

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

Институт: ИРЭ Кафедра: Радиотехнических систем  
Специальность: 11.05.01 Радиоэлектронные системы и  
КОМПЛЕКСЫ

**ОТЧЕТ по курсовой работе**

**СТУДЕНТ**

\_\_\_\_\_/ Мялова К.А.  
(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

Цель работы - добавление в программное обеспечение приемника функциирасчета положения спутника Beidou на заданное время по данным его эфемерид.

Спутник № 12 системы Beidou.

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 юниттестирование в Check.

Конечная цель всего курсового проекта - получить библиотеку функций на «C++», позволяющую рассчитывать положение спутника Beidou по его эфемеридам.

### **Этапы работы:**

На первом этапе подготовим вспомогательные данные для разработки: эфемериды и оценки положения спутника от сторонних сервисов.

На втором этапе требуется реализовать расчет координат для заданного спутника, также получить сравнимые значения и графические отображения траекторий основных характеристик.

На третьем этапе требуется разработать на языке C/C++ функцию расчета положения спутника Beidou на заданное время по шкале UTC, минимизируя время её исполнения и количество затрачиваемой оперативной памяти.

## ЭТАП 1 ИСПОЛЬЗОВАНИЕ СТОРОННИХ СРЕДСТВ

### 1.1 Описание задания

Дан номер спутника BEIDOU, вариант – С14, значения эфемерид для спутников указаны в бинарном и текстовом файлах. Значения получены от антенны Narxон НХ-CSX601А, 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 вечером 16 февраля 2021 года.

С14	38775	МЕО-6	BDS-2	19.09.12	3089	Используется по ЦН
-----	-------	-------	-------	----------	------	--------------------

Рисунок 1 – Состояние 14-го спутника BEIDOU с «Информационно аналитического центра координатно-временного и навигационного обеспечения»

Компас М6	С14	18.09.2012 19:10	CZ-3В/Е	2012-050В	38775	СОО, ~21 500 км	действующий
-----------	-----	------------------	---------	-----------	-------	-----------------	-------------

Рисунок 2 – Состояние 14-го спутника BEIDOU с сайта Википедия

По рисункам 1 и 2 видно номер спутника – 38775, название спутника – «Компас М6».

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

Для выполнения данного пункта нужно перейти на сайт CelesTrak (<https://celestrak.com>), настроить параметры и выбрать нужный спутник, после чего будет определена орбита и его положение.

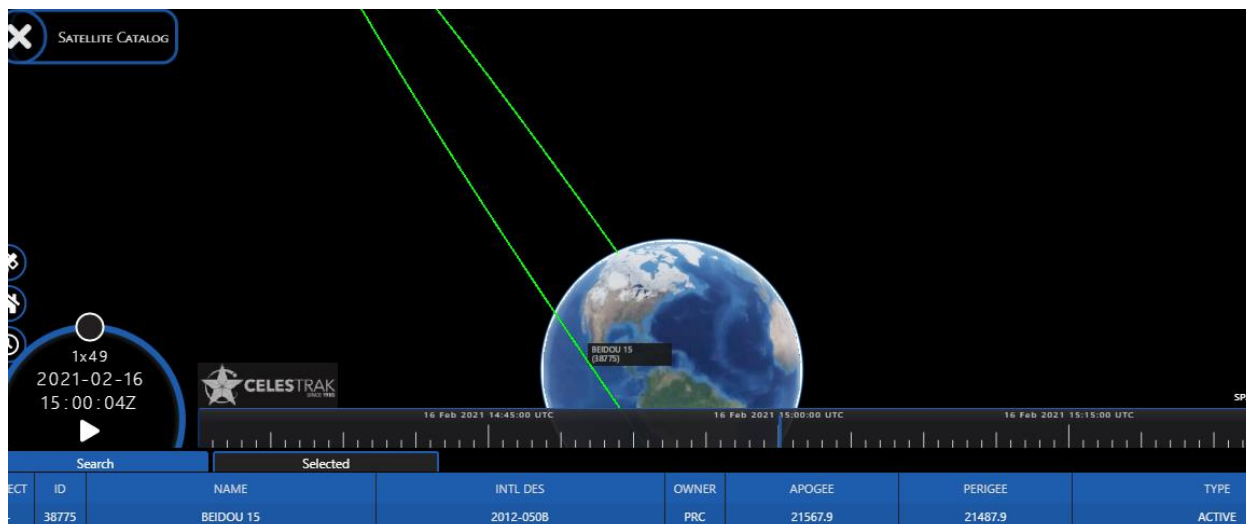


Рисунок 3 – Положение спутника на орбите

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

Введём параметры для моделирования GNSS Planning Online, координаты установим в соответствии с расположением антенны соответственно значению корпуса Е МЭИ, также начальное время будет соответствовать 18:00, временной пояс +3 (UTC +3) на всем этапе моделирования в сервисе GNSS Planning Online.

Система: активная	Спутники	
	Выбранный	Здоровый
GPS	<input checked="" type="checkbox"/>	32
ГЛОНАСС	<input checked="" type="checkbox"/>	23
Галилей	<input checked="" type="checkbox"/>	22
Бейду	<input checked="" type="checkbox"/>	49
QZSS	<input checked="" type="checkbox"/>	4

Мои Настройки	
Время альманаха:	2021-02-16
Часовой пояс:	UTC +03:00
Видимый период:	2021-02-16 18:00 - 2021-02-17 06:00
Широта:	N 55° 45' 23.5491"
Долгота:	E 37° 42' 13.4571"
Высота:	200 м
Отсечка высоты:	10 °

Настройки	
Широта:	N 55° 45' 23.5491"
Долгота:	E 37° 42' 13.4571"
Высота:	200 м
Отсечка высоты:	10 °
День:	16.02.2021
Время начала:	18:00 UTC +03:00
Период [часов]:	12
Часовой пояс:	(UTC+03:00) Москва, Санкт-Петербург

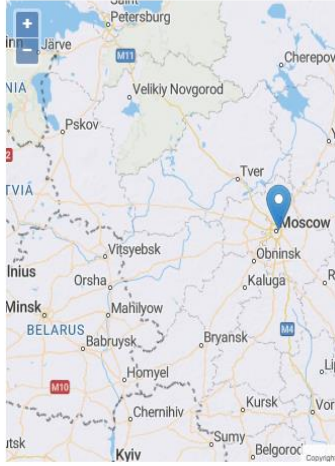


Рисунок 4 – Моделирование с помощью сервиса Trimble GNSS Planning

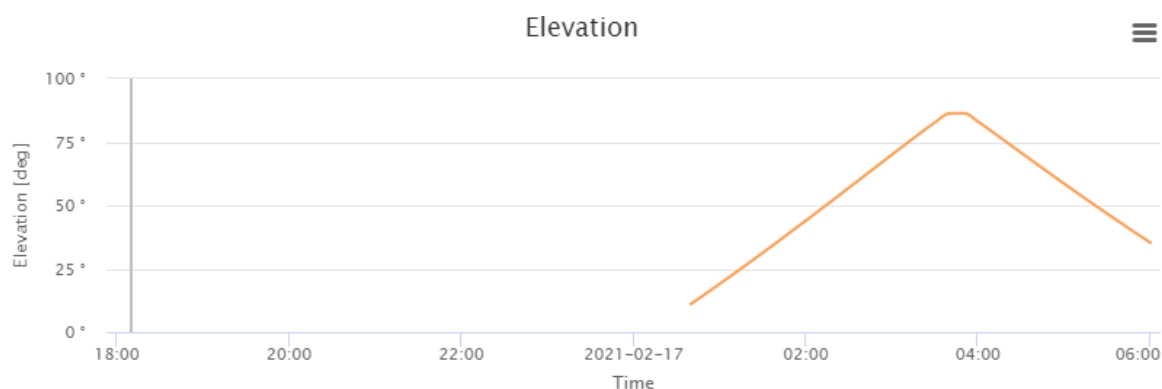


Рисунок 5 – График угла места собственного спутника от времени

Из графика видно, что спутник на указанном временном интервале с 18:00 до 06:00 был в области видимости с 00:40 до 06:00.

## 1.4 Расчет диаграммы угла места и азимута спутника (SkyView, он же SkyPlot) по данным Trimble GNSS Planning Online

Проведем моделирование Sky Plot во временном интервале 18:00-06:00 и зафиксируем положение спутника в критических точках.

2 графика положения спутника:

- 16 февраля 2021 в 00:40:

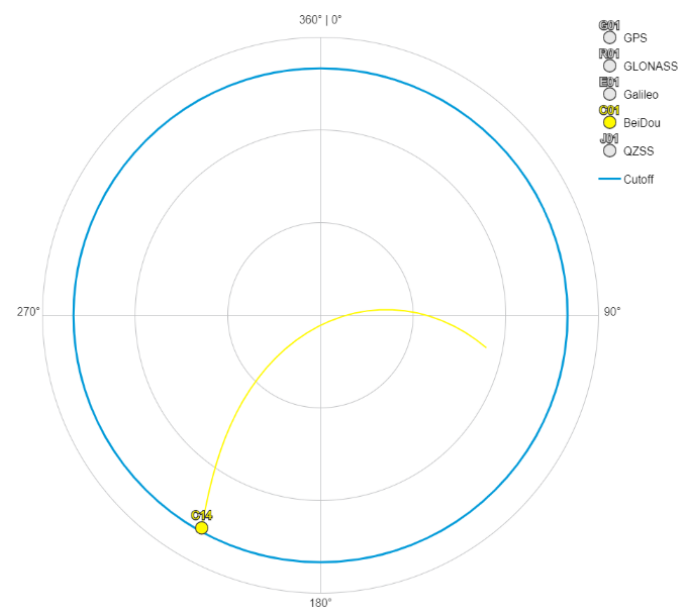


Рисунок 7 – Диаграмма угла азимута спутника

- 17 февраля в 06:00:

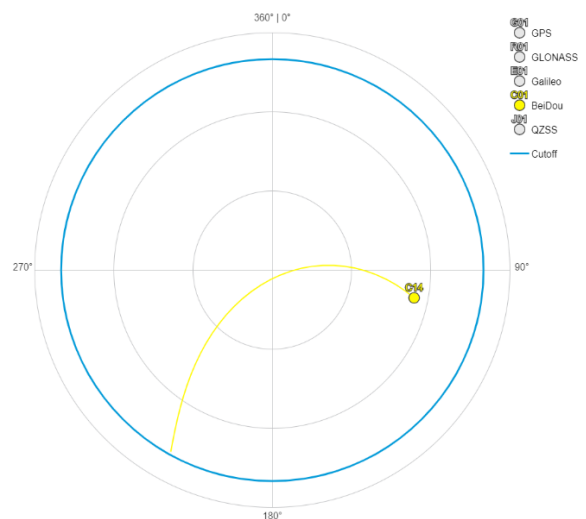


Рисунок 8 – Диаграмма угла азимута спутника

## 1.5 Формирование списка и описание параметров, входящих в состав эфемерид

Таблица 1 – Значения эфемерид спутника С14

Параметр	Обозначение параметра	Значение
SatNum	PRN	14
Toe (мс)	$t_{oe}$	219600000.000
Crs (рад)	-	-7.2312500000000000e+01
Dn (рад/мс)	$\Delta n$	4.05445468865117675e-12
M0 (рад)	$M_0$	2.55684508480358019e+00
Cuc (рад)	-	-3.59397381544113159e-06
e	e	1.28501909784972668e-03
Cus (рад)	-	5.57675957679748535e-06
$\text{sqrt}A \text{ (м}^{\frac{1}{2}}\text{)}$	$\sqrt{A}$	5.28261658287048340e+03
Cic (рад)	-	1.95577740669250488e-08
Omega0 (рад)	$\Omega_0$	-2.81773662124041036e-01
Cis (рад)	-	5.91389834880828857e-08
i0 (рад)	$i_0$	9.62975188353317302e-01
Crc (рад)	-	2.4753125000000000e+02
Omega (рад)	$\omega$	-6.40880762456192743e-01
OmegaDot (рад/мс)	$\dot{\Omega}$	-7.00850621812976967e-12
iDot (рад/сек)	$\dot{i}_{DOT}$	-1.62149611325022453e-13
Tgd (мс)	$T_{gd}$	6.9000000000000000e+04
Toc (мс)	$T_{oc}$	2.1960000000000000e+08
af2 (мс/мс <sup>2</sup> )	-	8.13151611188773069e-22
af1 (мс/мс)	-	8.97282248502051516e-11
af0 (мс)	-	1.49921745061874390e-01
URA	-	0
IODE	-	2570



IODC	-	9
codeL2	-	0
L2P	-	0
WN	-	789

## ЭТАП 2. МОДЕЛИРОВАНИЕ

На предыдущем этапе были получены эфемериды спутника. Эфемериды – параметры некоторой модели движения спутника. В разных ГНСС эти модели разные, а значит отличается и формат эфемерид, и алгоритмы расчета положения спутника.

Одна из самых простых и удобных моделей – в системе GPS. Beidou наследует эту модель.

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 предыдущим этапе.

Построить трехмерные графики множества положений спутника Beidou.

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 предыдущем этапе.

Моделирование проводится в программе Matlab. Код программы приведен в приложении 1.

Построим траектории движения спутника (рисунок 9).

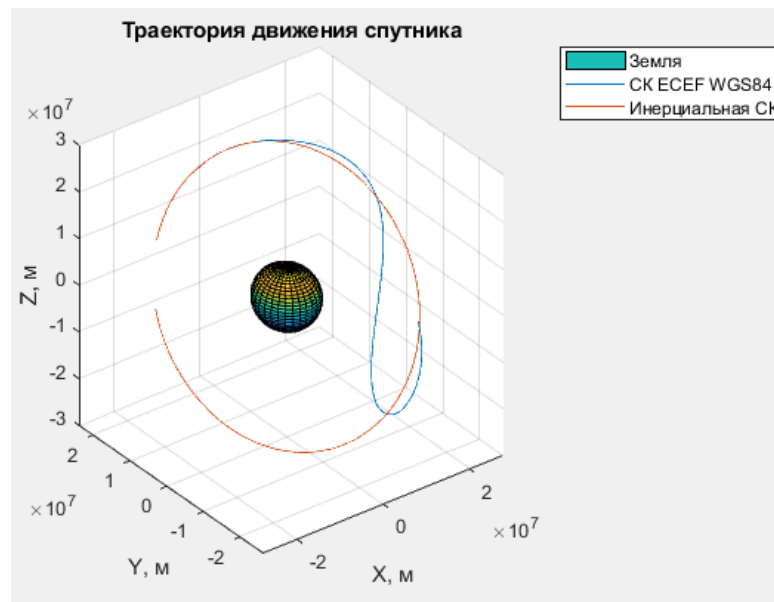


Рисунок 9 – Траектории движения спутника

Расчет графиков SkyView изображен на рисунке 10.

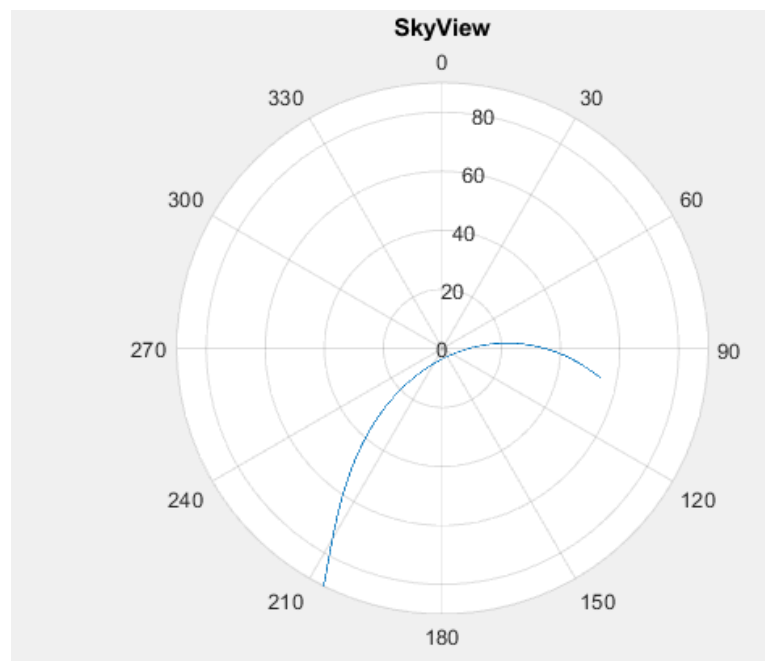


Рисунок 10 – SkyView

Построим график угла места (рисунок 11).

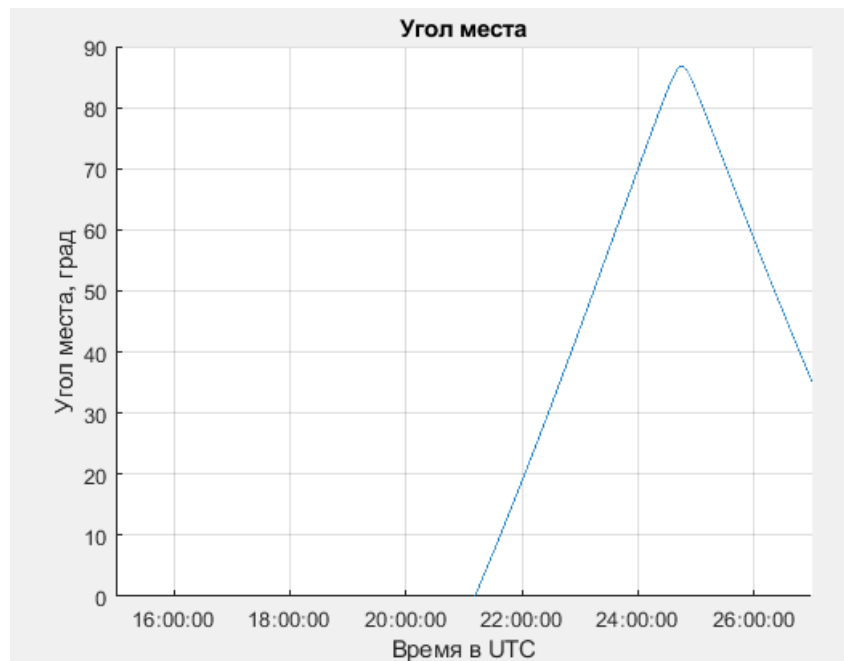


Рисунок 11 – График угла места

Данные результатов моделирования совпадают с данными Trimble GNSS Planning Online с погрешностью из-за того, что использовались одни и те же эфемериды на весь интервал расчета.

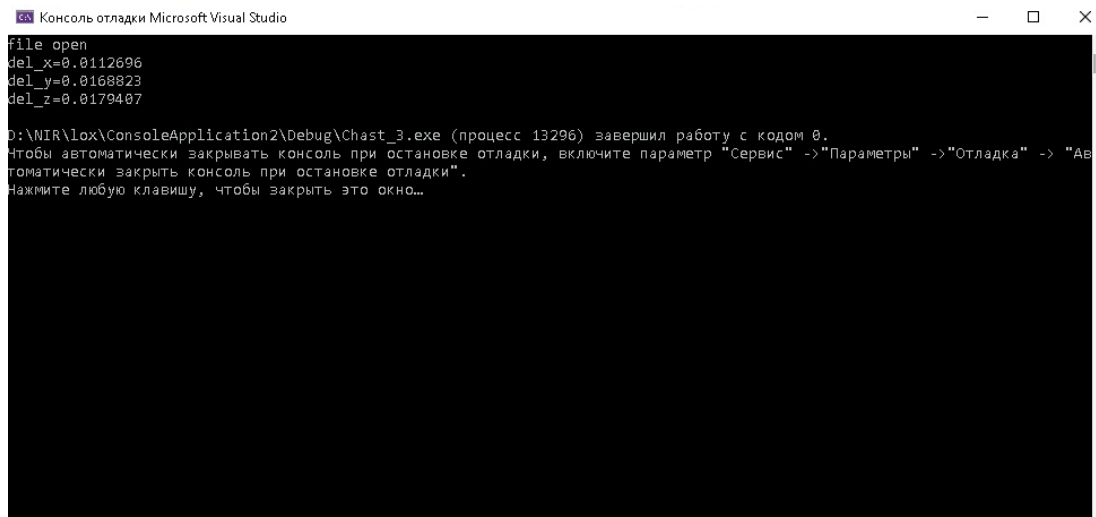
### ЭТАП 3. РЕАЛИЗАЦИЯ

Требуется разработать на языке C/C++ функцию расчета положения спутника Beidou на заданное время по шкале UTC, минимизируя время её исполнения и количество затрачиваемой оперативной памяти. Вызов функции не должен приводить к выбросу исключений или утечкам памяти при любом наборе входных данных.

Функция расчета положения спутника в Matlab/Python относительно проста, т.к. доступны библиотеки линейной алгебры и решения уравнений. Но при разработке встраиваемого ПО приходится сохранять лицензионную частоту, минимизировать вычислительную нагрузку и затраты памяти. Поэтому отобразить модель из Matlab/Python в прошивку приемника дословно, как правило, не получается. В рассматриваемом примере потребуется, как минимум, выполнить свою реализацию решения трансцендентного уравнения.

Код реализации представлен в приложении 2.

Результат запуска программы приведен на рисунке 12:



```
Консоль отладки Microsoft Visual Studio
file open
del_x=0.0112696
del_y=0.0168823
del_z=0.0179407

D:\NIR\lox\ConsoleApplication2\Debug\Chast_3.exe (процесс 13296) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" -> "Параметры" -> "Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рисунок 12 – Результат запуска программы

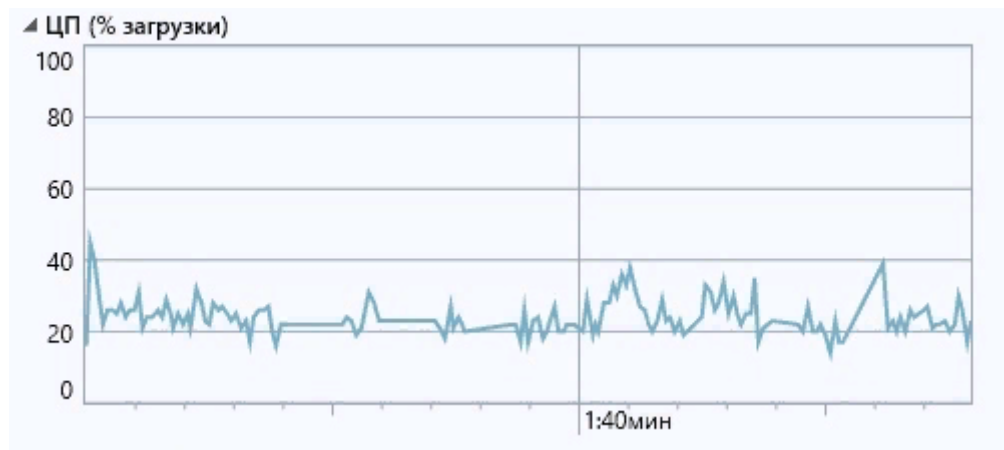


Рисунок 13 – График загрузки ЦП

В итоге был получен график использования памяти при проведении сеанса диагностики. Зависимость представлена на рисунке 14.

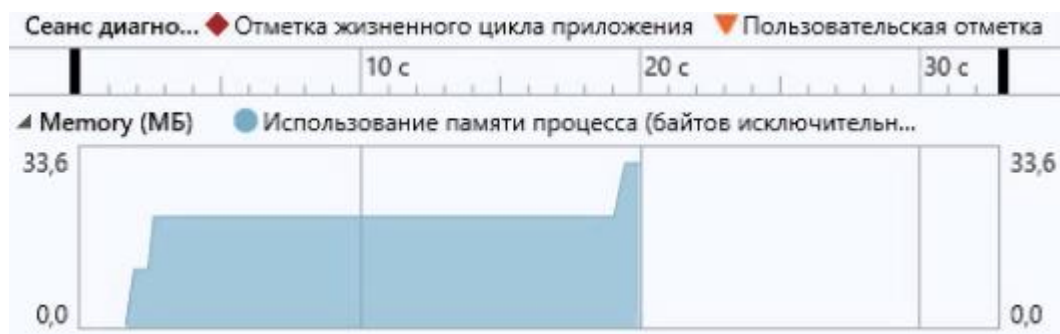


Рисунок 14 – Использование памяти при проведении сеанса диагностики

## **Заключение**

В данной работе был произведен анализ ИКД Beidou, который состоял из трех этапов.

На первом этапе подготовили вспомогательные данные для разработки: эфемериды и оценки положения спутника от сторонних сервисов, с помощью которых была построена орбита заданного спутника Beidou, также получена траектория его движения на заданный промежуток времени.

На втором этапе был реализован расчет координат для заданного спутника, были получены сравнимые значения и графические отображения траекторий основных характеристик.

На третьем, заключительном, этапе была разработка функции расчета местоположения спутника Beidou на языке C++. В ходе выполнения данного этапа было произведено сравнение результатов с программой из Matlab.

## СПИСОК ЛИТЕРАТУРЫ

1. Электронный ресурс: [www.glonass-iac.ru](http://www.glonass-iac.ru) - информационно-аналитический центр координатно - временного и навигационного обеспечения.
2. Сайт: <https://www.celertrak.com> - определение формы орбиты и положения спутника на ней.
3. Интернет-сервис: <https://www.gnssplanningonline.com> – программа, моделирующая положение спутников и их характеристики над заданной точкой земной поверхности.
4. BeiDou Navigation Satellite System. System In Space. Interface Control Document.



## ПРИЛОЖЕНИЕ

### Приложение 1

```
clear all;
clc;
close all;
%% Эфемериды
SatNum = 14;
toe = 219600000.000 * 10^-3;
Crs = -7.2312500000000000e+01;
Dn = 4.05445468865117675e-12;
M0 = 2.55684508480358019e+00;
Cuc = -3.59397381544113159e-06;
e = 1.28501909784972668e-03;
Cus = 5.57675957679748535e-06;
sqrtA = 5.28261658287048340e+03;
Cic = 1.95577740669250488e-08 ;
Omega0 = -2.81773662124041036e-01;
Cis = 5.91389834880828857e-08;
i0 = 9.62975188353317302e-01;
Crc = 2.4753125000000000e+02;
omega = -6.40880762456192743e-01;
OmegaDot = -7.00850621812976967e-12;
iDot = -1.62149611325022453e-13;
Tgd = 6.9000000000000000e+04;
toc = 2.1960000000000000e+08;
af2 = 8.13151611188773069e-22 ;
af1 = 8.97282248502051516e-11;
af0 = 1.49921745061874390e-01;
URA = 0;
IODR = 2570;
```

```

IODC = 9;
codeL2 = 0;
L2P = 0;
WN = 789;

%% Константы
mu = 3.986004418e14; % гравитационная постоянная
Omega_E = 7.2921151467e-5; % скорость вращения

%% Расчет
tstart = (24*2 + 18 - 3)*60*60; % время старта 18:00 МСК 16 февраля
tstop = (24*3 + 6 - 3)*60*60; % время окончания 6:00 МСК 17 февраля
% Массив времени
t_arr = tstart:1:tstop;

% Большая полуось
A = sqrt(A^2);

% Среднее движение
n0 = sqrt(mu/A^3);
n = n0 + Dn;
for k = 1:length(t_arr)
    % Время
    t(k) = t_arr(k) - toe;
    if t(k) > 302400
        t(k) = t(k) - 604800;
    end
    if t(k) < -302400
        t(k) = t(k) + 604800;
    end
end

```

% Средняя аномалия

$$M(k) = M_0 + n \cdot t(k);$$

% Решение уравнения Кеплера

$$E(k) = M(k);$$

$$E\_old(k) = M(k) + 1;$$

$$\epsilon = 1e-6;$$

while abs(E(k) - E\_old(k)) > epsilon

$$E\_old(k) = E(k);$$

$$E(k) = M(k) + e \cdot \sin(E(k));$$

end

% Истинная аномалия

$$\nu(k) = \operatorname{atan2}\left(\frac{\sqrt{1 - e^2} \cdot \sin(E(k))}{1 - e \cdot \cos(E(k))}, \cos(E(k)) - e\right) / (1 - e \cdot \cos(E(k)));$$

% Коэффициенты коррекции

$$\Phi(k) = \omega + \nu(k);$$

$$\operatorname{cor\_cos}(k) = \cos(2 \cdot \Phi(k));$$

$$\operatorname{cor\_sin}(k) = \sin(2 \cdot \Phi(k));$$

% Аргумент широты

$$\delta_u(k) = \Phi(k) + C_{uc} \cdot \operatorname{cor\_cos}(k) + C_{us} \cdot \operatorname{cor\_sin}(k);$$

% Радиус

$$\delta_r(k) = A \cdot (1 - e \cdot \cos(E(k))) + C_{rc} \cdot \operatorname{cor\_cos}(k) + C_{rs} \cdot \operatorname{cor\_sin}(k);$$

% Наклон

$$\delta_i(k) = i_0 + i_{\text{Dot}} \cdot t(k) + C_{ic} \cdot \operatorname{cor\_cos}(k) + C_{is} \cdot \operatorname{cor\_sin}(k);$$

```

% 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,wgs84Elli
psoid,'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(t_arr/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')

```

```
#include <iostream>
#include <fstream>
#include <cmath>
#include "Rash_cepler.h"
#include "Coordinate.h"

using namespace std;

int main()
{
    long double Coordin[3];
    long double e = 1.28501909784972668e-03;
    long double mu = 3.98600442E+14;
    long double Omega_e = 7.2921151467E-5;
    long double toe = 219600000;
    long double A = pow(5.28261658287048340e+03, 2);
    long double Dn = 4.05445468865117675e-12;
    long double n0 = pow(mu / (pow(A, 3)), 0.5);
    long double n = n0 + Dn;
    long double Omega0 = -2.81773662124041036e-01;
    long double omega = -6.40880762456192743e-01;
    long double OmegaDot = -7.00850621812976967e-12;
    long double M0 = 2.55684508480358019;
    long double i0 = 9.62975188353317302e-01;
    long double IDOT = -1.62149611325022453e-13;
    long double Crs = -7.2312500000000000e+01;
    long double Cuc = -3.59397381544113159e-06;
    long double Cus = 5.57675957679748535e-06;
    long double Cic = 1.95577740669250488e-08;
    long double Cis = 5.91389834880828857e-08;
```



```

long double Crc = 2.475312500000000000e+02;
long double t_start = ((24 * 2) + 15) * 3600;
long double t_finish = ((24 * 3) + 3) * 3600;
int N_max = 432000;

long double* x = new long double[N_max];
long double* y = new long double[N_max];
long double* z = new long double[N_max];

int k = 0;
for (long double i = t_start; i < t_finish; i += 0.1)
{
    Coordinate(e, mu, Omega_e, toe,
    A, n0, i0, Omega0, omega, M0,
    Dn, n, OmegaDot, IDOT, Crs, Cuc, Cus,
    Cic, Cis, Crc, i, Coordin);
    x[k] = Coordin[0];
    y[k] = Coordin[1];
    z[k] = Coordin[2];
    k++;
}

long double* x_e2 = new long double[N_max];
long double* y_e2 = new long double[N_max];
long double* z_e2 = new long double[N_max];
int i = 0;

ifstream fin("cord.txt");
if (!fin.is_open())
    cout << "file dont open" << endl;
else

```

```

{
cout << "file open" << endl;
while (fin >> x_e2[i] >> y_e2[i] >> z_e2[i])
{
i++;
}
fin.close();
}

double d_x = 0;
double d_y = 0;
double d_z = 0;
double dxmax = 0;
double dymax = 0;
double dzmax = 0;
long double* xx_e2 = new long double[N_max];
long double* yy_e2 = new long double[N_max];
long double* zz_e2 = new long double[N_max];
for (int i = 0; i < N_max; i++)
{
d_x += (fabs(x[i] - x_e2[i])); //сравнение значений матлаба и си
d_y += (fabs(y[i] - y_e2[i]));
d_z += (fabs(z[i] - z_e2[i]));
xx_e2[i] = d_x;
yy_e2[i] = d_y;
zz_e2[i] = d_z;

}

double del_x = d_x / N_max;
double del_y = d_y / N_max;

```

```

double del_z = d_z / N_max;

/*ofstream ff;
ff.open("Znachdelta.txt");
k = 0;
for (long double i = t_start; i < t_finish; i += 0.1)
{
    ff << "k = " << k << "\t" << "t =" << i << "\t" << xx_e2[k] << "\t" << yy_e2[k] << "\t" <<
zz_e2[k] << endl;
    k++;
}*/
// ff.close();
//del_x = 1.78954e-006; del_y = 1.78954e-006; del_z = 1.78954e-006;

cout << "del_x=" << del_x << endl;
cout << "del_y=" << del_y << endl;
cout << "del_z=" << del_z << endl;
delete[] x;
delete[] y;
delete[] z;
return 0;
}

#include <math.h>
#include "Rash_cepler.h"
#include "Coordinate.h"
#include <iostream>
using namespace std;
void Coordinate(
long double e,
long double mu,

```

```

long double Omega_e,
long double toe,
long double A,
long double n0,
long double i0,
long double Omega0,
long double omega,
long double M0,
long double Dn,
long double n,
long double OmegaDot,
long double IDOT,
long double Crs,
long double Cuc,
long double Cus,
long double Cic,
long double Cis,
long double Crc,
long double t,

long double* Coordinate)
{
long double tk,
Mk,
E_k,
nu_k,
Phi_k,
del_u_k,
del_r_k,
del_i_k,

```

```

u_k,
r_k,
i_k,
x_k,
y_k,
Omega_k,
x_k1,
y_k1,
z_k1;
//long double x_k, y_k, Omega_k, x_k1, y_k1, z_k1;
tk = t - toe;
if (tk > 302400)
{
tk = 604800 - tk;
}
else if (tk < -302400)
{
tk = 604800 + tk;

}
Mk = M0 + n * tk;
E_k = kep(e, Mk);
nu_k = atan2((sqrt(1 - pow(e, 2)) * sin(E_k)) / (1 - e * cos(E_k)), (cos(E_k) -
e) / (1 -
e * cos(E_k)));
Phi_k = nu_k + omega;
del_u_k = Cus * sin(2 * Phi_k) + Cuc * cos(2 * Phi_k);
del_r_k = Crs * sin(2 * Phi_k) + Crc * cos(2 * Phi_k);
del_i_k = Cis * sin(2 * Phi_k) + Cic * cos(2 * Phi_k);
u_k = Phi_k + del_u_k;

```

```

r_k = A * (1 - e * cos(E_k)) + del_r_k;
i_k = i0 + del_i_k + IDOT * tk;
x_k = r_k * cos(u_k);
y_k = r_k * sin(u_k);
Omega_k = Omega0 + (OmegaDot - Omega_e) * tk - Omega_e * toe;
// earth-fixed
x_k1 = x_k * cos(Omega_k) - y_k * cos(i_k) * sin(Omega_k);
y_k1 = x_k * sin(Omega_k) + y_k * cos(i_k) * cos(Omega_k);
z_k1 = y_k * sin(i_k);
//coord[3] = { x_k1 , y_k1, z_k1 };
Coordinate[0] = x_k1;
Coordinate[1] = y_k1;
Coordinate[2] = z_k1;
}
#include <math.h>

#include <iostream>

#include "Rash_cepler.h"

using namespace std;
long double kep(
long double en,
long double Mk)
{
long double Ek = en * sin(0) + Mk;
long double Ekold = 0;
int i = 0;
while (fabs(Ek - Ekold) > 0.000000001)
{

```

```

Ekold = Ek;
Ek = en * sin(Ek) + Mk;
i++;

}

return Ek;
}

#ifdef Rash_cepler_H
#define Rash_cepler_H
long double kep(long double en, long double Mk);
#endif#pragma once
#ifdef Coordinate_H
#define Coordinaate_H
void Coordinate(long double e, long double mu, long double Omega_e, long
double toe,
long double A, long double n0, long double i0, long double Omega0, long
double
omega, long double M0,
long double Dn, long double n, long double OmegaDot, long double IDOT,
long double
Crs, long double Cuc, long double Cus,
long double Cic, long double Cis, long double Crc, long double t, long
double* Coordinate);
#endif#pragma once

```

## ОГЛАВЛЕНИЕ

Постановка задачи.....	2
Этап 1. Использование сторонних средств.....	4
Этап 2. Моделирование.....	10
Этап 3. Реализация.....	14
Заключение.....	15
Список литературы.....	16
Приложение.....	17