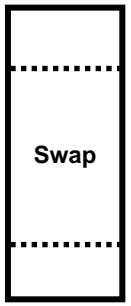
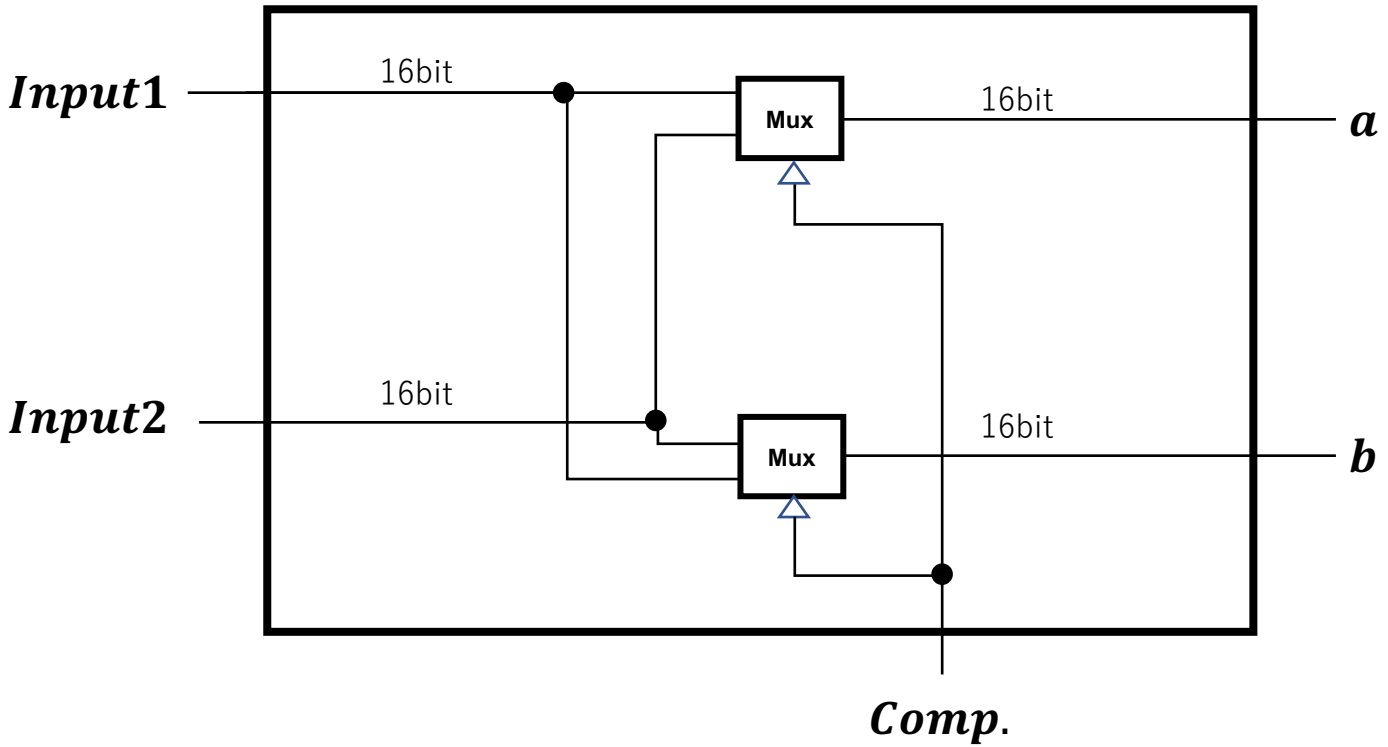
BF16Adder論理回路実装詳細

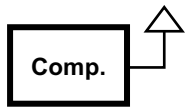
Swap

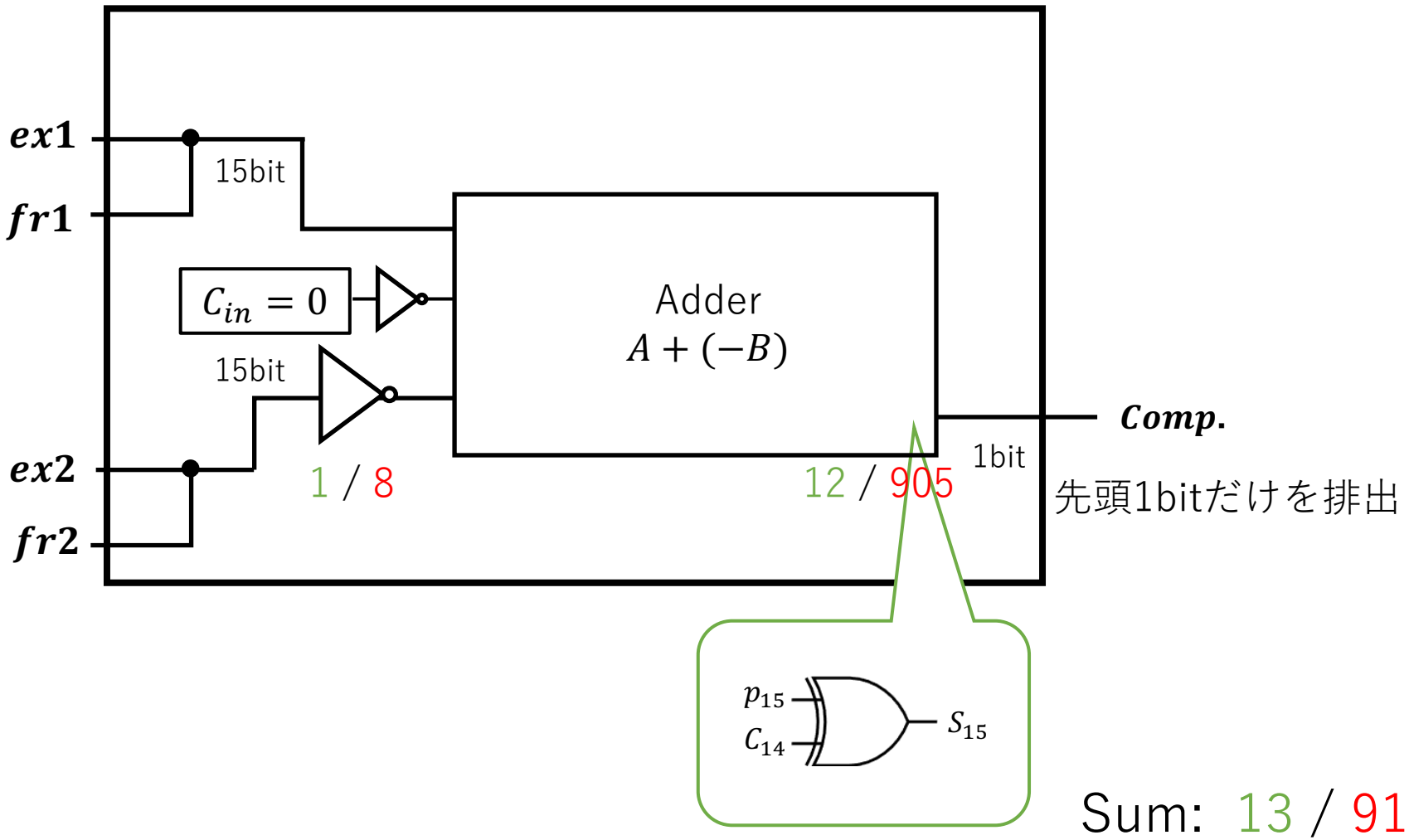


$Comp. = 1$ でSwapさせる。この時、Input1 がbに行き、Input2がaに行く。



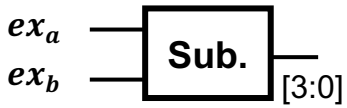
Sum: 2 / 96

比較器(Comp.) 

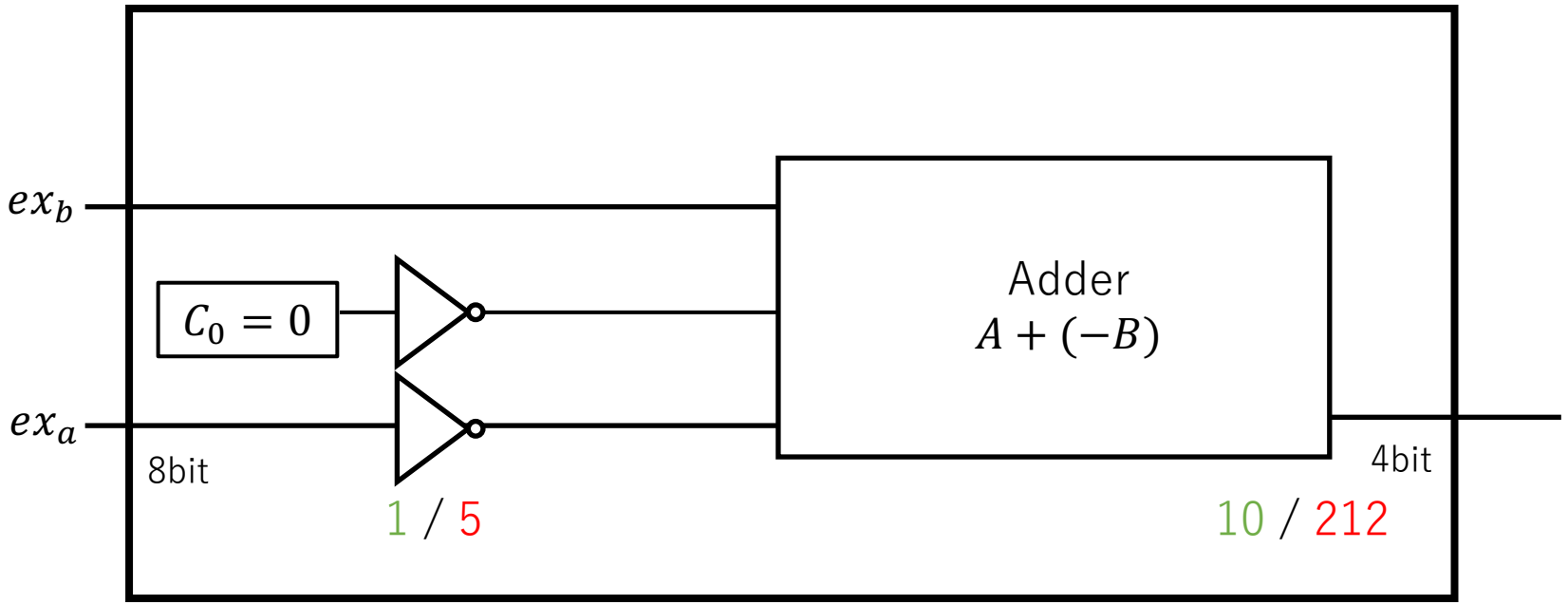


Sum: 13 / 913

Sub.

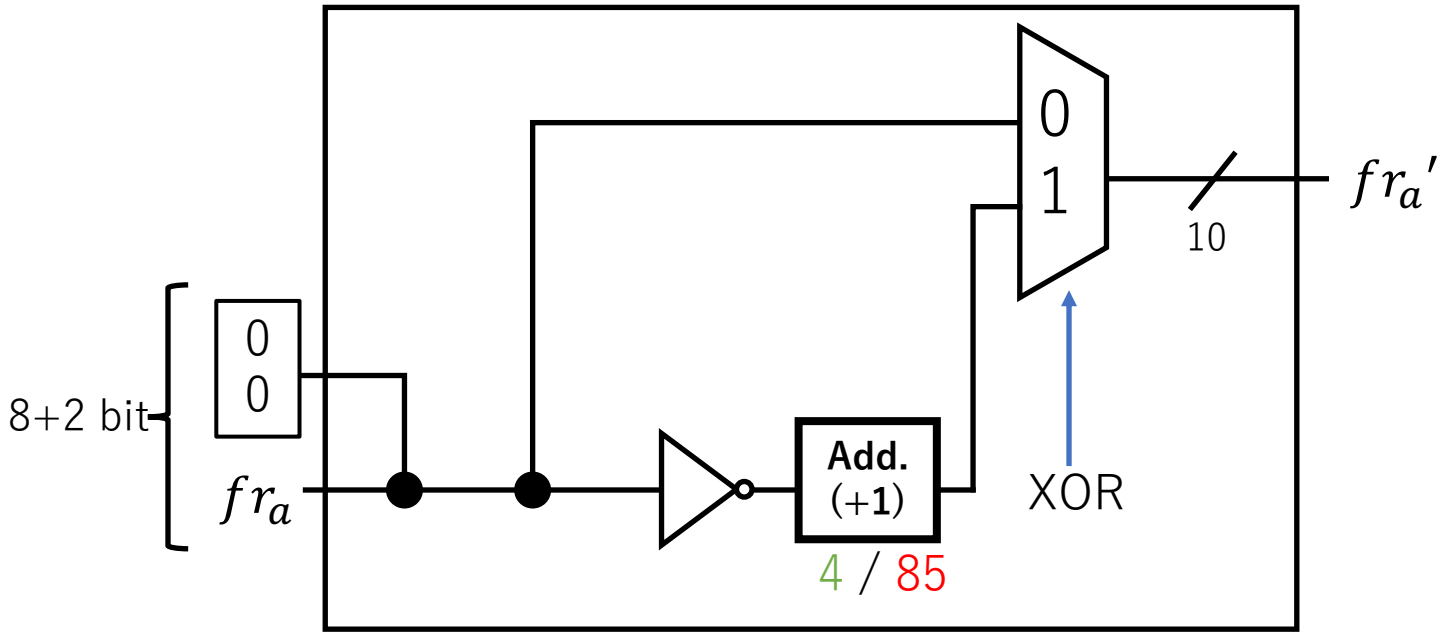


- (to Cmpl. contains “only NOT gate”)



Sum: 11 / 217

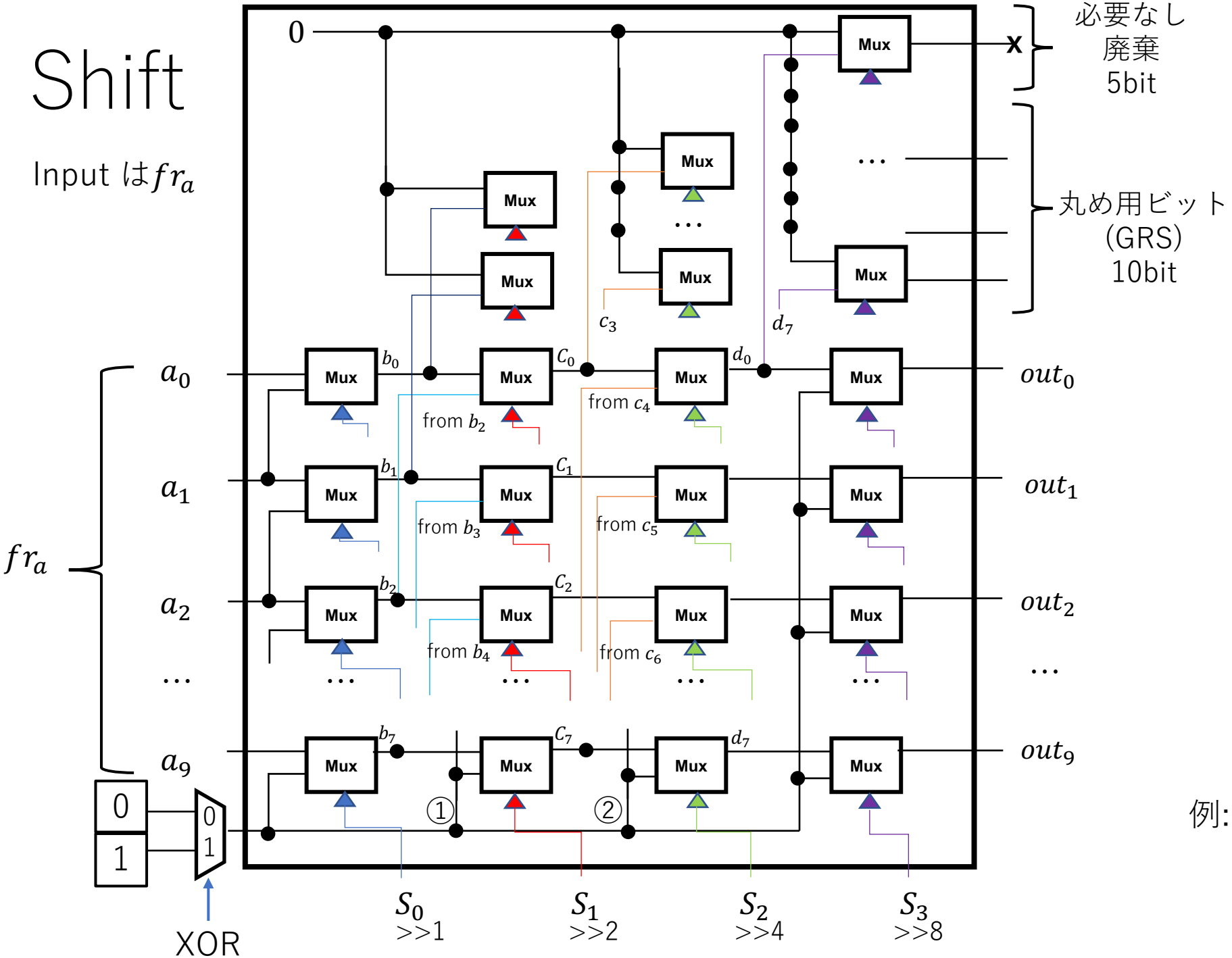
to Cmpl. (with Mux)



Sum: 6 / 123

Shift

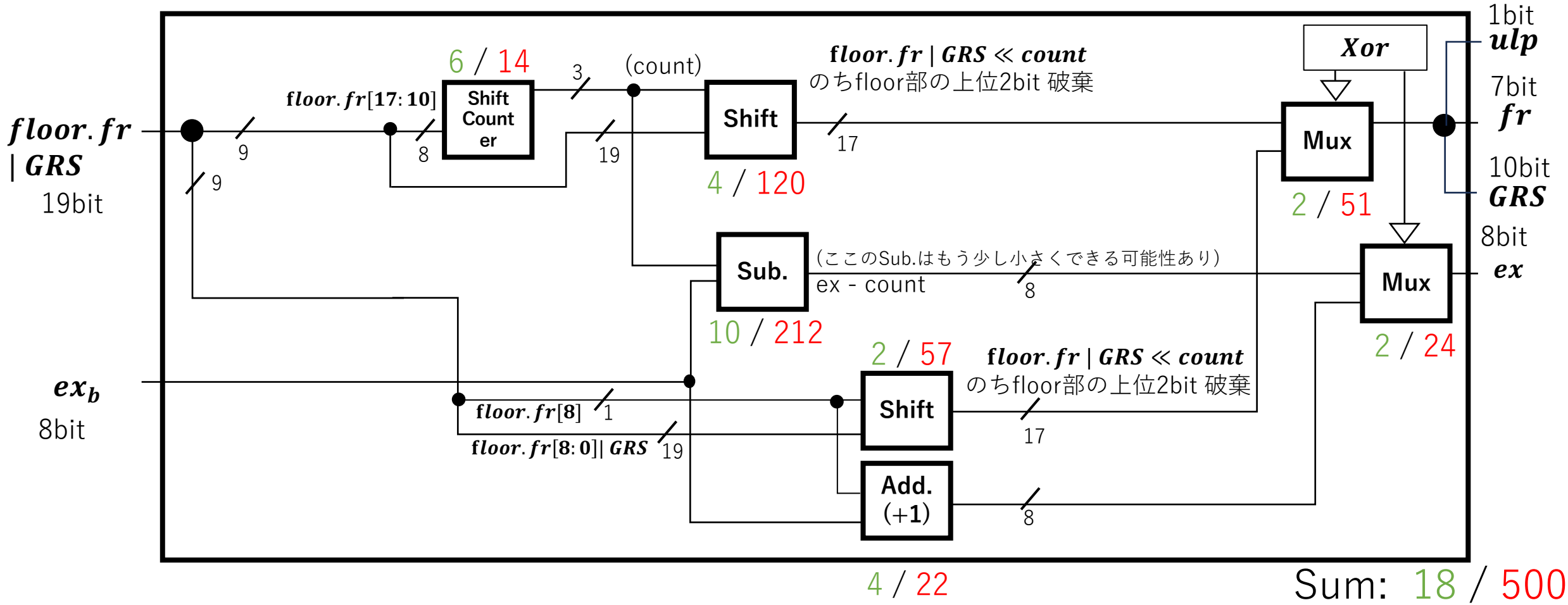
Input はfr_a



例: >>5 ならば 0 1 0 1
S_3 S_2 S_1 S_0

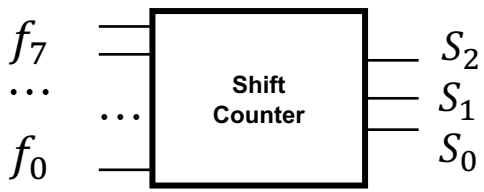
Sum: 8 / 165

Normalize (上層: Sub時 , 下層: Add時)Norm.



(Normalize内部) Shift Counter

概要：
「いくつシフトすればいいか」を数える回路。基本的にプライオリティエンコーダーと同じ。



真理値表：

f7	f6	f5	f4	f3	f2	f1	f0
1	*	*	*	*	*	*	*
0	1	*	*	*	*	*	*
0	0	1	*	*	*	*	*
0	0	0	1	*	*	*	*
0	0	0	0	1	*	*	*
0	0	0	0	0	1	*	*
0	0	0	0	0	0	1	*
0	0	0	0	0	0	0	1

s2	s1	s0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

例：0.1100100なら上から2番目であり、
出力としては001(1)が出る。

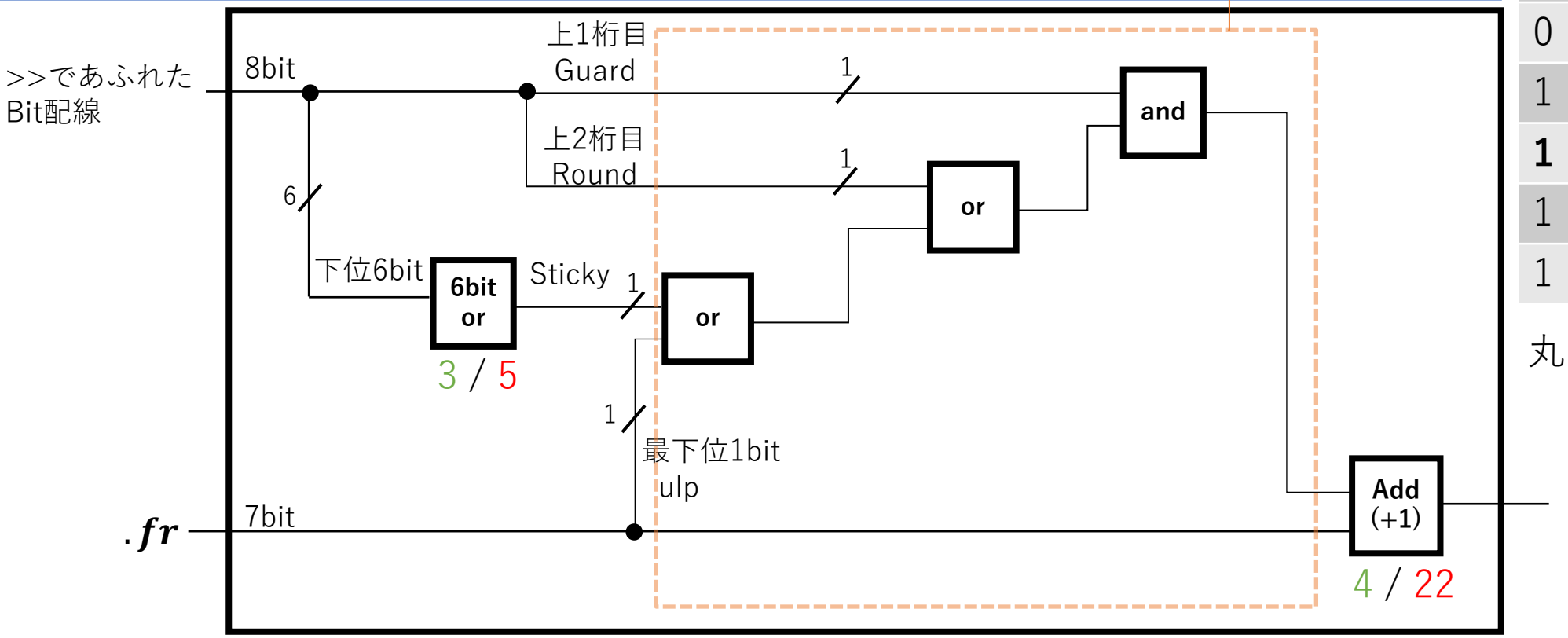
$$S_2 = \neg f_7 + \neg f_6 + \neg f_5 + \neg f_4$$
$$S_1 = \neg f_7 \neg f_6 (f_5 + f_4 + \neg f_3 \neg f_2 (f_1 + f_0))$$
$$S_0 = \neg f_7 (f_6 + \neg f_5 (f_4 + \neg f_3 (f_2 + \neg f_1 f_0)))$$

（←and , or , notでしか考えてないのでxorを入れたらもっとCountが減らせるかも）

Round

Round

概要：丸めの仕方は「最近点への丸め」を採用。
加えて，ちょうど真ん中(ulp * 1/2)のときは、「最も近い偶数に丸める」。
fr_aを右シフト(>>)したとき、右シフトであふれたBit配線から
Guard , Round, Sticky bitを生成し（以下G,R,S）、
G,R,S bitを用いて最下位bit(ulp)を＋１するかしないかを判定して処理する。



ulp	G	R	S	丸め操作
0	0	*	*	0
0	**1**	**0**	**0**	**0**
0	1	0	1	1
0	1	1	*	1
1	0	*	*	0
1	**1**	**0**	**0**	**1**
1	1	0	1	1
1	1	1	*	1

丸め操作
=G(S+R+ulp)

Sum: 10 / 30