

# National Cheng Kung University

## Department of Electrical Engineering

### *Introduction to VLSI CAD (Spring 2025)*

#### Lab Session 4

#### Register File and FIFO Design with Manhattan and Euclidean Distance Applications

Name	Student ID	
Practical Sections	Points	Marks
Lab in class	10	
Prob A	40	
Prob B-1	10	
Prob B-2	20	
Report	20	
Notes:		

**Due Date: 15:10, March 19, 2025 @ moodle**

## Deliverables

- 1) All Verilog codes including testbenches for each problem should be uploaded.  
NOTE: Please **DO NOT** include source code in the paper report!
- 2) All homework requirements should be uploaded in this file hierarchy or you will not get the full credit.  
NOTE: Please **DO NOT** upload waveforms!
- 3) **Important! TA will use the command in Appendix A to check your design under SoC Lab environment, if your code can not be recompiled by TA successfully using the commands, you will not get the full credit.**
- 4) If you upload a dead body which we can't even compile you will get **NO** credit!
- 5) All Verilog file should get at least **90%** superLint Coverage.
- 6) **File hierarchy should not be changed; it may cause your code can not be recompiled by TA successfully using the autograding commands**

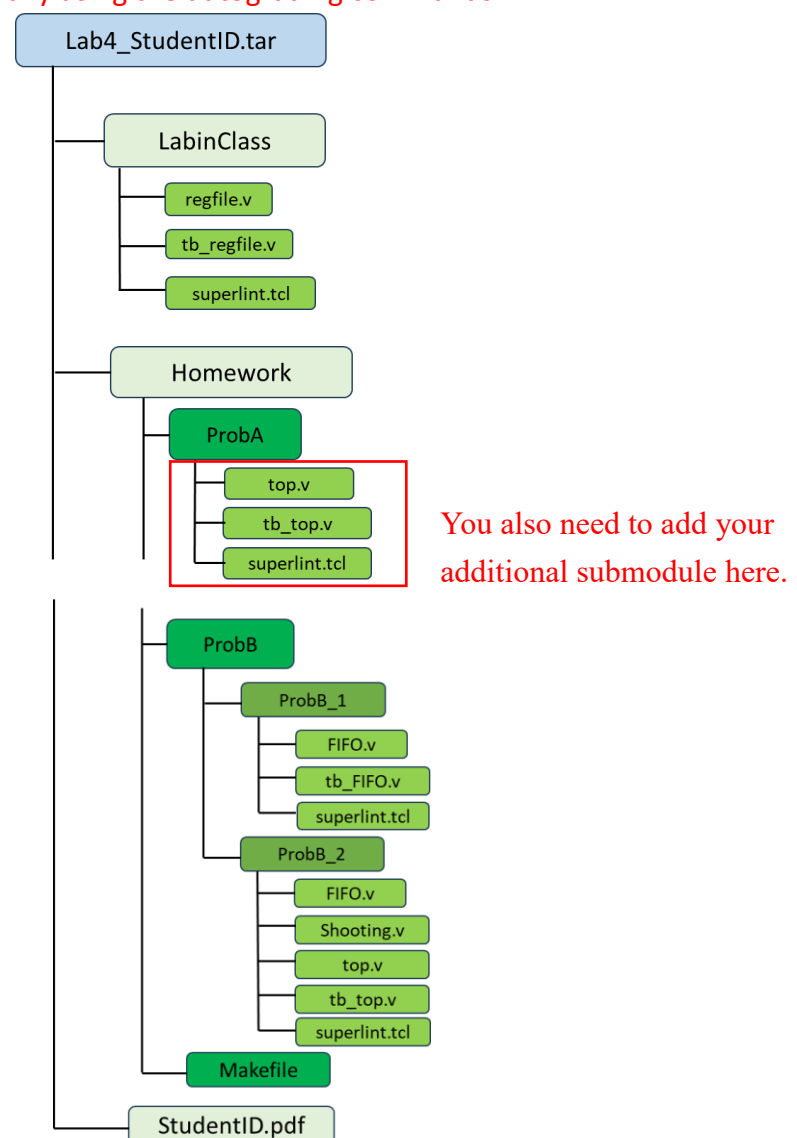
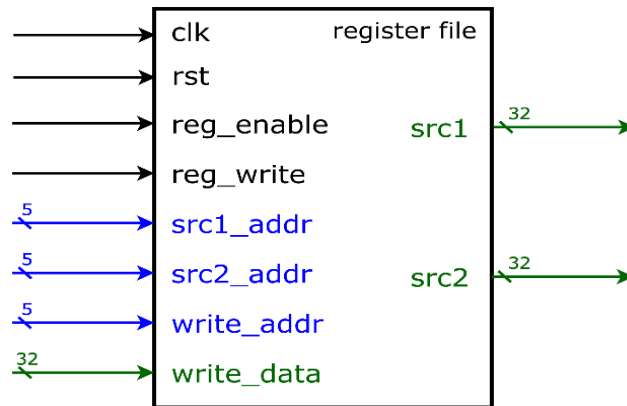


Fig.1 File hierarchy for Homework submission

## Lab in class: 32x32 bit Register File



1. Design a **32 x 32** bit register file with **2** output ports.
2. Port list

Signal	Type	Bits	Description
clk	input	1	clock
rst	input	1	reset
reg_enable	input	1	0 → off 1 → on
reg_write	input	1	0 → read 1 → write
src1_addr	input	5	source1 address
src2_addr	input	5	source2 address
write_addr	input	5	write address
write_data	input	32	write data
src1	output	32	read data source1
src2	output	32	read data source2

- 3.

Show the simulation result on the terminal

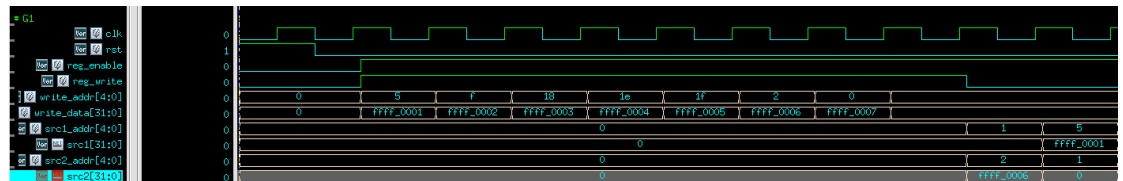
```

*****
**                                     **
**  Congratulations !!               **
**  Simulation PASS!!               **
**                                     **
*****
*****
$finish called from file "tb_regfile.v", line 208.
$finish at simulation time 33600
VCS Simulation Report
Time: 336000 ps
CPU Time: 0.120 seconds; Data structure size: 0.0Mb
Fri Mar 14 00:46:06 2025
CPU time: .122 seconds to compile + .139 seconds to elab + .113 seconds to link + .142 sec

```

4. Show waveforms to explain that your register work correctly when **read** and **write**.

Use waveforms to explain that your register work correctly when **read**

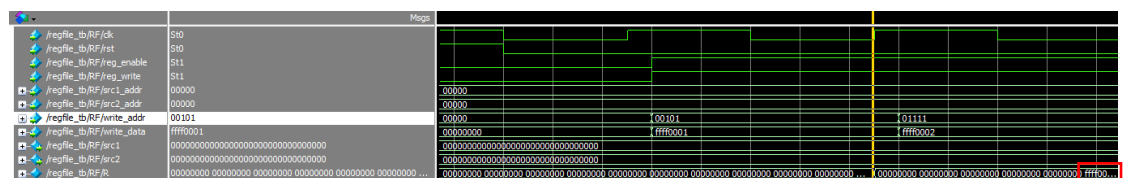


在寫入完成後，當 `reg_enable = 1`, `Reg_write = 0` 時要讀取 `scr1=1` 和 `scr2=2` 的值，因此讀取 `R[1]`和 `R[2]`的值。

`R[1]`因為沒有寫入，維持初始值 0，

`R[2]`有寫入 `ffff_0006`，輸出 `ffff_0006`。

Use waveforms to explain that your register work correctly when **write**

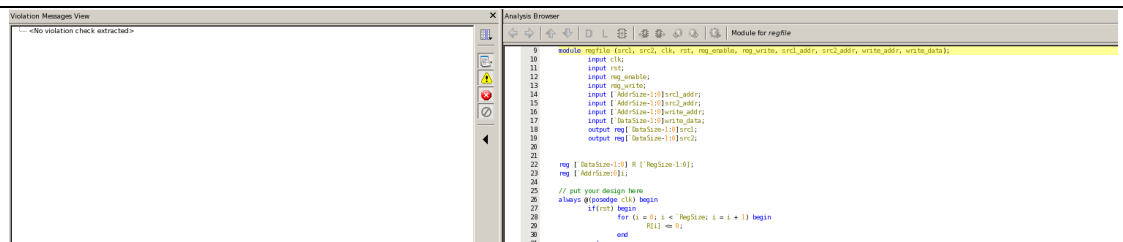


`Reg_write = 1`, `reg_enable = 1`, `write_addr = 5`, `write_data = ffff0001`，

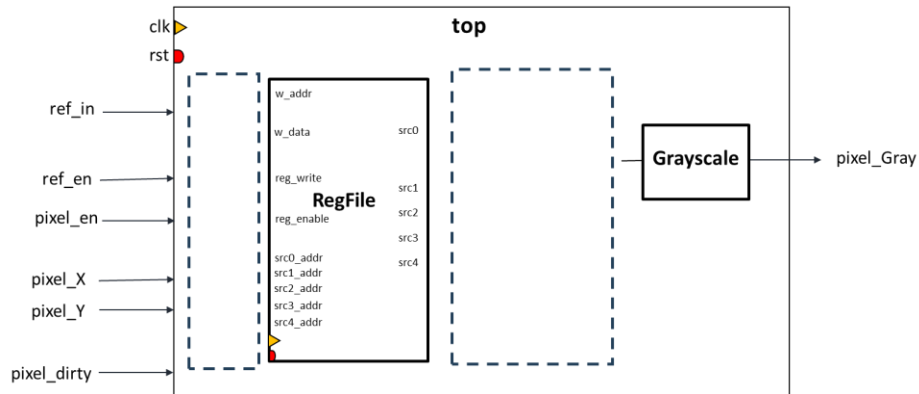
因此在下一個 `clk`, `R[5]`內儲存的資料從 0->ffff0001

- 5.

Show SuperLint coverage



## Prob A: Image Repair



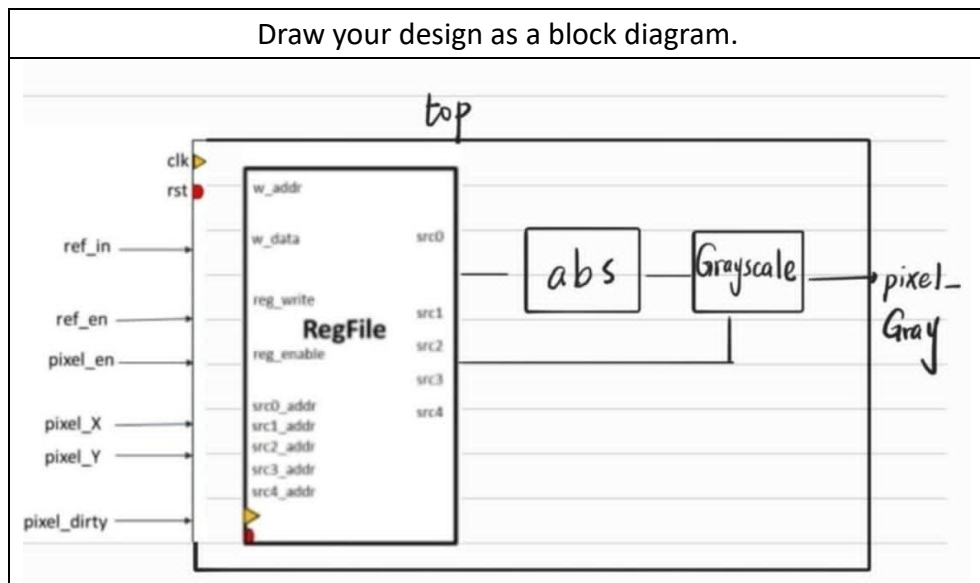
1. Design an image repair circuit in top.v, and set the ports of the top module according to the diagram below.

Signal	Type	Bits	Description
clk	input	1	Clock signal
rst	input	1	Reset, active high
ref_en	input	1	Write compared pixel enable. When ref_en is high, then ref_in is available
ref_in	input	28	Reference pixel
pixel_en	input	1	Write input pixel coordinate enable When pixel_en is high, then pixel_X, pixel_Y, pixel_dirty is available
pixel_X	input	2	x-coordinate
pixel_Y	input	2	y-coordinate
pixel_broken	input	1	1 :Pixel is broken 0: Pixel isn't broken
pixel_Gray	output	8	Grayscale output

2. Design a register file to store the RGB values of each pixel.
3. Design a Grayscale module that can convert RGB to grayscale.
4. Connect all the designed modules in top.v.

**Note:** Make sure to include the used modules in top.v.

5.



6.

Show the simulation result on the terminal

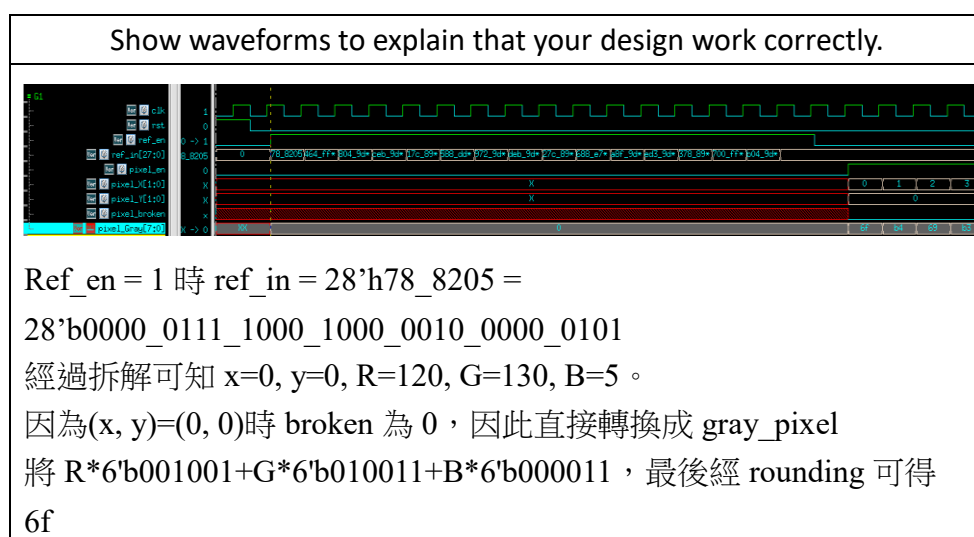
```

*****
**                                     **
** Congratulations !!                 **
** Simulation PASS!!                 **
**                                     **
*****

$finish called from file "tb_top.v", line 557.
$finish at simulation time 119600
VCS Simulation Report
Time: 1196000 ps
CPU Time: 0.110 seconds; Data structure size: 0.0Mb
Fri Mar 14 01:52:17 2025
CPU time: .141 seconds to compile + .144 seconds to elab + .115 seconds to link + .133 seconds

```

7.



8.

### Show SuperLint coverage

**Description (Order by Category)**

- Category CODINGSTYLE (5)
  - Tag CSC\_MS\_LARG (1)
    - "Constant '16' will be left-padded by 8 '0' bits"
  - Tag INS\_NH\_PTEX (1)
    - "A constant is used in a port expression"
  - Tag ASS\_MS\_SMD (3)
    - Unequal length operands in assignment in module/design-unit 'top'. Length of RHS is less than LHS. LHS 'products' L**
    - "Unequal length operands in assignment in module/design-unit 'top'. Length of RHS is less than LHS. LHS 'products' L"
    - "Unequal length operands in assignment in module/design-unit 'top'. Length of RHS is less than LHS. LHS 'products' L"
- Category FLEETFORMAT (2)
  - Tag APF\_MS\_DRNG (2)
    - "The latches 'neighbor\_src1' in the process/always block are mixed with combinational logic"
- Category SYNTHESIS (3)
  - Tag LSC\_NH\_MKCB (1)
    - "In module/design-unit regfile, latch is assigned by blocking assignments"
  - Tag LAF\_NH\_BIAS (1)
    - "Sensitivity list incomplete in module top, missing signal(s): pixel\_en"
  - Tag AUP\_IC\_SENL (1)
    - "In module/design-unit regfile, latch is assigned by blocking assignments"

```

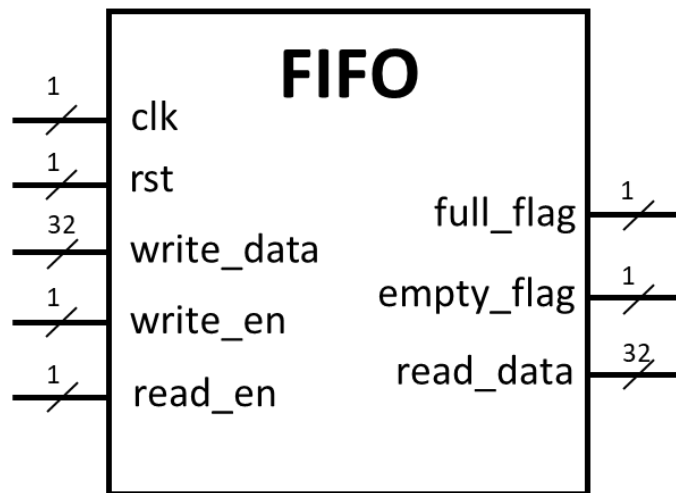
1 timescale 1ns/10ps
2
3 //include your design here
4 //do include "Example.v"
5 include "regfile.v"
6 include "aio.v"
7
8 module top (pixel_gray, clk, rst, ref_en, ref_in, pixel_en, pixel_x, pixel_y, pixel_broken);
9   input
10     clk;
11   input
12     rst;
13   input [27:0] ref_in;
14   input [1:0] pixel_en;
15   input [1:0] pixel_x;
16   input [1:0] pixel_y;
17   output [7:0] pixel_gray;
18   output pixel_broken;
19
20   // put your design here
21   wire [1:0] in_addr; not addr;
22   wire [27:0] neighbor_src[0:3];
23   assign in_addr = ref_in[27:26] * (ref_in[25:24] << 2);
24   assign out_addr = pixel_y * (pixel_x << 2);
25   regfile regfile1 (.src(pixel_rgb), .clk(clk), .rst(rst), .reg_enable[1:0], .reg_write(ref_en), .s
26
27   reg [7:0] R_distance, G_distance, B_distance;
28   wire [1:0] result_distance[0:3];
  
```

Coverage = 142/152 = 93%

---

*Prob B-1: FIFO*

---



1. Design a synchronous FIFO with a depth of 8 (able to store 8 data) and a data bit-width of 32 bits.
2. Port list

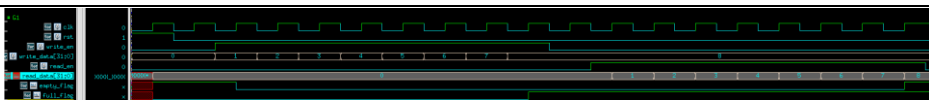
Signal	I/O	Bits	Description
clk	Input	1	Clock signal
rst	Input	1	Reset signal(active high)
write_en	Input	1	Write enable
write_data	Input	32	Data to be written into FIFO
read_en	Input	1	Read enable
read_data	Output	32	Data being read out of FIFO
full_flag	Output	1	Indicates FIFO is full or not
empty_flag	Output	1	Indicates FIFO is empty or not



3.

Show the simulation result on the terminal
<pre>-----FIFO test-----  100 The FIFO is full. full_flag is 1. 110 The FIFO is full. full_flag is 1. 120 read_data 00000001 is correct 130 read_data 00000002 is correct 140 read_data 00000003 is correct 150 read_data 00000004 is correct 160 read_data 00000005 is correct 170 read_data 00000006 is correct 180 read_data 00000007 is correct 190 read_data 00000008 is correct 190 The FIFO is empty. empty_flag is 1.  ***** **                                     ** ** Congratulations !!                ** **                                     ** ** Simulation PASS!!                 ** **                                     ** *****            / \               / 0.0           /_____         /  ^  ^  ^  \       /  ^  ^  ^  \ w      /  ^  ^  ^  \     /  ^  ^  ^  \    /  ^  ^  ^  \   /  ^  ^  ^  \  /  ^  ^  ^  \ /  ^  ^  ^  \ \m  ^  ^  ^  \  \  ^  ^  ^  \  \$finish called from file "tb_FIFO.v", line 178. \$finish at simulation time 19000 VCS Simulation Report Time: 190000 ps CPU Time: 0.130 seconds; Data structure size: 0.0Mb Fri Mar 14 02:06:51 2025 CPU time: .125 seconds to compile + .148 seconds to elab + .116 seconds to link + .139 seconds in s 74101034@CSH: ~\$</pre>

4.

Show waveforms to explain that your design work correctly
<div><p>一開始先將 input 1, 2, ...7, 8 依序讀入 queue, 結束後 full 訊號升起, 之後開始做 pop, 依序為 1,2,...7,8, 此時 empty_flag 升起。</p></div>

5.

Show SuperLint coverage
-------------------------



#### Show the simulation result on the terminal

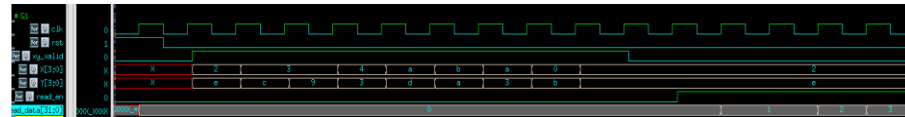
```
----- Second shot -----
FIFO[      0] is correct
FIFO[      1] is correct
FIFO[      2] is correct
FIFO[      3] is correct
FIFO[      4] is correct
FIFO[      5] is correct
FIFO[      6] is correct
FIFO[      7] is correct

*****
**                                     **
** Congratulations !!                 **
** Simulation PASS!!                 **
**                                     **
*****
                                     \_/_/
                                     / 0.0 \
                                     / ^ ^ ^ \
                                     | ^ ^ ^ |w
                                     \m _ m _/

$finish called from file "tb_top.v", line 142.
$finish at simulation time 48100
VCS Simulation Report
Time: 481000 ps
CPU Time: 0.120 seconds; Data structure size: 0.0Mb
Fri Mar 14 02:12:25 2025
CPU time: .124 seconds to compile + .143 seconds to elab + .116 seconds to link + .142 sec
```

4.

#### Show waveforms to explain that your design work correctly



Input1 x=2, y=14 因在 A 圈內得 1 分並寫入 queue,  
Input2 x=3, y=12 因在 A 圈內得 1 分並寫入 queue,  
Input2 x=3, y=9 因在 B 圈內得 2 分並寫入 queue,  
依此類推，最後再將值 pop 出來。

5.

#### Show SuperLint coverage

Violation Messages View

Category: CODING STYLE (1)

Sup: C89, vs. UCSD9 (1)

"Unequal length operands in equality operator encountered (padding produces incorrect result) in module/design-unit"

Category: SYNTHESIS (2)

Sup: IDEAL, ORNG (2)

"Variable index/range selection of 'write\_ptr' is potentially outside the defined range"

"Variable index/range selection of 'read\_ptr' is potentially outside the defined range"

Category: RACES (3)

Sup: REG, MR, TRAC (3)

"A trigger-propagation race exists between 'shooting\_dist\_square' and 'shooting\_x\_dist[00]'"

"A trigger-propagation race exists between 'shooting\_x\_dist[11]' and 'shooting\_x\_dist[00]'"

"A trigger-propagation race exists between 'shooting\_x\_dist[00]' and 'shooting\_x\_dist[01]'"

"A trigger-propagation race exists between 'shooting\_x\_dist[11]' and 'shooting\_x\_dist[01]'"

Analysis Browser

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

Coverage = 121/127= 95%

At last, please write the lesson you learned from Lab4

之前練習 IC contest 時有寫到與圓距離相關寫法，這次練習到讓我更確切知道如何判斷點與圓的相對關係。另外，之前在寫 verilog 就有聽說過如果資料比較大，就會需要寫一個 regfile，如果不大可以直接存在陣列，原本不知道是甚麼意思，經過這次親自實作 regfile，讓我更了解這個觀念。

這次 lab 我覺得最複雜的地方是 always@(\*)和 always@(posedge clk)使用，我有時候不確定某個訊號要放在哪一種 block。

Problem	Command
<b>Lab in class</b>	% vcs -R tb_regfile.v -debug_access+all -full64 +define+FSDB
<b>ProbA</b>	make probA
<b>ProbB-1</b>	make probB_1
<b>ProbB-1</b>	make probB_2

