find_eph.m

```
function icol = find_eph(Eph,sv,time)
%FIND_EPH    Finds the proper column in ephemeris array

%Kai Borre and C.C. Goad 11-26-96
%Copyright (c) by Kai Borre
%$Revision: 1.1 $  $Date: 1998/07/01  $

icol = 0;
isat = find(Eph(1,:) == sv);
n = size(isat,2);
if n == 0
   break
end;
icol = isat(1);
dtmin = Eph(21,icol)-time;
for t = isat
   dt = Eph(21,t)-time;
   if dt < 0
      if abs(dt) < abs(dtmin)
         icol = t;
         dtmin = dt;
      end
   end
end
```

satpos.m

```
function satp = satpos(t,eph);
%SATPOS Calculation of X,Y,Z coordinates at time t
%       for given ephemeris eph

%Kai Borre 04-09-96
%Copyright (c) by Kai Borre
%$Revision: 1.1 $  $Date: 1997/12/06  $

GM = 3.986005e14;          % earth's universal gravitational
                           % parameter m^3/s^2
Omegae_dot = 7.2921151467e-5; % earth rotation rate, rad/s

%  Units are either seconds, meters, or radians
%  Assigning the local variables to eph
svprn    =    eph(1);
af2    =    eph(2);
M0       =    eph(3);
roota    =    eph(4);
deltan   =    eph(5);
ecc    =    eph(6);
omega    =    eph(7);
```

```
cuc    =   eph(8);
cus    =   eph(9);
crc    =  eph(10);
crs    =  eph(11);
i0       =  eph(12);
idot     =  eph(13);
cic    =  eph(14);
cis    =  eph(15);
Omega0  =  eph(16);
Omegadot=  eph(17);
toe    =  eph(18);
af0    =  eph(19);
af1    =  eph(20);
toc    =  eph(21);

% Procedure for coordinate calculation
A = roota*roota;
tk = check_t(t-toe);
n0 = sqrt(GM/A^3);
n = n0+deltan;
M = M0+n*tk;
M = rem(M+2*pi,2*pi);
E = M;
```

Solution to EASY2

```
for i = 1:10
   E_old = E;
   E = M+ecc*sin(E);
   dE = rem(E-E_old,2*pi);
   if abs(dE) < 1.e-12
      break;
   end
end
E = rem(E+2*pi,2*pi);
v = atan2(sqrt(1-ecc^2)*sin(E), cos(E)-ecc);
phi = v+omega;
phi = rem(phi,2*pi);
u = phi           + cuc*cos(2*phi)+cus*sin(2*phi);
r = A*(1-ecc*cos(E)) + crc*cos(2*phi)+crs*sin(2*phi);
i = i0+idot*tk       + cic*cos(2*phi)+cis*sin(2*phi);
Omega = Omega0+(Omegadot-Omegae_dot)*tk-Omegae_dot*toe;
Omega = rem(Omega+2*pi,2*pi);
x1 = cos(u)*r;
y1 = sin(u)*r;
satp(1,1) = x1*cos(Omega)-y1*cos(i)*sin(Omega);
satp(2,1) = x1*sin(Omega)+y1*cos(i)*cos(Omega);
satp(3,1) = y1*sin(i);
```

check_t.m

```
function tt = check_t(t);
%CHECK_T  accounting for beginning or end of week crossover

%Kai Borre 04-01-96
%Copyright (c) by Kai Borre
%$Revision: 1.1 $  $Date: 1998/10/28  $

half_week = 302400;
tt = t;
if t >  half_week
   tt = t-2*half_week;
end
if t < -half_week
   tt = t+2*half_week;
end
```

easy2.m

```
%EASY2  Convert observation time into sow.
%        We read the corresponding RINEX navigation file
%        and reformat the data into the Matlab matrix Eph.
%        For given SV we find the corresponding column in Eph
%        and call the basic satpos function

%Kai Borre 27-07-2002
%Copyright (c) by Kai Borre
%$Revision: 1.0 $  $Date: 2002/07/27  $

% Compute sow for first epoch in observation file site247j.01o
% 01  9  4  9 40  0.0000000  0  7G 1G 4G 7G13G20G24G25
jd = julday(2001,9,4,9+40/60);
[week,sow] = gps_time(jd);

% Read RINEX ephemerides file and convert to
% internal Matlab format
rinexe('SITE247J.01N','eph.dat');
Eph = get_eph('eph.dat');
```

```
% We identify the observed satellites in line 29 of RINEX file site247j.01o
% 01  9  4  9 40  0.0000000  0  7G 1G 4G 7G13G20G24G25
svs = [1 4 7 13 20 24 25];
for t = 1:length(svs)
    col_Eph(t) = find_eph(Eph,svs(t),sow);
    sat(1:3,t) = satpos(sow,Eph(:,col_Eph(t)));
end

sat     % position of svs in ECI system
```

Solution to EASY2

```
>> easy2
head_lines =
          8.00
noeph =
          7.00
status =
             0
ans =
             0
sat =
   14789352.27   11785007.12   20131358.72   22053406.80   12654414.18   -1514425.21   -9091424.31
    7334724.92  -10589687.83  -17092030.49   -4245755.32   17685504.16  -16394867.10   13349700.37
   20976567.34   21426953.64    1367003.86   14103438.07   15150293.42   21142937.47   21347308.03
```

Solution to EASY2