COL215: Help session: HW assignment 2

Naman Jain

SIT, Indian Institute of Technology

Overview

1. Assignment overview
2. Image basics
3. ROM/RAM read/write
4. Image gradient
5. VGA controller
6. Common Vivado errors

Assignment overview

Assignment overview

Read only memory

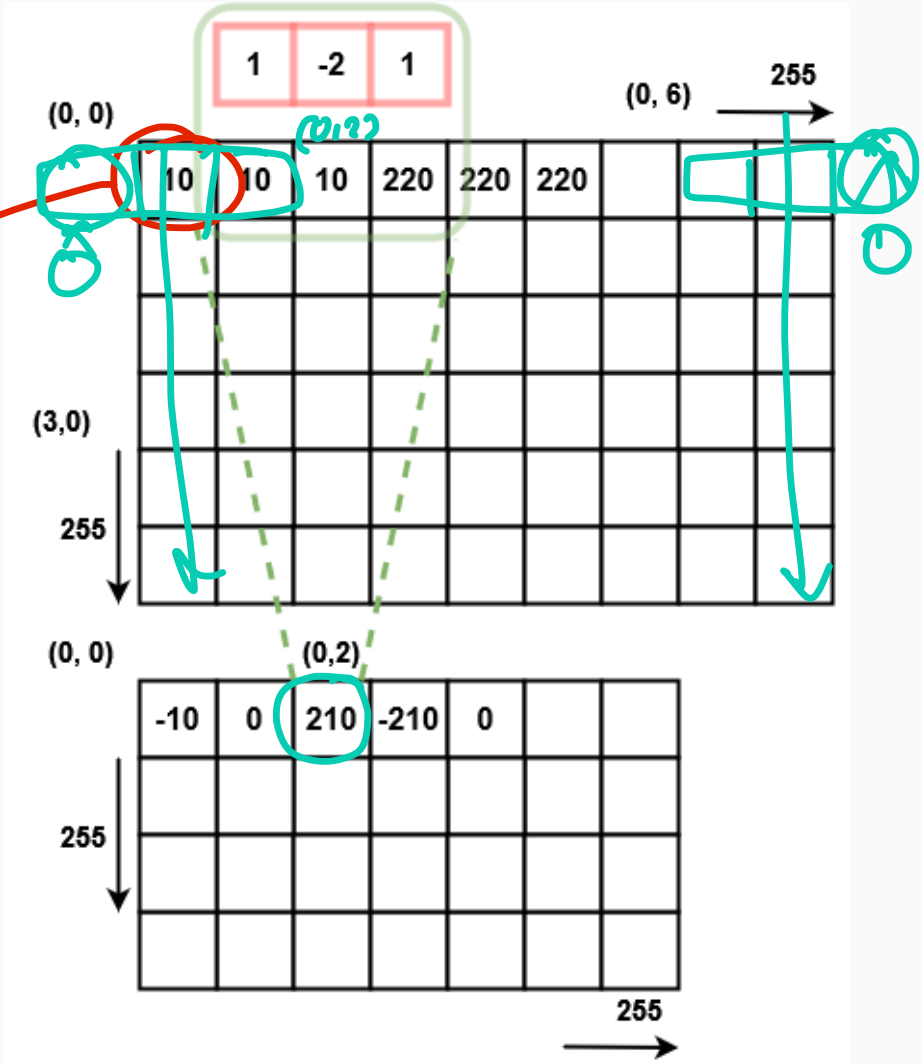
I/p image

- Module 1: Reading and Writing from ROM/RAM → RWM → read write memory
- Module 2: Calculating the image gradient pixelwise
- Module 3: Displaying the Output Image

O/p image

Image basics

Image basics



0 - 255
(8-bit unsigned int)

2D array/mat
256x256
↓
grayscale image

i, j-1 →
i, j →
i, j+1 →

256x256
2D array

< 0 255 >
0

256 258 → 255

ROM/RAM read/write

ROM/RAM basics

- Storage elements in the FPGA
- Contagious form of storage, read and write occur synchronously. At a time, you can perform one read or write operation.

LUT / BlockRAM
Distributed / Block

Table 1: RAM/ROM in table form

Index (decimal address)	Value (8 bit decimal data)
0	255
1	80
10	16
100	4
.	.
.	.
16383	0

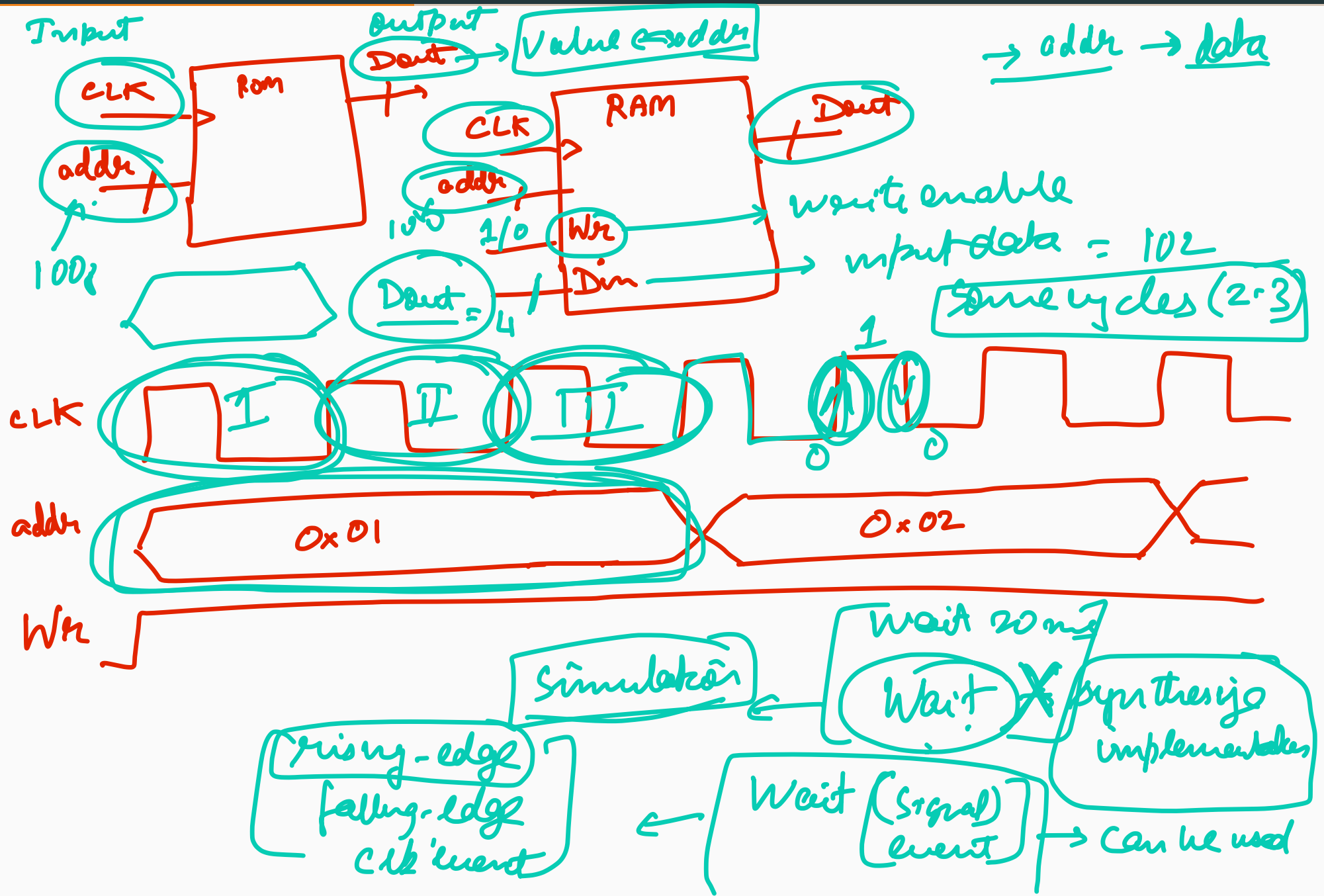
Location address

Value RAM/ROM

0-255

16384 memory

ROM/RAM basics



ROM/RAM basics

```
-- Stimulus process, copying content from ROM to RAM
-- enabling write mode for RAM
wr <= "1";
stim_proc: process (clk)
begin
  if (i < 16384) then
    if(rising_edge(clk)) then
      if (wr_comp = 0) then
        address <= std_logic_vector(to_unsigned(i, 14));
        --increment the address
        i <= i + 1;
        wr_comp <= wr_comp + 1;
      elsif (wr_comp = 1) then
        wr_comp <= wr_comp + 1;
      elsif (wr_comp = 2) then
        wr_comp <= 0;
      end if;
    end if;
  end if;
end process;
```

port / portmap

sensitivity list

ROM → RAM

Dout

Din

16384

integer

wr_comp = 0 or 16383

integer

2 → 10

4 → 100

5 → 101

0 1 2

3 clock delay

clk = 100MHz

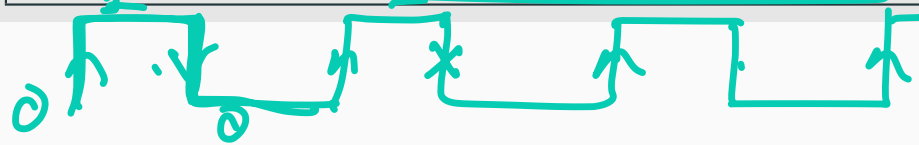


Image gradient

Image gradient operation

1	-2	1

i, j-1	i, j	i, j+1

Accumulate three pixel values to get the output pixel

⇒ Three read operations on ROM

p1

p2

p3

tmp = 0

tmp += p1 x 1

tmp += p2 x (-2)

tmp += p3 x 1

Output = tmp

⟹ Read value from ROM

⇓

store / Add the value / signal

if three pixel done

Write value to RAM

Accumulator

Image gradient

Sample code for understanding

```
tmp := 0
-- Stimulus process, to accumulate the contents
stim_proc: process (clk, enable)
begin
   if (enable = 1) then
        if(rising_edge(clk)) then
           tmp <= tmp + pixel_value;
        end if;
   else
        tmp <= std_logic_vector(0);
   end if;
end process;

Output_pixel <= tmp;
```

enable = '1' '0'

clk

1 → enable

0 → pixel value

Out put pixel

concurrent

pixel value

addr

ROM

Dout port value

port map

VGA controller

VGA

Refer to assignment handout

Common Vivado errors

Error overview

- Error 1: Poor placement for routing between an IO pin and BUFG
- Error 2: Rule violation (MDRV-1) Multiple Driver Nets
- Error 3: Rule violation (LUTLP-1) Combinatorial Loop

Error: Poor placement

```
entity poor_placement is
  Port (
    clk: IN std_logic;
    sw: IN std_logic_vector(13 downto 0);
    led: OUT std_logic_vector(1 downto 0)
  );
end poor_placement;

architecture Behavioral of poor_placement is
signal i: integer := 0;
begin
    led <= std_logic_vector(to_unsigned(i, 2));
    sample: process (clk, sw(0))
    begin
        if(rising_edge(sw(0))) then
            i <= i + 1;
        end if;
    --    if(rising_edge(clk)) then
    --        if (sw(0) = '1') then
    --            i <= i + 1;
    --        end if;
    --    end if;
    end process;
end Behavioral;
```

sync

sw(0) input is async

Error: Poor placement

Implementation (3 errors)

Place Design (3 errors)

[Place 30-574] Poor placement for routing between an IO pin and BUFG. If this sub optimal condition is acceptable for this design, you may use the CLOCK_DEDICATED_ROUTE constraint in the .xdc file to demote this message to a WARNING. However, the use of this override is highly discouraged. These examples can be used directly in the .xdc file to override this clock rule.
< set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets sw_IBUF[0]] >

sw_IBUF[0]_inst (IBUF.O) is locked to IOB_X0Y11
and sw_IBUF_BUFG[0]_inst (BUFG.I) is provisionally placed by clockplacer on BUFGCTRL_X0Y0

Error: Multiple Driver nets

```
architecture Behavioral of poor_placement is
signal i: integer := 0;
begin
    led <= std_logic_vector(to_unsigned(i, 2));
    sample: process (clk, sw(0))
    begin
        if(rising_edge(clk)) then
            if (sw(0) = '1') then
                i <= i + 1;
            end if;
        end if;
    end process;
    sample2: process (clk)
    begin
        if(rising_edge(clk)) then
            i <= i + 2;
        end if;
    end process;
end Behavioral;
```

Error: Multiple driver net

Net (1 error)

[DRC MDRV-1] Multiple Driver Nets: Net led_OBUF[1] has multiple drivers: i_reg[1]__0/Q, and i_reg[1]/Q.

[Vivado_Tcl 4-78] Error(s) found during DRC. Opt_design not run.

i ~~reg~~

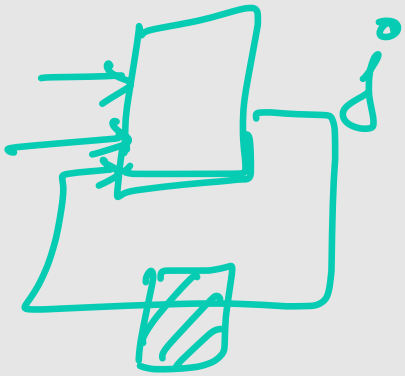
port

signal

Error: Combinatorial Loop

```
architecture Behavioral of poor_placement is
signal i: integer := 0;
signal j, k: std_logic := '0';
begin
    j <= sw(0) and sw(1) and j;
    led(0) <= j;
    --j <= sw(0) and sw(1);

    --sample3: process(clk)
    --begin
    --    if (rising_edge(clk)) then
    --        k <= j;
    --    end if;
    --end process;
    --led(0) <= j and k
end Behavioral;
```



Error: Combinatorial Loop

Write Bitstream (2 errors)
- DRC (1 error)
  - Netlist (1 error)
    - Design Level (1 error)
      - Combinatorial Loop (1 error)
        - [DRC LUTLP-1] Combinatorial Loop Alert: 1 LUT cells form a combinatorial loop. This can create a race condition. Timing analysis may not be accurate. The preferred resolution is to modify the design to remove combinatorial logic loops. If the loop is known and understood, this DRC can be bypassed by acknowledging the condition and setting the following XDC constraint on any one of the nets in the loop: 'set_property ALLOW_COMBINATORIAL_LOOPS TRUE [get_nets <myHier/myNet>]'. One net in the loop is led_OBUF[0]. Please evaluate your design. The cells in the loop are: led_OBUF[0]_inst_i_1.
- [Vivado 12-1345] Error(s) found during DRC. Bitgen not run.