

CprE 488 – Embedded Systems Design

Exam 1 Review

Phillip Jones

Electrical and Computer Engineering

Iowa State University

Try not. Do or do not, there is no try. – Yoda

Announcements

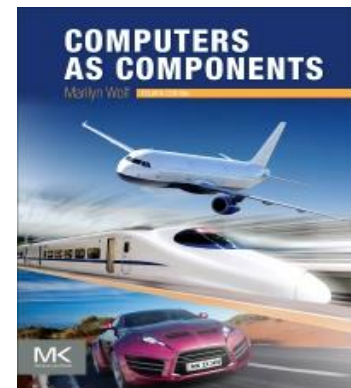
- Exam 1: in class
 - **One-side** 8.5x11" page of **hand-written** notes
 - 70 minutes
 - Recommend having a calculator
- Two Parts:
 - Part 1: Canvas (Bring an electronic device to use)
 - Part 2: Paper written

Exam Focus

- General Concepts (textbook + lab + lecture)
 - Definitions, examples, trends
 - System specification and design
- Embedded Platforms
 - System-on-Chip
 - Memory Mapped I/O
 - Buses (topologies, protocols, etc.)
 - DMA
 - FPGA-based platforms
- ARM ISA and Architecture, C55x, C64x
- Interfacing Technologies: UART, I2C, SPI, CAN
- Hardware Design (VHDL)

Textbook Readings

- Chapter 1: Embedded Computing
- Chapter 4: Computing Platforms
- Chapter 2: Instruction Sets
- Chapter 3.5: Memory Systems
- Interface Technologies
 - 3rd Edition: Section 8.4 (CAN, I2C, IP)
 - 4th Edition: Chapter 8.4.1, 8.4.2(IP), 9(CAN), 10.4.5(I2C)



Lost, but (hopefully) not forgotten

Lab Assignments

- **MP-0: NES emulator**
 - VGA interface
 - VDMA
 - IP datasheet interpretation
 - System development using IP integration
 - SW framework analysis
- **MP-1: UAV interface**
 - PPM protocol
 - State machine design (Free Range VHDL)
 - Memory mapped device registers
 - Waveform analysis (simulation and oscilloscope)

- Introduction to Embedded Systems
 - Properties of an embedded system
 - Design trade-offs
 - Levels of System abstraction
 - Design Flow

Lecture: Embedded Platforms

- Embedded Platforms
 - IP cores: relevance for SoC design
 - Platform teardown case studies
 - Bus-based communication
 - Types, attributes and tradeoffs
 - Arbitration
 - AMBA protocol example
 - Direct Memory Access (DMA)
 - Memory mapped I/O:
 - In terms of MP-0 / MP-1, and in general
 - FPGA technology and demos

Lecture: Processors and memory & VHDL

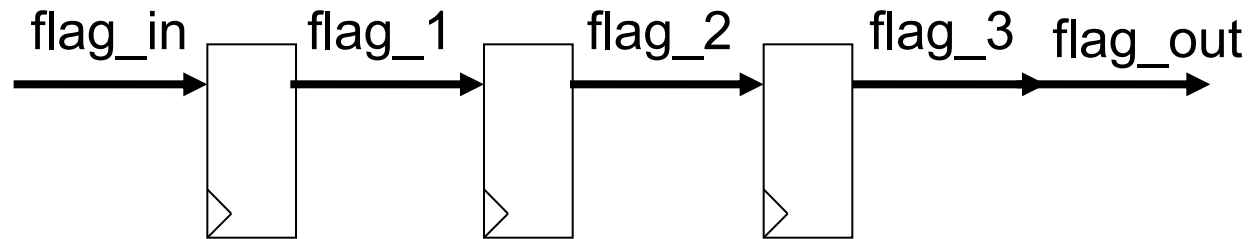
- Processors and memory
 - Taxonomy of computer architectures
 - ARM Instruction Set Architecture (ISA)
 - Register set
 - Program status and conditional execution
 - Addressing modes
 - TI C55x DSP (CISC)
 - TI C64x (VLIW)
- Discussion of hardware design in the context of
 - MP-1
 - FSM template (3 process: 2 clocked, 1 non-clocked)
 - General class discussions

Lecture: Interfacing Technologies

Protocol	Expandable	Multi-Master	Duplex	Robustness	Speed	Flow Control	Protocol Overhead
UART (2-wire)	No	No	Full	(Parity bit)	~100Kbps	No	Start, Stop, (parity)
I2C	Large	Yes	Half	ACK/NACK	S:100Kbps F:400Kbps Hs:3.4Mbps	Yes	Start, 7-bit addr, R/W, ACK/NACK
SPI	At cost of extra wire per slave	No	Full	None	8Mbps+	No	None
CAN	Large	Yes	Half	16-bit CRC ACK/NACK Differential signaling	1Mbps	No (for single frame)	Start, 11-bit ID, 6-bit control, 16-bit CRC, 2-bit ACK, 7-bit EOF

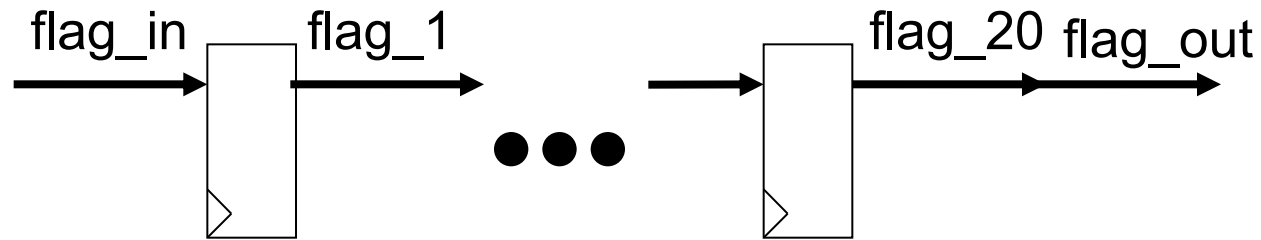
VHDL: Light Examples

Array Example (Delay Shift Register)



```
BEGIN
  My_process_1 : process (clk)
  Begin
    IF (clk'event and clk = '1') THEN
      flag_1 <= flag_in;
      flag_2 <= flag_1;
      flag_3 <= flag_2;
    END IF;
  End My_process_1;
  flag_out <= flag_3
END;
```

Array Example (Delay Shift Register)



```
type flag_reg_array is array (DLY-1 downto 0) of std_logic;  
signal flag_reg : flag_reg_array;
```

```
BEGIN
```

```
  My_process_1 : process (clk)
```

```
  Begin
```

```
    IF (clk'event and clk = '1') THEN
```

```
      flag_reg(flag_reg'high downto 0) <=
```

```
        flag_reg(flag_reg'high-1 downto 0) & flag_in;
```

```
    END IF;
```

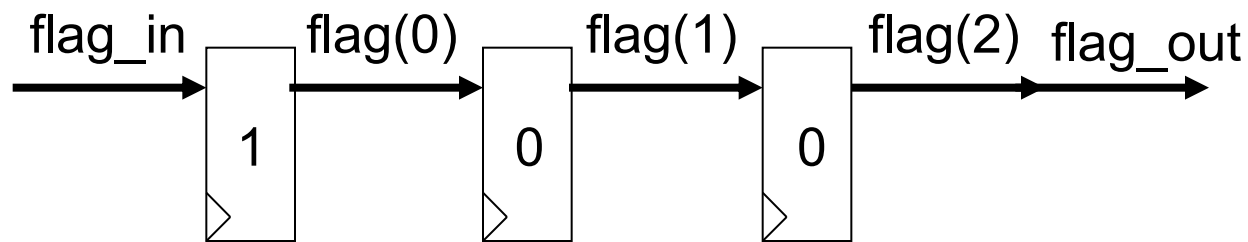
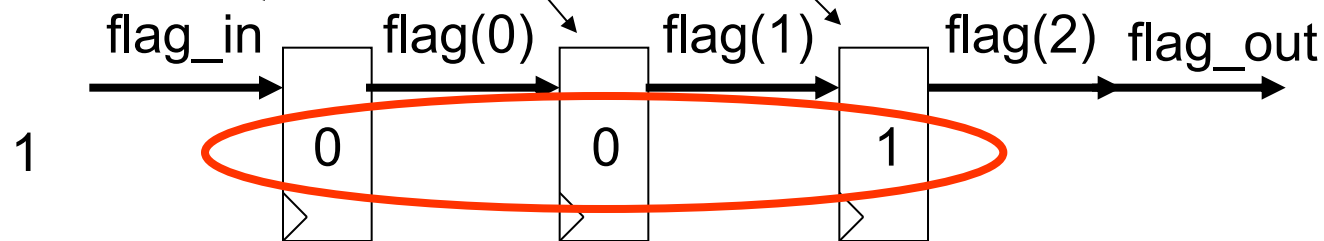
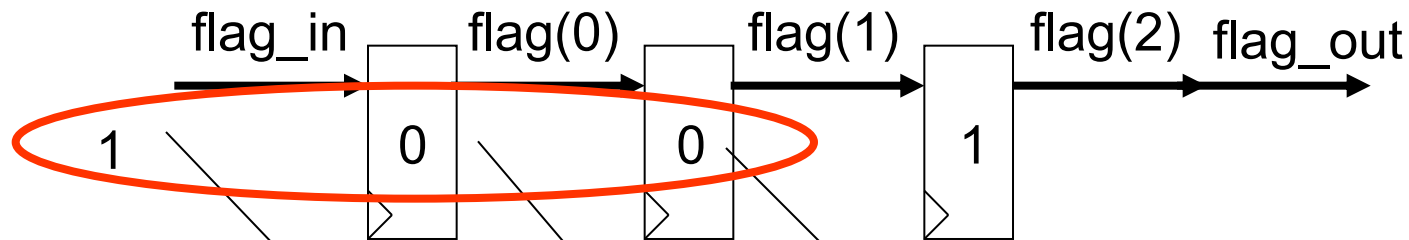
```
  End My_process_1;
```

```
  flag_out <= flag_reg(flag_reg'high);
```

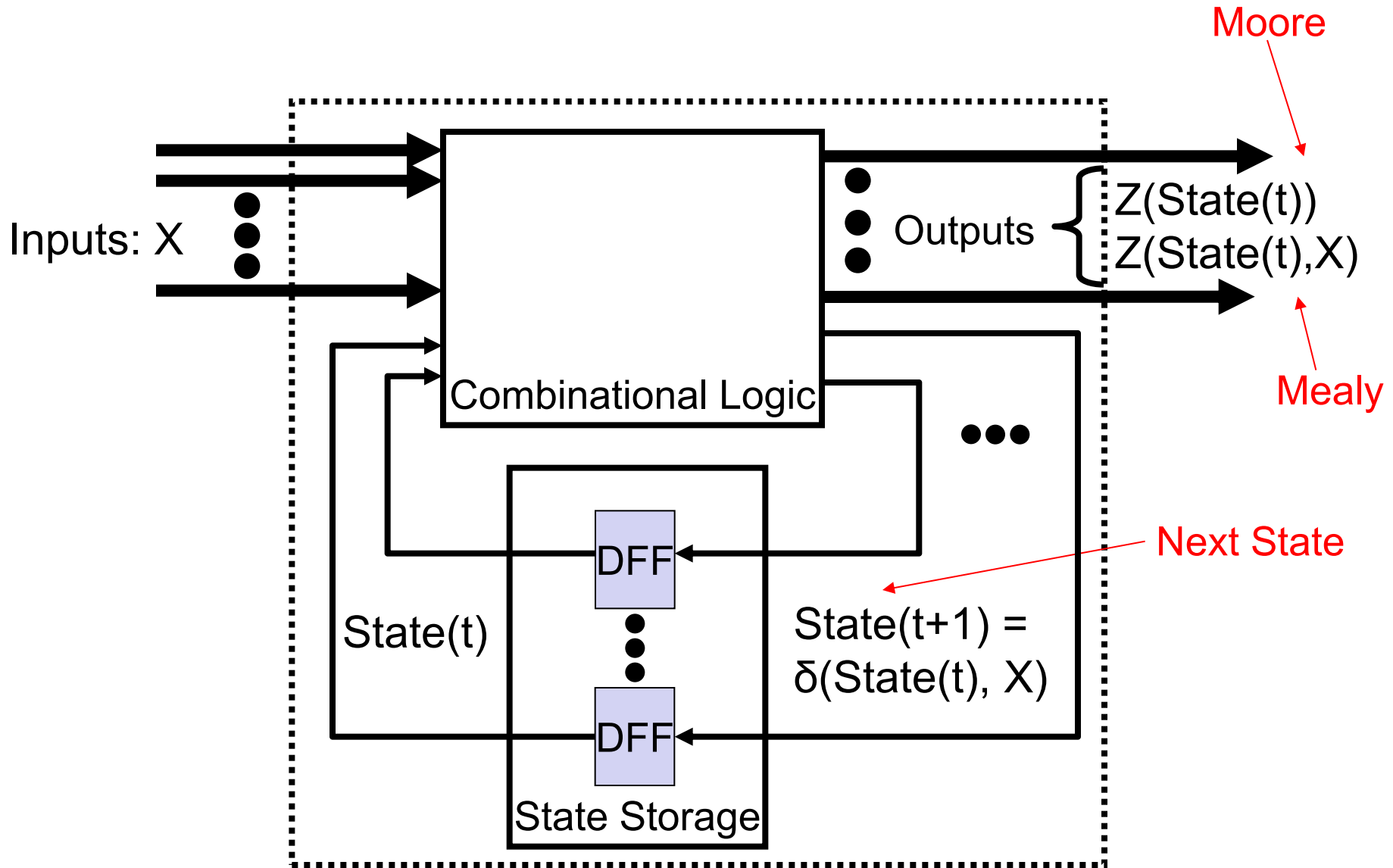
```
END;
```

Array Example (Delay Shift Register)

```
flag_reg(flag_reg'high downto 0) <= flag_reg(flag_reg'high-1  
downto 0) & flag_in;
```



FSM: General Circuit Architecture



VHDL for Mealy ("1011") Example

-- Store the "state"

```
Update_State: process(clk)
```

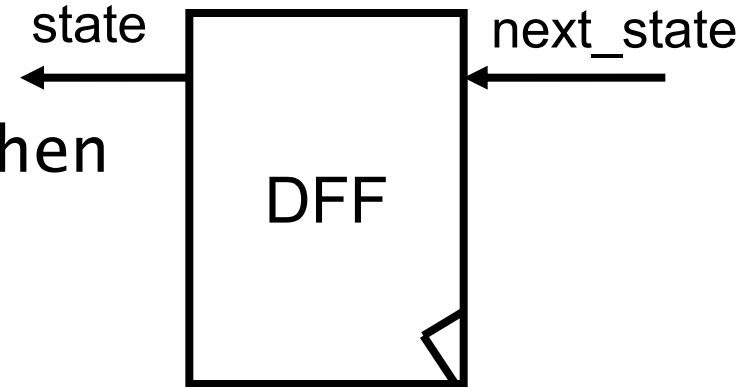
```
begin
```

```
  if(clk'event and clk='1') then
```

```
    state <= next_state;
```

```
  end if;
```

```
end process Update_State;
```



VHDL for Mealy ("1011") Example

-- Compute combinational logic

Combinational: process(x, state)

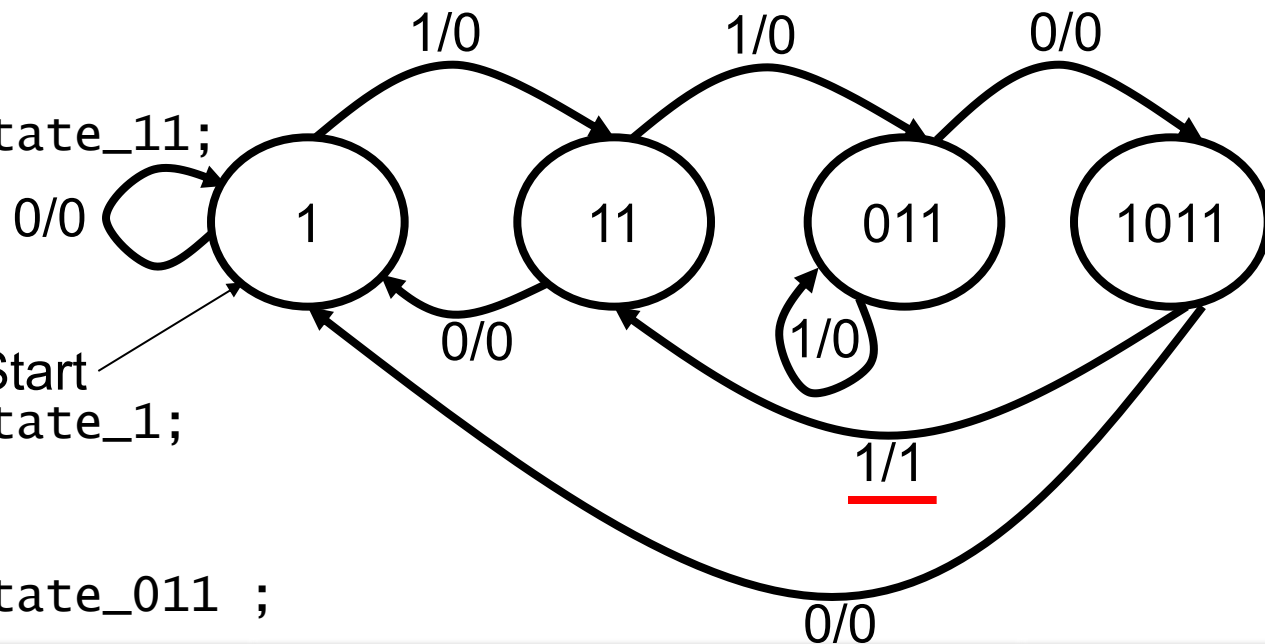
Begin

Defaults

Compute output

Compute next_state

```
z <= 0;
next_state <= state
case state is
when state_1 =>
  if(x = '0') then
    z <= '0';
    next_state <= state_1;
  else
    z <= '0';
    next_state <= state_11;
  end if;
when state_11 =>
  if(x = '0') then
    z <= '0';
    next_state <= state_1;
  else
    z <= '0';
    next_state <= state_011 ;
  end if;
```



VHDL for Mealy ("1011") Example

```
when state_011 =>  
  if(x = '0') then  
    z <= '0';  
    next_state <= state_1011;  
  else  
    z <= '0';  
    next_state <= state_011;  
  end if;
```

```
when state_1011 =>  
  if(x = '0') then  
    z <= '0';  
    next_state <= state_1;  
  else  
    z <= '1';  
    next_state <= state_11;  
  end if;  
when others =>
```

```
end case;  
end process Combinational;
```

