```matlab
function [sys,x0,str,ts,simStateCompliance] = mass_dynamics(t,x,u,flag,P)
switch flag,

  %%%%%%%%%%%%%%%%%%
  % Initialization %
  %%%%%%%%%%%%%%%%%%
  case 0,
    [sys,x0,str,ts,simStateCompliance]=mdlInitializeSizes(P);

  %%%%%%%%%%%%%%%
  % Derivatives %
  %%%%%%%%%%%%%%%
  case 1,
    sys=mdlDerivatives(t,x,u,P);

  %%%%%%%%%%
  % Update %
  %%%%%%%%%%
  case 2,
    sys=mdlUpdate(t,x,u);

  %%%%%%%%%%%
  % Outputs %
  %%%%%%%%%%%
  case 3,
    sys=mdlOutputs(t,x,u);

  %%%%%%%%%%%%%%%%%%%%%%%
  % GetTimeOfNextVarHit %
  %%%%%%%%%%%%%%%%%%%%%%%
  case 4,
    sys=mdlGetTimeOfNextVarHit(t,x,u);

  %%%%%%%%%%%%%
  % Terminate %
  %%%%%%%%%%%%%
  case 9,
    sys=mdlTerminate(t,x,u);

  %%%%%%%%%%%%%%%%%%%%
  % Unexpected flags %
  %%%%%%%%%%%%%%%%%%%%
  otherwise
    DAStudio.error('Simulink:blocks:unhandledFlag', num2str(flag));

end

% end sfuntmpl

%
%=================================================================
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the
% S-function.
%=================================================================
%
function [sys,x0,str,ts,simStateCompliance]=mdlInitializeSizes(P)

sizes = simsizes;

sizes.NumContStates  = 2;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 1;
sizes.NumInputs      = 1;
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;   % at least one sample time is needed

sys = simsizes(sizes);

%
% initialize the initial conditions
%
x0  = [P.z0; P.zdot0];

%
% str is always an empty matrix
%
str = [];

%
% initialize the array of sample times
%
ts  = [0 0];

simStateCompliance = 'UnknownSimState';

% end mdlInitializeSizes

%
%================================================================
% mdlDerivatives
% Return the derivatives for the continuous states.
%================================================================
%
function sys=mdlDerivatives(t,x,u,P)
  z    = x(1);
  zdot = x(2);
  F    = u(1);

  zddot = (1/P.m)*(F-P.b*zdot-P.k*z);

sys = [zdot; zddot];

% end mdlDerivatives

%
%================================================================
% mdlUpdate
% Handle discrete state updates, sample time hits, and major time
% step requirements.
%================================================================
%
function sys=mdlUpdate(t,x,u)

sys = [];

% end mdlUpdate

%
%================================================================
% mdlOutputs
% Return the block outputs.
%================================================================
%
function sys=mdlOutputs(t,x,u)
  z    = x(1);
```

```
sys = z;

% end mdlOutputs

%
%=================================================================
% mdlGetTimeOfNextVarHit

%=================================================================
%
function sys=mdlGetTimeOfNextVarHit(t,x,u)

sampleTime = 1;
sys = t + sampleTime;

% end mdlGetTimeOfNextVarHit

%
%=================================================================
% mdlTerminate
% Perform any end of simulation tasks.
%=================================================================
%
function sys=mdlTerminate(t,x,u)

sys = [];

% end mdlTerminate
```