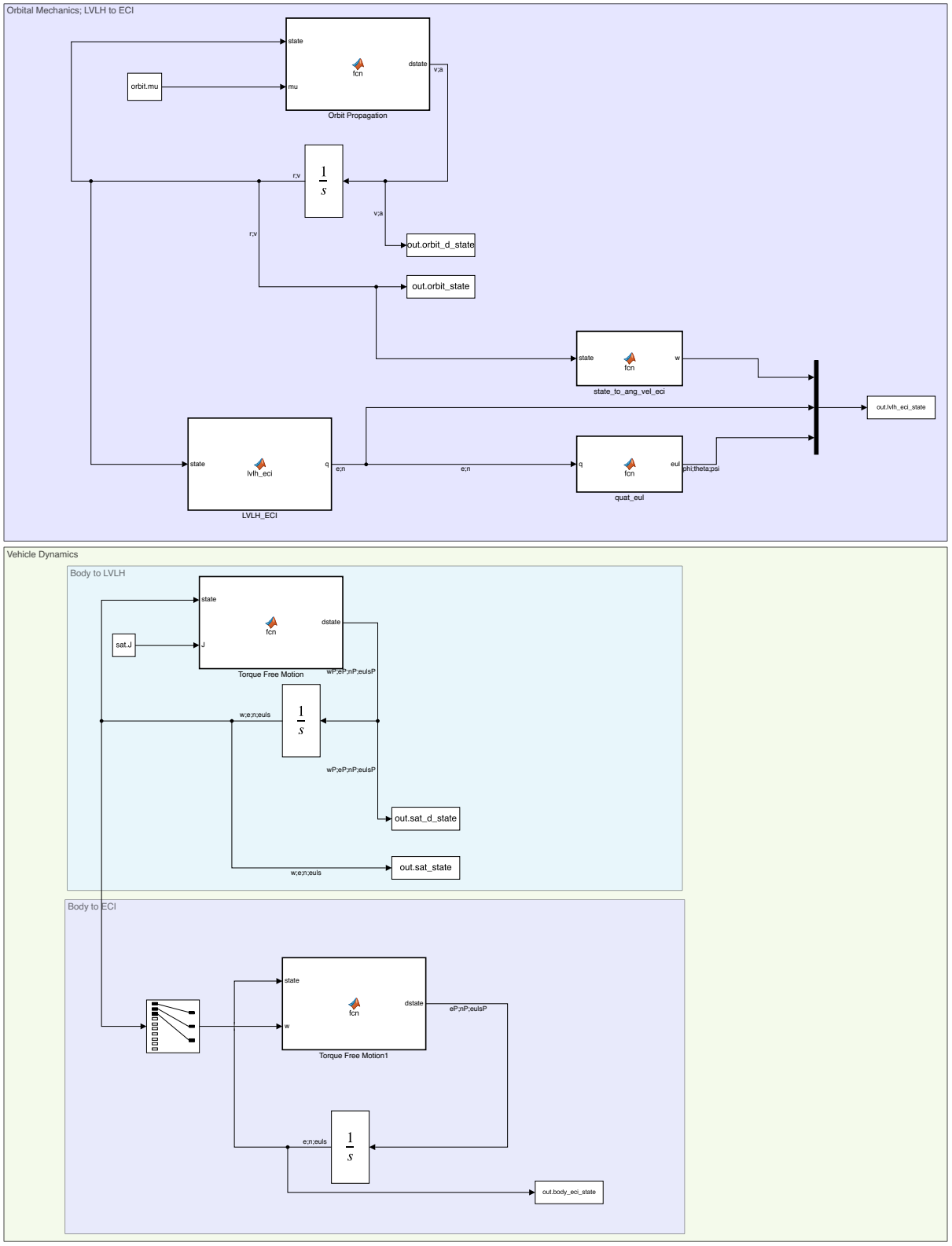
Orbital Mechanics; LVLH to ECI
state
fcn
dstate
v;a
orbit.mu
mu
Orbit Propagation
r;v
1
s
v;a
r;v
out.orbit_d_state
out.orbit_state
state
fcn
w
state_to_ang_vel_eci
out.lvlh_eci_state
state
lvlh_eci
q
e;n
e;n
q
fcn
eul
phi;theta;psi
quat_eul
LVLH_ECI
Vehicle Dynamics
Body to LVLH
state
fcn
dstate
sat.J
J
Torque Free Motion
wP;eP;nP;eulsP
w;e;n;euls
1
s
wP;eP;nP;eulsP
out.sat_d_state
w;e;n;euls
out.sat_state
Body to ECI
state
fcn
dstate
eP;nP;eulsP
w
Torque Free Motion1
e;n;euls
1
s
out.body_eci_state

```
function q = lvlh_eci(state)

    function Q = ECILVLH(R,V)

    if size(R) == [1 3]
        R = R';
    end

    if size(V) == [1 3]
        V = V';
    end

    z = -1*R / norm(R);
    y = -1 * cross(R,V) / norm(cross(R,V));

    x = cross(y,z);

    Q = [x,y,z]';

    end

    function [e, n] = c2quat(C)
        e = zeros(3,1);
        n = 0.5 * sqrt(1 + trace(C));
        e(1) = 0.25 * (C(2,3) - C(3,2))/n;
        e(2) = 0.25 * (C(3,1) - C(1,3))/n;
        e(3) = 0.25 * (C(1,2) - C(2,1))/n;

    end

R = state(1:3);
V = state(4:6);

Q = ECILVLH(R,V);
[e,n] = c2quat(Q);

e = -1*e;
n = -1*n;

q = [e(1);e(2);e(3);n];

end
```

```
function dstate = fcn(state, mu)

    dstate = zeros(6,1);

    rx = state(1);
    ry = state(2);
    rz = state(3);
    vx = state(4);
    vy = state(5);
    vz = state(6);

    rad = norm([rx;ry;rz]);

    ax = -mu*rx/rad^3;
    ay = -mu*ry/rad^3;
    az = -mu*rz/rad^3;

    dstate(1:3) = [vx;vy;vz];
    dstate(4:6) = [ax;ay;az];

end
```

```
function dstate = fcn(state, J)

    function wx = skewSymmetric(w)
        wx = [0, -1*w(3), w(2);
             w(3), 0, -1*w(1);
             -1*w(2), w(1), 0];
    end

    w = state(1:3);
    e = state(4:6);
    n = state(7);
    euls = state(8:10);

    wP = -1*J\(skewSymmetric(w)*J*w);
    eP = 0.5*(n*eye(3) + skewSymmetric(e))*w;
    nP = -0.5*e'*w;


    phi = euls(1);
    theta = euls(2);
    psi = euls(3);
    eulsP = 1/(cos(theta)) * [cos(theta), sin(phi)*sin(theta), cos(phi)*sin(theta);
                              0, cos(phi)*cos(theta), -1*sin(phi)*cos(theta);
                              0, sin(phi), cos(phi)] * w;


    dstate = [wP(1);wP(2);wP(3);eP(1);eP(2);eP(3);nP; eulsP(1); eulsP(2); eulsP(3)];


end
```

```
function dstate = fcn(state, w)

    function wx = skewSymmetric(w)
        wx = [0, -1*w(3), w(2);
              w(3), 0, -1*w(1);
              -1*w(2), w(1), 0];
    end

    e = state(1:3);
    n = state(4);
    euls = state(5:7);

    eP = 0.5*(n*eye(3) + skewSymmetric(e))*w;
    nP = -0.5*e'*w;


    phi = euls(1);
    theta = euls(2);
    psi = euls(3);
    eulsP = 1/(cos(theta)) * [cos(theta), sin(phi)*sin(theta), cos(phi)*sin(theta);
                              0, cos(phi)*cos(theta), -1*sin(phi)*cos(theta);
                              0, sin(phi), cos(phi)] * w;


    dstate = [eP(1);eP(2);eP(3);nP; eulsP(1); eulsP(2); eulsP(3)];


end
```

```
function eul = fcn(q)

    n = q(4);
    e = q(1:3);

    q = [n, e(1), e(2), e(3)];

    phi = atan2(2*(q(1)*q(2) + q(3)*q(4)), 1 - 2*(q(2)^2 + q(3)^2));
    theta = asin(2*(q(1)*q(3) - q(4)*q(2)));
    psi = atan2(2*(q(1)*q(4) + q(2)*q(3)), 1-2*(q(3)^2 + q(4)^2));

    eul = [phi; theta; psi];

end
```

```
function w = fcn(state)

function Q = ECILVLH(R,V)

        if size(R) == [1 3]
            R = R';
        end

        if size(V) == [1 3]
            V = V';
        end

        z = -1*R / norm(R);
        y = -1 * cross(R,V) / norm(cross(R,V));

        x = cross(y,z);

        Q = [x,y,z]';

    end

r = [state(1);state(2);state(3)];
v = [state(4);state(5);state(6)];

w = ECILVLH(r,v) * cross(r,v)/(norm(r)^2);

w = [w(1);w(2);w(3)];

end
```