Exercise 6 (Saturating and P_term) (HW2 prob 2):

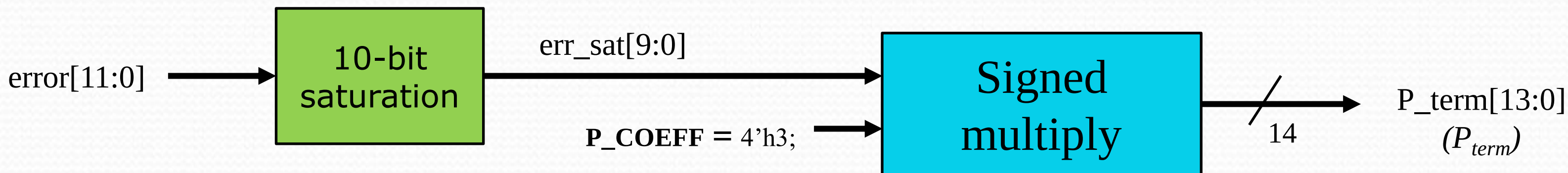
With digital control schemes. control input decisions are made based on sensor inputs. Take our specific case of the MazeRunner robot following heading. A MEMs gyro provides the current heading to *(12-bit precision)* and covers a 360° range. If it differs from our desired heading by 90° or more we are not going to do anything different than drive one wheel hard forward and the other wheel hard back to correct course. Thus we can reduce the width of the data coming from the gyro and not lose any "needed" information.

By reducing to a smaller number of bits we can make down stream calculations narrower (fewer bits wide) and thus use fewer gates, and less power. OK…now that I have setup the motivation for the problem. Here is the problem:

You are going to write dataflow Verilog to saturate the **error** term, and implement the **P_term** of the PID controller.

Exercise 6 (Reducing and Saturating)

Signal:	Dir:	Description:
error[11:0]	in	12-bit signed error term (heading – desired_heading)
P_term[13:0]	out	14-bit signed P component of PID controller



Consumed later when P, I, D terms are combined to form **PID**

- There are some **hints on saturating** presented on the next slide.
- The DUT should be in a file called **P_term.sv**
- The testbench should test several values (saturating both directions and not saturating) and be self checking. It should be in a file called **P_term_tb.sv**
- Turn in as much as you have completed by end of class. You have to finish this as part of HW2.

Hints on Saturating:

- Lets take a look at some examples saturating an 8-bit signed number to a 5-bit signed number.

Lets say you had the 8-bit number: **01010110**
You know it is positive because the MSB is 0
What is the greatest positive number we can represent in 5-bits? … **01111**
Is **01010110** greater than **01111**? Yes. How do we know? Because it is positive and it has bit(s) in the [6:4] range set. So we saturate to **01111**

Lets say you had the 8-bit number: **11010110**
You know it is negative because the MSB is 1
What is the greatest negative number we can represent in 5-bits? … **10000**
Is **11010110** more negative than **10000**? Yes. How do we know? Because it is negative and it has bit(s) in the [6:4] range clear. So we saturate to **10000**

Lets say you had the 8-bit number: **11110110**
You know it is negative because the MSB is 1
What is the greatest negative number we can represent in 5-bits? … **10000**
Is **11110110** more negative than **10000**? No. How do we know? Because it is negative but all the bits in the [6:4] are also set. So we simply keep bits [4:0] as our 5-bit result.

(more on next slide)

Hints on Saturating:

- So..from the previous slide you can derive the following pseudo code:

```
if ((number is negative) &&
    (any upper bits (in certain range) are zero)) {
        saturate to most negative number
}
else if ((number is positive) &&
        (any upper bits (in certain range) are one)) {
    saturate to most positive number
}
else {
    number not out of range so just copy over lower bits
}
```

Of course this is **dataflow** Verilog, so we are NOT using **if**. Instead use ternary **assign** statements.

What is an effective way to tell if any bits in a range are 1 or 0? Think about the reduction operator.