Hardware Modeling [VU] (191.011)
– WS25 –
Finite-State Machine Implementation (in VHDL)

Florian Huemer & Sebastian Wiedemann & Dylan Baumann

WS 2025/26

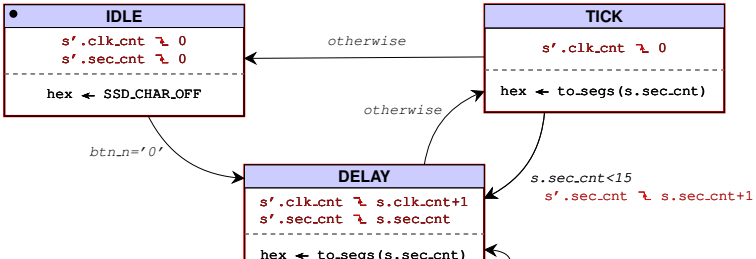
Modified: 2025-12-16, 16:08 (f8a58e9)

Implementing FSMs in VHDL

- Conversion of FSM model to VHDL code follows simple steps
  1. Create an enumeration type for the FSM state
  2. Create a record type for the state register
  3. Create signals for the current (`s`) and next (`s_nxt`) register value
  4. Implement the state register in a sync. process
  5. Implement the next-state and output logic in comb. process(es)
- 2- vs. 3-process method
  - Overlap of next-state and output logic $\Rightarrow$ 2-process usually more readable
  - Recommendation: Use the 2-process method

Example: FSM Types

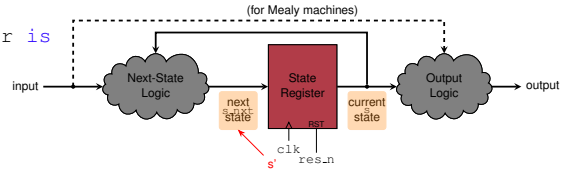
1. Create an enumeration type for the FSM state
2. Create a record type for the state register



Example: State Register

3. Create signals for the current (`s`) and next (`s_nxt`) register value
4. Implement the state register in a sync. process

```
18 architecture arch of advanced_timer is
19   -- enum and record declarations
```

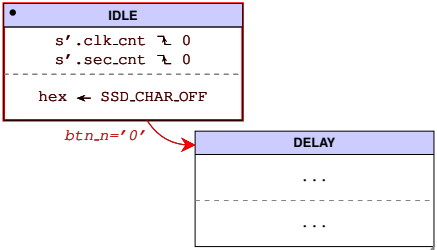


Example: Next-State and Output Logic

5 Implement the next-state and output logic in comb. process
- **Recall**: Path without assignment through comb. process ⇒ latch!

⇒ Recommendation: Use default assignments
- Define next-state and output logic per state

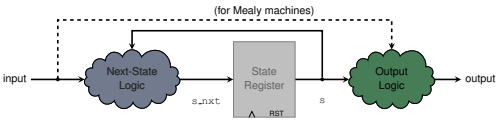
```
36 comb : process(all)
37 begin
38   s_nxt <= s;
39   hex <= to_segs(std_ulogic_vector(s.tick_cnt));
```



Example: Next-State and Output Logic (cont'd)


```
36 comb : process(all)
37 begin
38   s_nxt <= s;
39   hex <= to_segs(std_ulogic_vector(s.tick_cnt));
40
41   case s.state is
42     when IDLE =>
43       s_nxt.clk_cnt <= (others => '0');
44       s_nxt.tick_cnt <= (others => '0');
45       hex <= SSD_CHAR_OFF;
46       if btn_n = '0' then
47         s_nxt.state <= DELAY;
48       end if;
49
50     when DELAY =>
51       if s.clk_cnt < CLK_FREQ-1 then
52         s_nxt.clk_cnt <= s.clk_cnt + 1;
53       else
54         s_nxt.state <= TICK;
55       end if;
```

```
55     when TICK =>
56       s_nxt.clk_cnt <= (others => '0');
57       s_nxt.state <= DELAY;
58       if s.tick_cnt < 15 then
59         s_nxt.tick_cnt <= s.tick_cnt + 1
60       else
61         s_nxt.state <= IDLE;
62       end if;
```



Lecture Complete!