Hardware System Design 4190.309A
Seoul National University

Practice #4. How to use IP catalog & Synthesize
Jiwon Lee, Sangjun Son

Goal

- Implement base of Float32 multiplier + accumulator.
- Implement Floating point / Integer fused multiply-adder using IP catalog.
- Implement adder-array using accumulator module.

1 Implementation

1.1 Adder

이 누산기는 2개의 input, 1개의 output, 1개의 overflow detecting bit 로 이루어져 있다. 같은 크기의 bitwidth를 가지는 2개의 input을 더했을 때, output의 bitwidth도 동일하기 때문에, overflow가 발생할 수 있다. Overflow detecting은 concatenation assign을 통해 구현하였다. Overflow bit와 output의 총 길이는 bitwidth+1이고, 2개의 input을 더했을 때 나올 수 있는 최대 bit수는 bitwidth+1이기 때문에, 이를 통해 overflow를 detect 할 수 있다.

```
1  `timescale 1ns / 1ps
2  module my_add #(
3      parameter BITWIDTH = 32
4  )
5  (
6      input [BITWIDTH-1:0] ain,
7      input [BITWIDTH-1:0] bin,
8      output [BITWIDTH-1:0] dout,
9      output overflow
10 );
11     // concatnate (overflow, dout) & detect overflow
12     assign {overflow, dout} = ain + bin;
13 endmodule
```

2 Results & Conclusion

2.1 Adder

testbench에서 주어지는 ain, bin의 값은 0 $2^{31}-1$ 까지의 임의의 값이다. 따라서, dout의 값은 최대 $2^{31}-2$ 이고, 이는 32bit로 만들 수 있는 최댓값인 $2^{31}-1$ 보다 작으므로 주어진 testbench에서는 overflow가 발생하지 않는다. 하지만, 실제로 ain, bin의 값을 증가시켜 overflow가 발생하도록 하였을 때, overflow bit가 1이 되어 detecting 한다는 것을 알 수 있었다.