Carry look ahead adder

Why another adder when ripple-carry-adder is there

As we have implemented ripple carry adder, we saw that carry-in of each one bit full adder is generated by the previous one bit adder. This causes a lot of delay especially when we increase the number of bits. To overcome this problem, a new type of adder i.e. Carry look ahead adder was needed.

Carry look ahead Adder does not wait for intermediate carries to be generated because it generates the carry-in for each bit simultaneously.

Working

From the following truth table, we can get an idea of how we can design it.

a	b	cin	cout
0	0	0	0
0	0	1	0
0	1	0	0
1	0	0	0
0	1	1	1
1	0	1	1
1	1	0	1
1	1	1	1

From this, we can conclude that

cout= (a & b) + ((a ^ b) & cin)

For our simplicity, lets define

- g = a & b
- p = a ^ b

Now for an 8 bit adder, we can take a 8 bit carry-in register and assign it values using the above designed logic.

Let's say it is carries[8:0] where carries[i] will be carry-in for ith bit of inputs i.e a and b.

Then,

carries[0] will be carry-in itself

carries[1] = g[0]+(p[0]&carries[0]) and we can substitute for carries[0]
So,
carries[1]= g[0]+(p[0]&cin)

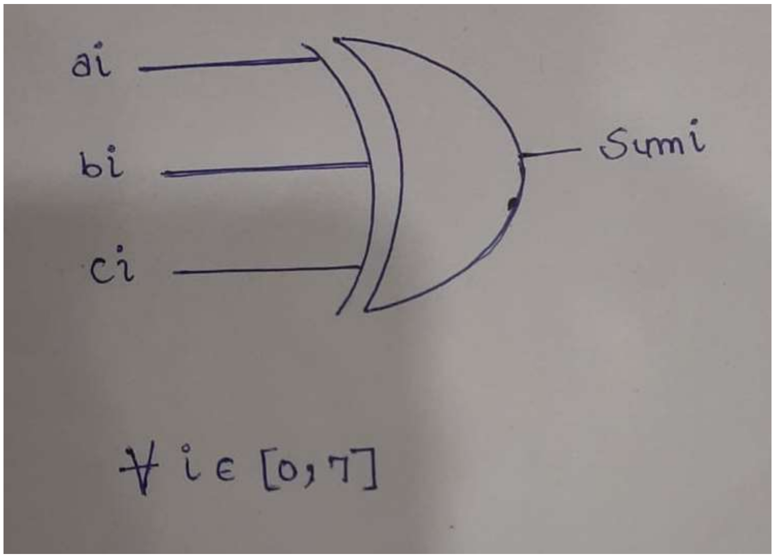
carries[2] = g[1]+(p[1]&carries[1]) , again we can substitute for carries[1]
So,
carries[2] = g[1]+(p[1]&(g[0]+(p[0]&cin)))

And so on till carries[7].

USing this method, we can also find the carry-out.

From this we can see intermediate carries can be calculated with no such propagation delay as it depends only on:
i) Bits being added
ii) Initial Carry-in

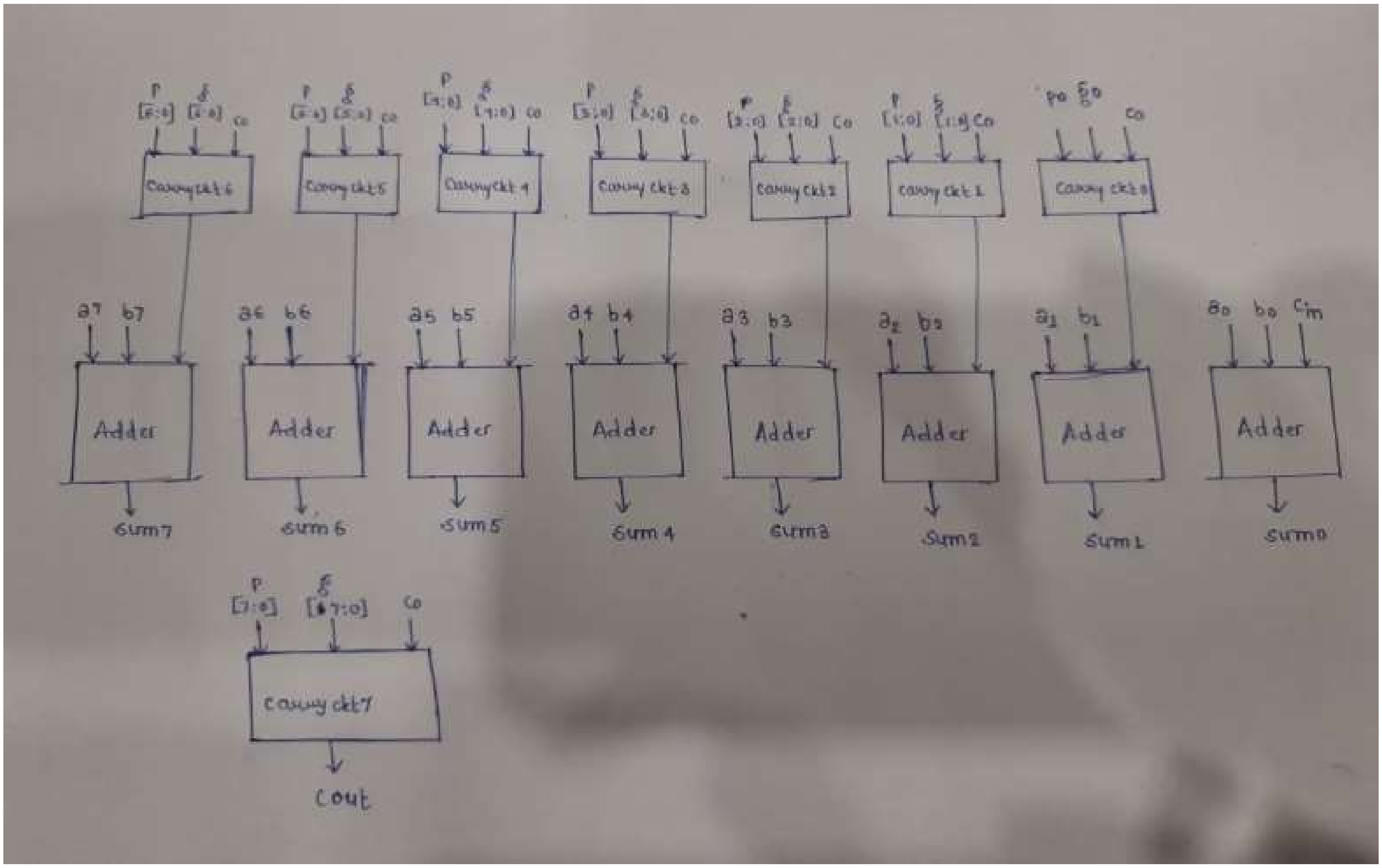
Then we can easily get the sum by using xor gates as described below:



sum[i]=a[i]^b[i]^carries[i]

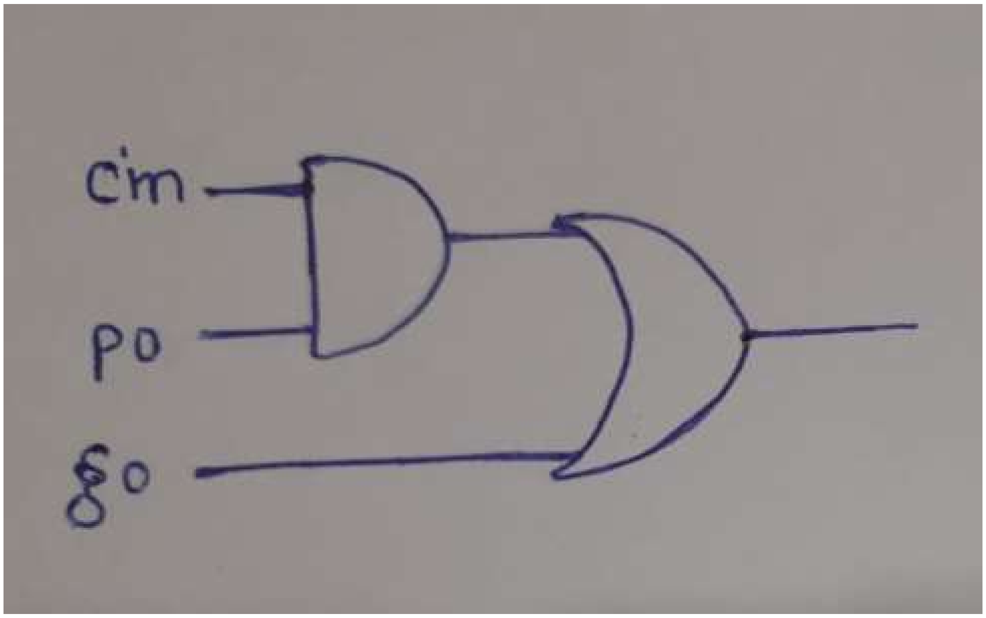
Let's also talk about some of the drawbacks of this adder:

- It gets more and more complex as we increase the number of bits.
- It involves more hardware as compared to ripple carry adder, so it is bit costly.

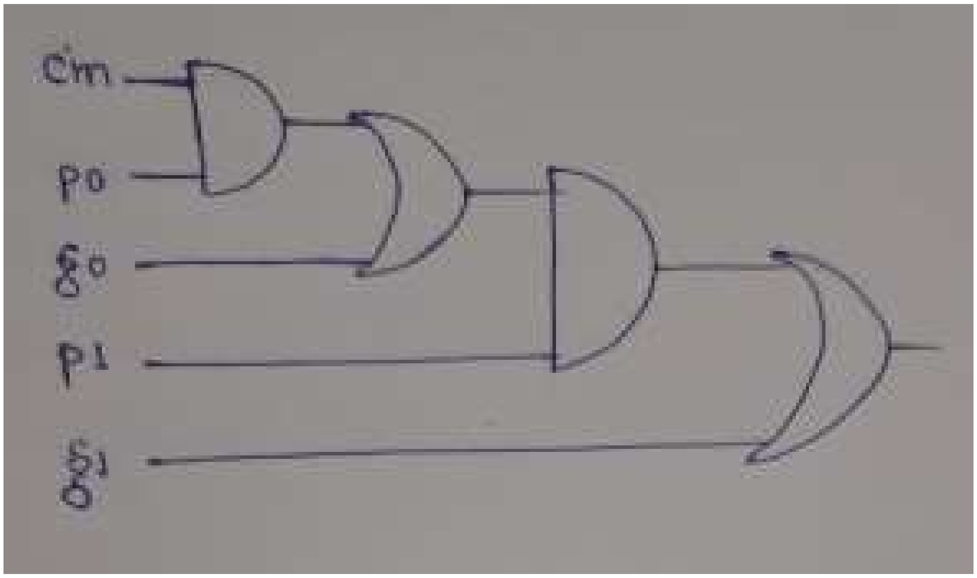


Circuit diagram

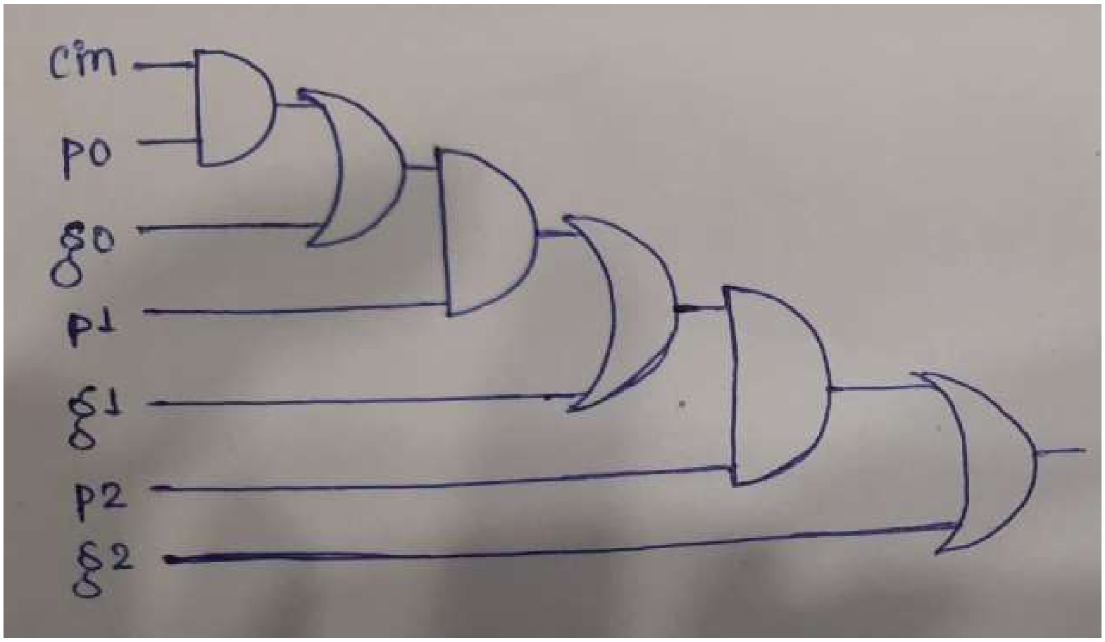
Gate level circuit diagram of different components are shown below



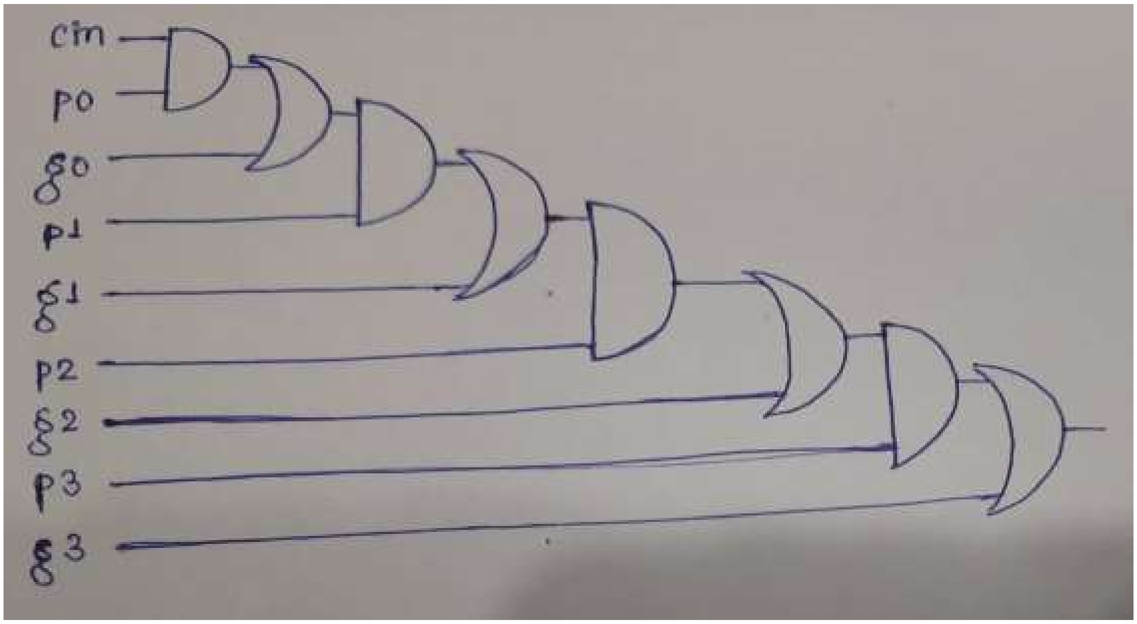
Implementation of Carry ckt 1



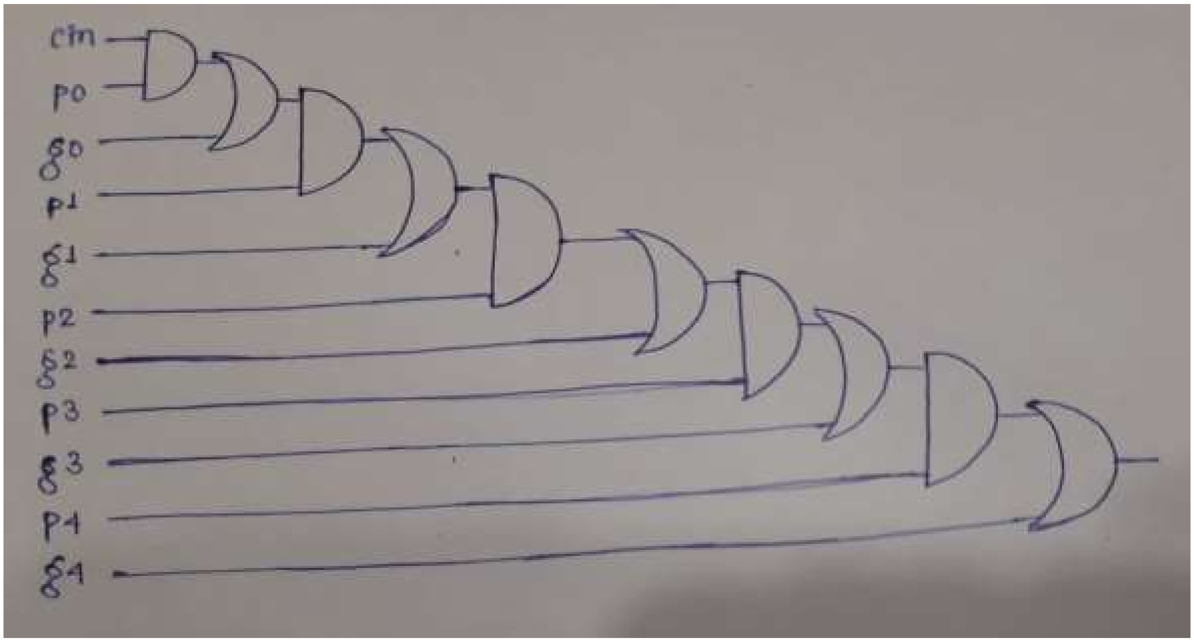
Implementation of Carry ckt 2



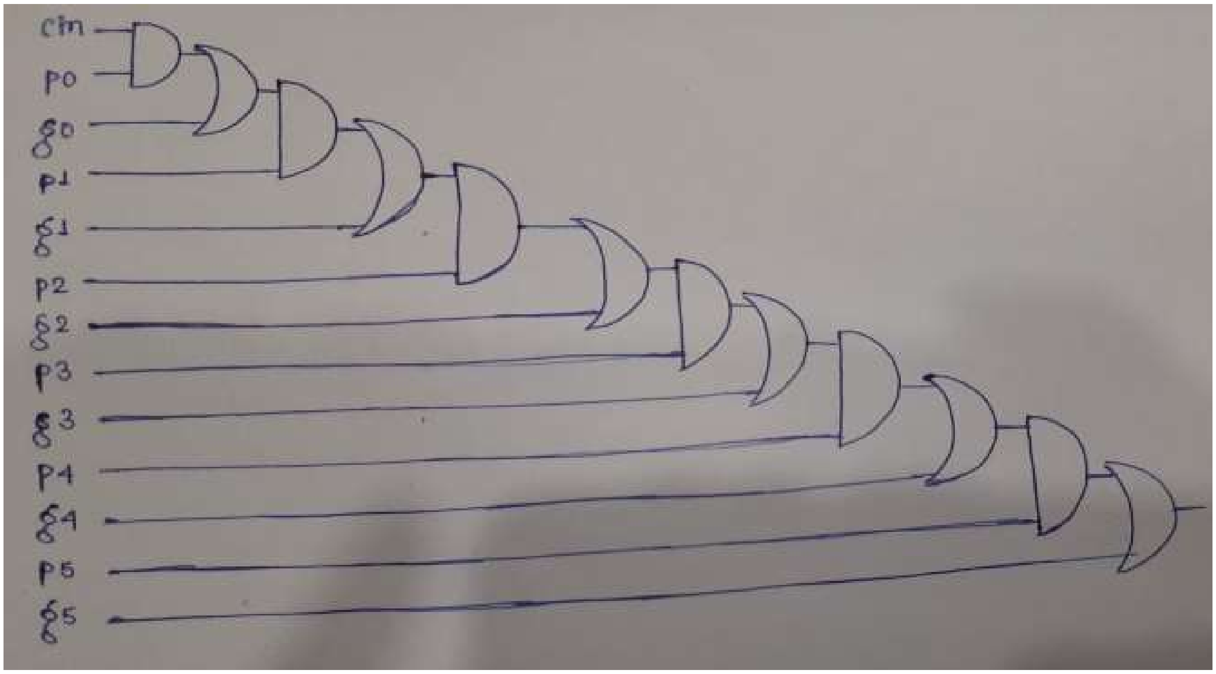
Implementation of carry ckt 3



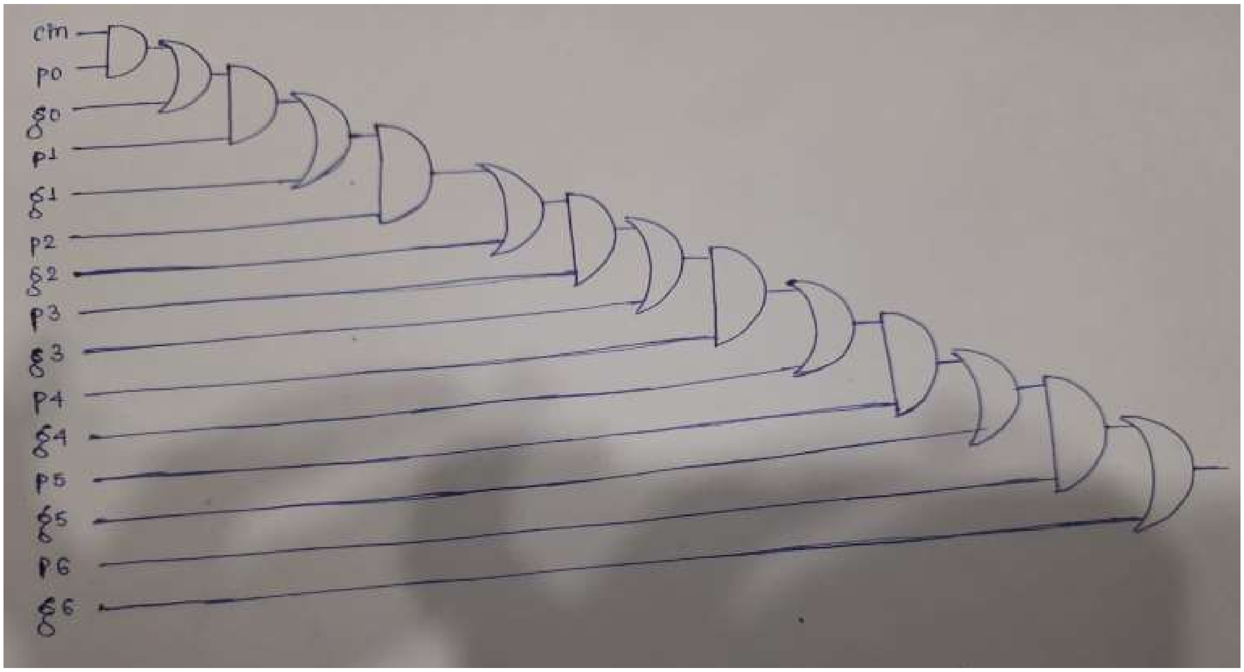
Implementation of carry ckt 4



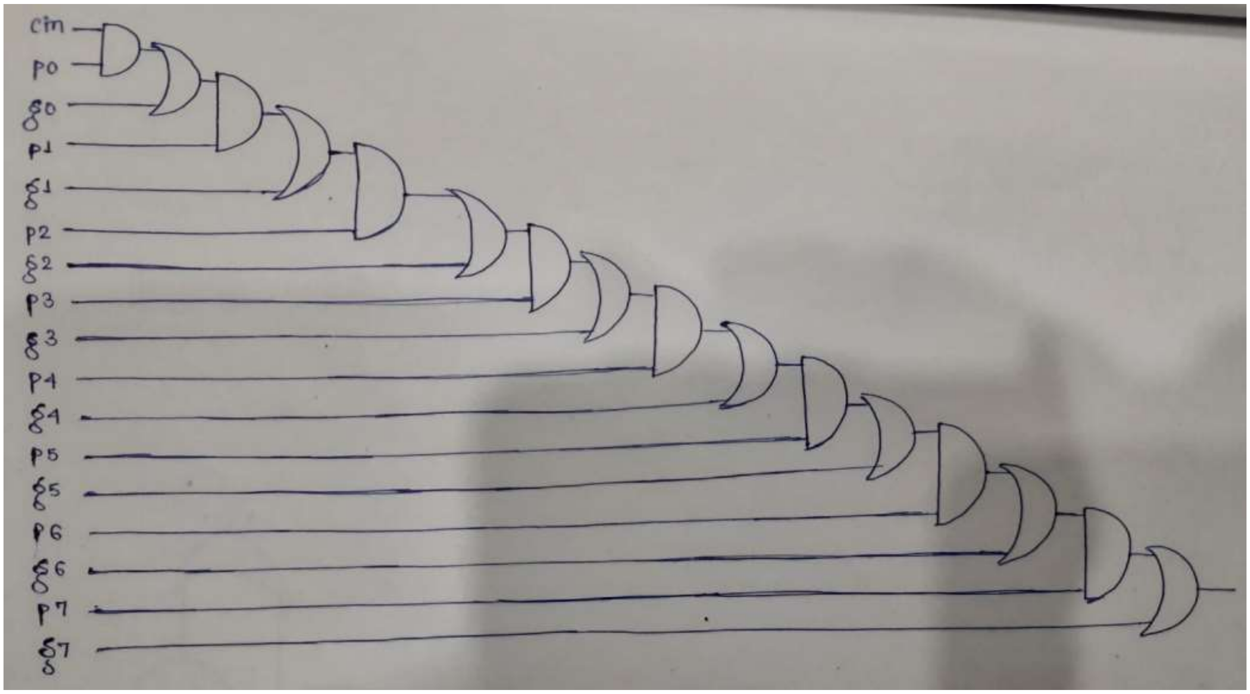
Implementation of carry ckt 5



Implementation of carry ckt 6



Implementation of carry ckt 7



Implementation of carry-out