

НИУ «МЭИ»  
Институт Радиотехники и электроники  
Им. В.А. Котельникова

Аппаратура потребителей спутниковых радионавигационных систем  
Курсовой проект  
«Расчет траектории движения спутника GPS по данным с демодулятора его сигнала»

Студент: Коробков А.Ю.

Группа: ЭР-15-17

Преподаватель: Корогодин И.В.

Москва 2022

## Оглавление

Цель работы .....	3
Исходные данные .....	3
Решение .....	3
Приложение А .....	5

## Цель работы

Изучение особенностей сигналов спутников GPS для определения положения спутника по данным с демодулятора его сигнала L1 C/A. На первом этапе происходит моделирование модуля разбора навигационного сообщения до структуры эфемерид.

## Исходные данные

Файл “in.txt” с записанными в нем данными, зафиксированными навигационным приемником по сигналу GPS L1C/A. Файл содержит наблюдения псевдодальностей и прочих радионавигационных параметров, демодулированные и разобранные данные навигационного сообщения.

## Решение

На первом этапе необходимо создать программу в среде C++, выполняющую функции аналогичные модулю разбора навигационного сообщения. Листинг созданной части программы приведен в приложении. После завершения расчета данные выводятся в окно консоли (данные представлены на рисунке 1).

```
LNAV Ephemeris (slot = 221472010) =
  Crs      = 3,850000e+001
  Dn       = 1,511921e-009          [deg/s]
  M0       = -1,101883e+002        [deg]
  Cuc      = 2,538785e-006
  e        = 2,460024e-002
  Cus      = 1,490116e-006
  sqrtA    = 5,153582e+003
  toe      = 93584
  Cic      = -2,924353e-007
  Omega0   = -1,146463e+002        [deg]
  Cis      = 4,917383e-007
  i0       = 5,496088e+001        [deg]
  Crc      = 3,547188e+002
  omega    = -5,765251e+001        [deg]
  OmegaDot = -4,759227e-007        [deg/s]
  iDot     = 5,832135e-009        [deg/s]
  Tgd      = -1,024455e-008
  toc      = 93584
  af2      = 0,000000e+000
  af1      = 6,821210e-013
  af0      = 1,579938e-004
  WN       = 149
  IODC     = 2
  URA      = 0
  Health   = 0
  IODE2    = 2
  IODE3    = 2
  codeL2   = 1
  L2P      = 0
```

Рисунок 1 – Данные полученные в результате работы программы

Правильность результатов необходимо проверить. Для этого файл BINR.bin обрабатывается с помощью программы RTKNAVI из состава RTKLIB (данные, полученные из RTKNAVI приведены на рисунке 2).

SAT	PRN	Stat	IOD	IODC	Acc	Hea	Toe	Toc	Ttrans	A (m)	e	U (deg)	OMEGA0 (deg)	omega (deg)	M0 (deg)	deltan (deg/s)	C
G19	19	OK	9766	38	0	00	2022/02/14 02:00:00	2022/02/14 02:00:00	2022/04/23 08:18:41	26560503.242	0.00897000	56.08838	-164.03953	115.03807	17.44108	2.3429E-07	-
G20	20	-	-	-	0	00	-	-	-	0.000	0.00000000	0.00000	0.00000	0.00000	0.00000	0.0000E+00	C
G21	21	OK	514	2	0	00	2022/02/14 01:59:41	2022/02/14 01:59:41	2022/04/23 08:18:41	26559409.620	0.02460024	54.96088	-114.64634	-57.65251	-110.1882	2.7215E-07	-
G22	22	OK	3135	122	0	00	2022/02/14 02:00:00	2022/02/14 02:00:00	2022/04/23 08:18:41	26560169.875	0.01288441	55.09904	14.59397	-107.3007	-127.8451	2.4147E-07	-

  

OMEGAdot (deg/s)	IDOT (deg/s)	af0 (ns)	af1 (ns/s)	af2 (ns/s)	TGD (ns)	BGD5a(r)	BGD5b(r)	Cuc(rad)	Cus(rad)	Crc(m)	Crs(m)	Cic(rad)	Cis(rad)	Code	Flag
-4.5779E-07	1.0293E-08	118619.3	0.0050	0.0000	-15.8	0.0	0.0	6.3479E-06	7.8883E-06	2.4034E+02	1.2191E+02	-5.5879E-08	-9.6858E-08	1	0
0.0000E+00	0.0000E+00	0.0	0.0000	0.0000	0.0	0.0	0.0	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
-4.7592E-07	5.8321E-09	157993.7	0.0007	0.0000	-10.2	0.0	0.0	2.5388E-06	1.4901E-06	3.5472E+02	3.8500E+01	-2.9244E-07	4.9174E-07	1	0
-4.2965E-07	2.8281E-08	213230.0	0.0057	0.0000	-9.8	0.0	0.0	-3.8557E-07	9.9484E-06	1.8750E+02	-6.5938E+01	1.6950E-07	-1.2107E-07	1	0

Рисунок 2 – Данные программы RTKNAVI

При сопоставлении результатов работы двух программ видно, что полученные из них данные совпадают. Следовательно, можно приступить к выполнению второй части КП.

## Приложение А

```
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <string.h>
#include <cmath>
#include <fstream>
#include <iostream>

using namespace std;

//setlocale(LC_ALL, "rus"); // корректное отображение Кириллицы

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;
};
const int32_t subFrameLength = 300;
struct SF1_3 {
    uint32_t slot;
    char sf1[subFrameLength+1];
    char sf2[subFrameLength+1];
    char sf3[subFrameLength+1];
};
void printEmp(Ephemeris* ep);
int32_t file2subFrames(SF1_3* sf, FILE* fid, uint8_t svNum);
int32_t subFrames2Eph(Ephemeris* ep, SF1_3* subframes);

int main(void)
{
    setlocale(LC_ALL, "rus"); // корректное отображение Кириллицы
    uint8_t svNum = 21;
    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(" Сабфреймы не декодированы\n ");
    }
    free(ep);
    fclose(fid);
}
else {
    printf(" Сабфреймы не найдены\n ");
}
}
else {
    printf(" Не открыт файл 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 str3int(uint64_t ans, int count_bit) {

```

```

int64_t Ians = 0;
if (count_bit == 8) {
    if (bool((1<<7) & ans)){
        ans |= 0xFFFFFFFFFFFFF00;
        Ians = ~(ans - 1);
        return -Ians;

    }
}
if (count_bit == 14) {
    if (bool((1<<13) & ans)) {
        ans |= 0xFFFFFFFFFFC000;
        Ians = ~(ans - 1);
        return -Ians;
    }
}
if (count_bit == 16) {
    if (bool((1 << 15) & ans)) {
        ans |= 0xFFFFFFFFF0000;
        Ians = ~(ans - 1);
        return -Ians;
    }
}
if (count_bit == 22) {
    if (bool((1 << 21) & ans)) {
        ans |= 0xFFFFFFFFFC00000;
        Ians = ~(ans - 1);
        return -Ians;
    }
}
if (count_bit == 24) {
    if (bool((1 << 23) & ans)) {

```



```

        ans |= 0xFFFFFFFF00000000;

        Ians = ~(ans - 1);

        return -Ians;

    }

}

if (count_bit == 32) {
    if (bool((1 << 31) & ans)) {
        ans |= 0xFFFFFFFF00000000;

        Ians = ~(ans - 1);

        return -Ians;

    }

}

return ans;

}

int64_t str4uint(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->WN = str2uint(subframes->sf1, 61, 71);

```

```

ep->URA = str2uint(subframes->sf1, 73, 77);
ep->toe = str2uint(subframes->sf2, 271, 287)*pow(2,4);
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);
ep->Crc = str3int(str2uint(subframes->sf3,181,181+16),16)*pow(2, -5);
ep->Dn = str3int(str2uint(subframes->sf2, 91, 91+16), 16)*pow(2, -43);
ep->Cuc = str3int(str2uint(subframes->sf2,151,151+16),16)*pow(2, -29);
ep->Cus = str3int(str2uint(subframes->sf2,211,211+16),16)*pow(2, -29);
ep->e = str4uint(subframes->sf2,167, 167+8, 181, 181+24) * pow(2, -33);
ep->sqrtA = str4uint(subframes->sf2,227, 227+8, 241, 241+24) * pow(2, -19);
ep->Cic = str3int(str2uint(subframes->sf3,61,61+16),16)*pow(2, -29);
ep->Omega0 = str3int(str4uint(subframes->sf3,77, 77+8, 91, 91+24),32)*pow(2, -31)*180;
ep->Cis = str3int(str2uint(subframes->sf3,121,121+16),16)*pow(2, -29);
ep->i0 = str3int(str4uint(subframes->sf3,137, 137+8, 151, 151+24),32)*pow(2, -31)*180;
ep->omega = str3int(str4uint(subframes->sf3,197, 197+8, 211, 211+24),32)*pow(2, -31)*180;
ep->OmegaDot = str3int(str2uint(subframes->sf3,241,241+24),24)*pow(2, -43)*180;
ep->iDot = str3int(str2uint(subframes->sf3,279,279+14),14)*pow(2, -43)*180;
ep->Tgd = str3int(str2uint(subframes->sf1,197,197+8),8)*pow(2, -31);
ep->toc = str3int(str2uint(subframes->sf1,219,219+16),16)*pow(2, 4);
ep->af2 = str3int(str2uint(subframes->sf1,241,241+8),8)*pow(2, -55);
ep->af1 = str3int(str2uint(subframes->sf1,249,249+16),16)*pow(2, -43);
ep->af0 = str3int(str2uint(subframes->sf1,271,271+22),22)*pow(2, -31);
ep->IODC = str4uint(subframes->sf1,83, 83+2, 211, 211+8);
ep->Crs = str3int(str2uint(subframes->sf2,69,69+16),16)*pow(2, -5);
ep->M0 = str3int(str4uint(subframes->sf2,107, 107+8, 121, 121+24),32)*pow(2, -31)*180;

return 0;

```

```

}

int32_t file2subFrames(SF1_3* sf, FILE* fid, uint8_t svNum){
    int32_t sth1, sth2, sth3, sth4, sth5;
    char str_OR[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_OR, 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)
{
    printf("LNAV Ephemeris (slot = %u) = \n", ep->slot    );
    printf("\tCrs    = %e          \n", ep->Crs    );
    printf("\tDn     = %e \t[deg/s]    \n", ep->Dn     );
    printf("\tM0     = %e \t[deg]      \n", ep->M0     );
    printf("\tCuc    = %e          \n", ep->Cuc    );
    printf("\te      = %e          \n", ep->e      );
    printf("\tCus    = %e          \n", ep->Cus    );
    printf("\tsqrtA  = %e          \n", ep->sqrtA   );
    printf("\ttoe    = %u          \n", ep->toe    );
    printf("\tCic    = %e          \n", ep->Cic    );
    printf("\tOmega0  = %e \t[deg]      \n", ep->Omega0  );
    printf("\tCis    = %e          \n", ep->Cis    );
    printf("\ti0     = %e \t[deg]      \n", ep->i0     );
    printf("\tCrc    = %e          \n", ep->Crc    );
    printf("\tomega   = %e \t[deg]      \n", ep->omega   );
    printf("\tOmegaDot = %e \t[deg/s]    \n", ep->OmegaDot );
    printf("\tiDot    = %e \t[deg/s]    \n", ep->iDot    );
    printf("\tTgd    = %e          \n", ep->Tgd    );
    printf("\ttoc    = %u          \n", ep->toc    );
}

```

```

printf("\taf2   = %e      \n", ep->af2   );
printf("\taf1   = %e      \n", ep->af1   );
printf("\taf0   = %e      \n", ep->af0   );
printf("\tWN    = %u      \n", ep->WN    );
printf("\tIODC   = %u      \n", ep->IODC   );
printf("\tURA    = %u      \n", ep->URA   );
printf("\tHealth = %u      \n", ep->Health );
printf("\tIODE2  = %u      \n", ep->IODE2  );
printf("\tIODE3  = %u      \n", ep->IODE3  );
printf("\tcodeL2 = %u      \n", ep->codeL2 );
printf("\tL2P    = %u      \n", ep->L2P    );
}

//Описывает поток для записи данных в файл.
//ofstream f;

//Открываем файл в режиме записи,
// режим ios::out устанавливается по умолчанию.
//f.open("D:\\result.txt");

```