# IC Design Homework 4

B07502028 吳宗翰

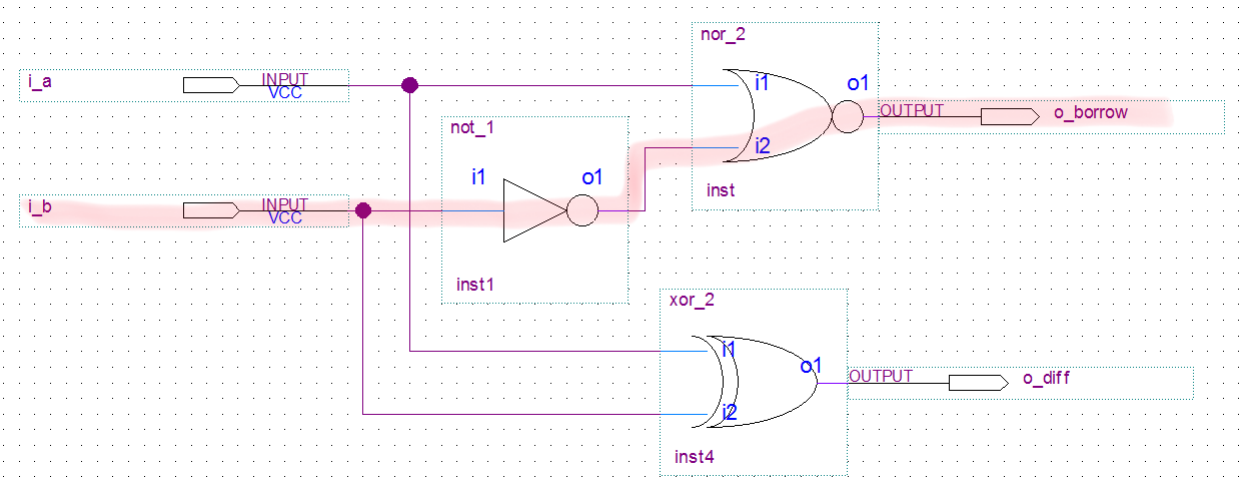
## a. Simulation.

1. Minimum cycle time: 0.885 ns
2. Strategy: recursive
3. Screenshot of the summary

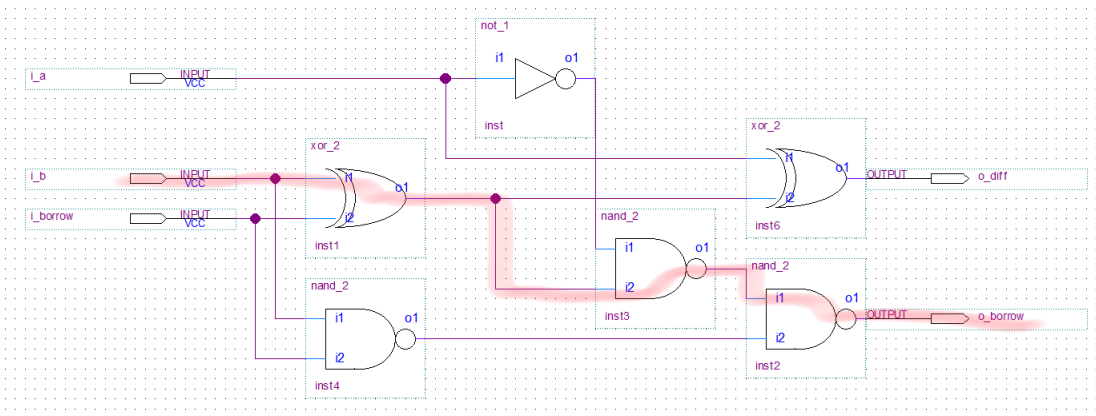
```
==================================================
                    Summary
==================================================
        Clock cycle:              0.885 ns
        Number of transistors:    1594
        Total excution cycle:     11462
        Correctness Score:        40.0
        Performance Score:        16169328.8
==================================================
```

## b. Circuit Diagram. (Red paths: critical paths)

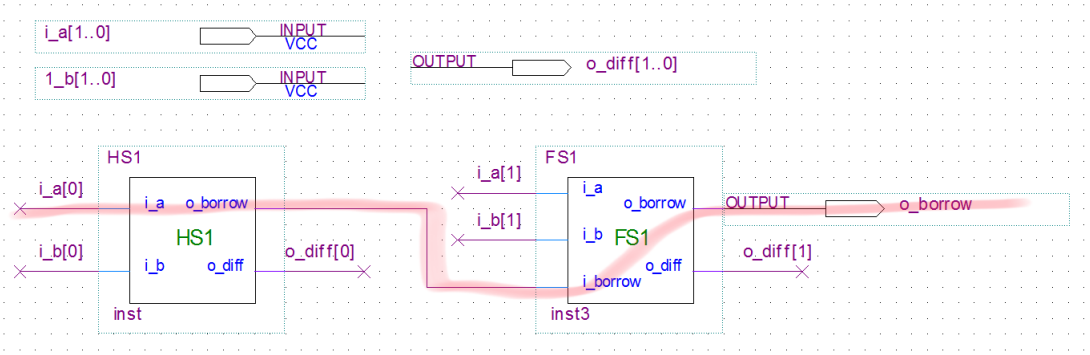
1. HS1



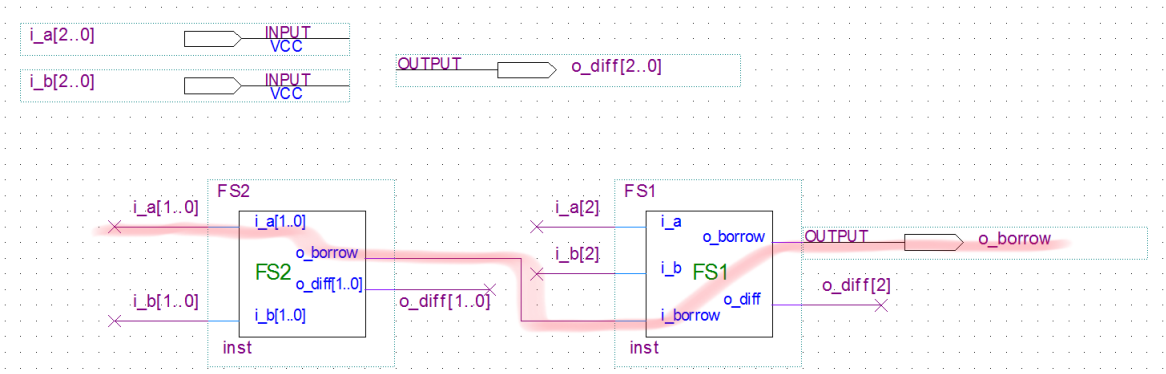

2. FS1



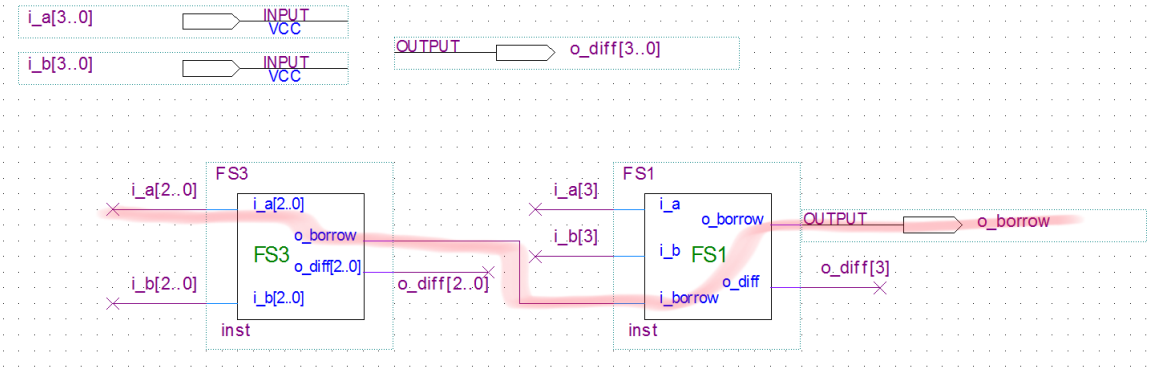
3. FS2



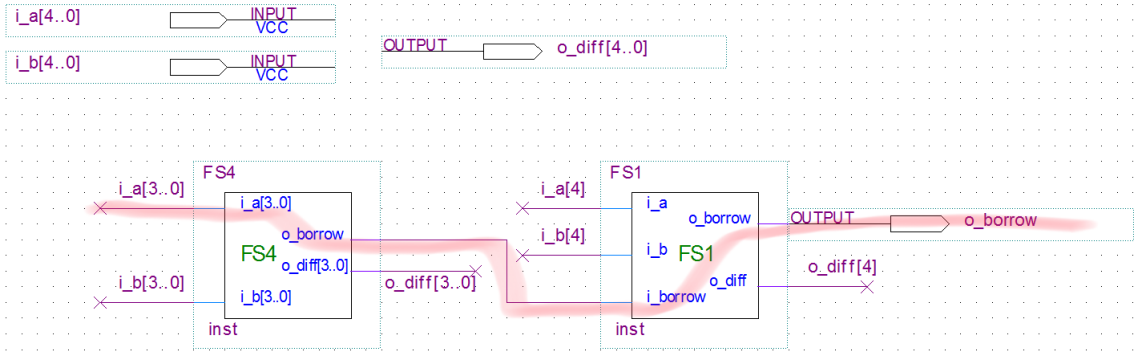

4. FS3



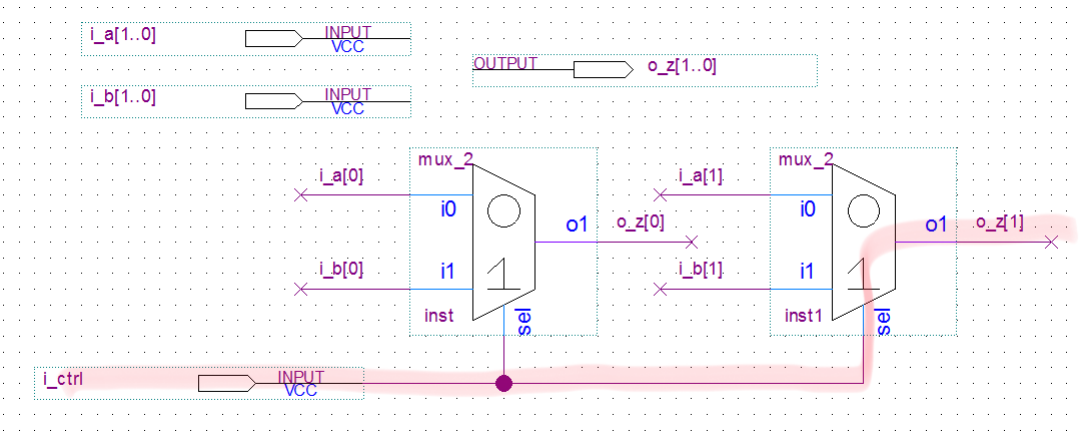

5. FS4



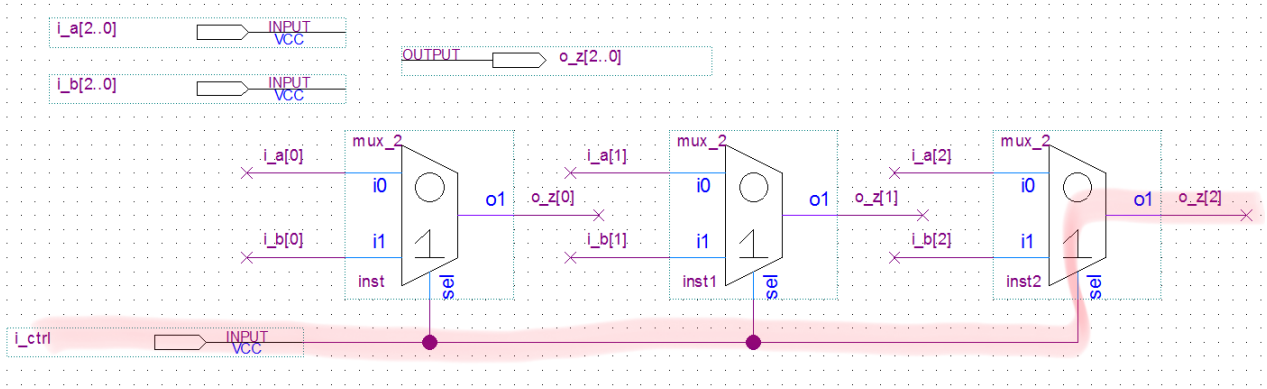

6. FS5



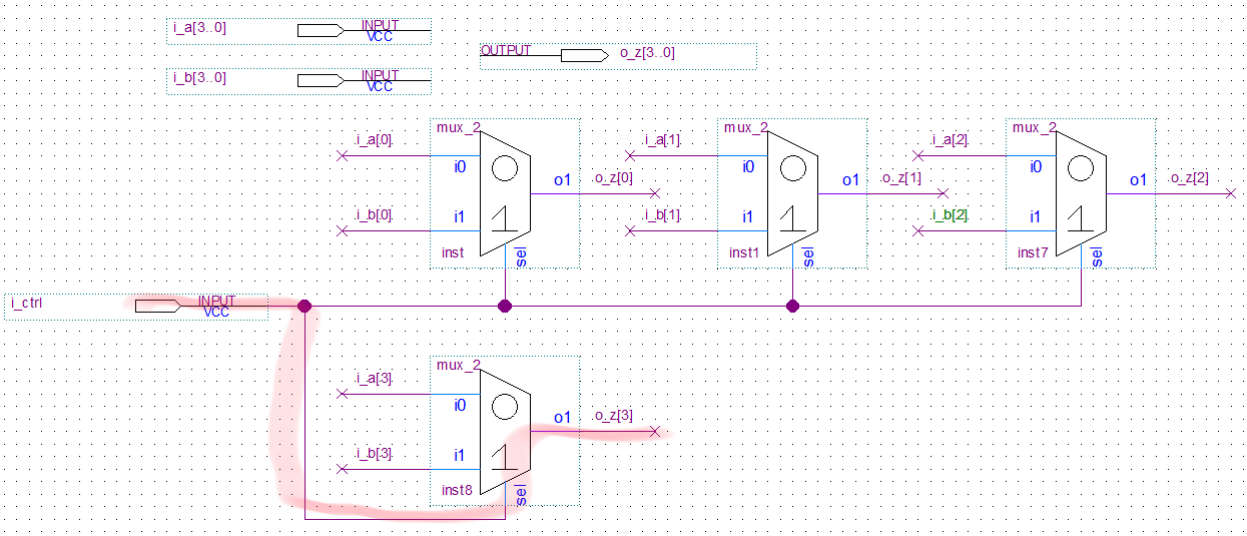
7. MUX2



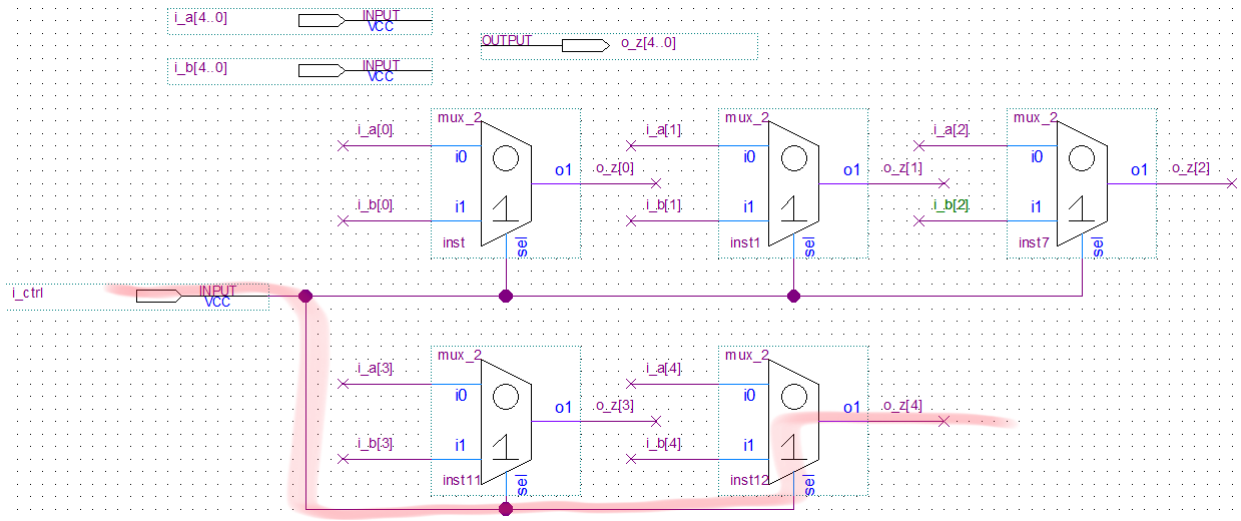

8. MUX3



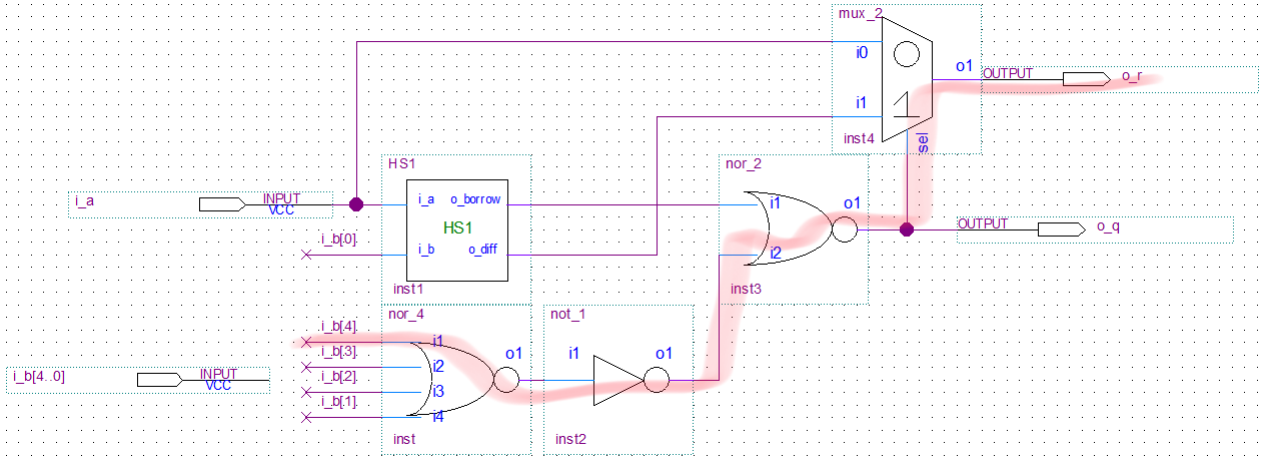

9. MUX4



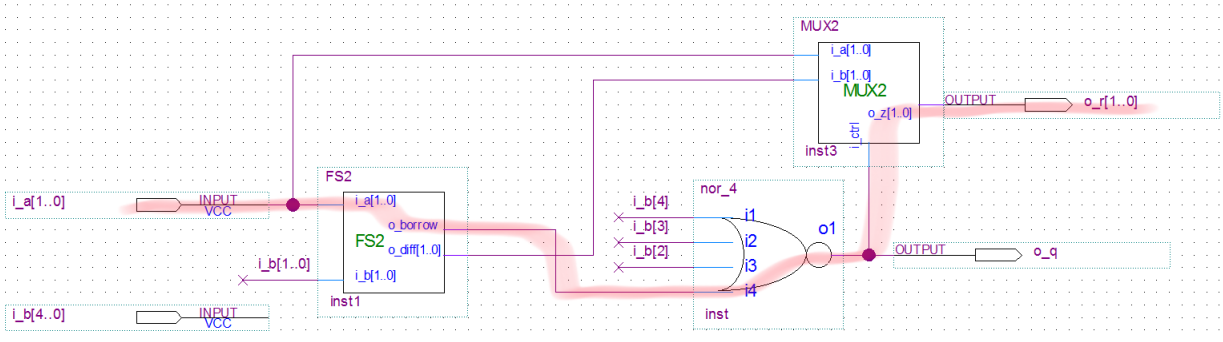
10. MUX5



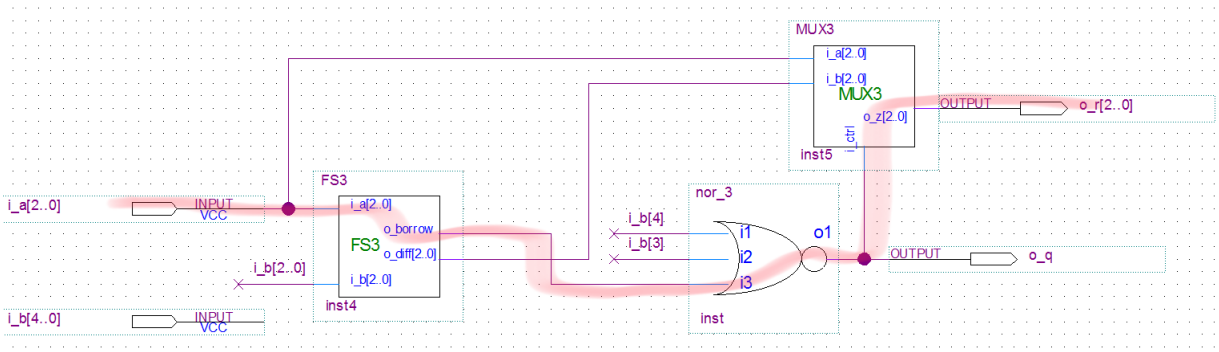

11. STAGE1



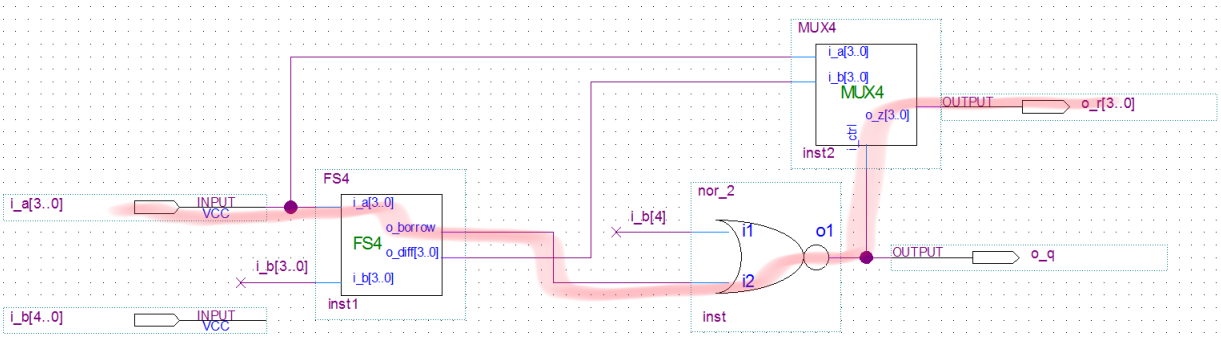

12. STAGE2



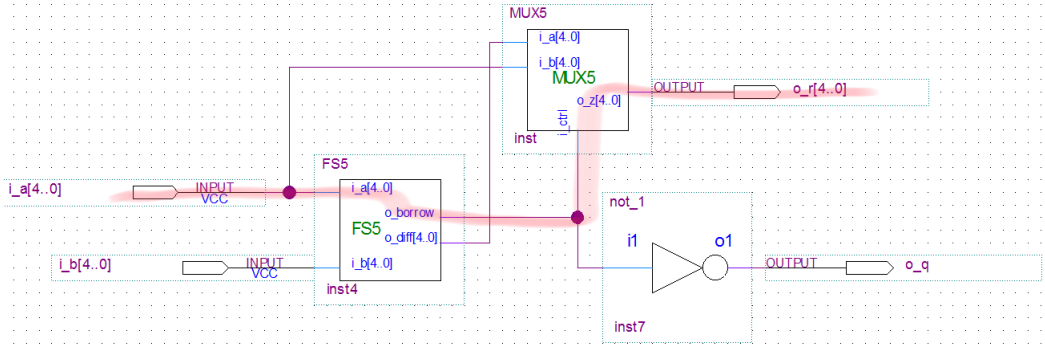
13. STAGE3



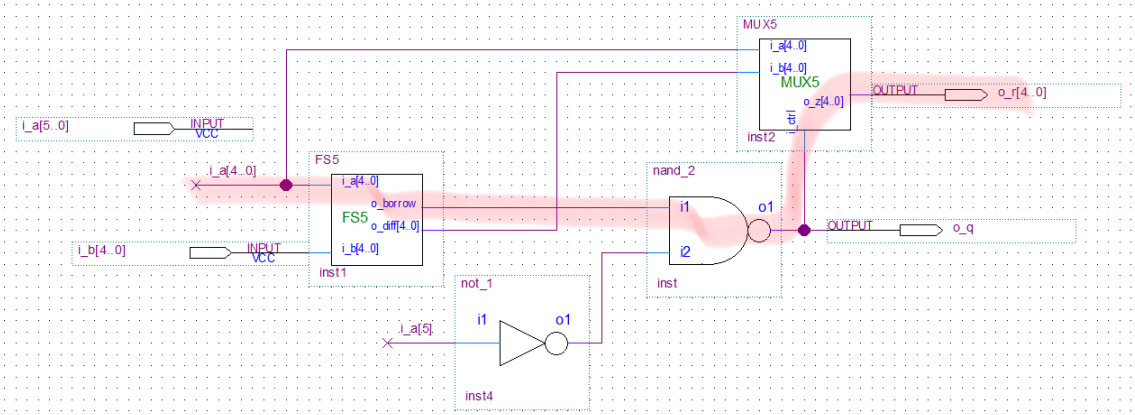

14. STAGE4



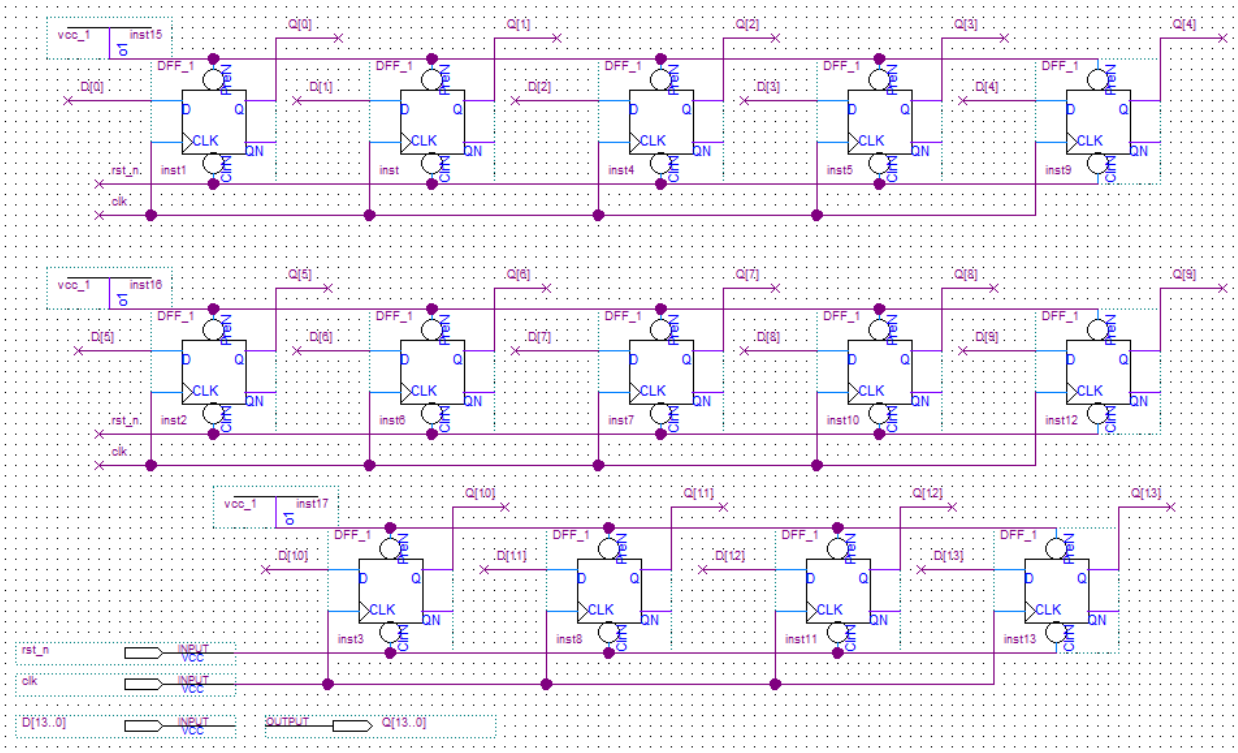

15. STAGE5

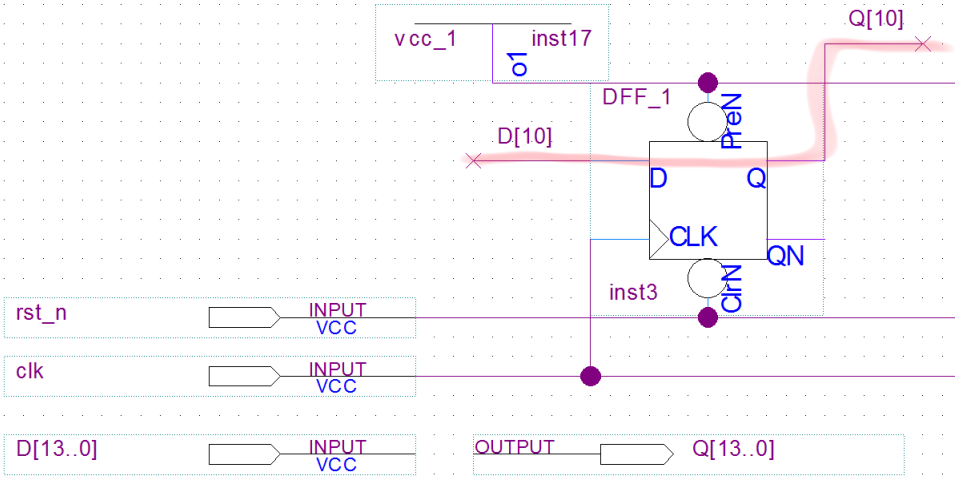



16. STAGE678

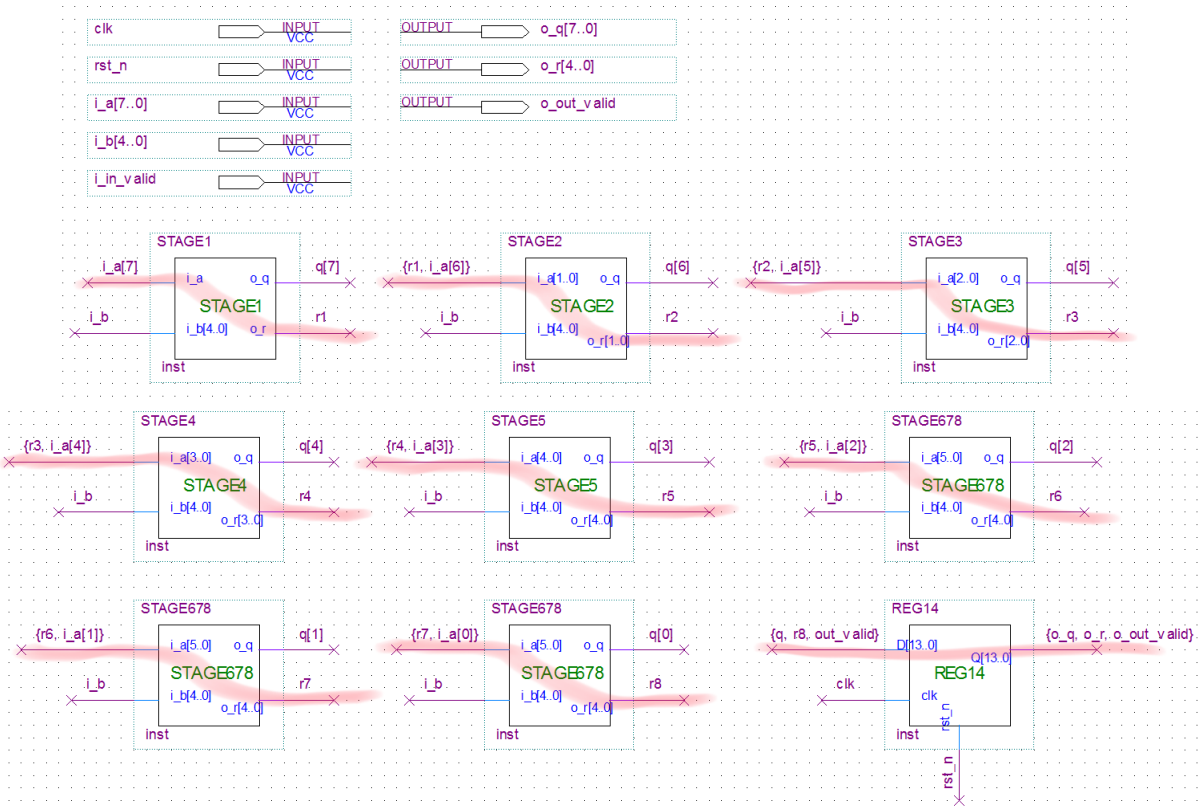


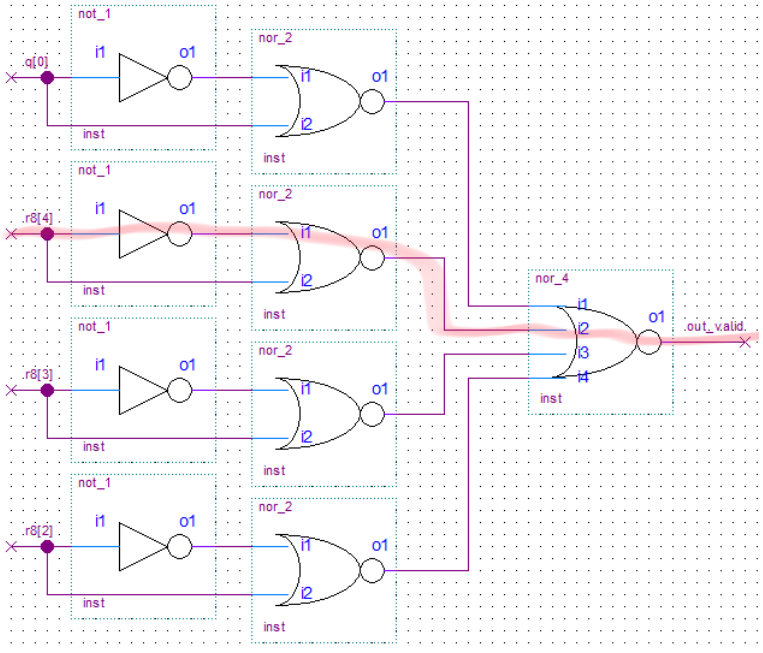
17. REG14


vcc_1
inst15
inst16
inst17
DFF_1
PreN
D
Q
CLK
QN
ClrN
rst_n
clk
D[13..0]
Q[13..0]
INPUT
OUTPUT
VCC


vcc_1
inst17
Q[10]
DFF_1
D[10]
PreN
D
Q
CLK
QN
ClrN
inst3
rst_n
clk
D[13..0]
INPUT
VCC
OUTPUT
Q[13..0]

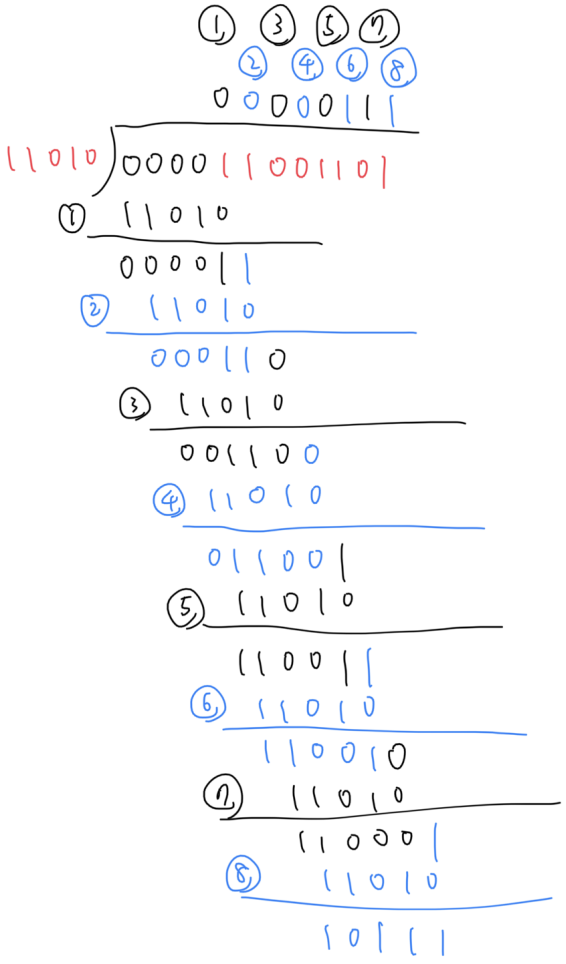
18. div


clk
rst_n
i_a[7..0]
i_b[4..0]
i_in_valid
INPUT
VCC
OUTPUT
o_q[7..0]
o_r[4..0]
o_out_valid
STAGE1
STAGE2
STAGE3
STAGE4
STAGE5
STAGE678
REG14
inst


not_1
nor_2
nor_4
i1
o1
i2
i3
i4
inst

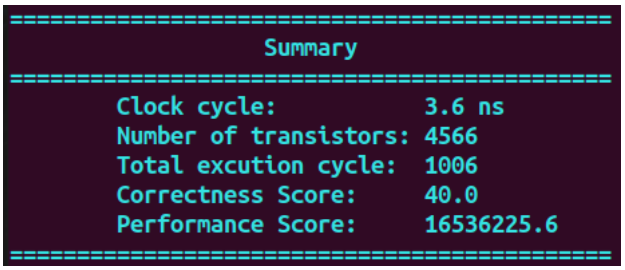
# c. Discussion.

1. **Introduce your design.**
   The working principle is the same as long division. In each stage, try subtracting i_b from the output number from previous stage. If the result is negative, output corresponding bit of o_q = 0 and pass the original output number from previous stage to next stage; otherwise, output corresponding bit of o_q = 1 and pass the subtracted result to next stage. Below is an example.



2. **How do you cut your pipelined or recursive design?**
   The way of splitting stages comes from the intuition of long division. Since I use recursive design, balancing the delay of each stage is not necessary.
3. **How do you improve your critical path and the number of transistors?**
   Take stage 1 for example. We want to subtract $y = 11010$ from $x = 00001$. Note that the first 4 bits of $x$ are all 0. Thus, if one of the first 4 bits of $y$ is 1, then the subtracted result must be negative. This can be implemented using only one NR4 gate. 5 full subtractors (FS1) are avoided. The same trick is also used in stages 2, 3, and 4.
   Also, it seems like 6 FS1 are required in each stage to perform $x - y$. However, the 6-th bit of subtracted result must be 0, so we don't really need the 6-th FS1. We can just use the 6-th bit of $x$, the output borrow port of FS5 and an ND2 gate to detect negative subtraction result.

4. **How do you trade-off between area and speed?**
   The implementation of FS1 is a good example. Instead of using EO3 to compute output diff port, two EO2's are used. Also, the output borrow port is calculated from the result of one EO2. This implemntation takes fewer transistors but results in huge delay on the ripple path.
5. **Compare with other architectures you have designed (if any).**
   Below is the summary of pipelined design.




   Additional transistors are required for the pipelined registers. Number of cycles are approximately the same as the number of testcases. Cycle time is limited to the delay of the longest pipeline stage.