NCTU-EE IC LAB – Spring2022

Lab10 Exercise

Design: Image Display Control (IDC)

➢ Data Preparation

1. Extract test data from TA's directory:

   % tar -xvf ~iclabta01/Lab10.tar

2. The extracted LAB directory contains:
   a. EXERCISE/
   b. EXERCISE_wocg/
   c. PRACTICE/
   d. JG/

3700 × 3.632 × 107605 = 1446039032

3263 × 3.77 × 104362 = 1283810187

➢ Design Description

In this lab, you are asked to design an Image Display Control unit, which will replace a region by its Midpoint, Average, Rotation, Flip and the region is depending on the operation point. The image size will be 8*8 and the original operation point will be (3,3). These operations are defined in operation table. You should do this operation one-by-one depends on the op which is one of the input signals. After 15 times of operations, you should output the Zoom-in or Zoom-out graph depending on the position of the operation point. If the final operation point x < 4 and y < 4 do Zoom-in, else do Zoom-out.

➢ Operation Description

- Operation table

Operation	op	Operation description
Midpoint (round down)	0	Replace 2x2 by the midpoint of this region.
Average (round down)	1	Replace 2x2 by the average of this region.
Counterclockwise Rotation	2	Rotate 2x2 counterclockwise.
Clockwise Rotation	3	Rotate 2x2 clockwise.
Flip(*-1)	4	Change the sign of the 2x2 data.
Shift up y-1 x	5	Shift the operation point up.
Shift left y x-1	6	Shift the operation point left.
Shift down y+1 x	7	Shift the operation point down.
Shift right y x+1	8	Shift the operation point right.

12 13 14 4 3 2 1

23 24 34 3 4 1 2

1 3

- Operation example

Midpoint: Sort 2*2 region and find the midpoint.

midpoint = (5+6)/2 = 5 (round down, ex -3/2=-1, 11/2=5)

X=0,Y=0

X →

Y ↓

-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

-9 5 6 11

5

-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	5	5	8	9	-2
12	11	18	5	5	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

Average :

average = (-9+6+5+11)/4=3(round down)

-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	3	3	8	9	-2
12	11	18	3	3	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

Clockwise rotation :

-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

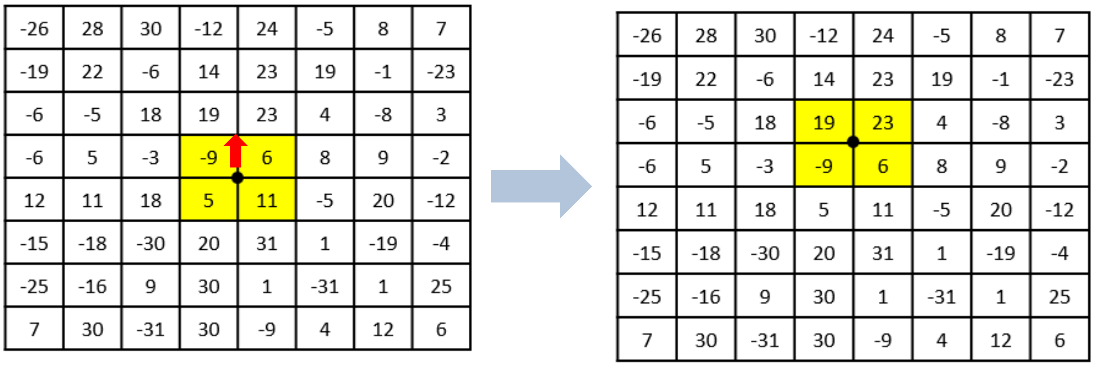
-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	5	-9	8	9	-2
12	11	18	11	6	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

Flip(*-1):

-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	9	-6	8	9	-2
12	11	18	-5	-11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

Operation point shift up : operation point form(3,3) to (3,2)



-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

- < Boundary Notice>

  When Y-coordinate is 0, Y-coordinate will remain 0 after shift up operation.

  When X-coordinate is 0, X-coordinate will remain 0 after shift left operation.

  When Y-coordinate is 6, Y-coordinate will remain 6 after shift down operation.

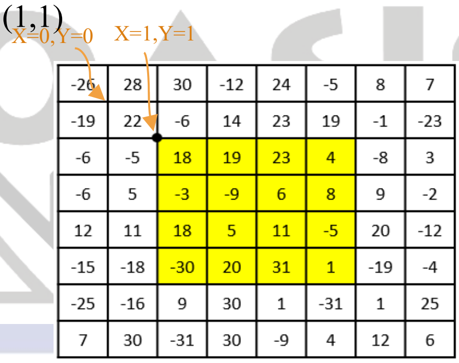
  When X-coordinate is 6, X-coordinate will remain 6 after shift right operation.

After doing 15 times of operations, you have to output the graph after Zoom-in or Zoom-out depending on the position of the operation point after all operations.

- Zoom example

  Zoom-in : Output would be 4*4 right-down region of operation point.

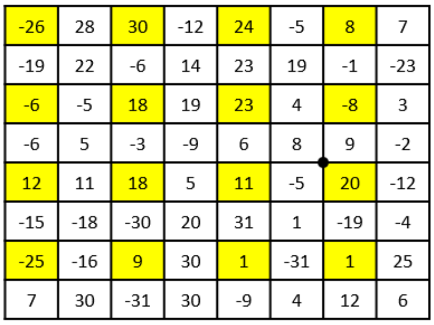
  operation point in (1,1)



-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

Zoom-out : If x≥4 or y≥4, output would be like the below graph.

operation point in (5,3) (Because 5 ≥ 4, you should do zoom-out.)

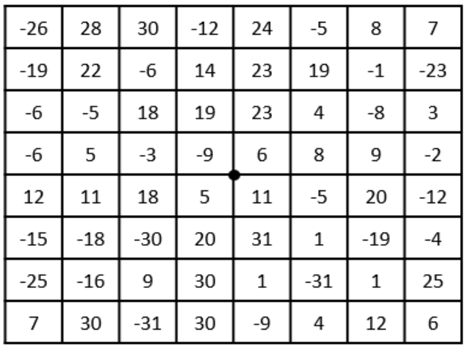


-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

➢ Example

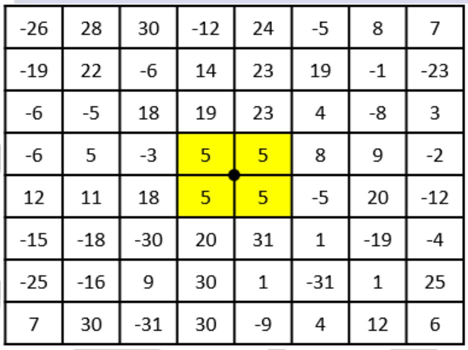
Example 1 : op = 0→5→8→8→4

Input image :



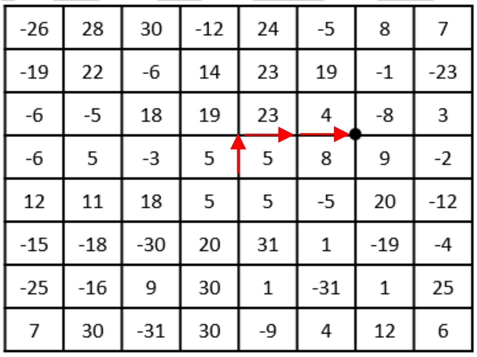
-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	-9	6	8	9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

0 (midpoint):



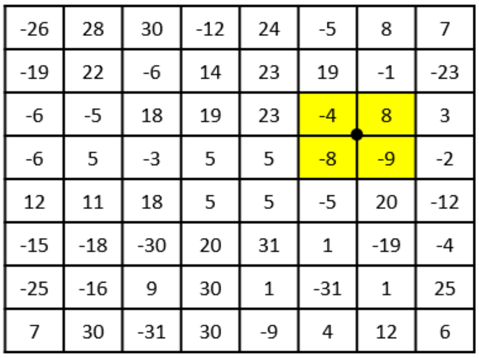
-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	5	5	8	9	-2
12	11	18	5	5	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

5(up) → 8(right) → 8(right):



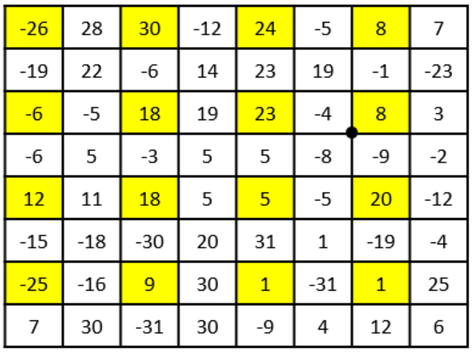
-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	4	-8	3
-6	5	-3	5	5	8	9	-2
12	11	18	5	5	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

4 (flip):



-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	-4	8	3
-6	5	-3	5	5	-8	-9	-2
12	11	18	5	5	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

Operation point at (5,2), x≥4 do Zoom-out



-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	18	19	23	-4	8	3
-6	5	-3	5	5	-8	-9	-2
12	11	18	5	5	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

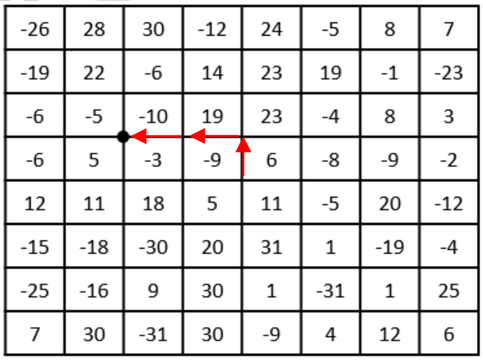
Output (left to right, up to down)

-26→30→24→8→-6 →…→1→1

-26	30	24	8
-6	18	23	8
12	18	5	20
-25	90	1	1

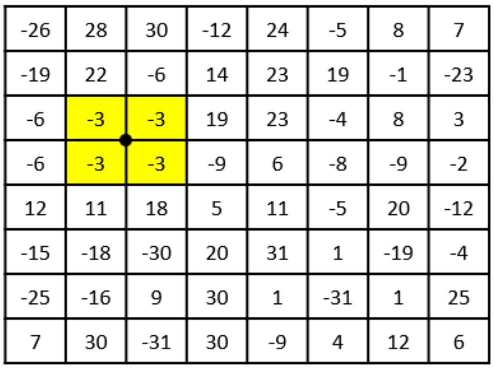
Example 2 : op = 5→6→6→1→3→6→6

5(up)→6(left)→6(left):



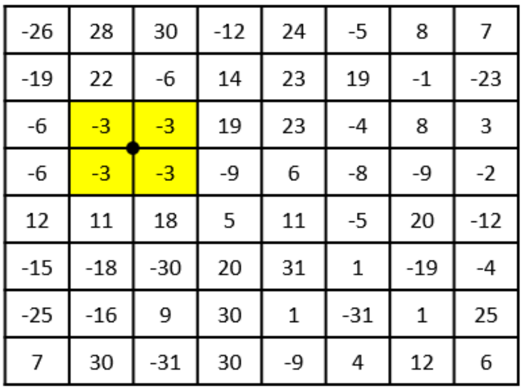
-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-5	-10	19	23	-4	8	3
-6	5	-3	-9	6	-8	-9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

1(average) : (-5+5-10-3)/4 = -3(round down)



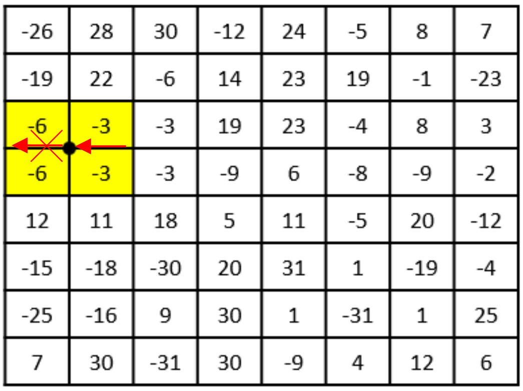
-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-3	-3	19	23	-4	8	3
-6	-3	-3	-9	6	-8	-9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

3 (clockwise rotation) :



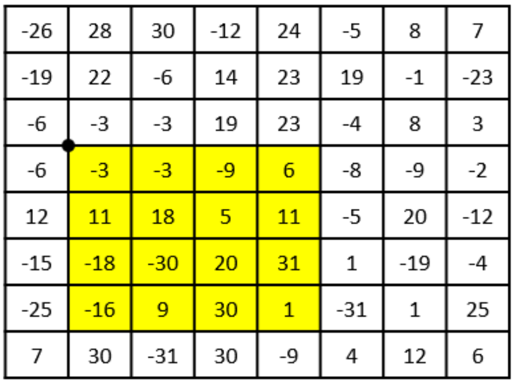
-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-3	-3	19	23	-4	8	3
-6	-3	-3	-9	6	-8	-9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

6(left)→6(left) : boundary!



-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-3	-3	19	23	-4	8	3
-6	-3	-3	-9	6	-8	-9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

Operation point at (0,2), x<4,y<4 do Zoom-in



-26	28	30	-12	24	-5	8	7
-19	22	-6	14	23	19	-1	-23
-6	-3	-3	19	23	-4	8	3
-6	-3	-3	-9	6	-8	-9	-2
12	11	18	5	11	-5	20	-12
-15	-18	-30	20	31	1	-19	-4
-25	-16	9	30	1	-31	1	25
7	30	-31	30	-9	4	12	6

Output (left to right, up to down)

-3→-3→-9→6→11 →…→30→1

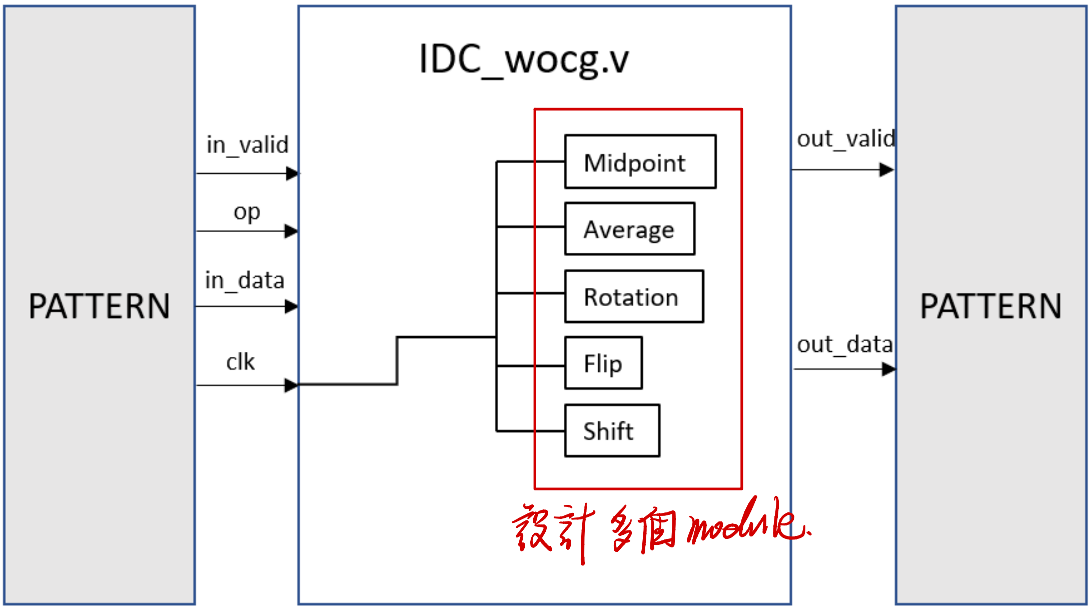
-3	**-3**	**-9**	**6**
11	**18**	**5**	**11**
-18	**-30**	**20**	**31**
-16	**9**	**30**	**1**

➢ Lab Hint

In this lab, you need to design two version of module.

- **Stage 1:**

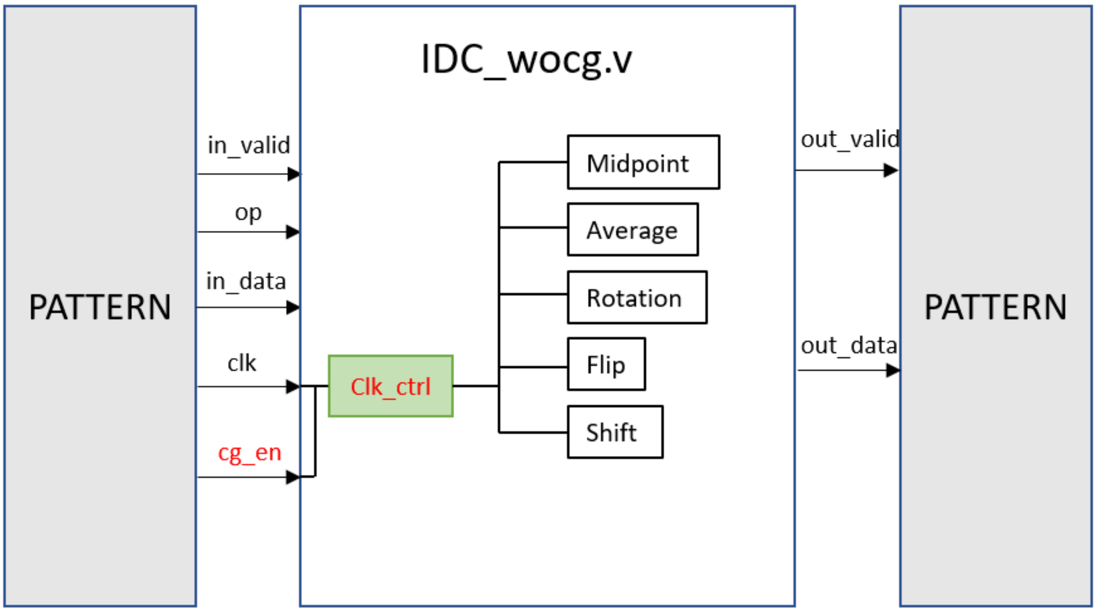
At EXERCISE_wocg/ **IDC_wocg.v**, which means IDC without clock gating, you should design a IDC module as below figure.



- **INPUT**：Receive **number series** & **op** from PATTERN.
  - ➔ The in_data [6:0] will be given for **64** cycles.
  - ➔ The op [3:0] will be given for **15** cycles.
- **IDC**：Design module with some submodules perform series processing described as above.
  - ➔ This module should be design without clock gating.
- **OUTPUT**：Output resulting data series.

- **Stage 2:**

At EXERCISE / **IDC.v**, you should design an IDC **module with clock gating cell** as below figure. Main different part is clock gating cell and cg_en input signal.



- **INPUT**：Receive **number series**, **mode and cg_en** from PATTERN
  - ➔ The in_data [6:0] will be given for **64** cycles.
  - ➔ The op [3:0] will be given for **15** cycles.
- **IDC**：Design module. You should add clock gating cell in the design module.

  If cg_en is high, processing blocks can perform clock gating; otherwise, if cg_en is low, processing blocks follow **clk.**
- **OUTPUT**：Output resulting data series

➢ Inputs

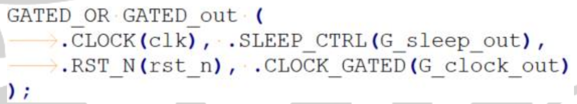
I/O	Signal name	Bit Width	Description
Input	**clk**	1	Clock
Input	**rst_n**	1	Asynchronous active low reset
Input	**cg_en**	1	If cg_en is high, the series processing blocks should execute clock gating. Otherwise, if cg_en is low, the image processing blocks follow **clk**.
Input	**in_valid**	1	High when input signals are valid.
Input	**in_data**	7	**in_data**(signed 2's complement) is valid when **in_valid** is high The in_data will be given in **64** cycles continuously in raster scan order.
Input	**op**	4	op will be given in first 15 in_valid cycles. Each op represents an operation defined in operation table

➢ Outputs

I/O	Signal name	Bit Width	Description
output	**out_valid**	1	Should set to high when your **out_data** is ready.
output	**out_data**	7	Output the resulting data series. out_data should be given in 16 cycles.

➢ Specifications

1. Top module name：IDC (File name: IDC.v)
2. Input pins ：**clk, rst_n, cg_en, in_valid, in_data[6:0], op [3:0]**
   Output pins：**out_valid, out_data[6:0]**
3. Use **asynchronous** reset active low architecture.
4. All your output register should be set zero after reset.
5. Changing clock period is prohibited **(fixed at 12ns)**.
6. The instance name of the clock gating cell (GATED_OR) should be GATED_XXX (e.g., GATED_cnt).

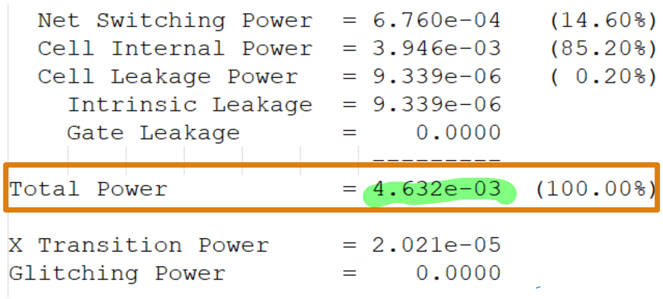


```
GATED_OR GATED_out (
    .CLOCK(clk), .SLEEP_CTRL(G_sleep_out),
    .RST_N(rst_n), .CLOCK_GATED(G_clock_out)
);
```

7. After synthesis, check the “IDC.area” and “IDC.timing” in the folder “Report”. The area report is valid only when the slack in the end of “IDC.timing” is **non-negative** and the result should be **MET**.
8. The next input will come in **2~5** cycles after your **out_valid** is pulled down.
9. The synthesis result **cannot** contain any **LATCH except for clocking gating latch.**
10. The synthesis result **cannot** contain any error**.**
11. The output loading is set to 0.05.
12. Input delay and output delay are 0.5*Clock Period.
13. You can’t have timing violation in gate-level simulation.
14. You **can’t use memory** in this lab.
15. You **can’t use DesignWare IP** in this lab.
16. The execution latency is limited in **1000 cycles.**
17. The **out_valid** cannot overlap with **in_valid**.
18. **The redundant cycle is forbidden in this lab.**
19. Your design should have **at least 15% power reduction** from **cg_en-off to cg_en-on**, otherwise it will be regarded as failed. $\frac{P_{cg\_enoff}-P_{cg\_enon}}{P_{cg\_enoff}} \geq 15\%$
20. **Power report position: 04_PTPX/Report/IDC_POWER or IDC _CG_POWER**
    Report Total Power Example:

wocg without clock gating.

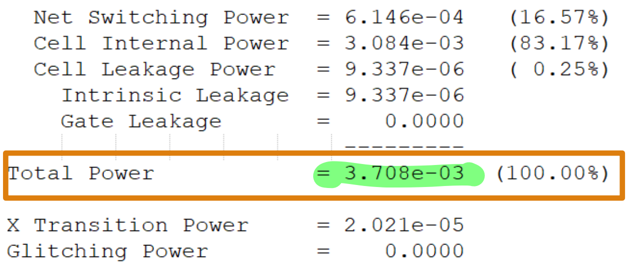
<table>
<tr><th>IDC _POWER (w/o clock gating)</th><th>IDC_CG_POWER (with clock gating)</th></tr>
<tr><td>



```
  Net Switching Power  = 6.760e-04   (14.60%)
  Cell Internal Power  = 3.946e-03   (85.20%)
  Cell Leakage Power   = 9.339e-06   ( 0.20%)
    Intrinsic Leakage  = 9.339e-06
    Gate Leakage       =    0.0000
                         ---------
Total Power            = 4.632e-03  (100.00%)

X Transition Power     = 2.021e-05
Glitching Power        =    0.0000
```

5.633

</td><td>



```
  Net Switching Power  = 6.146e-04   (16.57%)
  Cell Internal Power  = 3.084e-03   (83.17%)
  Cell Leakage Power   = 9.337e-06   ( 0.25%)
    Intrinsic Leakage  = 9.337e-06
    Gate Leakage       =    0.0000
                         ---------
Total Power            = 3.708e-03  (100.00%)

X Transition Power     = 2.021e-05
Glitching Power        =    0.0000
```

</td></tr>
</table>

21. The gate level simulation cannot include any timing violations without the *notimingcheck* command

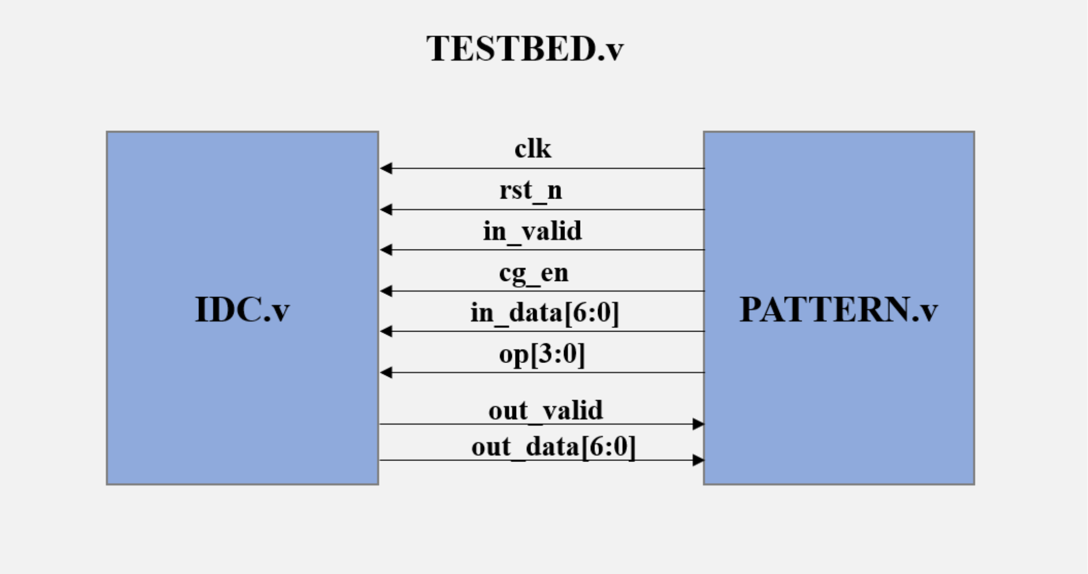
22. Don't use any wire/reg/submodule/parameter name called *error*, *congratulation*, *latch* or *fail* otherwise you will fail the lab. Note: * means any char in front of or behind the word. e.g: error_note is forbidden.

23. Don' t write Chinese comments or other language comments in the file you turned in.

24. Verilog commands //synopsys dc_script_begin, //synopsys dc_script_end //synopsys translate_off, //synopsys translate_on are only allowed during the usage of including and setting designware IPs, other design compiler optimizations are forbidden.

25. Using the above commands are allowed, however any error messages during synthesis and simulation, regardless of the result will lead to failure in this lab.Using the above commands are allowed, however any error messages during synthesis and simulation, regardless of the result will lead to failure in this lab.

➢ Block Diagram



➢ Grading Policy

- **Functionality (70%)：**
  - IDC_wocg and IDC：**60%**
  - JasperGold SEC check (10%)：**Run1 (5%) , Run2 (5%)**

    (JasperGold No 2nd demo chance.)
- **Performance (30%)：**

  (**Total Latency** * **Total Power** *(gated with CG)*) * **Area**

➢ Note

1. Please upload the following file on NewE3 platform before **23:59** on **May. 22**

   RTL design：**IDC_wocg_iclabXXX.v, IDC_iclabXXX.v** (**XXX** is your account no.)
2. Template folders and reference commands:

   01_RTL/ (RTL simulation)

   "./01_run", **"./02_run_cg"***(only in EXERCISE folder)*

   02_SYN/ (Synthesis)

   "./01_run_CG_dc"

   (Check the design which contains **latch and error** or not in **syn.log**)

   03_GATE_SIM/ (Gate Level simulation)

   "./01_run", **"./02_run_cg"***(only in EXERCISE folder)*

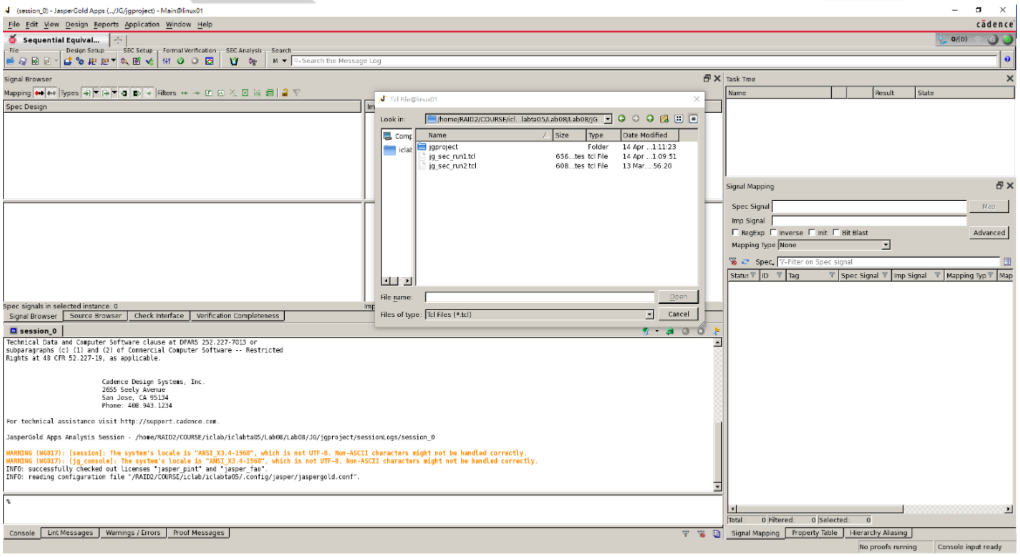
   04_PTPX/

   "./01_run_ptpx" & **"./02_run_cg_ptpx"**to get the power of your design.

● You can key in **./09_clean_up** to clear all log files and dump files in each folder

3. JasperGold SEC execution steps: In folder JG/

   (1) **"./01_run"**、**"./02_run"** or "**jg –sec &**" and click the "File/Tcl_scripts/source".

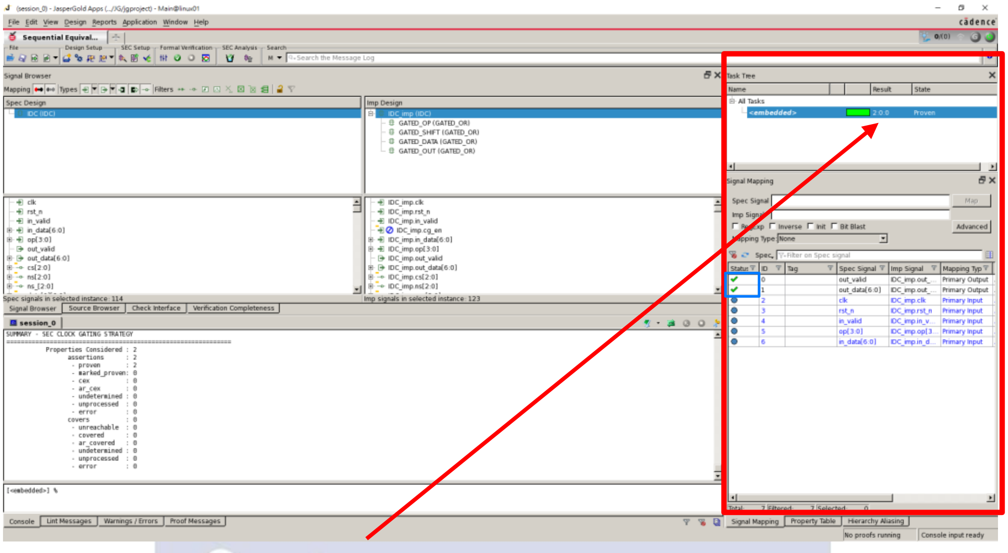


Name	Size	Type	Date Modified
jgproject		Folder	14 Apr ...1:11:23
jg_sec_run1.tcl	656...tes	tcl File	14 Apr ...1:09:51
jg_sec_run2.tcl	608...tes	tcl File	13 Mar...:56:20

```
Technical Data and Computer Software clause at DFARS 252.227-7013 or
subparagraphs (c) (1) and (2) of Commercial Computer Software -- Restricted
Rights at 48 CFR 52.227-19, as applicable.


                          Cadence Design Systems, Inc.
                          2655 Seely Avenue
                          San Jose, CA 95134
                          Phone: 408.943.1234

For technical assistance visit http://support.cadence.com.

JasperGold Apps Analysis Session - /home/RAID2/COURSE/iclab/iclabta05/Lab08/Lab08/JG/jgproject/sessionLogs/session_0

WARNING (WG017): [session]: The system's locale is "ANSI_X3.4-1968", which is not UTF-8. Non-ASCII characters might not be handled correctly.
WARNING (WG017): [jg_console]: The system's locale is "ANSI_X3.4-1968", which is not UTF-8. Non-ASCII characters might not be handled correctly.
INFO: successfully checked out licenses "jasper_pint" and "jasper_fao".
INFO: reading configuration file "/RAID2/COURSE/iclab/iclabta05/.config/jasper/jaspergold.conf".
```

```
%
```

(2) Check all properties pass.



Name		Result	State
All Tasks			
<embedded>	🟩	2:0:0	Proven

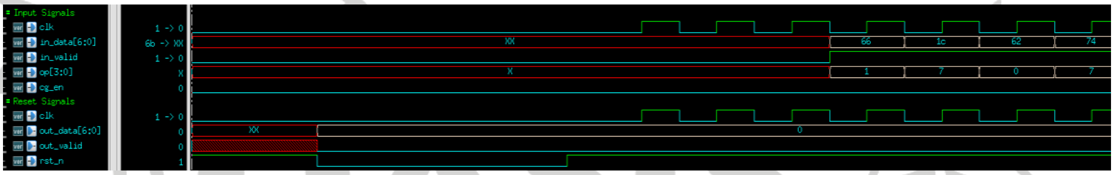
Status	ID	Tag	Spec Signal	Imp Signal	Mapping Typ
✓	0		out_valid	IDC_imp.out_...	Primary Output
✓	1		out_data[6:0]	IDC_imp.out_...	Primary Output
●	2		clk	IDC_imp.clk	Primary Input
●	3		rst_n	IDC_imp.rst_n	Primary Input
●	4		in_valid	IDC_imp.in_v...	Primary Input
●	5		op[3:0]	IDC_imp.op[3...	Primary Input
●	6		in_data[6:0]	IDC_imp.in_d...	Primary Input

```
SUMMARY - SEC CLOCK GATING STRATEGY
=================================================================
          Properties Considered : 2
                assertions      : 2
                 - proven       : 2
                 - marked_proven: 0
                 - cex          : 0
                 - ar_cex       : 0
                 - undetermined : 0
                 - unprocessed  : 0
                 - error        : 0
                covers          : 0
                 - unreachable  : 0
                 - covered      : 0
                 - ar_covered   : 0
                 - undetermined : 0
                 - unprocessed  : 0
                 - error        : 0

[<embedded>] %
```

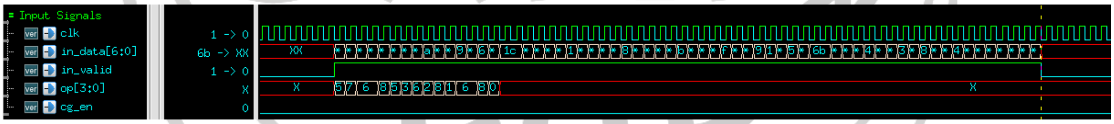
You should pass all tasks in both "./01_run" 、"./02_run".

➢ Waveform Example

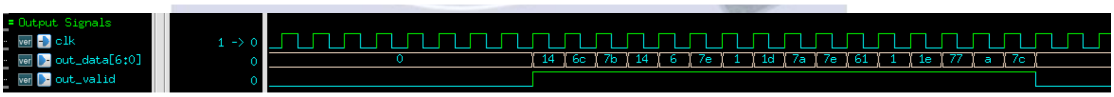
1. Reset Signal



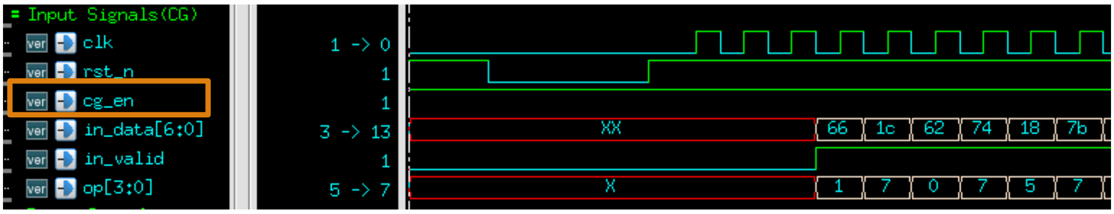
2. Input of **op** and **in_data**



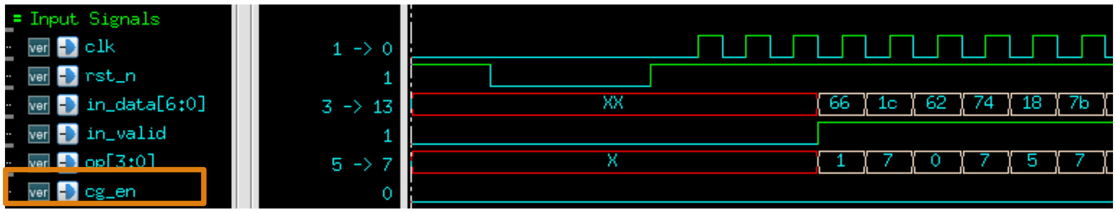
3. Output

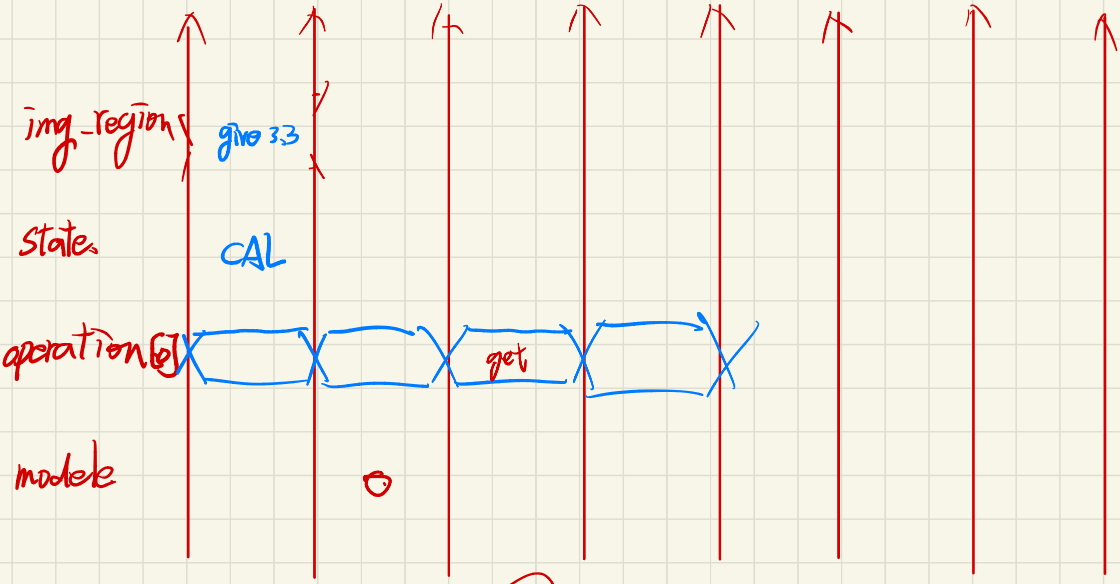


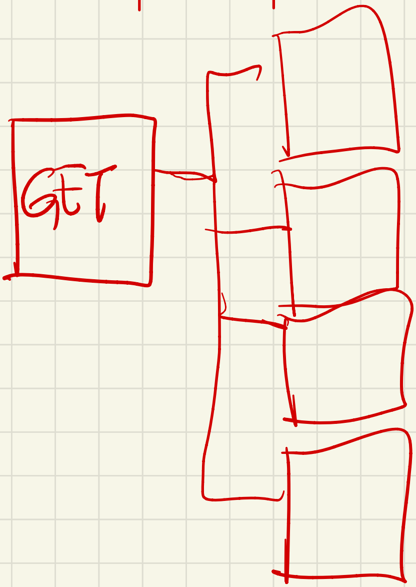
4. For PATTERN_CG.v, you need to set cg_en = 1：

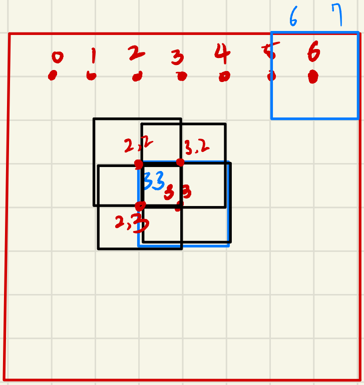


5. For PATTERN.v, you need to set cg_en = 0：




img_region
give 3.3
state
CAL
operation[0]
get
modele
0


GET


6
7
0
1
2
3
4
5
6
2,2
3,2
33
3,3
2,3