**Протокол обмена между скважинными приборами и ПО верхнего уровня Ramtools**

1. ПК с запущенной программой Ramtools выступает мастером. Программа отправляет прибору команды и обрабатывает ответы.
2. Используется протокол UDP, порт прибора 48889, порт ПК 48888.
3. Список используемых команд с кодами:

|  |  |  |
| --- | --- | --- |
| Hex code | Команда | Описание |
| 0x00 | Reset | Перезапуск МК |
| 0x01 | GetState | Запрос структуры состояния (State) – значение RTC, акселерометров, датчиков температуры, режима работы, блока microSD и др. Данную команду программа Ramtools отправляет по настраиваемому таймеру (обычно 0.1-1 с) |
| 0x02 | GetSettings | Запрос структуры параметров прибора (Settings) – имя прибора, серийный номер, масса, длина и др., а также сетевые настройки и параметры интерфейса RS485 (только для СКПБ).  Данную команду Ramtools отправляет только в двух случаях:  - при появлении связи с прибором;  - при изменении параметров. |
| 0x03 | GetCalibrates | Запрос структуры калибровок – специфичный массив параметров, зависящий от конкретного прибора. Например, для АМАК – это массив режимов работы.  Команда отправляется однократно аналогично GetSettings |
| 0x04 | GetBaseChannels | Запрос массива каналов. Для АМАК это список с описанием скалярных каналов (время, акселерометры, температура и т.п.) и каналов волновых картин.  Команда отправляется однократно аналогично GetSettings |
| 0x05 | GetStartups | Запрос массива с информацией о запусках. Когда прибору подается питание, сразу после инициализации контроллер записывает структуру данных с текущими значениями датчиков.  Команда отправляется однократно аналогично GetSettings |
| 0x06 | SetCommon | Установка основных параметров прибора – имя, серийный номер и др.  Отправляется по действию пользователя |
| 0x07 | SetNet | Установка сетевых параметров прибора.  Отправляется по действию пользователя |
| 0x08 | SetWork | Установка параметров работы. Для АМАК это интервал измерений и время старта (опционально).  Отправляется по действию пользователя |
| 0x09 | SetRS485 | Установка параметров интерфейса RS485 (только СКПБ) – для АМАК не используется.  Отправляется по действию пользователя |
| 0x0a | SetFromFile | Установка параметров SetCommon, SetNet и SetRS485 из заголовков имеющегося ADT-файла – сервисная команда.  Отправляется по действию пользователя |
| 0x0b | Stop | Команда остановки – немедленно останавливает процесс измерений.  Отправляется по действию пользователя |
| 0x0c | SetCalibrates | Установка калибровок – для АМАК это список режимов работы с параметрами.  Отправляется по действию пользователя. |
| 0x0d | SetTime | Синхронизация часов с ПК.  Отправляется по действию пользователя. |
| 0x0e | ReadBlock | Чтение блока microSD.  Отправляется по действию пользователя. |
| 0x0f | WriteBlock | Запись блока microSD.  Отправляется по действию пользователя. |
| 0x10 | ResetSD | Сброс счетчика блоков microSD.  Отправляется по действию пользователя. |
| 0x11 | ResetEEPROM | Сброс структуры данных EEPROM с установкой значений по умолчанию (которые прошиты в МК).  Отправляется по действию пользователя. |
| 0x12 | ResetStartups | Сброс массива с информацией о перезапусках.  Отправляется по действию пользователя. |
| 0x13 | GetData | Запрос текущего кадра данных (его фрагмента, помещающегося в один кадр Ethernet). |

1. Структуры данных

#pragma pack (push, 1)

typedef struct {

char Day; //0x00yymmdd

char Month;

char Year;

char Res1;

char Second; //0x00hhnnss

char Minute;

char Hour;

char Res2;

} TRtc; //8 bytes

typedef struct {

char Dll[8]; //Dll name AMAK

char Name[16]; //tool name АМАК

int Serial; //tool serial 1

float Diameter;

float Length;

float Weight;

} TCommonSettings; //40 bytes

typedef struct {

char MAC[6];

char IP[4];

char Mask[4];

int DHCP;

char DHCPClientIP[4];

} TNetSettings; //22 bytes

typedef struct {

int Mode; //режим работы (0=stop, 1=wait, 2=work)

int Interval; //интервал работы в секундах (work)

TRtc Alarm; //начало работы (переход из wait в work)

int WaitTillMove; //1 означает, что перед переходом из wait в work нужно дождаться движения

float MoveThreshold; //порог акселерометров для определения движения

int MoveDuration; //столько секунд должен держаться порог

int WorkWhenStay; //1 означает, что нужно что-то делать на стоянке – только СКПБ

float StayThreshold; //порог акселерометров для определения стоянки

int StayDuration; //столько секунд должен держаться порог

int StayCheckInterval; //минимальный интервал проверки стоянки в секундах

int WorkByRS485; //запускать задание WorkBy... по команде старт от телесистемы. 1 - Interval, 2 - Stay

} TWorkSettings; //52 bytes

typedef struct {

int BaudRate;

char Address;

char ReplyLength;

} TRS485Settings; //только СКПБ

typedef struct {

int Mark; //метка EEPROM=”ADT”

TCommonSettings Common;

TNetSettings Net;

TWorkSettings Work;

TRS485Settings RS;

int Version; //версия прошивки

} TSettings; //4+40+22+52+6=124 bytes

typedef struct {

int Mode; //режим работы - дублируется из Settings.Work

TRtc Rtc; //значение выбранных часов

float Axel[3]; //значения ускорений в mG, в соответствии с осями прибора

float Temperature; //значение выбранного температурного сенсора

float Voltage; //напряжение аккумуляторов

} TStateSensors; //32 bytes

typedef struct {

TStateSensors Sensors;

int SDBlock;

int SDOffset;

int SDSize; //число блоков

int StartupCount;

int Errors; //маска ошибок – например, неисправность RTC , акселерометра и пр.

int Pending;

char HostMAC[6];

char HostIP[4];

} TState; //66 bytes – отправляется по запросу GetState

typedef struct {

int StartupCount;

TStateSensors Sensors[31];

} TStartups; //массив перезапусков, хранится в первых двух блоках microSD

typedef struct {

char Name[14];

char Type; //определяет тип данных (char, int...) и, соответственно, размер элемента

char Flags; //например, unsigned, hex, string

short int SubChannels; //например, 6 подканалов по 1000 точек

int Length; //общее число элементов

short int MeasurePoint; //в сантиметрах

float Step; //в мкс

short int TsmDistance; //в сантиметрах

short int RecDistance; //в сантиметрах

} TChannel; //32 bytes

typedef struct {

char Dll[8]; //AMAK

char Name[16]; //АМАК

int Serial; //1

char Direction; //0 - from PC to controller, 1 - from controller to PC

char Command; //код команды

unsigned short int Id; //порядковый номер команды в данной сессии - опционально

unsigned short int DataLength; //длина поля данных (payload) в текущем кадре Ethernet

unsigned short int FragmentId; //номер фрагмента, если ответ не умещается в один кадр Ethernet

unsigned short int FragmentLeft; //число оставшихся фрагментов, если ответ не умещается в один кадр Ethernet

} TDirectHeader; //38 bytes – заголовок команды либо ответа по Ethernet

#pragma pack (pop)

1. Когда прибор получает команду от Ramtools, он выполняет требуемое действие и затем отправляет ПК ответ. Ответ содержит такой же заголовок DirectHeader, но поле Direction устанавливается в 1.
2. После DirectHeader следуют данные, зависящие от конкретной команды.

|  |  |  |
| --- | --- | --- |
| Команда | Поле данных в запросе | Поле данных в ответе |
| Reset | отсутствует | отсутствует |
| GetState | отсутствует | Структура TState |
| GetSettings | отсутствует | Структура TSettings |
| GetCalibrates | отсутствует | Структура TCalibrates |
| GetBaseChannels | отсутствует | Массив из TChannels |
| GetStartups | отсутствует | Структура TStartups |
| SetCommon | Структура TCommonSettings | Обновленная структура TCommonSettings |
| SetNet | Структура TNetSettings | Обновленная структура TNetSettings |
| SetWork | Структура TWorkSettings | Обновленная структура TWorkSettings |
| SetRS485 | Структура TRS485Settings | Обновленная структура TRS485Settings |
| SetFromFile | Массив 512 байт из структуры TSettings и структуры TCalibrates (со смещением 256 байт) | Копия принятого массива 512 байт |
| Stop | отсутствует | отсутствует |
| SetCalibrates | Массив 256 байт | Копия принятого массива 256 байт |
| SetTime | Структура TRtc | Копия принятой структуры TRtc |
| ReadBlock | Первые 4 байта номер блока (INT);  Следующие 4 байта признак чтения двух блоков подряд («0» – читаем 1 блок, «1» – 2 блока) | Первые 8 байт – без изменений.  Затем 512 либо 1024 байт (1 либо 2 прочитанных блока) |
| WriteBlock | Первые 4 байта номер блока (INT);  Следующие 4 байта – константа для записи в блок | Копия принятых 8 байт |
| ResetSD | отсутствует | отсутствует |
| ResetEEPROM | отсутствует | отсутствует |
| ResetStartups | отсутствует | отсутствует |
| GetData | 4 байта – номер запрашиваемого фрагмента текущего кадра данных | Для АМАК  фрагмент текущего кадра: 1400 байт у всех фрагментов кроме последнего; 604 байт у последнего. Общая длина 41204 байт, всего 30 фрагментов |

1. Структура калибровок TCalibrates для АМАК:

#pragma pack (push, 1)

typedef struct {

char RegParams; //старшие 2 бита – шаг оцифровки (0..3 – 4, 8, 16, 32 мкс); следующие 4 бита – КУ (0..8 – 10, 20, 50, 100, 200, 500, 1000, 1500, 2000); младшие 2 бита – фильтр (0..2 – до 5, до 10, до 20 кГц)

char ChargeTime; //ограничение времени зарядки буферной емкости (одна дискрета соответствует 2 мс; оптимально значение «100» – то есть 200 мс)

char DelayTime; //задержка регистрации (одна дискрета соответствует 4 мкс)

char TransmitMode; //режим излучателя – 0x10 ВЧ-монополь, 0x20 СЧ-квадруполь; 0xCy – НЧ-блок, младшая тетрада «y» задает маску срабатывания 4-х линеек НЧ-блока – то есть можно получить диполь, монополь, квадруполь и т.п.

char TransmitPolarity; //имеет смысл только для TransmitMode=0xCy; младшая тетрада задает маску полярности линеек. 0 означает +, 1 означает -

char Reserved; //для выравнивания до 6 байт

} TWorkMode; //параметры одного режима измерений

typedef struct {

//common 52 bytes

int Mark; //EEPROM mark

char RtcIndex; //какие часы использовать для State

char TemperatureIndex; //какую температуру использовать -||-

short int AxelAxis[3]; //соответствие осей акселерометра.

//Пример: AxelAxis[0]=0x81 означает, что ось X прибора (0)

//противоположна оси Y акселерометра

//AxelAxis[1]=0x02 означает, что ось Y прибора соответствует оси Z акселерометра

float AxelLinear[2]; //коэффициенты для пересчета показаний акселерометра в mG

float TemperatureRtcLinear[2]; //пересчет температуры Rtc в градусы

float TemperatureAxelLinear[2]; //пересчет температуры Axel в градусы

float TemperatureInternalLinear[2]; //то же самое для внутреннего сенсора в градусы

float VoltageLinear[2]; //то же самое для напряжения

//amak 68 bytes

short int ModeCount; //число режимов работы

TWorkMode Modes[CUSTOM\_MAXMODECOUNT]; //описание режимов работы

char ModeEthernet; //задает номер режима работы, кадры для которого будут передаваться по Ethernet

char Service; //значение f в старшой тетраде означает, что код программы для работы с антенной будет выполняться; f в младшей тетраде – то же самое для блока излучателей. Использовать значение для работы 0xff

char BlockMask; //маска подключенных блоков приемников. Для работы использовать значение 0xff

char CompOffset; //флаг компенсации ненулевого среднего волновых картин (только для Ramtools)

char ReadRate; //скорость чтения данных из антенны по внутреннему RS485: 0 – 3 Мбит/с; 1 – 4 Мбит/с. Для работы использовать значение 1 (4 Мбит/с)

char ApplyShift; //применять коэффициенты сдвига к волновым картинам (только для Ramtools)

char ApplyMult; //применять мультипликативные коэффициенты к волновым картинам (только для Ramtools)

} TCalibrates; //полный объем - 52+69=121 байт

#pragma pack (pop)

1. Структура данных кадра

#define CUSTOM\_WAVELEN 640

#define CUSTOM\_TIMESTAT\_COUNT 3

#define CUSTOM\_TSMSTAT\_COUNT 5

#define CUSTOM\_RSTSTAT\_COUNT 8

#define CUSTOM\_CFGSTAT\_COUNT 16

#define CUSTOM\_READSTAT\_COUNT 16

#pragma pack (push, 1)

typedef struct {

int Mark; //SD work mark = 0x00000246

TRtc Rtc[2]; //два значения RTC – от внешней микросхемы и от встроенных часов STM32

short int Axel[3]; //3 оси акселерометра в кодах

short int Temperature[3]; //3 температуры по датчикам внешней микросхемы RTC, по микросхеме акселерометра и по встроенному сенсору STM32

unsigned short int Voltage; //напряжение в кодах

float Incline; //зенитный угол в градусах

float Sight; //визир в градусах

TWorkMode Mode; //режим измерения для этого кадра данных

char Block1[CUSTOM\_WAVELEN\*4\*2]; //данные волновых картин от одного блока в запакованном виде

char Block2[CUSTOM\_WAVELEN\*4\*2];

char Block3[CUSTOM\_WAVELEN\*4\*2];

char Block4[CUSTOM\_WAVELEN\*4\*2];

char Block5[CUSTOM\_WAVELEN\*4\*2];

char Block6[CUSTOM\_WAVELEN\*4\*2];

char Block7[CUSTOM\_WAVELEN\*4\*2];

char Block8[CUSTOM\_WAVELEN\*4\*2];

int TimeStats[CUSTOM\_TIMESTAT\_COUNT]; //статистика по времени выполнения операций

int RstStats[CUSTOM\_RSTSTAT\_COUNT]; //статистика по перезапускам гермоблоков

int TsmStats[CUSTOM\_TSMSTAT\_COUNT]; //статистика по ошибкам команд блоку излучателей

int CfgStats[CUSTOM\_CFGSTAT\_COUNT]; //статистика по ошибкам команд конфигурирования гермоблоков

int ReadStats[CUSTOM\_READSTAT\_COUNT]; //статистика по ошибкам команд чтения гермоблоков

int CRC32; //полином 0x4C11DB7, считается по всей структуре TBaseDate без учета поля CRC32, то есть «sizeof(TBaseData)-4»

} TBaseData;

#pragma pack (pop)

1. Функция crc32 (для простоты длина массива АМАК кратна 4 байтам)

int Crc32(int \*Data, int Length)

{

int Init=0xFFFFFFFF;

int RetVal;

for (int i=0;i<Length;i++)

{

RetVal=Init ^ Data[i];

for(int i=0;i<32;i++)

{

if(RetVal & 0x80000000)

{

RetVal=(RetVal<<1) ^ 0x4C11DB7;

}

else

{

RetVal<<=1;

}

}

Init=RetVal;

}

return RetVal;

}