Руководство программиста АГАВА ПЛК-86М

Редакция от 29.10.2026 · docs.kb-agava.ru

Содержание

1 Область применения и конфигурация 2
2 Аппаратные ресурсы и программные интерфейсы 2
3 Архитектура доступа к оборудованию 3
4 Релейные выходы 4
4.1 Интерфейс и контакты 4
4.2 Программное управление на языке C 5
4.3 Состояния выходов при запуске, остановке и отказе 6
5 Обмен по интерфейсу RS-485 6
5.1 Подключение и параметры 6
5.2 Организация прикладного обмена 7
5.3 Modbus RTU 7
6 Ethernet и USB RNDIS 8
6.1 Ethernet 8
6.2 USB RNDIS 8
7 Wi-Fi и Bluetooth 9
7.1 Wi-Fi 9
7.2 Bluetooth 9
8 Экран и сенсор 10
8.1 Распределение доступа к графической подсистеме 10
8.2 Непосредственное обращение к кадровому буферу 10
8.3 Сенсор GT911 11
9 Подсветка и индикатор work 11
9.1 Яркость подсветки 11
9.2 Индикатор work 12
10 Звуковые сигналы и микрофон 12
10.1 Формирование звукового сигнала 12
10.2 Воспроизведение 13
10.3 Запись микрофона 13
11 АЦП и температура 13
11.1 Чтение SARADC 13
11.2 Температура процессора 14
12 Системное время и RTC 14
13 Память и хранение данных 15
13.1 Распределение оперативной памяти 15
13.2 Постоянное хранение данных 15
14 Watchdog и ресурсы SoC 16
14.1 Контроль работоспособности 16
14.2 Генератор случайных чисел 17
14.3 Нейронный процессор NPU 17
15 Развёртывание приложения 17
15.1 Передача и ручной запуск 17

Настоящая инструкция определяет порядок использования аппаратных ресурсов АГАВА ПЛК-86М при разработке прикладного программного обеспечения. Рассматриваются интерфейсы Linux, управление релейными выходами, сетевой и последовательный обмен, графическая подсистема, аудиоустройства, часы реального времени и встроенные ресурсы системы на кристалле.

Настоящая инструкция предназначена для разработчиков программного обеспечения.

Приведённые сведения об аппаратуре и драйверах относятся к указанной далее конфигурации. Архитектура приложения, примеры программных адаптеров и порядок развёртывания рассматриваются как рекомендации по разработке прикладного программного обеспечения.

1 Область применения и конфигурация

Конфигурация, для которой в исходной методике приведены результаты инвентаризации, представлена в таблице.

Параметр	Значение
Модель устройства	`AGAVA-PLK-86M RV 4' v1`
Образ прошивки	Yocto Agava3 2026.03
Ядро Linux	`5.10.160-rt79-rockchip-rv1106`
Процессор	RV1106, Cortex-A7
Установленная RAM	256 МБ DDR3L

В рассматриваемой конфигурации использование USB host и аппаратного сторожевого таймера watchdog функционально не обеспечено.

2 Аппаратные ресурсы и программные интерфейсы

Прикладное взаимодействие с оборудованием осуществляется через интерфейсы соответствующих подсистем Linux. Номера GPIO приведены для сопоставления с электрической схемой и не заменяют интерфейсы установленных драйверов.

Ресурс	Интерфейс	Условие использования
Реле 1 и 2	`/dev/relay1`, `/dev/relay2`	Драйвер `agava-relay`, текстовые команды 0 и 1
RS-485	`/dev/ttyS4`	Один владелец порта; параметры линии задаёт проект
Ethernet	`end0`	Стандартные сетевые сокеты Linux
USB RNDIS	`usb0`	Периферийный режим, IP контроллера 192.168.7.1
Wi-Fi	`wlan0`	AIC8800DC, SDIO; настройка существующей сетевой службой
Bluetooth	`hci0`	Внутренний `ttyS1` занят `luckfox-bt-attach.service`
Экран	`/dev/fb0`	720 × 720, XRGB8888, stride 2880 в проверенном образе

Сенсор	`/dev/input/touchscreen0`	GT911, события evdev; в проверенной конфигурации соответствует event0
Подсветка	`/sys/class/backlight/backlight`	Предел шкалы определяется `max_brightness`; в проверенном образе 255
Индикатор work	`/sys/class/leds/work`	Управление через LED class с учётом trigger
Динамик и микрофон	ALSA card 0 `rv1106-acodec`	Доступ с учётом использования кодека PulseAudio
Beeper	`/dev/tty5`	Программный звук через динамик
RTC	`/dev/rtc0`	Драйвер `rockchip-rtc`, соглашение UTC
АЦП	`/sys/bus/iio/devices/iio:device0`	SARADC, каналы 0 и 1, scale в мВ на отсчёт
Температура SoC	`/sys/class/thermal/thermal_zone0/temp`	Целое число в тысячных долях °C
eMMC	`/dev/mmcblk0`	Прикладное хранение через файловую систему
NPU	`/dev/rknpu`	Требуются совместимые RKNN runtime и модель
Аппаратный RNG	`/dev/hwrng`	Наличие не подтверждает качество энтропии
Watchdog	`/dev/watchdog*` отсутствует	Требуется отдельная доработка BSP

Устройство `/dev/ttyS1` предназначено для Bluetooth и не используется для RS-485.

Устройство `/dev/ttyFIQ0` обслуживает консоль.

Линии GPIO1_A0 и GPIO1_A1 заняты драйвером реле, а GPIO3_C6 — драйвером светодиодного индикатора.

3 Архитектура доступа к оборудованию

В структуре проекта целесообразно выделять прикладную логику, программный адаптер оборудования и системные службы. Адаптер проверяет доступность ресурсов, выполняет операции ввода-вывода и передаёт прикладной логике результаты и сведения об ошибках. Настройка сети, подключение Bluetooth-контроллера и штатное звуковое сопровождение относятся к функциям соответствующих системных служб.

Для релейных выходов, порта RS-485 и экрана назначается единственный компонент, непосредственно управляющий ресурсом; далее он обозначается как владелец ресурса. Команды остальных компонентов передаются владельцу через согласованный интерфейс. Успешное открытие файла устройства не обеспечивает межпроцессного взаимного исключения. При необходимости владелец реализуется отдельной службой с механизмом межпроцессного взаимодействия IPC.

Рекомендуемая последовательность запуска приложения включает следующие операции:

1. Прочитать конфигурацию проекта и проверить её диапазоны.
2. Найти обязательные устройства и проверить права доступа.
3. Установить исходное состояние выходов, определённое проектом.
4. Открыть интерфейсы связи, запустить обмен и определить достоверность входных данных.
5. Разрешить исполнительные команды после выполнения условий готовности.

Для каждого измерения сохраняются значение, время получения и признак качества. После нарушения связи последнее полученное значение помечается как устаревшее. Операции с UART, сетью и файловой системой должны предусматривать ограничение времени ожидания и обработку ошибок.

Остановка приложения включает прекращение приёма новых команд, установку предусмотренных проектом состояний выходов, сохранение необходимых данных и закрытие интерфейсов. Обработчик сигнала в программе на C/C++ устанавливает только признак завершения. Операции с файлами и оборудованием выполняются в основном потоке.

Наличие PREEMPT_RT не является гарантией соблюдения временных ограничений прикладной операции.

Запись на eMMC, воспроизведение звука, обработка графического интерфейса и восстановление сетевых соединений выносятся за пределы критического цикла.

Период опроса и допустимые задержки определяются требованиями проекта и проверяются измерениями при рабочей нагрузке.

4 Релейные выходы

4.1 Интерфейс и контакты

Выход	Устройство	Вывод RV1106	Контакты разъёма
Реле 1	`/dev/relay1`	GPIO1_A0, номер sysfs 32	1 — NO, 2 — COM
Реле 2	`/dev/relay2`	GPIO1_A1, номер sysfs 33	3 — NO, 4 — COM

Назначение контактов до подключения сверяется с маркировкой конкретного исполнения и руководством по эксплуатации. Допустимые напряжение и ток, категория нагрузки и необходимость внешней защиты определяются документацией изделия и схемой проекта. В исходном файле значения этих параметров не установлены.

Запись текстового значения `0` переводит реле в выключенное состояние, а `1` — во включённое. Операция чтения возвращает логическое состояние управления, зарегистрированное драйвером. При включении реле замыкается соответствующая пара контактов NO–COM. В приведённом примере после начального выключения обоих каналов выполняется включение и последующее выключение реле 1. Пример предназначен для подготовленного стенда:

```
test -c /dev/relay1 && test -c /dev/relay2 || exit 1
printf '0\n' > /dev/relay1
printf '0\n' > /dev/relay2
cat /dev/relay1 /dev/relay2
printf '1\n' > /dev/relay1
cat /dev/relay1
```

После проверки контактов вернуть исходное выключенное состояние.

```
printf '0\n' > /dev/relay1
```

Считанное значение `1` характеризует состояние управления и не подтверждает физическое замыкание контактов либо работоспособность нагрузки. Если проектом предусмотрена обратная связь, фактическое состояние определяется отдельным входом или соответствующим датчиком.

4.2 Программное управление на языке C

Приведённая функция выполняет запись двух байтов и контролирует результат системного вызова. Допустимые номера каналов ограничены значениями 1 и 2. Передача произвольного пути к устройству в качестве пользовательского аргумента не предусмотрена.

```
include <errno.h>
include <fcntl.h>
include <unistd.h>
int relay_set(unsigned channel, int on)
{
   const char *path;
   int fd, saved;
   ssize_t n;
   char cmd[2];
```

```
   if ((channel != 1 && channel != 2) ||
       (on != 0 && on != 1)) {
       errno = EINVAL;
       return -1;
   }
   path = channel == 1 ? "/dev/relay1" : "/dev/relay2";
   fd = open(path, O_WRONLY | O_CLOEXEC);
   if (fd < 0) return -1;
   cmd[0] = on ? '1' : '0';
   cmd[1] = '\n';
   do { n = write(fd, cmd, sizeof cmd); }
   while (n < 0 && errno == EINTR);
   if (n != (ssize_t)sizeof cmd) {
       saved = n < 0 ? errno : EIO;
       (void)close(fd);
       errno = saved;
       return -1;
   }

   return close(fd);
}
```

При ошибке записи фактическое состояние выхода рассматривается как неопределённое до проведения повторной диагностики. Ошибка закрытия файлового дескриптора также регистрируется, поскольку предшествующая команда могла быть выполнена. Допустимость автоматического повторения команды определяется режимом работы исполнительного механизма.

Чтение состояния выполняется после повторного открытия устройства в режиме чтения. Полученная строка проверяется на соответствие значениям `0` или `1`. Поддержка `lseek`, произвольных ioctl и дополнительного двоичного протокола драйвера не предполагается.

4.3 Состояния выходов при запуске, остановке и отказе

После загрузки драйвера, до поступления команд приложения, для обоих каналов ожидаются значения `0` и разомкнутые контакты. Завершение процесса и закрытие файловых дескрипторов не изменяют ранее установленное состояние выхода. После аварийного завершения программы реле может оставаться включённым.

Исходные состояния и состояния при отказе определяются проектной документацией. Если предусмотрено выключение обоих реле, приложение явно записывает значения `0` при запуске и штатной остановке. Указанный программный механизм неприменим при `SIGKILL`, зависании ядра и иных отказах, исключающих выполнение кода завершения.

Управление двумя выходами осуществляется отдельными операциями. Интерфейс не гарантирует атомарного изменения состояния пары. При запрете одновременного включения используется последовательность выключения активного канала, проверки условий и включения другого канала. Требуемая аппаратная блокировка обеспечивается электрической схемой.

Ручной экспорт GPIO32/33, создание узлов устройств посредством `mknod` и выгрузка `agava-relay` не относятся к штатным операциям управления. При отсутствии `/dev/relayN` проверяется соответствие образа и DTB предусмотренной конфигурации.

5 Обмен по интерфейсу RS-485

5.1 Подключение и параметры

Для обмена по RS-485 используется устройство `/dev/ttyS4`. Подключение оборудования с электрическим интерфейсом RS-232 к данному порту не допускается. Назначение клемм, полярность A/B, общий провод, оконечные сопротивления и параметры линии определяются электрической схемой конкретного исполнения. Замыкание проводников A и B не допускается.

Открытию порта предшествует исключение доступа со стороны другого владельца, включая среду исполнения ПЛК. Поиск открытых файловых дескрипторов на контроллере выполняется следующим образом:

```
for fd in /proc/[0-9]*/fd/*; do
   [ "$(readlink "$fd" 2>/dev/null)" != /dev/ttyS4 ] || \
       printf '%s\n' "$fd"
done
```

Исходная стендовая конфигурация предусматривает скорость 115200 бит/с, восемь бит данных, отсутствие контроля чётности и один стоп-бит. Рабочие параметры определяются протоколом проекта и должны быть согласованы для всех участников линии.

Прикладной владелец порта должен быть остановлен.

```
saved=$(stty -F /dev/ttyS4 -g) || exit 1
stty -F /dev/ttyS4 115200 raw -echo -ixon -ixoff \
   -crtscts cs8 -parenb -cstopb clocal cread
```

После отдельной стендовой проверки восстановить настройку.

```
stty -F /dev/ttyS4 "$saved"
```

5.2 Организация прикладного обмена

В приложении на C/C++ порт открывается вызовом `open`, текущие параметры считываются посредством `tcgetattr`, после чего устанавливаются режим raw, скорость через `cfsetispeed`/`cfsetospeed` и требуемый формат кадра. Применение параметров выполняется вызовом `tcsetattr`. Для организации ожидания допускается открытие с флагами `O_NONBLOCK | O_NOCTTY | O_CLOEXEC` и использование `poll`.

Последовательность выполнения обмена включает следующие операции:

1. Сформировать запрос и определить крайний срок операции по монотонным часам.
2. Передать запрос, учитывая частичную запись, `EINTR` и `EAGAIN`.
3. При необходимости дождаться завершения физической передачи через `tcdrain` с учётом ограничений выбранного драйвера.
4. Принимать ответ через `poll`/`read` до полного кадра или истечения срока. Один `read` не обязан вернуть весь ответ.
5. Проверить адрес, длину, код функции, контрольную сумму и соответствие запросу.
6. Вернуть значение и качество данных либо зарегистрировать тайм-аут или ошибку протокола.

При заданном предельном времени операции учитываются возможная блокировка `tcdrain` и особенности драйвера UART. Прерывание ожидания `poll` сигналом не должно приводить к повторному отсчёту полного интервала. Оставшееся время рассчитывается по показаниям монотонных часов.

Способ управления направлением трансивера определяется электрической схемой и драйвером данной ревизии. Параметры `TIOCSRS485`, полярность RTS и задержки DE задаются только после подтверждения их соответствия аппаратной реализации. Приведённая в исходной методике проверка обмена строками не устанавливает универсальную конфигурацию управления направлением для всех образов.

5.3 Modbus RTU

Реализация Modbus RTU основывается на совместимой библиотеке, собранной средствами целевого SDK, либо на проверенной реализации протокола. Помимо настройки UART предусматриваются межкадровые интервалы, контроль CRC, обработка исключений, предельное время ответа и порядок повторения запросов. Количество повторных запросов ограничивается.

Роли ведущего и ведомых устройств, а также их адреса задаются конфигурацией проекта с исключением конфликтов адресации. При выполнении команд записи учитывается возможность выполнения команды при потере ответного сообщения. Допустимость повторения определяется семантикой команды и способом проверки результата.

Функциональная проверка включает двусторонний обмен по рабочему протоколу на заданной скорости с использованием кабеля проекта. Контролируются превышения времени ожидания, ошибки CRC, сообщения с непредусмотренными адресами и восстановление после отключения ведомого устройства. Обмен текстовыми строками не заменяет проверку Modbus RTU.

6 Ethernet и USB RNDIS

6.1 Ethernet

Ethernet представлен интерфейсом `end0`, обслуживаемым драйвером `rk_gmac-dwmac`; физический порт поддерживает режимы 10/100 Мбит/с. Прикладной обмен осуществляется через стандартные TCP/UDP-сокеты с использованием фактической IP-адресации. Непосредственное обращение к PHY для такого обмена не требуется.

```
ip -br link show end0
ip -4 addr show dev end0
ip -4 route show dev end0
cat /sys/class/net/end0/carrier
cat /sys/class/net/end0/address
ip -s link show dev end0
```

Значение `carrier=1` указывает на наличие физического соединения. IP-адресация, маршрутизация, доступность сервера и работоспособность прикладного протокола проверяются независимо. Для проверенного драйвера возможны ответ `No data available` на `ethtool end0` и ошибка `Invalid argument` при чтении sysfs-атрибутов speed/duplex. Эти сообщения сами по себе не свидетельствуют об отказе порта.

Передача через заданный сетевой интерфейс обеспечивается конфигурацией адресов и маршрутов, а при необходимости — привязкой сокета к локальному адресу. Привязка исходного адреса не заменяет проверку маршрутизации. В следующем примере адрес компьютера подлежит замене в соответствии с конфигурацией стенда:

```
ETH_PEER=172.16.3.10
ETH_IP=$(ip -4 -o addr show dev end0 | \
   awk '{split($4,a,"/"); print a[1]; exit}')
test -n "$ETH_IP" && ip -4 route get "$ETH_PEER" from "$ETH_IP"
```

Результат проверки маршрута должен содержать `dev end0`. При совместном использовании Ethernet и Wi-Fi устанавливаются приоритеты маршрутов. Для стендовых измерений рекомендуется разнесение интерфейсов по различным подсетям.

Сетевая подсистема приложения должна обрабатывать разрыв соединения, изменение DHCP-адреса и истечение времени ожидания, а также обеспечивать восстановление связи с ограниченной частотой попыток. Нулевой результат `recv` при приёме данных TCP означает закрытие соединения. Поскольку TCP обеспечивает передачу потока байтов, границы сообщений определяются прикладным протоколом.

6.2 USB RNDIS

В рассматриваемом DT установлено `dr_mode = "peripheral"`. Контроллер подключается к компьютеру в качестве сетевого USB-устройства. Прикладной доступ осуществляется через интерфейс `usb0` средствами сетевого стека Linux. Разработка отдельного протокола взаимодействия с USB-контроллером не требуется.

```
ip -br addr show usb0
ip -s link show dev usb0
for udc in /sys/class/udc/*; do
   [ -d "$udc" ] || continue
```

```
   cat "$udc/state" "$udc/current_speed"
done
```

При подключённом компьютере для проверенной конфигурации ожидаются состояние `configured`, скорость `high-speed` и адрес `192.168.7.1/24`. Отключение кабеля приводит к разрыву сетевого соединения. Приложение должно завершать прерванную сессию и обеспечивать установление новой после восстановления подключения.

Использование USB-накопителя в данном образе не обеспечивается. Режим host/OTG требует согласованной конфигурации прошивки, DT и питания VBUS. Произвольное изменение роли посредством записи в sysfs не допускается. Переход на host-образ требует повторной проверки карты устройств и способа сервисного доступа.

7 Wi-Fi и Bluetooth

7.1 Wi-Fi

Интерфейс `wlan0` обслуживается штатным экземпляром `wpa_supplicant`. Подключение к SSID настраивается через сетевую вкладку `sysutil40` либо предусмотренный образом профиль. Прикладной обмен осуществляется через сокеты. Одновременный запуск дополнительного экземпляра `wpa_supplicant`, управляющего тем же интерфейсом, не допускается.

```
systemctl status wpa_supplicant-nl80211@wlan0.service --no-pager
wpa_cli -i wlan0 ping
wpa_cli -i wlan0 status
iw dev wlan0 link
ip -4 addr show dev wlan0
```

Признаками установленного подключения являются ответ `PONG`, состояние `wpa_state=COMPLETED`, соответствие SSID/BSSID заданной точке доступа и наличие IP-адреса. Сообщение об отсутствии управляющего сокета требует проверки службы и каталога `/run/wpa_supplicant` и не является достаточным основанием для заключения об аппаратной неисправности модуля.

Пароль Wi-Fi хранится штатным средством конфигурации с ограничением прав доступа и исключается из журналов и диагностических сообщений. Восстановление связи после потери точки доступа проверяется отдельно. Использование режима AP должно соответствовать конфигурации образа; наличие `hostapd` не подтверждает функциональную готовность точки доступа.

7.2 Bluetooth

Bluetooth представлен контроллером `hci0`. Внутренний порт `/dev/ttyS1` обслуживается службой `luckfox-bt-attach.service`. Прикладная интеграция осуществляется через интерфейсы установленной версии BlueZ и выбранного профиля, в частности D-Bus. Доступность профиля, порядок сопряжения и модель доступа устанавливаются по конфигурации целевого образа.

```
systemctl status luckfox-bt-attach.service bluetooth.service --no-pager
bluetoothctl list
bluetoothctl show
```

При ручном обнаружении в `bluetoothctl` выполняется `scan on`. После обнаружения известного тестового устройства последовательно выполняются `info АДРЕС`, `scan off` и `quit`. Наличие записи в списке ранее обнаруженных устройств не подтверждает текущий радиообмен.

Для BLE задаются сервисы, UUID характеристик, формат данных, способ получения данных посредством чтения или уведомлений и порядок восстановления соединения. Функциональная готовность подтверждается обменом с целевым устройством. Использование `ttyS1` прикладной программой и остановка attach-службы с целью освобождения UART не допускаются.

В рассматриваемой конфигурации отдельная platform-запись `rfkill` может указывать на блокировку при работоспособном `hci0`. Состояние контроллера оценивается по `bluetoothctl show` и результатам обмена. Массовое снятие блокировок не является обязательной операцией настройки.

8 Экран и сенсор

8.1 Распределение доступа к графической подсистеме

В проверенном образе кадровый буфер framebuffer имеет разрешение 720 × 720 пикселей, глубину 32 бита на пиксель, формат XRGB8888 и шаг строки 2880 байт. Физический интерфейс матрицы RGB666 не изменяет указанный формат представления данных для приложения.

```
fbset
cat /sys/class/graphics/fb0/virtual_size
cat /sys/class/graphics/fb0/bits_per_pixel
cat /sys/class/graphics/fb0/stride
```

Для графического приложения рекомендуется использовать поддерживаемую образом конфигурацию Qt и соответствующее окружение. Перед ручным запуском графического интерфейса на контроллере выполняется инициализация окружения:

```
. /etc/profile.d/qt_env.sh
```

Затем запустить собранное приложение с установленными в образе плагинами.

Перед обращением к экрану устанавливается состав процессов, выполняющих отрисовку: штатная визуализация, `sysutil40`, `test40` и среда исполнения ПЛК. Для ручной проверки допускается остановка `sysutil.service` после регистрации её исходного состояния с последующим восстановлением. Окончательная конфигурация проекта должна исключать одновременную отрисовку конкурирующими приложениями.

8.2 Непосредственное обращение к кадровому буферу

При непосредственной отрисовке из приложения на C/C++ параметры framebuffer считываются посредством `FBIOGET_FSCREENINFO` и `FBIOGET_VSCREENINFO`. Проверяются разрядность, компоненты цвета, `line_length`, размеры, смещения и допустимая длина отображения памяти. Определение размера `mmap` только как произведения ширины, высоты и четырёх байтов без контроля границ памяти не допускается.

Для подтверждённого в исходном образе формата little-endian последовательность байтов пикселя имеет вид B, G, R, X. Красному цвету соответствует `00 00 FF 00`. Смещение пикселя относительно начала отображённой области определяется выражением:

```
offset = (y + yoffset) * line_length + (x + xoffset) * bytes_per_pixel
```

Операции записи ограничиваются доступным диапазоном отображённой памяти. Полный кадр в рассматриваемом режиме занимает 2 073 600 байт, что составляет около 1,98 МиБ. Двойная буферизация в памяти приложения требует около 3,96 МиБ без учёта дополнительных буферов.

Наличие `/dev/fb0` не подтверждает поддержку аппаратного переключения страниц page flip или отсутствие разрывов изображения. Сохранение и восстановление содержимого кадрового буфера не обеспечивают восстановления внутреннего состояния стороннего графического приложения.

8.3 Сенсор GT911

Доступ к сенсору осуществляется через `/dev/input/touchscreen0`. Использование фиксированного имени `/dev/input/event0` нежелательно вследствие возможного изменения номера устройства. Контроллер GT911 подключён к I²C3 по адресу `0x5d`; прикладной доступ выполняется через подсистему input без непосредственного обращения к его регистрам.

```
ls -l /dev/input/touchscreen0
cat /proc/bus/input/devices
evtest /dev/input/touchscreen0
```

Для Qt применяется штатная конфигурация подсистемы ввода, установленная окружением образа. Собственная реализация клиента evdev должна обрабатывать `struct input_event`, границы кадров `SYN_REPORT` и объявленный устройством протокол мультитач. Диапазоны координат определяются посредством `EVIOCGABS` и могут отличаться от диапазонов экранных координат.

Поворот и масштабирование координат сенсора согласуются с ориентацией изображения. Проверка включает центр, углы, края, непрерывные траектории и одновременные касания, если они предусмотрены проектом. При получении `SYN_DROPPED` состояние восстанавливается в соответствии с правилами evdev для исключения ошибочного сохранения признака активного касания.

Применение `evtest --grab` при работающем графическом приложении допускается только при обоснованной необходимости. Сканирование всей шины I²C и произвольные вызовы `i2cset` не используются для проверки сенсора, уже обслуживаемого драйвером.

9 Подсветка и индикатор work

9.1 Яркость подсветки

Предельное значение яркости считывается из `max_brightness`. Заданный процент преобразуется в целое значение шкалы драйвера и записывается в `brightness`. Значение 255 относится к проверенному образу и не рассматривается как универсальная константа.

Пример установки 60 процентов. Выполнять при согласованном владельце.

```
bl=/sys/class/backlight/backlight
max=$(cat "$bl/max_brightness") || exit 1
value=$((max * 60 / 100))
printf '%s\n' "$value" > "$bl/brightness"
```

В приложении на C/C++ управление выполняется открытием соответствующего атрибута sysfs, записью десятичного числа с символом перевода строки и проверкой результата. Перед следующей операцией файл открывается повторно либо его позиция возвращается в начало. Допустимость значения в диапазоне 0…100 процентов проверяется до обращения к драйверу.

При программном гашении экрана предусматриваются восстановление яркости по действию пользователя и сервисная процедура восстановления изображения. Во время стендовой проверки исходная яркость регистрируется и восстанавливается по её завершении.

9.2 Индикатор work

Индикатор представлен интерфейсом LED class. Активный механизм автоматического управления определяется по значению, заключённому в квадратные скобки в файле `trigger`. Для непосредственного управления включением и выключением устанавливается значение `none`:

```
led=/sys/class/leds/work
cat "$led/trigger"
printf 'none\n' > "$led/trigger"
cat "$led/max_brightness" > "$led/brightness"
```

Для выключения:

```
printf '0\n' > "$led/brightness"
```

Функциональное назначение индикации и единственный владелец LED определяются проектом. Временная диагностика предусматривает сохранение и последующее восстановление значений trigger и яркости. Непосредственный экспорт GPIO118 не допускается, поскольку GPIO3_C6 занят драйвером LED. Возможность визуального контроля индикатора зависит от конструкции корпуса.

10 Звуковые сигналы и микрофон

10.1 Формирование звукового сигнала

Отдельный зуммер с управлением через GPIO в проверенной конфигурации не используется. Функция beeper реализована службой `agava-console-bell.service` с воспроизведением через PulseAudio и динамик. Проверка на контроллере выполняется следующими командами:

```
systemctl status agava-console-bell.service --no-pager
printf '\007' > /dev/tty5
```

Символ BEL передаётся в локальный виртуальный терминал. Выполнение `printf '\a'` в сеансе SSH может инициировать сигнал на компьютере инженера. Прикладное звуковое оповещение реализуется через согласованный локальный механизм terminal bell либо воспроизведение средствами штатной аудиоподсистемы. Частота сигналов ограничивается; их формирование выносится за пределы критического цикла.

10.2 Воспроизведение

Звуковая карта ALSA с номером 0 имеет имя `rv1106-acodec`. В штатной конфигурации кодек может использоваться PulseAudio. Для ручной проверки применяется устройство `default`:

```
cat /proc/asound/cards
aplay -l
pactl list short sinks
aplay -D default /usr/share/sounds/piano.wav
```

Приложение может обращаться к поддерживаемому образом библиотечному аудиоинтерфейсу: ALSA через настроенное устройство default либо средствам Qt при наличии соответствующего модуля. Непосредственное открытие `hw:0,0` в обход PulseAudio может завершаться ошибкой `Device or resource busy`.

Диагностика воспроизведения начинается с проверки устройства вывода sink, состояния отключения звука mute и уровня громкости. Непосредственная установка максимальной громкости не допускается. Два канала PCM не свидетельствуют о наличии двух физических динамиков. Изменение общей громкости выполняется в соответствии с пользовательскими настройками и требованиями проекта.

10.3 Запись микрофона

Перед записью проверяется выбор аппаратного источника source. Использование monitor-источника звукового выхода не подтверждает работу микрофона:

```
arecord -l
pactl list short sources
arecord -D default -f S16_LE -r 16000 -c 1 -d 5 /tmp/myproject-microphone.wav
aplay -D default /tmp/myproject-microphone.wav
```

В течение пятисекундного интервала записи произносится контрольная фраза. Приведённый формат соответствует исходной методике; рабочие форматы приложения подлежат проверке на целевом аудиотракте. Программная реализация должна обрабатывать переполнение и недогрузку буферов, прекращение работы аудиосервера и недоступность устройства.

11 АЦП и температура

11.1 Чтение SARADC

Для проверенного устройства `iio:device0` доступны атрибуты `in_voltage0_raw`, `in_voltage1_raw` и `in_voltage_scale`. Масштабный коэффициент scale в рассматриваемом драйвере выражен в мВ на отсчёт. Напряжение на входе АЦП рассчитывается следующим образом:

```
U_ADC_mV = raw * scale
U_ADC_V  = raw * scale / 1000
```

Чтение и преобразование результата на контроллере выполняются следующим образом:

```
adc=/sys/bus/iio/devices/iio:device0
cat "$adc/name"
raw=$(cat "$adc/in_voltage0_raw") || exit 1
scale=$(cat "$adc/in_voltage_scale") || exit 1
awk -v raw="$raw" -v scale="$scale" \
   'BEGIN {printf "ADC0 pin: %.3f mV\n", raw * scale}'
```

Полученное значение характеризует напряжение на входе АЦП. Его интерпретация как напряжения питания контроллера или внешнего датчика требует подтверждения электрической схемой. Схемой определяются назначение каналов, параметры делителей, смещение, допустимый диапазон и точки подключения. При заданных коэффициенте восстановления K и смещении B внешний сигнал рассчитывается по выражению `U_external = K * U_ADC + B` с согласованием единиц измерения.

В C/C++ текстовые атрибуты считываются с контролем корректности преобразования числа, допустимого диапазона и ошибок ввода-вывода. Разбор десятичного scale должен обеспечивать использование точки в качестве разделителя независимо от локали. Частота опроса и параметры фильтрации определяются требованиями процесса. Наличие атрибута raw не подтверждает поддержку буферизованного режима IIO и заданной частоты дискретизации.

При изменении состава IIO-устройств требуемый ресурс идентифицируется по имени и аппаратной привязке с последующей проверкой набора атрибутов. Использование фиксированного имени `iio:device0` без идентификации создаёт риск обращения к другому устройству.

11.2 Температура процессора

```
cat /sys/class/thermal/thermal_zone0/type
awk '{printf "SoC: %.1f C\n", $1/1000}' \
   /sys/class/thermal/thermal_zone0/temp
```

Измеренная температура SoC характеризует систему на кристалле и не эквивалентна температуре окружающего воздуха или внешнего технологического объекта. На другом образе требуется подтверждение назначения температурной зоны thermal zone. Пороги предупреждения и аварии, а также гистерезис определяются требованиями изделия и проекта. Произвольное назначение температурных пределов вместо установленных аппаратных допусков не допускается.

12 Системное время и RTC

Временные метки событий формируются по системным часам Linux. Измерение интервалов, контроль предельных сроков операций и периода опроса выполняются по монотонным часам, например `CLOCK_MONOTONIC`. Коррекция календарного времени посредством NTP не должна изменять установленный интервал ожидания оборудования.

Часы реального времени RTC, представленные устройством `/dev/rtc0`, сохраняют календарное время при наличии предусмотренного резервного питания. В исходной методике используется представление времени в UTC:

```
date -u
timedatectl status
hwclock -r -u -f /dev/rtc0
```

Система должна иметь согласованный порядок выбора источника времени, работы NTP, загрузки времени из RTC и его сохранения в RTC. Периодическая запись RTC из прикладного цикла не предусматривается. Команда `hwclock -w -u -f /dev/rtc0` применяется при настройке при условии достоверного системного UTC и согласования с механизмом синхронизации образа.

После запуска при отсутствии сети оценивается достоверность календарного времени. События с неподтверждёнными временными метками должны содержать соответствующий признак. Допускается регистрация признака «время не синхронизировано» совместно с событием и идентификатором текущей загрузки.

Проверка сохранения времени RTC проводится на стенде после отключения всех источников питания на срок не менее 60 секунд. На период проверки NTP-синхронизация отключается, а показания сопоставляются с внешним эталоном. После завершения проверки исходное состояние NTP восстанавливается. Корректное время, установленное сетевой синхронизацией, не подтверждает исправность резервного питания RTC.

13 Память и хранение данных

13.1 Распределение оперативной памяти

Часть установленных 256 МБ оперативной памяти зарезервирована для ядра, SoC и мультимедийной подсистемы. В исходной конфигурации значение `MemTotal` составляет приблизительно 175816 КиБ, или около 172 МиБ. Указанная величина характеризует конкретный образ и не определяет гарантированный объём памяти, доступный приложению.

```
grep -E 'MemTotal|MemAvailable|CmaTotal|CmaFree|SwapTotal' /proc/meminfo
free
```

Расчёт потребления памяти учитывает графический интерфейс, изображения, шрифты, сетевые буферы, кэши, среду исполнения ПЛК и NPU. Для очередей, журналов и числа одновременных клиентов устанавливаются ограничения. Отсутствие утечек памяти проверяется длительным рабочим прогоном; кратковременный запуск не обеспечивает достаточного подтверждения.

13.2 Постоянное хранение данных

По результатам исходной инвентаризации доступная ёмкость eMMC составляет около 7,1 ГиБ. Корневая файловая система размещена на `mmcblk0p2`, а `/opt` — на `mmcblk0p3`. Прикладное хранение данных осуществляется через файлы смонтированной файловой системы без записи в сырые разделы и область vendor storage.

Для сохранения критически значимых настроек рекомендуется следующая последовательность контролируемого обновления файла:

1. Создать временный файл в том же каталоге и файловой системе, что и окончательный.
2. Полностью записать и проверить содержимое, обработать ошибки записи.

3. Выполнить `fsync` файла и проверить результат.
4. Переименовать временный файл в окончательное имя.
5. Выполнить `fsync` каталога и обработать результат с учётом поддержки файловой системы.

Предыдущая корректная конфигурация и версия её формата сохраняются для восстановления. Приведённая последовательность снижает вероятность частичной записи. Устойчивость к потере питания подтверждается испытанием на целевом образе и используемом носителе.

Хранение телеметрии предусматривает буферизацию, ротацию, ограничение объёма и резерв свободного пространства. Частота записи на eMMC определяется требованиями к данным; сохранение каждого изменения короткого цикла без соответствующего требования не рекомендуется. Реакции на `ENOSPC`, ошибки ввода-вывода и переход файловой системы в режим read-only определяются на этапе проектирования.

Область vendor storage может содержать индивидуальные данные платы, включая MAC-адрес. Её копирование между экземплярами и использование в качестве прикладного хранилища не допускаются. Неизменность MAC-адреса не гарантирует постоянства адреса, назначаемого DHCP-сервером.

Стендовые нагрузочные проверки RAM и CPU, сброс страничного кэша, испытания eMMC через сырое устройство и исправляющий `fsck` на смонтированной файловой системе не включаются в рабочий цикл приложения.

14 Watchdog и ресурсы SoC

14.1 Контроль работоспособности

В проверенном образе установлены `CONFIG_WATCHDOG=y` и `CONFIG_DW_WATCHDOG=y`, однако устройства `/dev/watchdog*` отсутствуют, а `CONFIG_WATCHDOG_SYSFS` отключён. Следовательно, аппаратное восстановление при зависании не может рассматриваться как функционально обеспеченная возможность данной конфигурации.

Использование аппаратного watchdog требует отдельного согласования конфигурации BSP: узла DT, тактирования и сброса clock/reset, драйвера, параметров timeout и nowayout, порядка остановки и владельца таймера. Функциональная проверка включает фактический сброс, измерение времени срабатывания, установление причины перезапуска и контроль состояний выходов. Управление таймером закрепляется за одним владельцем: приложением, systemd либо отдельным демоном.

Параметр `Restart=on-failure` обеспечивает перезапуск процесса, завершившегося с ошибкой, при соблюдении условий systemd, но сам по себе не выявляет зависание выполняющегося процесса. Применение `WatchdogSec=` требует реализации протокола уведомлений приложением и совместимой конфигурации systemd. Данный механизм не эквивалентен аппаратному watchdog.

Контроль работоспособности должен подтверждать выполнение значимых операций приложения. Безусловная периодическая отправка уведомлений отдельным потоком не является достаточным подтверждением работоспособности основной логики. При зависании приложения ранее установленные состояния релейных выходов могут сохраняться.

14.2 Генератор случайных чисел

Устройство `/dev/hwrng` предоставляет доступ к аппаратному генератору случайных чисел. Различие двух выборок, рассмотренное в исходной методике, подтверждает возможность чтения, но не криптографическую пригодность данных. Для прикладной генерации случайных данных предпочтительно применение системного генератора через `getrandom` с обработкой ожидания готовности и ошибок с учётом требований проекта и поддержки libc.

Непосредственное чтение аппаратного RNG не рассматривается как достаточное подтверждение пригодности данных для формирования криптографических ключей. Механизм инициализации системного источника энтропии подлежит проверке в целевом образе.

14.3 Нейронный процессор NPU

Устройство `/dev/rknpu` свидетельствует о наличии интерфейса драйвера. Выполнение вычислений требует совместимых версий драйвера, RKNN runtime, средств подготовки модели и модели, предназначенной для RV1106. Версии, API и состав библиотек в исходном файле не определены.

Включению NPU в прикладную функцию предшествует проверка тестовой модели, предварительной обработки входных данных, формата и размеров тензоров, ожидаемого результата, времени выполнения и потребления памяти. Нулевая загрузка NPU при отсутствии вычислительного приложения соответствует ожидаемому состоянию. Наличие файла устройства не подтверждает корректность вычислений.

15 Развёртывание приложения

15.1 Передача и ручной запуск

Развёртывание начинается с создания предусмотренных проектом каталогов в доступном для записи `/opt` и передачи исполняемого файла по SFTP. Первоначальный запуск выполняется вручную с проверкой зависимостей и доступа к оборудованию.

На контроллере после проверки /opt.

```
mkdir -p /opt/myproject/bin /opt/myproject/config /opt/myproject/data
```

Передача файла с компьютера инженера осуществляется следующим образом:

```
sftp root@192.168.7.1
```

В приглашении sftp:

```
put myapp /opt/myproject/bin/myapp
bye
```

После исключения одновременного доступа других владельцев на контроллере выполняются следующие команды:

```
chmod 0755 /opt/myproject/bin/myapp
/opt/myproject/bin/myapp
```

Сообщение об отсутствии ELF-интерпретатора или библиотеки требует проверки соответствия SDK и состава целевого образа. Замещение системных библиотек произвольно выбранными ARM-библиотеками не допускается. Запуск графического приложения предусматривает инициализацию окружения посредством `/etc/profile.d/qt_env.sh`.