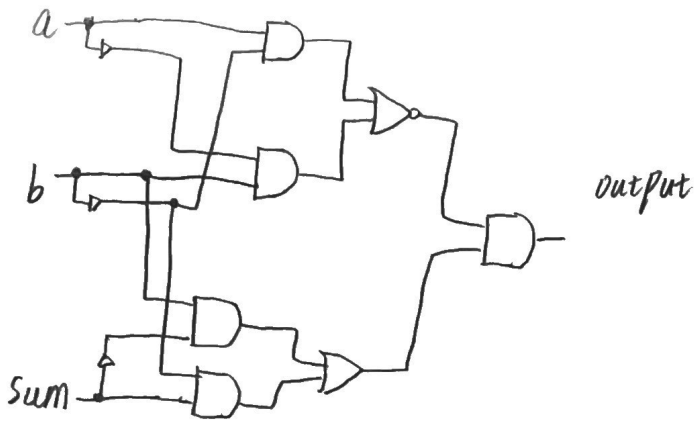
Mingkuan Pang

1.

Overflow diagram



2. There are four cases for the possibility of combine of sign_bit.

for case 1,0 or 0,1 the carry in and carry out always stay the same.

$$\begin{array}{r} {\scriptstyle 0\ \nwarrow 1\ \swarrow 0} \\ +\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} {\scriptstyle 1\ \nwarrow 1\ \swarrow 1} \\ +\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} {\scriptstyle R\ 0\ \swarrow 0} \\ 1 \\ \hline 1 \end{array} \qquad \begin{array}{r} {\scriptstyle \nwarrow 1\ 0\ \swarrow 1} \\ 1 \\ \hline 0 \end{array}$$

Thus, if exists carry in is different with carry out, the sign_bit must be the same: 0,0 case and 1,1 case.

For 0,0 case:

$$\begin{array}{r} {\scriptstyle R\ 0\ \swarrow 0} \\ +\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} {\scriptstyle 0\ \nwarrow\ \swarrow 1} \\ +\ 0 \\ \hline 1 \end{array}$$

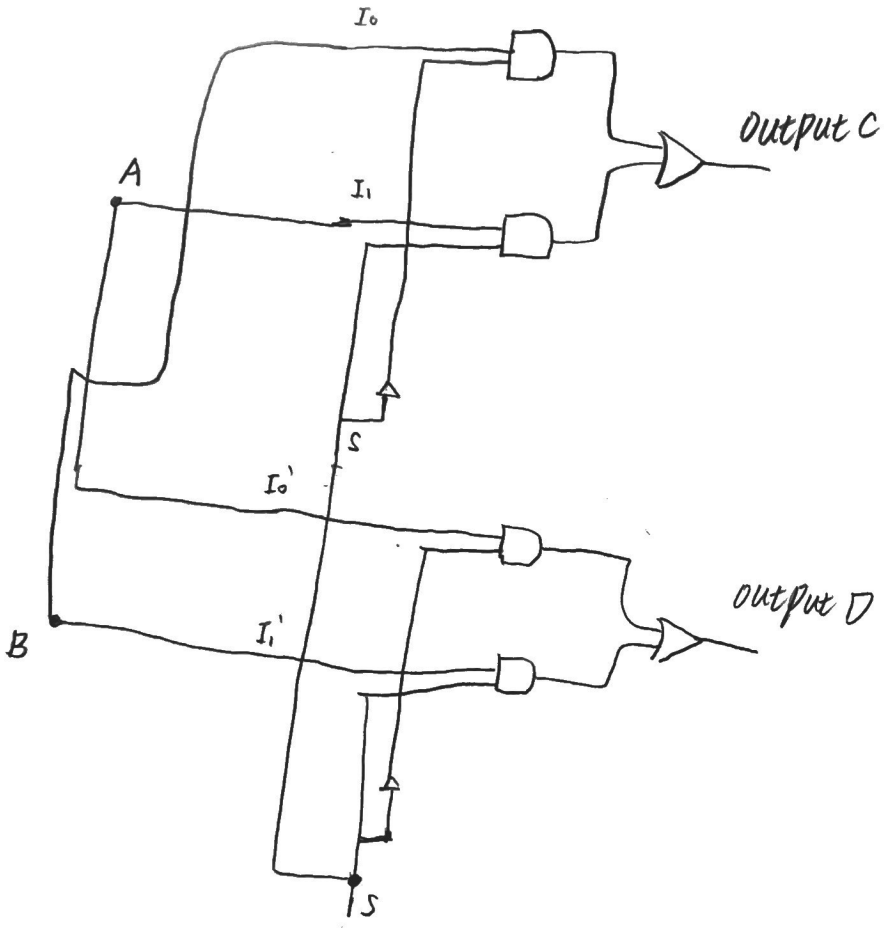
For 1,1 case:

$$\begin{array}{r} {\scriptstyle 1\ \nwarrow 1\ \swarrow 0} \\ +\ 1 \\ \hline 0 \end{array} \qquad \begin{array}{r} {\scriptstyle \nwarrow 1\ \swarrow 1} \\ +\ 1 \\ \hline 1 \end{array}$$

we can see. when carry_in_sign_bit != carry_out_sign_bit, the sign of sum will be different from the sign of operators, and overflow will happen. Therefore, when carry_in_sign_bit != carry_out_sign_bit. operation signs will be equal and sign of sum will be different from sign of operations. They have the same condition for overflow.

3.

S	I
0	I_0
1	I_1



4.

$$Speedup = \frac{1}{(1-FracEnh) + \frac{FracEnh}{SpeedupEnh}}$$

In this case here, Fraction Enhance is 25%

Thus speedup $= \frac{1}{(1-0.25) + \frac{FracEnh}{SpeedupEnh}} = \frac{1}{0.75 + \frac{FracEnh}{SpeedupEnh}}$

In the best case $\frac{FracEnh}{SpeedupEnh} \approx 0 \rightarrow \lim_{SpeedupEnh \to \infty} \frac{FracEnh}{SpeedupEnh} = 0$

Thus, the best we can get for speedup is $\frac{1}{0.75} \approx 1.33$.

However, $1.7 > 1.33$. Therefore, that is impossible to speedup of 1.7 to the whole program by just enhencing floating point instruction.