NCTU-EE IC LAB - Fall 2021

Lab06 Exercise

Design: Security Attack

Data Preparation

1. Extract test data from TA's directory:

```
% tar xvf ~iclabta01/Lab06.tar
```

2. The extracted LAB directory contains:
   a. **00_TESTBED**
   b. **01_RTL**
   c. **02_SYN**
   d. **03_GATE**

Design Description

In mathematics, a finite field or Galois field (so-named in honor of Évariste Galois) is a field that contains a finite number of elements. As with any field, a finite field is a set on which the operations of multiplication, addition, subtraction and division are defined and satisfy certain basic rules.

The most common examples of finite fields **GF(p)** are given by the integers **mod *p*** when ***p*** is a prime number, or the field **GF(q)** (k power of p is q, k is an integer) may be explicitly constructed in an chosen irreducible polynomial $\mathrm{f(x)}$ in **GF(p)[X]**(GF(p)[X] is a polynomial finite field with coefficient in GF(p)) of degree n. Then the quotient ring of the polynomial ring **GF(p)[X]** by the ideal generated by $\mathrm{f(x)}$ is a field of order **q. GF(q) = GF(p)[X]/(**$\mathrm{f(x)}$**)**

Finite fields are fundamental in a number of areas of mathematics and computer science, including number theory, algebraic geometry, Galois theory, finite geometry, cryptography and coding theory.

For example, GF(2^2) is a finite field has elements only under "4", and the chosen irreducible polynomial is $\mathrm{f(x)} = \mathrm{x}^2 + \mathrm{x} + 1$. GF(4) = GF(2)[x]/($\mathrm{x}^2 + \mathrm{x} + 1$)

That is the whole elements are { 0 , 1 , 2 , 3 } or {00, 01 ,10, 11} . And arithmetic under GF(4) are described below.

1. Addition (A+B) A is column B is row

	00	01	10	11
00	00	01	10	11
01	01	00	11	10
10	10	11	00	01
11	11	10	01	00

	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

For example : A = 3, B =2, A+B = ?

A = 11 = $x+1$

B = 10 = x

A + B = $(x+1)$ + x = (1+1)x + 1 = 0x + 1 = 1

2. Substation (A-B) A is column B is row

	00	01	10	11
00	00	01	10	11
01	01	00	11	10
10	10	11	00	01
11	11	10	01	00

	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

For example : A = 3, B =2, A-B = ?

A = 11 = $x+1$

B = 10 = x

A - B = $(x+1)$ - x = (1+(-1))x + 1 = (1+1)x + 1 = 0x + 1 = 1

3.Multiplication (A*B) A is column B is row

	00	01	10	11
00	00	00	00	00
01	00	01	10	11
10	00	10	11	01
11	00	11	01	10

	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	3	1
3	0	3	1	2

For example : A = 3, B =2, A*B = ?

A = 11 = $x+1$

B = 10 = x

A * B = $(x+1)$ * x = (x^2+x)% (x^2+x+1) = 1

Hint:

$x^2+x+1 = 0$, $x^2 = x+1$

A * B = x^2+x = $(x+1)+x$ = 1

4.Division (A/B) A is column B is row

	00	01	10	11
00	-	00	00	00
01	-	01	11	10
10	-	10	01	11
11	-	11	10	01

	0	1	2	3
0	-	0	0	0
1	-	1	3	2
2	-	2	1	3
3	-	3	2	1

A / B can seen as $A * B^{-1}$.

By using Extended Euclid’s algorithm can easily find the inverse of divisor B.

For Extended Euclid algorithm has three part

Part A : Polynomial divide

Part B : Multiple quotient

Part C : Multiple inverse

For example, in GF(16), f(x) = $x^4 + x^3 + 1$, A = 2, B = 9, find A / B = ?

$\mathrm{A} / \mathrm{B} = \mathrm{A} * \mathrm{B}^{-1}$

$\mathrm{B} * \mathrm{B}^{-1} = 1 \,\%\, f(x)$

Part A : Polynomial divide

Step 1: initial

$\mathrm{R}_{-1} = \mathrm{f(x)} = x^4 + x^3 + 1$

$\mathrm{R}_0 = \mathrm{B} = x^3 + 1$

$\boldsymbol{i}$	Remainder($\mathbf{R_i}$)	Quotient($\boldsymbol{Q}_\mathbf{i}$)	division($\mathbf{DN_i}$)	divisor($\mathbf{DR_i}$)
-1	$x^4 + x^3 + 1$	-	-	-
0	$x^3 + 1$	-	-	-

Step 2: divide R_i until $R_i = 1$

Step 2-1 : first divide round

R_{-1} is bigger than $\mathrm{R}_0, \mathrm{E}_1 = 0,\quad \mathrm{DN}_0 = \mathrm{R}_{-1},\ \mathrm{DR}_0 = \mathrm{R}_0$

$\mathrm{DN}_1 + \mathrm{Q}_1 * \mathrm{DR}_1 = \mathrm{R}_1$

$(x^4 + x^3 + 1) + x * (x^3 + 1) = x^3 + x + 1$

$\boldsymbol{i}$	$\mathbf{R_i}$	$\boldsymbol{Q}_\mathbf{i}$	$\mathbf{DN_i}$	$\mathbf{DR_i}$
-1	$x^4 + x^3 + 1$	-	-	-
0	$x^3 + 1$	-	-	-
1	$x^3 + x + 1$	x	$x^4 + x^3 + 1$	$x^3 + 1$

Step 2-2 : second divide round

If DR_1 >= R_1

$\mathrm{DN}_2 = \mathrm{DR}_1,\ \mathrm{DR}_2 = \mathrm{R}_1$, else

$\mathrm{DN}_2 = \mathrm{R}_1,\ \mathrm{DR}_2 = \mathrm{DR}_1$

$\mathrm{DN}_2 + \mathrm{Q}_2 * \mathrm{DR}_2 = \mathrm{R}_2$

$x^3 + 1 < x^3 + x + 1$

$(x^3 + x + 1) + (x^3 + 1) * 1 = x$

$\boldsymbol{i}$	$\mathbf{R_i}$	$\boldsymbol{Q}_\mathbf{i}$	$\mathbf{DN_i}$	$\mathbf{DR_i}$
-1	$x^4 + x^3 + 1$	-	-	-
0	$x^3 + 1$	-	-	-
1	$x^3 + x + 1$	x	$x^4 + x^3 + 1$	$x^3 + 1$
2	x	1	$x^3 + x + 1$	$x^3 + 1$

Step 2-3 : i-th divide round

If $\mathrm{DR}_{\mathrm{i}-1}$ >= R_{i-1}

$\mathrm{E}_i = 0, \mathrm{DN}_\mathrm{i} = \mathrm{DR}_{\mathrm{i}-1},\ \mathrm{DR}_\mathrm{i} = \mathrm{R}_{\mathrm{i}-1}$, else

$\mathrm{E_i} = 1, \mathrm{DN_i} = \mathrm{R_{i-1}}\ , \mathrm{DR_i} = \mathrm{DR_{i-1}}$

$\mathrm{R_i} = \mathrm{DN_i} + \mathrm{DR_i} * \mathrm{Q}_i$

i	$\mathbf{R_i}$	$\boldsymbol{Q}_\mathbf{i}$	$\mathbf{DN_i}$	$\mathbf{DR_i}$
-1	$x^4 + x^3 + 1$	-	-	-
0	$x^3 + 1$	-	-	-
1	$x^3 + x + 1$	x	$x^4 + x^3 + 1$	$x^3 + 1$
2	x	1	$x^3 + x + 1$	$x^3 + 1$
3	1	x^2	$x^3 + 1$	x

Part B : multiple quotient

Step 1 : initial

MQ(multiple quotient)

mql(MQ low)

mqh(MQ high)

$\mathbf{MQ_{-1}} = 0 \qquad \mathbf{MQ_0} = 1$

Step 2 : find $\mathbf{MQ_n}$, while $\mathrm{R}_n = 1$

Step 2-1: first round

$\boldsymbol{mqh_1} = \mathbf{MQ_{-1}} \quad \boldsymbol{mql_1} = \mathbf{MQ_0}$

$\mathbf{MQ_1} = \boldsymbol{mqh_1} + \boldsymbol{mql_1} * \mathrm{Q_1}$

i	$\mathbf{MQ_i}$	$\boldsymbol{mqh}_\mathbf{i}$	$\boldsymbol{mql}_\mathbf{i}$	$\mathbf{R_i}$	$\boldsymbol{Q}_\mathbf{i}$	$\mathbf{DN_i}$	$\mathbf{DR_i}$
-1	0			$x^4 + x^3 + 1$	-	-	-
0	1			$x^3 + 1$	-	-	-
1	x	0	1	$x^3 + x + 1$	x	$x^4 + x^3 + 1$	$x^3 + 1$
2				x	1	$x^3 + x + 1$	$x^3 + 1$
3				1	x^2	$x^3 + 1$	x

Step 2-2: second round

If $\mathrm{DR_1}$ >= $\mathrm{R_1}$, $\mathrm{mq}h_2 = mql_1,\ mql_2 = \mathrm{MQ_1}$

else $\mathrm{mqh_2} = \mathrm{MQ_1},\ mql_2 = \mathrm{mql_1}$

$MQ_2 = mqh_2 + mql_2 * Q_2$

i	$\mathbf{MQ_i}$	$\boldsymbol{mqh}_\mathbf{i}$	$\boldsymbol{mql}_\mathbf{i}$	$\mathbf{R_i}$	$\boldsymbol{Q}_\mathbf{i}$	$\mathbf{DN_i}$	$\mathbf{DR_i}$
-1	0	-	-	$x^4 + x^3 + 1$	-	-	-
0	1	-	-	$x^3 + 1$	-	-	-
1	x	0	1	$x^3 + x + 1$	x	$x^4 + x^3 + 1$	$x^3 + 1$
2	x + 1	x	1	x	1	$x^3 + x + 1$	$x^3 + 1$
3				1	x^2	$x^3 + 1$	x

Step 2-3: i -th round

If DR_{i-1} >= R_{i-1}, $\mathrm{mq}h_i = mql_{i-1},\ mql_i = \mathrm{MQ}_{i-1}$

else $\mathrm{mqh}_i = \mathrm{MQ}_{i-1},\ mql_i = \mathrm{mql}_{i-1}$

$MQ_i = mqh_i + mql_i * Q_i$

i	$\mathrm{MQ_i}$	mqh_i	mql_i	$\mathrm{R_i}$	Q_i	$\mathrm{DN_i}$	$\mathrm{DR_i}$
-1	0	-	-	$x^4 + x^3 + 1$	-	-	-
0	1	-	-	$x^3 + 1$	-	-	-
1	x	0	1	$x^3 + x + 1$	x	$x^4 + x^3 + 1$	$x^3 + 1$
2	$x + 1$	x	1	x	1	$x^3 + x + 1$	$x^3 + 1$
3	$x^3 + x^2 + 1$	1	$x + 1$	1	x^2	$x^3 + 1$	x

Until n-th round, while $\mathrm{R}_n = 1$, $\mathrm{T_n}$ is the inverse function of B.

$\mathrm{B}^{-1} = x^3 + x^2 + 1$

$\mathrm{B}^{-1} = 13$

$\mathrm{A} / \mathrm{B} = \mathrm{A} * \mathrm{B}^{-1} = 2 * 13 = 3$

In this lab, you need to first design a GF(2^k) arithmetic soft IP and use it to design a 2 dimension inverse array calculator supports GF(4), GF(8), GF(16) and GF(32).

$$A = \begin{bmatrix} a_0 & a_1 \\ a_2 & a_3 \end{bmatrix}$$

$$\mathrm{A}^{-1} = \begin{bmatrix} a_3 & -a_1 \\ -a_2 & a_0 \end{bmatrix} * \frac{1}{\det(\mathrm{A})} \ ; \det(\mathrm{A}) \neq 0$$

$$\mathrm{B} = \mathrm{A}^{-1} = \begin{bmatrix} b_0 & b_1 \\ b_2 & b_3 \end{bmatrix}$$

If A has inverse array, output element of array B by b_0 , b_1 , b_2 , b_3

Or output all zero for four cycles.

Inputs and Outputs

Input signal	Bit width	Definition
clk	1	Clock
rst_n	1	Asynchronous active-low reset
in_valid	1	When in_valid high means in_data is valid. Degree and poly is valid in first high cycle.
in_data	5	Input array element from a_0 , a_1 , a_2 , a_3.
degree	3	finite field GF(2^k) degree data k, eq. if degree k = 2, GF(2^k) = GF(4)
poly	6	Irreducible polynomial f(x) to modulo result to fit finite field GF(2^k). For example, f(x) = $x^2 + x + 1$, poly = 6'd7.

Output signal	Bit width	Definition
out_valid	1	Output data is valid
out_data	5	Output inverse array elements by b_0 , b_1 , b_2 , b_3. If there's no inverse array for input array, output "0" for 4 cycles.

1. **in_valid** will come after reset.
2. All input signals are synchronized at **negative edge** of the clock.
3. There is only **one reset** before the first pattern, **out_valid** and **out_result** should be low after initial reset
4. **out_valid** should not be raised when **in_valid** is high. And should not be high more than 4 cycles.
5. The TA’s pattern will capture your output for checking at **clock negative edge**.

Specifications

1. Top module name: **GF_IA** (design file name: **GF_IA.v**)
2. It is **asynchronous** reset and **active-low** architecture.
3. The reset signal would be given only once at the beginning of simulation. All output signals should be reset after the reset signal is asserted.
4. The clock period is within **20ns**.
5. The input delay is set to **0.5*(clock period)**.
6. The output delay is set to **0.5*(clock period)**, and the output loading is set to **0.05**.
7. The input delay of **clk** and **rst_n** should be **zero**.
8. The synthesis result (syn.log) of data type **cannot** include any **latches and error.**
9. After synthesis, you can check **GF_IA.area** and **GF_IA.timing**. The area report is valid when the slack in the end of timing report should be **non-negative (MET)**.
10. The gate level simulation **cannot** include any timing violations **without** the **notimingcheck** command.
11. The next degree will come in 1~3 cycles after your **out_valid** is pulled down.
12. The **out_valid** should be high within **300 cycles** after **last** pulls to low.
13. The performance is determined by **area** and **latency**. The lower, the better.
14. In this lab, you should write your own **syn.tcl file** and **pattern**.
15. Use **top** wire load mode and **compile ultra**.
16. Use analyze + elaborate to read your design.
17. 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.
18. Don’t write chinese comments or other language comments in the file you turned in.
19. 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.
20. Using the above commands are allowed, however any error messages during synthesize and simulation, regardless of the result will lead to failure in this lab.
21. Any form of display or printing information in verilog design is forbidden. You may use this methodology during debugging, but the file you turn in should not contain any coding that is not synthesizable.

Specifications (Soft IP)

1. Top module name: **GF2k** (design file name: **GF2k.v**)
   Input signals : **POLY**, **IN1**, **IN2**
   Output signals：**RESULT**
2. The clock period is **50ns**. Finish calculating within **one** clock cycle.
3. Output loading is set to **0.05**.
4. Using **top** wire load mode and **compile ultra**.
5. Two parameters: **DEG, OP**. DEG means this finite field GF(2^k) degree k. OP means type of operation of this IP. 0 : addition, 1: substation, 2: multiplication, 3: division
6. Input POLY is an irreducible polynomial can expand GF(2) to GF(2^k), and its coefficient in every power of x will be given as bits.
   Ex : DEG = 2, POLY = 7(111)
   GF(2^k) = GF(2^2) = GF(4)
   Irreducible polynomial $\mathrm{f(x)} = 1\mathrm{x}^2 + 1\mathrm{x} + 1$
7. You need to use **generate** to design this soft IP.
8. 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.
9. Don’t write chinese comments or other language comments in the file you turned in.
10. 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.
11. Using the above commands are allowed, however any error messages during synthesize and simulation, regardless of the result will lead to failure in this lab.
12. Any form of display or printing information in verilog design is forbidden. You may use this methodology during debugging, but the file you turn in should not contain any coding that is not synthesizable.

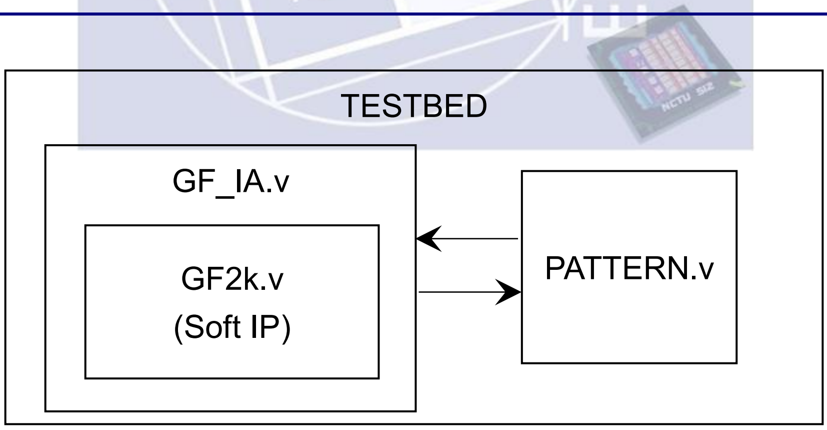
Soft IP Testing environment

```
//synopsys translate_off
`include "GF2k.v"
//synopsys translate_on

module GF_IA(
// input port
POLY, IN1, IN2
// output port
RESULT
);
….

GF2k #(DEG, OP) I_GF2k(
.POLY(POLY)
.IN1 (IN1)
.IN2 (IN2)
.RESULT (RESULT)
);
…
endmodule
```

Block diagram



Note

You need to fill the space in syn.tcl by yourself, but you don’t need to upload to new e3 with your design.

1. **Grading policy:**

   RTL and Gate-level simulation correctness: 40%

   Performance: 30%

   - Performance = Simulation Time * Area

   Soft IP function correctness: 30% (No second demo)

   - Randomly choose 4 values for DEG(2~8), and for each DEG there are 4 OPs.
   - OP:
     - ■ Addition : 1.25%
     - ■ Subtraction : 1.25%
     - ■ Multiple : 2.5%
     - ■ Divide : 2.5%

2. **Please upload the following files on e3 platform before noon (12:00 p.m.) on Next Monday:**

   - Top : GF_IA_iclabXXX.v (XXX = Your account)
   - Soft IP : GF2k_iclabXXX.v
   - Cycle Time : CYCLE_iclabXXX.txt

   Ex: GF_IA_iclab999.v、GF2k_iclab999.v、8.7_iclab999.txt

   If there’s naming error, you will lose 5 points.

3. **Template folders and reference commands:**

   01_RTL/ (RTL simulation) **./01_run**

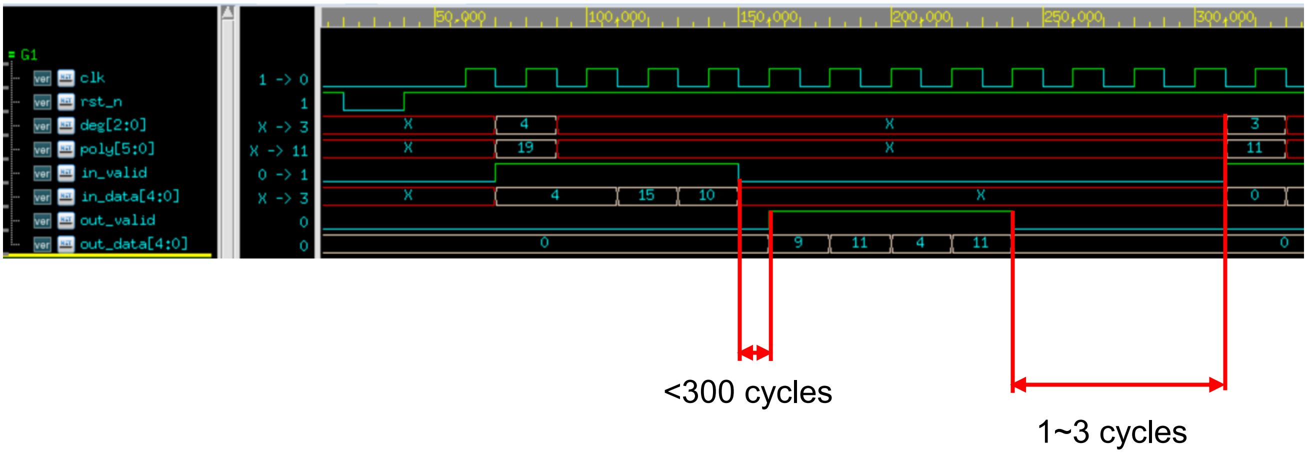
   02_SYN/ (Synthesis) **./01_run_dc**

   (Check the design if there’s latch or not in ***syn.log***)

   (Check the design’s timing in /Report/***CP.timing***)

   03_GATE / (Gate-level simulation) **./01_run**

Sample Waveform



<300 cycles

1~3 cycles

Hint

If you need finite field + - */ answer can find out at this web

http://www.ee.unb.ca/cgi-bin/tervo/calc2.pl?num=6&den=3&f=d&p=2&d=1&y=1&m=1