Exercise 15 (fast_sim parameter & balance_cntrl):

- There are some long time constants in PID
  - **ss_tmr** ramps up over 2-seconds (100+ million clks)
  - Integrator integrates slowly

- Kept as is, these take too long to simulate.

- We need a way of speeding up simulation. We will introduce a **parameter** falled ***fast_sim***

- Look about 2/3 of the way through Lecture02. You will find a few examples of **parameters** and how to use them.

- Look about 40% of the way through Lecture08 *(yes jumping ahead)*. You will see example of a **generate** conditional.

Exercise 15 (introduce fast_sim to PID):

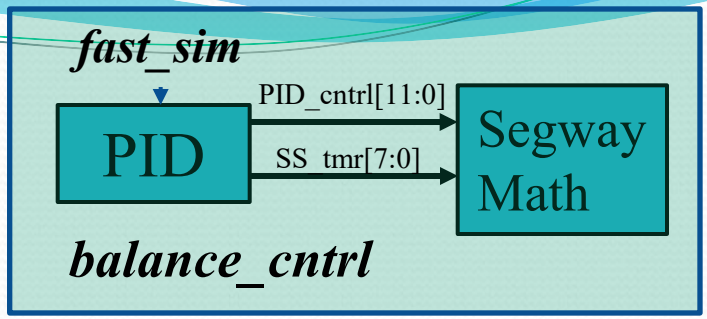
- To speed up **ss_tmr**
  - As is, we increment the 27-bit timer by 1 every **clk**
  - Under the condition of ***fast_sim=1*** we will now increment the 27-bit timer by 256 every **clk**
  - Embedding this decision inside a **generate** conditional makes sense.

- To speed up integrator
  - As is, **I_term** is tapped from bits [17:6] if the **integrator** accumulator, *make it {{3{integrator[17]}},integrator[17:6]};*
  - Under the condition of ***fast_sim=1*** we will now tap bits [15:1] to form **I_term**.
  - Not using bits [17:16] means there is danger that the integrator is too positive or too negative to represent in **I_term**, so this assignment under ***fast_sim*** should be a saturating assign that inspects bits [17:15] to determine if +/- saturation occurs.
  - Again…use a **generate** conditional *(or combine with above)*

Exercise 15 (introduce fast_sim to PID):

- Testing modified PID
- There is a provided testbench: **PID_fastsim_tb.sv**
- Might be a good idea to run it against your modified PID.

Exercise 15 (create balance_cntrl):



- ***balance_cntrl*** is just a combination of ***PID*** and ***SegwayMath***

Signal:	Dir:	Description
clk,rst_n	in	50MHz system clock & active low reset
vld	in	High whenever new inertial sensor reading (**ptch**) is ready
ptch[15:0]	in	Pitch of Segway from ***inertial_intf***
ptch_rt[15:0]	in	Pitch rate (degrees/sec). Used for D_term of PID
pwr_up	in	Asserted when Segway balance control is powered up. Used to keep ss_tmr at zero until then.
rider_off	in	Asserted when no rider detected. Zeros out integrator.
steer_pot[11:0]	in	From A2D_intf (converted from steering potentiometer)
en_steer	in	Enables steering control
lft_spd[11:0]	out	12-bit signed speed of left motor
rght_spd[11:0]	out	12-bit signed speed of left motor
too_fast	out	Rider approaching point of minimal control margin

Interface of ***balance_cntrl*** *(should also have **fast_sim** as a parameter, defaulted to 1)*

Exercise 15 (create **balance_cntrl**):

- Create **balance_cntrl.sv**
  - Note the connections between ***PID*** and ***SegwayMath*** (**PID_cntrl[11:0]** and **ss_tmr[7:0]**)

- ***fast_sim*** should also be a parameter at the ***balance_cntrl*** level of hierarchy and should be defaulted to 1

- Compile **balance_cntrl.sv** in ModelSim so you know at least it is syntaxually correct.
  - We will test it in Exercise17 so don't worry about that yet.

- Submit **balance_cntrl.sv** and proof it compiled in ModelSim.