RIFTEK
Sensors & Instruments

RF627 SDK И СЕРВИСНЫЙ ПРОТОКОЛ УПРАВЛЕНИЯ СКАНЕРОМ

Руководство программиста

Логойский тракт, 22, г. Минск
220090, Республика Беларусь
тел/факс: +375 17 281 36 57
info@riftek.com
www.riftek.com

Содержание

1. Назначение

Данное руководство создано в помощь разработчикам и содержит детальное описание библиотеки RF627 SDK и сервисного протокола управления сканером.

RF627 SDK и сервисный протокол позволяют создавать собственные программные приложения для работы с лазерными сканерами серии РФ627 производства компании РИФТЭК.

2. RF627 SDK

2.1. Системные требования

- Операционная система Windows 7 или выше.
- Microsoft Visual C++ Runtime Redistributable для Windows x64/x86.

2.2. Компоненты RF627 SDK

RF627 SDK включает в себя:

Каталог	Описание
doc	Руководство программиста.
include	Набор заголовочных файлов.
x64	LIB и DLL файлы для x64 и x86.
x86	

2.3. Имя класса

Имя класса: **rf627**.

2.4. Функции RF627 SDK

2.4.1. Инициализация библиотеки

```
static bool init();
```
Инициализация библиотеки. Вызвать один раз перед использованием библиотеки.
Возвращает true при успешном завершении.

```
static void cleanup();
```
Вызвать в конце программы для очистки памяти, выделенной библиотекой.

2.4.2. Поиск сканеров в сети

```
static rf627_list search(bool *ok = nullptr);
```
Выполняет поиск сканеров в сети. Возвращает список найденных сканеров.
`ok`: опциональный указатель на boolean, устанавливается в false в случае ошибки.

2.4.3. Подключение / отключение сканера

```
bool connect();
```
Выполняет подключение к сканеру по UDP. Данный метод создает и настраивает UDP сокеты для обмена данными со сканером. Возвращает false в случае ошибки.

```
void disconnect();
```
Выполняет отключение от сканера. Данный метод закрывает сокеты, открытые методом connect().

2.4.4. Управление параметрами сканера

2.4.4.1. Чтение / запись конфигурационных параметров

```
bool read_params(rf627_user_params_t* pparams = nullptr);
```
Чтение конфигурационных параметров из оперативной памяти сканера. Возвращает false в случае ошибки.
`pparams`: указатель на структуру конфигурационных параметров для хранения.

```
bool write_params(rf627_user_params_t* pparams = nullptr);
```
Запись конфигурационных параметров в оперативную память сканера. Возвращает false в случае ошибки.
`pparams`: указатель на структуру конфигурационных параметров для записи.

2.4.4.2. Сохранение параметров во флеш-память

```
bool flush_params();
```
Сохранение конфигурационных параметров во флеш-память сканера. Возвращает false в случае ошибки.

2.4.4.3. Восстановление заводских параметров

```
bool reset_params();
```
Восстановление конфигурационных параметров до заводских настроек. Возвращает false в случае ошибки.

2.4.4.4. Чтение системных параметров

```
bool read_sysmon_params(rf627_sysmon_params_t* psysmon = nullptr);
```
Чтение системных параметров из памяти сканера. Системные параметры также считываются методом read_params как часть структуры ref627_user_params. Возвращает false в случае ошибки.
`psysmon`: указатель на структуру sysmon.

2.4.5. Получение профилей

```
bool get_result(rf627_profile& profile);
```
Чтение данных профиля из потока. Возвращает false в случае ошибки.
`profile`: ссылка на структуру профиля.

2.4.6. Получение изображения

```
bool get_image(uint8_t* ppixmap);
```
Запрос и чтение изображения со сканера. Возвращает false в случае ошибки.
`ppixmap`: указатель на 8-разрядный пиксельный массив. Размер массива - RF627_IMAGE_SIZE. Каждый байт представляет яркость пикселей в диапазоне 0-255.

2.4.7. Обновление прошивки

```
bool write_firmware_image(const char* file_name);
```
Запись образа прошивки в память. Возвращает false в случае ошибки.
`file_name`: имя файла прошивки (.rf627).

```
bool flush_firmware_image();
```
Сохранение образа прошивки, переданного методом write_firmware_image, во флеш-память. В случае успешного сохранения, сканер будет перезагружен. Возвращает false в случае ошибки.

2.4.8. Перезагрузка сканера

```
bool reboot();
```
Перезагрузить сканер. Возвращает false в случае ошибки.

2.4.9. Журнал

```
bool read_log(uint32_t nstart_line, rf627_log_record_t *plog_entries, int
nlines);
```
Чтение записей журнала из памяти сканера. Возвращает false в случае ошибки.
nstart_line: номер первой строки для чтения.
plog_entries: указатель на массив структур rf627_log_record_t.
nlines: количество строк для чтения, максимальное количество - RF627_MAX_LOG_ENTRIES_PER_PAYLOAD.

```
uint32_t read_log_record_count();
```
Получение общего количества записей журнала. Возвращает количество записей или 0 при ошибке.

2.4.10. Сообщения об ошибках

```
const char *error_msg();
```
Получение указателя на последнее сообщение об ошибке.

2.4.11. Версия SDK

```
static uint32_t version();
```
Возвращает версию SDK.

2.5. Структуры данных

```
struct rf627_point
{
    double x;
    double z;
};
Координаты точки профиля (мм)

struct rf627_profile
{
    rf627_stream_msg_t header;
    std::vector<rf627_point> points;
};
Профиль, считанный из потока

typedef enum: uint8_t
{
    DTY_PixelsNormal        = 0x10,    // Пиксели (до 648 точек)
    DTY_ProfileNormal       = 0x11,    // Профиль (до 648 точек
    DTY_PixelsInterpolated  = 0x12,    // 2х интерполированные пиксели (до
1296)
    DTY_ProfileInterpolated = 0x13     // 2х интерполированный профиль (до
1296)
} data_type_t;
Тип профиля

#pragma pack(push,1)
typedef struct
{
    data_type_t data_type;     // Тип профиля (один из data_type_t enum)
```

```
    uint8_t     flags;          // Флаги (бит 7 указывает, что пакет требует
подтверждения от хоста)
    uint16_t    device_type;    // Тип устройства - 627
    uint32_t    serial_number;
    uint64_t    system_time;

    uint8_t     proto_version_major;
    uint8_t     proto_version_minor;
    uint8_t     hardware_params_offset;
    uint8_t     data_offset;
    uint32_t    packet_count;  // Порядковый номер отправленного пакета
    uint32_t    measure_count; // Порядковый номер измерения

    uint16_t    zmr;            // Диапазон измерения по Z
    uint16_t    xemr;           // Диапазон по X в конце диапазона Z
    uint16_t    discrete_value;
    uint8_t     reserved_0[14];

    uint32_t    exposure_time; // Время экспозиции
    uint32_t    laser_value;
    uint32_t    step_count;    // Значение STEP в режиме STEP/DIR
    uint8_t     dir;           // Значение DIR
    uint8_t     reserved_1[3];
}
rf627_stream_msg_t;
#pragma pack(pop)
Заголовок пакета с профилем

#pragma pack(push,1)
typedef struct
{
    char        name[64];       // Читаемое имя сканера
    uint16_t    device_id;      // 627
    uint32_t    serial_number;
    uint32_t    firmware_version;
    uint32_t    hardware_version;
    uint32_t    config_version;
    uint32_t    fsbl_version;
    uint32_t    z_begin;        // Начало диапазона Z
    uint32_t    z_range;        // Диапазон измерения по Z
    uint32_t    x_smr;          // Диапазон измерения по X в начале диапазона Z
    uint32_t    x_emr;          // Диапазон измерения по X в конце диапазона Z
    uint8_t     reserved_0[36];

    uint16_t    eth_speed;      // 100 или 1000 (Мбит/с)
    uint32_t    ip_address;
    uint32_t    net_mask;
    uint32_t    gateway_ip;
    uint32_t    host_ip;        // IP адрес хоста (устройства, принимающего
профили)
    uint16_t    stream_port;    // Порт хоста
    uint16_t    http_port;
    uint16_t    service_port;
    uint16_t    eip_broadcast_port;
    uint16_t    eip_port;
    uint8_t     hardware_address[6];
    uint8_t     reserved_1[26];

    uint32_t    max_payload_size;
    uint8_t     reserved_2[32];
```

```
    uint8_t     stream_enabled;         // Ненулевой, если поток данных включен
    uint8_t     stream_format;          // Тип профиля (data_type_t & 0x0F)
    uint8_t     reserved_3[32];

    uint8_t     reserved_4[256];
}
rf627_device_info_t;
#pragma pack(pop)
Информация о сканере

#pragma pack(push,1)
typedef struct
{
    char        name[64];       // Читаемое имя сканера
    uint8_t             reserved[128];
}
rf627_general_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    int16_t             fpga_temp;      // Температура FPGA (Цельсий * 10)
    uint8_t             reserved[80];
}rf627_sysmon_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint8_t             dhs;            // Включить режим удвоенной скорости
    uint8_t             gain_analog;
    uint8_t             gain_digital;
    uint32_t    exposure;               // Время экспозиции, нс
    uint32_t    max_exposure;
    uint32_t    frame_rate;             // Ограничение частоты кадров
    uint32_t    max_frame_rate;
    uint8_t     reserved_0;
    uint8_t     auto_exposure;          // Включить автоэкспозицию
    uint8_t     frame_by_request;
    uint8_t             reserved_1[61];
}rf627_sensor_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint8_t             enable;         // Переключить сканер в режим эмуляции
РФ625
    uint16_t    tcp_port;
    uint8_t             reserved[32];
}rf627_rf625compat_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint8_t             enable;         // Включить режим ROI
    uint8_t             active;
```

```
    uint16_t    window_height;
    uint8_t             position_mode; // 0 - ручной, nonzero - автоматический
    uint16_t    window_top;
    uint16_t    current_window_top;    // Текущая вершина ROI в автоматическом
режиме
    uint16_t    profile_size;          // Требуемый размер профиля
    uint8_t             reserved[80];
}rf627_roi_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint16_t    speed;                          // 100 или 1000 (Мбит/с)
    uint8_t             autonegotiation;        // Автосогласование скорости
подключения
    uint32_t    ip_address;
    uint32_t    net_mask;
    uint32_t    gateway_ip;
    uint32_t    host_ip;
    uint16_t    stream_port;
    uint16_t    http_port;
    uint16_t    service_port;
    uint16_t    eip_broadcast_port;
    uint16_t    eip_port;
    uint8_t             reserved[64];
}rf627_network_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint8_t             enable;         // Включить поток данных
    uint8_t             format;         // Тип профиля (data_type_t & 0x0F)
    uint8_t             ack;            // Для каждого пакета данных требуется
подтверждение (0 - выключено, по умолчанию)
    uint8_t             reserved[32];
}rf627_stream_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint32_t    brightness_threshold;
    uint8_t             stg1_filter_width;
    uint8_t             stg1_processing_mode;
    uint8_t             stg2_reduce_noise;
    uint32_t    frame_rate;
    uint8_t             reserved[60];
}rf627_image_processing_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint8_t             enable;         // 0 - выключить лазер
    uint8_t             auto_level;
    uint16_t    level;
    uint8_t             reserved[32];
}rf627_laser_params_t;
```

```
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint16_t    params_mask;          // Каждый бит указывает, настраиваются
ли параметры в выбранном режиме (LSB - in1_enable и т. д.).
    uint8_t             in1_enable;
    uint8_t             in1_mode;
    uint32_t    in1_delay;
    uint8_t             in1_decimation;
    uint8_t             in2_enable;
    uint8_t             in2_mode;
    uint8_t             in2_invert;
    uint8_t             in3_enable;
    uint8_t             in3_mode;
    uint8_t             reserved[12];
}rf627_inputs_preset_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint8_t                 preset_index;     // Выбранный режим
    rf627_inputs_preset_t   params[12];
    uint8_t                 reserved[32];
}rf627_inputs_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    uint8_t             out1_enable;
    uint8_t             out1_mode;
    uint32_t    out1_delay;
    uint32_t    out1_pulse_width;
    uint8_t             out1_invert;
    uint8_t             out2_enable;
    uint8_t             out2_mode;
    uint32_t    out2_delay;
    uint32_t    out2_pulse_width;
    uint8_t             out2_invert;
    uint8_t             reserved[32];
}rf627_outputs_params_t;
#pragma pack(pop)

#pragma pack(push,1)
typedef struct
{
    rf627_general_params_t          general;
    rf627_sysmon_params_t           sysmon;
    rf627_rf625compat_params_t      rf625_compat;
    rf627_sensor_params_t           sensor;
    rf627_roi_params_t              roi;
    rf627_network_params_t          network;
    rf627_stream_params_t           stream;
    rf627_image_processing_params_t image_processing;
    rf627_laser_params_t            laser;
    rf627_inputs_params_t           inputs;
    rf627_outputs_params_t          outputs;
```

```
    uint8_t                         reserved[283];
}rf627_user_params_t;
#pragma pack(pop)

Режимы запуска измерений:
0 - Internal Clock
1 - External Trigger
2 - 1-phase Encoder
3 - 1-phase Encoder w/Zero
4 - 2-phase Encoder
5 - 2-phase Encoder w/Zero
6 - Step/Dir
7 - Ext. Trigger/Int. Clock
8 - Software Request
```


3. Сервисный протокол управления сканером

3.1. Формат передачи сканером профилей

Результатом обработки каждого кадра изображения, поступающего от CMOS-сенсора является профиль отраженного от контролируемой поверхности луча лазера, представленный в одном из четырех форматов. Каждый профиль передается сканером в виде отдельного UDP-пакета на сетевой адрес и порт, указанные на web-странице сканера на вкладке **Network**.

Область пользовательских данных (полезная нагрузка) UDP-пакета содержит данные от сканера, разделенные на заголовок и данные профиля.

Функционально поля заголовка объединены в группы:

- **Message fingerprint** - поля данной группы формируют уникальный идентификатор сообщения, используемый для подтверждения доставки UDP-пакета.
- **Application specific data** - специфические для конкретной области применения сканера данные, предназначены для размещения информации по требованию заказчика сканера специального исполнения (например, дополнительные счетчики, состояние входов в момент получения профиля, параметры региона интереса и т.д.).
- **Hardware parameters** - параметры аппаратных модулей сканера в момент начала экспонирования кадра.

Заголовок имеет следующую структуру:

Смещение	Тип	Имя поля	Комментарий	Группа
0	uint8_t	Data_type	Маркер формата профиля. Указывает, в каком формате представлены результаты обработки, и имеет следующие значения: 0x10 - «raw profile»; 0x11 - «calibrated profile»; 0x12 - «extended raw profile»; 0x13 - «extended calibrated profile».	Message fingerprint
1	uint8_t	Flags	Битовые флаги различного назначения: Flags[7] – требование подтверждения доставки пакета; Flags[6:0] – резерв.	
2	uint16_t	Device_ID	Идентификатор типа устройства, всегда 627.	
4	uint32_t	Serial_number	Серийный номер сканера.	
8	uint64_t	System_time	Системное время в наносекундах (время после включения сканера) начала экспозиции кадра,	

Смещение	Тип	Имя поля	Комментарий	Группа
			по которому рассчитан профиль.	
16	uint8_t	Protocol_Version_Major	Текущая версия протокола.	
17	uint8_t	Protocol_Version_Minor	Текущая версия ревизии протокола.	
18	uint8_t	Hardware_params_shift	Смещение от начала заголовка в байтах к области аппаратных параметров. Используется в случае изменения размера области «Application specific data».	
19	uint8_t	Data_shift	Смещение от начала заголовка в байтах к области данных. Используется в случае изменения размера области «Hardware parameters».	
20	uint32_t	Software_packets_counter	Счетчик отправленных пакетов с профилями.	
24	uint32_t	Measures_counter	Аппаратный счетчик выполненных измерений.	
28	uint16_t	ZMR	Диапазон измерения по оси Z в 0,1*мм.	
30	uint16_t	XEMR	Диапазон измерения по оси X в конце диапазона в 0,1*мм.	
32	uint16_t	Discrete_Value	Значение дискреты, на которую разделен диапазон измерения сканера.	
34	uint16_t	Reserved		Application specific data
36	uint16_t	Reserved		
38	uint16_t	Reserved		
40	uint16_t	Reserved		
42	uint16_t	Reserved		
44	uint16_t	Reserved		
46	uint16_t	Reserved		
48	uint32_t	Exposure_time	Значение времени накопления CMOS-сенсором отраженного излучения (время экспозиции) в наносекундах.	Hardware parameters
52	uint32_t	Laser_value	Значение времени включения лазера во время экспозиции кадра в наносекундах.	
56	uint32_t	Step_counter	Значение счетчика импульсов, поступивших на Вход №1.	
60	uint8_t	Dir	Направление перемещения, определяемое по Входу №2, зафиксированное при старте накопления кадра CMOS-сенсором.	
61	uint8_t[3]	Reserved		

Данные профиля в зависимости от формата имеют следующую структуру.
Формат **Raw profile**:

Смещение	Тип данных	Имя поля	Комментарий
64	uint16_t	Z[0]	Значение Z (в дискретах) точки 1.
66	uint16_t	Z[1]	Значение Z (в дискретах) точки 2.
68	uint16_t	Z[2]	Значение Z (в дискретах) точки 3.
70	uint16_t	Z[3]	Значение Z (в дискретах) точки 4.
...	...	...	...
1358	uint16_t	Z[647]	Значение Z (в дискретах) точки 648.

Формат **Calibrated profile**:

Смещение	Тип данных	Имя поля	Комментарий
64	uint16_t	Z[0]	Значение Z (в дискретах) точки 1.
66	uint16_t	Z[1]	Значение Z (в дискретах) точки 2.
68	uint16_t	Z[2]	Значение Z (в дискретах) точки 3.
70	uint16_t	Z[3]	Значение Z (в дискретах) точки 4.
...	...	...	...
2654	uint16_t	Z[1295]	Значение Z (в дискретах) точки 1296.

Формат **Extended raw profile**:

Смещение	Тип данных	Имя поля	Комментарий
64	int16_t	X[0]	Значение X (в дискретах) точки 1.
66	uint16_t	Z[0]	Значение Z (в дискретах) точки 1.
68	int16_t	X[1]	Значение X (в дискретах) точки 2.
70	uint16_t	Z[1]	Значение Z (в дискретах) точки 2.
...	...	...	...
2652	int16_t	X[647]	Значение X (в дискретах) точки 648.
2654	uint16_t	Z[647]	Значение Z (в дискретах) точки 648.

Формат **Extended calibrated profile**:

Смещение	Тип данных	Имя поля	Комментарий
64	int16_t	X[0]	Значение X (в дискретах) точки 1.
66	uint16_t	Z[0]	Значение Z (в дискретах) точки 1.
68	int16_t	X[1]	Значение X (в дискретах) точки 2.
70	uint16_t	Z[1]	Значение Z (в дискретах) точки 2.
...	...	...	...
5180	int16_t	X[1295]	Значение X (в дискретах) точки 1296.
5182	uint16_t	Z[1295]	Значение Z (в дискретах) точки 1296.

Поскольку тип передаваемых данных целочисленный (uint16_t, int16_t), для преобразования положения точек профиля в вещественный формат (float) необходимо использовать следующие соотношения:

- для форматов профиля **Raw profile** и **Extended raw profile** в субпиксельные значения:

PointZ[N] = Z[N] / Discrete_Value;
PointX[N] = N;

- для форматов профиля **Calibrated profile** и **Extended calibrated profile** в миллиметрах:

*PointZ[N] = Z[N] * ZMR / Discrete_Value;*
*PointX[N] = X[N] * XEMR / Discrete_Value.*

Подтверждение доставки пакета с профилем обеспечивает гарантированную доставку каждого выполненного измерения потребителю. В качестве подтверждения необходимо отослать UDP-пакет на сетевой адрес сканера, порт такой же, как порт хоста для приема данных (при заводских настройках 50001). Пакет должен содержать копию первых 16 байтов заголовка пакета с профилем (поля группы **Message fingerprint**).

3.2. Типы сообщений

Конфигурирование и управление параметрами сканера возможно не только с помощью встроенной WEB-страницы, но и с использованием сервисного протокола управления, основанного на обмене UDP-пакетами (называемыми сообщениями) между сканером и управляющим компьютером (или другим сетевым устройством).

Сообщения разделены на три типа:

- **Команда** - требует от устройства, которому направлена, какого-либо действия, установки параметра(ов) или возврата значения параметра(ов), может требовать подтверждения и уникально идентифицируется.
- **Подтверждение команды** - подтверждает прием команды, содержит уникальный идентификатор команды и если действия не требуют длительной подготовки результата (чтение FLASH-памяти, захват кадра и т.д.), то в теле подтверждения могут содержаться запрошенные данные.
- **Ответ на команду** - содержит запрошенные командой данные. Ответ на команду высылается по мере готовности данных и содержит собственный уникальный идентификатор, может требовать подтверждения.

Сообщения в виде UDP-пакетов необходимо отправлять на сетевой адрес сканера (для заводских настроек 192.168.1.30) и порт, указанный в поле **Service port** вкладки **Network** WEB-страницы (для заводских настроек 50011). Сообщения с подтверждением и ответами сканер высылает на сетевой адрес и порт, с которого получена соответствующая команда.

Типы сообщений определены следующим образом:

```
/*=============== ТИПЫ СООБЩЕНИЙ ==========================================*/
#define              OP_CODE_COMMAND            (uint8_t)0x01
#define              OP_CODE_CONFIRM            (uint8_t)0x02
#define              OP_CODE_ANSWER             (uint8_t)0x03
#define              OP_FLAGS_CONFIRM_BIT       (uint8_t)0x08
#define              OP_FLAGS_FINAL_BIT         (uint8_t)0x04

typedef      uint8_t                            SrvcMsgOperation_Type;
enum
{
      MSG_COMMAND                = (OP_CODE_COMMAND << 4),
      MSG_CONFIRM                = (OP_CODE_CONFIRM << 4),
      MSG_ANSWER                 = (OP_CODE_ANSWER << 4),
      MSG_COMMAND_CNFRM_FINAL    = (OP_CODE_COMMAND << 4) | OP_FLAGS_CONFIRM_BIT |
OP_FLAGS_FINAL_BIT,
      MSG_COMMAND_CNFRM          = (OP_CODE_COMMAND << 4) | OP_FLAGS_CONFIRM_BIT,
      MSG_COMMAND_FINAL          = (OP_CODE_COMMAND << 4) | OP_FLAGS_FINAL_BIT,
      MSG_CONFIRM_FINAL          = (OP_CODE_CONFIRM << 4) | OP_FLAGS_FINAL_BIT,
      MSG_ANSWER_CNFRM           = (OP_CODE_ANSWER << 4) | OP_FLAGS_CONFIRM_BIT,
      MSG_ANSWER_FINAL           = (OP_CODE_ANSWER << 4) | OP_FLAGS_FINAL_BIT,
      MSG_ANSWER_CNFRM_FINAL     = (OP_CODE_ANSWER << 4) | OP_FLAGS_CONFIRM_BIT |
OP_FLAGS_FINAL_BIT
};
/*==========================================================================*/
```

Константы OP_CODE_COMMAND, OP_CODE_CONFIRM, OP_CODE_ANSWER, OP_FLAGS_CONFIRM_BIT, OP_FLAGS_FINAL_BIT следует использовать для проверки соответствующих битов поля **Тип сообщения** и не должны записываться в это поле.

Для установки типа сообщения должны использоваться следующие константы:

Константа	Описание
MSG_COMMAND	Команда, не требующая подтверждения и не являющаяся последней в цепочке команд.
MSG_CONFIRM	Подтверждение команды, не являющееся последним в цепочке.
MSG_ANSWER	Ответ на команду. Не требует подтверждения и не является последним в цепочке.
MSG_COMMAND_CNFRM_FINAL	Команда, требующая подтверждение доставки и являющаяся последней в цепочке команд.
MSG_COMMAND_CNFRM	Команда, требующая подтверждение доставки и не являющаяся последней в цепочке команд.
MSG_COMMAND_FINAL	Команда, не требующая подтверждения и являющаяся последней в цепочке команд.
MSG_CONFIRM_FINAL	Подтверждение команды, являющееся последним в цепочке.
MSG_ANSWER_CNFRM	Ответ на команду, требующий подтверждения и не являющийся последним в цепочке.
MSG_ANSWER_FINAL	Ответ на команду, не требующий подтверждения и являющийся последним в цепочке ответов.
MSG_ANSWER_CNFRM_FINAL	Ответ на команду, требующий подтверждения и являющийся последним в цепочке.

3.3. Структура сообщения

Обобщенная структура сообщения:

Индекс	Функция	Размер, байт	Описание
0	Тип сообщения	1	Биты 7-4: b0001 - команда; b0010 - подтверждение команды; b0011 - ответ на команду; Бит 3 - требование подтверждения ответа или команды: b0 - подтверждение не требуется; b1 - подтверждение требуется; Бит 2 - признак последней команды или ответа в цепочке: b0 - команда/ответ не последний; b1 - команда/ответ последний в цепочке; Остальное (биты 1 и 0) - резерв.
1	Параметры сообщения	3	Команды: не используется. Подтверждения, ответы: байт[0] содержит результат выполнения команды (0 - успешно, иное - ошибка).
4	Идентификатор устройства	4	Уникальный идентификатор устройства (серийный номер сканера), которому отправлено сообщение. Может быть широковещательным: 0xFFFFFFFF. При отправке сообщения или подтверждения сканером, содержит идентификатор сканера.
8	Уникальный идентификатор сообщения (для подтверждения)	2	Счетчик, должен быть уникальным от сообщения к сообщению, период повтора 65536 сообщений вполне достаточен для работы.
10	Идентификатор команды	1	Идентификатор модуля: {

Индекс	Функция	Размер, байт	Описание
			MID_SYSTEM = 0x50, MID_USER_PARAMS = 0x5E, MID_FRAME_CAPTURE = 0x53 }
11		1	Код команды - специфичен для каждого модуля.
12	Размер области данных	2	Размер данных в полезной нагрузке.
14-(N+14)	Атрибуты команды или данные ответа.	N	В случае отправки команды, поле содержит атрибуты (данные, которые необходимо применить). В случае подтверждения или ответа – затребованные командой данные.

При разработке пользовательского ПО, структура сообщения может быть представлена в следующем виде:

```
/*=============== СТРУКТУРА-ЗАГОЛОВОК СООБЩЕНИЯ ==============================*/
#pragma pack(push, 1)
volatile typedef struct
{
	union{
		struct{
			uint8_t		OpFlags	: 4;
			uint8_t		OpCode	: 4;
		};
		SrvcMsgOperation_Type	Operation;
	};
	uint8_t				MsgParams[3];
	union{
		struct{
			uint32_t	DevID;
			uint16_t	UniqID;
			ModuleID_Type	ModuleID;
			uint8_t		Cmd;
		};
		uint64_t		MsgID;
	};
	uint16_t			PayloadLen;
}SrvcMsgHeader_Type;
/*============================================================================*/
/*=============== СТРУКТУРА СООБЩЕНИЯ ========================================*/
#define		MSG_MAX_PAYLOAD	32768 - sizeof(SrvcMsgHeader_Type)
typedef struct
{
	union{
		struct{
			union{
				SrvcMsgHeader_Type	Header;
				uint8_t			HeaderData[sizeof(SrvcMsgHeader_Type)];
			};
			uint8_t			Payload[MSG_MAX_PAYLOAD];
		};
		uint8_t		RawData[MSG_MAX_PAYLOAD+sizeof(SrvcMsgHeader_Type)];
	};
}SrvcMsg_Type;
/*============================================================================*/
```

Поле **Идентификатор команды** содержит идентификатор модуля, которому адресована команда и непосредственно код команды, это позволяет функционально разделить команды на группы и в целом иметь до 65536 команд.

Пользователю доступны следующие модули:

- **SYSTEM** - идентификатор 0x50, системный модуль, обеспечивающий управление глобальными параметрами сканера (получением и записью всех настроек одним пакетом, сохранение параметров, перезагрузка и т.д.).
- **USER_PARAMS** - идентификатор 0x5E, модуль управления пользовательскими параметрами, предназначен для чтения и записи групп параметров и отдельных параметров сканера.
- **FRAME_CAPTURE** - идентификатор 0x53, модуль захвата изображения, формируемого CMOS-сенсором, обеспечивает получение кадра.

3.3.1. Модуль SYSTEM

Модуль **SYSTEM** имеет следующие команды:

Название команды:	CMD_GET_USER_PARAMS
Код команды:	0x02
Описание:	Получение структуры с пользовательскими параметрами сканера.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { .... }UserParams_TypeDef. **Примечание:** описание полей структуры UserParams_TypeDef раскрыто в пар. 3.4.
Название команды:	CMD_SET_USER_PARAMS
Код команды:	0x03
Описание:	Запись пользовательских параметров в текущие параметры без сохранения в флеш-память.
Атрибуты команды:	typedef struct { .... }UserParams_TypeDef.
Ответ:	Пакет с подтверждением.
Название команды:	CMD_SAVE_PARAMS
Код команды:	0x10
Описание:	Сохранение текущей конфигурации сканера в флеш-память. Сохраняются все параметры.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением.
Название команды:	CMD_SAVE_AS_DEFAULT_PARAMS
Код команды:	0x11
Описание:	Сохранение текущей конфигурации сканера в область восстановления на флеш-памяти. Сохраняются все параметры. Данные параметры используются при повреждении основных.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением.
Название команды:	CMD_RESET
Код команды:	0x12
Описание:	Сброс сканера. При этом при загрузке будут использованы параметры, сохраненные в флеш-памяти.
Атрибуты команды:	Нет.

Ответ:	Пакет с подтверждением.
Название команды:	CMD_LOAD_DEFAULT_PARAMS
Код команды:	0x13
Описание:	Установка всех параметров из области восстановления и сохранение их во флеш-память.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением.

3.3.2. Модуль USER_PARAMS

Модуль **USER_PARAMS** имеет следующие команды:

Название команды:	CMD_U_GENERAL_HELLO
Код команды:	0x00
Описание:	Команда на передачу пакета с основными параметрами устройства, используется при поиске устройств в сети.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { .... }HelloAnswer_TypeDef. **Примечание:** описание полей структуры HelloAnswer_TypeDef раскрыто в п. 3.5.1..
Название команды:	CMD_U_GENERAL_GET
Код команды:	0x01
Описание:	Запрос общих пользовательских параметров.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_General_TypeDef;
Название команды:	CMD_U_GENERAL_SET
Код команды:	0x02
Описание:	Установка общих пользовательских параметров.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_General_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_SYSMONITOR_GET
Код команды:	0x03
Описание:	Запрос параметров и результатов измерений системным монитором.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными:

	typedef struct { … }User_SystemMonitor_TypeDef;
Название команды:	CMD_U_SYSMONITOR_SET
Код команды:	0x04
Описание:	Установка параметров системного монитора. Поля, содержащие результаты измерений, игнорируются.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_SystemMonitor_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_COMPATIBILITY_GET
Код команды:	0x05
Описание:	Запрос параметров режима совместимости.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_Compatibility_TypeDef;
Название команды:	CMD_U_COMPATIBILITY_SET
Код команды:	0x06
Описание:	Установка параметров режима совместимости.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_Compatibility_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_SENSOR_GET
Код команды:	0x07
Описание:	Запрос параметров CMOS-сенсора.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_Sensor_TypeDef;
Название команды:	CMD_U_SENSOR_SET
Код команды:	0x08
Описание:	Установка параметров CMOS-сенсора.
Атрибуты команды:	Данные в структуре: typedef struct

	{ … }User_Sensor_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_ROI_GET
Код команды:	0x09
Описание:	Запрос параметров режима ROI.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_ROI_TypeDef;
Название команды:	CMD_U_ROI_SET
Код команды:	0x0A
Описание:	Установка параметров режима ROI.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_ROI_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_NETWORK_GET
Код команды:	0x0B
Описание:	Запрос параметров сетевого подключения.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_Network_TypeDef;
Название команды:	CMD_U_NETWORK_SET
Код команды:	0x0C
Описание:	Установка параметров сетевого подключения.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_Network_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_STREAMS_GET
Код команды:	0x0D
Описание:	Запрос параметров потоков передачи данных.
Атрибуты команды:	Нет.

Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_Streams_TypeDef;
Название команды:	CMD_U_STREAMS_SET
Код команды:	0x0E
Описание:	Установка параметров потоков передачи данных.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_Streams_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_PROCESSING_GET
Код команды:	0x0F
Описание:	Запрос параметров обработки изображения.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_Processing_TypeDef;
Название команды:	CMD_U_PROCESSING_SET
Код команды:	0x10
Описание:	Установка параметров обработки изображения.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_Processing_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_LASER_GET
Код команды:	0x11
Описание:	Запрос параметров лазера.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_Laser_TypeDef;
Название команды:	CMD_U_LASER_SET
Код команды:	0x12
Описание:	Установка параметров лазера.
Атрибуты команды:	Данные в структуре: typedef struct

	{ … }User_Laser_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_INPUTS_GET
Код команды:	0x13
Описание:	Запрос параметров входных каналов.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_Inputs_TypeDef;
Название команды:	CMD_U_INPUTS_SET
Код команды:	0x14
Описание:	Установка параметров входных каналов.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_Inputs_TypeDef;
Ответ:	Пакет с подтверждением.
Название команды:	CMD_U_OUTPUTS_GET
Код команды:	0x15
Описание:	Запрос параметров выходных каналов.
Атрибуты команды:	Нет.
Ответ:	Пакет с подтверждением и данными: typedef struct { … }User_Outputs_TypeDef;
Название команды:	CMD_U_OUTPUTS_SET
Код команды:	0x16
Описание:	Установка параметров выходных каналов.
Атрибуты команды:	Данные в структуре: typedef struct { … }User_Outputs_TypeDef;
Ответ:	Пакет с подтверждением.

3.3.3. Модуль FRAME_CAPTURE

Модуль **FRAME_CAPTURE** имеет следующие команды:

Название команды:	CMD_U_FRAME_CAPTURE_GET_FRAME
Код команды:	0x10
Описание:	Запрос кадра. После получения команды, сканер захватит 1 кадр изображения и передаст его по фрагментам.
Атрибуты команды:	uint8_t: 0 – не требовать подтверждение доставки фрагментов; др – требовать подтверждение доставки фрагментов. Доставка последнего фрагмента всегда требует подтверждения.
Ответ:	Пакет с подтверждением. После захвата кадра передача его по фрагментам в формате: uint32_t: смещение данных в кадре; uint8_t[…]: данные о яркости точек.

3.4. Структура пользовательских параметров

Структура **UserParams_TypeDef** содержит все доступные пользователю параметры сканера и включает следующие поля (выравнивание данных внутри структуры – по байтам):

```
volatile typedef struct __attribute__ ((__packed__))
{
	User_General_TypeDef			General;
	User_SystemMonitor_TypeDef		SystemMonitor;
	User_Compatibility_TypeDef		Compatibility;
	User_Sensor_TypeDef			Sensor;
	User_ROI_TypeDef			ROI;
	User_Network_TypeDef			Network;
	User_Streams_TypeDef			Streams;
	User_Processing_TypeDef			Processing;
	User_Laser_TypeDef			Laser;
	User_Inputs_TypeDef			Inputs;
	User_Outputs_TypeDef			Outputs;
/*-------Резерв-----------------------------------------*/
	uint8_t					Reserved[283];
/*------------------------------------------------------*/
}UserParams_TypeDef;
```

Каждое поле структуры **UserParams_TypeDef** представляет собой экземпляр структуры, содержащий конкретные параметры одной из подсистем сканера.

3.4.1. Структура общих параметров

Структура общих параметров **User_General_TypeDef**:

```
typedef struct __attribute__ ((__packed__))
{
	uint8_t				Name[64];
	uint8_t				Reserved[128];
}User_General_TypeDef;
```

Название поля	Описание
Name[64]	Назначаемое пользователем имя сканера. Может использоваться для упрощения идентификации сканеров в сети предприятия.
Reserved[128]	Зарезервировано для новых параметров.

3.4.2. Структура параметров системного монитора

Структура параметров системного монитора **User_SystemMonitor_TypeDef** содержит параметры и результаты работы системного монитора, позволяющие оценивать текущее состояние сканера:

```
typedef struct __attribute__ ((__packed__))
{
	int16_t 			FPGA_Temp;
	uint8_t 			ParamsChanged;
	uint8_t 			Reserved[80];
}User_SystemMonitor_TypeDef;
```

Название поля	Описание
FPGA_Temp	Температура вычислительного модуля сканера в 10*C, т.е. если температура 51,2°C, то значение будет 512.
ParamsChanged	Параметры были изменены, но не сохранены: 0 – изменений не было; 1 – изменения были, они не сохранены.
Reserved[80]	Зарезервировано для новых параметров.

3.4.3. Структура параметров режима совместимости

Структура параметров режима совместимости **User_Compatibility_TypeDef** содержит параметры, используемые в режиме эмуляции других типов устройств (например, сканера РФ625):

```
typedef struct __attribute__ ((__packed__))
{
	uint8_t 			RF625_Enabled;
	uint16_t 			RF625TCPPort;
	uint8_t 			Reserved[32];
}User_Compatibility_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
RF625_Enabled	0	Разрешение работы в режиме эмуляции сканера РФ625. 0 – запрещено; др. – разрешено.
RF625TCPPort	620	Номер порта TCP соединения, используемого в режиме эмуляции сканера РФ625 для управления.
Reserved[32]		Зарезервировано для новых параметров.

3.4.4. Структура параметров CMOS-сенсора

Структура параметров CMOS-сенсора **User_Sensor_TypeDef** содержит параметры, влияющие на формирование кадра:

```
typedef struct __attribute__ ((__packed__))
{
	uint8_t		DoubleSpeedMode;
	uint8_t		Gain_Analog;
	uint8_t		Gain_Digital;
	uint32_t	Exposure;
	uint32_t	MaxExposure;
	uint32_t	FrameRate;
	uint32_t	MaxFrameRate;
	uint8_t		Reserved_0;
	uint8_t		AutoExposure;
	uint8_t		Reserved[62];
}User_Sensor_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
DoubleSpeedMode	0	Включение удвоенной частоты кадров и соответственно удвоенной частоты профилей. В данном режиме точность снижается с ±0,05% от диапазона до ±0,085% от диапазона. 0 – выключен; др. – включен.
Gain_Analog	6	Аналоговое усиление сигнала, формируемого каждым пикселем изображения. Допустимые значения: от 1 до 15.
Gain_Digital	108	Цифровое усиление сигнала, формируемого каждым пикселем изображения. Допустимые значения: от 96 до 114.
Exposure	300000	Время экспозиции CMOS-сенсором кадра (время накопления сигнала) в наносекундах, шаг 10 нс. Минимальное значение составляет 100 нс, максимальное определяется требуемой частотой поступления профилей.
MaxExposure	1443298	Максимально возможное в данном режиме работы сканера время накопления кадра в наносекундах.
FrameRate	485	Текущая частота кадров, формируемых CMOS-сенсором, равна текущей частоте расчёта профилей.
MaxFrameRate	485	Максимально возможная в данном режиме работы сканера (с учетом DoubleSpeedMode, параметров режима «Регион интереса» и др.) частота кадров и, соответственно, частота профилей.
Reserved_0		Резерв.
AutoExposure	0	Флаг, разрешающий автоматическое управление экспозицией: 0 – выключен; др. – включен.
Reserved[62]		Зарезервировано для новых параметров.

3.4.5. Структура параметров режима "Регион интереса"

Структура параметров режима "Регион интереса" **User_ROI_TypeDef** содержит параметры области кадра, используемой для получения профиля. Данные параметры существенно влияют на частоту профилей.

```
typedef struct __attribute__ ((__packed__))
{
	uint8_t		Enabled;
	uint8_t		Active;
	uint16_t	Size;
	uint8_t		PositionMode;
	uint16_t	FixedPosition;
	uint16_t	AutoPosition;
	uint16_t	RequiredProfileSize;
	uint8_t		Reserved[80];
}User_ROI_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
Enabled	0	Включение режима «Регион интереса». 0 – выключен; др. – включен.
Active	0	Флаг активности «Региона интереса». Если режим включен и выполняются условия устойчивого обнаружения профиля в заданной параметрами области, то значение будет «1», иначе – «0». Анализ данного флага позволяет оценить устойчивость работы режима «Регион интереса» при текущих настройках.
Size	64	Размер анализируемой области в линиях CMOS-сенсора. Допустимые значения: от 24 до 480 с шагом 8 линий.
PositionMode	0	Режим управления положением анализируемой области. 0 – ручное управление: положение фиксировано и задается оператором в настройках сканера; др. – автоматическое управление положением анализируемой области на основе анализа профиля.
FixedPosition	300	Положение анализируемой области в режиме ручного управления. Допустимые значения: от 0 до (488-Size).
AutoPosition	100	Текущее положение анализируемой области в режиме автоматического управления положением.
RequiredProfileSize	324	Количество точек в профиле, необходимых для нахождения в режиме «Регион интереса». Если количество точек в профиле менее данного количества, сканер перейдет в режим поиска профиля с анализом изображения от всего CMOS-сенсора. Допустимые значения: от 1 до 648 (до 1296 в режимах «extended…»).
Reserved[80]		Зарезервировано для новых параметров.

3.4.6. Структура параметров сетевого интерфейса

Структура параметров сетевого интерфейса **User_Network_TypeDef** содержит параметры сетевых настроек сканера:

```
typedef struct __attribute__ ((__packed__))
{
	uint16_t	Speed;
	uint8_t	Autonegotiation;
	uint8_t	IP[4];
	uint8_t	Mask[4];
	uint8_t	Gateway[4];
	uint8_t	Host_IP[4];
	uint16_t	Host_Data_Port;
	uint16_t	HTTP_Port;
	uint16_t	ServiceCtrlPort;
	uint16_t	EIP_BroadcastRcvPort;
	uint16_t	EIP_ListeningTCPPort;
	uint8_t	Reserved[64];
}User_Network_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
Speed	1000	Скорость соединения. В режиме автосогласования для изменения не доступно. Если режим автосогласования выключен, задает скорость соединения: 100 – 100 Мбит/с; 1000 – 1000 Мбит/с.
Autonegotiation	1	Включение и отключение режима автоматического согласования скорости сетевого соединения. 0 – выключен; др. – включен.
IP[4]	192.168.1.30	Сетевой адрес сканера.
Mask[4]	255.255.255.0	Маска подсети.
Gateway[4]	192.168.1.1	Сетевой адрес шлюза.
Host_IP[4]	192.168.1.2	Сетевой адрес компьютера (или другого сетевого устройства) принимающего профили.
Host_Data_Port	50001	Номер порта компьютера (или другого сетевого устройства) принимающего профили, на который сканер должен слать UDP пакеты с профилями.
HTTP_Port	80	Номер порта сканера для подключения по протоколу HTTP для доступа к WEB-странице сканера.
ServiceCtrlPort	50011	Номер порта сканера для сервисного протокола управления.
EIP_BroadcastRcvPort	44818	Служебный порт Ethernet IP.
EIP_ListeningTCPPort	44818	Служебный порт Ethernet IP.
Reserved[64]		Зарезервировано для новых параметров.

3.4.7. Структура параметров потоков передачи данных

Структура параметров **User_Streams_TypeDef** содержит параметры потоков передачи данных:

```
typedef struct __attribute__ ((__packed__))
{
	uint8_t		UDP_Profiles_Enabled;
	uint8_t		Profiles_Format;
	uint8_t		Profiles_Confirmation;
	uint8_t		Reserved[32];
}User_Streams_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
UDP_Profiles_Enabled	1	Включение и отключение потока UDP пакетов с профилями. 0 – выключен; др. – включен.
Profiles_Format	1	Установка формата передачи профилей: 0x10 - «raw profile»; 0x11 - «calibrated profile»; 0x12 - «extended raw profile»; 0x13 - «extended calibrated profile». др. – не должно использоваться.
Profiles_Confirmation	0	Включение и отключение требования подтверждения доставки UDP пакетов с профилями. 0 – выключено; др. – включено.
Reserved[32]		Зарезервировано для новых параметров.

3.4.8. Структура параметров алгоритма обработки изображений

Структура параметров **User_Processing_TypeDef** содержит параметры алгоритма обработки изображений, формируемых CMOS-сенсором:

```
typedef struct __attribute__ ((__packed__))
{
	uint32_t		Threshold;
	uint8_t		Stg1_FilterWidth;
	uint8_t		Stg1_ProcessingMode;
	uint8_t		Stg2_ReduceProfileNoise;
	uint32_t		ProfilesPerSecond;
	uint8_t		Reserved[60];
}User_Processing_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
Threshold	2000	Порог фильтрации недостаточно ярких точек. Применяется после первичной фильтрации точек изображения и поэтому имеет широкие пределы изменения: от 0 до 1632000. Точки, имеющие яркость меньше порога исключаются из обработки.
Stg1_FilterWidth	25	Ширина фильтра первичной обработки точек изображения. Допустимые пределы: от 1 до 25.

Название поля	Значение при заводских настройках	Описание
Stg1_ProcessingMode	2	Режим дополнительной математической обработки результатов первичной фильтрации: 0 – без дополнительной обработки; 1 – аддитивная обработка центральным ядром; 2 – мультипликативная обработка центральным ядром; 3 – смешанная обработка; др. – так же, как для значения 2.
Stg2_ReduceProfileNoise	0	Режим фильтрации нестабильных во времени точек. Включение данного режима позволяет устранить нестабильно обнаруживаемые точки профиля. 0 – выключено; др. – включено.
ProfilesPerSecond	–	Реальная частота профилей в секунду после обработки.
Reserved[60]		Зарезервировано для новых параметров.

3.4.9. Структура параметров, управляющих яркостью лазера

Структура параметров **User_Laser_TypeDef** содержит параметры, управляющие яркостью лазера (в настоящее время не используется):

```
typedef struct __attribute__ ((__packed__))
{
	uint8_t		Enabled;
	uint8_t		AutoMode;
	uint16_t	Value;
	uint8_t		Reserved[32];
}User_Laser_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
Enabled	1	Включение и отключение излучения лазера. 0 – выключен; др. – включен.
AutoMode	0	Включение и отключение автоматического управления яркостью лазера. 0 – выключено; др. – включено.
Value	10	Заданное значение яркости лазера. Используется в режиме выключенного автоматического управления. Допустимые пределы: от 0 – лазер выключен, до 100 – лазер излучает с максимально допустимой яркостью.
Reserved[32]		Зарезервировано для новых параметров.

3.4.10. Структура параметров, управляющих работой входных каналов

Структура **User_InputsPreset_TypeDef**:

```
typedef struct __attribute__ ((__packed__))
{
	uint16_t		ParamsMask;
	uint8_t		In1_Enabled;
	Input1Mode		In1_Mode;
	uint32_t		In1_Delay;
	uint8_t		In1_Divider;
	uint8_t		In2_Enabled;
	Input2Mode		In2_Mode;
	uint8_t		In2_Inverse;
	uint8_t		In3_Enabled;
	Input3Mode		In3_Mode;
	uint8_t		Reserved[12];
}User_InputsPreset_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
ParamsMask		Маска используемых в пресете параметров (мл. бит - первое поле и т.д.).
In1_Enabled	0	Включение и отключение использования Входа №1. 0 – выключен; др. – включен.
In1_Mode	0	Режим работы Входа №1: 0 – запуск накопления кадра CMOS-сенсором по фронту импульса на входе; 1 – запуск накопления кадра CMOS-сенсором по спаду импульса на входе; 2 – стробирование внутреннего генератора импульсов запуска накопления кадра CMOS-сенсором логической «1» на входе; 3 – стробирование внутреннего генератора импульсов запуска накопления кадра CMOS-сенсором логическим «0» на входе; др. – не должно использоваться.
In1_Delay	100	Задержка запуска накопления кадра CMOS-сенсором относительно события на входе в наносекундах. Шаг 10 нс.
In1_Divider	0	Значение прореживания событий запуска накопления кадра CMOS-сенсором. Если установлено значение «0», то кадр формируется на каждое событие, если значение «1», то кадр будет формироваться через одно событие запуска, если значение «2», то кадр будет формироваться через два события запуска и т.д.
In2_Enabled	0	Включение и отключение использования Входа №2. 0 – выключен; др. – включен.
In2_Mode	0	Режим работы Входа №2: 0 – определение направления перемещения по уровню на входе; 1 – определение направления перемещения по соотношению фаз на Входе №1 и Входе №2; др. – не должно использоваться.
In2_Inverse	0	Режим работы инверсии для Входа №2.
In3_Enabled	0	Включение и отключение использования Входа №3. 0 – выключен;

Название поля	Значение при заводских настройках	Описание
		др. – включен.
In3_Mode	0	Режим работы Входа №3: 0 – сброс внутреннего счетчика профилей по фронту импульса на входе; 1 – сброс внутреннего счетчика профилей по спаду импульса на входе; др. – не должно использоваться.
Reserved[12]		Зарезервировано для новых параметров.

Структура **User_Inputs_TypeDef**:

```
typedef struct __attribute__ ((__packed__))
{
	uint8_t 			PresetIdx;
	User_InputsPreset_TypeDef 	PresetParams[12];
	uint8_t 			Reserved[32];
}User_Inputs_TypeDef;
```

Название поля	Описание
PresetIdx	Индекс используемого пресета. Определяет номер структуры из массива структур PresetParams[12].
PresetParams[12]	Массив структур с пресетами. Первые 9 структур (с индексами 0...8) зарезервированы произведителем и не должны изменяться пользователем.
Reserved[32]	Зарезервировано для новых параметров.

3.4.11. Структура параметров, управляющих работой выходных каналов

Структура параметров **User_Outputs_TypeDef** содержит параметры, управляющие работой выходных каналов сканера:

```
typedef struct __attribute__ ((__packed__))
{
	uint8_t 		Out1_Enabled;
	OutputMode 		Out1_Mode;
	uint32_t 		Out1_Delay;
	uint32_t 		Out1_PulseWidth;
	uint8_t 		Out1_Inverse;
	uint8_t 		Out2_Enabled;
	OutputMode 		Out2_Mode;
	uint32_t 		Out2_Delay;
	uint32_t 		Out2_PulseWidth;
	uint8_t 		Out2_Inverse;
	uint8_t 		Reserved[32];
}User_Outputs_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
Out1_Enabled, Out2_Enabled	0, 0	Включение и отключение использования выходов. 0 – выключен; др. – включен.
Out1_Mode, Out2_Mode	1, 1	Режим работы выходов: 0 – формирование импульса в момент запуска накопления кадра CMOS-сенсором; 1 – повторение логического уровня на Входе №1;

Название поля	Значение при заводских настройках	Описание
		2 – формирование импульса в момент перехода логического уровня с «0» на «1» на Входе №1; 3 – формирование импульса в момент перехода логического уровня с «1» на «0» на Входе №1; 4 – повторение логического уровня на Входе №2; 5 – формирование импульса в момент перехода логического уровня с «0» на «1» на Входе №2; 6 – формирование импульса в момент перехода логического уровня с «1» на «0» на Входе №2; 7 – повторение логического уровня на Входе №3; 8 – формирование импульса в момент перехода логического уровня с «0» на «1» на Входе №3; 9 – формирование импульса в момент перехода логического уровня с «1» на «0» на Входе №3; др. – не должно использоваться.
Out1_Delay, Out2_Delay	500, 50	Задержка импульса на выходе в наносекундах относительно запускающего события (с учетом режима работы выхода). Шаг 10 нс.
Out1_PulseWidth, Out2_PulseWidth	1000, 100	Ширина импульса на выходе в наносекундах. Шаг 10 нс.
Out1_Inverse, Out2_Inverse	0, 0	Включение и отключение инверсии выходного значения. 0 – выключена; др. – включена.
Reserved[32]		Зарезервировано для новых параметров.

3.5. Примеры управления сканером по сервисному протоколу

3.5.1. Поиск сканеров в сети

Поиск сканеров в сети осуществляется с помощью отправки UDP пакета с командой CMD_U_GENERAL_HELLO модулю USER_PARAMS на широковещательный сетевой адрес (для заводских настроек 192.168.1.255) на порт сервисного протокола (для заводских настроек 50011). Пример отправленного пакета:

```
0000  ff ff ff ff ff ff f8 32  e4 bb 8a 91 08 00 45 00
0010  00 2a 7c a7 00 00 80 11  39 ca c0 a8 01 02 c0 a8
0020  01 ff ff 6e c3 5b 00 16  3e a5 1c 00 00 00 ff ff
0030  ff ff 00 00 5e 00 00 00
```

В ответ сканер пришлет UDP пакет, содержащий сообщение с ответом на команду CMD_U_GENERAL_HELLO и в качестве полезной нагрузки структуру **HelloAnswer_TypeDef**:

```
0000  f8 32 e4 bb 8a 91 00 0a  35 3b 56 45 08 00 45 00
0010  02 36 dd ef 00 00 ff 11  58 56 c0 a8 01 1e c0 a8
0020  01 02 c0 01 c3 5b 02 22  c1 04 24 00 00 00 00 3b
0030  56 45 00 00 5e 00 0c 02  52 46 36 32 37 20 32 44
0040  20 4c 61 73 65 72 20 73  63 61 6e 6e 65 72 00 00
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0070  00 00 00 00 00 00 00 00  73 02 00 3b 56 45 04 01
0080  01 01 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00c0  00 00 e8 03 c0 a8 01 1e  ff ff ff 00 c0 a8 01 01
00d0  c0 a8 01 02 51 c3 50 00  5b c3 12 af 12 af 00 00
00e0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 05
0100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0110  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0120  00 00 01 01 00 00 00 00  00 00 00 00 00 00 00 00
0130  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0150  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0160  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0170  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0180  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0190  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
01a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
01b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
01c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
01d0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
01e0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
01f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0200  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0210  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0220  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0230  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0240  00 00 00 00
```

Поля структуры имеют следующие значения:

```
typedef struct
{
	struct
	{
		uint8_t			Name[64];
		uint16_t		DeviceID;
		uint32_t		Serial;
		uint32_t		FirmWareVer;
		uint8_t			Reserved[64];
	}General;
	struct
	{
		uint16_t		Speed;
		uint32_t		IP;
		uint32_t		Mask;
		uint32_t		Gateway;
		uint32_t		HostIP;
		uint16_t		HostProfilesPort;
		uint16_t		HTTPPort;
		uint16_t		ServicePort;
		uint16_t		EIPBroadcastRcvPort;
		uint16_t		EIPListeningTCPPort;
		uint8_t			Reserved[32];
	}Network;
	struct
	{
		uint32_t		MaxPayloadSize;
		uint8_t			Reserved[32];
```

```
        }ServiceProtocol;
        struct
        {
                uint8_t                 ProfilesEnabled;
                uint8_t                 ProfilesFormat;
                uint8_t                 Reserved[32];
        }Streams;
        uint8_t                         Reserved[256];
}HelloAnswer_TypeDef;
```

Название поля	Значение при заводских настройках	Описание
Name	RF627 2D Laser scanner	Назначаемое пользователем имя сканера. Может использоваться для упрощения идентификации сканеров в сети предприятия.
DeviceID	627	Идентификатор типа сканера – для серии РФ627 всегда 627.
Serial	----	Серийный номер сканера – присваивается компанией-производителем, уникален для каждого сканера.
FirmWareVer	----	Текущая версия прошивки сканера.
Speed	----	Текущая скорость сетевого соединения: 100 – 100 Мбит/с; 1000 – 1000 Мбит/с.
IP	192.168.1.30, uint32_t: 0x1e01a8c0	Сетевой адрес сканера.
Mask	255.255.255.0, uint32_t: 0x00ffffff	Маска подсети сканера.
Gateway	192.168.1.1, uint32_t: 0x0101a8c0	Сетевой адрес шлюза.
HostIP	192.168.1.2, uint32_t: 0x0201a8c0	Сетевой адрес компьютера (другого сетевого устройства), принимающего UDP пакеты с профилями.
HostProfilesPort	50001	Номер порта компьютера (другого сетевого устройства), на который отправляются UDP пакеты с профилями.
HTTPPort	80	Номер порта для HTTP соединения.
ServicePort	50011	Номер порта компьютера (другого сетевого устройства), используемого для передачи сообщений сервисного протокола.
EIPBroadcastRcvPort	44818	Служебный порт протокола Ethernet IP.
EIPListeningTCPPort	44818	Служебный порт протокола Ethernet IP.
MaxPayloadSize	----	Максимальный размер полезной нагрузки в сообщениях сервисного протокола в байтах.
ProfilesEnabled	1	Флаг разрешения отправки UDP пакетов с профилями: 0 – отправка пакетов запрещена; др. – отправка пакетов разрешена.
ProfilesFormat	1	Формат передачи профилей в UDP пакете: 0 – «plain measure»; 1 – «plain profile»; 2 – «extended measure»; 3 – «extended profile»; др. – не должно использоваться.

3.5.2. Установка времени экспозиции

Для установки времени экспозиции кадра (в примере 50000 нс, т.е. 50 мкс) CMOS-сенсором необходимо отправить сообщение с командой CMD_U_SENSOR_SET модулю USER_PARAMS и структурой User_Sensor_TypeDef в полезной нагрузке на сетевой адрес сканера, на порт сервисного протокола:

```
SrvcMsg_Type         Msg;
Msg.Header.Operation = MSG_COMMAND_CNFRM_FINAL;
             Msg.Header.DevID            = Factory.General.Serial;
             Msg.Header.ModuleID         = MID_USER_PARAMS;
             Msg.Header.Cmd                    = CMD_U_SENSOR_SET;
             Msg.Header.PayloadLen       = sizeof(User_Sensor_TypeDef);
             User_Sensor_TypeDef* Data   = (User_Sensor_TypeDef*)Msg.Payload;
             Data->DoubleSpeedMode       = 0;
             Data->Gain_Analog           = 6;
             Data->Gain_Digital          = 108;
             Data->Exposure                    = 50000;
             Data->FrameRate                   = 485;
             Data->AutoExposure          = 0;
             udpServiceSocket.send_data(Msg.RawData,     sizeof(SrvcMsgHeader_Type)     +
Msg.Header.PayloadLen);
```

Соответствующий UDP пакет:

```
0000  00 0a 35 64 c6 e0 f8 32  e4 bb 8a 91 08 00 45 00
0010  00 7d 42 05 00 00 80 11  74 fa c0 a8 01 02 c0 a8
0020  01 1e c3 5b c3 5b 00 69  b1 e7 1c 00 00 00 e0 c6
0030  64 00 00 00 5e 08 53 00  00 06 6c 50 c3 00 00 00
0040  00 00 00 e5 01 00 00 00  00 00 00 00 00 00 00 00
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0080  00 00 00 00 00 00 00 00  00 00 00
```

Получение и выполнение данной команды сканер подтвердит:

```
0000  f8 32 e4 bb 8a 91 00 0a  35 64 c6 e0 08 00 45 08
0010  00 2a 32 e9 00 00 80 11  84 61 c0 a8 01 1e c0 a8
0020  01 02 c3 5b c3 5b 00 16  2e ca 24 00 00 00 e0 c6
0030  64 00 00 00 5e 08 00 00  00 00 00 00
```

3.5.3. Запрос сетевых настроек сканера

Для получения сетевых параметров сканера необходимо отправить сообщение с командой CMD_U_NETWORK_GET модулю USER_PARAMS в виде UDP пакета:

```
0000  00 0a 35 3b 56 45 f8 32  e4 bb 8a 91 08 00 45 00
0010  00 2a 6c 2f 00 00 80 11  4b 23 c0 a8 01 02 c0 a8
0020  01 1e c3 5b c3 5b 00 16  23 0e 1c 00 00 00 00 3b
0030  56 45 02 00 5e 0b 00 00
```

Получение команды сканер подтвердит сообщением с полезной нагрузкой в виде структуры User_Network_TypeDef:

```
0000  f8 32 e4 bb 8a 91 00 0a  35 3b 56 45 08 00 45 00
0010  00 87 2f f8 00 00 ff 11  07 fd c0 a8 01 1e c0 a8
0020  01 02 c0 01 c3 5b 00 73  d6 42 24 00 00 00 00 3b
0030  56 45 02 00 5e 0b 5d 00  e8 03 01 c0 a8 01 1e ff
0040  ff ff 00 c0 a8 01 01 c0  a8 01 02 51 c3 50 00 5b
0050  c3 12 af 12 af 00 00 00  00 00 00 00 00 00 00 00
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0090  00 00 00 00 00
```

Установка и получение других параметров сканера осуществляется аналогичным образом.

4. Техническая поддержка

Запросы по технической поддержке следует направлять на адрес support@riftek.com или по телефону +375-17-2813513.

5. Изменения

Дата	Версия	Описание
16.11.2018	1.0.0	Исходный документ.