Протокол обмена с сервером RFMServer по UDP.

Порт сервера: 510
Порт клиента – любой

В начале сессии нужно на сервер послать команду выбора режима – один из символов:
‘0’ – FSK
‘1’ – Lora
Ответом будет тот же символ.

Все обменные пакеты имеют структуру:

Dst IP	Src IP	MsgId	Msg
4 bytes	4 bytes	2 bytes	N bytes

Dst IP – адрес БРК: для 1-го 10.6.1.201
для 2-го 10.6.1.202

Src IP – адрес клиента, для НС может быть любым, но не 0.0.0.0 и 255.255.255.255
для БРК он будет таким как указан выше

MsgId – идентификатор сообщения, см. ниже.

Все эти поля имеют байтовый порядок согласно стандарту Network-order, т.е. Big-endian. Для примера, поле Dst IP с адресом 10.6.1.201 будет представлено байтами в порядке:
0x0A, 0x06, 0x01, 0xC9.

В сообщении Msg порядок байт – платформо-определяемый, т.е. такой как они представлены в процессоре.

Идентификаторы сообщений:

Запрос ТМИ 0, 1, 2, 3, 4:
MsgId = 1, 3, 5, 7, 9
Msg – пусто

Ответ:
Msg – ТМИ0, ТМИ1, ТМИ2, ТМИ3, ТМИ4, соответственно с MsgId = 2, 4, 6, 8, 10.

Запрос архивного кадра:
MsgId = 11
Msg – uint16_t kadr_num - номер архивного кадра назад от текущего.

Ответ:
MsgId = 12
Msg – запрашиваемый ТМИ из архива

Запрос переменной:
MsgId = 13
Msg – структурная последовательность:

```
typeIdxMask  varid1;
uint8_t leng1;
typeIdxMask  varid2;
uint8_t leng2;
```

…
…
uint32_t 0x00000000;

Ответ:
MsgId = 14
Msg – последовательность из запрашиваемых переменных
var1 размера leng1;
var2 размера leng2
…
var_последняя размера leng_последняя

Здесь
varid_n – адрес запрашиваемой переменной,
leng_n – ее длина
перечень должен заканчиваться uint32_t = 0.

Установка переменной:
MsgId = 15
Msg – структурная последовательность:
; первая переменная
typeIdxMask varid1;
uint8_t leng1;
uint8_t var1[leng1];
; вторая переменная
typeIdxMask varid2;
uint8_t leng2;
uint8_t var2[leng2];
…
…
uint32_t 0x00000000;
uint8_t pad_bytes[x]=0;
uint32_t crc;

Ответ:
MsgId = 16

Здесь
varid_n – адрес переменной,
leng_n – ее размер,
var_n[leng_n] – значение переменной, размера leng_n,
pad_bytes[x] – дополнение нулями для выравнивания всего сообщения до 4-х байтовой границы
crc – контрольная сумма, ее алгоритм вычисления:
инициализационное значение = -1
вычисляем по массиву 4-х байтовых слов с использованием значения времени BTime, содержащегося в последнем маяке, в порядке:
1 (BTime ^ 0x01041964)
2.. все слова в сообщении Msg, включая pad_bytes[]

Текст программы вычисления crc

```
static const uint32_t CrcTable[16] = { // Nibble lookup table for 0x04C11DB7 polynomial
  0x00000000, 0x04C11DB7, 0x09823B6E, 0x0D4326D9,
  0x130476DC, 0x17C56B6B, 0x1A864DB2, 0x1E475005,
  0x2608EDB8, 0x22C9F00F, 0x2F8AD6D6, 0x2B4BCB61,
```

```
  0x350C9B64, 0x31CD86D3, 0x3C8EA00A, 0x384FBDBD
};


uint32_t CRCCalc(uint32_t *crc32, uint32_t *u32_arr, uint32_t dwsize) {
  uint32_t Crc;
  Crc = *crc32;

  while(dwsize--) {
    Crc = Crc ^ *u32_arr; // Apply all 32-bits
    u32_arr += 1;
    // Process 32-bits, 4 at a time, or 8 rounds
    Crc = (Crc << 4) ^ CrcTable[Crc >> 28]; // Assumes 32-bit reg, masking index to 4-bits
    Crc = (Crc << 4) ^ CrcTable[Crc >> 28]; //  0x04C11DB7 Polynomial used in STM32
    Crc = (Crc << 4) ^ CrcTable[Crc >> 28];
    Crc = (Crc << 4) ^ CrcTable[Crc >> 28];
    Crc = (Crc << 4) ^ CrcTable[Crc >> 28];
    Crc = (Crc << 4) ^ CrcTable[Crc >> 28];
    Crc = (Crc << 4) ^ CrcTable[Crc >> 28];
    Crc = (Crc << 4) ^ CrcTable[Crc >> 28];
    }
  *crc32 = Crc;
  return Crc;
}
```

Пример сформированного Msg на запись DevId=6, VarId=5, Offset=4, Leng=2, var=0x0201, BTime=1234(дес) :

0x20, 0x00, 0x00, 0x65,	0x02,	0x01, 0x02,	0x00, 0x00, 0x00, 0x00,	0x00,	0xBA, 0x0C, 0x88, 0x39
(varid)	(leng)	(var)	(end term)	(pad)	(crc)

Установка байтовой переменной с отложенной записью:
MsgId = 17
Msg – структурная последовательность:

```
; первая переменная
uint16_t  uncal_id1;
uint8_t  cmd = 0;
uint32_t  time1;
typeIdxMask  varid1;
uint8_t  var1;
; вторая переменная
uint16_t  uncal_id2;
uint8_t  cmd = 0;
uint32_t  time2;
typeIdxMask  varid2;
uint8_t  var2;
…
…
uint32_t  0x00000000;
uint8_t  pad_bytes[x]=0;
uint32_t  crc;
```

Ответ:
MsgId = 18

Здесь

uncial_id – любое уникальное число для данной переменной, кроме 0.
time – время отложенности в десятых долях секунды;
все остальное – как для MsgId = 15.

Пример сформированного Msg на отложенную запись uncial_id=0x321, time=100.0c, DevId=6, VarId=5, Offset=4, var=0x34, BTime=1234(дес)

0x21, 0x03, 0x00, 0xEB, 0x03, 0x00, 0x00,
(uncial_id) (cmd) (time)

0x20, 0x00, 0x00, 0x65, 0x34, 0x00, 0x00, 0x00, 0x00, 0x79, 0xFE, 0x4C, 0x0F
(varid) (var) (end term+pad) (crc)