


8Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

Институт: *Радиотехники и* Кафедра: *Радиотехнических систем*
электроники

Направление подготовки: 11.05.01 - Радиоэлектронные системы и
комплексы

ОТЧЕТ по курсовой работе

СТУДЕНТ

 / Лебедев Д.Д. /
(подпись) (Фамилия и инициалы)

Группа ЭР-15-16
(номер учебной группы)

ЗАЩИТА КУРСОВОЙ РАБОТЫ

(отлично, хорошо, удовлетворительно, неудовлетворительно,
зачтено, не зачтено)

/ Корогодин И.В. /
(подпись) (Фамилия и инициалы члена комиссии)

/ Шатилов А.Ю. /
(подпись) (Фамилия и инициалы члена комиссии)

**Москва
2021**

Постановка ЗАДАЧИ

В данной работе мы знакомимся с рядом инструментов и техник, используемых при разработке навигационных приемников.

Цель проекта - получить библиотеку функций на Си++, позволяющую рассчитывать положение спутника Beidou по его эфемеридам. На первом этапе подготовим вспомогательные данные для разработки: эфемериды и оценки положения спутника от сторонних сервисов (чтобы было с чем сравниваться на след. этапах)

Требования к разрабатываемому программному модулю:

- требования назначения;
- отсутствие утечек памяти;
- малое время выполнения;
- низкий расход памяти;
- корректное выполнение при аномальных входных данных.

Для достижения цели выполняется ряд задач, соответствующих этапам проекта и контрольным мероприятиям:

- обработка данных от приемника, работа со сторонними сервисами для подготовки входных и проверочных данных для разрабатываемого модуля;
- моделирование модуля в Matlab/Python;
- реализация программного модуля на C/C++, включая юнит-тестирование в Check.

Этапы курсовой работы отличаются осваиваемыми инструментами.

Введение

Издревле человек искал способы как добраться до определенного места, искал ориентиры, плыл по звездам, пользовался позиционными методами определения своего местоположения и т.п. На сегодняшний день любой человек имея смартфон в кармане, может определить где он находится и как ему добраться до пункта назначения. Все это возможно благодаря СРНС. Они стали неотъемлемой частью жизни в различных сферах. Наиболее распространенными являются системы ГЛОНАСС (Россия), GPS (США), Galileo (Евросоюз), Beidou (Китай).

Цель проекта - добавление в программное обеспечение приемника функции расчета положения спутника Beidou на заданное время по данным его эфемерид.

Требования к разрабатываемому программному модулю:

- 1) требования назначения;
- 2) отсутствие утечек памяти;
- 3) малое время выполнения;
- 4) низкий расход памяти;
- 5) корректное выполнение при аномальных входных данных.

Для достижения цели выполняется ряд задач, соответствующих этапам проекта и контрольным мероприятиям:

- 1) обработка данных от приемника, работа со сторонними сервисами для подготовки входных и проверочных данных для разрабатываемого модуля;
- 2) моделирование модуля в Matlab/Python;
- 3) реализация программного модуля на C/C++, включая юниттестирование в Check. Этапы курсовой работы отличаются осваиваемыми инструментами.

ЭТАП 1. ИСПОЛЬЗОВАНИЕ СТОРОННИХ СРЕДСТВ

1.1 Общие сведения

Навигационная система «Бэйдоу» - китайская спутниковая система навигации.

В задание дан номер спутника BEIDOU- 10, а также бинарный и текстовый файл со значениями эфемерид для различных спутников, полученный от трехдиапазонной антенны Narxon HX-CSX601A, установленной на крыше корпуса Е МЭИ.

Она через 50-метровый кабель, сплиттер, bias-tee и усилитель подключена к трем навигационным приемникам:

- Javad Lexon LGDD,
- SwiftNavigation Piksi Multi,
- Clonicus разработки ЛНС МЭИ.

Эти приемники осуществляют первичную обработку сигналов Beidou B1I, выдавая по интерфейсам соответствующие потоки данных - наблюдения псевдодальностей и эфемериды спутников. Данные от приемника Clonicus, записанные 16 февраля 2021 года.

C10	37948	IGSO-5	BDS-2	02.12.11	3378	Используется по ЦН
-----	-------	--------	-------	----------	------	--------------------

Рисунок 1 – состояние 10-го спутника системы BEIDOU с
«Информационноаналитического центра координатно-временного и
навигационного обеспечения»

Из рисунка 1 можно выяснить, что спутник используется по цели назначению, его номер 37948, тип космического аппарата, тип системы и дату запуска.

1.2 Определение формы орбиты и положения спутника на ней с помощью сервиса CelesTrak

Для выполнения этого пункта перейдем на сайт CelesTrak (<https://celestrak.com>). Настроим параметры и выберем необходимый спутник, после чего будет построена Земля и орбита спутника вокруг нее (рисунок 2).

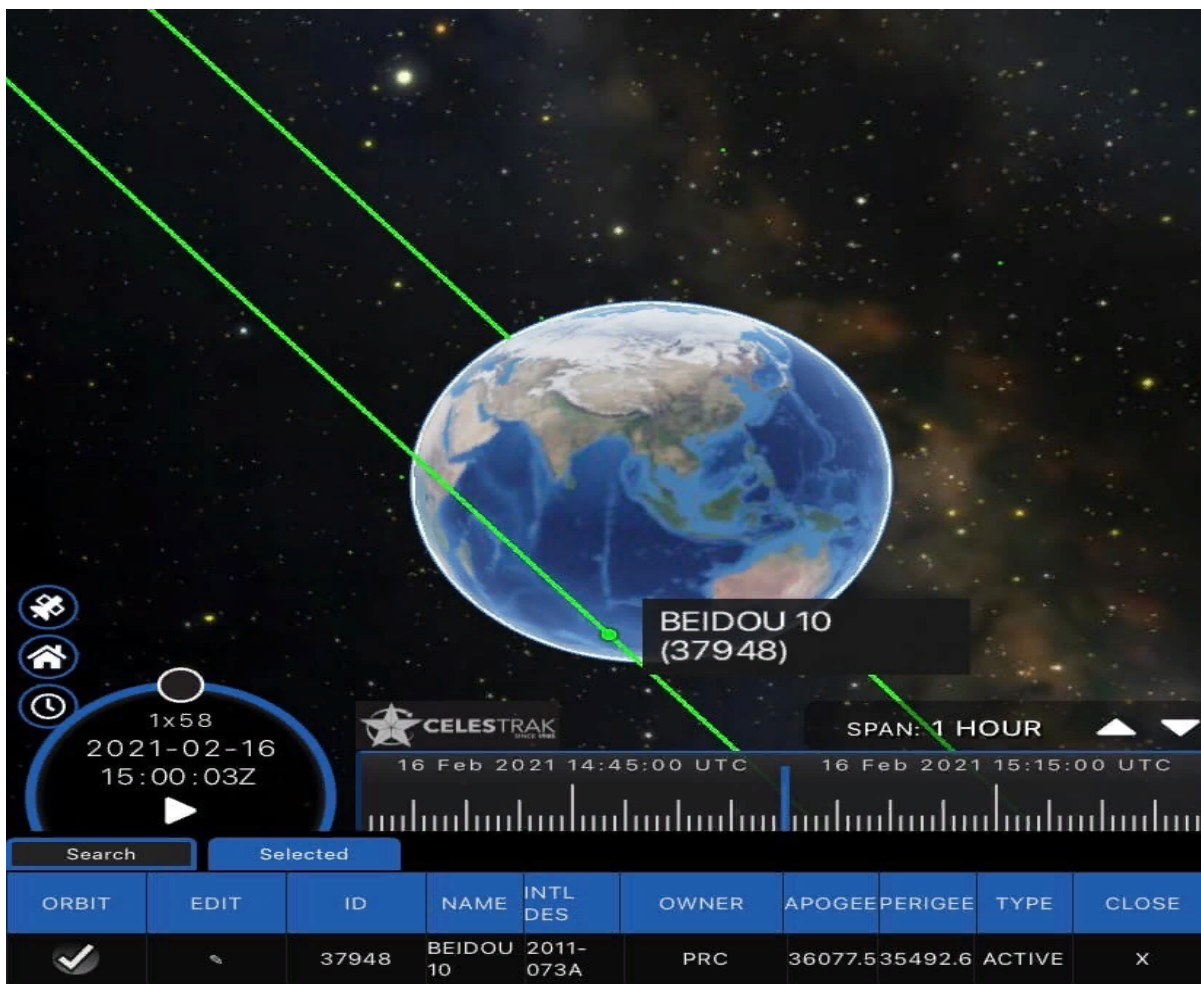


Рисунок 2 – Результат моделирования на CelesTrak

Значения совпадают, значит это действительно нужный нам спутник, проведем моделирование на момент времени 15:00, 16 февраля 2021, так как на данном сервисе отсчет времени происходит по UTC(+0).

1.3 Расчет графика угла места собственного спутника от времени по данным Trimble GNSS Planning Online

Зададим предварительные параметры для моделирования GNSS Planning Online, координаты установим в соответствии с расположением антенны – и они будут соответствовать значению корпуса Е, также начальное время будет соответствовать 15:00, временной пояс будет равен UTC (+0) на всем этапе моделирования в сервисе GNSS Planning Online, высота выбирается примерно, с учетом разных критериев и будет равна 150м.

The screenshot displays the 'Выбор спутника' (Satellite Selection) and 'Настройки' (Settings) sections of the Trimble GNSS Planning Online interface.

Выбор спутника (Satellite Selection):

Система:	активная	Выбранный	Здоровый
GPS	<input checked="" type="checkbox"/>	0	32
ГЛОНАСС	<input checked="" type="checkbox"/>	0	23
Галилей	<input checked="" type="checkbox"/>	0	20
BeiDou	<input checked="" type="checkbox"/>	1	49
QZSS	<input checked="" type="checkbox"/>	0	4

Мои Настройки (My Settings):

- Время альманаха: 2021-02-15
- Часовой пояс: UTC +00:00
- Видимый период: 2021-02-15 15:00 - 2021-02-16 15:00
- Широта: N 55° 45' 24.1764"
- Долгота: E 37° 42' 11.271"
- Высота: 150 м
- Отсечка высоты: 10 °

Настройки (Settings):

- Широта: N 55° 45' 24.1764"
- Долгота: E 37° 42' 11.271"
- Высота: 150 м
- Отсечка высоты: 10 °
- День: 16.02.2021
- Время начала: 15:00 UTC +00:00
- Период [часов]: 24
- Часовой пояс: (UTC) координированное универсальное время

Рисунок 3 – Моделирование с помощью сервиса Trimble GNSS Planning

В настройках ограничим выходные данные, дабы не загромождать графики и выбираем 10 спутник.

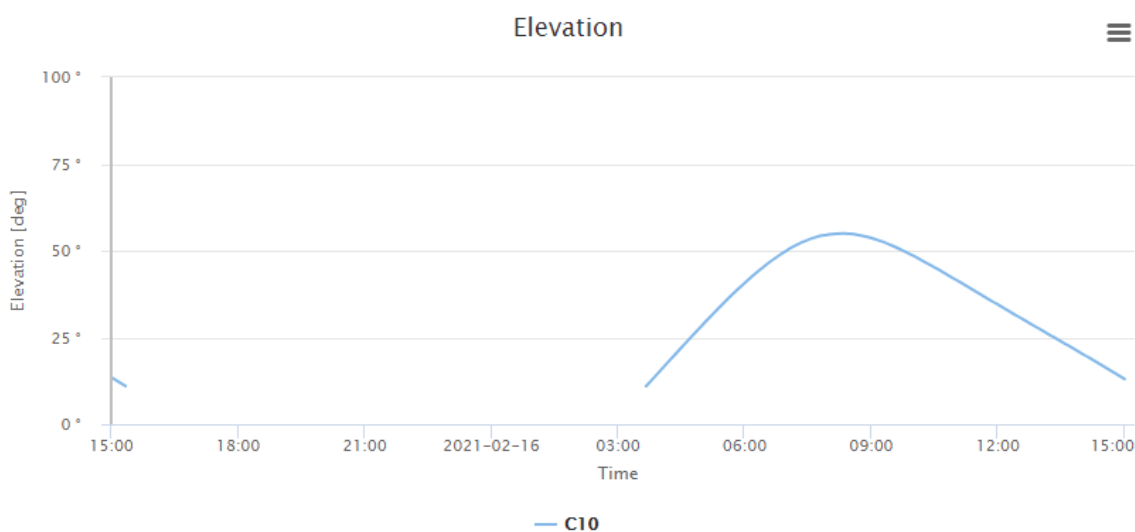


Рисунок 4 – График угла места спутника Beidou C10

По графику угла места можно сделать вывод, что спутник наблюдается с 3:40 до 15:10 по UTC(+0).

Далее переходим во вкладку Sky Plot и пользуемся возможностью пронаблюдать, траекторию спутника на небосводе в разное характерное время.

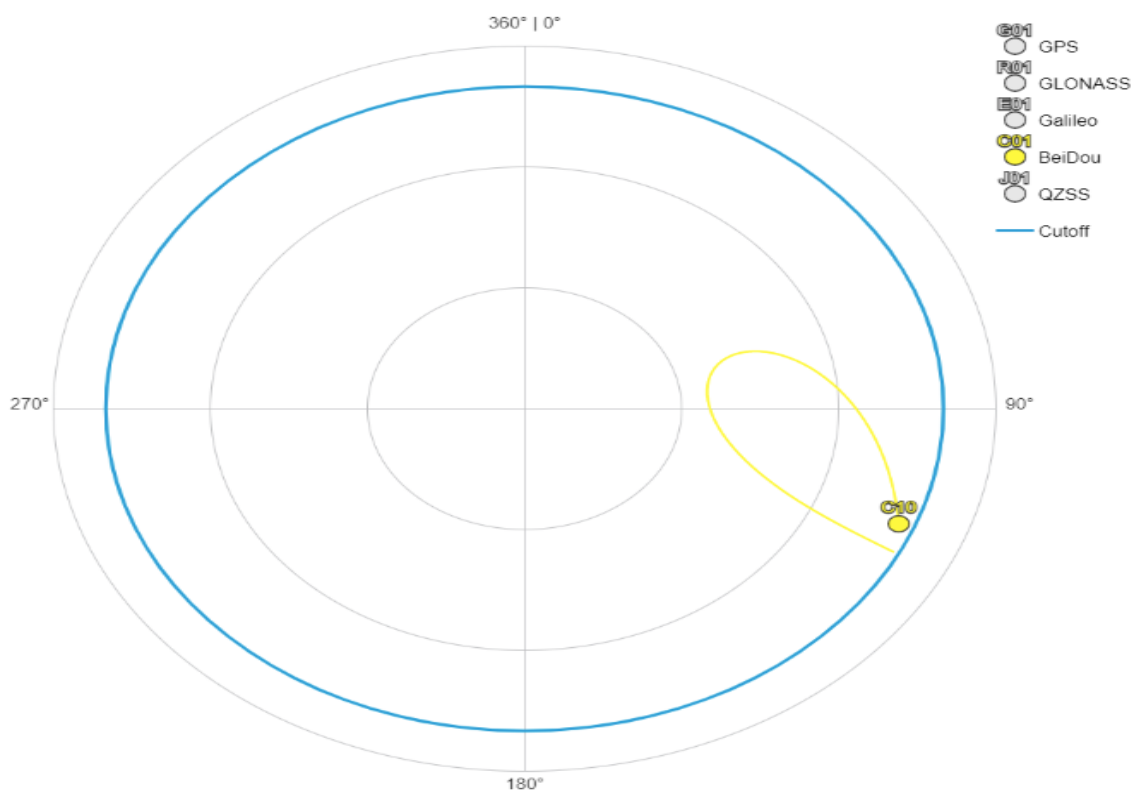


Рисунок 5 – SkyView спутника Beidou C10 в 15:00

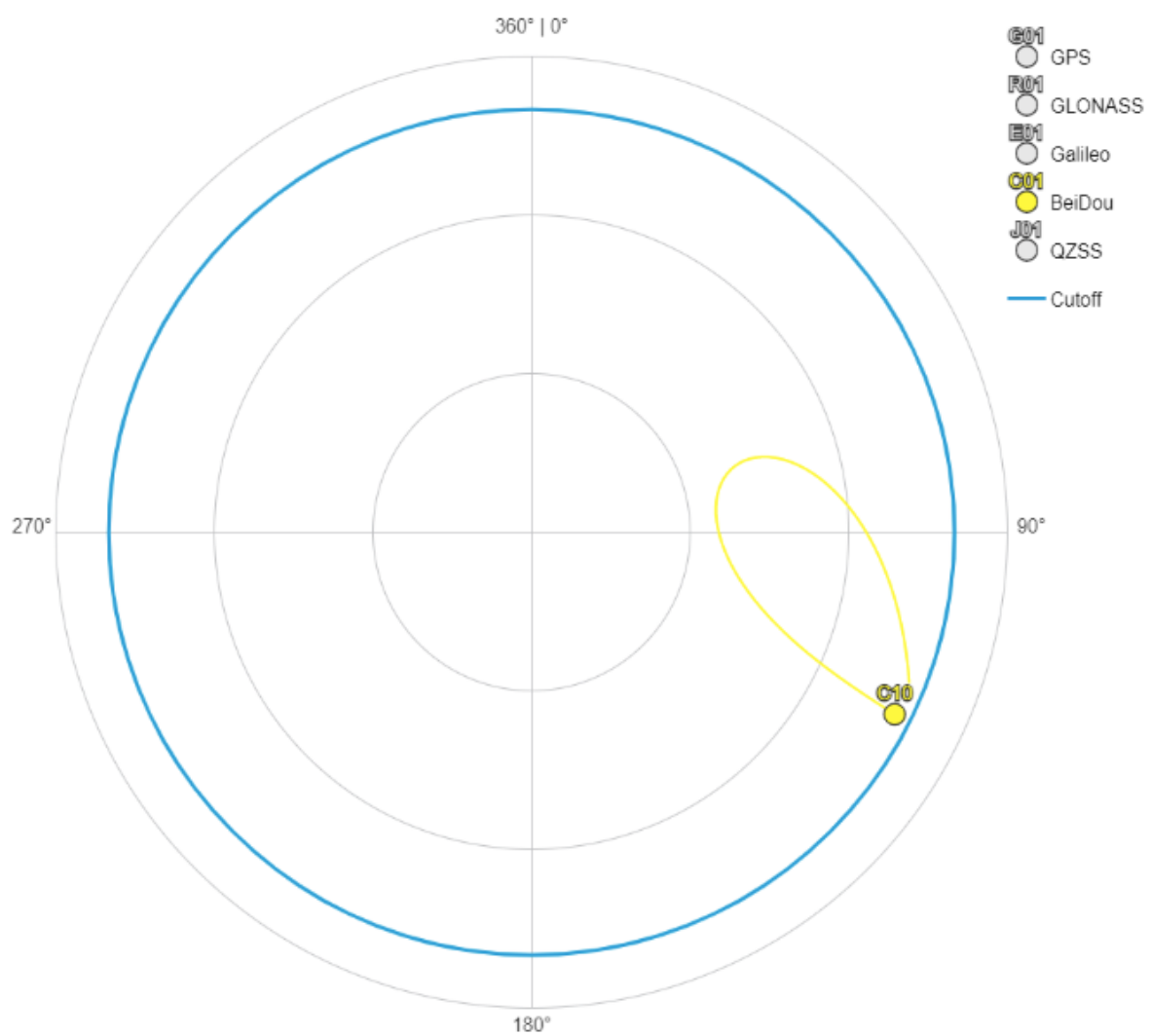


Рисунок 6 – SkyView спутника Beidou C10 в 3:40

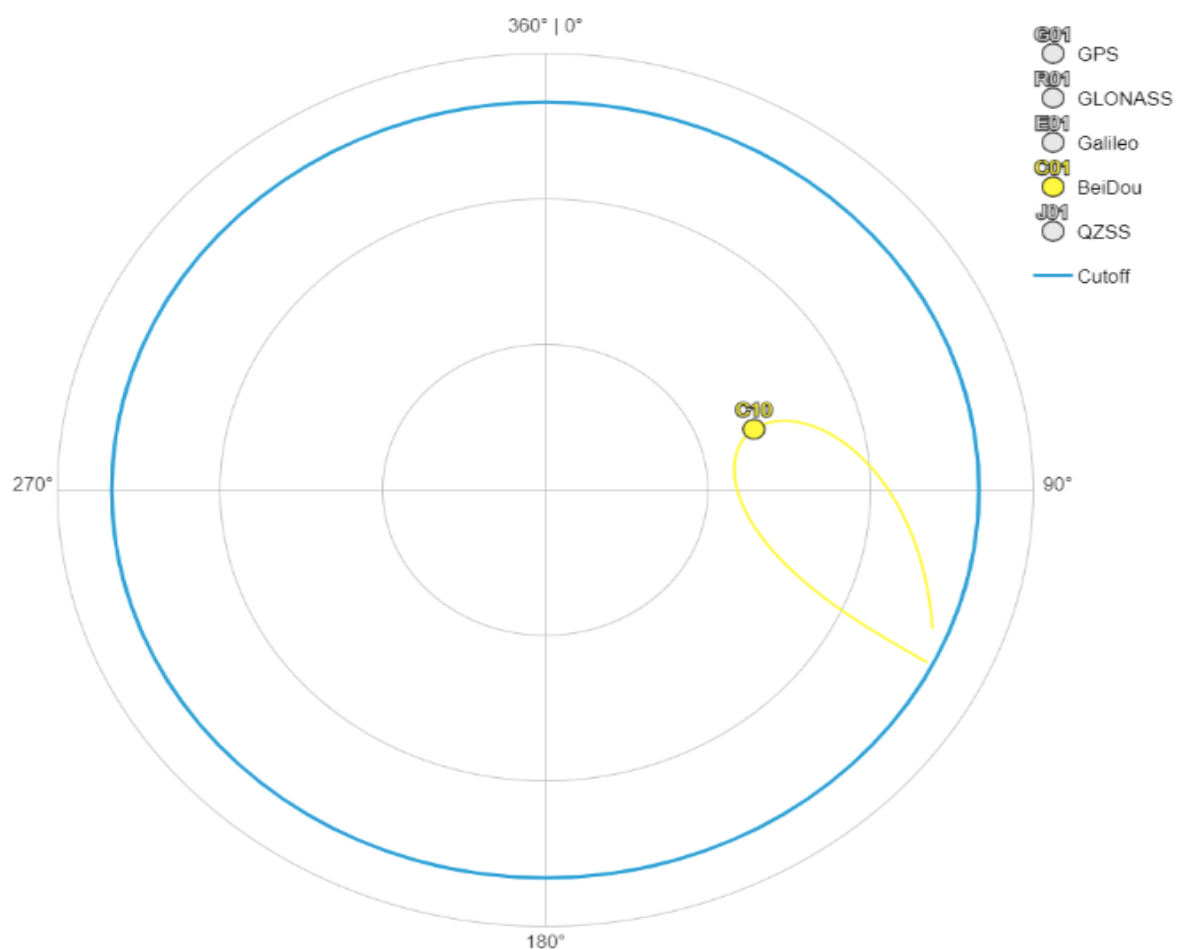


Рисунок 7 – SkyView спутника Beidou C10 в 10:00

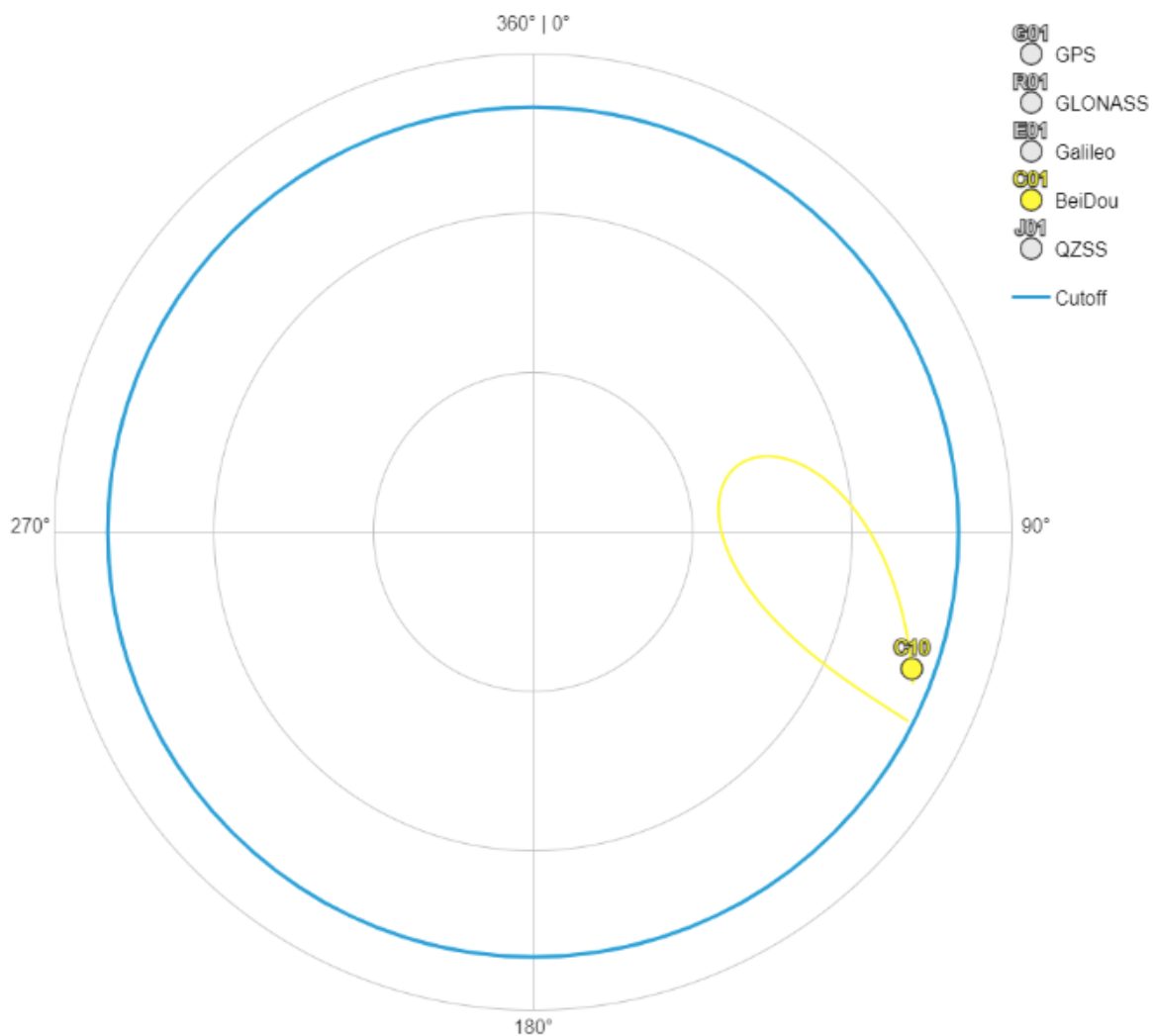


Рисунок 8 – SkyView спутника Beidou C10 в 15:10

По полученным данным подтверждаем, что спутник наблюдается с 3:40 до 15:10 по UTC(+0), в остальное время он находится вне зоны видимости.

1.4 Формирование списка и описание параметров, входящих в состав эфемерид

Таблица 1 – Значения эфемерид спутника С10

Параметр	Обозначение параметра	Значение
Satnum	PRN	10
toe (мс)	t_{oe}	219600000.000
Crs (рад)	-	-1.7112500000000000e+02
Dn (рад/мс)	Δn	2.01794115271825003e-12
M0 (рад)	M_0	-1.24602686448891453e+00
Cuc (рад)	-	-5.77326864004135132e-06
e	e	6.87512021977454424e-03
Cus (рад)	-	-3.77185642719268799e-08
$\sqrt{a} \text{ (m}^{\frac{1}{2}}\text{)}$	\sqrt{A}	6.49323067474365234e+03
Cic (рад)	-	-1.38301402330398560e-07
Omega0 (рад)	Ω_0	2.63308102577652559e+00
Cis (рад)	-	-1.04308128356933594e-07
i0 (рад)	i_0	8.93007494398020185e-01
Crc (рад)	-	2.18765625000000000e+02
Omega (рад)	ω	-2.55042188550297588e+00
OmegaDot (рад/мс)	$\dot{\Omega}$	-2.84118977552985422e-12
iDot (рад/сек)	\dot{i}_{DOT}	-5.40379651838676155e-13
Tgd (мс)	T_{gd}	2.0400000000000000e+05
Toc (мс)	T_{oc}	2.1960000000000000e+08
af2 (мс/мс ²)	-	0.0000000000000000e+00

af1 (mc/mc)	-	6.05826500077455421e-12
af0 (mc)	-	-8.54813233017921448e-02
URA	-	0
IODE	-	257
IODC	-	0
codeL2	-	0
L2P	-	0
WN	-	789

ЭТАП 2. МОДЕЛИРОВАНИЕ

Цель 2-го этапа: требуется реализовать на языке Matlab или Python функцию расчета положения спутника Beidou на заданный момент по шкале времени UTC. В качестве эфемерид использовать данные, полученные на предыдущем этапе.

Построить трехмерные графики множества положений спутника Beidou с системным номером, соответствующим номеру студента по списку. Графики в двух вариантах: в СК ECEF WGS84 и соответствующей ей инерциальной СК. Положения должны соответствовать временному интервалу с 18:00 МСК 16 февраля до 06:00 МСК 17 февраля 2021 года. Допускается использовать одни и те же эфемериды на весь рассматриваемый интервал.

Построить SkyView за указанный временной интервал и сравнить результат с Trimble GNSS Planning Online, полученный на прошлом этапе.

В нашем случае данный этап будем реализовывать с помощью языка Matlab.

1) Построение траектории движения спутника

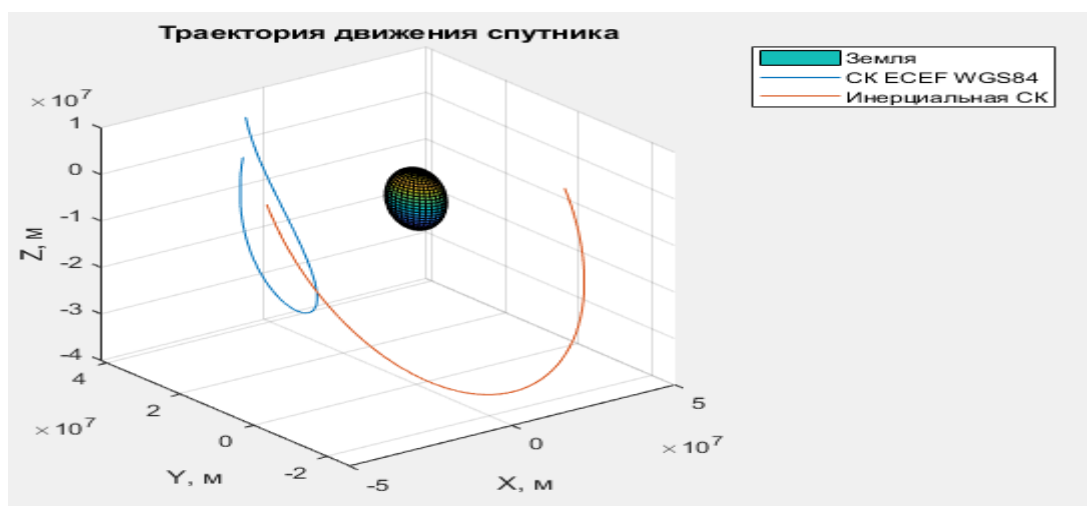
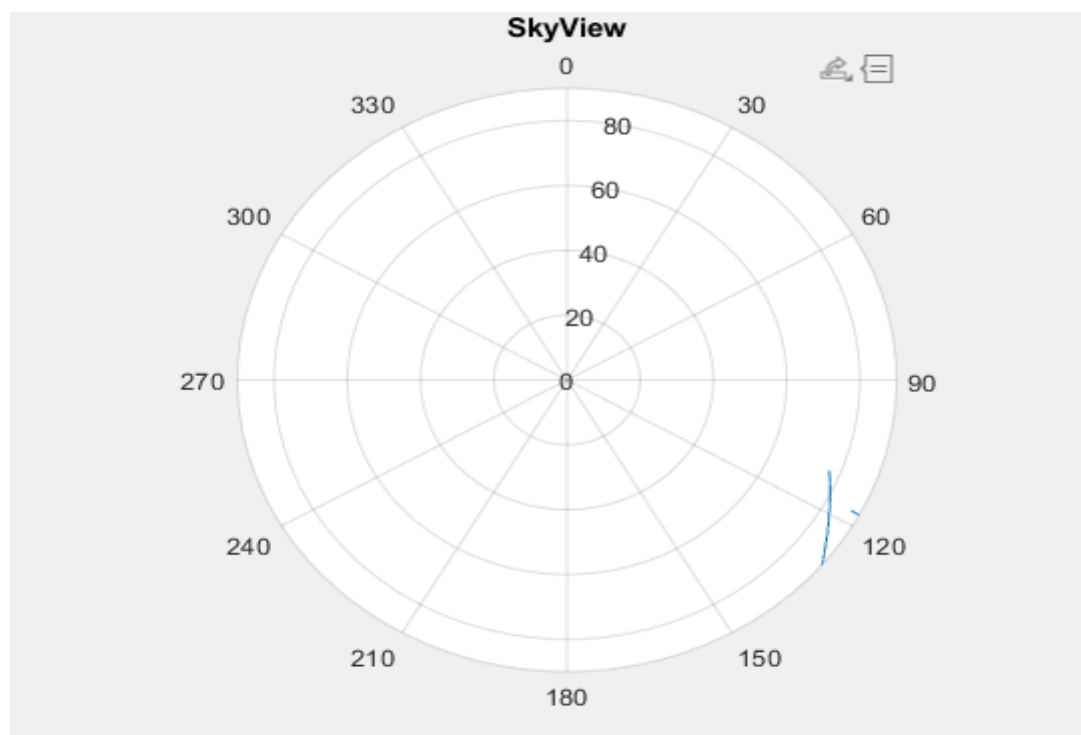


Рисунок 9 – Траектория движения спутника

2) Графики SkyView

а)



б)

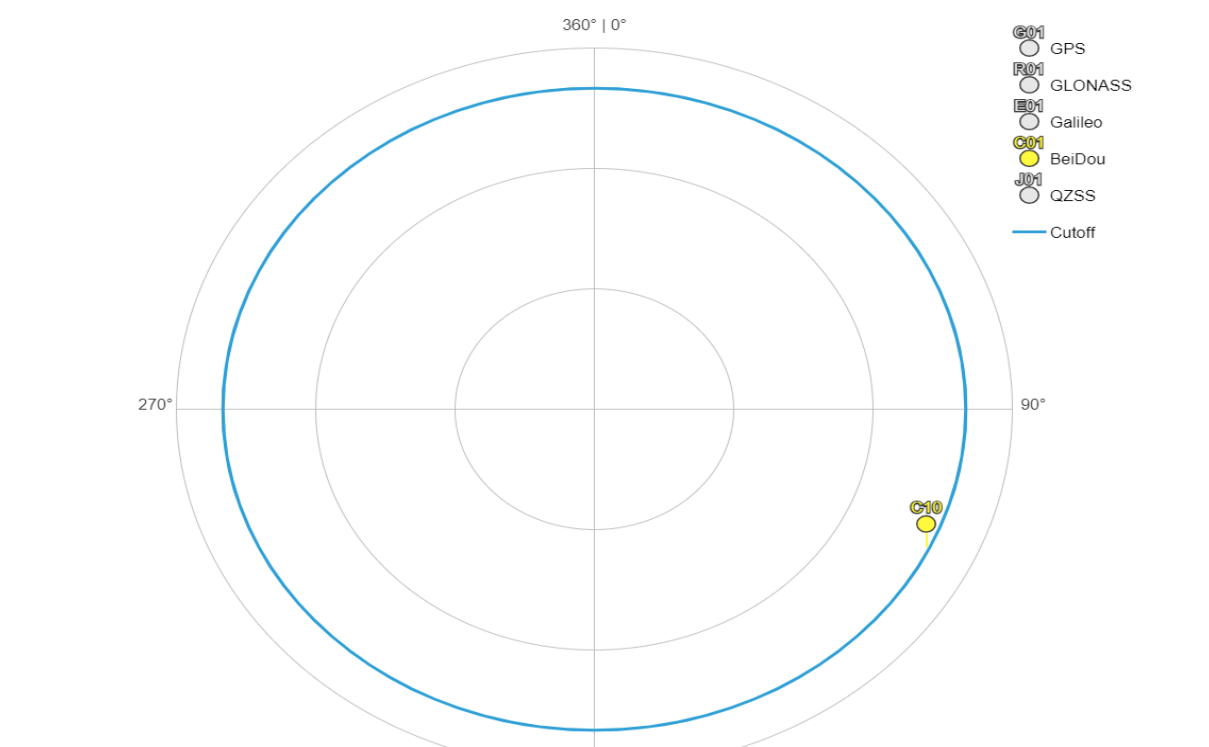


Рисунок 10 – а)SkyView полученный при моделировании
б)SkyView из сервиса Trimble GNSS Planning

3)График угла места

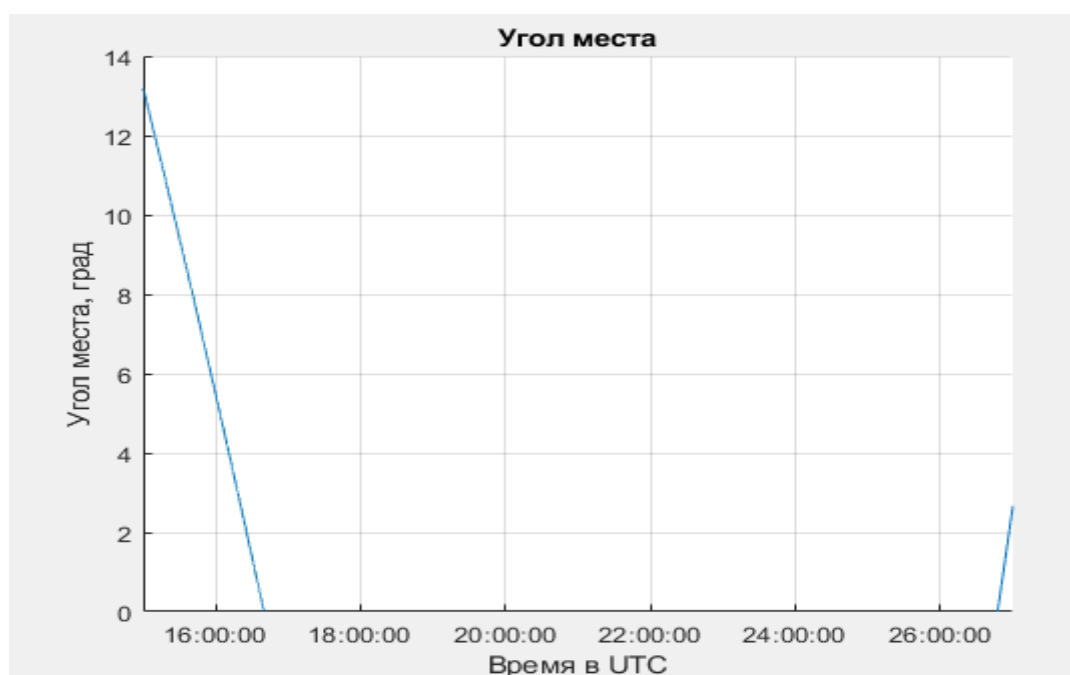
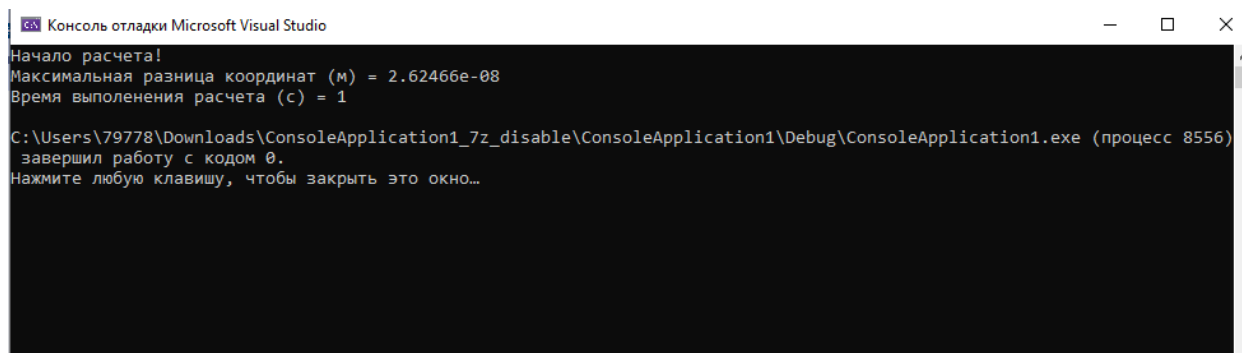


Рисунок 11 – График угла места

Данные полученные с помощью моделирования совпадают с данными полученными с помощью интернет сервиса Trimble GNSS Planning.

ЭТАП 3. РЕАЛИЗАЦИЯ

На этом этапе работы надо разработать на языке C++ функцию расчёта положения спутника Beidou на заданное время по шкале UTC. Написание программы производилось в Visual studio 2019. Алгоритм расчёта местоположения спутника включает в себя уравнение Кеплера. Необходимо протестировать программу.



```
Консоль отладки Microsoft Visual Studio
Начало расчета!
Максимальная разница координат (м) = 2.62466e-08
Время выполнения расчета (с) = 1

C:\Users\79778\Downloads\ConsoleApplication1_7z_disable\ConsoleApplication1\Debug\ConsoleApplication1.exe (процесс 8556)
завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рисунок 12 – Результаты теста программы

Критический путь		
Имя функции	Общее время ЦП [единицы, %]	Собственное время ЦП [единицы, %]
[Внешний код]	92 (100,00%)	19 (20,65%)
mainCRTStartup	73 (79,35%)	0 (0,00%)
__scrt_common_main	73 (79,35%)	0 (0,00%)
__scrt_common_main_seh	73 (79,35%)	0 (0,00%)
invoke_main	73 (79,35%)	0 (0,00%)
main	62 (67,39%)	9 (9,78%)
[Внешний код]	47 (51,09%)	41 (44,57%)

Рисунок 13 – Критический путь

Заключение. При выполнении данного курсового проекта была построена орбита заданного спутника Beidou, получена траектория движения этого спутника на заданный промежуток времени. Так же были изучены все этапы нахождения местоположения спутника и реализованы в Matlab. Последним этапом курсового проекта была разработка функции расчёта местоположения на языке программирования C++ и сравнение результатов с программой из Matlab.

Приложение

```
clear all;
clc;
close all;
%% Эфемериды
SatNum = 10;
toe = 219600;
Crs = -1.7112500000000000e+02;
Dn = 2.01794115271825003e-12;
M0 = -1.24602686448891453;
Cuc = -5.77326864004135132e-06;
e = 6.87512021977454424e-03;
Cus = -3.77185642719268799e-08;
sqrtA = 6.49323067474365234e+03;
Cic = -1.38301402330398560e-07 ;
Omega0 = 2.63308102577652559;
Cis = -1.04308128356933594e-07;
i0 = 8.93007494398020185e-01;
Crc = 2.18765625000000000e+02;
omega = -2.55042188550297588;
OmegaDot = -2.84118977552985422e-12;
iDot = -5.40379651838676155e-13;
Tgd = 2.04000000000000000e+05;
toc = 2.19600000000000000e+08;
af2 = 0 ;
af1 = 6.05826500077455421e-12;
af0 = -8.54813233017921448e-02;
URA = 0;
IODE = 257;
IODC = 0;
codeL2 = 0;
L2P = 0;
WN = 789;
%% Константы
mu = 3.986004418e14; % гравитационная постоянная
Omega_E = 7.2921151467e-5; % скорость вращения
%% Расчет
% Массив времени
```

```

time = 226800:1: 270000;
% Большая полуось
A = sqrt(A^2);
% Среднее движение
n = sqrt(mu/A^3) + Dn;
for k = 1:length(time)
    % Время
    t(k) = time(k) - toe;
    if t(k) > 302400
        t(k) = t(k) - 604800;
    end
    if t(k) < -302400
        t(k) = t(k) + 604800;
    end
    % Средняя аномалия
    M(k) = M0 + n*t(k);
    % Решение уравнения Кеплера
    E(k) = M(k);
    E0(k) = M(k)+1;
    epsilon = 1e-6;
    while abs(E(k) - E0(k)) > epsilon
        E0(k) = E(k);
        E(k) = M(k) + e*sin(E(k));
    end
    % Истинная аномалия
    v(k) = atan2( (sqrt(1 - e^2) * sin(E(k)))/(1 - e*cos(E(k))) , (cos(E(k))) - e)/(1 - e*cos(E(k))) );
    % Коэффициенты коррекции
    Phi(k) = omega + v(k);
    cor_cos(k) = cos(2*Phi(k));
    cor_sin(k) = sin(2*Phi(k));
    % Аргумент широты
    delta_u(k) = Phi(k) + Cuc*cor_cos(k) + Cus*cor_sin(k);
    % Радиус
    delta_r(k) = A * (1 - e * cos(E(k))) + Crc*cor_cos(k) + Crs*cor_sin(k);
    % Наклон
    delta_i(k) = i0 + iDot * t(k) + Cic*cor_cos(k) + Cis*cor_sin(k);
    % Положение на орбите

```

```

x = delta_r(k) * cos(delta_u(k));
y = delta_r(k) * sin(delta_u(k));
% Долгота восходящего угла
Omega(k) = Omega0 + (OmegaDot - Omega_E) * t(k) - Omega_E*toe;
% Координаты
coordx(k) = x * cos(Omega(k)) - y * cos(delta_i(k)) * sin(Omega(k));
coordy(k) = x * sin(Omega(k)) + y * cos(delta_i(k)) * cos(Omega(k));
coordz(k) = y * sin(delta_i(k));
coordx1(k) = coordx(k)*cos(Omega(k)) + coordy(k)*sin(Omega(k));
coordy1(k) = - coordx(k)*sin(Omega(k)) + coordy(k)*cos(Omega(k));
coordz1(k) = coordz(k);
end
%% Пересчет координат центра масс НКА в систему координат WGS-84
ppb = 1e-9;
mas = 1e-3/206264.8; % [рад]
MATRIX_WGS_84 = [-3*ppb -353*mas -4*mas;
353*mas -3*ppb 19*mas;
4*mas -19*mas -3*ppb];
crd_WGS_84 = [coordx; coordy; coordz];
for i = 1:length(crd_WGS_84(1,:))
    crd_WGS_84(:,i) = crd_WGS_84(:,i) + MATRIX_WGS_84 *crd_WGS_84(:,i) +
[0.07; -0; -0.77];
end
crd_WGS_84 = crd_WGS_84.'; % Переход к вектору-строки
%% построение графиков
R_Earth = 6371e3;
[XE,YE,ZE] = sphere(30);
figure
surf(XE*R_Earth,YE*R_Earth,ZE*R_Earth)
hold on
grid on
plot3(crd_WGS_84(:,1), crd_WGS_84(:,2), crd_WGS_84(:,3))
plot3(coordx1, coordy1, coordz1)
title('Траектория движения спутника', 'FontName', 'Arial')
xlabel('X, м', 'FontName', 'Arial')
ylabel('Y, м', 'FontName', 'Arial')
zlabel('Z, м', 'FontName', 'Arial')
hold off

```

```

lgd = legend('Земля','СК ECEF WGS84','Инерциальная СК');
lgd.FontName = 'Arial';
%% Координаты корпуса Е и их перевод в систему WGS-84
Earth_radius = 6378136;
H = 500;% высота [м]
a = Earth_radius;
B = deg2rad(55.45241346);% широта
N = a/sqrt((1-e^2*(sin(B))^2));
L = deg2rad(37.42114473); % долгота
llh = [N E H];
crd_PRM = llh2xyz(llh)';
%% Построение SkyPlot
for i = 1:length(crd_WGS_84(:,1))
    [X(i)                                Y(i)                                Z(i)]
    =ecef2enu(crd_WGS_84(i,1),crd_WGS_84(i,2),crd_WGS_84(i,3),B,L,H,wgs84Ell
    ipsoid,'radians');
    if Z(i) > 0
        r(i) = sqrt(X(i)^2 + Y(i)^2 + Z(i)^2);
        teta(i) = acos(Z(i)/r(i));
        if X(i) > 0
            phi(i) = -atan(Y(i)/X(i))+pi/2;
        elseif (X(i)<0)&&(Y(i)>0)
            phi(i) = -atan(Y(i)/X(i))+3*pi/2;
        elseif (X(i)<0)&&(Y(i)<0)
            phi(i) = -atan(Y(i)/X(i))-pi/2;
        end
    else teta(i) = NaN;
        r(i) = NaN;
        phi(i) = NaN;
    end
end
% Скайплот
figure
ax = polaraxes;
polarplot(ax,phi,teta*180/pi)
ax.ThetaDir = 'clockwise';
ax.ThetaZeroLocation = 'top';
title('SkyView')

```

```

% Угол места
th = hours(time/60/60 - 2*24); % Перевод временной оси в формат hh:mm:ss
figure
grid on
hold on
plot(th,(-teta)*180/pi+90,'DurationTickFormat','hh:mm:ss')
xlim([th(1) th(end)])
title('Угол места', 'FontName', 'Arial')
xlabel('Время в UTC', 'FontName', 'Arial')
ylabel('Угол места, град', 'FontName', 'Arial')

```

ConsoleApplication1.cpp

```

//

#include <iostream>
#include <fstream>

using namespace std;

int main()
{
    setlocale(LC_ALL, "rus");
    std::cout << "Начало расчета!\n";
    time_t t_start, t_stop;
    time(&t_start);

    double SatNum = 10;
    double toe = 219600;
    double Crs = -1.7112500000000000e+02;
    double Dn = 2.01794115271825003e-12;
    double M0 = -1.24602686448891453e+00;
    double Cuc = -5.77326864004135132e-06;
    double e = 6.87512021977454424e-03;
    double Cus = -3.77185642719268799e-08;
    double sqrtA = 6.49323067474365234e+03;
    double Cic = -1.38301402330398560e-07;
    double Omega0 = 2.63308102577652559e+00;
    double Cis = -1.04308128356933594e-07;

```

```

double i0 = 8.93007494398020185e-01;
double Crc = 2.187656250000000000e+02;
double omega = -2.55042188550297588e+00;
double OmegaDot = -2.84118977552985422e-12;
double iDot = -5.40379651838676155e-13;
double Tgd = 2.04000000000000000e+05;
double toc = 2.19600000000000000e+08;
double af2 = 0.00000000000000000e+00;
double af1 = 6.05826500077455421e-12;
double af0 = -8.54813233017921448e-02;
double URA = 0;
double IODE = 257;
double IODC = 0;
double codeL2 = 0;
double L2P = 0;
double WN = 789;

// Значения констант
double mu = 3.986004418e14; // гравитационная постоянная
double omega_e = 7.2921151467e-5; // скорость вращения

// Временной промежуток
int begin_time = (24 * 2 + 18 - 3) * 60 * 60; // время начала 8:00 по МСК
16 февраля
int end_time = (24 * 3 + 6 - 3) * 60 * 60; // время окончания 6:00 по МСК
17 февраля

//Длина временного промежутка
int t_arr = end_time - begin_time;

//Большая полуось
double A = pow(sqrtA,2);

// Среднее движение
double n0 = sqrt(mu /pow(A,3));
double n = n0 + Dn;

// Координаты

```

```

double** coord = new double*[3];
for (int i = 0; i < 3; i++) {
    coord[i] = new double[t_arr];
}

// Координаты из матлаба
double** coordMat = new double*[3];
for (int i = 0; i < 3; i++) {
    coordMat[i] = new double[t_arr];
}

for (int k = 0; k < t_arr; k++) {
    // Время
    double t = begin_time + k - (int)toe;

    if (t > 302400)
        t = t - 604800;
    if (t < -302400)
        t = t + 604800;

    // Средняя аномалия
    double M = M0 + n * t;

    // Решение уравнения Кеплера
    double E = M;
    double E_old = M + 1;
    double epsilon = 1e-6;

    while (abs(E - E_old) > epsilon) {
        E_old = E;
        E = M + e * sin(E);
    }
    // Истинная аномалия
    double nu = atan2(sqrt(1 - e*e) * sin(E), cos(E) - e);

    // Коэффициент коррекции
    double cos_correction = cos(2 * (omega + nu));
    double sin_correction = sin(2 * (omega + nu));

```

```

        // Аргумент широты
        double u = omega + nu + Cuc * cos_correction + Cus *
sin_correction;

        // Радиус
        double r = A * (1 - e * cos(E)) + Crc * cos_correction + Crs *
sin_correction;

        // Наклон
        double i = i0 + iDot * t + Cic * cos_correction + Cis * sin_correction;

        // Долгота восходящего угла
        double lambda = Omega0 + (OmegaDot - omega_e) * t - omega_e *
toe;

        // Положение на орбите
        double x = r * cos(u);
        double y = r * sin(u);

        // Координаты
        double Xk = x * cos(lambda) - y * cos(i) * sin(lambda);
        double Yk = x * sin(lambda) + y * cos(i) * cos(lambda);
        double Zk = y * sin(i);

        coord[0][k] = Xk;
        coord[1][k] = Yk;
        coord[2][k] = Zk;
    }

    // Чтение координат из матлаба
    ifstream file("matlab.txt");

    if (!file.is_open())
        cout << "Файл не может быть открыт!" << endl;
    else {
        for (int k = 0; k < t_arr; k++) {
            file >> coordMat[0][k] >> coordMat[1][k] >> coordMat[2][k];

```



```

    }
    file.close();
}

// Сравнение с матлабом
double max_del = 0;
for (int i = 0; i < 3; i++) {
    for (int k = 0; k < t_arr; k++) {
        if (abs(coord[0][k] - coordMat[0][k]) > max_del) {
            max_del = abs(coord[0][k] - coordMat[0][k]);
        }
    }
}

time(&t_stop);

double del_time = difftime(t_stop, t_start);

cout << "Максимальная разница координат (м) = " << max_del << endl;
cout << "Время выполнения расчета (с) = " << del_time << endl;

delete[] *coord;
delete[] coord;
delete[] *coordMat;
delete[] coordMat;
}

```