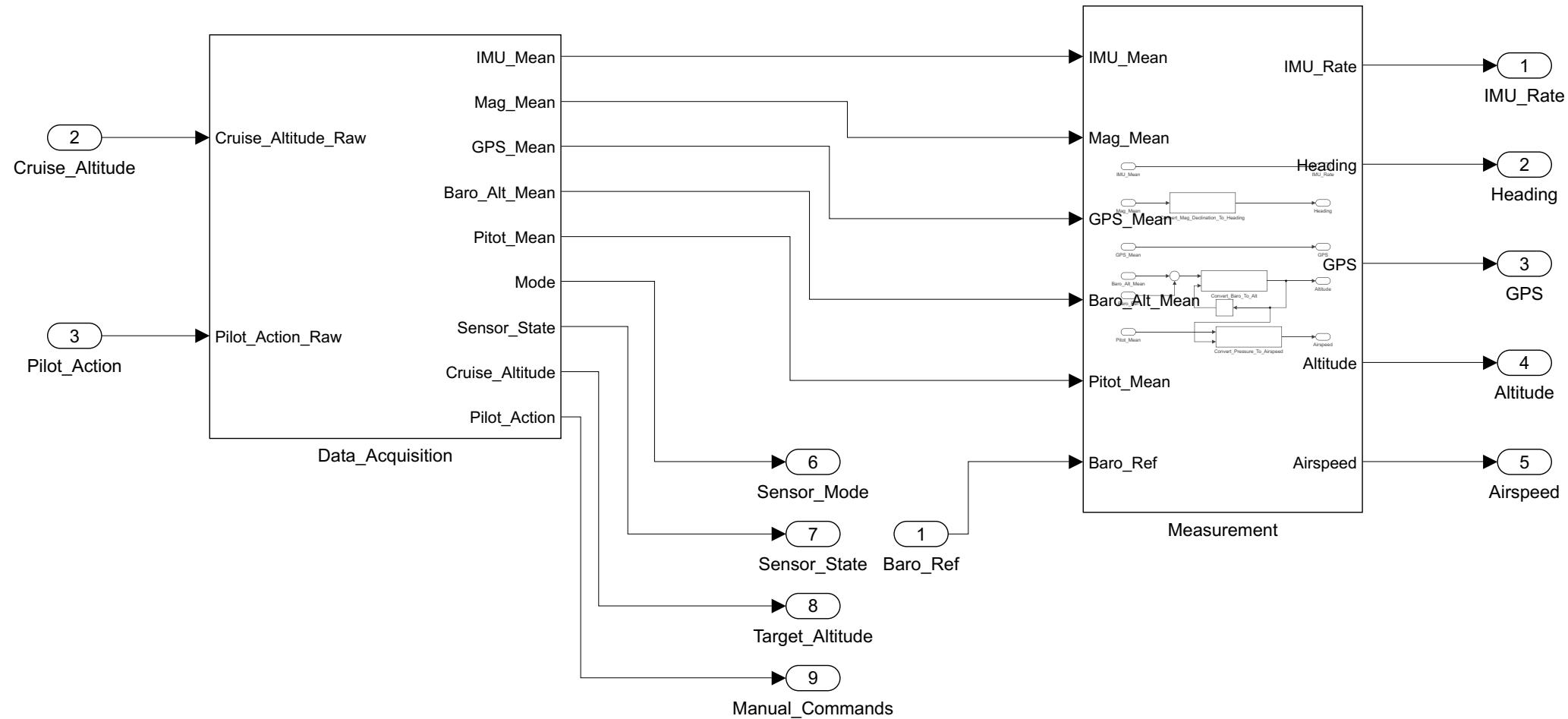
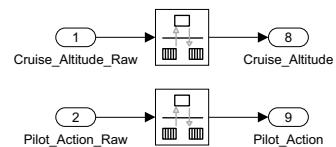
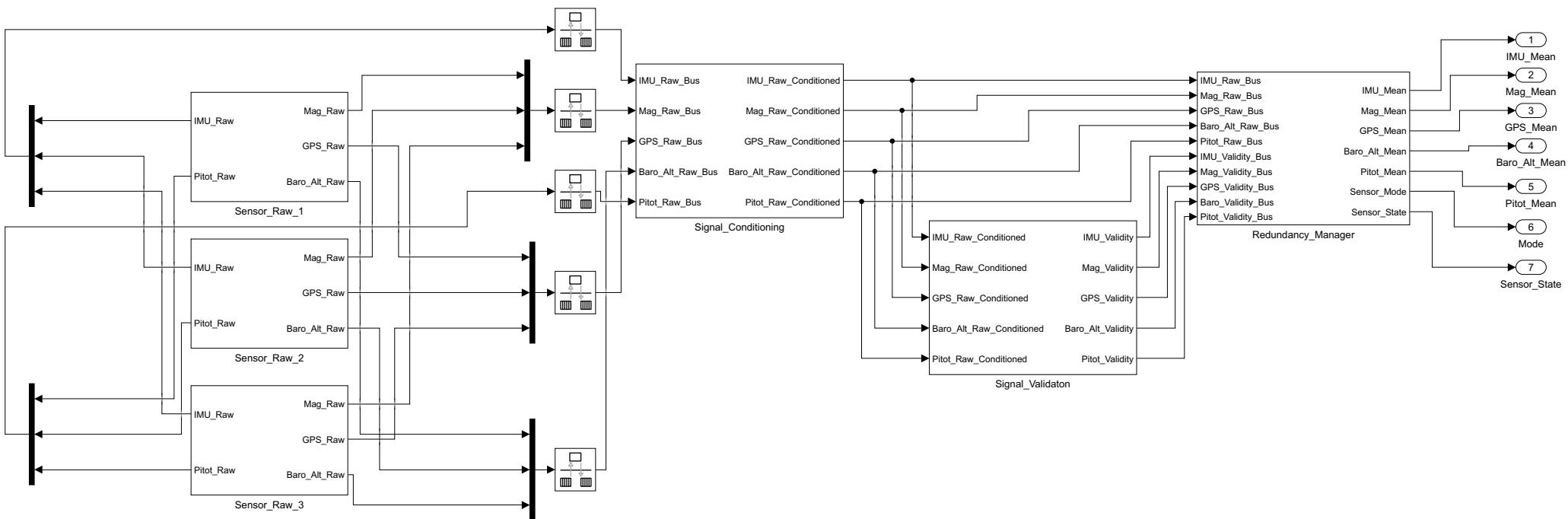


1 ➤
Autopilot_Mode

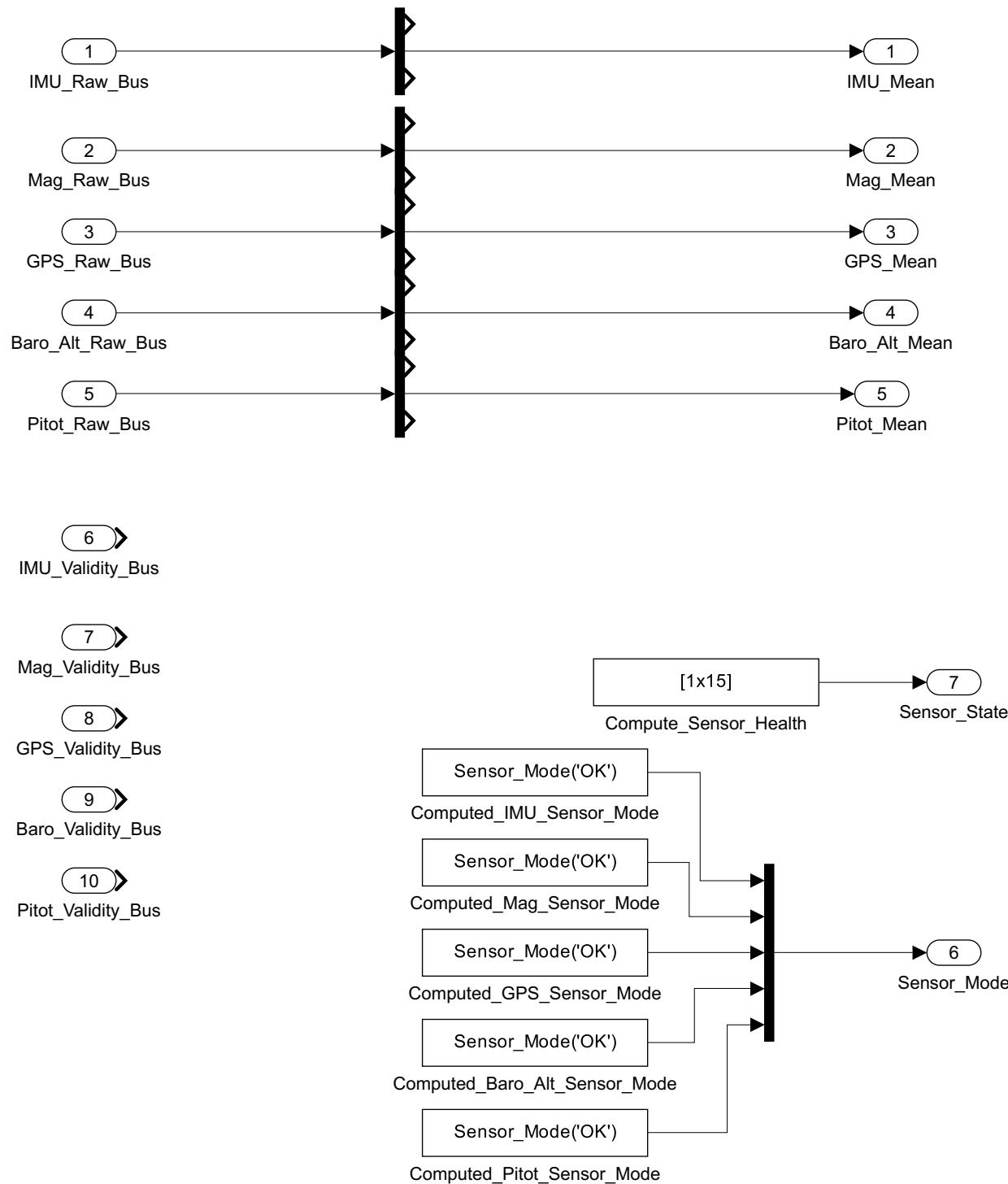
2 ➤
Flight_Phase

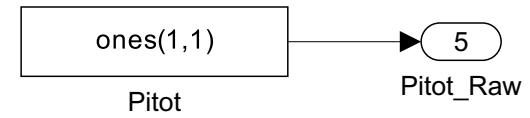
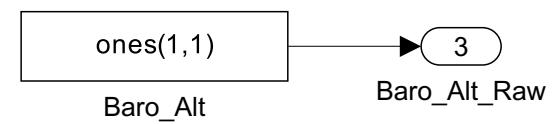
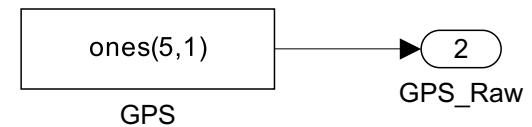
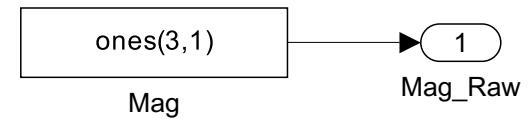
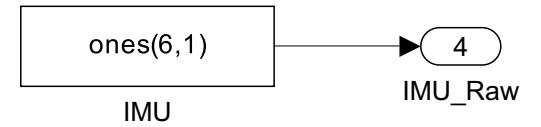
✗ 1
Controller_Gain

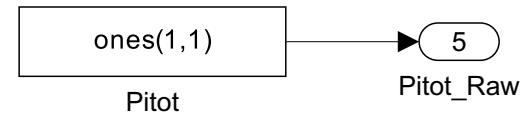
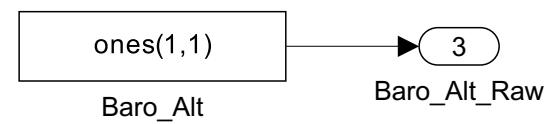
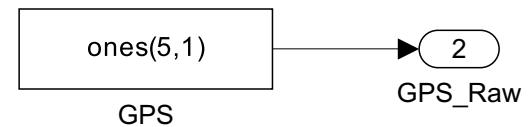
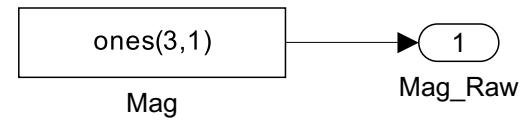
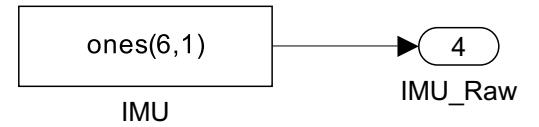


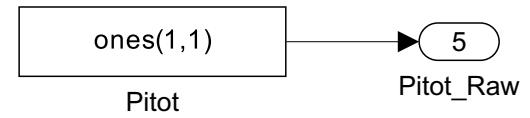
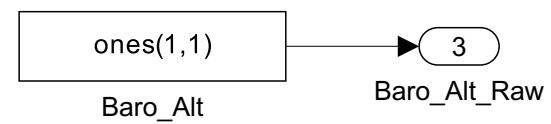
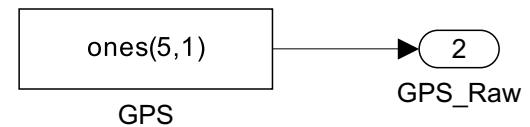
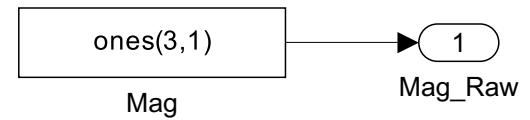
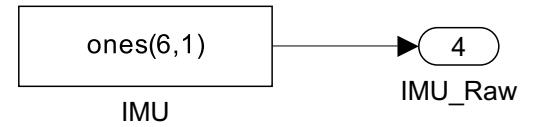


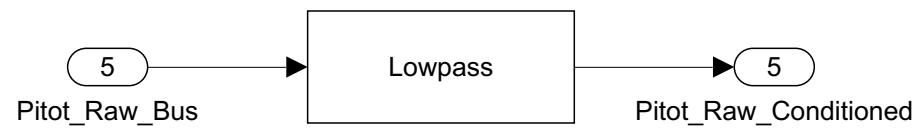
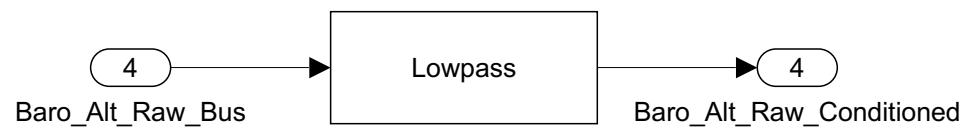
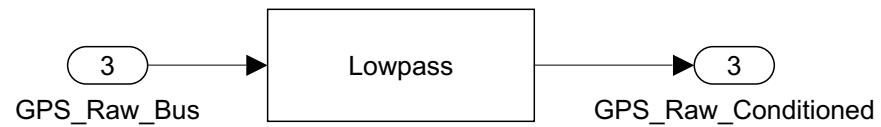
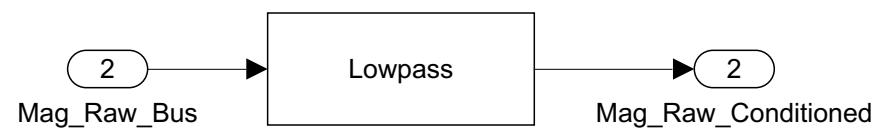
Dummy_Implementation

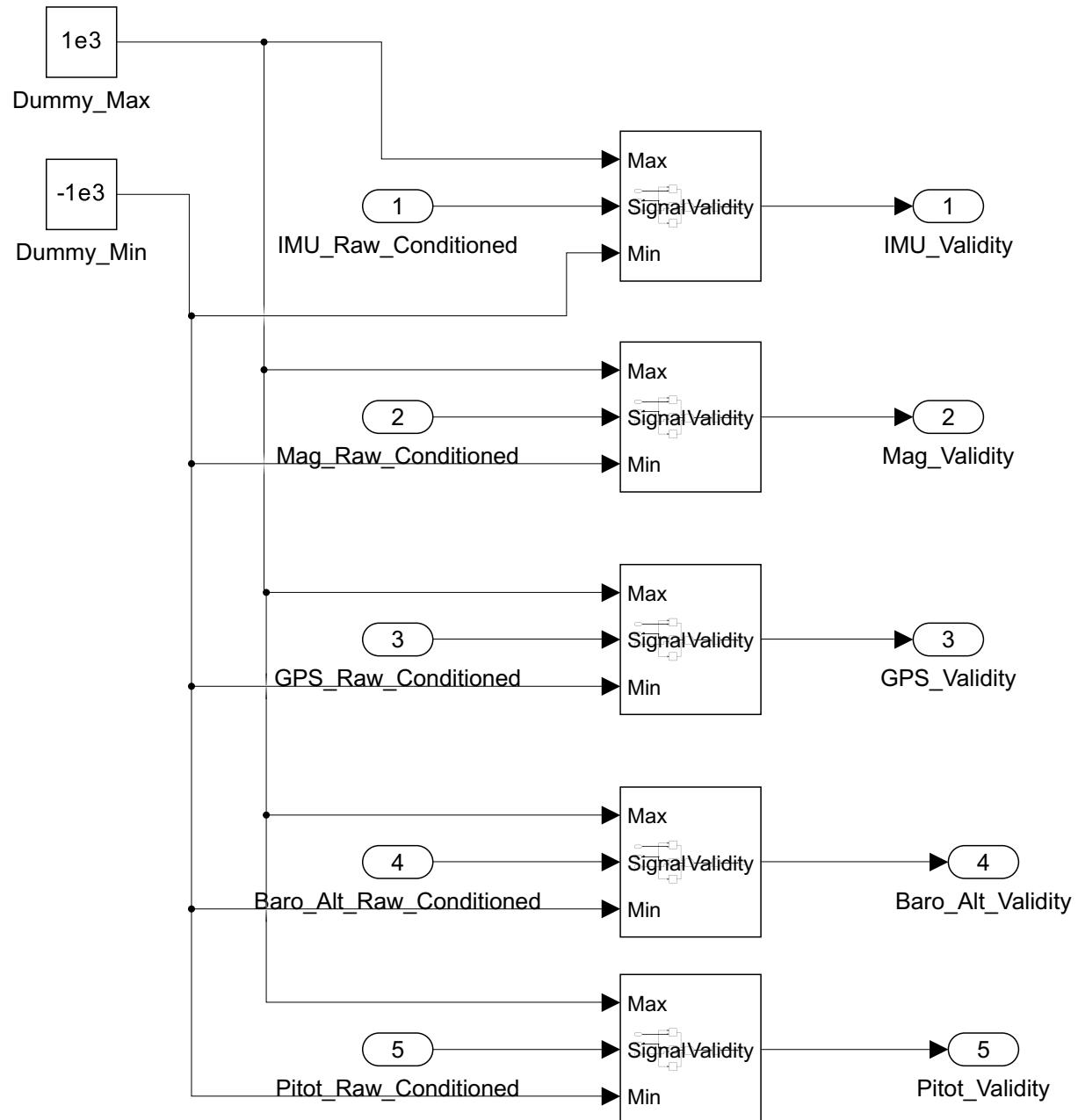


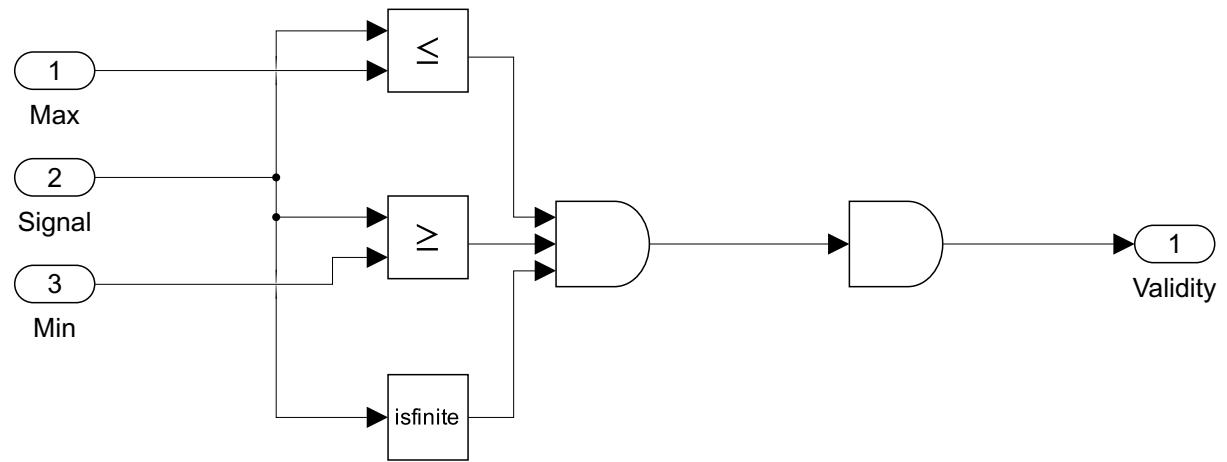


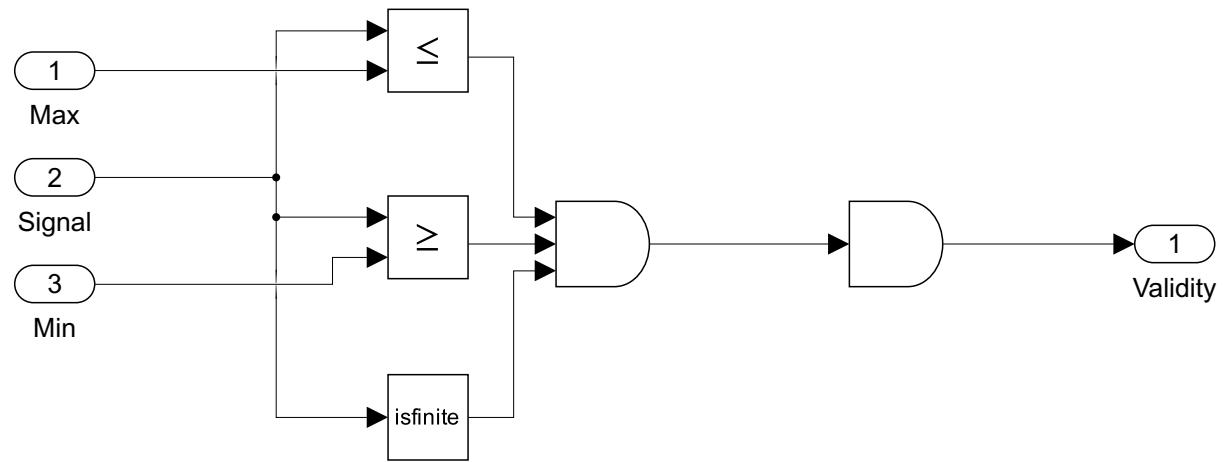


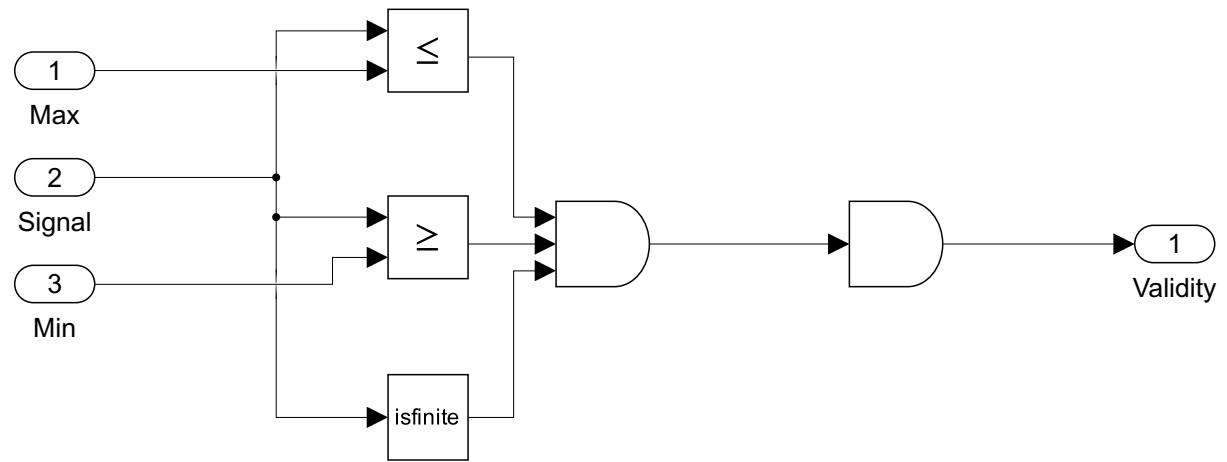


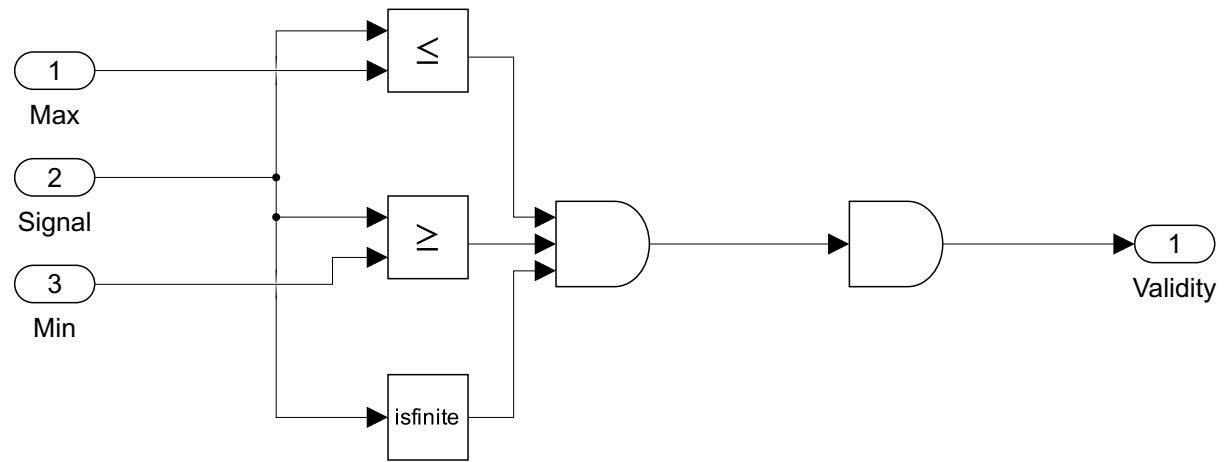


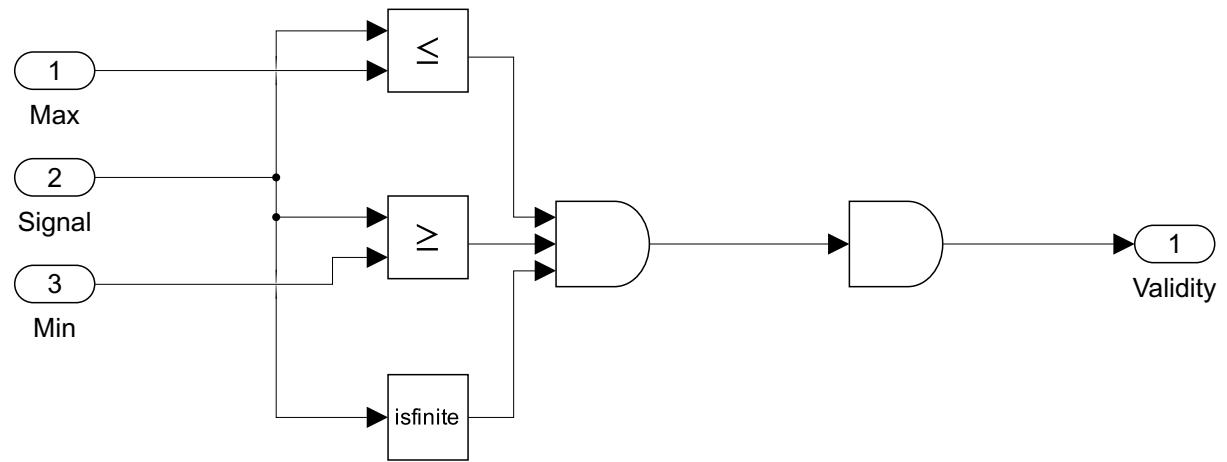


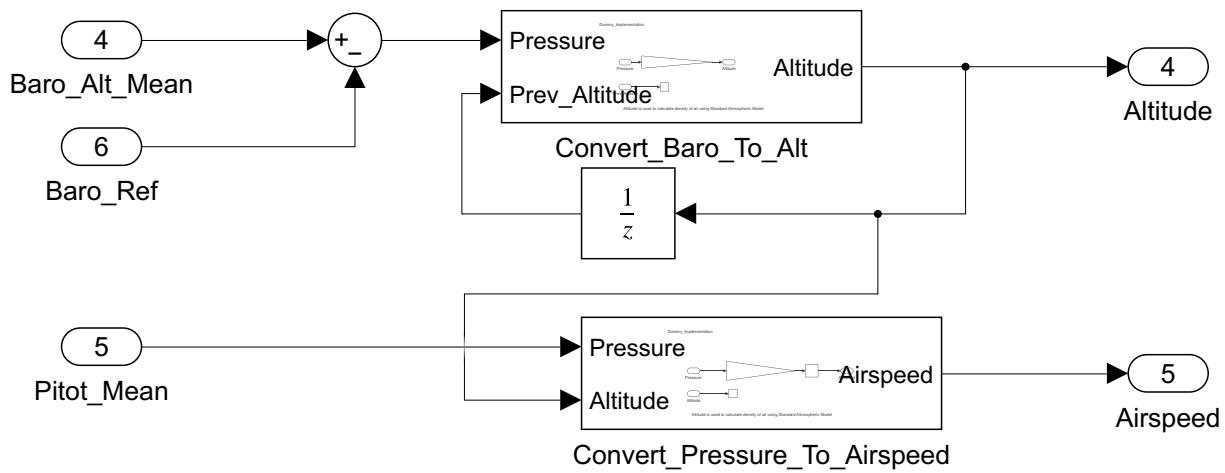
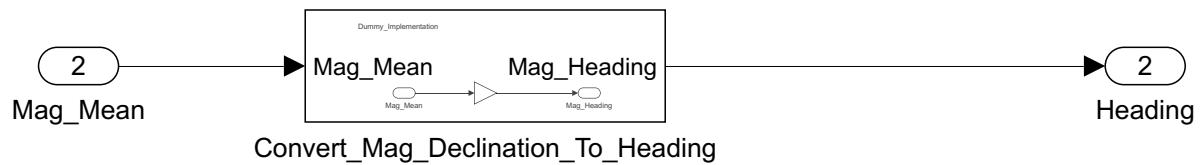




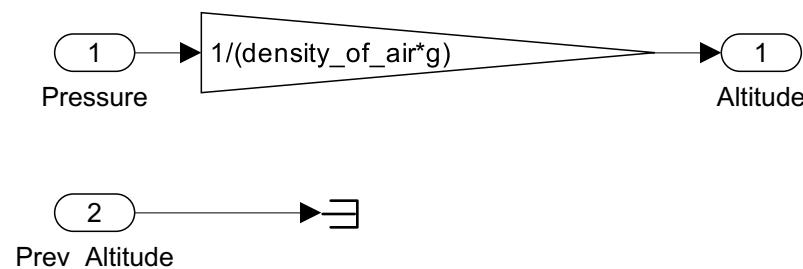






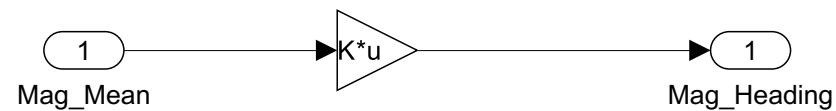


Dummy_Implementation

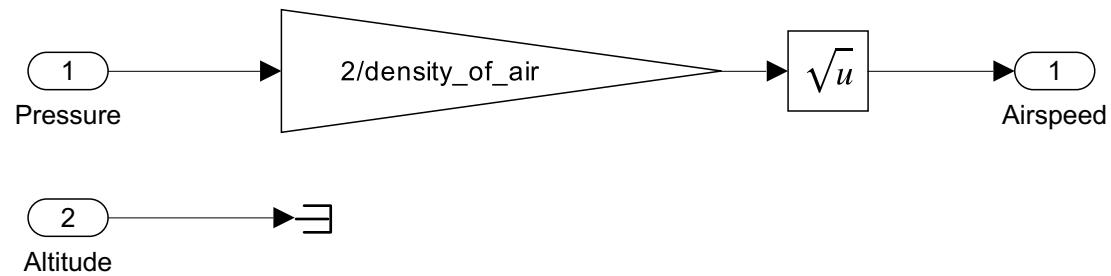


Altitude is used to calculate density of air using Standard Atmospheric Model

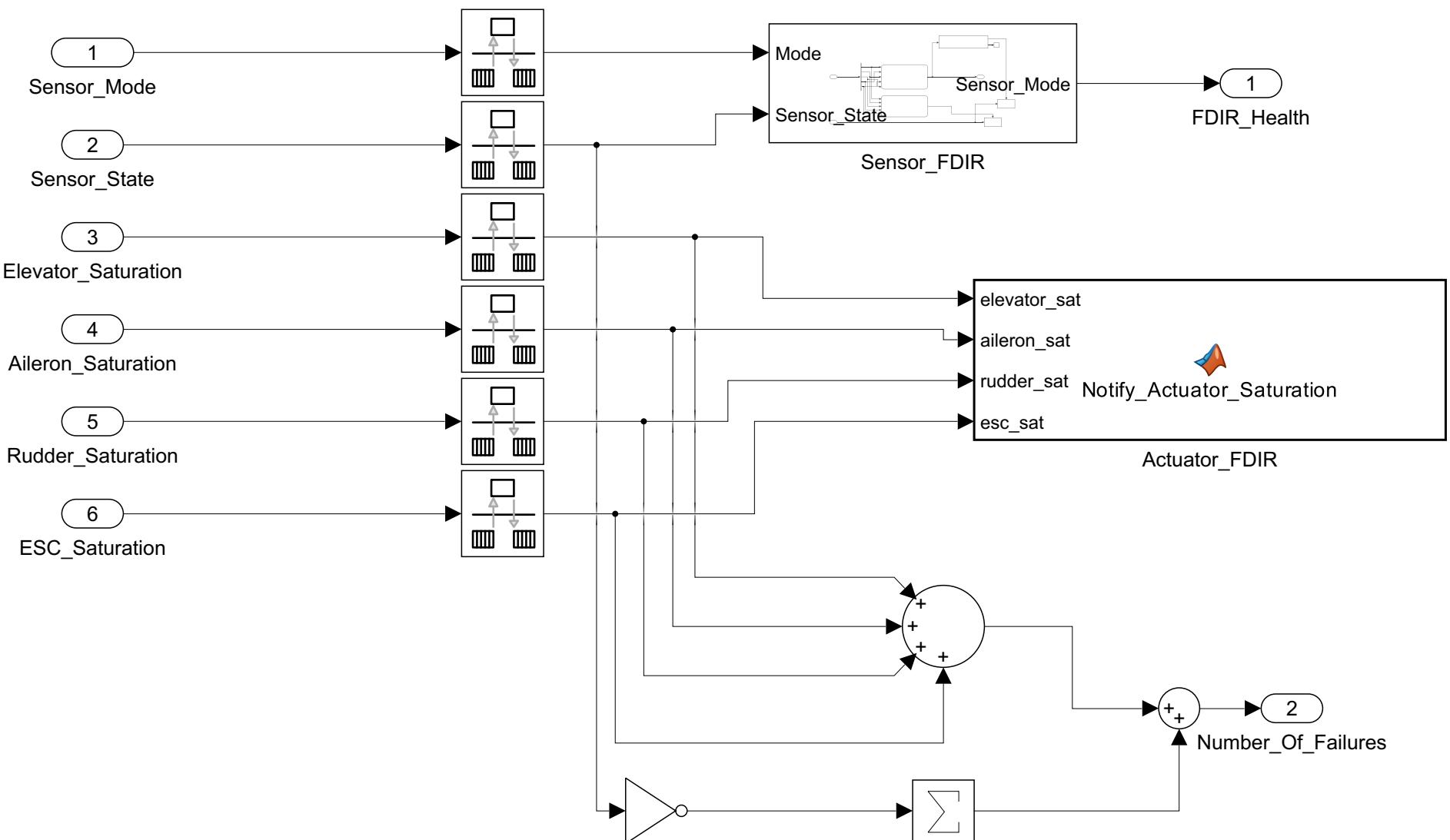
Dummy_Implementation



Dummy_Implementation



Altitude is used to calculate density of air using Standard Atmospheric Model

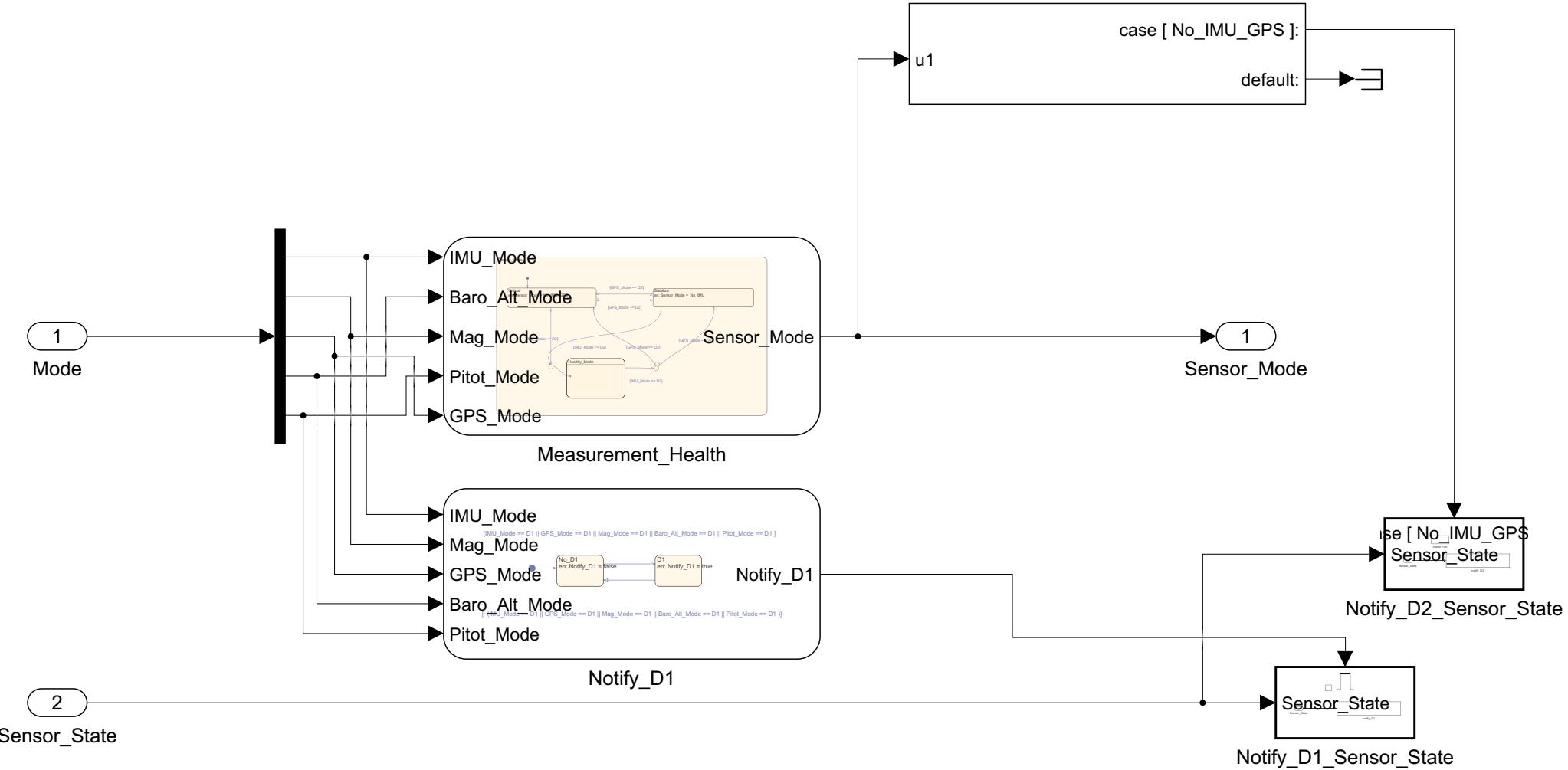


```
function Notify_Actuator_Saturation(elevator_sat, aileron_sat, rudder_sat, esc_sat)
    if elevator_sat
        disp('Elevator Servo saturated')
    end

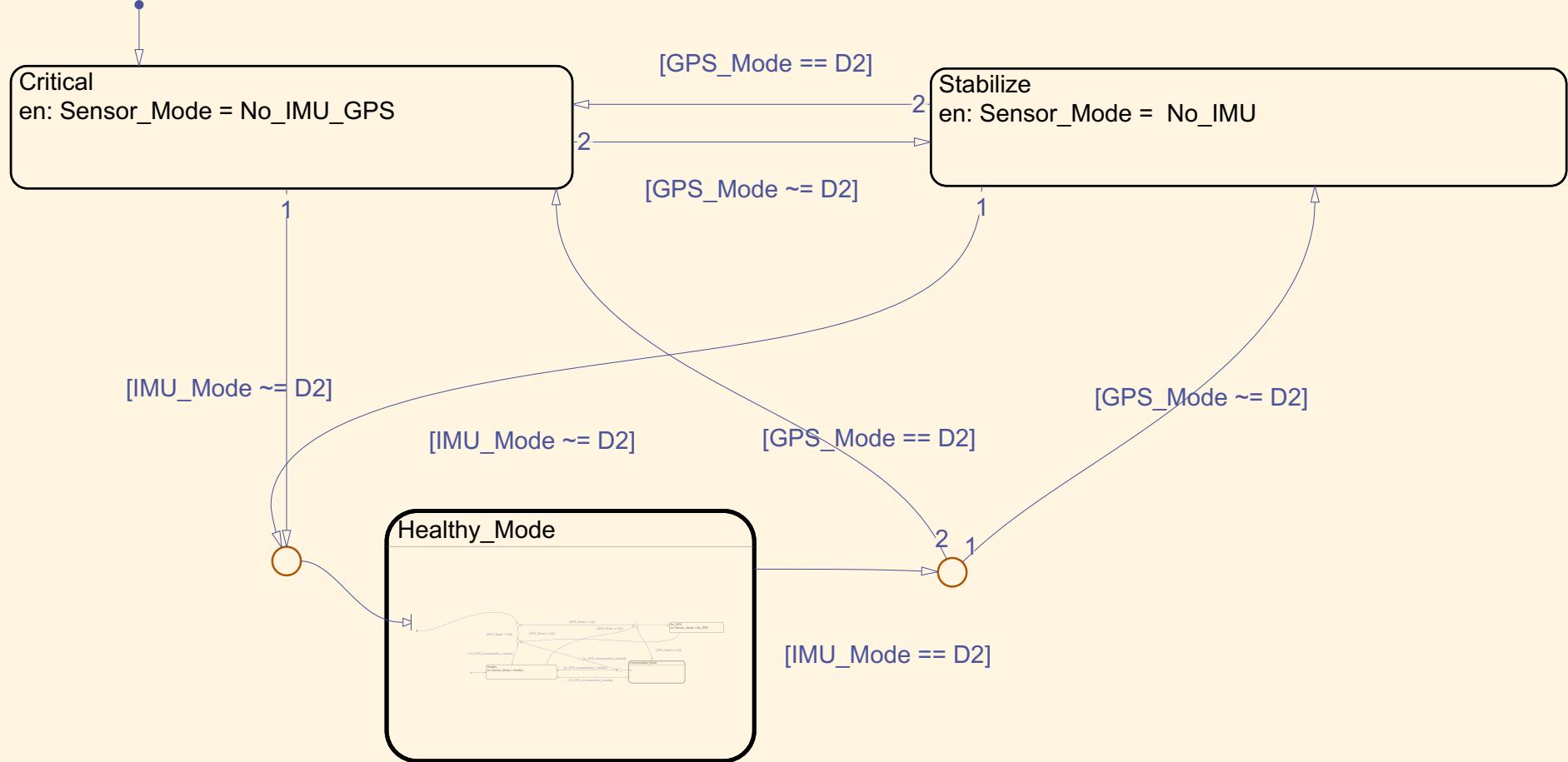
    if aileron_sat
        disp('Aileron Servo saturated')
    end

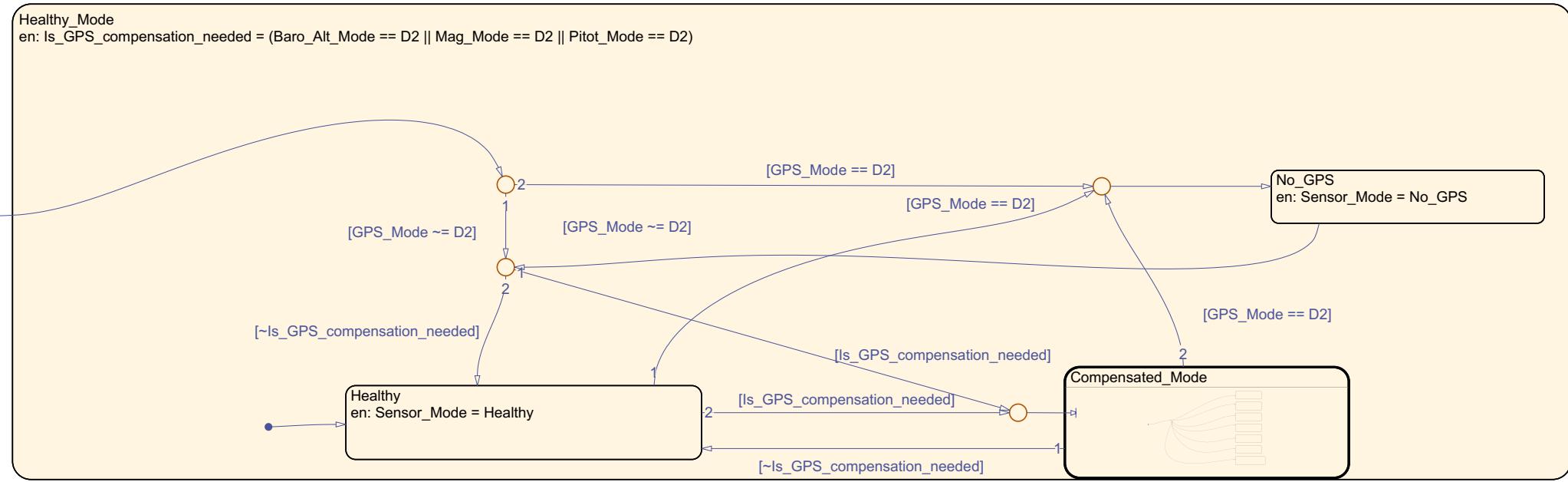
    if rudder_sat
        disp('Rudder Servo saturated')
    end

    if esc_sat
        disp('ESC saturated')
    end
end
```

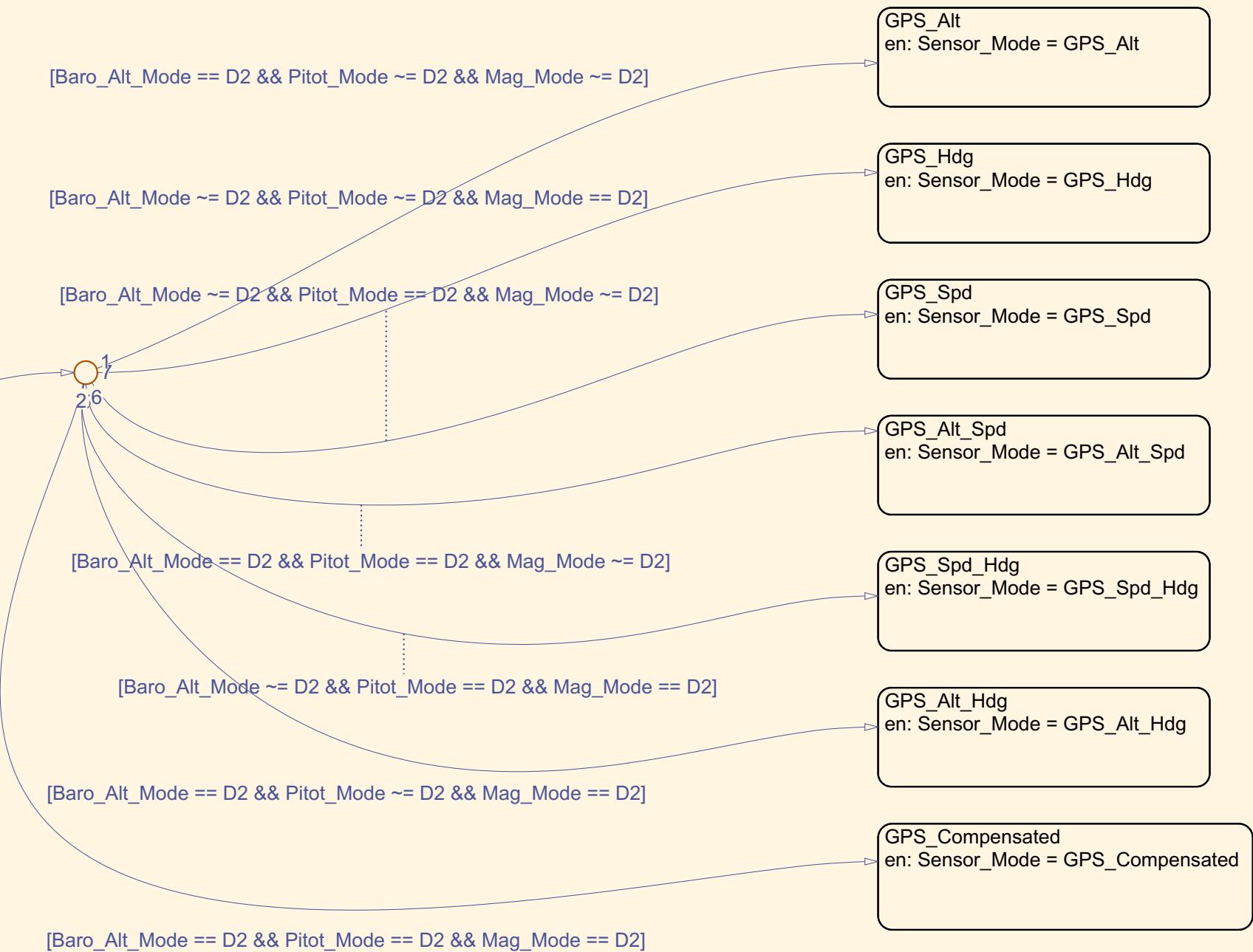


Sensor_Mode

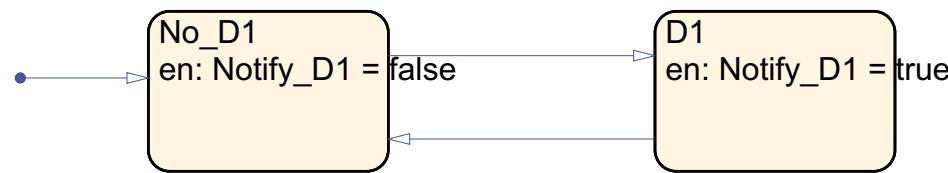




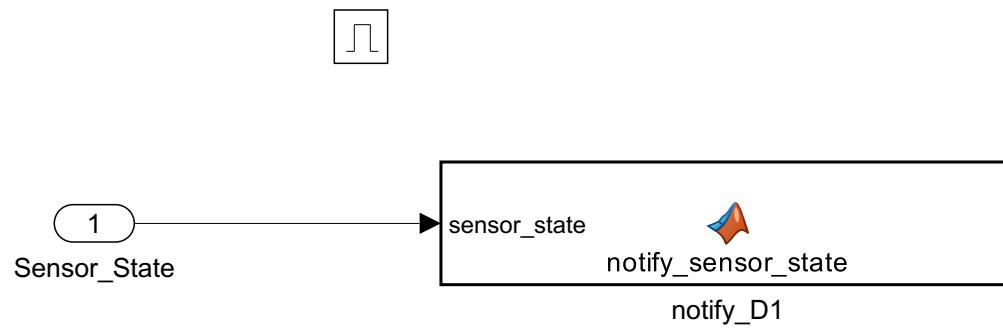
Compensated_Mode



$[IMU_Mode == D1 \parallel GPS_Mode == D1 \parallel Mag_Mode == D1 \parallel Baro_Alt_Mode == D1 \parallel Pitot_Mode == D1]$

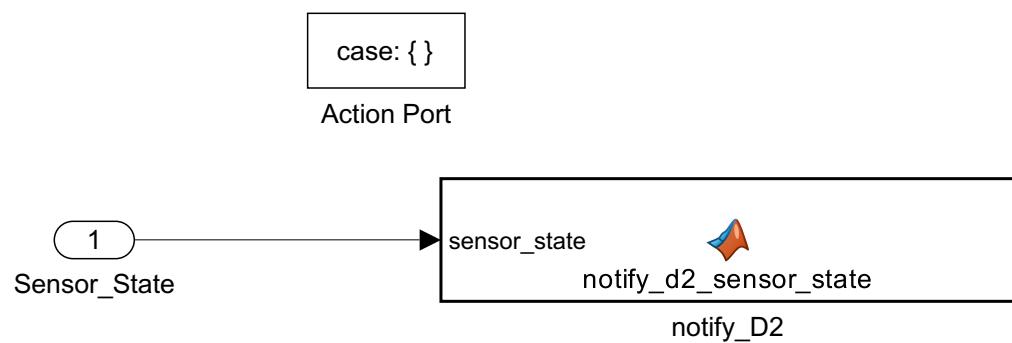


$[\sim(IMU_Mode == D1 \parallel GPS_Mode == D1 \parallel Mag_Mode == D1 \parallel Baro_Alt_Mode == D1 \parallel Pitot_Mode == D1)]$

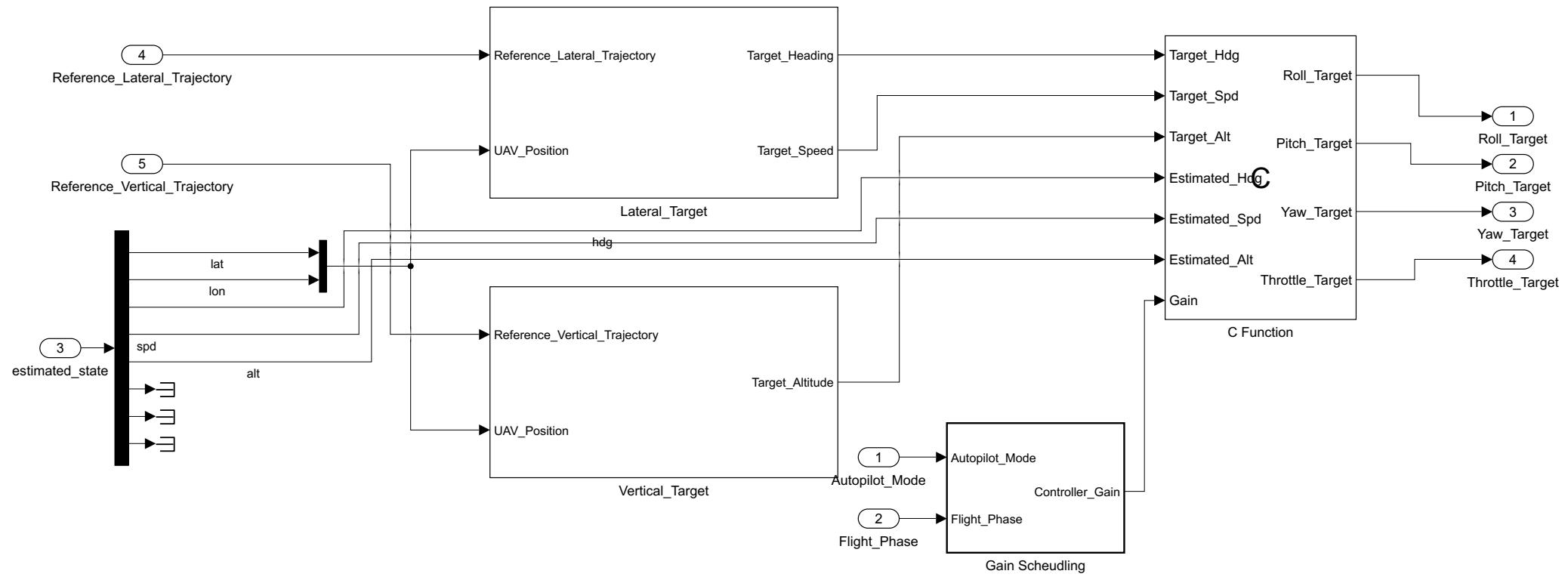


```
function notify_sensor_state(sensor_state)

for i = 1:length(sensor_state)
if sensor_state(i) ~= true
    if i >= 1 && i <= 3
        disp('IMU_%d degraded', i)
    elseif i >= 4 && i <= 6
        disp('Mag_%d degraded', i)
    elseif i >= 7 && i <= 9
        disp('GPS_%d degraded', i)
    elseif i >= 10 && i <= 12
        disp('Baro_Alt_%d degraded', i)
    else
        disp('Pitot_%d degraded', i)
    end
end
end
```



```
function notify_d2_sensor_state(sensor_state)
    for i = 1:3:length(sensor_state)
        if sensor_state(i) ~= true
            if i == 1
                disp('All IMU degraded')
            elseif i == 4
                disp('All Mag degraded')
            elseif i == 7
                disp('All GPS degraded')
            elseif i == 10
                disp('All Baro_Alt degraded')
            else
                disp('All Pitot degraded')
            end
        end
    end
```



1 ➤
Autopilot_Mode

2 ➤
Flight_Phase

✗ 1
Controller_Gain

1 ➞

Reference_Lateral_Trajectory

2 ➞

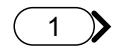
UAV_Position

X 1

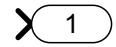
Target_Heading

X 2

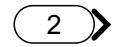
Target_Speed



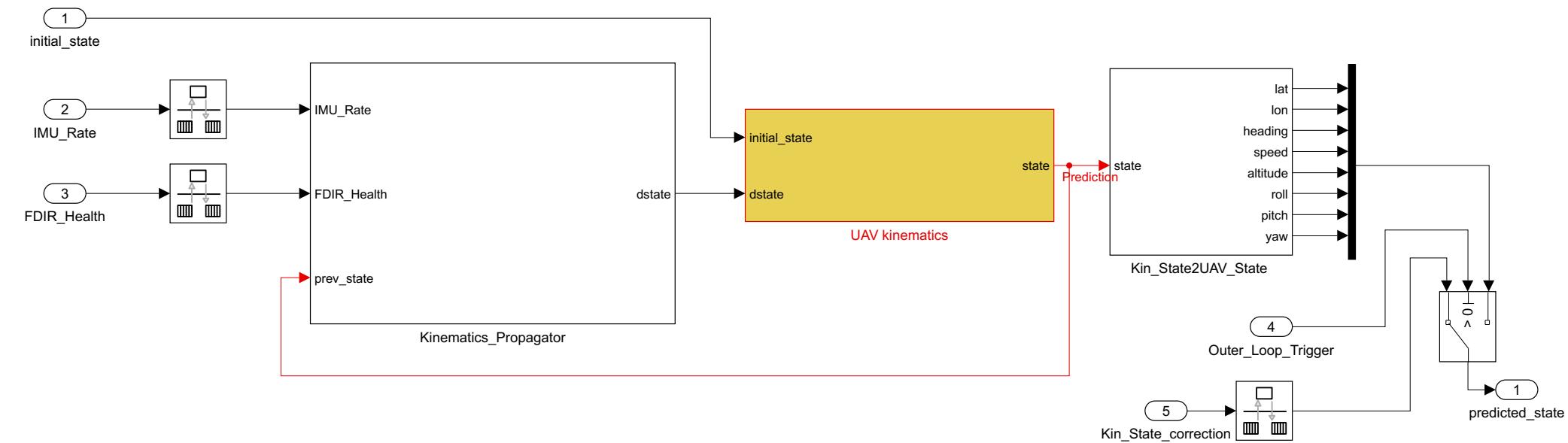
Reference_Vertical_Trajectory



Target_Altitude



UAV_Position



1 ➞

state

X 1

lat

X 2

lon

X 3

heading

X 4

speed

X 5

altitude

X 6

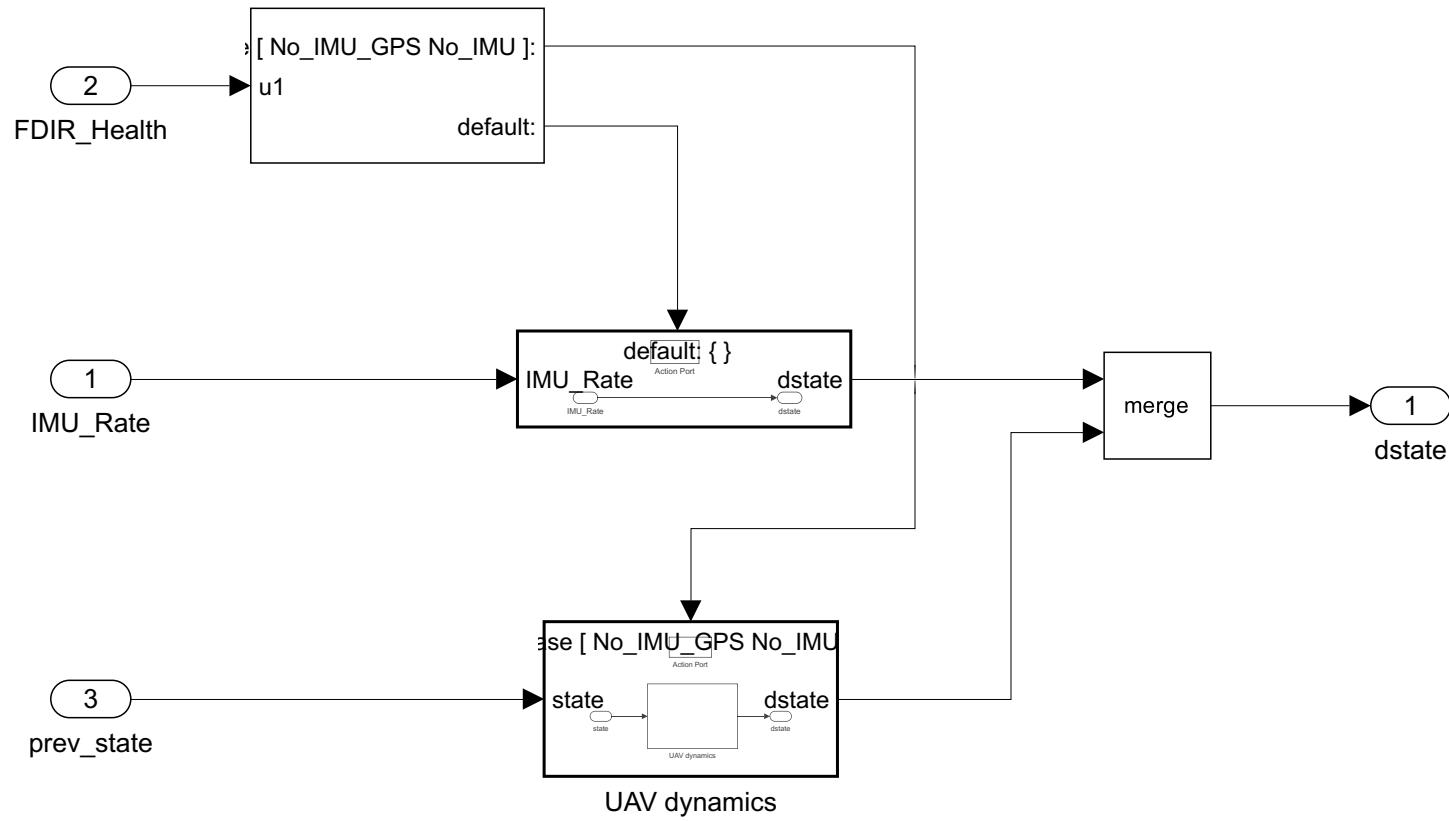
roll

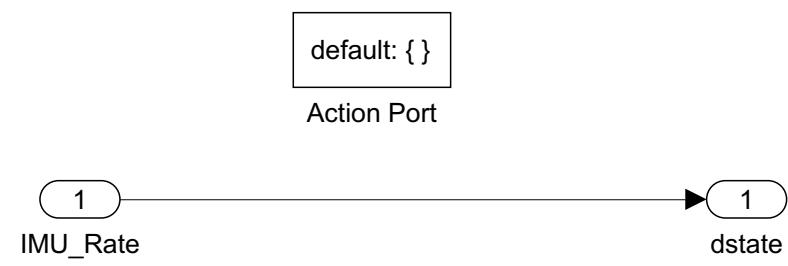
X 7

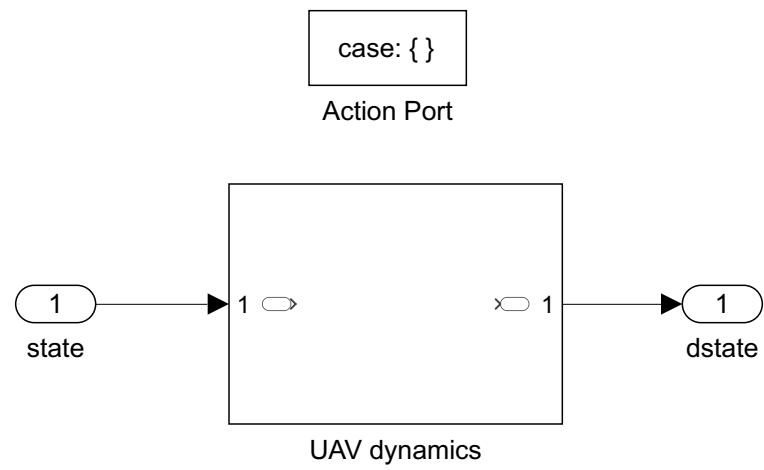
pitch

X 8

yaw

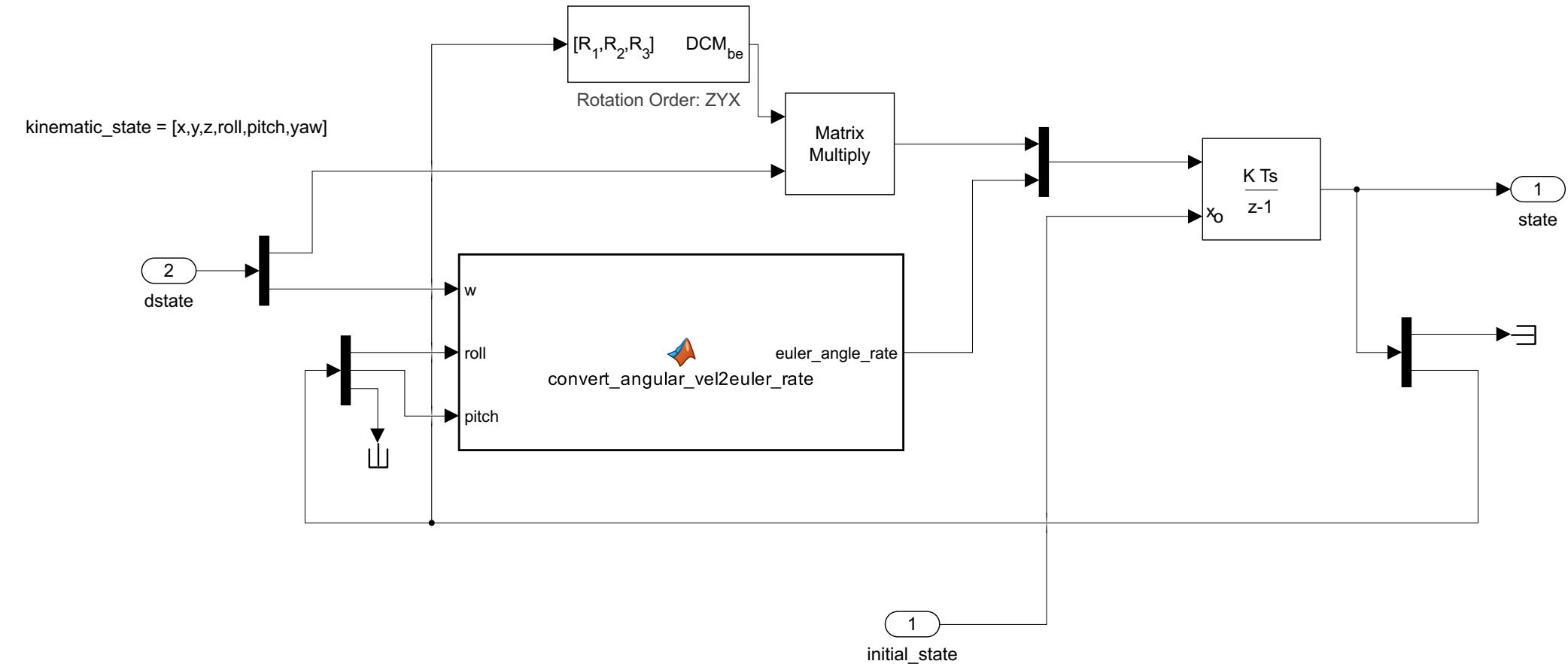






1 ➤

✖ 1

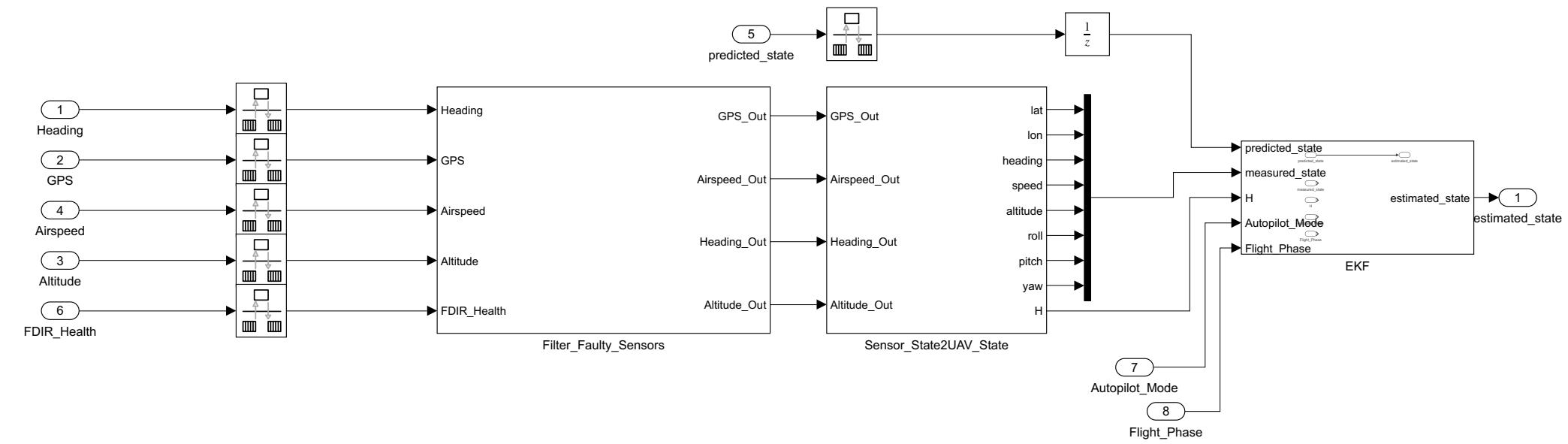


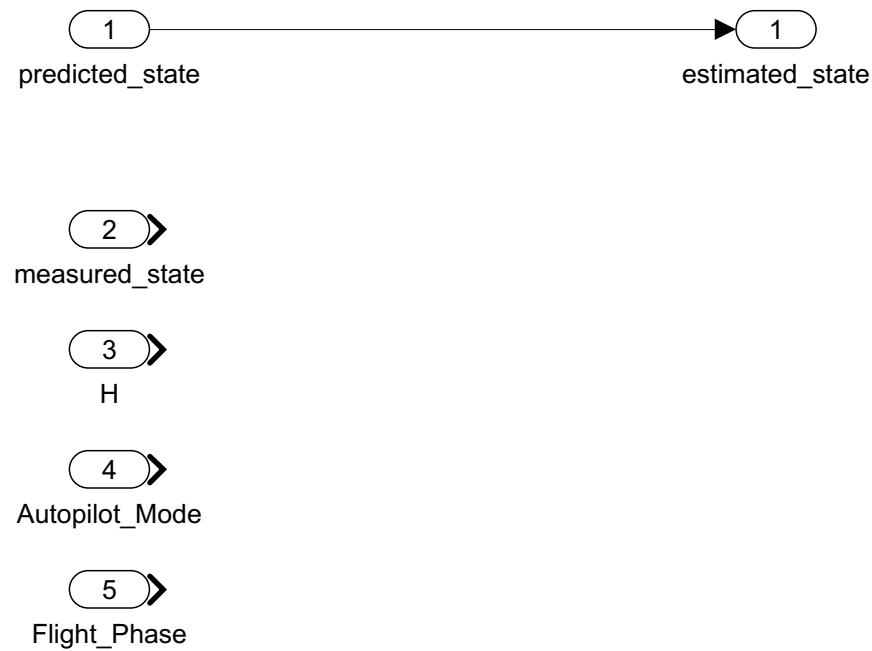
```

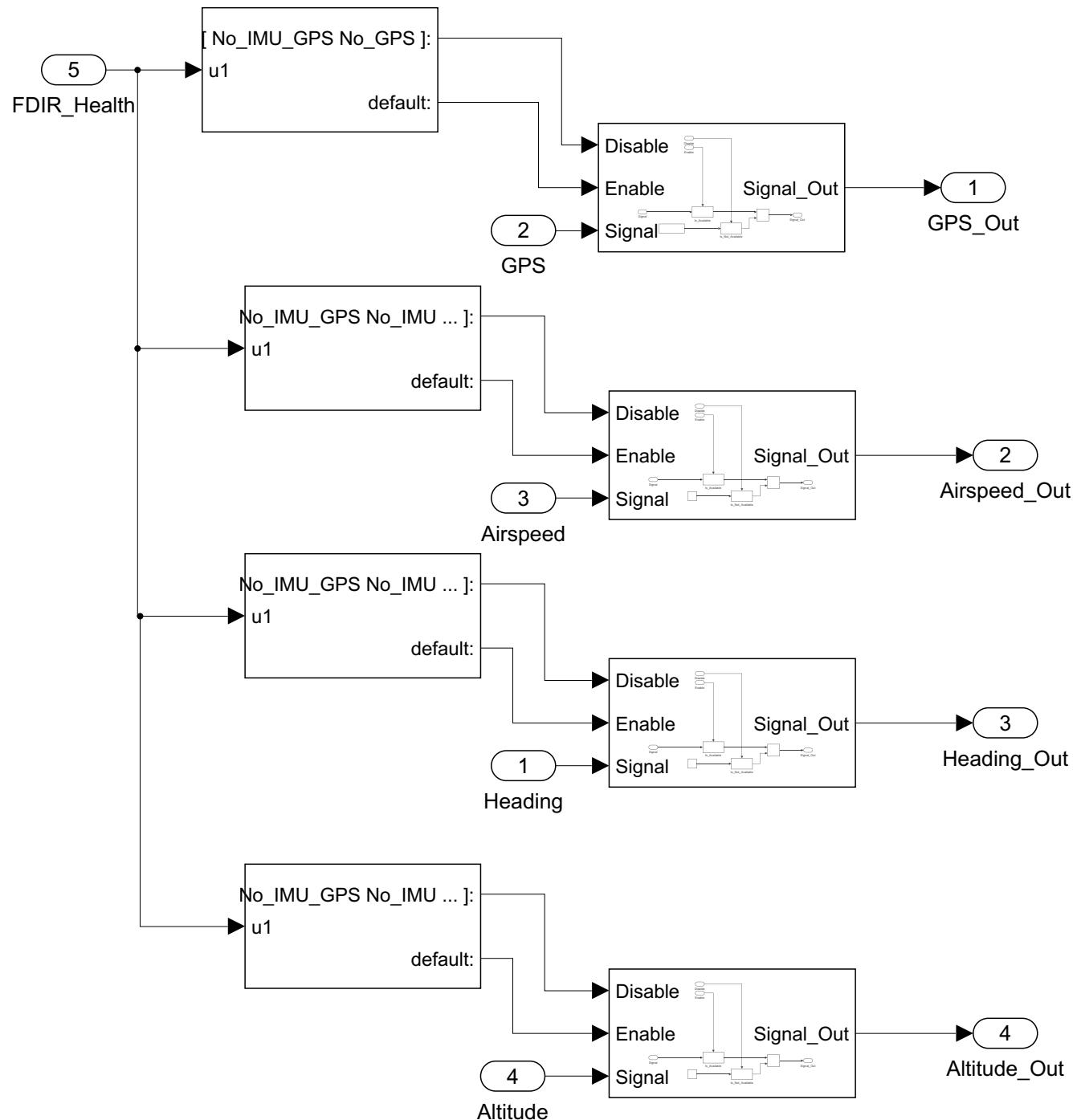
function euler_angle_rate = convert_angular_vel2euler_rate(w, roll, pitch)
p = w(1);
q = w(2);
r = w(3);

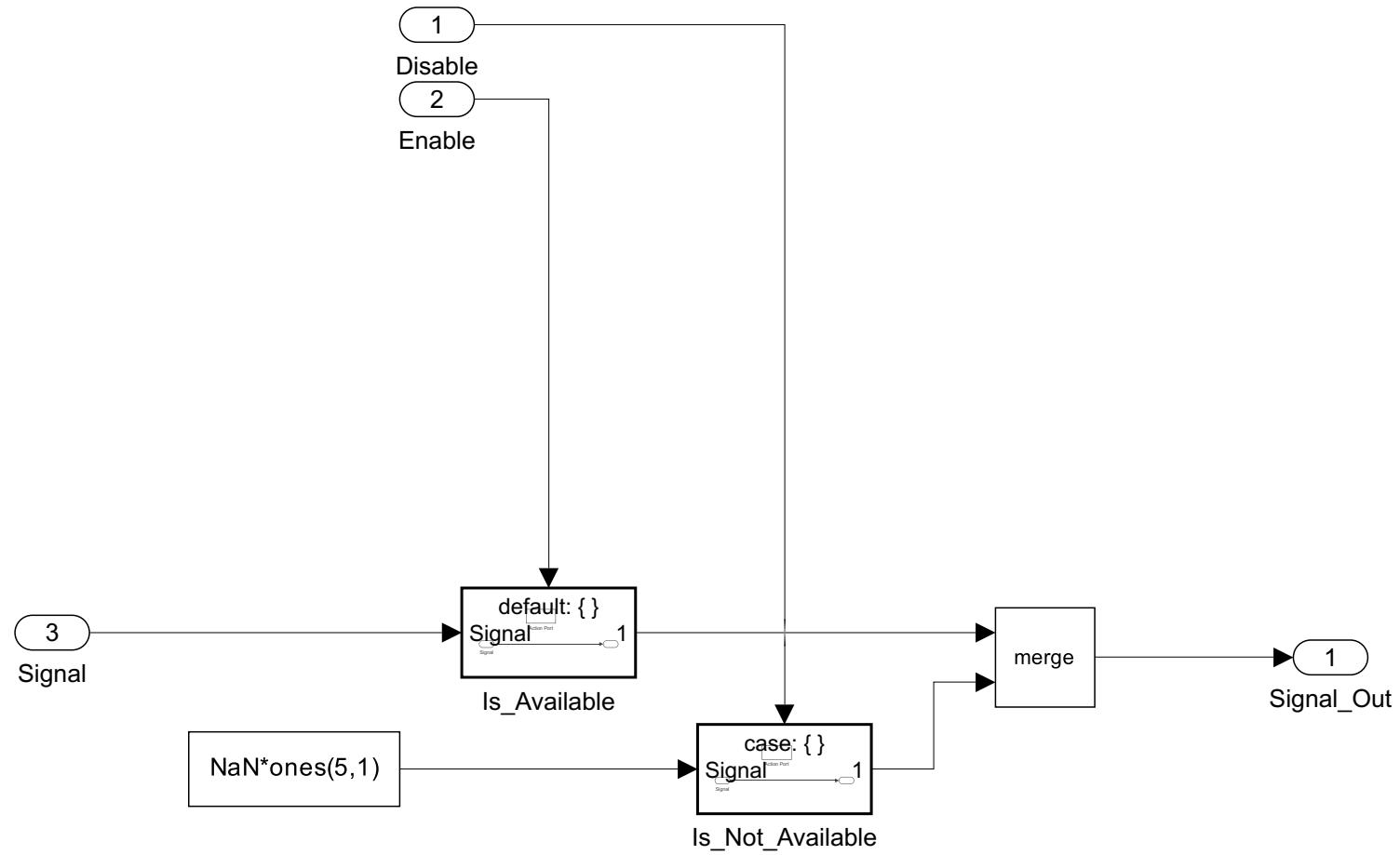
euler_angle_rate = [p + (q*sin(roll)+r*cos(roll))*tan(pitch),
                    q*cos(roll)-r*sin(roll),
                    (q*sin(roll)+r*cos(roll))/cos(pitch))];

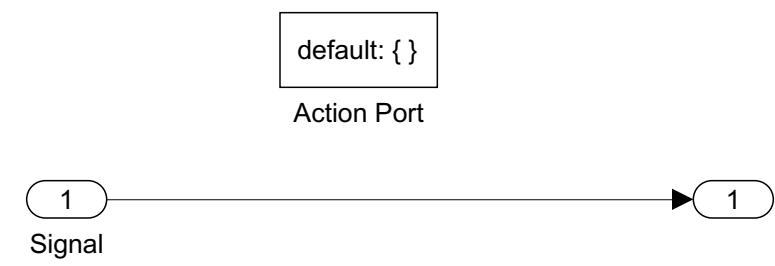
```

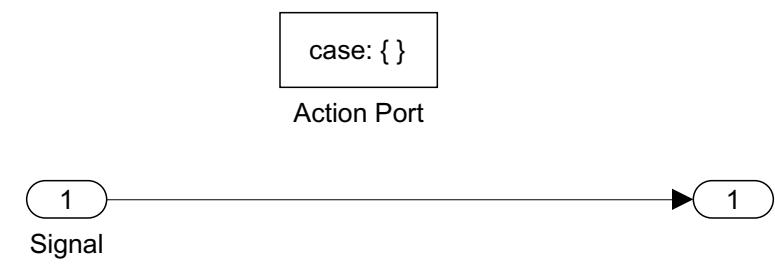


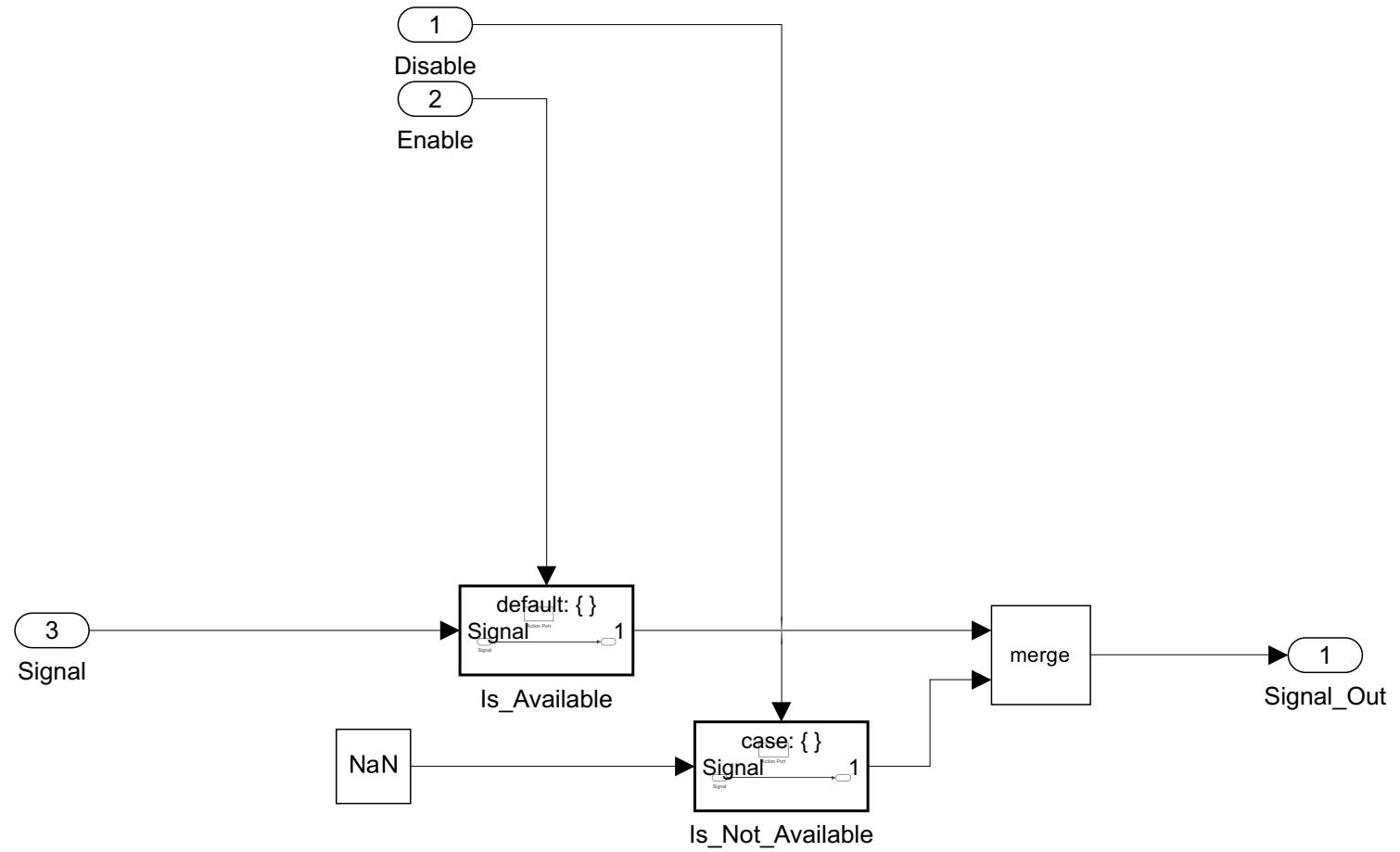












default: { }

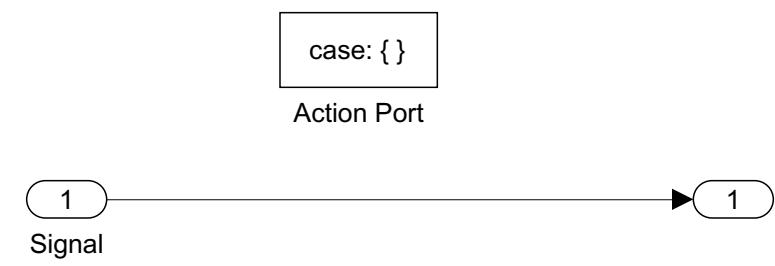
Action Port

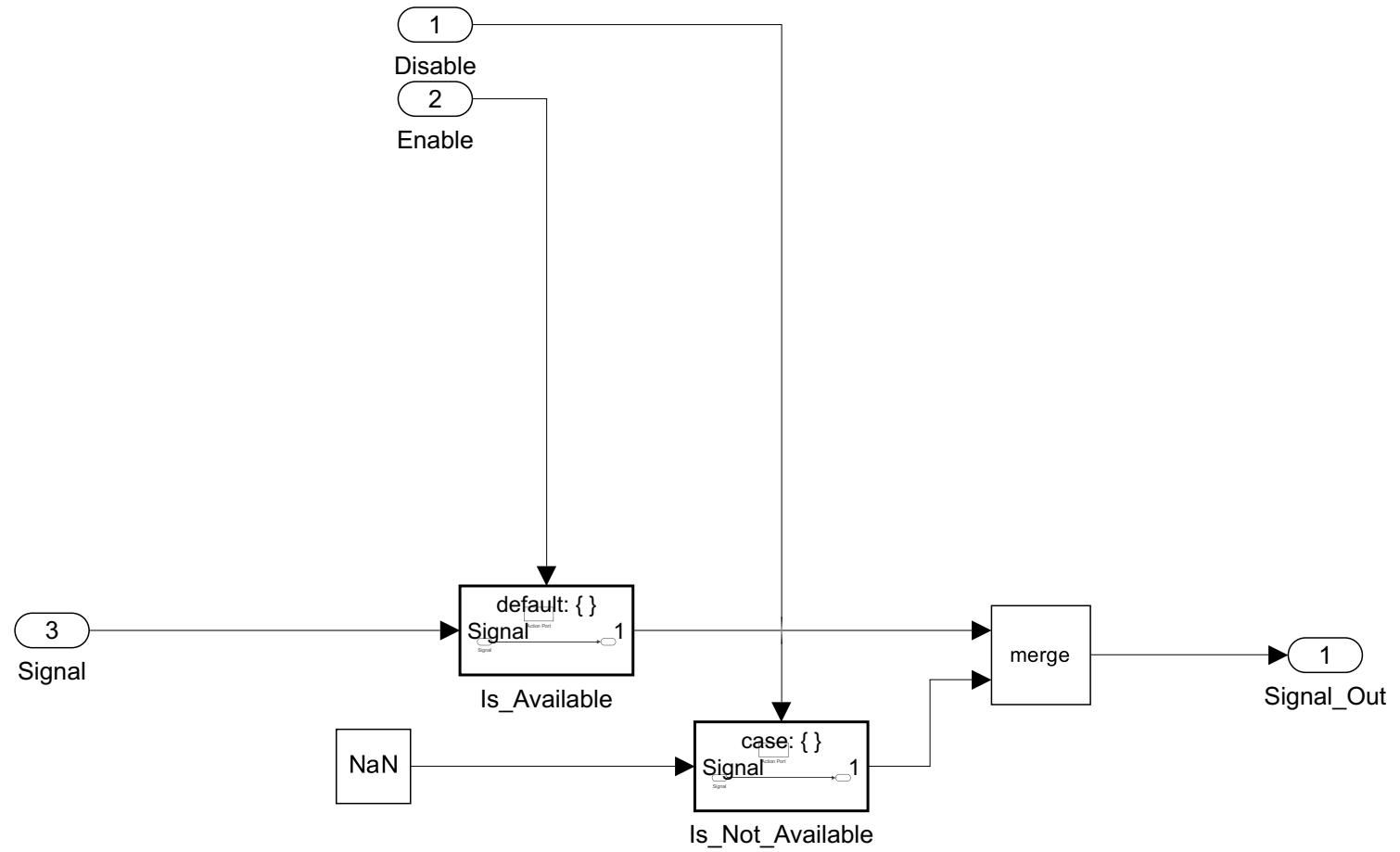


1

Signal

1





default: { }

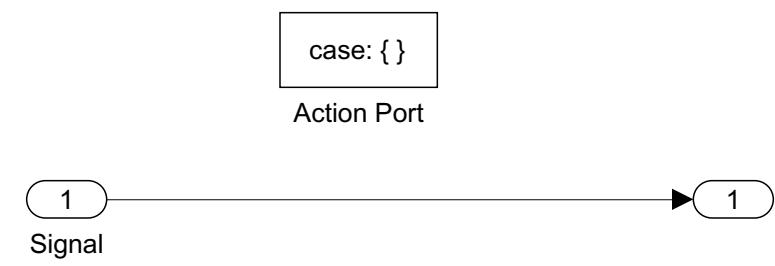
Action Port

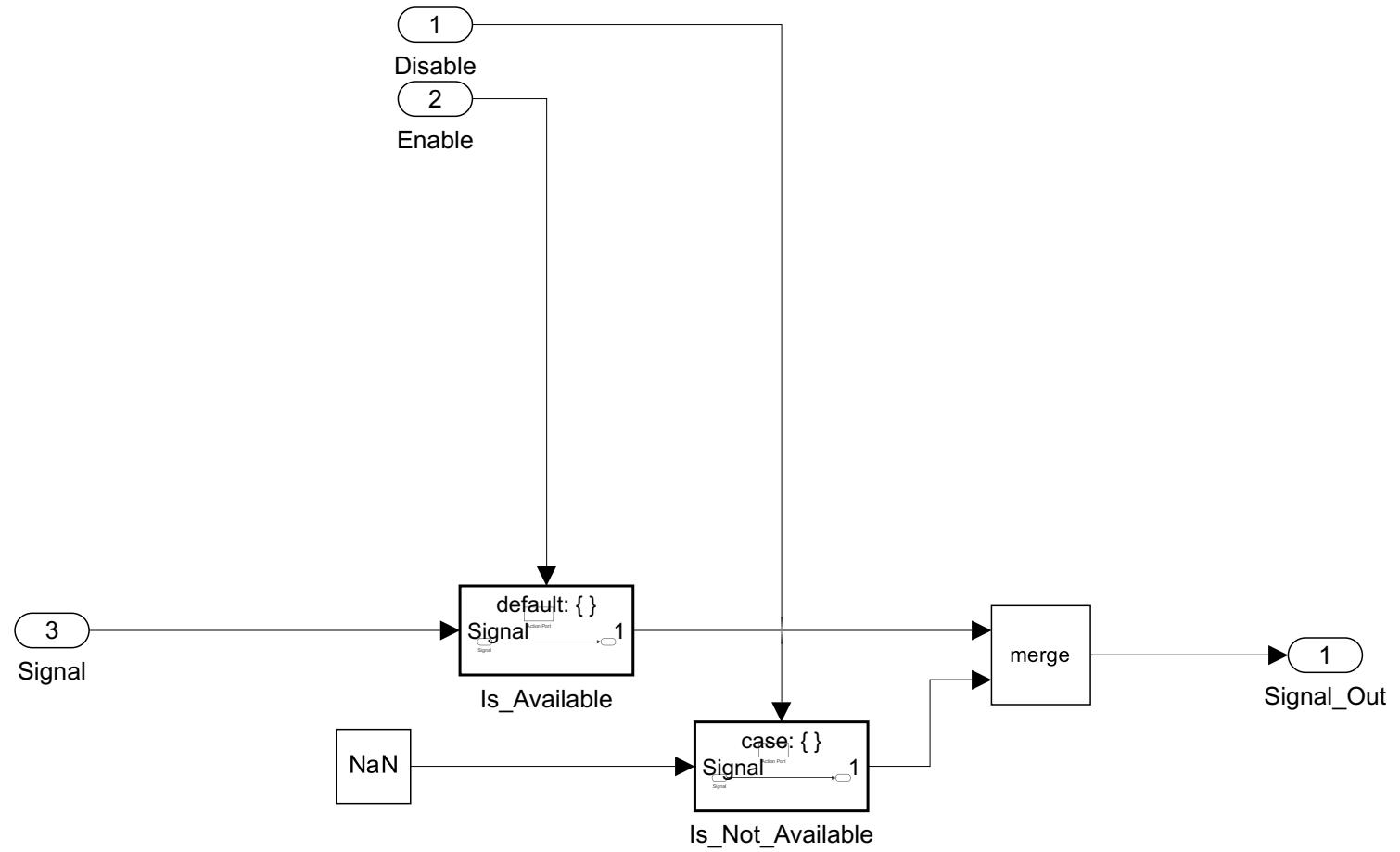


1

Signal

1





default: { }

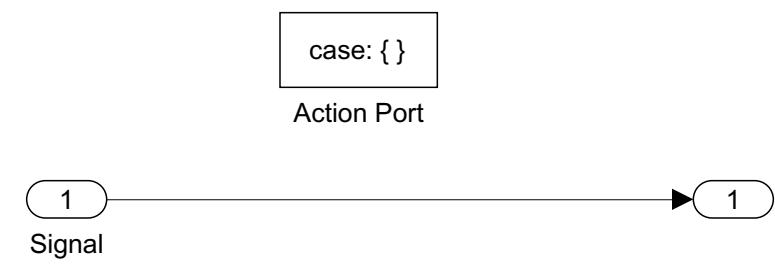
Action Port



1

Signal

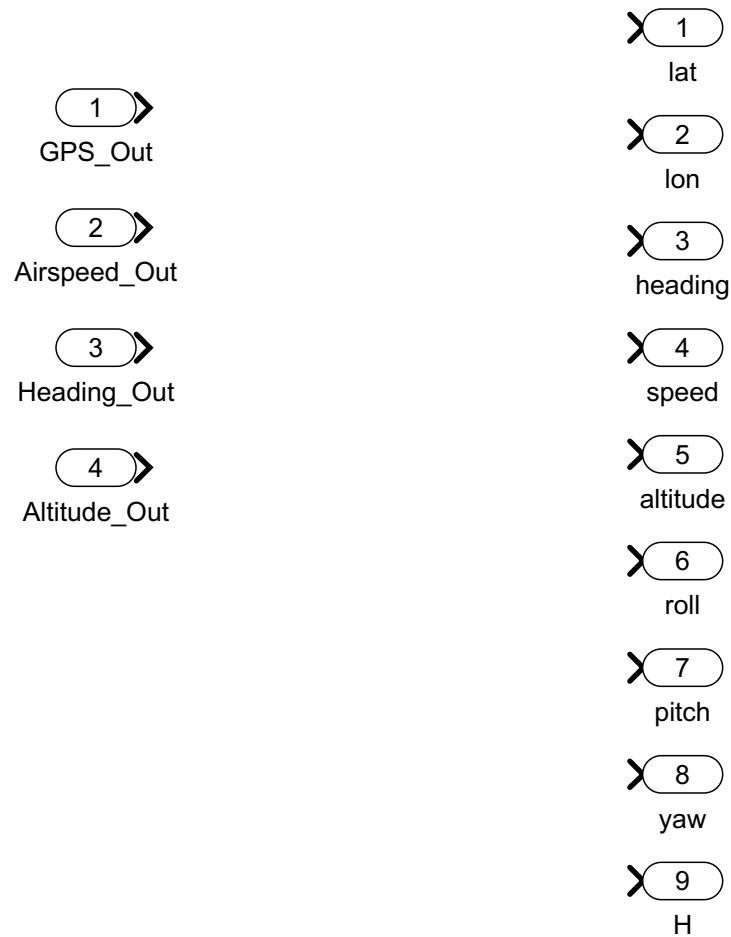
1

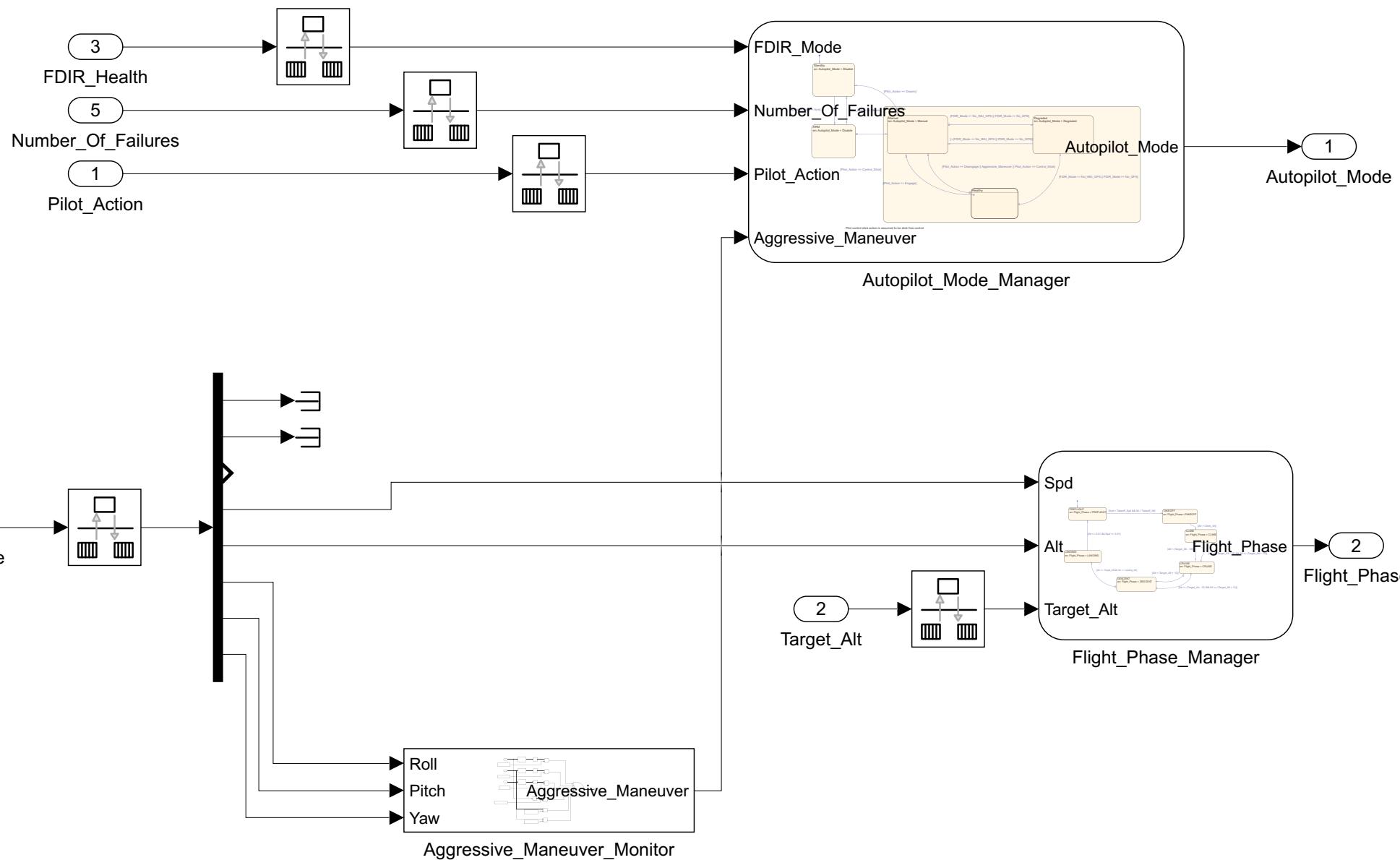


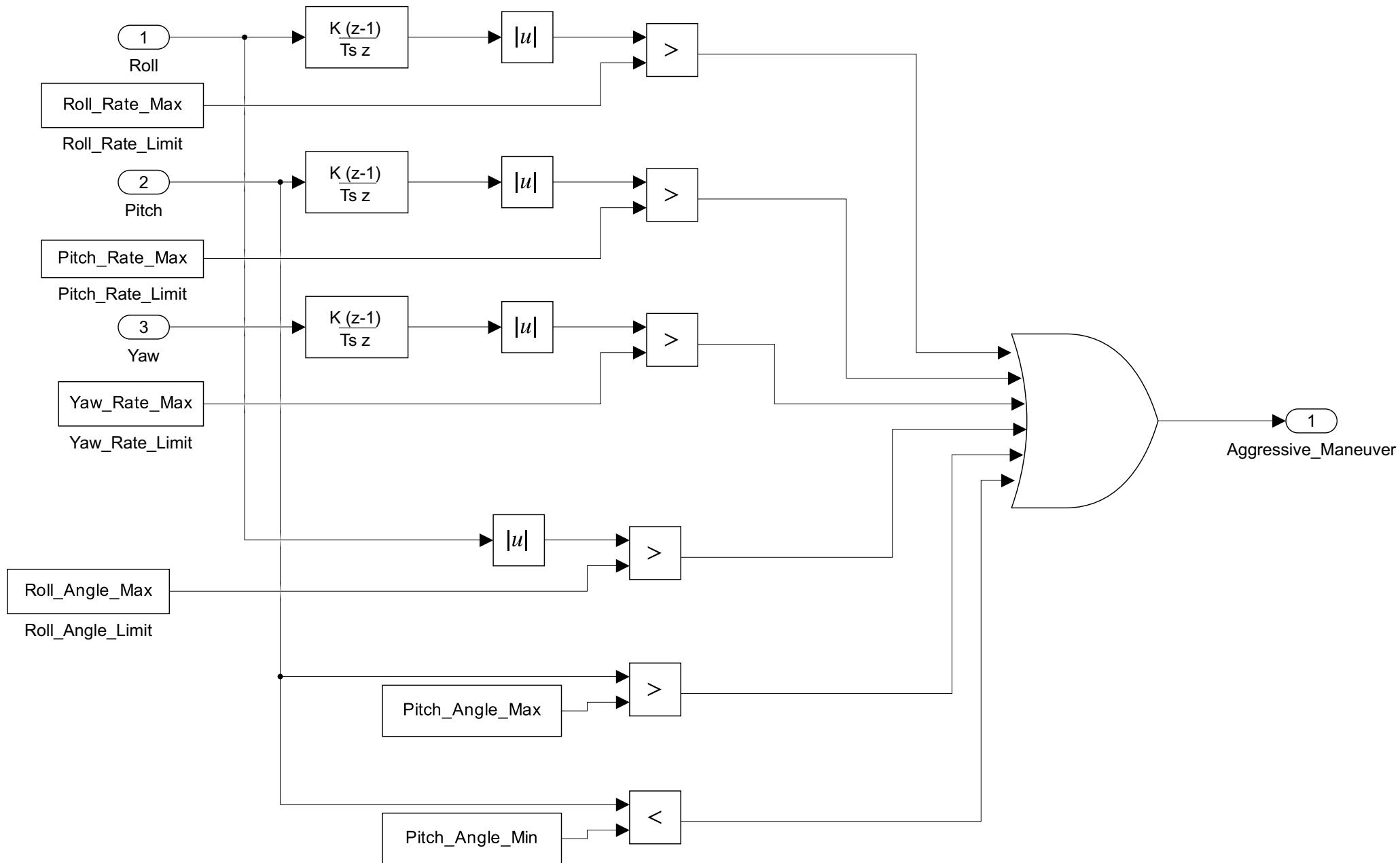
Dummy_Implementation

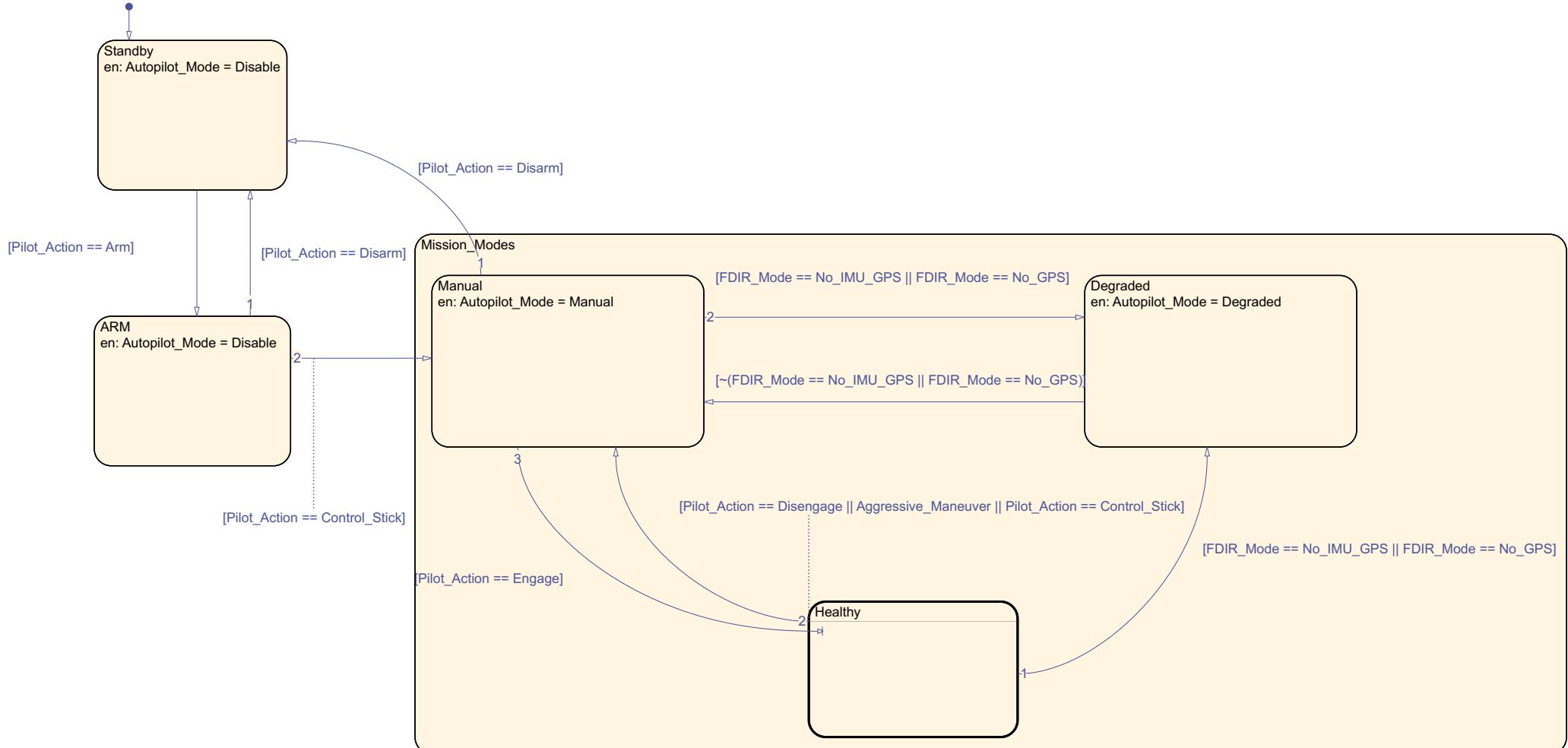
Logic needs to be implemented to pick the sensor fusion quantities based on quality of the data (Airspeed from pitot tube is more preferable than GPS speed when both are available) and filter out the NaN signals

Entries of H matrix for measurement equation can be made zero to not consider them for measurement update









Pilot control stick action is assumed to be stick free control

Healthy
du: Is_Autonomous_Mode_Possible = FDIR_Mode == No_GPS && Number_Of_Failures < 2

