A
```
11101011
11101011
01000100
10111100
```

B
```
01011100
00011100
01001010
01010011
```

Sa `00111`

LeftShift32
B
Cin
Sa
C

RightShift32
Op
Sa
B
C

Add32
A
B
Cin
C
V

Subtract32
A
B
C
V

The right shifter isn't
like a left shifter and done
all over again, but uses the
left shifter by transforming
the inputs and outputs to it.
same thing with subtracter.

MUX
0

V

C
```
01001000
00000111
10001111
00001111
```

I implemented the Op codes using multi-input (3- or 4- input) and gates and transferred the corresponding operation result to the output through a multiplexer.

I did not make a right shifter like how I made a left shifter, but followed the hint about transforming inputs and outputs to the right shifter.

Similarly with the subtractor; I just transformed B to be its two's complement and implemented overflow following rules about signed binary number addition/subtraction overflow.