Verilog Quick Start

CS 513200 Deep Learning Hardware Accelerator Design

YLLab 2019/05/02

Outline

- Workstation environment
- IC Design Flow
- Survival Guide for Verilog HDL
- Example code (Image Processing Filter)

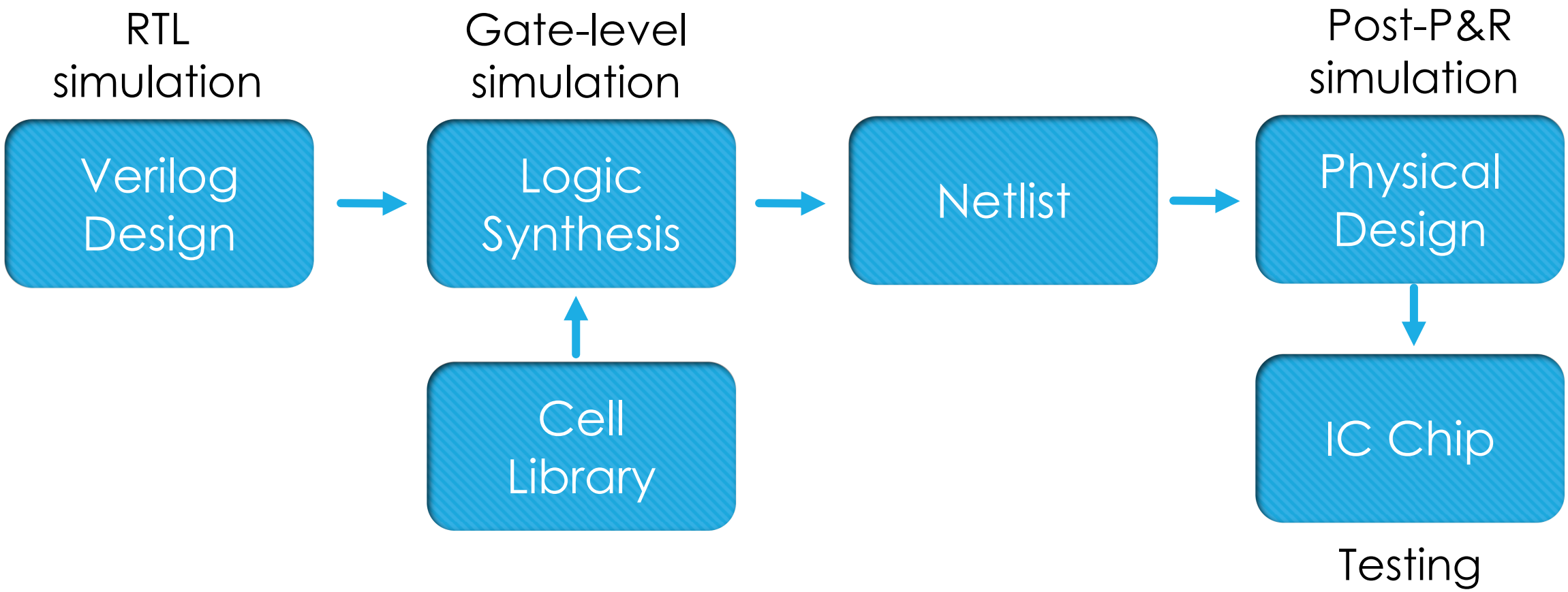
Workstation environment

SSH
Windows: MobaXterm
Mac: Terminal

ssh –X Account@nthucad.cs.nthu.edu.tw
ssh –X ic21

Tool
- NC-Verilog
- nWave
- Design compiler

IC Design Flow



Survival Guide for Verilog HDL

- Module Structure
  - Input / Output port
  - Data type
  - Value set
- Module Instantiation
- FAQ
- Online tutorial
  - Asic-world: http://www.asic-world.com/verilog/index.html

Module Structure

1. module / endmodule
2. Port Declaration

```
module Counter #(
    parameter DATA_Width = 4
)(
    input clk,
    input rst,
    input start,
    output finish,
    output reg [DATA_Width-1:0] out
);

endmodule
```

Data type

1. wire
2. reg

```
wire c;
assign c = a+b;
```

Combinational circuit

```
reg d;
always@(*)begin
    d = a+b;
end
```

Combinational circuit

Data type

1. wire
2. reg

```
always@(posedge clk or negedge rst)
begin
    if( ! rst )begin
        …
    end else begin
        …
    end
end
```

Sequential circuit

Data type

1. wire
2. reg

```
reg d;
initial begin
        d = a+b;
end
```

Testbench

Value set

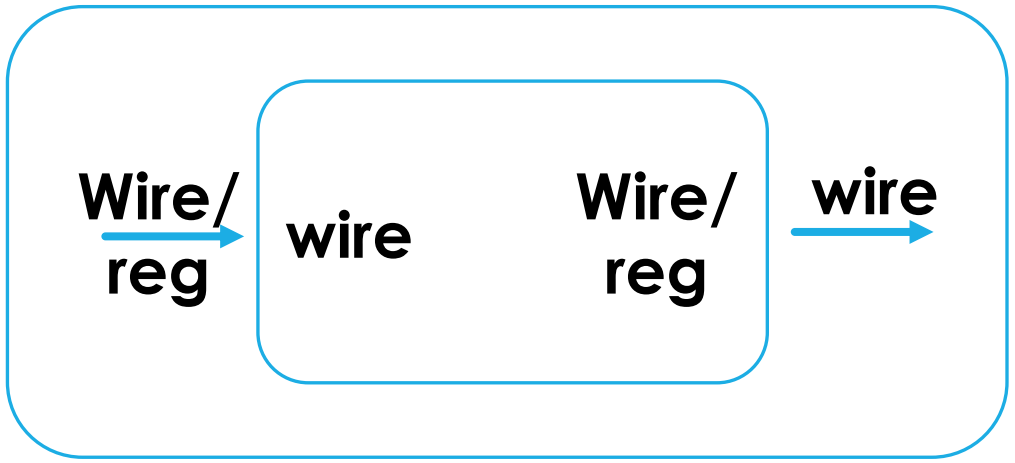
1. 1
2. 0
3. High Z
4. Unknown value

```
wire a, b, c;

assign a = 1'b1;
assign b = a + c;

/*
a = 1
b : Unknown value
c : High Z
*/
```

Module Instantiation




```
module Counter #(
    parameter DATA_Width = 4 )(
    input clk,
    input rst,
    input start,
    output finish,
    output reg [DATA_Width-1:0] out
);
…
endmodule
```

```
Counter #(.DATA_Width(4)) counter (
    .clk(clk),          // reg / wire
    .rst(rst),          // reg / wire
    .start(…),          // reg / wire
    .finish(…),         // wire
    .out(…)             // wire
);
```

FAQ

1. Non-blocking & Blocking

```
reg [2:0] a, b;
always@(posedge clk or …)begin
    if( !rst) begin
        a <= 3'd3;
        b <= 3'd5;
    end else begin
        a <= b;
        b <= a;
    end
end
```

Sequential circuit (Non-blocking)

FAQ

2. Combinational loop

```
reg [2:0] a, b;
always@(*)begin
        a = a + b;
end
```

Unknown value

FAQ

3. Multi-driven problem

```
reg [2:0] a;
always@(*)begin
      a = …;
end
always@(*)begin
      a = …;
end
```

Unknown value

Image Processing Filter (Convolution)

Kernel 0

0	0	0
0	0.5	0
0	0	-0.5

Kernel 1

-1/8	-1/8	-1/8
-1/8	1	-1/8
-1/8	-1/8	-1/8

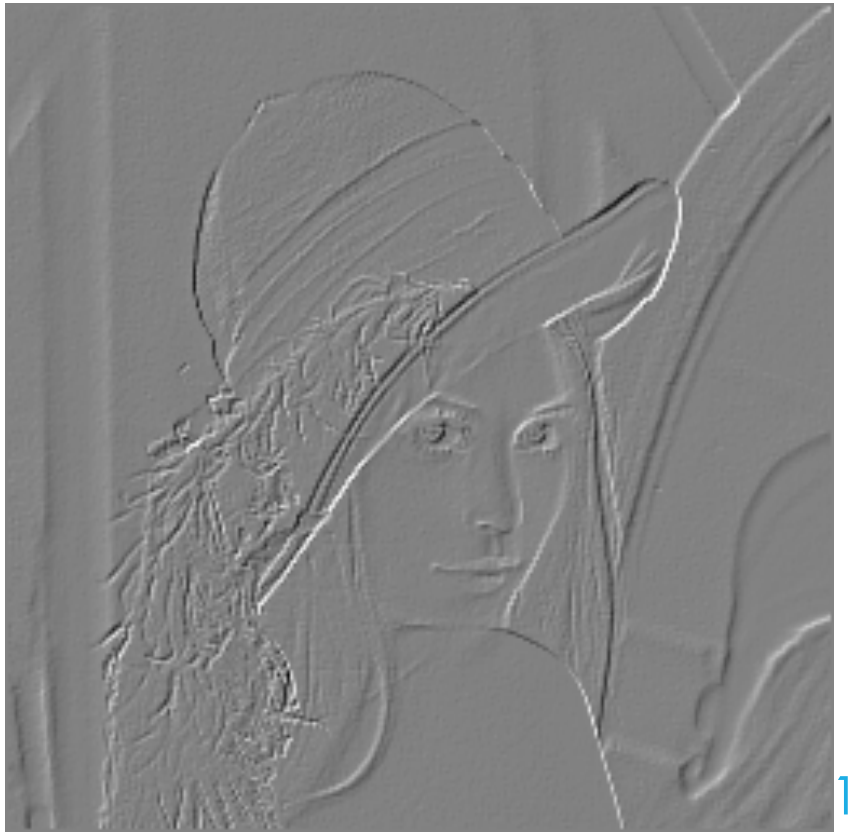
Kernel 2

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

Kernel – Shift & Subtract

0	0	0
0	0.5	0
0	0	-0.5

Stride = 1,
Pad = 0



Kernel – Edge detection

-1/8	-1/8	-1/8
-1/8	1	-1/8
-1/8	-1/8	-1/8

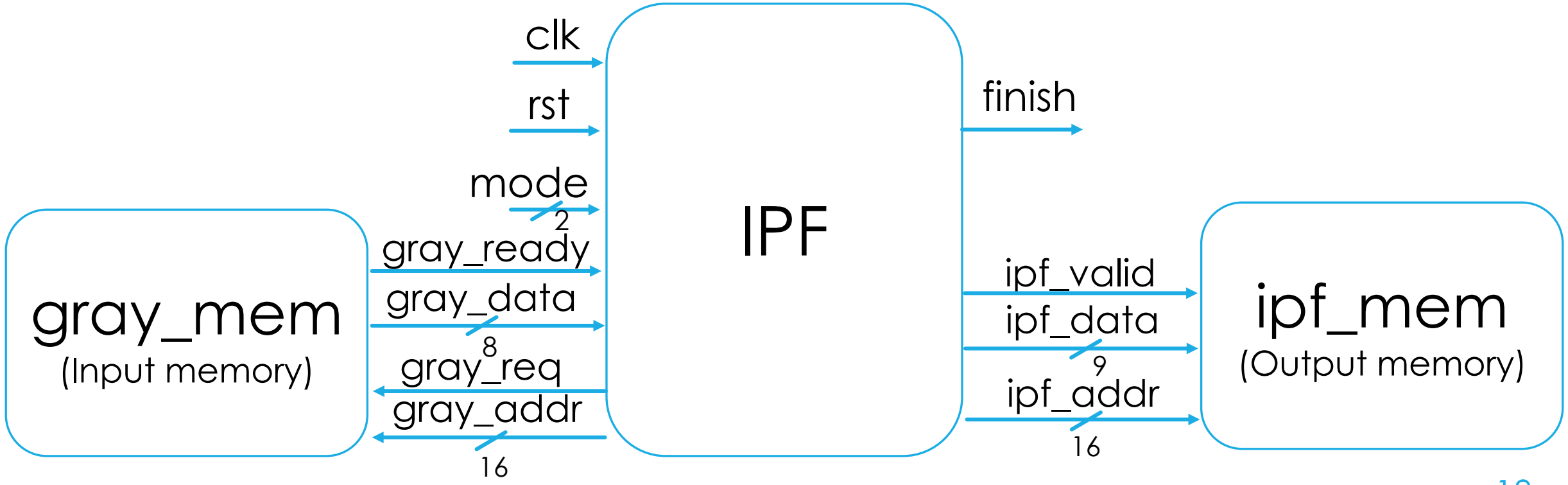
Stride = 1,
Pad = 0

Kernel – Gaussian Blur

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

Stride = 1,
Pad = 0

Interface


clk
rst
mode
2
gray_ready
gray_data
8
gray_req
gray_addr
16
IPF
finish
ipf_valid
ipf_data
9
ipf_addr
16
gray_mem
(Input memory)
ipf_mem
(Output memory)

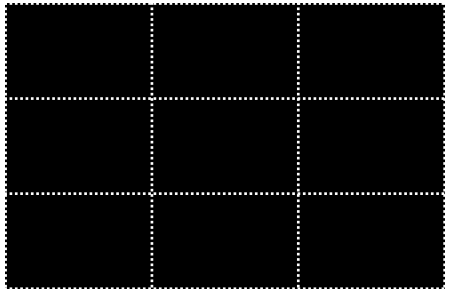
Data arrange in Gray (Input) Memory

256x256

gray_addr	gray_data
0	
1	
…	
256	
257	
…	
256*256-1	

Data arrange in IPF (Output) Memory

256x256



Kernel
3x3
Stride = 1,
Pad = 0

254x254

Data arrange in IPF (Output) Memory

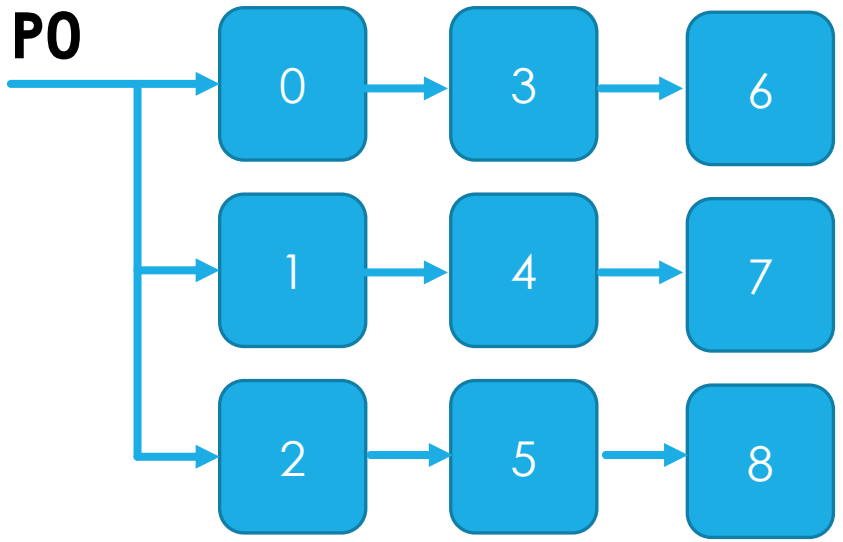
254x254

Initial Memory: zero value

gray_addr	gray_data
0	0
1	0
…	0
256	0
257	**Store**
…	…
256*256-1	0

Slide window - Cycle 0

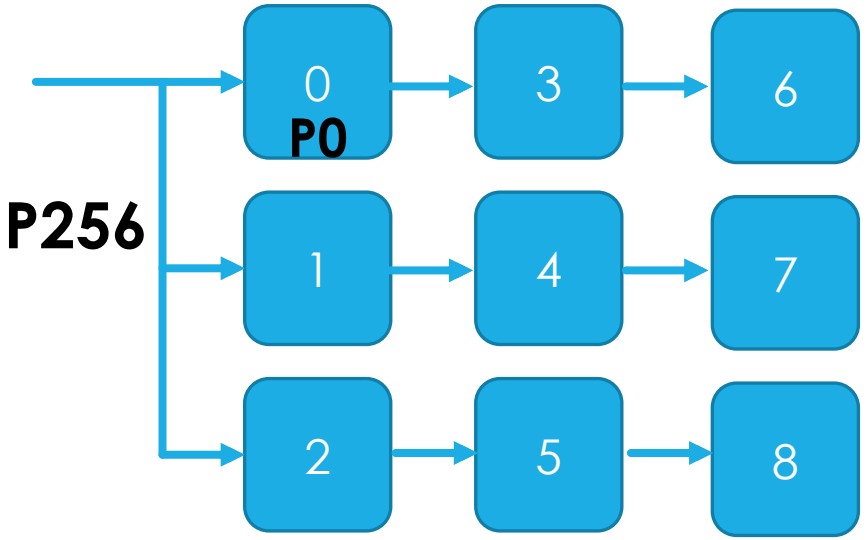
256x256



DFF

Slide window - Cycle 1

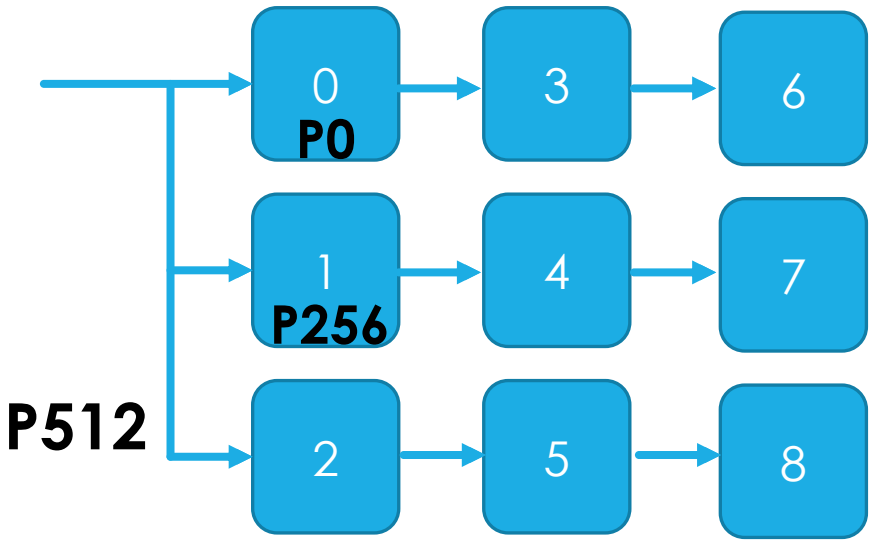
256x256



DFF

Slide window - Cycle 2

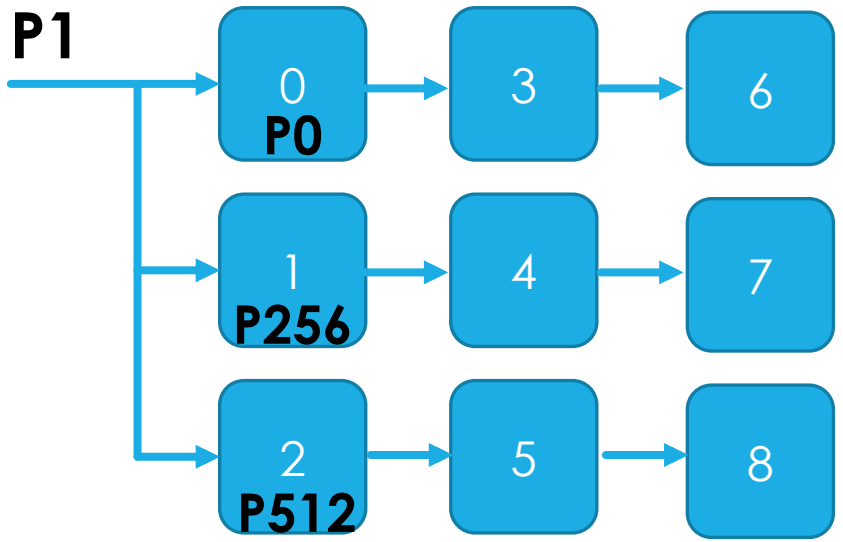
256x256



DFF

Slide window - Cycle 3

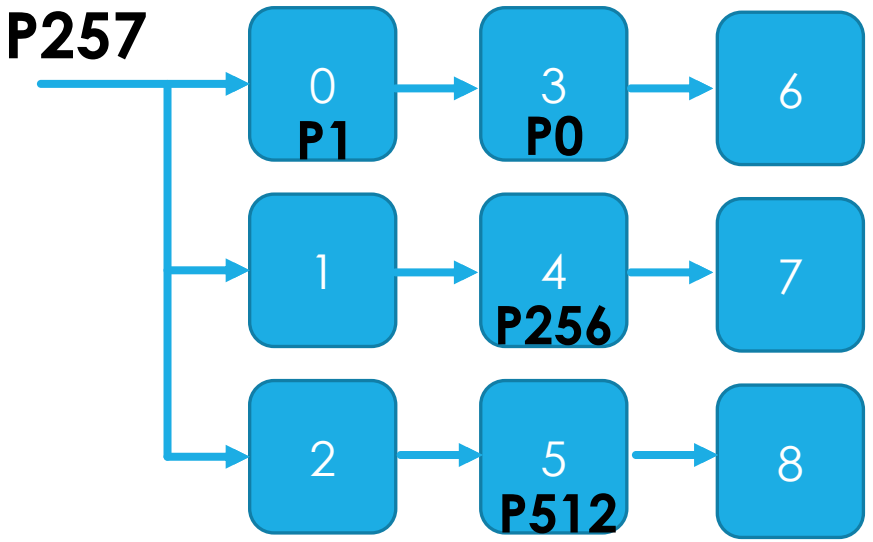
256x256



DFF

Slide window - Cycle 4

256x256

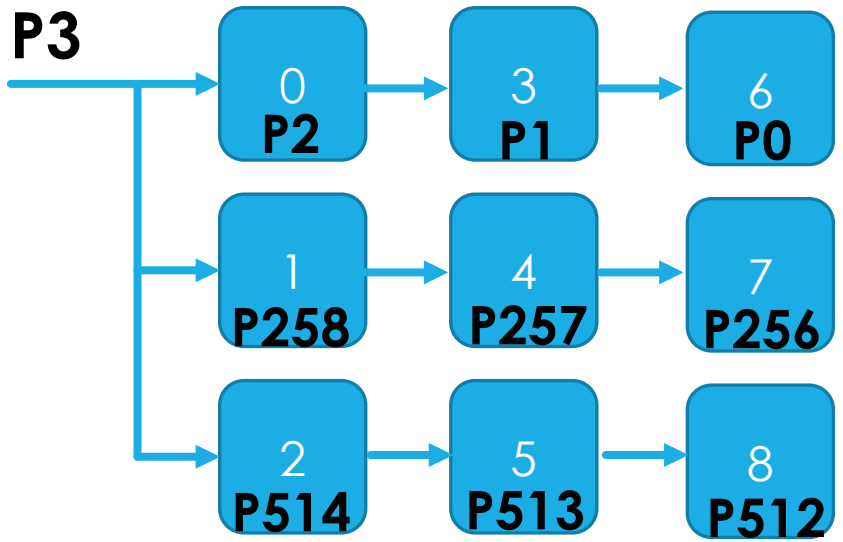


DFF

Slide window - Cycle 9

256x256

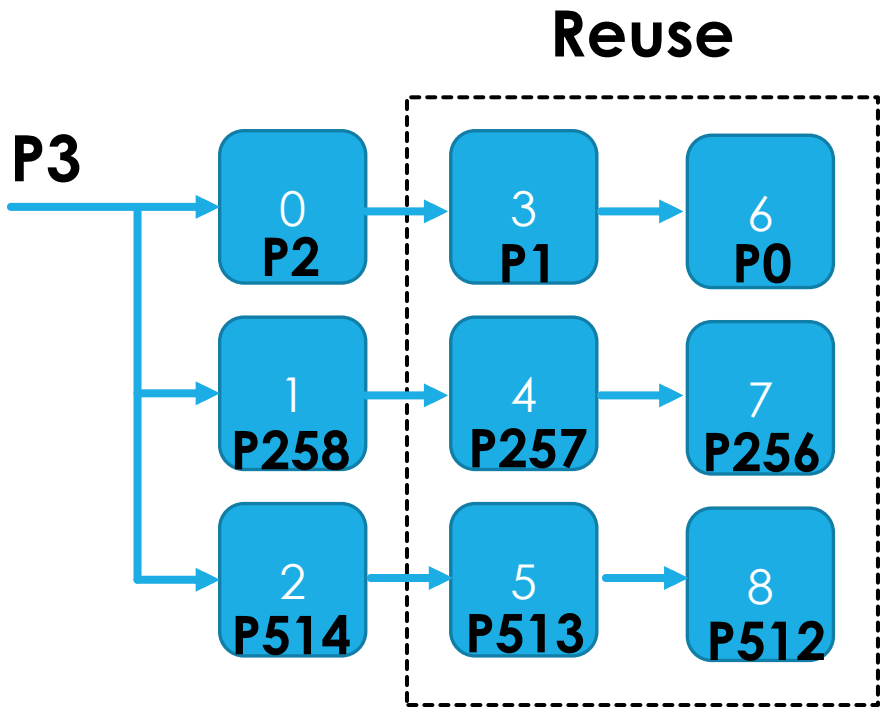
Computing



DFF

Slide window - Cycle 9

256x256



DFF

Review code

- makefile
- IPF.v
- IPF_tb.v
- IPF.py
- Testdata
  - Input: pattern.dat
  - Output: Golden0.dat, Golden1.dat, Golden2.dat
- synopsys_dc.setup
- IPF.tcl

Execution Flow

- Makefile command (For Kernel 0~2)
  - [RTL simulation] make, make sim1, make sim2
  - [Gate-level simulation] make syn, make syn1, make syn2
- Check Waveform (https://hackmd.io/s/rJB_1B7tx)
  - nWave
- Run Logic Synthesis (https://hackmd.io/s/BJfK9MKcl)
  - mv synopsys_dc.setup .synopsys_dc.setup
  - dc_shell
  - source IPF.tcl

Q&A