

Hardware Modeling [VU] (191.011)

– WS25 –

Finite-State Machine Basics

Florian Huemer & Sebastian Wiedemann & Dylan Baumann

WS 2025/26

Introduction

HWMod
WS25

FSM Basics
Introduction
FSM Circuits
Example: Timer

- Fundamental structure in digital designs
- An FSM consists of a
 - set of inputs, outputs and states
 - transition function
- Synchronous FSMs: State changes **only** happen at clock events!
- Important VHDL concepts used in the lecture
 - Enumeration and record types
 - Behavioral modeling of combinational logic
 - Sequential circuit elements (i.e., registers)

State Machine as Digital Circuits

HWMod
WS25

FSM Basics

Introduction

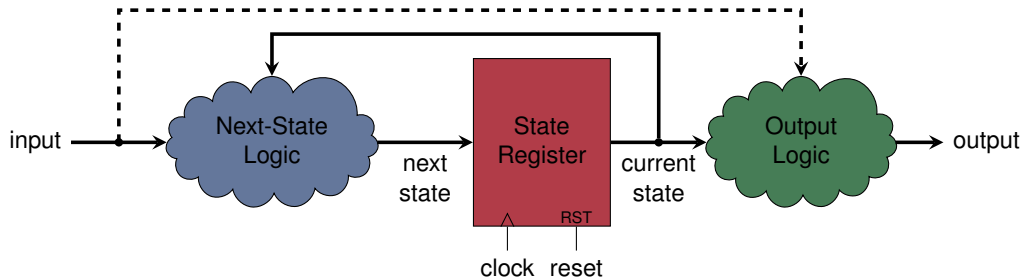
FSM Circuits

Next-State Logic

State Register

Output Logic

Example: Timer



Next-State Logic

HWMod
WS25

FSM Basics

Introduction

FSM Circuits

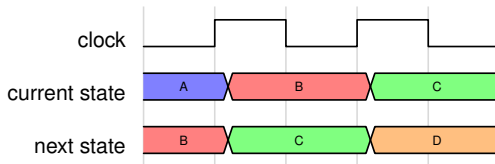
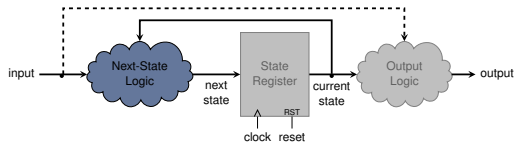
Next-State Logic

State Register

Output Logic

Example: Timer

- Purely combinational
 - **Must not** read the clock or reset
 - **Must not** contain latches
- Implements the transition function
 - by reading the **input** and the **current state**
 - to calculate the **next state** of the FSM
- State changes are synchronous!



State Register

HWMod
WS25

FSM Basics

Introduction

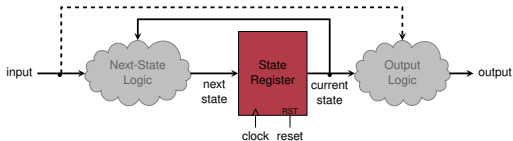
FSM Circuits

Next-State Logic

State Register

Output Logic

Example: Timer



- Only synchronous component in the FSM
- Reads **next state** and outputs **current state**
- **Strictly** stick to the code patterns for registers!
- Reset value defines initial state

Output Logic

HWMod
WS25

FSM Basics

Introduction

FSM Circuits

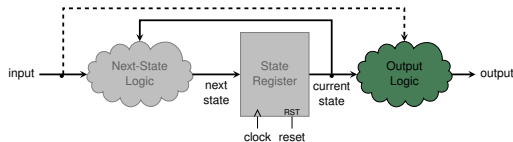
Next-State Logic

State Register

Output Logic

Example: Timer

- Purely combinational
 - **Must not** read the clock or reset
 - **Must not** contain latches
- Produces the actual **output** signals
- Defines the FSM type
 - Moore: Output depends solely on **current state**
 - Mealy: Output depends on **current state** and **input**
- Mealy machines
 - can react to input changes in the **same** clock cycle (i.e., combinational)
 - (sometimes) require fewer states than a similar Moore machine
- Can share logic with the next-state logic



Output Logic - Combinational Loops

HWMod
WS25

FSM Basics

Introduction

FSM Circuits

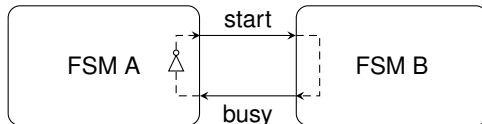
Next-State Logic

State Register

Output Logic

Example: Timer

- Interacting Mealy FSMs can lead to combinational loops!
- Example
 - FSM A: “Start when not busy”
 - FSM B: “Assert busy when started”



- Only use Mealy machines when **really** necessary

Simple Timer - Specification

HWMod
WS25

FSM Basics

Introduction

FSM Circuits

Example: Timer

Specification

Circuit

Implementation

Behavior Description

The `simple_timer` module keeps an internal N -bit counter that is (synchronously) incremented as long as `en` is asserted. When the counter reaches its maximum value ($2^N - 1$) it overflows back to zero. If the counter hits its maximum and `en` is asserted the `tick` output is asserted.

Interface

```
1 entity simple_timer is
2   generic (
3     N : positive := 8
4   );
5   port (
6     clk    : in std_ulogic;
7     res_n  : in std_ulogic;
8     en     : in std_ulogic;
9     tick   : out std_ulogic
10  );
11 end entity;
```


Simple Timer - Circuit

HWMod
WS25

FSM Basics

Introduction

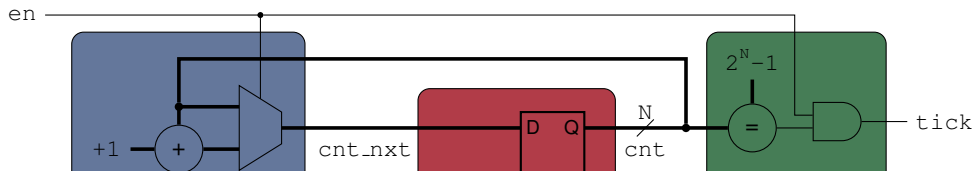
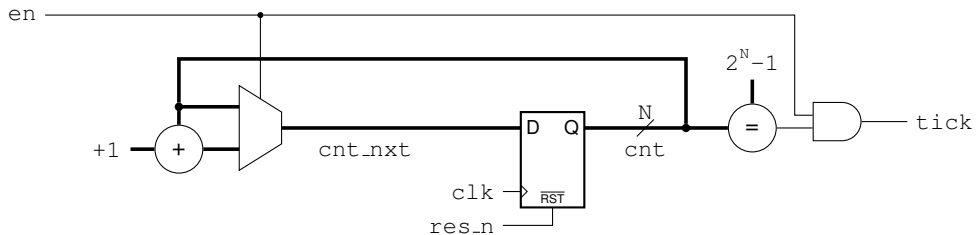
FSM Circuits

Example: Timer

Specification

Circuit

Implementation



Simple Timer - Implementation

HWMod
WS25

FSM Basics

Introduction

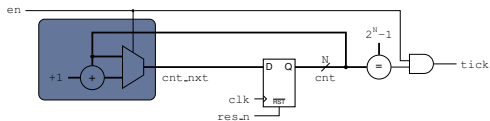
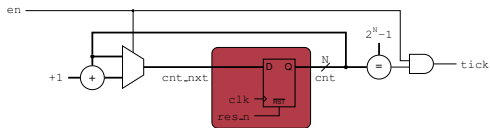
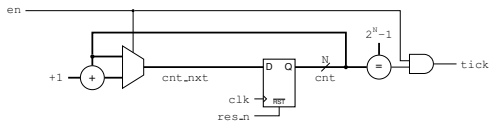
FSM Circuits

Example: Timer

Specification

Circuit

Implementation



Lecture Complete!