UNITED STATES
AIR FORCE
ACADEMY

ECE 383 - Embedded Computer Systems II
Lecture 10 - Datapath and Control

Lab1: lessons learned
- testbenches
- HW → Labs

UNITED STATES AIR FORCE ACADEMY

Lesson Outline

- Generics
- Datapath and Control
- Design Process
  - If
  - For
  - While
  - Assignment
- Basic Building Blocks – BBBs

"Mini-C" ⟹ VHDL

Generics

UNITED STATES AIR FORCE ACADEMY

Generics – Entity Declaration

- **Entity Declaration:**

```
entity lec10 is
 generic (N: integer := 4);
  port(     clk: in  STD_LOGIC;
            reset : in  STD_LOGIC;
            crtl: in std_logic_vector(1 downto 0);
            D: in unsigned (N-1 downto 0);
            Q: out unsigned (N-1 downto 0));
end lec10;
```

What device is this?

- **Note:**
  - The variable N is available in the entity and architecture context. In this case, you will need it to define the width of vectors.
  - The value of N must be an integer, not a binary string. Just use positive integers for N.

UNITED STATES AIR FORCE ACADEMY

Generics – Instantiation

- **Note:**
  - The default value for N is 4. That means, if you do not use the generic map statement in the instantiation below, you will get a 4-bit counter.
  - The D and Q vectors use N-1 because the vector starts at 0.
- **Instantiation:**

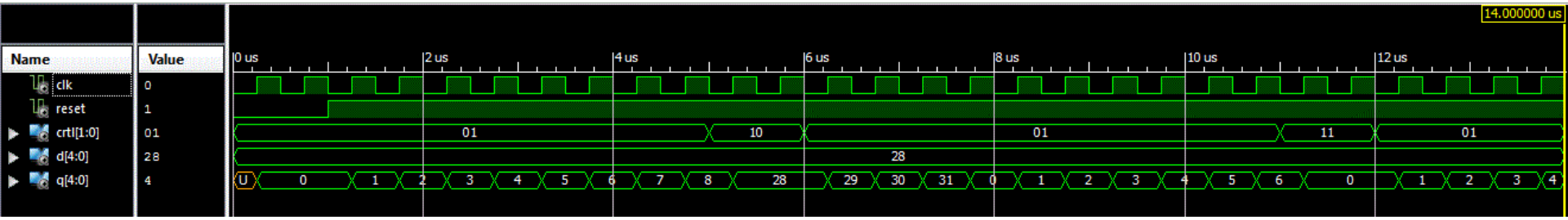
```
uut: lec10
     generic map(5)
     port map (  clk => clk,
                 reset => reset,
                 crtl => crtl,
                 D => D,
                 Q => Q);
```

→ see code

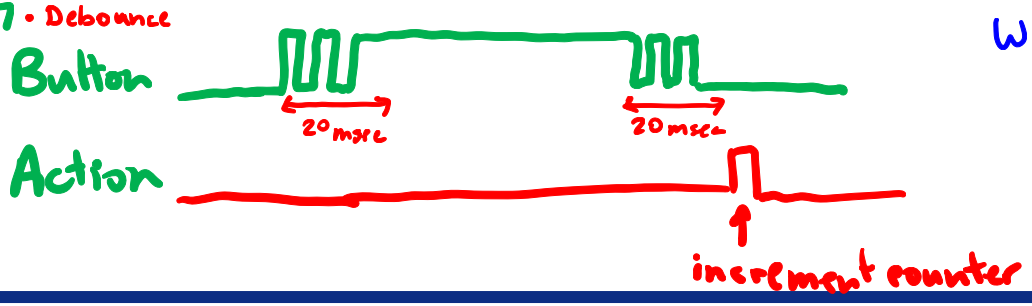
UNITED STATES AIR FORCE ACADEMY

Generics – 5-Bit Counter

- In this case, I made a 5-bit counter. The testbench linked on Lesson 10 runs the counter through all four control modes and even shows how it rolls over (around 7.5uS).



Control	Description
00	Hold
01	Count up
10	Load D
11	Synchronous Reset



World Record for presses per second? ____

↳ ____ msec

Can use this generic counter twice in HW07

1. N = 4
2. N = ?

Clock = 100MHz
so T = $\frac{1}{100MHz}$ = ____

How many counts for 20msec?

____

Simulation Time issue? See hw07_tb

TB has 100kHz option

Datapath and Control

BBBs

STATEMACHINE

sw = status word

cw = control word

Datapath and Control

- Datapath and Control Design Methodology
  - Datapath - responsible for data manipulations
  - Control - responsible for sequencing the actions of the datapath

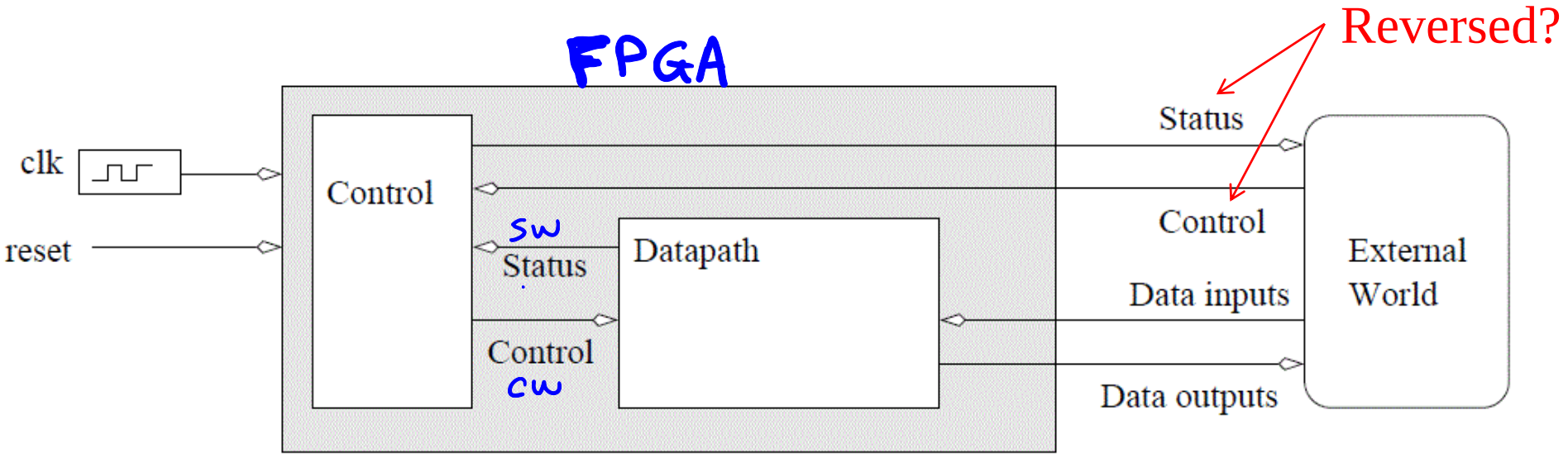


Fig 10.0 - An abstract digital system constructed from a datapath and a control unit.

Design Process

UNITED STATES AIR FORCE ACADEMY

Design Process

Step 0:

- Building digital systems using the datapath and control approach is a three-step process.
  1. Write an algorithmic description for the solution to the problem. "Mini-C"
  2. Parse the algorithmic description into datapath building blocks and control states.
  3. ~~Define the MIEs and OEs for the control unit.~~ Nope!

(a) State Transition Diagram

(b) State Output Table cw ↓ sw ↑

(c) Datapath (BBBs)

mini-C

- The programming language used to formalize an algorithmic solution to design problem is referred to as mini-C (a derivative of the C-programming language)
  - **IF**

    if (condition) then BODY_1 else BODY_2
  - **FOR**

    for (i=A; i<B; i += 1) BODY
  - **WHILE**

    while(condition) BODY
  - **ASSIGNMENT**

    X = value;

    X = (A + B) >> 4 ;

Design Process – If/Then/Else

```
if (x == y) {
    Body1;
} else {
    Body2;
}
```

How to make comparitor?

Process or CSA?

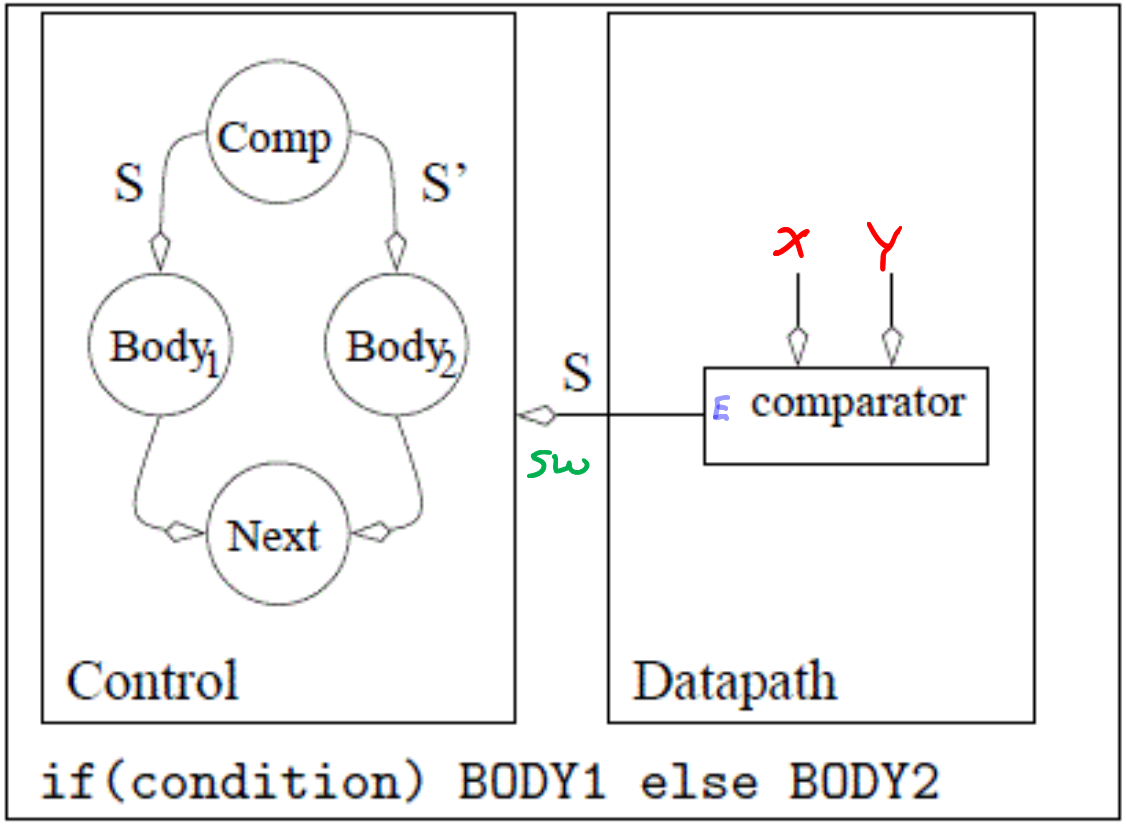


Fig 10.1 - The datapath and control components required to realize an if/then/else structure.

Design Process – For Loop

for (i=7; i<14; i++) {
Body;
}

Counter control? { cw, 00, 01, 10, 11 }

How to implement counter?

How to implement comparitor?

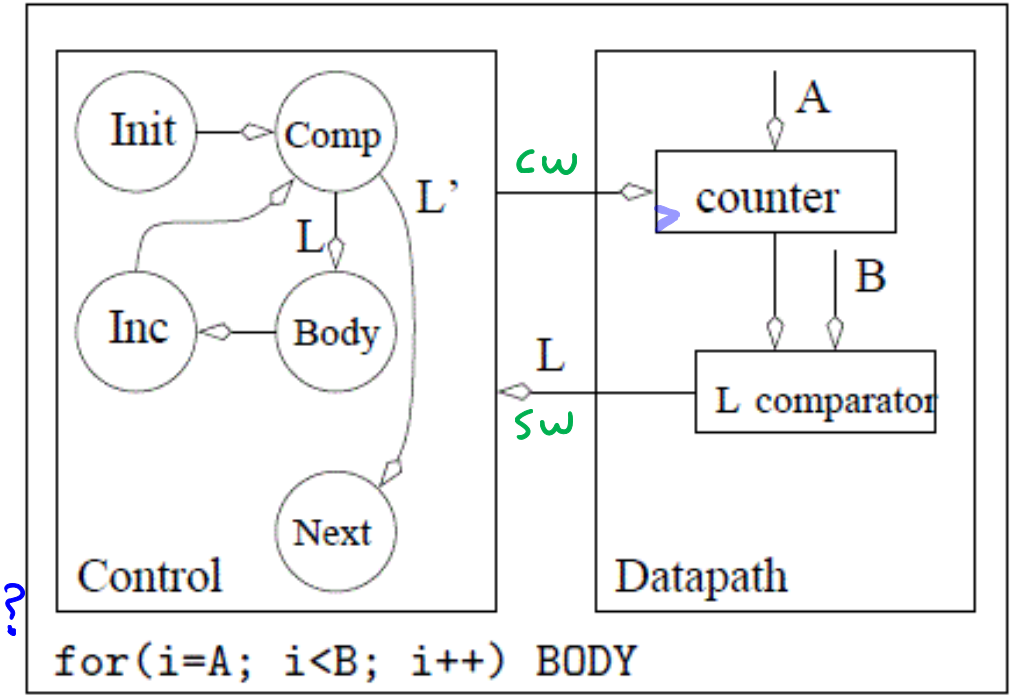


Fig 10.2 - The datapath and control components required to realize a for loop.

Design Process – While Loop

```
while (x == y) {
    Body;
}
```

How to implement comparitor?

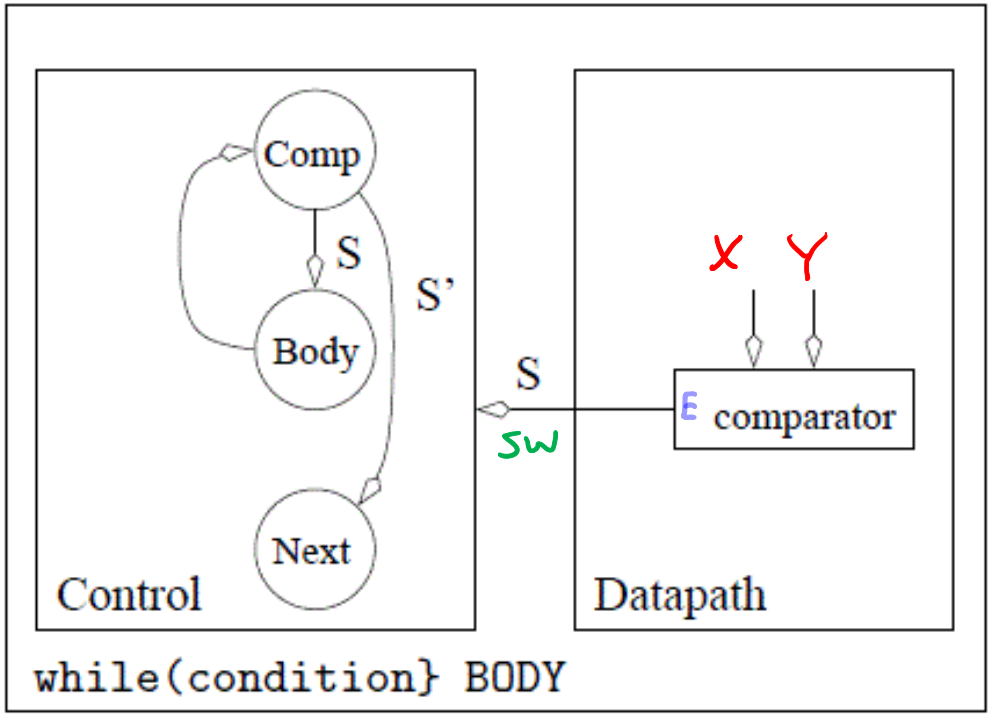


Fig 10.3 - The datapath and control components required to realize a while statement.

Design Process – Assignment

or Operation

X = X + Y;

How to implement?

Adder?

X Register?

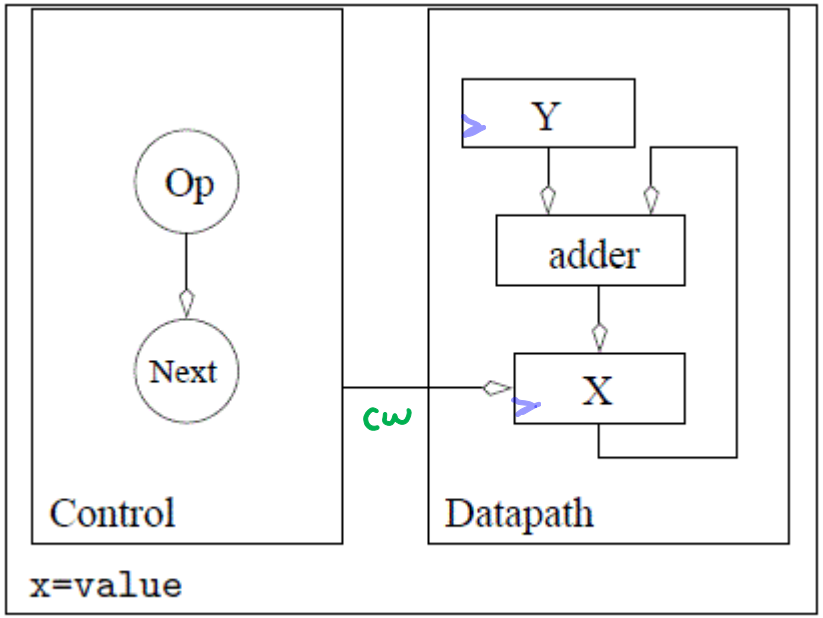


Fig 10.5 - The datapath and control components required to realize an assignment statement of the form X=X+Y.

Basic Building Blocks

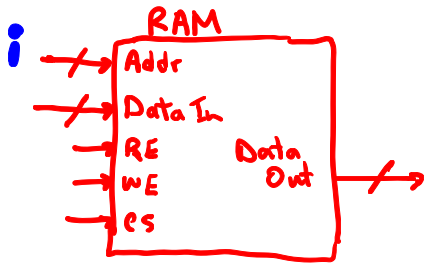
UNITED STATES AIR FORCE ACADEMY

Basic Building Blocks

Device	Data in	Data out	Status	Control
N:M Decoder	1 bit	M bits		N bits
N:1 Mux	N bits	1 bit		$\log_2(N)$ bits
M-bit N:1 Mux	N, each M-bits	M bits		$\log_2(N)$ bits
N-bit adder	2, each N-bits	N bits	Overflow	
N-bit add/sub	2, each N-bits	N bits	Overflow	1 bit
N-bit comparator	2, each N-bits		3 bits	
BCD to 7-segment	4 bits	7 bits		
N-bit priority encoder	N bits	$\log(N)$-bits		
N-bit register	N bits	N-bits		1 bit
N-bit shift register	N bits	N-bits		2 bits
N-bit counter	N bits	N bits		2 bits
Three state buffer	N bits	N bits		1 bit
N:M RAM	$\log_2(N)$ bits, M bits	M bit		2 bits
N-bit Bus transceiver	N bits	N bits		2 bit

Table 10.6 - The list of all the basic building blocks and some of their attributes.

UNITED STATES AIR FORCE ACADEMY



$2^n = 128$ $n =$ ____

Handout Problem

Step 0

Minimum Search Design a digital circuit that looks for the smallest 8-bit integer in a 128x8 bit RAM. The numbers are stored at addresses 0...99, you may assume that the RAM is preloaded with data.

Step 1

```
1.  min = 0xFF;                  // Set the min reg to largest value
2.  for (i=0; i<100; i++) {      // Search through the entire array
3.      MBR=RAM[i];              // read an 8-bit value from the RAM
4.      if (MBR<min) then        // If MBR is smaller than min
5.          min = MBR;           //   then set min to the smallest value
6.  } // end for
```

variables are _____

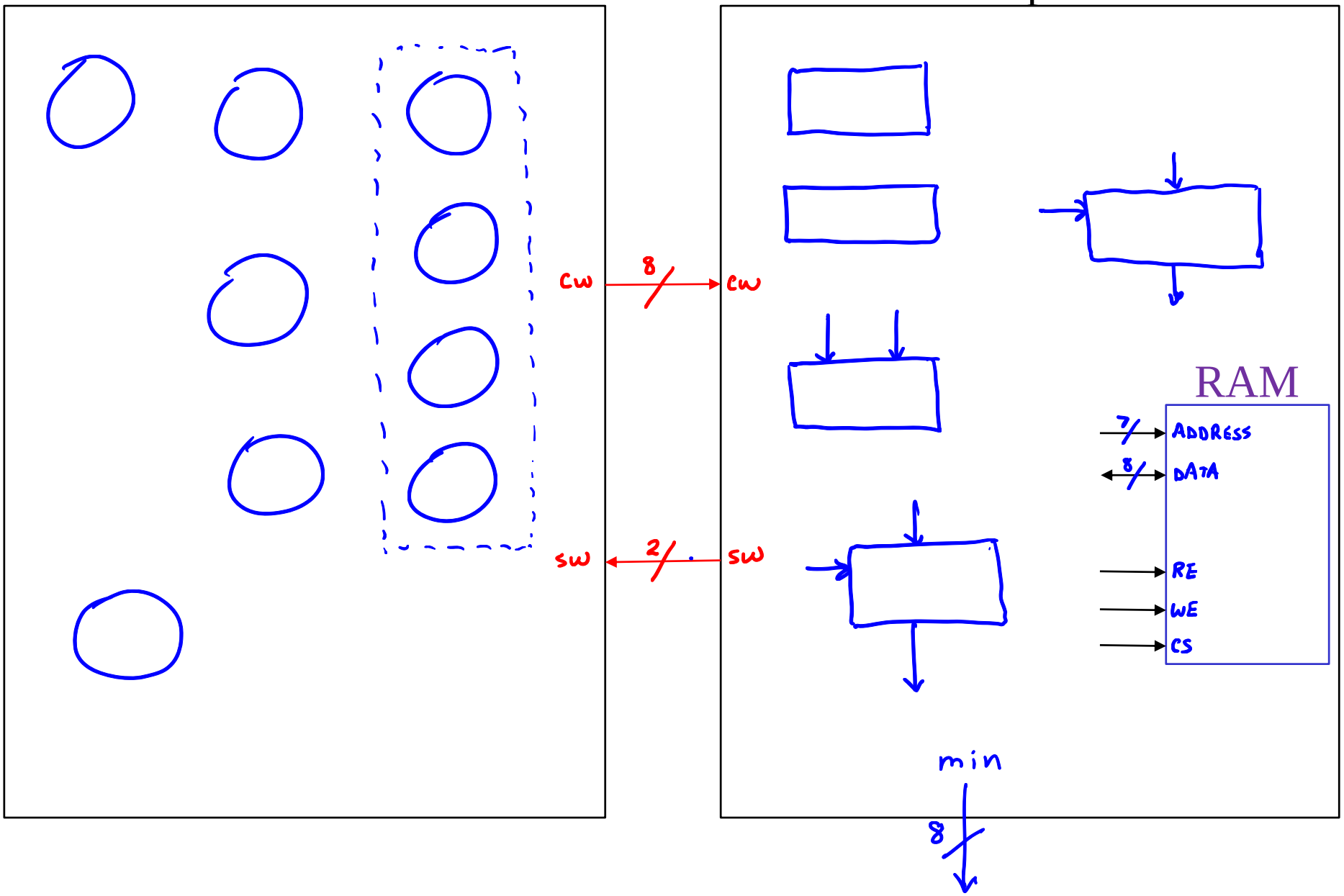
step 2

FSM – State machine

BBBs – Basic Building Blocks

Control

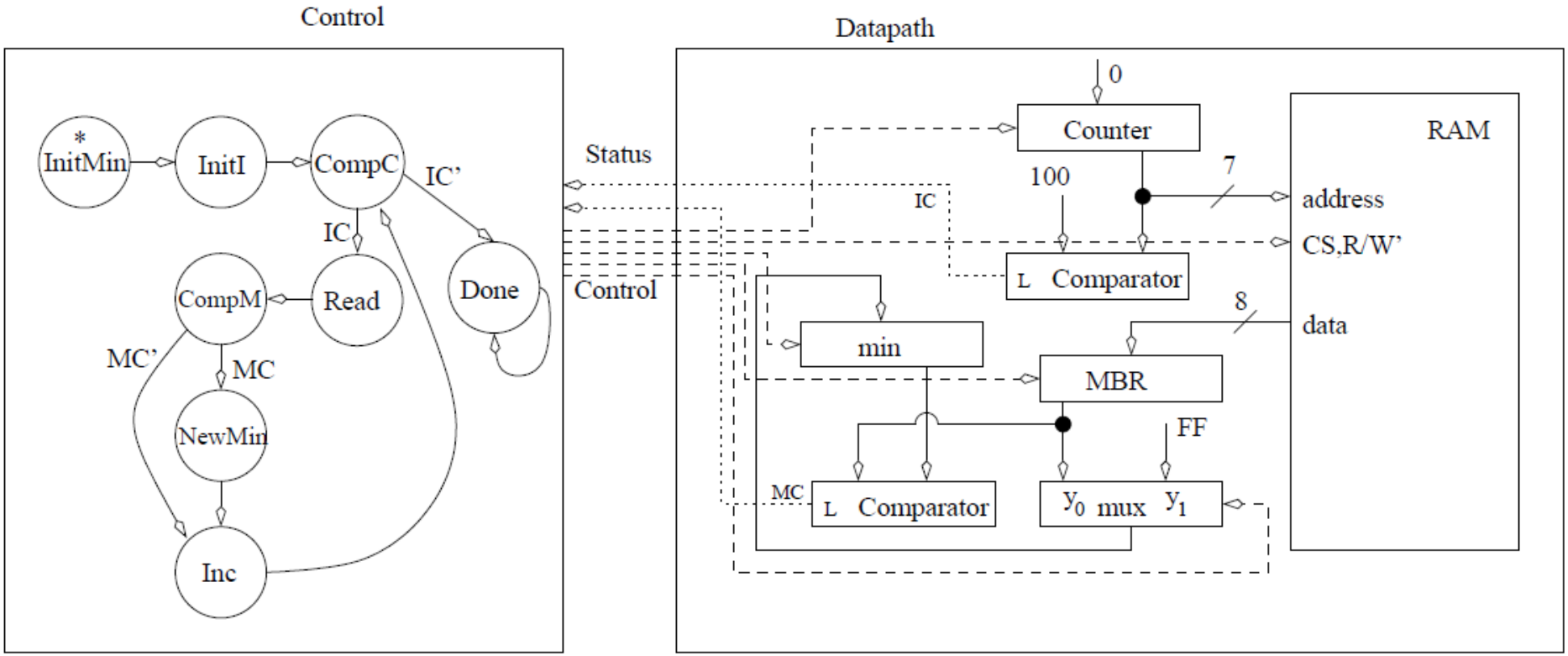
Datapath



CW output table

State	CS	RE	WE	Reg Min	Min mux	Counter	MBR
	0 off	0 idle	0 idle	0 hold	0 load FF	00 hold	0 hold
	1 active	1 read	1 write	1 load	1 load RAM	01 load	1 load
						10 count	
States						11 reset	
InitMin							
InitI							
CompC							
Read1							
CompM							
NewMin							
Inc							
Done							

READ2


Control
Datapath
*
InitMin
InitI
CompC
IC'
IC
CompM
Read
Done
MC'
MC
NewMin
Inc
Status
Control
0
Counter
RAM
100
7
IC
address
CS,R/W'
L Comparator
8
data
min
MBR
FF
MC
L Comparator
y0 mux y1

- **Do we need to now go and find the state transition equations and state output equations?**

```
1   ----------------------------------------------------------------------
2   -- Name:     Chris Coulston
3   -- Date:     Jan 28, 2015
4   -- File:     lec09.vhdl
5   -- Event:    Lecture 9
6   -- Crs:      ECE 383
7   ----------------------------------------------------------------------
8   library IEEE;
9   use IEEE.STD_LOGIC_1164.ALL;
10  use IEEE.NUMERIC_STD.ALL;
11
12  entity lec09 is
13        Port(  clk: in  STD_LOGIC;
14               reset : in  STD_LOGIC;
15               sw: in STD_LOGIC_VECTOR(2 downto 0);
16               cw: out STD_LOGIC_VECTOR(4 downto 0));
17  end lec09;
18
19  architecture behavior of lec09 is
20
21     type state_type is (WaitEnter, WaitRead, Set30, WaitLeave, Set3, Goose);
22     signal state: state_type;
23
24     constant rfid: integer := 2;        -- helps keep status bits straight
25     constant cow: integer := 1;
26     constant timer: integer := 0;
27
28  begin
29     state_process: process(clk,reset)
30        begin
31        if (rising_edge(clk)) then
32            if (reset = '0') then
33               state <= WaitEnter;
34            else
35               case state is
36                   when WaitEnter =>
37                      if (sw(cow) = '1') then state <= WaitRead; end if;
38                   when WaitRead =>
39                      if (sw(rfid) = '1') then state <= Set30; end if;
40                   when Set30 =>
41                      state <= WaitLeave;
42                   when WaitLeave =>
43                      if (sw(cow) = '0') then state <= WaitEnter;
44                      elsif (sw(timer) = '1' and sw(cow) = '1') then state <= Set3; end if;
45                   when Set3 =>
46                      state <= Goose;
47                   when Goose =>
48                      if (sw(timer) = '1') then state <= Set30; end if;
49               end case;
50            end if;
51        end if;
52     end process;
53
54     cw <=     "10000" when state = WaitEnter else
55               "00000" when state = WaitRead else
56               "01010" when state = Set30 else
57               "01110" when state = WaitLeave else
58               "01100" when state = Set3 else
59               "01111"; -- when state = Goose;
60
61  end behavior;
62
63
```

FSM Transition

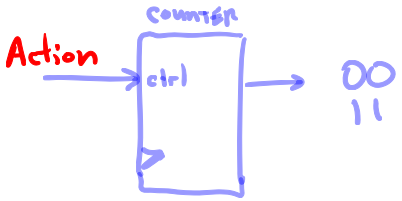
CW

DATA PATH

← NEW

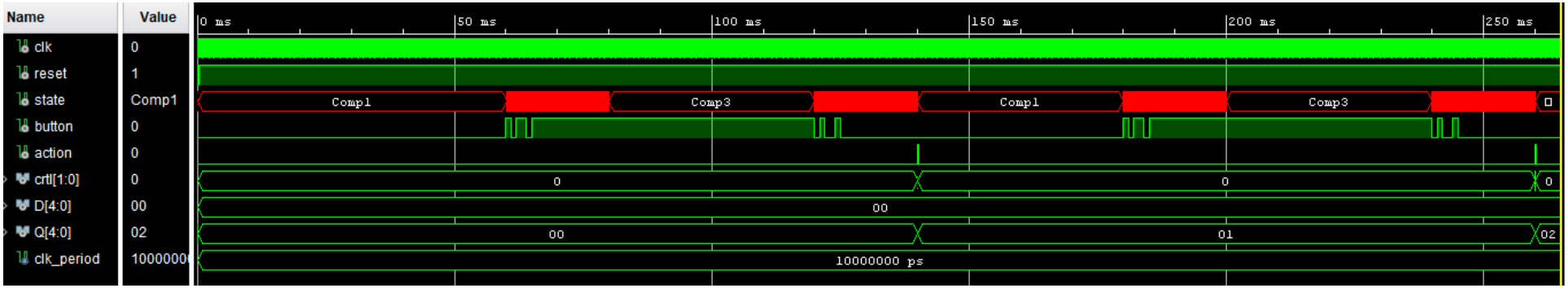
HW07

- Lesson 9 State Machine style?
- Lesson 10 Mini-C method?



Button Debouncer… statemachine with delays

```
action = 0;
while (button == 0);                    // wait for button press
for (i=0; i < 20_msec_count; i++);  // delay 20 msec for bouncing to stop
while (button == 1);                    // wait for button release
for (i=0; i < 20_msec_count; i++);  // delay 20 msec for bouncing to stop
action = 1;                          // do the action for the button event
action = 0;                         // undo the action for the button event
```


Name
Value
0 ms
50 ms
100 ms
150 ms
200 ms
250 ms
clk
0
reset
1
state
Comp1
Comp3
Comp1
Comp3
button
0
action
0
ctrl[1:0]
0
0
0
D[4:0]
00
00
Q[4:0]
02
00
01
02
clk_period
10000000 ps

How to do…

```
while (button == 1);
```

Easy Way

2 ways

(1)

```
action = 0;
action = 1;
action = 0;
```

(2)

Testbench… 100 MHz vs 100 KHz clock

HW06 solution?

Asynch Reset