IC Design HW4 Report

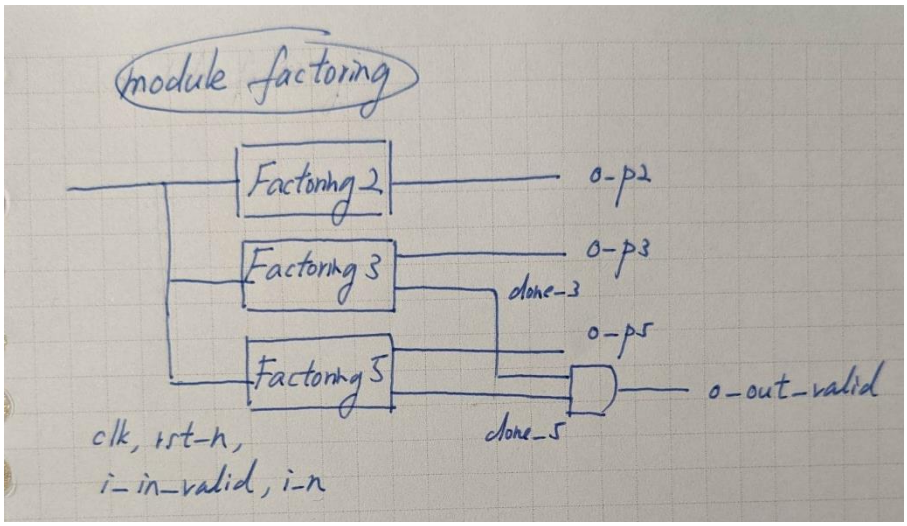
電機三 b08901201 胡雅晴

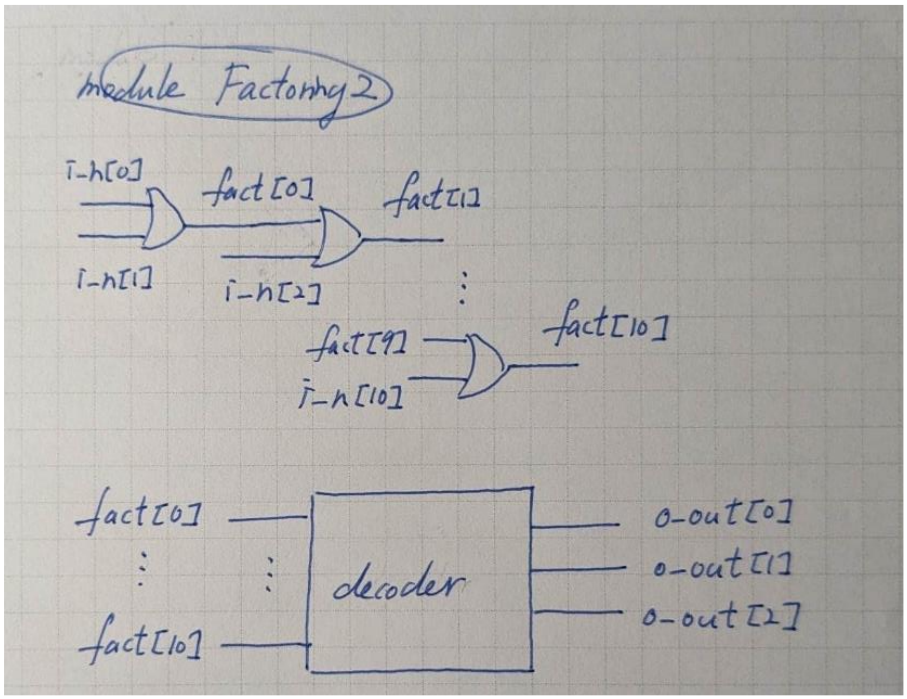
I. Simulation

```
============================================
                 Summary
============================================
       Clock cycle:           10.0 ns
       Number of transistors: 36778
       Total excution cycle:  95829
       Correctness Score:     36.1
       Performance Score:     35243989620.0
============================================
```

II. Circuit diagram

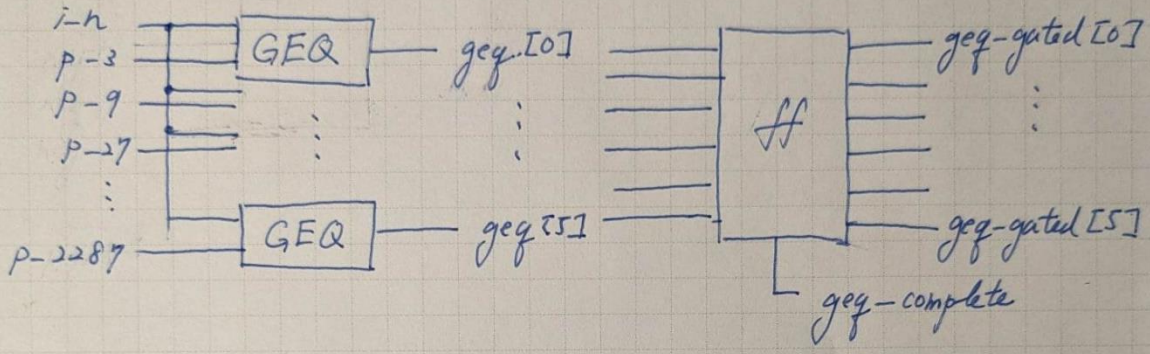
1. Plot the gate-level circuit diagram of your design.

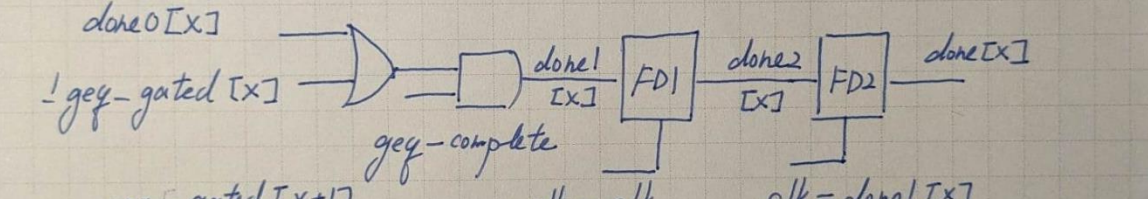


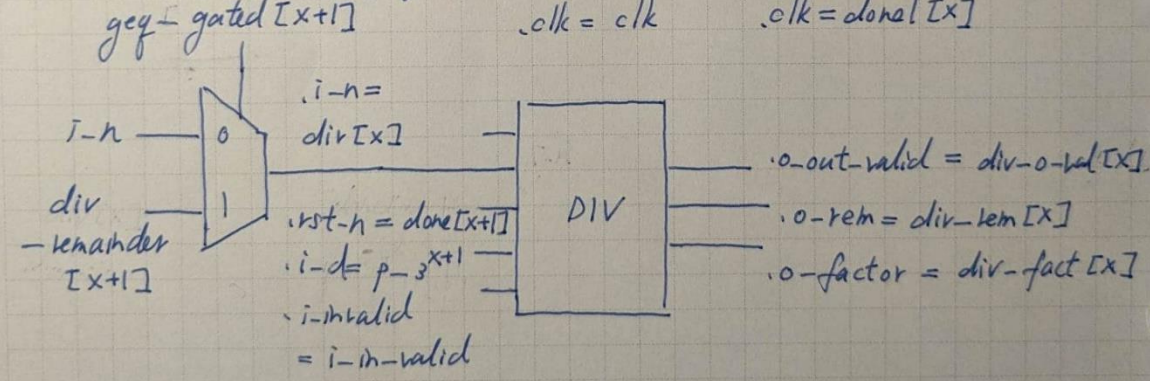


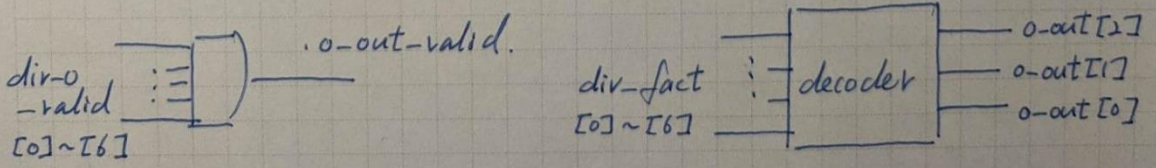
module Factoring 3 (5)

※ Factoring 5 is of similar structure, the only difference is with the number of wires and factor power numbers.

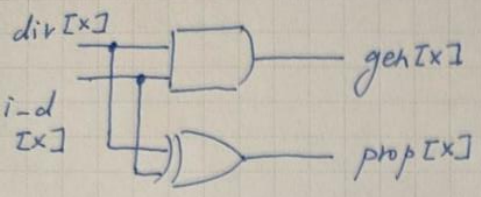


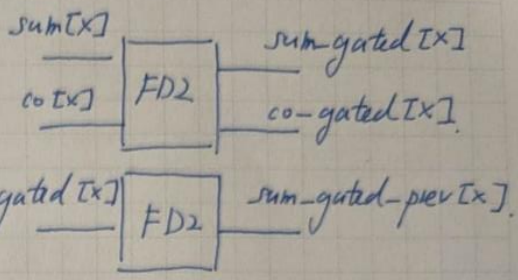


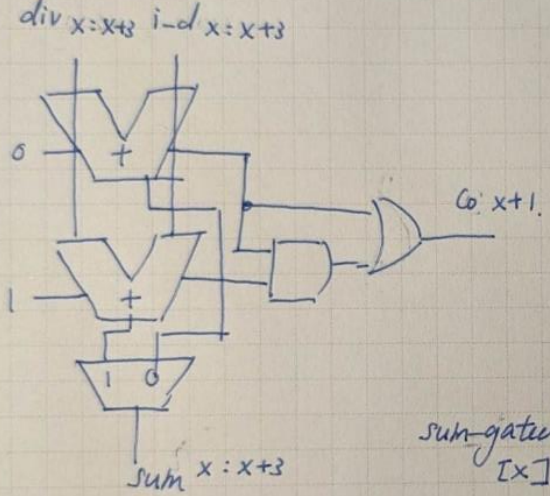




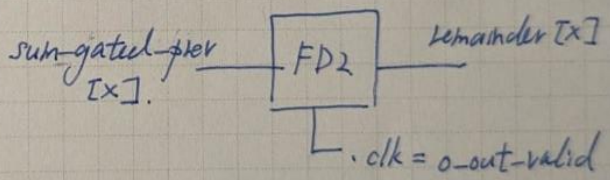
module DIV

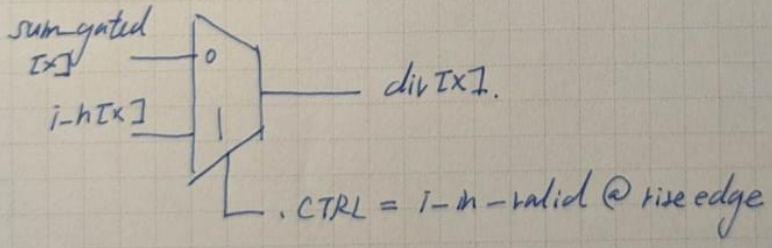


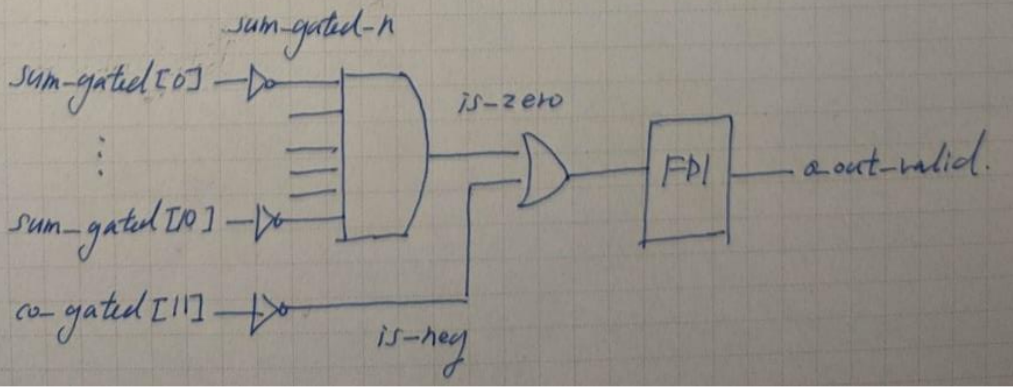




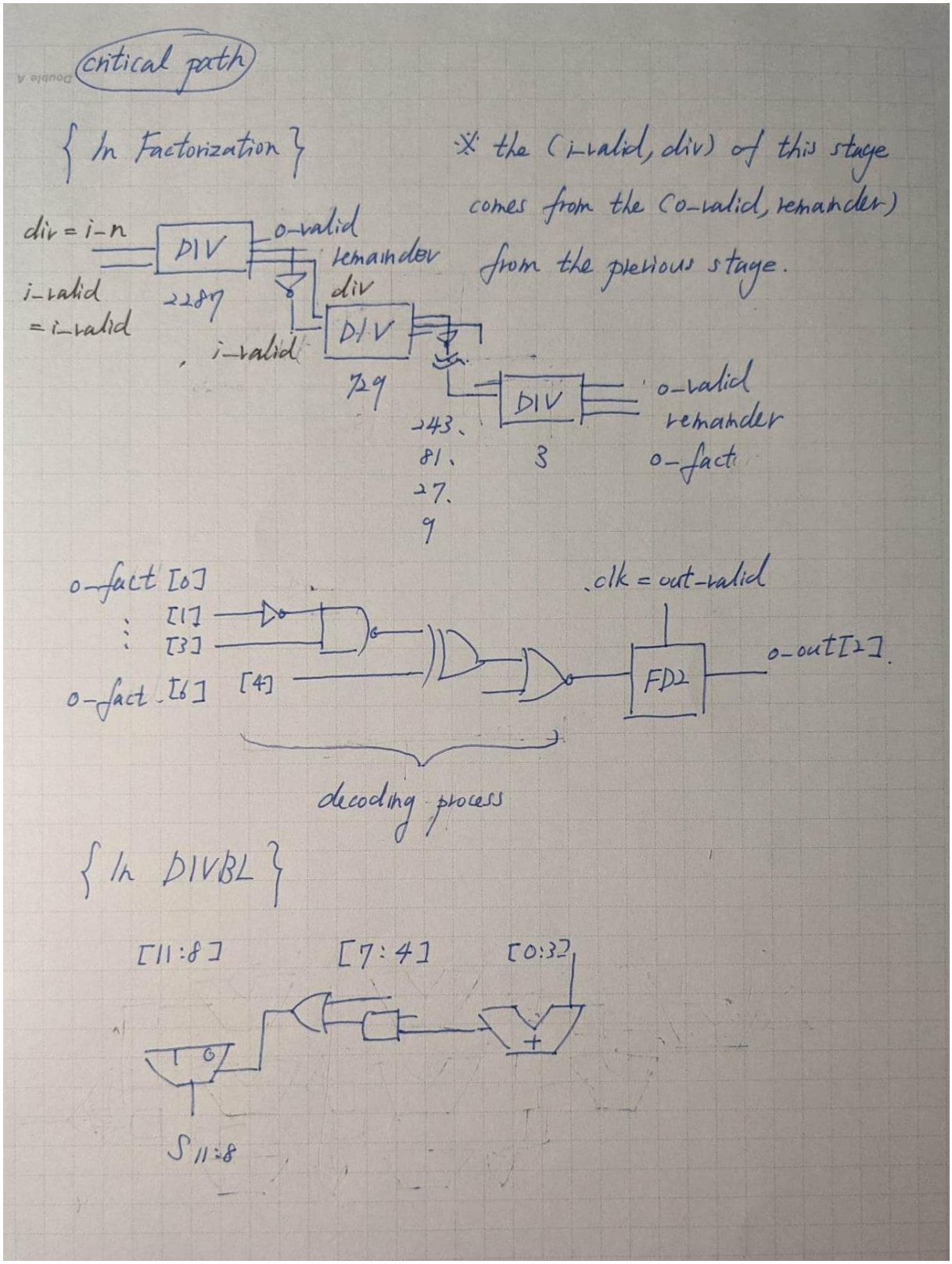
* the twelve digits are grouped in four; they are summed using the carry select adder as left except for group x=0.







2. Plot critical path on the diagram above.



III. Discussion

1. Introduce your design.

(1) Factorization

The factorization method used in this circuit was inspired by the roll-and-divide method. Modules were built explicitly for all 2, 3, and 5 factors. Each module contains multiple divisors, each corresponding to some power of the factor. For example, the Factor5 module contains modules Div5, Div25, Div125, Div625, and Div3125.

The division models send an o_valid signal and a remain signal to the lower levels once the dividend is smaller or equal to the divisor. The lower levels then take on the remain signal as its divisor and continue the operation.

(2) Division and Subtraction

The division was done by repetitive subtractions. Modules return the o_valid signal once its quotient signal is negative. Subtraction was implemented by adding the 2's complement of subtrahend to the minuend. Since subtraction is on the critical path and would be performed repetitiously, the designer trade off area for speed and adopted the carry-skip adder.

2. How do you cut your pipelined or recursive design?

The circuit was designed using the recursive input strategy. Stages were mainly split for maintaining the correctness of the circuit. For example, bits of addition results must be gated to prevent glitches in the output sum.

3. How do you improve your critical path and number of transistors?

(1) Adopting efficient algorithms and designs. See problem 5. for more.
(2) Trading ANDs and ORs with NORs and NANDs whenever possible. They cost less delay and transistors.

4. How do you trade-off between area and speed?

The choice between area or speed is determined by whether the part of the circuit is on critical path or not. Decreasing the speed of the critical path decreases the speed of the whole circuit, while it is unlikely to make a significant difference in the number of transistors since the circuit is large. Therefore, on critical paths, improving speed should be the priority. If the part of the circuit is not on critical path, decreasing speed wouldn't make a difference, thus decreasing the area should come first.

5. Compare with other architectures you have designed (if any).

(1) Parallel Division vs Roll-and-divide

While it is the designer's intuitive choice to perform parallel division on all possible powers of factors, she soon found out that this wouldn't work for the time limit. That is, dividing X^n by X^1, X^2, … , X^n in parallel, and compute o_pX when division by X^1 is done. Comparatively, the roll-and-divide method costs much less delay despites its complicacy. To find the powers of prime factor X by X^n, it takes X^{n-1} cycles by parallel division, and n-1 cycles by roll-and-divide.

(2) Specialization for Factoring2

It is likely that Factoring2 would take more time than Factoring3 and Factoring5. Improving the speed of Factoring2 would benefit the speed of the circuit as a whole. Since numbers are presented in binary in circuits, Factoring2 is a special case that could be sped up by just reviewing the number digits.

(3) Specialization for Div3265 and Div2287

For i_n of 12 digits, the maximum number that could be reached is 4095, <3265*2 and <2287*2. While Div3265 and Div2287 takes little time, they are on the critical path and could be sped up by performing plain comparison. The designer specialized a different model for these two numbers.