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

/ *Казанцев К.О.* /
(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 проекта - добавление в программное обеспечение приемника функции расчета положения спутника Beidou на заданное время по данным его эфемерид.

Исходными данными для этой цели являются: номер спутника Beidou и полученные с помощью навигационных приемников данные эфемерид на вечер 16-го февраля, перечень которых представлен на рисунке 1. Начальной точкой будем считать крышу корпуса «Е» МЭИ. Номер моего спутника 8, значения эфемерид для него представлены в таблице 2.

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 юнит-тестирование в Check.

Этапы курсовой работы отличаются осваиваемыми инструментами.

Этап 1. Использование сторонних средств

1.1. Описание этапа

Конечная цель всего курсового 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)

На крыше корпуса Е МЭИ установлена трехдиапазонная антенна Harxon HX-CSX601A. Она через 50-метровый кабель, сплиттер, bias-tee и усилительподключена 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, доступны в рабочем репозитории (директория logs) в нескольких форматах.

Во-первых, это дамп бинарного потока данных от приемника в формате NVS BINR. Во-вторых, текстовый файл данных пакета 0xF7, полученный из данного дампа с бинарным файлом и протоколом. Он получен подобным printf'ом для каждого спутника с периодом передачи эфемерид, рисунок 1, источник [1].

```
fprintf(fid, "%2u %9.3f %22.17e %22.17e %22.17e %22.17e %22.17e %22.17e %22.17e %22.17e %22.17e %22.17e %22.17e %22.17e
%22.17e %22.17e %22.17e %22.17e %22.17e %22.17e %22.17e %u %u %u %u %u %u\n",
        pack->SatNum,       // 1
        pack->bds[0].toe,   // 2
        pack->bds[0].Crs,   // 3
        pack->bds[0].Dn,    // 4
        pack->bds[0].M0,    // 5
        pack->bds[0].Cuc,   // 6
        pack->bds[0].e,     // 7
        pack->bds[0].Cus,   // 8
        pack->bds[0].sqrtA, // 9
        pack->bds[0].Cic,   // 10
        pack->bds[0].Omega0,// 11
        pack->bds[0].Cis,   // 12
        pack->bds[0].i0,    // 13
        pack->bds[0].Crc,   // 14
        pack->bds[0].omega, // 15
        pack->bds[0].OmegaDot, // 16
        pack->bds[0].iDot,  // 17
        pack->bds[0].Tgd,   // 18
        pack->bds[0].toc,   // 19
        pack->bds[0].af2,   // 20
        pack->bds[0].af1,   // 21
        pack->bds[0].af0,   // 22
        pack->bds[0].URA,   // 23
        pack->bds[0].IODE,  // 24
        pack->bds[0].IODC,  // 25
        pack->bds[0].codeL2,// 26
        pack->bds[0].L2P,   // 27
        pack->bds[0].WN     // 28
        );
```

Рисунок 1 – Очередность параметров эфемерид для каждого спутника

1.2. Определение формы орбиты и положения спутника на ней на начало рассматриваемого интервала времени по данным сервиса CelesTrak

Для начала проверим, действительно ли нумерация спутника совпадает с его PRN. Убедиться в совпадении можно на рисунке 2.

8	Компас IGSO-3	C08	09.04.2011 20:47	CZ-3A	2011-013A	37384	Геосинхронная, накл. 55°; 118° в. д.	действующий

Рисунок 2 – Проверка спутника (ист. 3)

Зайдем на сайт https://celestrak.com и начнем моделирование. Поскольку время у нас московское, в самом сервисе зададим смещенное время от заданного на 3 часа (UTC+3) и выберем интересующий нас спутник (Beidou 8). Результат можно увидеть на рисунке 3.

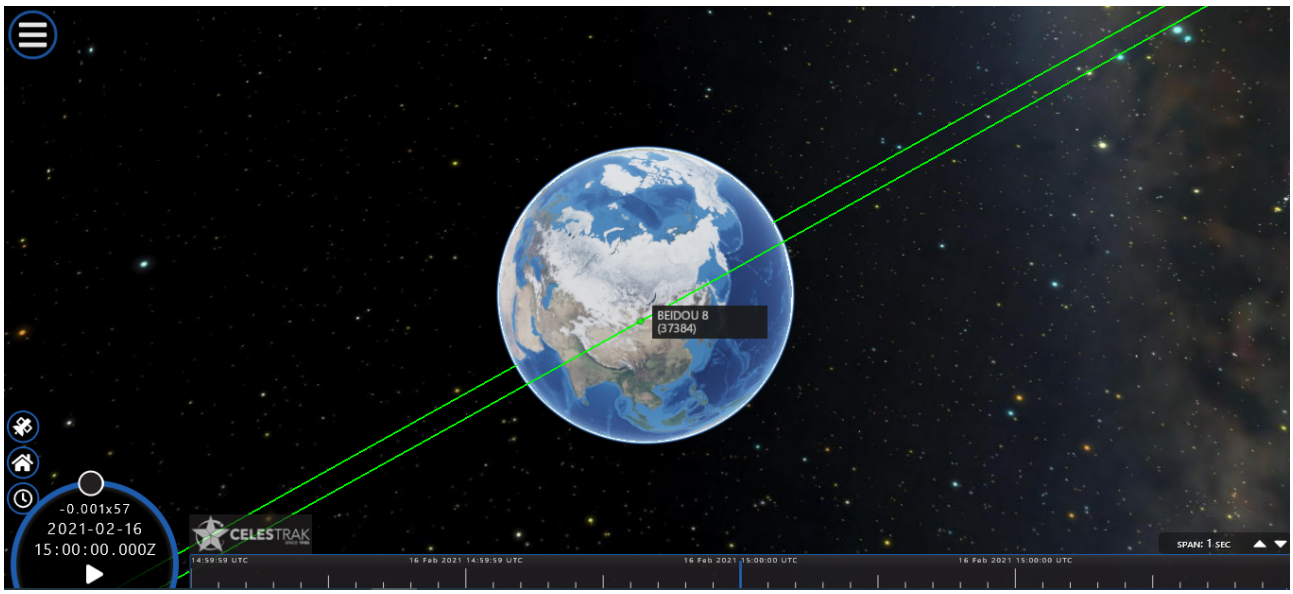

Рисунок 3 – Определение формы орбиты и положения спутника на ней на начало рассматриваемого интервала времени по данным сервиса CelesTrak: общий вид + положение спутника на 18:00 МСК 16 февраля 2021

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

Перейдем на сайт https://www.gnssplanning.com. Зададим параметры, как показано на рисунке 4.

My Settings	
Time of almanac:	2021-02-16
Time zone:	UTC +00:00
Visible period:	
2021-02-16 15:00 - 2021-02-17 03:00	
Latitude:	N 55° 45' 23.5679"
Longitude:	E 37° 42' 12.0145"
Height:	500 m
Elevation cutoff:	10 °

Рисунок 4 – Заданные параметры

Данные координаты соответствуют месту, что отмечено на рисунке 5.

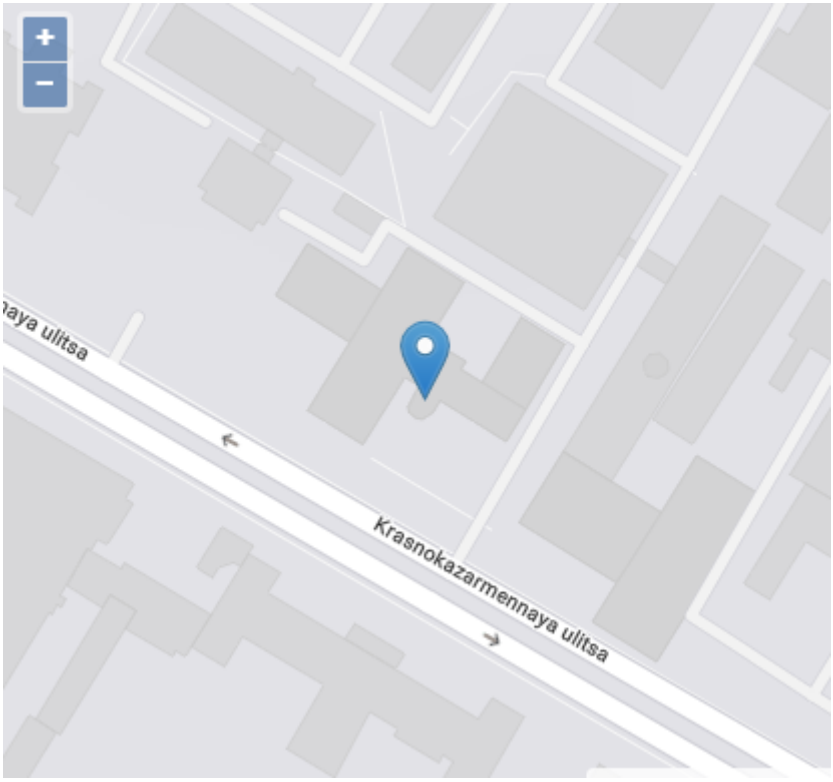

Рисунок 5 – Примерное место размещения антенны

Выберем нужный нам спутник, как на рисунке 6, а остальные — уберем.

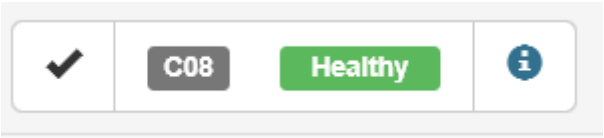

Рисунок 6 – Выбор нужного спутника

В результате получаем график, отображенный на рисунке 7.

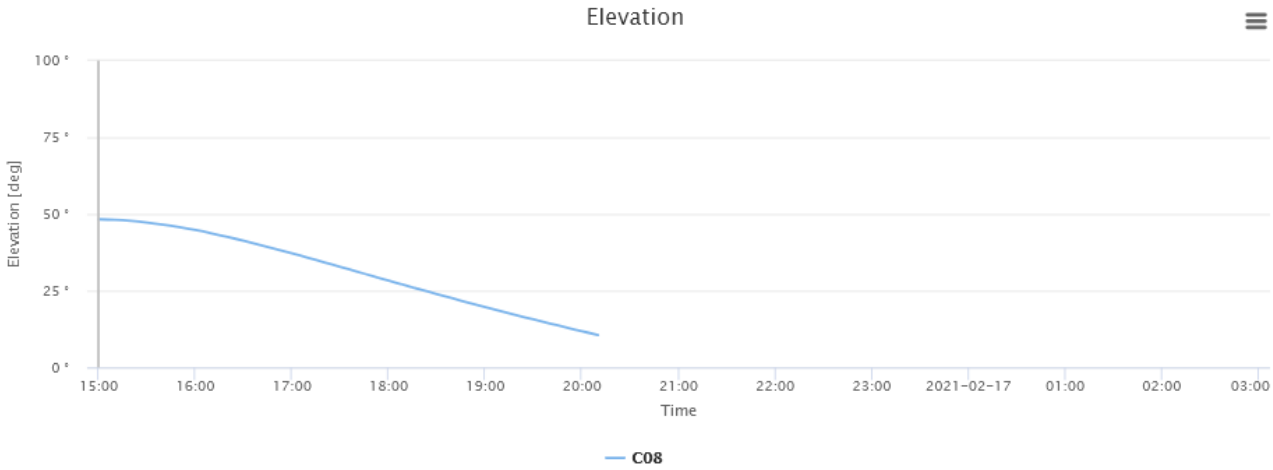

Рисунок 7 – График угла места

По полученной зависимости на рисунке 7 можно узнать, что на момент начала наблюдения (15:00) спутник был виден, а затем, находясь в видимости чуть более 5-ти часов, «скрылся». Данное наблюдение можно будет подтвердить в следующем подпункте.

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

Не меняя заранее установленных настроек, переходим во вкладку SkyPlot. Данный сервис нам показывает изображение, дублированное на рисунке 8.

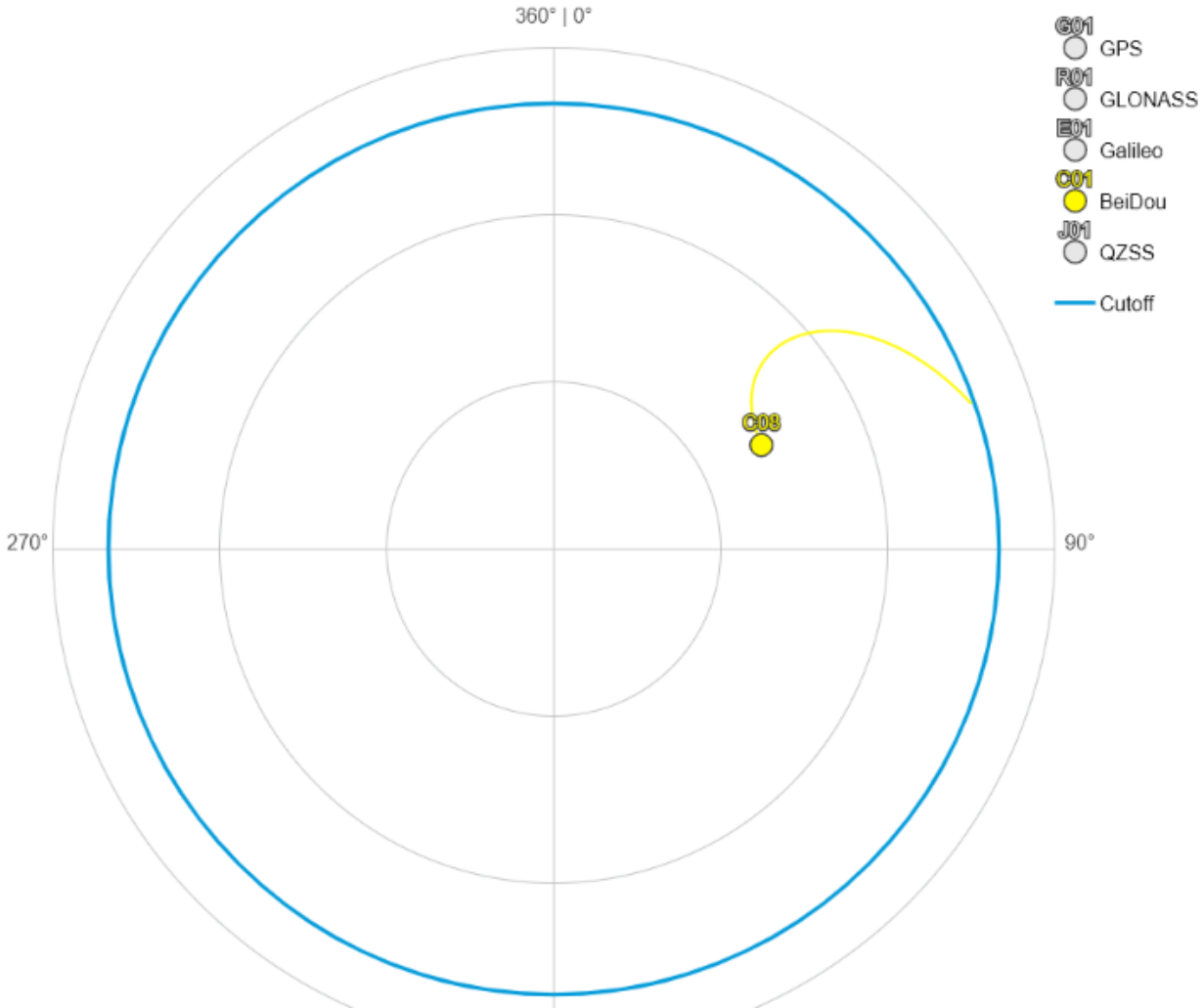

Рисунок 8 – SkyView спутника Beidou C08

Желтая линия на рисунке 8 доказывает, что спутник наблюдался с самого начала. Данная траектория описывает поведение спутника в момент его видимости. По истечении 5-ти часов он стал для нас невидим.

1.5. Формирование списка и описание параметров, входящих в состав эфемерид в сигнале B1I Beidou

Используя [2, стр. 34], составим описание параметров и занесем в таблицу 1.

Таблица 1 – Описание параметров эфемерид системы BEIDOU

Parameter	Definition	Units
t_{oe}	Ephemeris reference time	s
$\sqrt{(A)}$	Square root of semi-major axis	$m^{1/2}$
e	Eccentricity	—
ω	Argument of perigee	π
Δn	Mean motion difference from computed value	π/s
M_0	Mean anomaly at reference time	π
Ω_0	Longitude of ascending node of orbital of plane computed according to reference time	π
Ώ	Rate of right ascension	π/s
i_0	Inclination angle at reference time	π
IDOT	Rate of inclination angle	π/s
C_{uc}	Amplitude of cosine harmonic correction term to the argument of latitude	rad
C_{us}	Amplitude of sine harmonic correction term to the argument of latitude	rad

C_{rc}	Amplitude of cosine harmonic correction term to the orbit radius	m
C_{rs}	Amplitude of sine harmonic correction term to the orbit radius	m
C_{ic}	Amplitude of cosine harmonic correction term to the angle of inclination	rad
C_{is}	Amplitude of sine harmonic correction term to the angle of inclination	rad

1.6. Формирование таблицы эфемерид собственного спутника

Откроем файл с данными. Используя рисунок 1, сформируем таблицу 2. Найдем в документе нужный нам спутник и заполним таблицу.

Таблица 2 – Значения параметров эфемерид спутника C08

Обозначение параметра, размерность	Значение из файла
PRN	8
t_{oe}, мс	237600000
C_{rs}, рад	1.98671875000000000e+02
Δn, рад/мс	8.62178763712945218e-13
M_0, рад	-1.55333764299921384e+00
C_{uc}, рад	6.31529837846755981e-06
e	5.34449948463588953e-03
C_{us}, рад	8.50530341267585754e-06
$\sqrt{(A)}, \sqrt{(м)}$	6.49309631156921387e+03
C_{ic}, рад	-6.75208866596221924e-08
Ω_0, рад	-1.48546375932237629e+00
C_{is}, рад	-1.37835741043090820e-07

i_0, рад	1.03971822793466973e+00
C_{rc}, рад	1.32500000000000000e+01
ω, рад	-2.69237391183751207e+00
Ω, рад	-2.10187326574395889e-12
IDOT, рад/сек	5.20021660989499327e-13
T_{gd}, мс	2.40000000000000000e+05
T_{oc}, мс	2.37600000000000000e+08
af2, мс/мс^2	0.00000000000000000e+00
af1, мс/мс	6.34869934401649516e-12
af0, мс	-9.12912070751190186e-01
URA	0
IODE	257
IODC	0
codeL2	0
L2P	0
WN	789

Этап 2. Моделирование

2.1 Описание этапа

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 данную 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 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 (антенна на крыше корпуса E) и сравнить результат с Trimble GNSS Planning Online, полученный на прошлом этапе.

2.2 Реализация в Matlab

Для расчета были использованы данные из таблицы 2, полученные на прошлом этапе. Начальная точка во времени была определена следующим образом: Так как 16-ое число является началом 2-ой недели, то можем отнять от 16-ти 14 дней. Поскольку время у нас московское, то вычтем 3 часа (UTC+3) и переведём в секунды.

В итоге получаем: $((16-7*2)+18-3)*60^2=226800\,c$

2.3 Результаты моделирования

В данном разделе будут представлены результаты работы программы Matlab, код которой можно будет найти в приложении 1.

2.3.1 Траектории спутника

Алгоритм расчета был взят из [4]. Скорость вращения земли (Ωe) была взята из [5, стр. 2]. Построенные траектории в различных СК можно увидеть на рисунках 9 и 10.

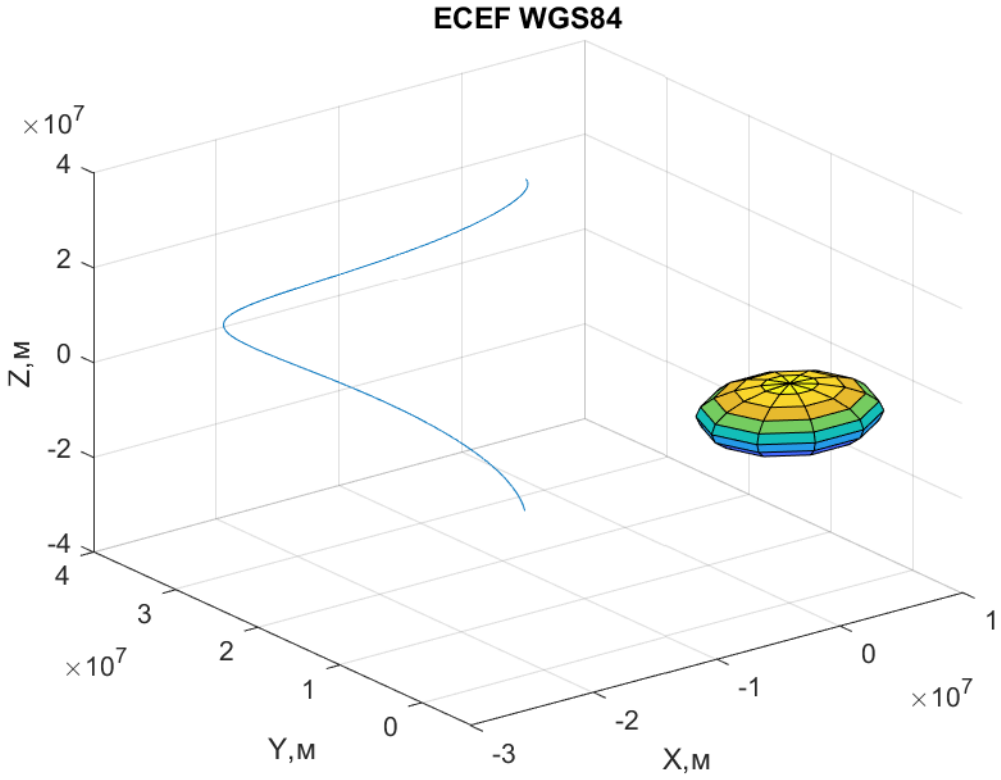

Рисунок 9 – Траектория спутника в СК ECEF WGS84

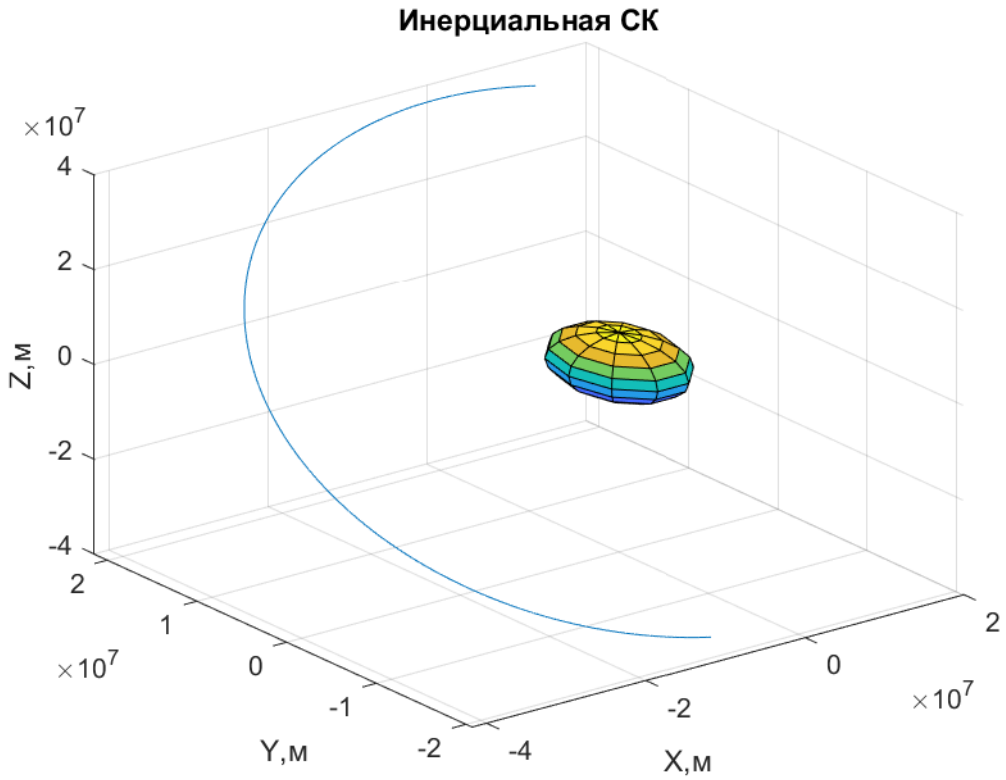

Рисунок 10 – Траектория спутника в инерциальной СК

2.3.2 SkyView

Далее, для сравнения с построенным в 1-ом этапе, был рассчитан SkyView. Полученный результат продемонстрирован на рисунке 11.

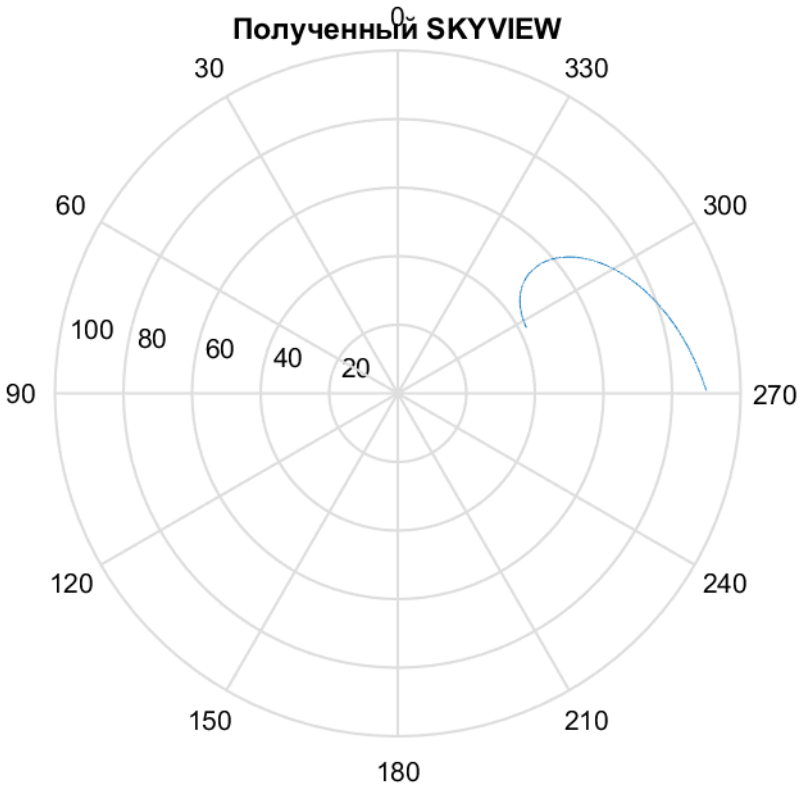

Рисунок 11 – Рассчитанный SkyView

2.4 Выводы

SkyView из этапа 1 и 2 практически совпадают. Использование одних и тех же эфемеридов может приводить к неточностям, для более точного определения следует использовать более «свежие», по мере их получения. Были замечены очевидные различия между СК, но при этом они обе образуют замкнутую траекторию, что говорит о правильности их расчета.

Этап 3. Реализация

3.1 Описание этапа

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.

3.2. Код реализации

Задача была решена с помощью компилятора Visual Studio. Программа представляет собой сборку из трех главных и двух заголовочных файлов. Далее будет приведен каждый файл с их кратким описанием.

main.cpp

```
#include <iostream>
#include <fstream>
#include <string>
#include "Coordinate.h"
#include "Kepler.h"
#include <ctime>
using namespace std;
int main()
```

```
{
    Efemeridi Efemer;
    Efemer.mu = 3.986004418E+14;
    Efemer.we = 7.2921150E-5;
    Efemer.toe = 237600;
    Efemer.A = 4.216029971131373E+07;
    Efemer.e = 5.34449948463588953E-03;
    Efemer.M0 = -1.55333764299921384;
    Efemer.omega = -2.69237391183751207;
    Efemer.i0 = 1.03971822793466973;
    Efemer.omega0 = -1.48546375932237629;
    Efemer.del_n = 8.62178763712945218E-13;
    Efemer.i_dot = 5.20021660989499327E-13;
    Efemer.omega_dot = -2.1018732657439588E-12;
    Efemer.c_uc = 6.31529837846755981E-06;
    Efemer.c_us = 8.50530341267585754E-06;
    Efemer.c_rc = 1.32500000000000000E+01;
    Efemer.c_rs = 1.98671875000000000E+02;
    Efemer.c_ic = -6.75208866596221924E-08;
    Efemer.c_is = -1.37835741043090820E-07;
    time_t start, end;
    double t = 226800;
    double delta_t = 0.1;
    double* coord = new double[3];
    double* coord_matlab = new double[3];
    double max_del = 0;
    ofstream out;
    out.open("vs.txt");
    ifstream in("matlab.txt");
```

```
time(&start);
for (int i = 0; i < (43200 / delta_t); i++)
{
	Ocoord(t, coord, Efemer);
	t += delta_t;
	string coord_str1 = to_string(coord[0]);
	string coord_str2 = to_string(coord[1]);
	string coord_str3 = to_string(coord[2]);
	out << coord_str1 << "  " << coord_str2 << "  " << coord_str3 << endl;
	in >> coord_matlab[0] >> coord_matlab[1] >> coord_matlab[2];
	for (int j = 0; j < 3; j++)
	{
		if (abs(coord[j] - coord_matlab[j]) > max_del)
		{
			max_del = abs(coord[j] - coord_matlab[j]);
		}
	}
}
max_del = max_del * 10000000;
time(&end);
in.close();
delete[] coord;
delete[] coord_matlab;
coord = nullptr;
coord_matlab = nullptr;
double seconds = difftime(end, start);
string seconds1 = to_string(seconds / 43200 / delta_t);
setlocale(LC_ALL, "rus");
cout << "\n\t\tОбщее время расчёта, с: " << seconds << endl;
```

```
	cout << "\n\t\tСреднее время расчёта, с: " << seconds1 << endl;
	string max_del1 = to_string(max_del);
	cout << "\n\t\tМаксимальная разность координат(значение домножено на 10^7): " << max_del1 << endl;
	out.close();
	in.close();
}
```

Это основа проекта. В нем происходит запись координат и сравнение со значениями, полученными в Matlab на этапе 2. Для записи и сравнения значений код матлаб был дополнен:

```
text = fopen('matlab.txt', 'w+');
for i = 1:length(Resfix(:,1))
fprintf(text,'%.7f\t%.6f\t%.6f\n', Resfix(i,1), Resfix(i,2), Resfix(i,3));
end
fclose(text);
```

Так же была введена функция tic-toc, которая нужна для сравнения производительности (времени выполнения) скриптов.

Coordinate.cpp

```
#include "Coordinate.h"
#include <iostream>
#include <math.h>
#include <ostream>
#include "Kepler.h"
using namespace std;
void Ocoord(double t, double* coord, Efemeridi Ef)
{
	double tk = t - Ef.toe;
	double Mk = Ef.M0 + (sqrt(Ef.mu) / pow(sqrt(Ef.A), 3) + Ef.del_n) * tk;
```

```
        double Ek = kepler(Mk, Ef.e);
        double Vk = atan2(sqrt(1 - pow(Ef.e, 2)) * sin(Ek), cos(Ek) - Ef.e);
        double Uk = Ef.omega + Vk + Ef.c_uc * cos(2 * (Ef.omega + Vk)) + Ef.c_us *
sin(2 * (Ef.omega + Vk));
        double rk = Ef.A * (1 - Ef.e * cos(Ek)) + Ef.c_rc * cos(2 * (Ef.omega + Vk)) +
Ef.c_rs * sin(2 * (Ef.omega + Vk));
        double ik = Ef.i0 + Ef.i_dot * tk + Ef.c_ic * cos(2 * (Ef.omega + Vk)) +
Ef.c_is * sin(2 * (Ef.omega + Vk));
        double lambk = Ef.omega0 + (Ef.omega_dot - Ef.we) * tk - Ef.we * Ef.toe;
        coord[0] = (cos(-lambk) * cos(-Uk) - sin(-lambk) * cos(-ik) * sin(-Uk)) * rk;
        coord[1] = (-sin(-lambk) * cos(-Uk) - cos(-lambk) * cos(-ik) * sin(-Uk)) * rk;
        coord[2] = (sin(-ik) * sin(-Uk)) * rk;
}
```

В данном программном коде происходит расчёт координат по эфемеридным данным, находящимся в main.cpp.

Kepler.cpp

```
#include "Kepler.h"
#include <math.h>

double kepler(double Mk, double e) {
        double Ek = Mk;
        double Ek1;
        do {
                Ek1 = Ek;
                Ek = Mk + e * sin(Ek);
        } while (fabs(Ek1 - Ek) > 0.0000001);
        return Ek;
}
```

В данном скрипте реализуется решение уравнения Кеплера. Далее представлены два заголовочных файла.

Kepler.h

```
#ifndef KEPLER_H
#define KEPLER_H
double kepler(double Mk, double e);
#endif /* #ifndef KEPLER_H */#pragma once
```

Coordinate.h

```
#ifndef COORDINATE_H
#define COORDINATE_H
typedef struct {
	double mu;
	double we;
	double toe;
	double A;
	double e;
	double M0;
	double omega;
	double i0;
	double omega0;
	double del_n;
	double i_dot;
	double omega_dot;
	double c_uc;
	double c_us;
	double c_rc;
	double c_rs;
	double c_ic;
```

```
        double c_is;
} Efemeridi;
void Ocoord(double t, double* coord, Efemeridi Ef);
#endif /* #ifndef GPSSVPOS_H */#pragma once
```

3.3. Результаты и сравнение

Выполнение программ без ошибок подтверждает окно консоли на рисунке 12.

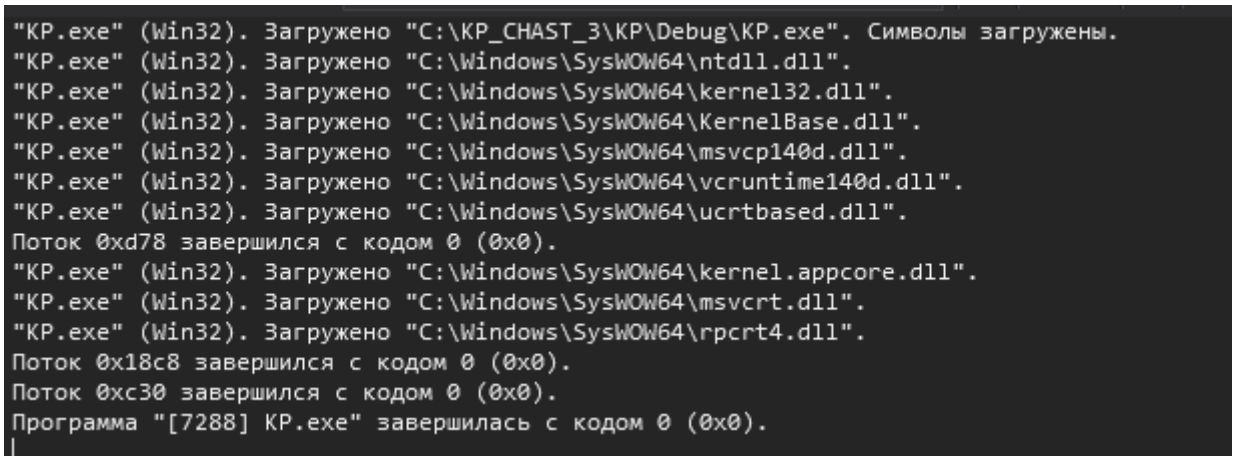

Рисунок 12 – Окно консоли по завершению расчета

На рисунке 13 представлен результат работы программы.

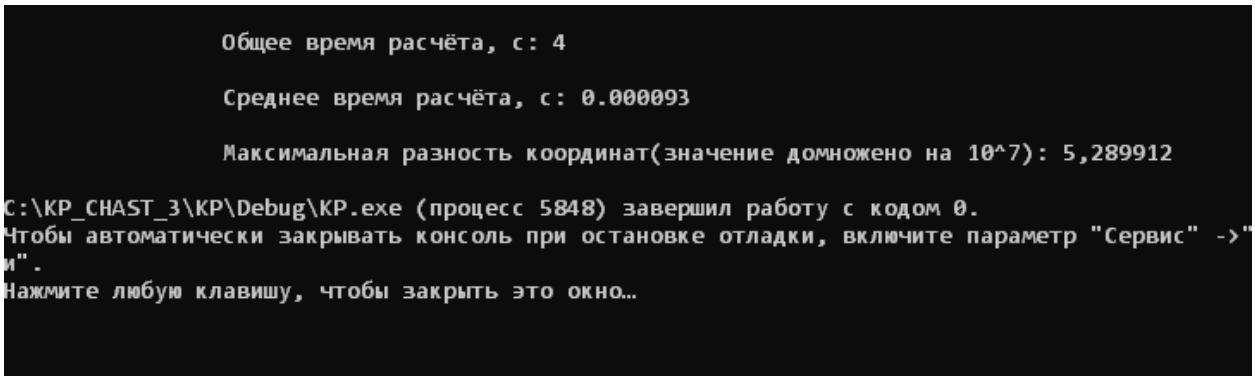

Рисунок 13 – Результат работы программы

Таким образом, максимальная разность координат получилась примерно 529 нм. Общее время расчета колеблется в районе 4-5 секунд, что на 2.5-3 секунды быстрее, чем в Matlab(6.5-7 секунд соответственно). Время одной итерации примерно 93 мкс.

3.4. Анализ работы программы

В Visual Studio есть вкладка отладки. С помощью неё мы и будем проводить анализ программы. Варианты профилирования показаны на рисунке 14.

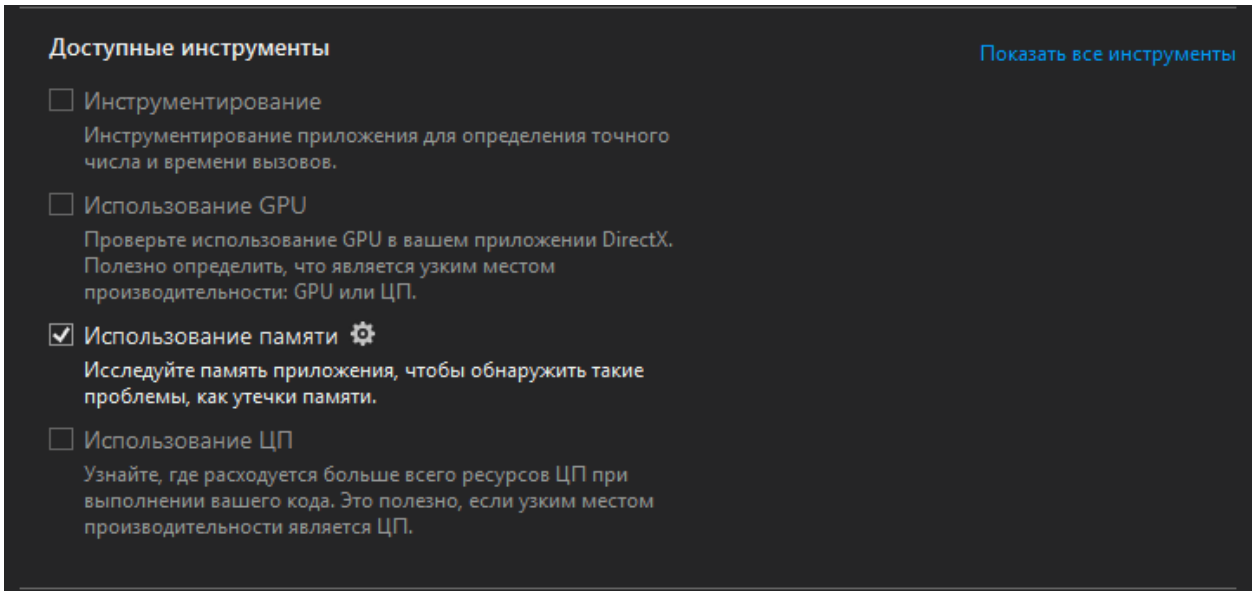

Рисунок 14 – Выбор анализа

Выбрав нужный пункт, можно проанализировать работу программы. Проверим память приложения. На рисунке 15 показано использование памяти процесса по времени. Утечки не были обнаружены.

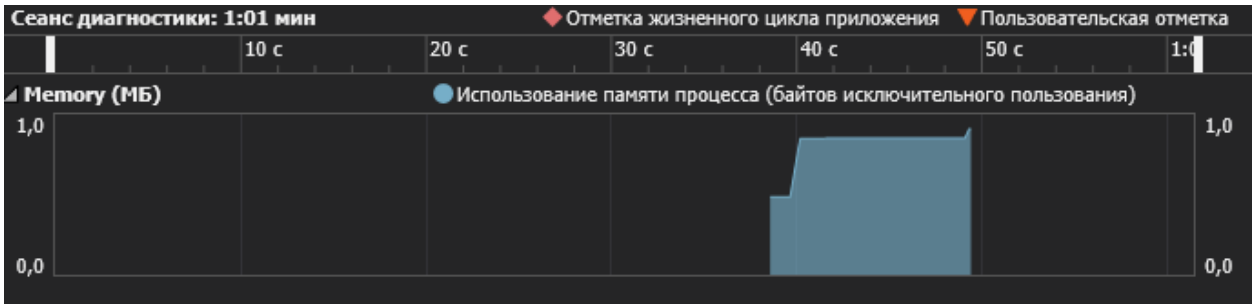

Рисунок 15 – Анализ использования памяти

Далее переходим в использование ЦП и инструментирование для анализа времени выполнения и нагрузки каждого процесса. На рисунке 16 показан процент использования ЦП от времени.

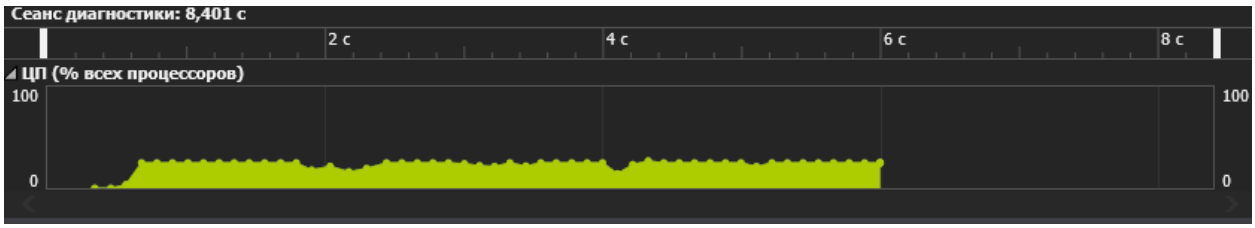

Рисунок 16 – График ЦП

А на рисунке 17 можно увидеть график загрузки ЦП.

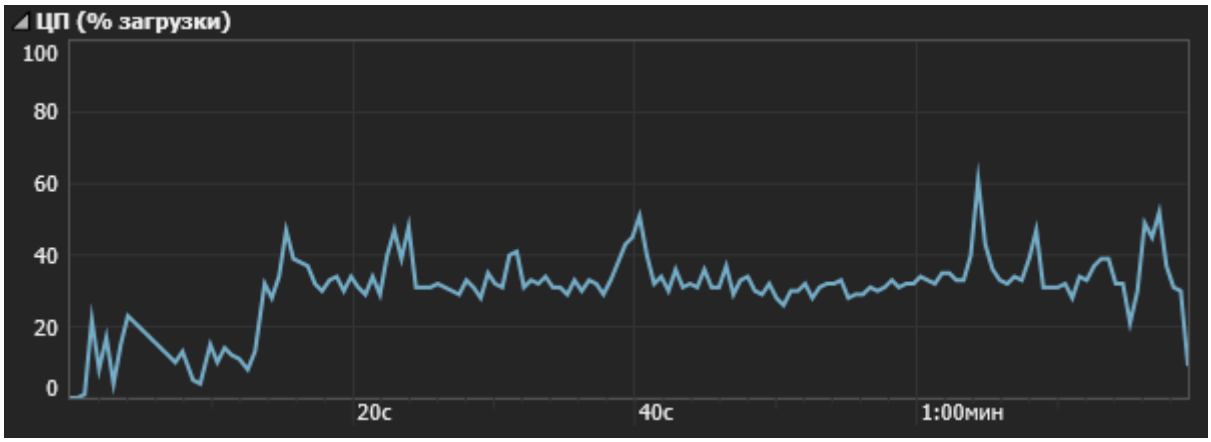

Рисунок 17 – График ЦП

За более подробными данными обратимся к рисунках 18,19 и 20. На них указаны основные «потребляющие» функции.

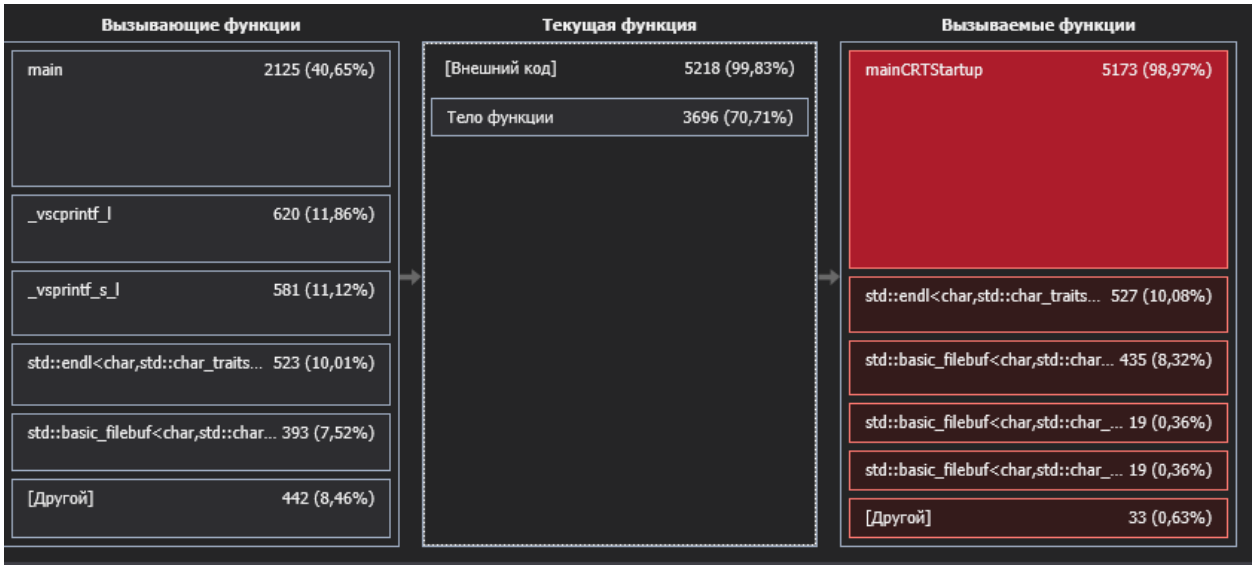

Рисунок 18 – Распределение функций по нагрузке

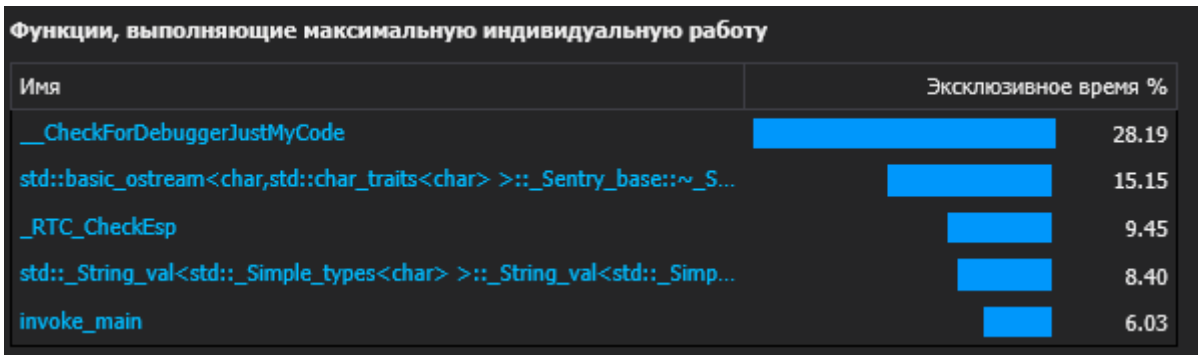

Рисунок 19 – Распределение функций по времени

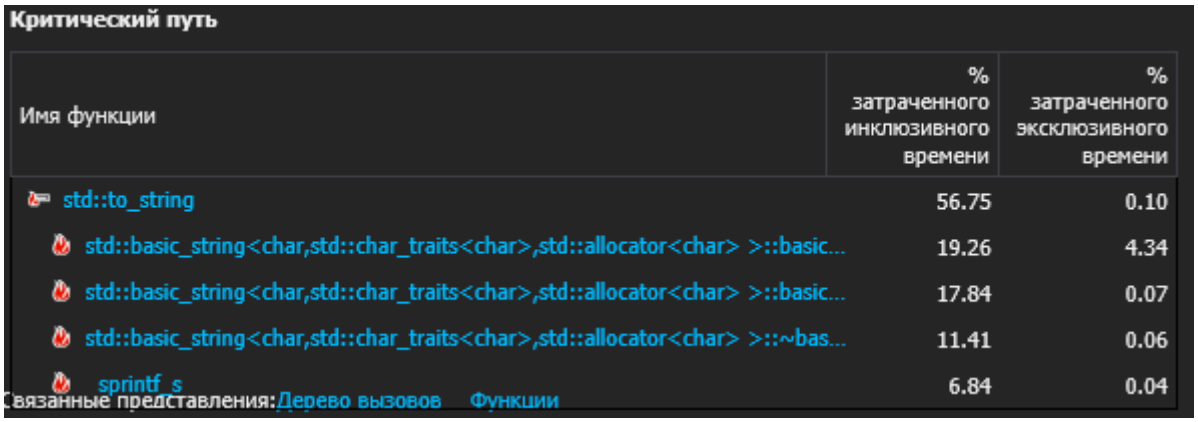

Рисунок 20 – Распределение путей(вызовов) по времени

3.5. Выводы по этапу

Как и ожидалось, расчет координат при использовании разных средств, в нашем случае — Matlab и C++ имеют некоторые отличия, как и во времени выполнения, так и в точности. Впрочем, различия несущественны, что говорит нам о том, что обе реализации были выполнены верно.

4. Заключение

Весь курсовой проект состоял из трех этапов. Начиная с первого, была проведена ознакомительная работа с помощью сторонних средств. Они показывали то, что мы должны будем реализовать в последующих этапах. Так, уже во втором этапе, в среде Matlab был реализован расчет координат для заданного спутника, были получены сравнимые значения и графические отображения траекторий и основных характеристик. Этап три, в свою очередь, носил более сравнительный характер, так, выбрал язык программирования C++ была проанализирована не только сама разработанная программа, но и полученные ранее результаты. Они оказались очевидными и вполне сравнимыми, что и было показано во время последнего этапа. Вся работа включала в себя полный спектр анализа и программирования в разных утилитах. Также данное задание помогло разобраться с ИКД иностранного происхождения, что дало некоторый опыт в работе со спутниками и понимаем их «природы» вцелом.

Список источников

1) https://www.srns.ru/wiki/Аппаратура_потребителей_спутниковых_радионавигационных_систем_(дисциплина)
2) BeiDou Navigation Satellite System Signal In Space, Interface Control Document, Open Service Signal B1I (Version 3.0), China Satellite Navigation Office, February 2019.
3) https://ru.wikipedia.org/wiki/Бэйдоу
4) https://gssc.esa.int/navipedia/index.php/GPS_and_Galileo_Satellite_Coordinates_Computation
5) http://www2.unb.ca/gge/Resources/beidou_icd_english_ver2.0.pdf

Приложение 1

Полный код Matlab для второго этапа

```
clear all; close all;format long;
tic;
Toe = 237600;
Crs = 1.98671875000000000e+02;
mu = 3.986004418e+14;
dn = 8.62178763712945218e-13;
Cuc = 6.31529837846755981e-06;
e = 5.34449948463588953e-03;
Cus = 8.50530341267585754e-06;
A = 6.49309631156921387e+03^2;
Cic = -6.75208866596221924e-08;
Wo = -1.48546375932237629e+00;
Cis = -1.37835741043090820e-07;
Io = 1.03971822793466973e+00;
Crc = 1.32500000000000000e+01;
Mo = -1.55333764299921384e+00;
We = 7.2921150e-5;
W = -2.69237391183751207e+00;
Wdot = -2.10187326574395889e-12;
idot = 5.20021660989499327e-13;
T = 226800;
wur = 55.45235679;
dl = 37.42120145;
H = 200;
no = sqrt(mu/(A^3));
n=no+dn;
for j=1:43201;
```

```
Tk=T-Toe;
M = Mo+n*Tk;
E=0;
Ek=1
 while (abs(Ek - E) > 0.0000001);
    Ek=E;
       E = M + e * sin(E);
end
nu = atan2(sqrt(1-e^2)*sin(E),cos(E)-e);
F1 = nu+W;
du = Cus*sin(2*F1)+Cuc*cos(2*F1);
dr = Crs*sin(2*F1)+Crc*cos(2*F1);
di = Cis*sin(2*F1)+Cic*cos(2*F1);
F2 = F1+du;
r = A*(1-e*cos(E))+dr;
i = Io+di+idot*Tk;
poX = r*cos(F2);
poY = r*sin(F2);
Omega = Wo+(Wdot-We)*(Tk)-We*Toe;
x = poX*cos(Omega)-poY*cos(i)*sin(Omega);
y = poX*sin(Omega)+poY*cos(i)*cos(Omega);
z = poY*sin(i);
Resfix(j,:) = [x y z];
phi = We*Tk;
xc = x*cos(phi)-y*sin(phi);
yc = x*sin(phi)+y*cos(phi);
zc = z;
ResECI(j,:)=[xc yc zc];
[East, North, Up] = ecef2enu(x, y, z, wur, dl,H, wgs84Ellipsoid);
```

```
R = sqrt(East^2 + North^2 + Up^2);
el(j) = rad2deg(-asin(Up/R))+90;
az(j) = atan2(East, North);
T=T+1;
end
text = fopen('matlab.txt', 'w+');
for i = 1:length(Resfix(:,1))
fprintf(text,'%.7f\t%.6f\t%.6f\n', Resfix(i,1), Resfix(i,2), Resfix(i,3));
end
fclose(text);
toc;
[X, Y, Z] = sphere(10);
figure;plot3(Resfix(:,1),Resfix(:,2),Resfix(:,3));
hold on;
surf(X*6.371*10^6, Y*6.371*10^6, Z*6.371*10^6);
grid on;
xlabel('X,м');
ylabel('Y,м');
zlabel('Z,м');
title('ECEF WGS84');
figure; plot3(ResECI(:,1),ResECI(:,2),ResECI(:,3));
hold on;
surf(X*6.371*10^6, Y*6.371*10^6, Z*6.371*10^6);
grid on;
xlabel('X,м');
ylabel('Y,м');
zlabel('Z,м');
title('Инерциальная СК');
s = 1;
```

```
for y = 1:length(el);
if el(y) <= 90;
 Cel(s) = el(y);
 Caz(s) = az(y);
 s = s+1;
end
end
figure;
polar(2*pi-Caz, Cel);
camroll(90);
grid on;
title('Полученный SKYVIEW');
```