МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**)» (**МАИ)**

Специальность 10.03.01 Информационная безопасность

Группа М4О-112Б-21

КУРСОВАЯ РАБОТА

по Языки программирования

На тему: Расшифровка сообщений файла приёмника (Вариант 6. Эфемериды GPS)

Автор курсовой работы: Доморощенов Данила Сергеевич

Руководитель: Герко Сергей Александрович

Москва
2021г.

СОДЕРЖАНИЕ

1. Код разработанной программы 3

2. Результаты программы 18

3. Список литературы 19

Код разработанной программы

По заданию необходимо произвести выборку и преобразования сообщений эфемерид GPS с идентификатором GE. Исходя из предоставленной спецификации, сообщения данного вида могут иметь длину 123, 160 и 168 байт в зависимости от наличия опциональных полей.

Принцип программы состоит в следующем:

1. Программа открывает бинарный файл на чтение и запись;
2. Подсчитывает количество сообщений с нужным идентификатором GE;
3. Читает файл с самого начала, выбирая нужные сообщения;
4. Находя нужное сообщение, определяет его размер и записывает в соответствующую структуру;
5. Выводит расшифрованное сообщение на экран;

Текст основного файла программы “main.c” представлен в листинге 1.

Листинг 1 – код файла “main.c”

```
#include <stdio.h>
#include <iostream>
#include <assert.h>
#include <string.h>
#include <stdlib.h>
#include <stdbool.h>
#include <locale.h>

#include "checksum.h"

#define IS_OK(g) (g.req.sv >= 1 && g.req.sv <= 37)

using namespace std;
```

```
// считывание сообщений длинной 168 байт

void printGPSEphem3(GPSEphem3* g)
{
   printf("Эфермиды GPS (Required data + Optional data(big):\n");
   printf("\t %u %u %u \n", g->req.sv, g->req.tow, g->req.flags);
   printf("\t %d %d %d \n", g->req.iodc, g->req.toc, g->req.ura);
   printf("\t %u %d %e \n", g->req.healthS, g->req.wn, g->req.tgd);

   printf("\t %e %e %e \n", g->req.af2, g->req.af1, g->req.af0);
   printf("\t %d %d %e \n", g->req.toe, g->req.iode, g->req.rootA);
   printf("\t %e %e %e \n", g->req.ecc, g->req.m0, g->req.omega0);

   printf("\t %e %e %e \n", g->req.inc0, g->req.argPer, g->req.deln);
   printf("\t %e %e %e \n", g->req.omegaDot, g->req.incDot, g->req.crc);
   printf("\t %e %e %e %e \n", g->req.crs, g->req.cuc, g->req.cic, g->req.cis);

   printf("\t %u %d %d \n", g->opt.navType, g->opt.lTope, g->opt.lTopc);
   printf("\t %e %e %d \n", g->opt.dADot, g->opt.fDelnDot, g->opt.cURAoe);
   printf("\t %d %d %d \n", g->opt.cURAoc, g->opt.cURAoc1, g->opt.cURAoc2);

   printf("\t %e %e \n", g->opt.isc.Isc1.fIscL1CA, g->opt.isc.Isc1.fIscL2C);

   printf("\t %e %e \n", g->opt.isc.Isc1.fIscL5I5, g->opt.isc.Isc1.fIscL5Q5);
   printf("\t %e %e %e \n", g->opt.isc.Isc2.fIscL1CP, g->opt.isc.Isc2.fIscL1CD, g-
>opt.DAf0);

   printf("Контрольная сумма: \t Заданная: %u \n", g->cs);
```

```
}
// считывание сообщений длинной 160 байт

void printGPSEphem2(GPSEphem2* g)
{
   printf("Эфермиды GPS (Required data + Optional data(small)): \n");
   printf("\t %u %u %u \n", g->req.sv, g->req.tow, g->req.flags);
   printf("\t %d %d %d \n", g->req.iodc, g->req.toc, g->req.ura);
   printf("\t %u %d %e \n", g->req.healthS, g->req.wn, g->req.tgd);

   printf("\t %e %e %e \n", g->req.af2, g->req.af1, g->req.af0);
   printf("\t %d %d %e \n", g->req.toe, g->req.iode, g->req.rootA);
   printf("\t %e %e %e \n", g->req.ecc, g->req.m0, g->req.omega0);

   printf("\t %e %e %e \n", g->req.inc0, g->req.argPer, g->req.deln);
   printf("\t %e %e %e \n", g->req.omegaDot, g->req.incDot, g->req.crc);
   printf("\t %e %e %e %e \n", g->req.crs, g->req.cuc, g->req.cic, g->req.cis);

   printf("\t %u %d %d \n", g->opt.navType, g->opt.lTope, g->opt.lTopc);
   printf("\t %e %e %d \n", g->opt.dADot, g->opt.fDelnDot, g->opt.cURAoe);
   printf("\t %d %d %d \n", g->opt.cURAoc, g->opt.cURAoc1, g->opt.cURAoc2);

   printf("\t %e %e \n", g->opt.isc.fIscL1CA, g->opt.isc.fIscL2C);
   printf("\t %e %e \n", g->opt.isc.fIscL5I5, g->opt.isc.fIscL5Q5);

   printf("Контрольная сумма: \t Заданная: %u \n", g->cs);
}
```

```
// считывание сообщений длинной 123 байт

void printGPSEphem1(GPSEphem1* g)
{
   printf("Эфемериды GPS (Only required data):\n");
   printf("\t %u %u %u \n", g->req.sv, g->req.tow, g->req.flags);
   printf("\t %d %d %d \n", g->req.iodc, g->req.toc, g->req.ura);
   printf("\t %u %d %e \n", g->req.healthS, g->req.wn, g->req.tgd);

   printf("\t %e %e %e \n", g->req.af2, g->req.af1, g->req.af0);
   printf("\t %d %d %e \n", g->req.toe, g->req.iode, g->req.rootA);
   printf("\t %e %e %e \n", g->req.ecc, g->req.m0, g->req.omega0);

   printf("\t %e %e %e \n", g->req.inc0, g->req.argPer, g->req.deln);
   printf("\t %e %e %e \n", g->req.omegaDot, g->req.incDot, g->req.crc);
   printf("\t %e %e %e %e \n", g->req.crs, g->req.cuc, g->req.cic, g->req.cis);

   printf("Контрольная сумма: \t Заданная: %u \n", g->cs);
}

int main()
{
   setlocale(LC_ALL, "rus");
   FILE* file = fopen("rks.dat", "r+b");
   assert(file);
   const char idef[] = {0x0A, 'G', 'E', '\0'}; // щаблон идентификатора
   char buf[4] = {0}; // массив для текущих 4 байт сообщения
   int count = 0; // номер сообщения
```

```
char mesSize[4] = {0}; // массив для дескриптора
unsigned char mes[173] = {0}; // массив хранения сообщения
mes[0] = 'G';
mes[1] = 'E';

// подсчет сообщений с нужным идентификатором

fread(buf, 1, 2, file);
while(fread(buf + 2, 1, 1, file) > 0) // пока не конец файла
{
   if (strcmp(buf, idef) == 0)
      count++;
   for(int i = 0; i < 3; i++)
      buf[i] = buf[i + 1];
}
printf("Всего сообщений: %d \n\n", count);

fseek(file, 0, SEEK_SET); // возвращаемся в начало файда
count = 0;
fseek(file, 2, SEEK_SET);
fread(buf, 1, 2, file);
while (fread(buf + 2, 1, 1, file) > 0)
{
   if (strcmp(buf, idef) == 0) // Если найдено сообщение с подходящим
идентификатором
   {
      count++;
```

```
fread(mesSize, 1, 3, file); // считываем дескриптор
long size = strtol(mesSize, NULL, 16); // определение размера сообщения
dataSize s = (size == 168) ? REQOPT_DATA2 : (size == 160) ? REQOPT_DATA1 : (size == 123) ? REQ_DATA : WRONG_DATA;
if (s == REQOPT_DATA2) // если нашлось сообщение размером 168 байт
{
   GPSEphem3 g;
   fread((void*)&g, 1, 168, file);
   if (!IS_OK(g))
      continue;
   mes[2] = '0';
   mes[3] = 'A';
   mes[4] = '8';
   memcpy(mes + 5, (void*)&g, 168);
   printf("%d. ", count);
   printGPSEphem3(&g);
   printf("\t\t\t Пересчитанная: %u\n\n", cs(mes, 172));
}
else
if (s == REQOPT_DATA1) // если нашлось сообщение размером 160 байт
{
   GPSEphem2 g;
   fread((void*)&g, 1, 160, file);
   if (!IS_OK(g))
      continue;
   mes[2] = '0';
```

```
            mes[3] = 'A';
            mes[4] = '0';
            memcpy(mes + 5, (void*)&g, 160);
            printf("%d. ", count);
            printGPSEphem2(&g);
            printf("\t\t\t Пересчитанная: %u\n\n", cs(mes, 164));
        }
        else
        if (s == REQ_DATA) // если нашлось сообщение размером 123 байта
        {
            GPSEphem1 g;
            fread((void*)&g, 1, 123, file);
            if (!IS_OK(g))
                continue;
            mes[2] = '0';
            mes[3] = '7';
            mes[4] = 'B';
            memcpy(mes + 5, (void*)&g, 123);
            printf("%d. ", count);
            printGPSEphem1(&g);
            printf("\t\t\t Пересчитанная: %u\n\n", cs(mes, 127));
        }
        else
        {
            printf("%d. Сообщение неверного размера\n\n", count);
            continue;
        }
    }
```

```
    for(int i = 0; i < 3; i++)
        buf[i] = buf[i + 1];
    }
    return 0;
}
```

Текст файла расчета контрольной суммы сообщения “checksum.h” представлен в листинге 2.

Листинг 2 – код файла “checksum.h”

```
#ifndef CS
#define CS
#include "GPSEphem.h"

typedef unsigned char u1;
enum
{
    bits = 8,
    lShift = 2,
    rShift = bits - lShift
};

#define ROT_LEFT(val) ((val << lShift) | (val >> rShift))

u1 cs(u1 const* src, int count)
{
    u1 res = 0;
    while (count--)
        res = ROT_LEFT(res) ^ *src++;
    return ROT_LEFT(res);
```

```
}


#endif
```

Текст файла “GPSEphem.h”, содержащего структуру, представлен в листинге 3.

Листинг 3 – код файла “GPSEphem.h”

```
#pragma pack(push, 1)

#define GPSEphem_H


#include <stdint.h>


typedef enum {REQ_DATA, REQOPT_DATA1, REQOPT_DATA2,
WRONG_DATA} dataSize;


typedef struct GPSEphemeris1

{

   struct GpsEphReqData1

   {

      uint8_t sv;

      uint32_t tow;

      uint8_t flags;

      int16_t iodc;

      int32_t toc;

      int8_t ura;

      uint8_t healthS;

      int16_t wn;

      float tgd;

      float af2;
```

```
        float af1;
        float af0;
        int32_t toe;
        int16_t iode;
        double rootA;
        double ecc;
        double m0;
        double omega0;
        double inc0;
        double argPer;
        float deln;
        float omegaDot;
        float incDot;
        float crc;
        float crs;
        float cuc;
        float cus;
        float cic;
        float cis;
    } req; // Required data
    uint8_t cs;
} GPSEphem1;

typedef struct GPSEphemeris2
{
    struct GpsEphReqData2
    {
        uint8_t sv;
```

```
uint32_t tow;
uint8_t flags;
int16_t iodc;
int32_t toc;
int8_t ura;
uint8_t healthS;
int16_t wn;
float tgd;
float af2;
float af1;
float af0;
int32_t toe;
int16_t iode;
double rootA;
double ecc;
double m0;
double omega0;
double inc0;
double argPer;
float deln;
float omegaDot;
float incDot;
float crc;
float crs;
float cuc;
float cus;
float cic;
float cis;
```

```
    } req; // Required data
    struct GpsEphOptData2
    {
        uint8_t navType;
        int32_t lTope;
        int32_t lTopc;
        double dADot;
        float fDelnDot;
        int8_t cURAoe;
        int8_t cURAoc;
        int8_t cURAoc1;
        int8_t cURAoc2;
        struct GpsEphCnavIsc2
        {
            float fIscL1CA;
            float fIscL2C;
            float fIscL5I5;
            float fIscL5Q5;
        } isc;
        float DAf0;
    } opt; // Optional data
    uint8_t cs;
} GPSEphem2;

typedef struct GPSEphemeris3
{
    struct GpsEphReqData3
    {
```

```
uint8_t sv;
uint32_t tow;
uint8_t flags;
int16_t iodc;
int32_t toc;
int8_t ura;
uint8_t healthS;
int16_t wn;
float tgd;
float af2;
float af1;
float af0;
int32_t toe;
int16_t iode;
double rootA;
double ecc;
double m0;
double omega0;
double inc0;
double argPer;
float deln;
float omegaDot;
float incDot;
float crc;
float crs;
float cuc;
float cus;
float cic;
```

```
    float cis;
} req; // Required data
struct GpsEphOptData3
{
    uint8_t navType;
    int32_t lTope;
    int32_t lTopc;
    double dADot;
    float fDelnDot;
    int8_t cURAoe;
    int8_t cURAoc;
    int8_t cURAoc1;
    int8_t cURAoc2;
    union
    {
        struct GpsEphCnavIsc3
        {
            float fIscL1CA;
            float fIscL2C;
            float fIscL5I5;
            float fIscL5Q5;
        } Isc1;
        struct GpsEphCnav2Isc3
        {
            float fIscL1CP;
            float fIscL1CD;
        } Isc2;
    } isc;
```

```
        float DAf0;
    } opt; // Optional data
    uint8_t cs;
} GPSEphem3;

#pragma pack(pop)
```

Результат работы программы

Вид бинарного файла (строки первого сообщения) до выполнения программы представлен на рисунке 1.

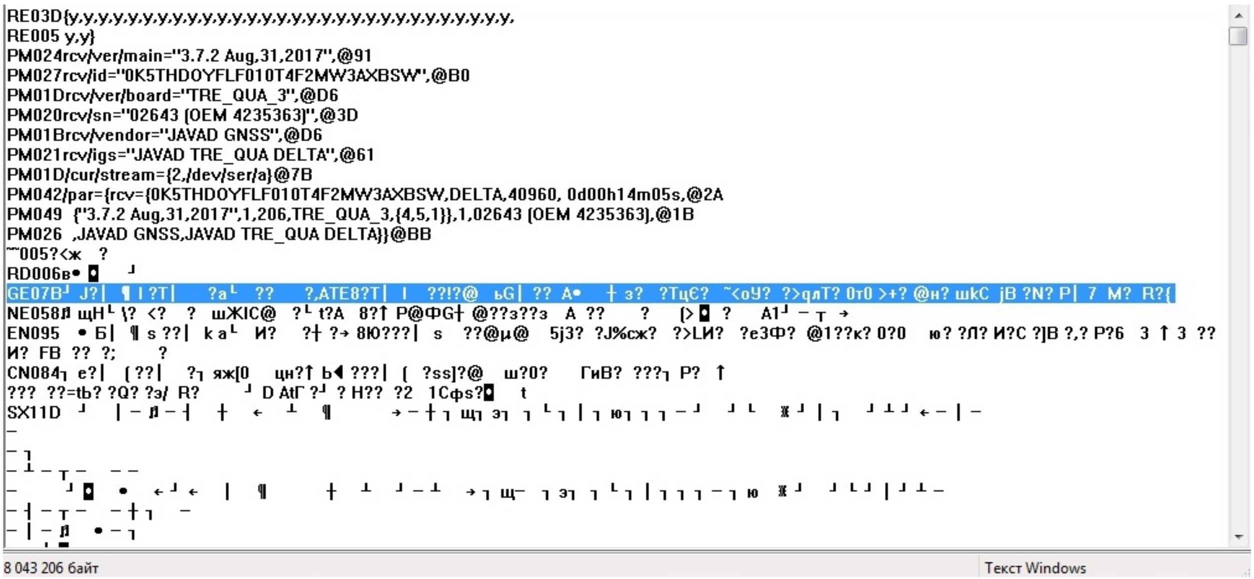

Рисунок 1 – вид бинарного файла до выполнения программы

Результат чтения бинарного файла представлен на рисунке 2.

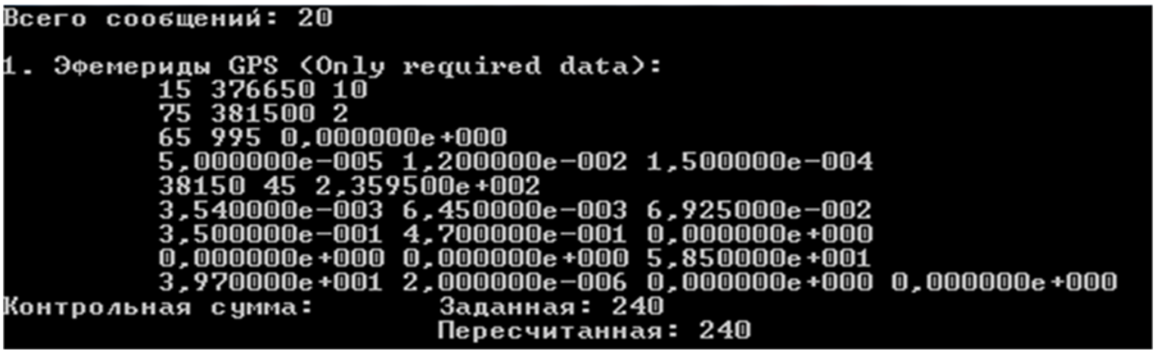

Рисунок 2 – результат работы программы

Список использованной литературы

1. GNSS Receiver External Interface Specification Reflects Firmware Version 4.1.00
2. BDS-SIS-ICD-B1I-3.0 2019-02
3. IS-GPS-200J 25-APR-2018