Аннотация

В настоящем документе представлено описание программного обеспечения ПО EVM IPx (программного обеспечения для IP-камер)

# Общие сведения

## Наименование: ПО EVM IPx.

## Обозначение ФАНВ.00119.

## Дополнительное программное обеспечение, необходимое для функционирования комплекса: ОС Linux kernel версия 4.9.

## Языки программирования: Cи, Си++, применительно к компилятору GCC версия 7.5.0.

# Функциональное назначение

## Программное обеспечение предназначено для решения следующих задач:

- прием процессором видеопотока от светочувствительной матрицы;

- обработка видео (фильтрация, усиление, компенсация засветки, выбор баланс белого);

- кодирование видео H.264 и H.265;

- оцифровка звука с микрофона;

- кодирование звука G.711(a/u);

- передача видеопотока и звука по сети передачи данных Ethernet;

- формирование и передача стоп-кадра по запросу;

- детекция движения;

- удалённая настройка и управление через WEB-консоль;

# Описание логической структуры

## Алгоритм программы

### В штатном режиме работы программа загружается автоматически после включения питания. При этом начинается параллельное выполнение следующих задач:

- прием данных с микрофона, светочувствительной матрицы, обработка и кодирование, определение движения (модуль ipc\_video);

- обработка запросов по сети, установка связи, передача видео и звуковых данных по RTSP протоколу и стоп-кадра по HTTP-протоколу, обработка API запросов по протоколу HTTP и запросов по протоколу ONVIF Specifications S (модуль onvifrtspserver);

- передача данных между модулем ipc\_video и onvifrtspserver реализована через общую память (библиотека libvintf.so);

- обработка запросов состояния и управление устройством через WEB (с помощью встроенного в linux вебсервера httpd и HASERL-cgi скриптов из папки /var/www).

### Логически программный комплекс можно разбить на следующие программные модули:

- модули инициализации изделия;

- модуль обработки звука, видео и подготовки стоп-кадра;

- модуль передачи видео, аудио данных и стопкадра по сети;

- библиотека интерфейса передачи данных через общую память;

- cgi скрипты для веб консоли;

- вспомогательные shell-скрипты для выполнения настроек устройства;

- конфигурационные файлы;

- вспомогательные утилиты.

## Назначение функциональных модулей

### Логику работы изделия задают программные сценарии и программные модули, представленные в Таблице 1.

Таблица 1– Перечень программных сценариев и программных модулей.

| **Наименование разделов, файлов** | **Назначение, входные и выходные данные** |
| --- | --- |
| **/usr/bin/setdhcp.sh****/usr/bin/getdhcp.sh****/usr/bin/startnetwok.sh****/usr/bin/stopnetwok.sh** | * Программные сценарии для включения/выключения DHCP и запуска/остановки взаимодействия по сети передачи данных.
* Входные данные для setdhcp: 0/1 - выкл./вкл. DHCP
* Выходные данные для getdhcp: код возврата 1/0 –DHCP вкл./выкл.
 |
| **send2ftp.sh****send2http.sh** | * Программный сценарий отправки стоп кадра на FTP и HTTP сервисы
* Входные данные: информация о сервере, логин/пароль, передаваемый файл
* Выходные данные: код возврата 0 успех, не 0 – ошибка.
 |
| **snapshot2file.sh** | * Программный сценарий получения стоп кадра перед отправкой.
* Входные данные: флаг принудительного запроса, в противном случае стопкадр обновится только если прошло более 120 секунда.
* Выходные данные: код возврата 0 успех, не 0 – ошибка.
 |
| **sysupgrade** | * Программный сценарий, позволяющий обновить прошивку или ядро linux;
* Входные данные: файл прошивки или ядра Linux, настройки обновления (очистка оверлея, принудительный перезапуск и т.д.)
* Выходные данные: -
 |
| **firstboot** | * Программный сценарий, очищающий оверлей
* Входные данные: флаг принудительной перезагрузки;
* Выходные данные: -
 |
| **S01syslogd** | * Программный сценарий, запуска syslogd с возможностью перенаправления на внешний сервер.
* Входные данные: адрес внешнего сервера
* Выходные данные: -
 |
| **S41factoryreset** | * Программный сценарий сброса на заводские настройки при запуске камеры.
 |
| **S50httpd** | * Программный сценарий запуска HTTP сервера для WEB-консоли.
 |
| **S95ipc** | * Программный сценарий запуска ipc\_videod и onvifrtspserver.
 |
| **S98tunnel** | * Программный сценарий установления туннеля для видео сервисов сбора видеоданных.
 |
| **/usr/lib/libvintf.so** | * Библиотека передачи аудио данных, видео данных и стопкадра между ipc\_videod и onvifrtspserver через общую память.
 |
| **/usr/bin/ipc\_videod** | * Программный модуль получения, обработки и кодирование видео и аудиоданных, формирование стоп кадра по запросу.
* Входные данные: поток данных с сенсора светочувствительной матрицы, оцифрованные аудиоданные с микрофона, настройки из файла vconfig.yaml.
* Выходные данные: видео и аудио поток данных, стоп кадр.
 |
| **/usr/bin/onvifrtspserver** | * Программный модуль получения запросов на установку соединения по протоколам RTSP и HTTP.
* Входные данные: запросы по сети передачи данных по протоколу RTSP и HTTP
* Выходные данные: видео и аудио потоки по протоколу RTSP и стоп кадр по протоколу HTTP
 |
| **/var/www/cgi-bin/\*** | * CGI скрипты для WEB консоли
 |
| **/var/www/js/\*** | * JavaScript скрипты для WEB консоли
 |

## Блок-схема программного обеспечения

### Основная блок-схема программного обеспечения ПО EVM IPx изделия EVM IPx:

Начало

Системный сценарий загрузки **/etc/init.d/rcS** запускает на выполнение сценарий запуска **/etc/init.d/S01syslogd**

Системный сценарий загрузки **/etc/init.d/rcS** запускает на выполнение сценарий запуска **/etc/init.d/S41factoryreset**

Системный сценарий загрузки **/etc/init.d/rcS** запускает на выполнение сценарий запуска **/etc/init.d/S50httpd**

Системный сценарий загрузки **/etc/init.d/rcS** запускает на выполнение сценарий запуска **/etc/init.d/S95ipc**, запускающий модули **ipc\_videod** и **onvifrtspserver**

Системный сценарий загрузки **/etc/init.d/rcS** запускает на выполнение сценарий запуска **/etc/init.d/S98tunnel**

Конец

# Используемые технические средства

## Назначение

### Программа предназначена для использования в изделии EVM-IPx и его модификациях.

## Минимальные требования к аппаратным средствам.

### Минимальные требования к аппаратным средствам представлены в Таблице 2.

Таблица 2 – Перечень минимальных требований к аппаратным средствам.

|  |  |
| --- | --- |
| **Характеристика** | **Примечание** |
| Процессор GK7205v200 или GK7205v300 | В зависимости от модификации EVM-IPx |
| ОЗУ 128Mb |  |
| SPI NOR FLASH 16Mb |  |
| Сенсор SC2335, GC4653, IMX307 или IMX335 | В зависимости от модификации EVM-IPx |

# Вызов и загрузка

## Первоначально программные модули комплекса записываются на установленную на модуле энергонезависимую память SPI NOR FLASH с использованием установочных скриптов.

##  При работе по прямому назначению загрузка комплекса в ОЗУ и запуск осуществляются автоматически при включении питания.

## По окончании загрузки программного комплекса изделие готово работе и установке корпус. Дальнейшие действия оператора приведены в Руководстве по Эксплуатации на изделие EVM IPx.

# Входные данные

## Входными данными ПО являются: поток данных с сенсора светочувствительной матрицы, оцифрованные аудиоданные с микрофона, команды управления от пользователя через WEB и API, запросы через сеть передачи данных на установку соединения и передачу потока по RTSP и HTTP.

# Выходная информация

## Выходной информацией программного комплекса является RTP потоки с видео и аудио в рамках RTSP-соединений, информация о состоянии, и стоп кадр в ответ на HTTP запрос.

# Архитектура ПО.

## Архитектура ПО EVM IPx является модульной архитектурой, что позволяет уменьшить сложность и формализовать взаимодействие между составными частями. Для обеспечения передачи данных в реальном времени, минимизации задержек и снижения нагрузки на аппаратные ресурсы для передачи данных между модулями используется общая память (shared memory). Работу с общей памятью осуществляет разработанная интерфейсная библиотека libvintf.so (рис.1 Архитектура ПО EVM IPx)



Рис.1 Архитектура ПО EVM IPx

## Для получения аудио и видео данных в цифровом виде программный модуль ipc\_videod настраивает и коммутирует видео и аудио подсистему процессоров GK7205v200 и GK7205v300 для получения двух потоков видео информации и потока аудиоинформации (Рис.2). Таким образом видео сигнал проходит стадии АЦП на сенсоре, поступает на видеовход (VI), систему видеобработки (VPSS) и модуль кодирования (VENC). А звук поступает с аудиовхода (AI) на кодировщик (AENC).



Рис.2 Архитектура видео и аудио подсистемы процессора GK7205v200/v300

# Приложение А. Процедура компиляции текста программы

## Выполнить git checkout к нужным версиям ПО по названию ветки или хэш-маркера изменения.

## Собрать модули и библиотеки командой

## PROC=GK7205v200 ./make\_all.sh

## Обновить в нужных overlay папках бинарники ipc\_videod, libvintf.so, onvifrtspserver

## Повторить пункты 2. и 3. для GK7205v300.

## Перейти в папку сборки прошивки IPC билдрутом.

## Выполнить сборку всех прошивок командой ./make\_firmwares.sh

## В результате работы скрипта сборки прошивок они будут лежать в папке /mnt/c/Temp/camhi\_my, разложенные по папкам вида: procname\_sensorname\_mod например GK7205v200\_imx307\_ipc