**НИУ «МЭИ»**

**Отчет по первой части курсовой работы**

**по дисциплине**

**«Аппаратура потребителей спутниковых радионавигационных систем»**

Выполнил: Кусакин П.А.

Вариант: 31

Москва, 2022

Этап 1. Обработка логов навигационного приемника

Цели:

1. Разработка программы, обрабатывающей входной файл in.txt и выводящей выходной файл out.txt, состоящий из таблицы эфемерид для спутника
2. Сравнить данные, полученные на выходе собственной программы и программы RTKNAVI.

**Пункт 1.**

Код программы написан в редакторе Qt Creator для упрощения работы по созданию Makefile и написанию кода программы. Для удобства части кода были разнесены на несколько файлов. На данном этапе реализован расчет всех целочисленных параметров спутника и с плавающей точкой.

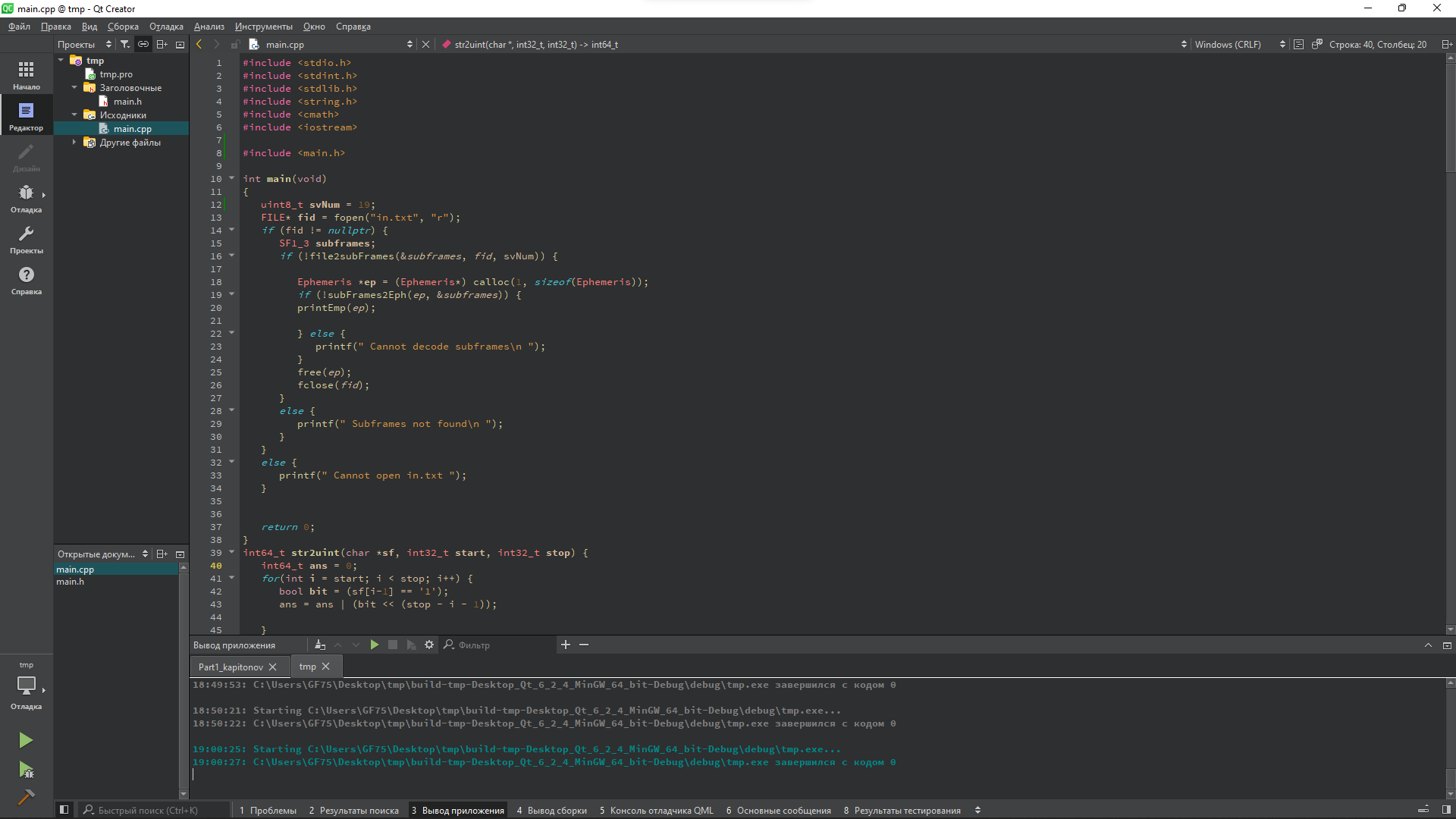


Рисунок 1 – интерфейс программы Qt и часть файла main.cpp

В файле main.cpp находятся вызовы основных функций:

subFramesEph – расчет эфемерид из входного файла in.txt.

file2subFrames – функция парсинга строк входного файла на переменные, необходимые для дальнейшей работы

printEph – вывод полученных данных в выходной файл OUT.txt и открытие этого файла через notepad

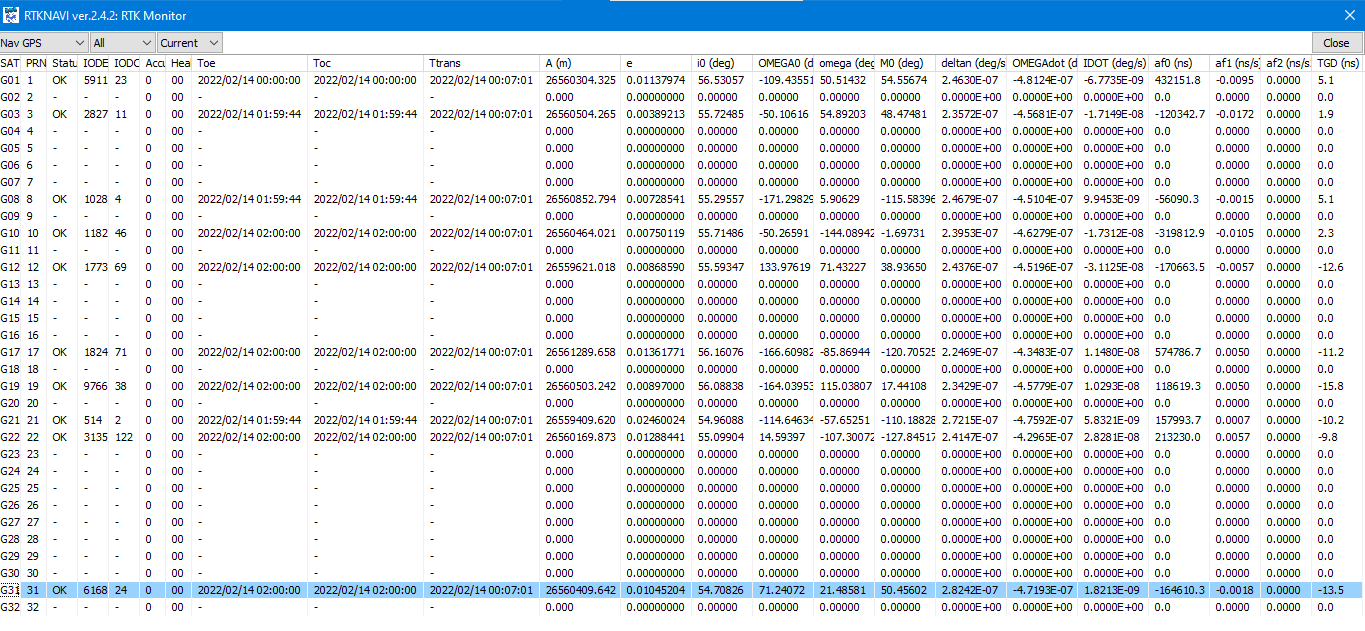


Рисунок 2 –Данные параметры с RTKNAVI

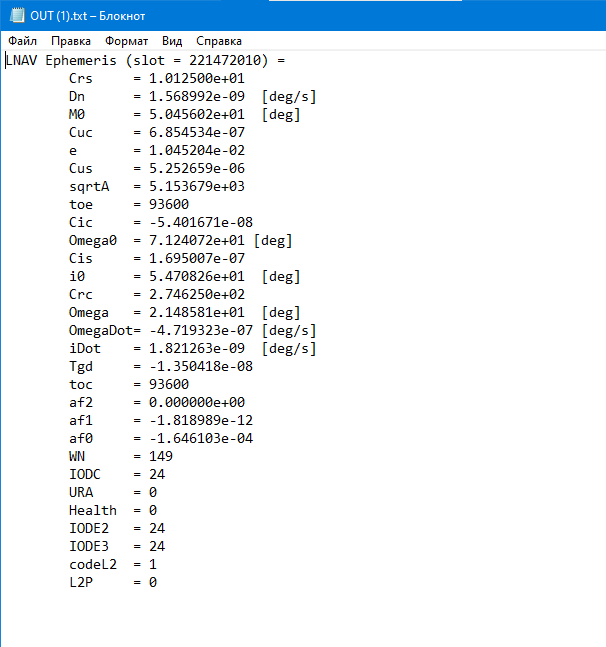


Рисунок 3 –Данные, полученные в ходе компиляции программы

Рассчитанные в коде программы данные и данные из программы RTKNAVI совпадают.

**Приложение**

Листинг программы для пункта 1

MAIN.CPP

#include <stdio.h>

#include <stdint.h>

#include <stdlib.h>

#include <string.h>

#include <cmath>

#include <iostream>

#define **LENGTH** 301

*struct* **Ephemeris** {

double **Crs**;

double **Dn**;

double **M0**;

double **Cuc**;

double **e**;

double **Cus**;

double **sqrtA**;

uint32\_t **toe**;

double **Cic**;

double **Omega0**;

double **Cis**;

double **i0**;

double **Crc**;

double **Omega**;

double **OmegaDot**;

double **iDot**;

double **Tgd**;

uint32\_t **toc**;

double **af2**;

double **af1**;

double **af0**;

uint32\_t **WN**;

uint16\_t **IODC**;

uint8\_t **URA**;

uint8\_t **Health**;

uint16\_t **IODE2**;

uint16\_t **IODE3**;

bool **codeL2**;

bool **L2P**;

uint32\_t **slot**;

};

*struct* **SF1\_3** {

uint32\_t **slot**;

char **sf1**[LENGTH];

char **sf2**[LENGTH];

char **sf3**[LENGTH];

};

void **printEmp**(Ephemeris\* **ep**);

int32\_t **file2subFrames**(SF1\_3\* **sf**, FILE\* **fid**, uint8\_t **svNum**);

int32\_t **subFrames2Eph**(Ephemeris\* **ep**, SF1\_3\* **subframes**);

int64\_t **str2uint**(char \***sf**, int32\_t **start**, int32\_t **stop**);

int64\_t **str2int**(uint64\_t **ans**, int **count**);

int64\_t **str64uint**(char \***sf**, int32\_t **start**, int32\_t **stop**, int32\_t **start2**, int32\_t **stop2**);

#endif *//* *MAIN\_H*

int **main**(void)

{

uint8\_t **svNum** = 31;

FILE\* **fid** = fopen("in.txt", "r");

*if* (fid != *nullptr*) {

SF1\_3 **subframes**;

*if* (!file2subFrames(&*subframes*, *fid*, svNum)) {

Ephemeris \***ep** = (Ephemeris\*) calloc(1, *sizeof*(Ephemeris));

*if* (!subFrames2Eph(*ep*, &*subframes*)) {

printEmp(*ep*);

} *else* {

printf(" Cannot decode subframes\n ");

}

free(*ep*);

fclose(*fid*);

}

*else* {

printf(" Subframes not found\n ");

}

}

*else* {

printf(" Cannot open in.txt ");

}

*return* 0;

}

int64\_t **str2uint**(char \***sf**, int32\_t **start**, int32\_t **stop**) {

int64\_t **ans** = 0;

*for*(int **i** = start; i < stop; i++) {

bool **bit** = (sf[i-1] == '1');

ans = ans | (bit << (stop - i - 1));

}

*return* ans;

}

int64\_t **str2int**(uint64\_t **ans**, int **count**) {

int64\_t **Ians** = 0;

*switch* (count){

*case* 8:

*if* (bool((1<<7) & ans)){

ans |= 0xFFFFFFFFFFFFFF00;

Ians = ~(ans - 1);

*return* -Ians;}

*break*;

*case* 14:

*if* (bool((1<<13) & ans)) {

ans |= 0xFFFFFFFFFFFFC000;

Ians = ~(ans - 1);

*return* -Ians;}

*break*;

*case* 16:

*if* (bool((1 << 15) & ans)) {

ans |= 0xFFFFFFFFFFFF0000;

Ians = ~(ans - 1);

*return* -Ians;}

*break*;

*case* 22:

*if* (bool((1 << 21) & ans)) {

ans |= 0xFFFFFFFFFFC00000;

Ians = ~(ans - 1);

*return* -Ians;}

*break*;

*case* 24:

*if* (bool((1 << 23) & ans)) {

ans |= 0xFFFFFFFFFF000000;

Ians = ~(ans - 1);

*return* -Ians;}

*break*;

*case* 32:

*if* (bool((1 << 31) & ans)) {

ans |= 0xFFFFFFFF00000000;

Ians = ~(ans - 1);

*return* -Ians;}

*break*;

}

*return* ans;

}

int64\_t **str64uint**(char \***sf**, int32\_t **start**, int32\_t **stop**, int32\_t **start2**, int32\_t **stop2**){

uint32\_t **ans** = 0;

*for*(int **i** = start; i < stop; i++) {

ans = (ans | ((sf[i-1] == '1')? 1 : 0)) << 1;

}

*for*(int **i** = start2; i < stop2-1; i++) {

ans = ans | ((sf[i-1] == '1')? 1 : 0);

*if* (i < stop2-1) {

ans = ans << 1;

}

}

*return* ans;

}

int32\_t **subFrames2Eph**(Ephemeris\* **ep**, SF1\_3\* **subframes**) {

ep->slot = subframes->slot;

ep->Crs = str2int(str2uint(*subframes*->*sf2*,69,69+16),16)\*pow(2, -5);

ep->Dn = str2int(str2uint(*subframes*->*sf2*, 91, 91+16), 16)\*pow(2, -43);

ep->M0 = str2int(str64uint(*subframes*->*sf2*,107, 107+8, 121, 121+24),32)\*pow(2, -31)\*180;

ep->Cuc = str2int(str2uint(*subframes*->*sf2*,151,151+16),16)\*pow(2, -29);

ep->e = str64uint(*subframes*->*sf2*,167, 167+8, 181, 181+24) \* pow(2, -33);

ep->Cus = str2int(str2uint(*subframes*->*sf2*,211,211+16),16)\*pow(2, -29);

ep->sqrtA = str64uint(*subframes*->*sf2*,227, 227+8, 241, 241+24) \* pow(2, -19);

ep->toe = str2uint(*subframes*->*sf2*, 271, 287)\*pow(2,4);

ep->Cic = str2int(str2uint(*subframes*->*sf3*,61,61+16),16)\*pow(2, -29);

ep->Omega0 = str2int(str64uint(*subframes*->*sf3*,77, 77+8, 91, 91+24),32)\*pow(2, -31)\*180;

ep->Cis = str2int(str2uint(*subframes*->*sf3*,121,121+16),16)\*pow(2, -29);

ep->i0 = str2int(str64uint(*subframes*->*sf3*,137, 137+8, 151, 151+24),32)\*pow(2, -31)\*180;

ep->Crc = str2int(str2uint(*subframes*->*sf3*,181,181+16),16)\*pow(2, -5);

ep->Omega = str2int(str64uint(*subframes*->*sf3*,197, 197+8, 211, 211+24),32)\*pow(2, -31)\*180;

ep->OmegaDot = str2int(str2uint(*subframes*->*sf3*,241,241+24),24)\*pow(2, -43)\*180;

ep->iDot = str2int(str2uint(*subframes*->*sf3*,279,279+14),14)\*pow(2, -43)\*180;

ep->Tgd = str2int(str2uint(*subframes*->*sf1*,197,197+8),8)\*pow(2, -31);

ep->toc = str2int(str2uint(*subframes*->*sf1*,219,219+16),16)\*pow(2, 4);

ep->af2 = str2int(str2uint(*subframes*->*sf1*,241,241+8),8)\*pow(2, -55);

ep->af1 = str2int(str2uint(*subframes*->*sf1*,249,249+16),16)\*pow(2, -43);

ep->af0 = str2int(str2uint(*subframes*->*sf1*,271,271+22),22)\*pow(2, -31);

ep->WN = str2uint(*subframes*->*sf1*, 61, 71);

ep->IODC = str64uint(*subframes*->*sf1*,83, 83+2, 211, 211+8);

ep->URA = str2uint(*subframes*->*sf1*, 73, 77);

ep->Health = str2uint(*subframes*->*sf1*, 73, 73+6);

ep->IODE2 = str2uint(*subframes*->*sf2*, 61, 69);

ep->IODE3 = str2uint(*subframes*->*sf3*, 271, 271+8);

ep->codeL2 = str2uint(*subframes*->*sf1*, 71, 73);

ep->L2P = str2uint(*subframes*->*sf1*, 90, 91);

*return* 0;

}

int32\_t **file2subFrames**(SF1\_3\* **sf**, FILE\* **fid**, uint8\_t **svNum**){

int32\_t **sth1**, **sth2**, **sth3**, **sth4**, **sth5**;

char **str\_0R**[8];

char **str\_GPSL1CA**[12];

char **str\_reh**[8];

char **str**[1000];

uint32\_t **svStr**;

uint32\_t **slot**;

int32\_t **subFrameNum**;

uint32\_t **slot\_SF1** = 0;

uint32\_t **slot\_SF2** = 0;

uint32\_t **slot\_SF3** = 0;

int32\_t **readres** = 0;

*while*(readres != EOF)

{

svStr = 0;

readres = fscanf( *fid*, "%d %d %d %s %s %s %u\t %u %d %d %d %s", &*sth1*, &*sth2*, &*sth3*, *str\_0R*, *str\_GPSL1CA*, *str\_reh*, &*svStr*, &*slot*, &*sth4*, &*sth5*, &*subFrameNum*, *str*);

*if* (( svStr == svNum ) && (slot >= (604800/6))) {

*if* ( subFrameNum == 1 ) {

slot\_SF1 = slot;

strncpy(*sf*->*sf1*, str, *sizeof*(*sf*->*sf1*));

}

*else* *if* (subFrameNum == 2) {

slot\_SF2 = slot;

strncpy(*sf*->*sf2*, str, *sizeof*(*sf*->*sf2*));

}

*else* *if* (subFrameNum == 3) {

slot\_SF3 = slot;

strncpy(*sf*->*sf3*, str, *sizeof*(*sf*->*sf3*));

}

*if* ((slot\_SF1 + 1 == slot\_SF2) && (slot\_SF2 + 1 == slot\_SF3)) {

sf->slot = slot\_SF1;

*return* 0;

}

}

}

*return* 1;

}

void **printEmp**(Ephemeris\* **ep**)

{

FILE\* **OUT**;

OUT = fopen("OUT.txt", "w");

printf(*OUT*, "LNAV Ephemeris (slot = %u) = \n", ep->slot );

printf(*OUT*, "\tCrs = %e \n", ep->Crs );

printf(*OUT*,"\tDn = %e \t[deg/s] \n", ep->Dn );

printf(*OUT*,"\tM0 = %e \t[deg] \n", ep->M0 );

printf(*OUT*,"\tCuc = %e \n", ep->Cuc );

printf(*OUT*,"\te = %e \n", ep->e );

printf(*OUT*,"\tCus = %e \n", ep->Cus );

printf(*OUT*,"\tsqrtA = %e \n", ep->sqrtA );

printf(*OUT*,"\ttoe = %u \n", ep->toe );

printf(*OUT*,"\tCic = %e \n", ep->Cic );

printf(*OUT*,"\tOmega0 = %e [deg] \n", ep->Omega0 );

printf(*OUT*,"\tCis = %e \n", ep->Cis );

printf(*OUT*,"\ti0 = %e \t[deg] \n", ep->i0 );

printf(*OUT*,"\tCrc = %e \n", ep->Crc );

printf(*OUT*,"\tOmega = %e \t[deg] \n", ep->Omega );

printf(*OUT*,"\tOmegaDot= %e [deg/s] \n", ep->OmegaDot );

printf(*OUT*,"\tiDot = %e \t[deg/s] \n", ep->iDot );

printf(*OUT*,"\tTgd = %e \n", ep->Tgd );

printf(*OUT*,"\ttoc = %u \n", ep->toc );

printf(*OUT*,"\taf2 = %e \n", ep->af2 );

printf(*OUT*,"\taf1 = %e \n", ep->af1 );

printf(*OUT*,"\taf0 = %e \n", ep->af0 );

printf(*OUT*,"\tWN = %u \n", ep->WN );

printf(*OUT*,"\tIODC = %u \n", ep->IODC );

printf(*OUT*,"\tURA = %u \n", ep->URA );

printf(*OUT*,"\tHealth = %u \n", ep->Health );

printf(*OUT*,"\tIODE2 = %u \n", ep->IODE2 );

printf(*OUT*,"\tIODE3 = %u \n", ep->IODE3 );

printf(*OUT*,"\tcodeL2 = %u \n", ep->codeL2 );

printf(*OUT*,"\tL2P = %u \n", ep->L2P );

}