0346005 蔡宇翔

Computer Organization

Architecture diagram:

32bit ALU which is layered by 32's 1-bit ALU

Detailed description of the implementation:

1-bit ALU

define wires which used to connect the output of gates
wire XX;

1. connect the wire in the gate like OR(output wire, input A, input B)
2. assign register to output
3. define operation cases (4 cases) which are AND OR ADD less

32bit ALU

1.define wires 32 Ws , 32 Rs and L
2.Ws are used to link the carry out of first ALU cout to second ALU cin then second ALU cout to the third cin ….
Ls are used for the result of each ALU
single L is used to link the sum of ALU31 to the less input of ALU1

3.setting all 32 alu0~ alu31 input and output wire
the first one ALU0 's carry in is different to the others , it needs to carry in when SUB (when A_invert equal to 0 and B_invert equal to 1. So I just use ~ALU_control[3]& ALU_control[2]
Also the ALU0's less is different to others, it consumes wire L

Since the last bit is also the MSB(signed bit), if the sum is 1 then it indicate it's negative which is a < b (a-b<0) then the less of ALU0 will be 1 the others will always be 0.

4.output zero will be set 1 when all results from ALUs are 0.
5. there are 4 situation that overflow bit will be set to 1 and it only happen when we do addition and subtraction
+ add + = -
- add - = +
- sub + = +
+ sub - = -
6. cout bit will be set when ALU31 carry out is 1 which also can happen when doing addition and subtraction

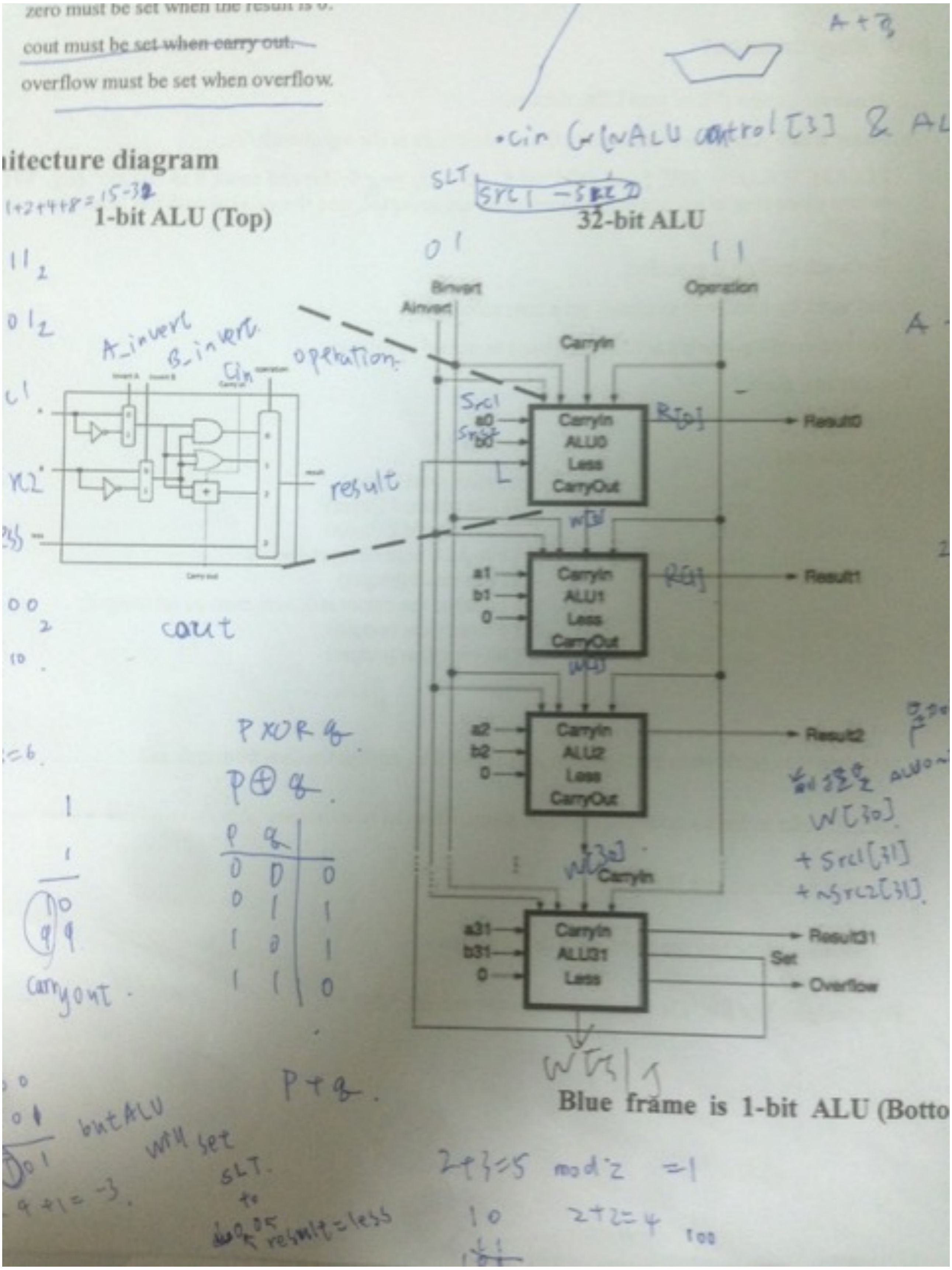
Problems encountered and solutions:

1. installing Xilinx on Linux is unbelievable hard. solution is to use command line to open the software.
2. gate level design or logic level design needs to separate. it's hard to combined both.

Lesson learnt (if any):

it's easier to implement after figured out all situations and the operation of ALU.

0346005　蔡宇翔



zero must be set when the result is 0.

~~cout must be set when carry out.~~

overflow must be set when overflow.

A+B

•cin G=[~ALU control[3] & AL

SLT

Src1 −Src2

itecture diagram

1+2+4+8=15−32

1-bit ALU (Top)

32-bit ALU

11₂
01₂
01
10₂
00₂
10
A_invert
B_invert
Cin
operation
result
cout
Invert A
Invert B
Carry in
operation
Carry out
0 1
1 1
Binvert
Ainvert
Operation
CarryIn
Src1
Src2
a0
b0
L
R[0]
CarryIn
ALU0
Less
CarryOut
Result0
W[1]
a1
b1
0
CarryIn
ALU1
Less
CarryOut
R[1]
Result1
W[2]
a2
b2
0
CarryIn
ALU2
Less
CarryOut
Result2
W[30]
CarryIn
a31
b31
0
CarryIn
ALU31
Less
Result31
Set
Overflow
W[31]

P XOR q.

P⊕q.

p	q	
0	0	0
0	1	1
1	0	1
1	1	0

carryout.

W[30]

+Src1[31]

+~Src2[31].

P+q.

but ALU will set SLT to 0 as result=less

2+3=5 mod 2 =1

10 2+2=4 100

Blue frame is 1-bit ALU (Botto