Solver: Sun ZhiHao

Email Address: ZSUN005@e.ntu.edu.sg

1-(a)

Date No

a	b	c	d	f
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

The LUT is CC17

(b) The routing fabric on FPGAs is a mix of wire lengths and segments to help improve performance.

(c) (i) always @ *
arf(4,4) + bsf(4,4) = sumf(5,4)
sumf(5,4) * cgf(2,5) = prodf(7,9)

(ii)
```
mod                signed
module ~~test~~ (input [7:0] arf, bsf,
                 input signed [6:0] cgf,
                 output signed [8:0] sumf,
                 output signed [15:0] prodf);

always @*
begin
    sumf = arf + bsf;
    prodf = sumf * cgf;
end
endmodule
```

(iii)
```
~~module test_mod~~
module test_mod ();
~~reg arf,~~
    reg [7:0] arf_bsf;
    reg [6:0] cgf;
    wire [8:0] sumf;
    wire [15:0] prodf;
    always #5 clk = ~clk;
    mod M (.arf(arf), .bsf(bsf), .cgf(cgf),
           .sumf(sumf), .prodf(prodf));
initial begin
    arf = 8'b 00101011, bsf = 8'b 00010100,
        cgf = 7'b 0101101;
  #10 arf = 8'b 1111 1100, bsf = 8'b 00100101,
        cgf = ~~8'b~~ 7'b 0110100;
  #10 arf = 8'b 1001 0011, bsf = 8'b 01000111,
        cgf = 7'b 0000111; $finish;
end
endmodule
```

(iv)

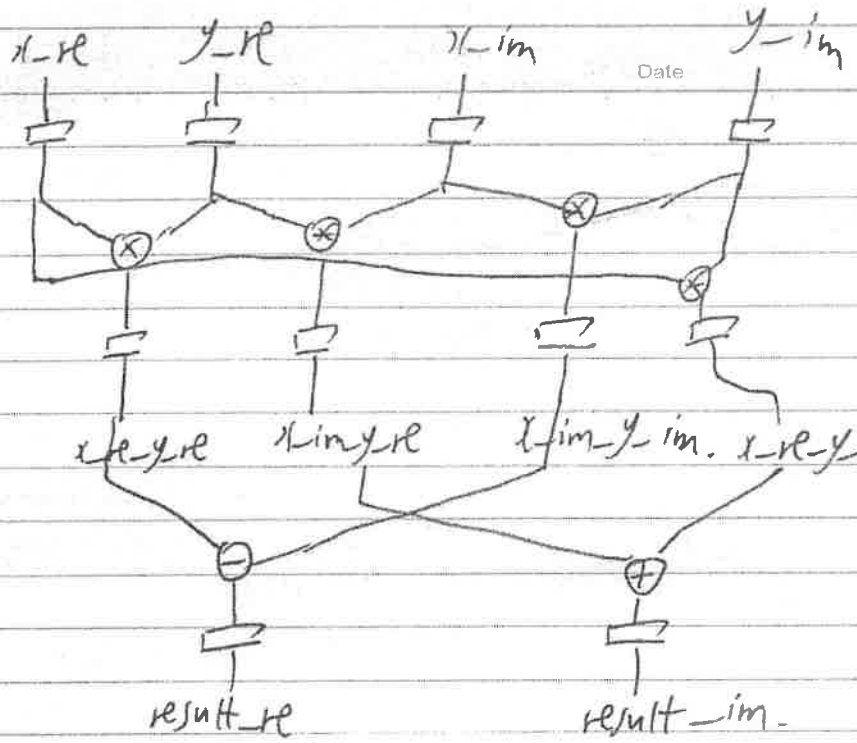
sum f [0] [3.9375] [2.0625] [-2.38]

prod f [0] [-4.3066] [3.35156] [-0.520625]

2. (a). always @ * begin
result_re = x_re * y_re - x_im * y_im;
result_im = x_re * y_im + x_im * y_re;
end

(b) module test2 (input rst, clk,
input [15:0] x_re, y_re, x_im, y_im,
output reg [31:0] result_re, result_im);
~~reg [31:0] xy_re_mid, xy_im_mid,~~
reg [31:0] x_re_y_re, x_im_y_im, x_re_y_im, x_im_y_re;

always @ posedge (clk) begin
x_re_y_re <= x_re * y_re;
x_im_y_im <= x_im * y_im;
x_re_y_im <= x_re * y_im;
x_im_y_re <= x_im * y_re;
result_re <= x_re_y_re - x_im_y_im;
result_im <= x_re_y_im + x_im_y_re;
end
endmodule



(d) For Q2(a) The frequency is $\frac{1}{0.5ns+2ns+1ns}$ = ~~250~~ 222.22 MHz
The latency is ~~4.5ns~~ 4.5 ns.

For Q2(b) The frequency is $\frac{1}{3+0.5ns}$ = 285.71 MHz
The latency is 7 ns.

(e)

```
assign reg[trim = result_re[31:16];
```

3. (a). To implement the instruction, two modifications are needed.

1. between ID/EX register and S2 MUX, on the path of Read Data 1, one more line extends all the way to S1. S1 input signal increases to 2bit

2. After IF/ID register, the P(plus) value extends through ID/EX, EX/Mem to S3. S3 increases to 2 bits control signal

(a)

	PQ = 00	01	11	10	Z
A	A	H	–	B	0
B	A	–	C	B	0
C	–	E	C	F	1
D	–	H	D	B	0
E	G	E	D	–	0
F	G	–	C	F	1
G	G	E	–	F	1
H	A	H	D	–	0

(ABDEH)(CFG)

(AB)(AD)(AH)(DH) (CF)(CG)(FG)

Reduce to 4 states.

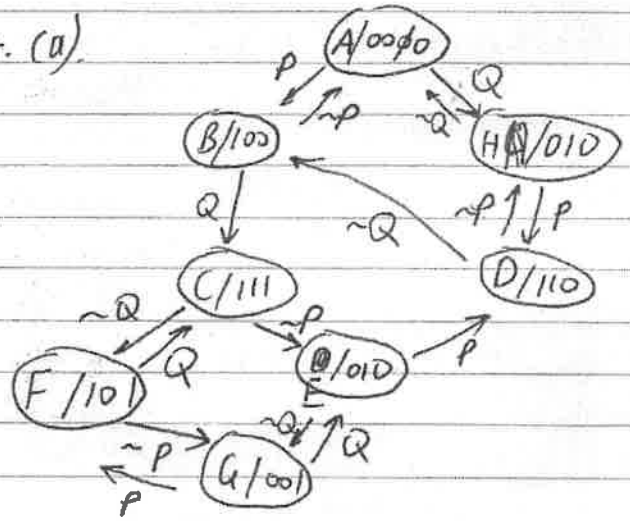
	00	01	11	10	Z
A	A	A	A	B	0
B	A	–	C	B	0
C	C	E	C	C	1
E	C	E	A	–	0

(b). Firstly, beq $1, $5, Loop 2 judges the condition in the ID stage. Even with data forwarding, the input $5 value is still not updated.

Secondly, add $4, $1, before the beq $3, $4, Loop 1, faces similar problem. data forwarding cannot help the ID stage judgement of beq instruction.

(c). The block RAM has two more parity bits to use, increase the 16 bits ISA to 18 bits instruction.

4. (a).



(Q4 (b) (i)

A/00 — C/01
| |
D/10 — B/11

The table changes to

Present State	PQ = 00	01	11	10	Z_2Z_1
A	A	A	D	C	00
B	B	C	B	D	01
C	A	A	B	C	10
D	B	D	D	A	11

(ii)

Y_2Y_1 Present State	PQ = 00	01	11	10	Z_2Z_1
00	00	00	10	11	00
01	01	00	01	10	01
11	00	--	01	11	10
10	01	10	10	11	11

$Z_1 = Y_2 \text{ xor } Y_1$

$Z_2 = Y_2$

$Y_2^+ = PQ' + Y_2'Y_1'P + Y_2Y_1'Q + Y_2Y_1'P$

$Y_1^+ = Y_2'Y_1P'Q' + Y_2Y_1'Q' + PQ'(Y_2 + Y_1') + Y_1PQ + Y_2Y_1P$