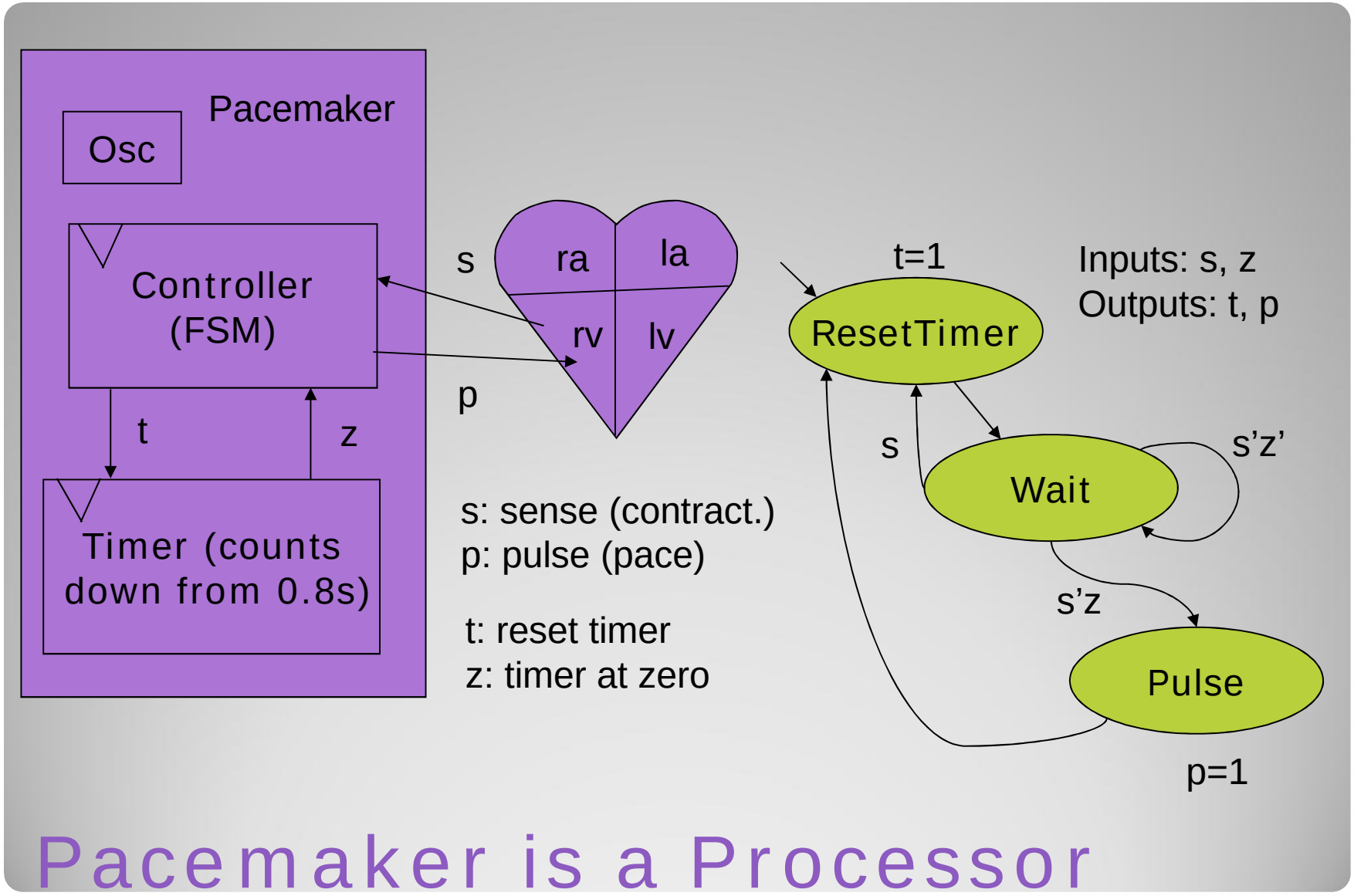
TCES 330

Processors

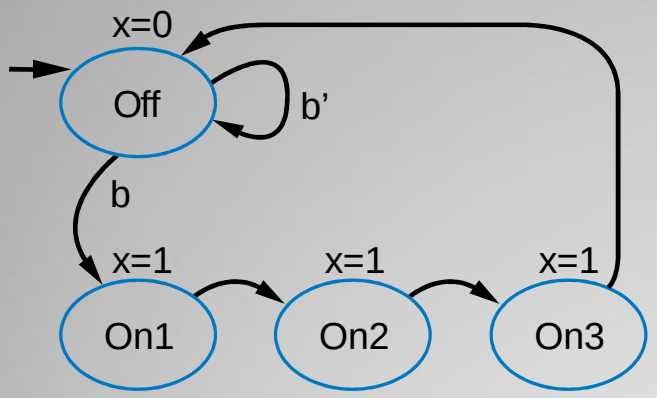


Pacemaker is a Processor

Recall the Laser Surgery FSM

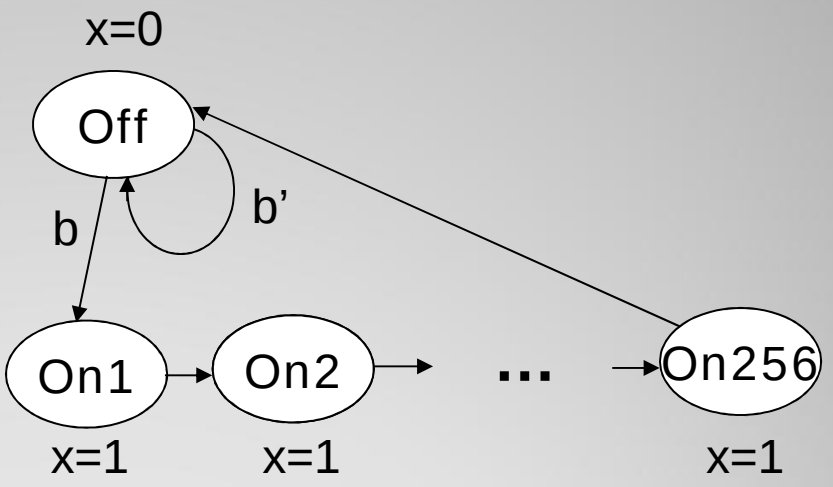
Recall:

Inputs: b; Outputs: x



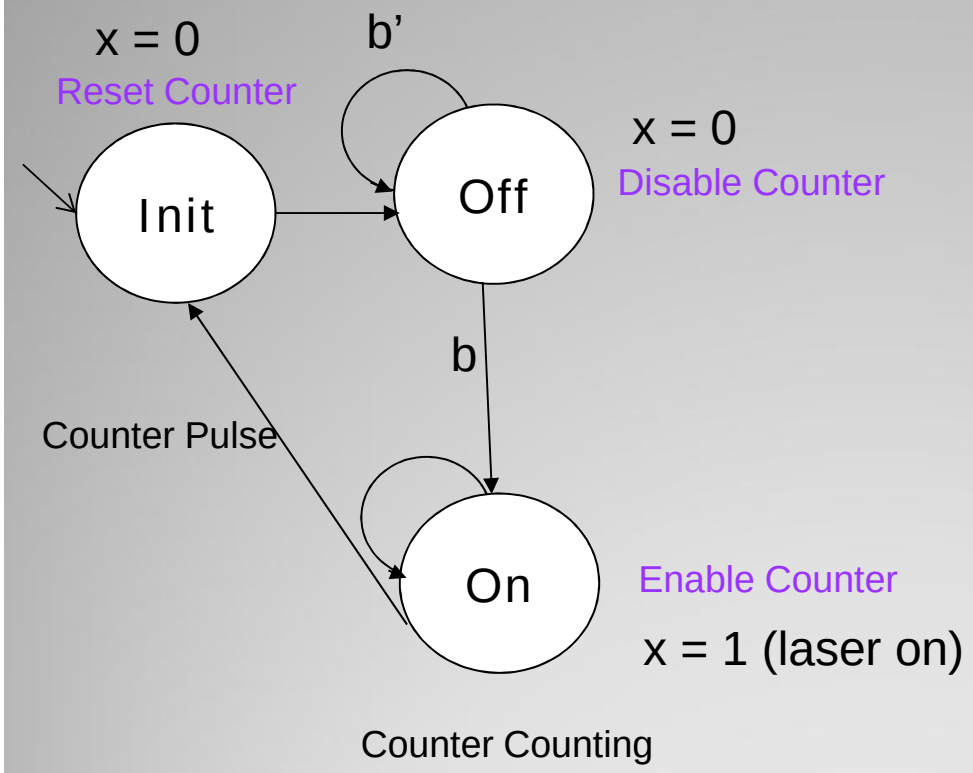
Laser is ON for three clock ticks

But what if we needed the laser to stay on, 16 or 256 clock cycles?

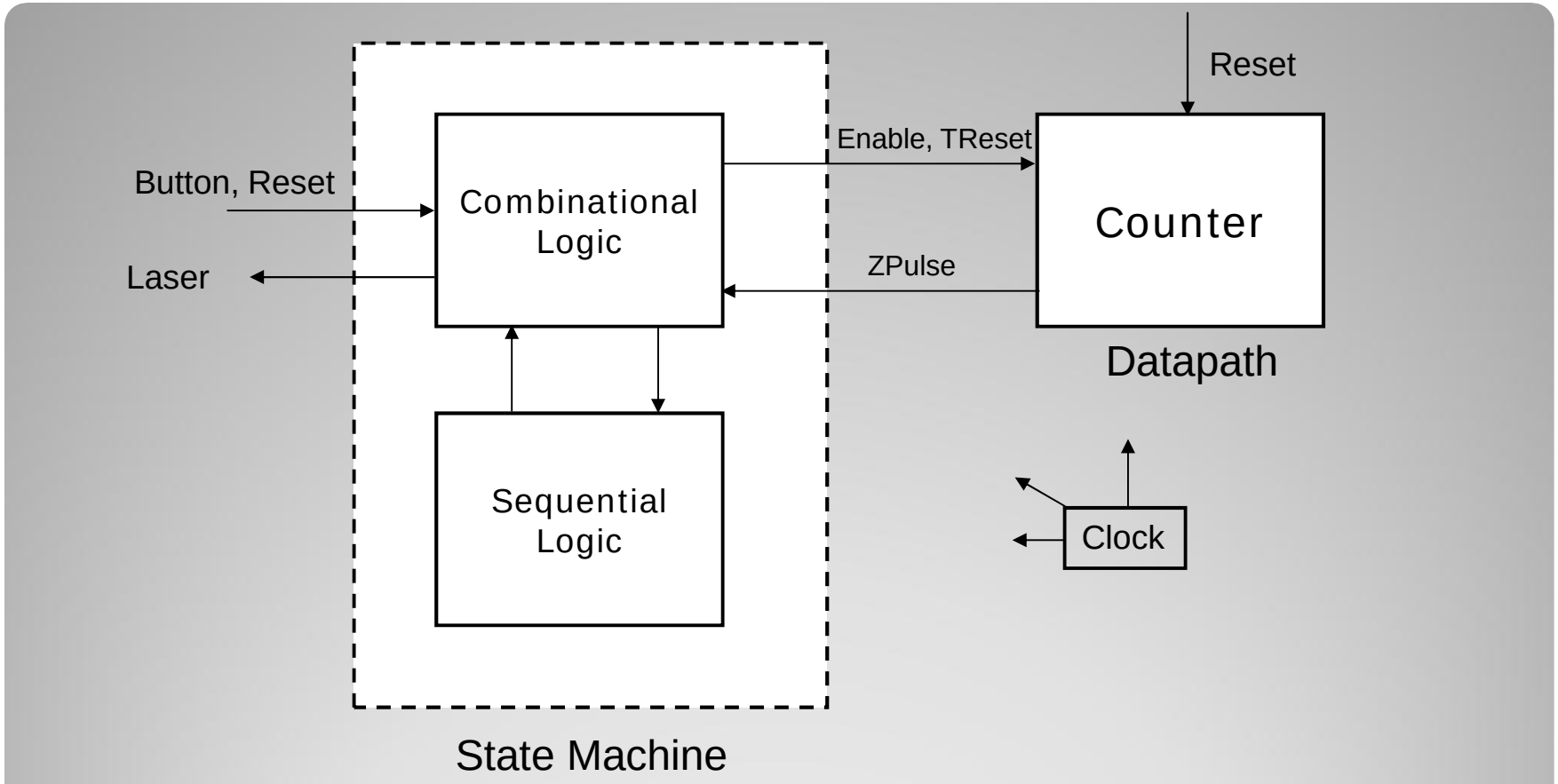


Maybe not

Laser FSM Using Datapath

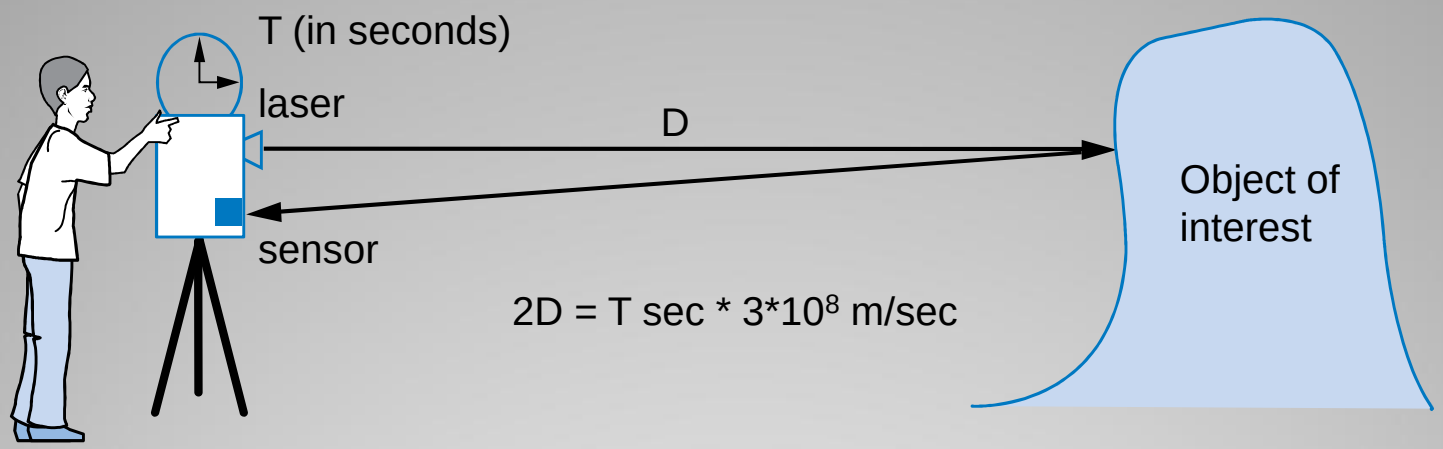


- Add a counter that we enable in the On State
- When the counter pulses at the desired count, we change state



Note: This looks a lot like the *Pacemaker* project

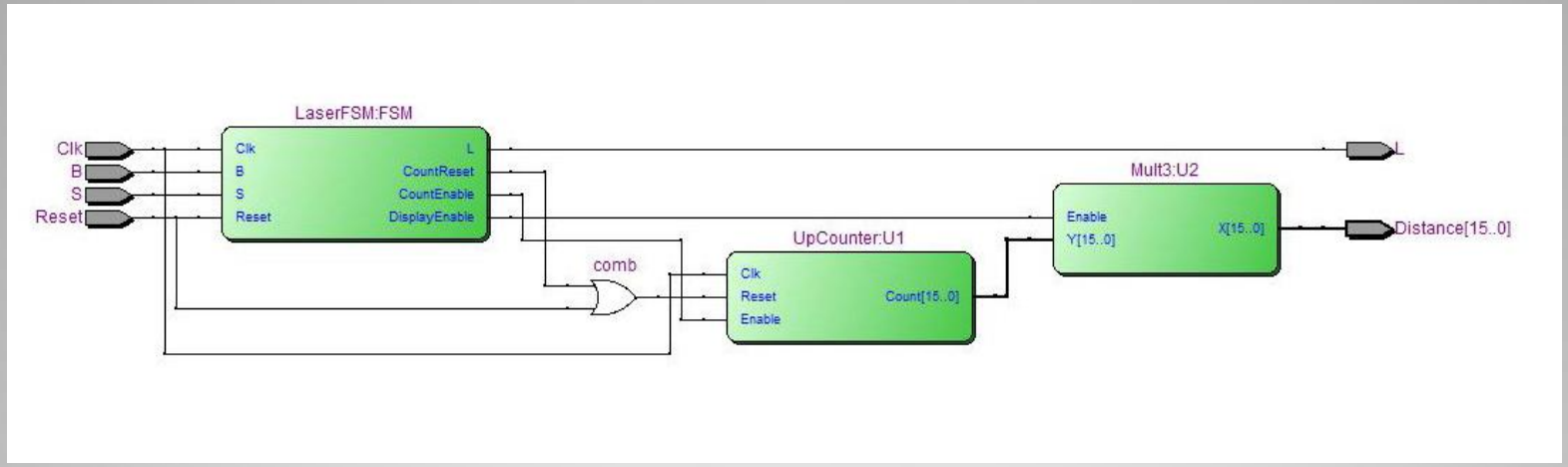
Laser Processor Block Diagram



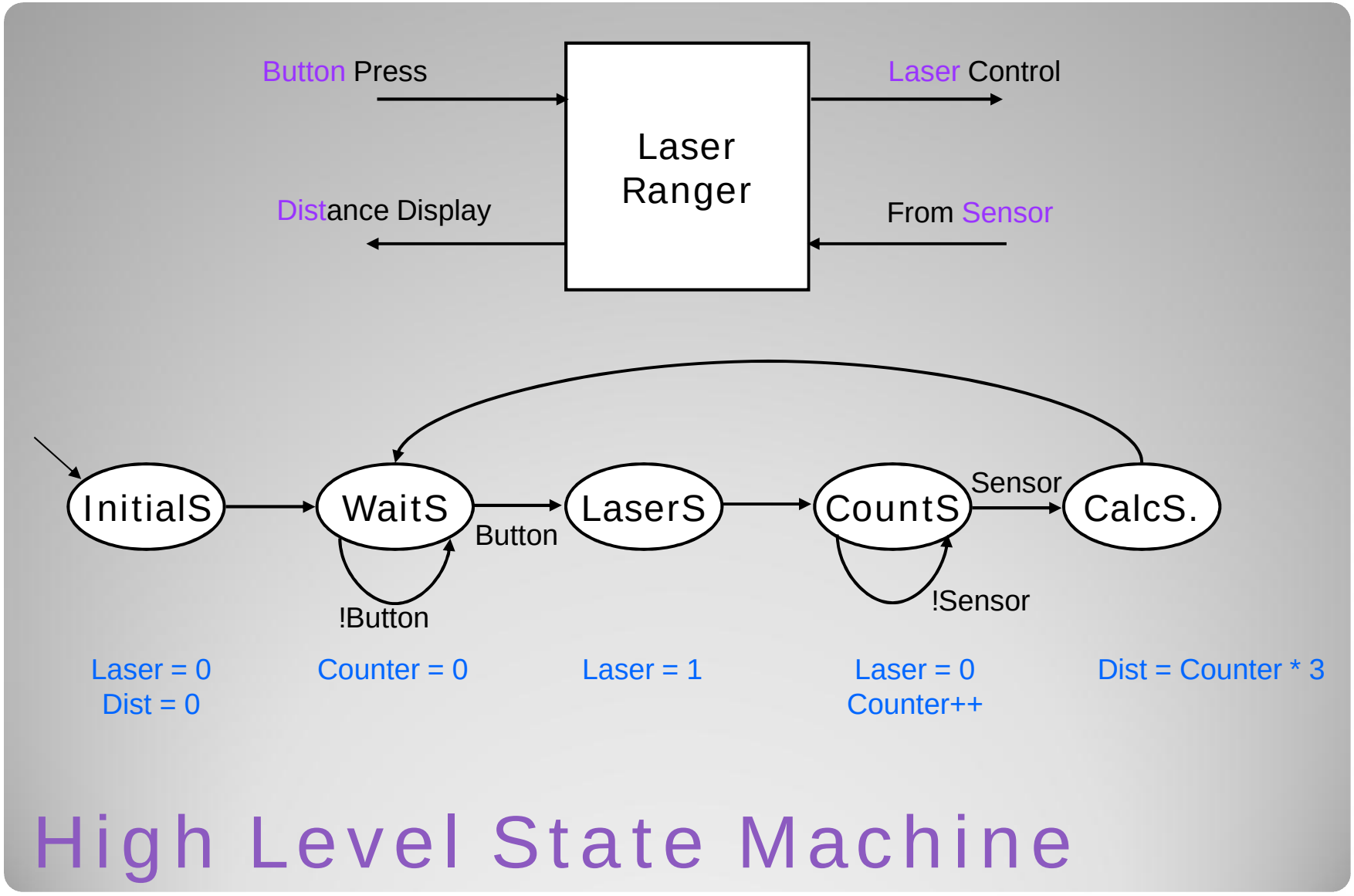
- Example of how to create a high-level state machine to describe desired processor behavior
- Laser-based distance measurement – pulse laser, measure time T to sense reflection
  - Laser light travels at speed of light, $3*10^8$ m/sec
  - Distance is thus D = T sec * $3*10^8$ m/sec / 2

From F. Vahid's *Digital Design*

Laser Ranging System

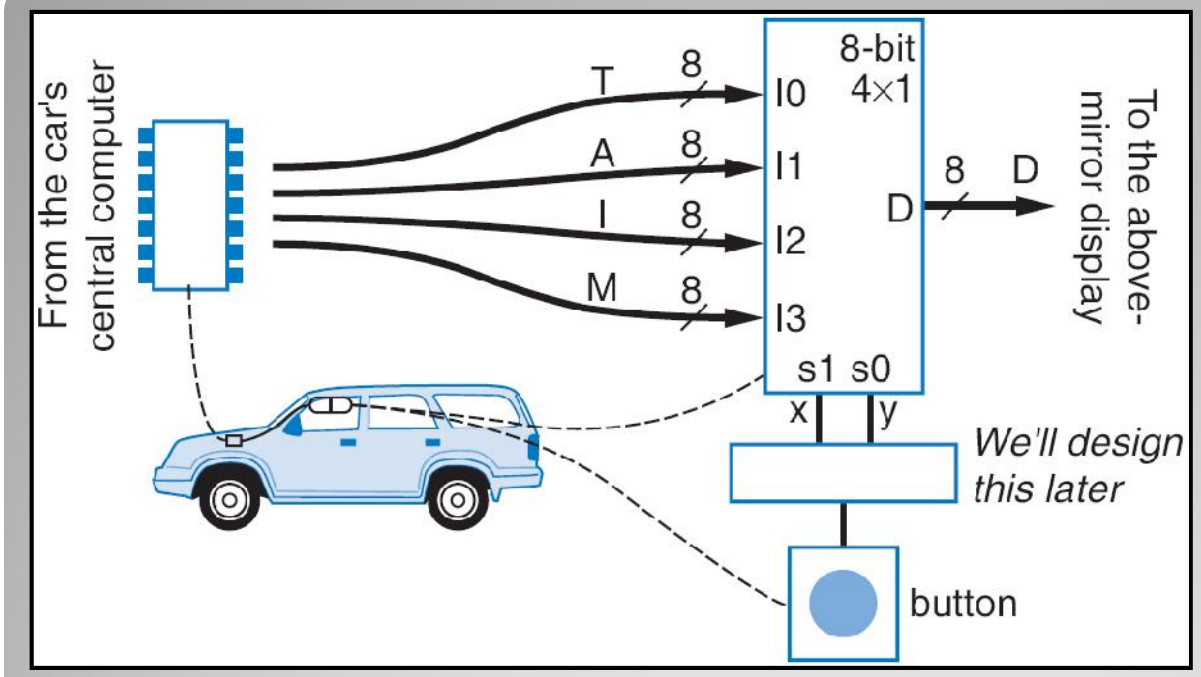


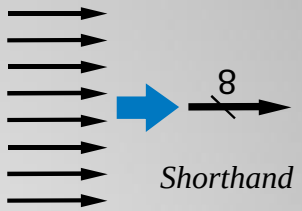
Laser Ranger Block Diagram



High Level State Machine For Laser Ranger

N-bit Mux Example

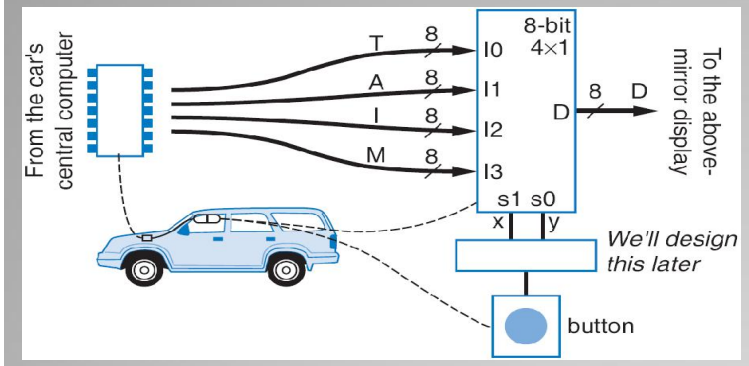




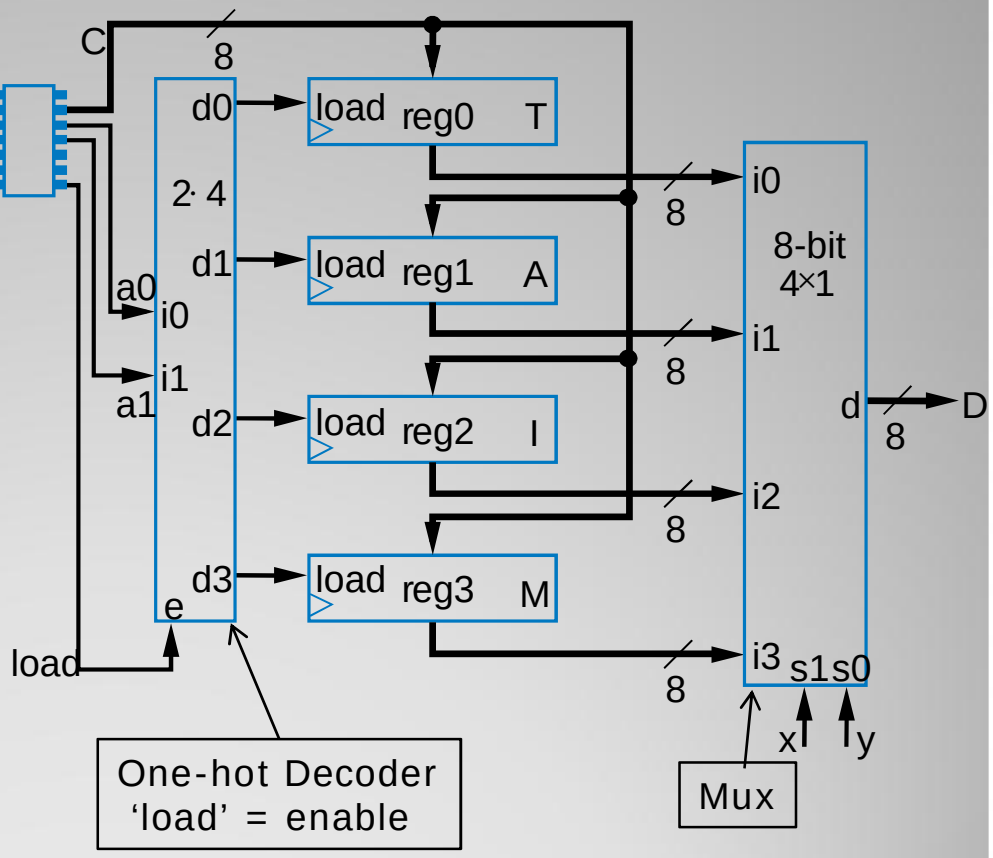
Shorthand notation

- Four possible display items
  - Temperature (T), Average miles-per-gallon (A), Instantaneous mpg (I), and Miles remaining (M) -- each is 8-bits wide
  - Choose which to display using two inputs x and y
  - Could Use 8-bit 4x1 mux

From F. Vahid



- Recall: Four simultaneous values from car's computer
- To reduce wires: Computer writes only 1 value at a time, loads into one of four registers
  - Was: 8+8+8+8 = 32 wires
  - Now: 8 +2+1 = 11 wires

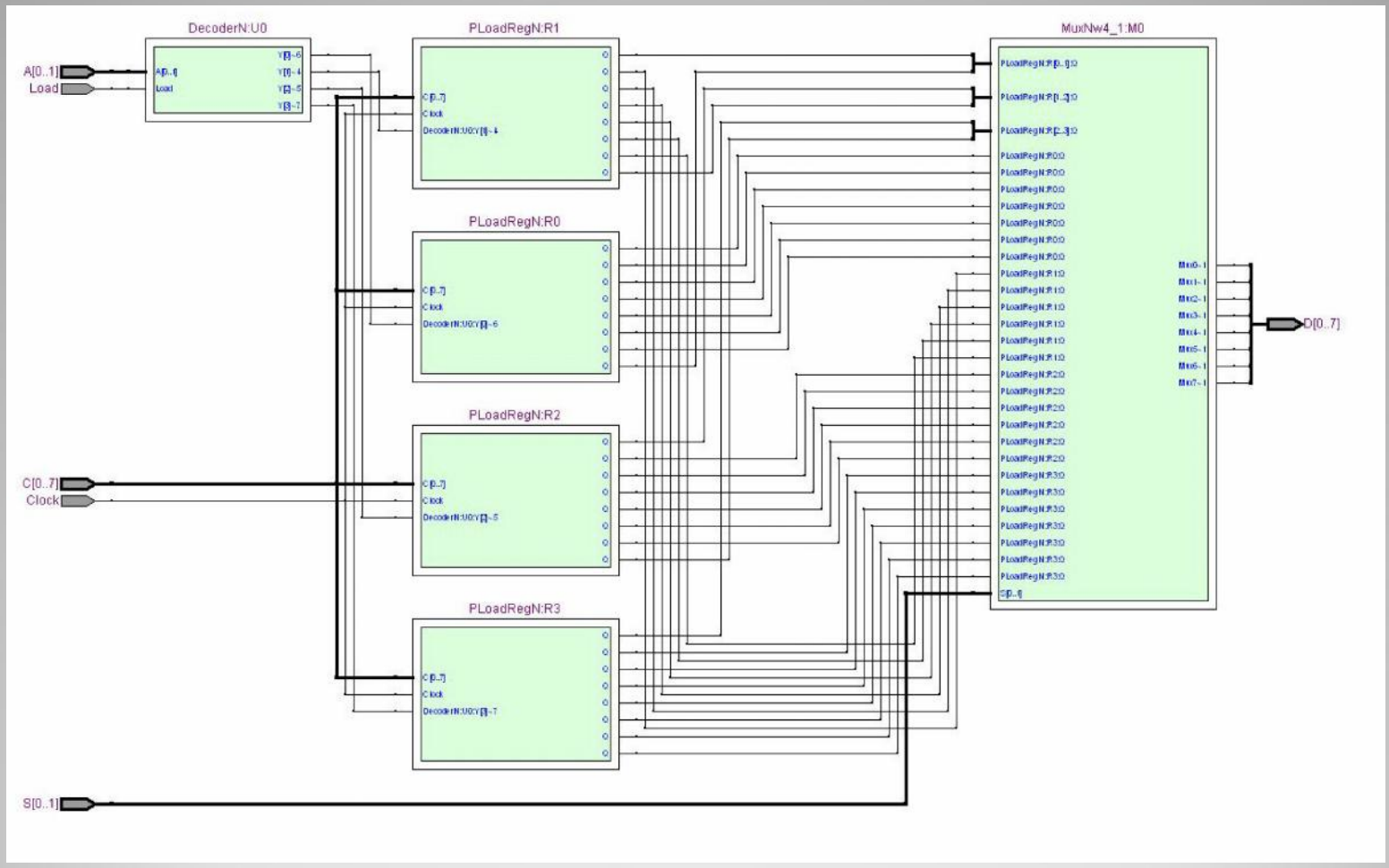


Register Example: Above-Mirror Display

From F. Vahid

```
module MirrorDisplay( C, A, Load, Clock, S, D );
  input [7:0] C; // data from microprocessor
  input [1:0] A; // data key (T=0, A=1, I=2, M=3)
  input Load;    // data ready signal from micro
  input Clock;   // our system clock
  input [1:0] S; // users selection (what to display)
  output [7:0] D;  // data to display (we don't worry about how)
  wire [3:0] RD;   // decoder output

  DecoderN #(.N(2)) U0( A, Load, RD ); // 2 address lines

  wire [7:0] I3, I2, I1, I0;  // register outputs, mux inputs
  // the registers:
  PLoadRegN #(.N(8)) R0( C, RD[0], Clock, I0 );  // T-register
  PLoadRegN #(.N(8)) R1( C, RD[1], Clock, I1 );  // A-register
  PLoadRegN #(.N(8)) R2( C, RD[2], Clock, I2 );  // I-register
  PLoadRegN #(.N(8)) R3( C, RD[3], Clock, I3 );  // M-register

  // the mux:
  MuxNw4_1 #(.N(8)) M0( S, I0, I1, I2, I3, D );

endmodule
```

Mirror Display Verilog



What Quartus Built