

Содержание

Использование периферии Q7_KOMDIV_MK	3
ETH	3
CAN	4
Подключение	4
Настройка	4
Приём	5
Отправка	5
Остановка	5
Проверка состояния	5
UART	6
Подключение	6
Настройка	6
Приём	6
Отправка	6
I2C	6
Просмотр шин	7
Поиск устройств	7
Чтение регистра	7
Запись регистра	7
Чтение нескольких байтов	7
Подключение VL53L0X	7
Чтение VL53L0X	8
SPI	9
Подключение	9
Конфигурация	10
Определение SPI Flash	10
SATA	11
RTC	12

Использование периферии Q7_KOMDIV_MK

На странице приведены примеры настройки и проверки периферийных интерфейсов модуля Q7_KOMDIV_MK.

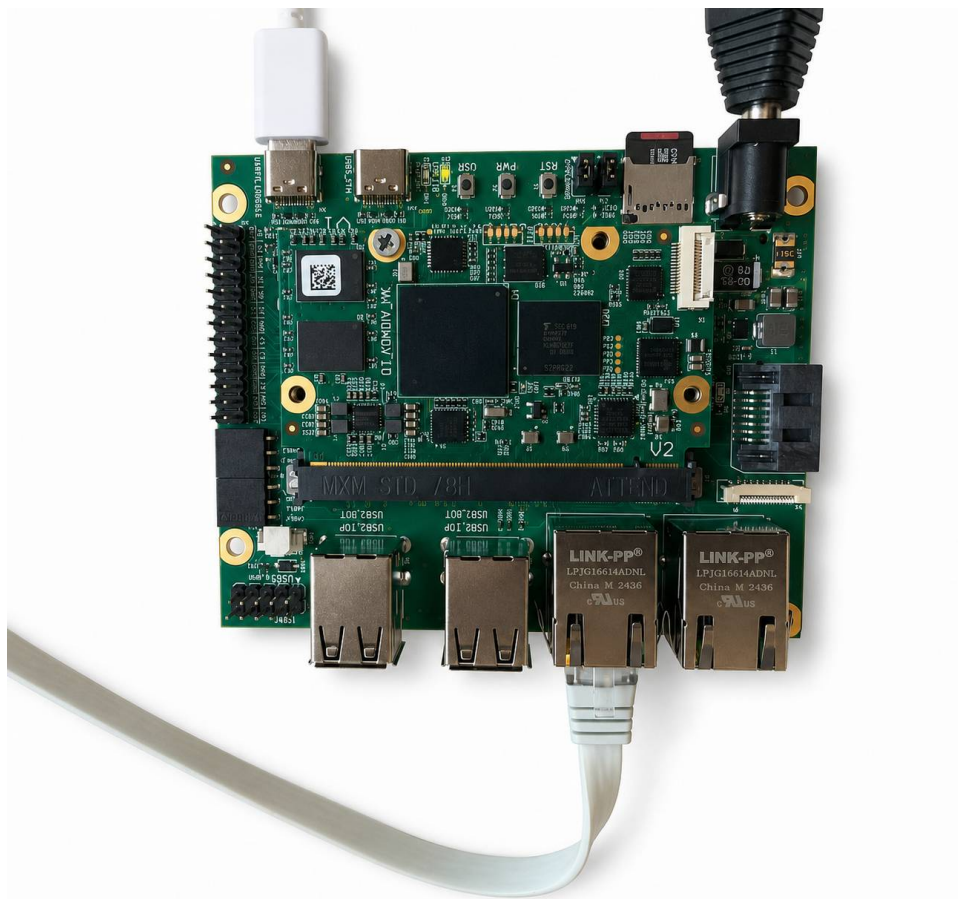
- CAN
- UART;
- I2C;
- SPI;
- SATA;
- RTC.

Скрипты для тестирования доступны в [репозитории test-suite](#).

```
git clone  
https://gitlab.inmys.online/srisa-public/k5500vk018-test-suite.git
```

ETH

ETH на краевом разъеме Q7 по стандарту работает на RJ45-розетке рядом с USB:



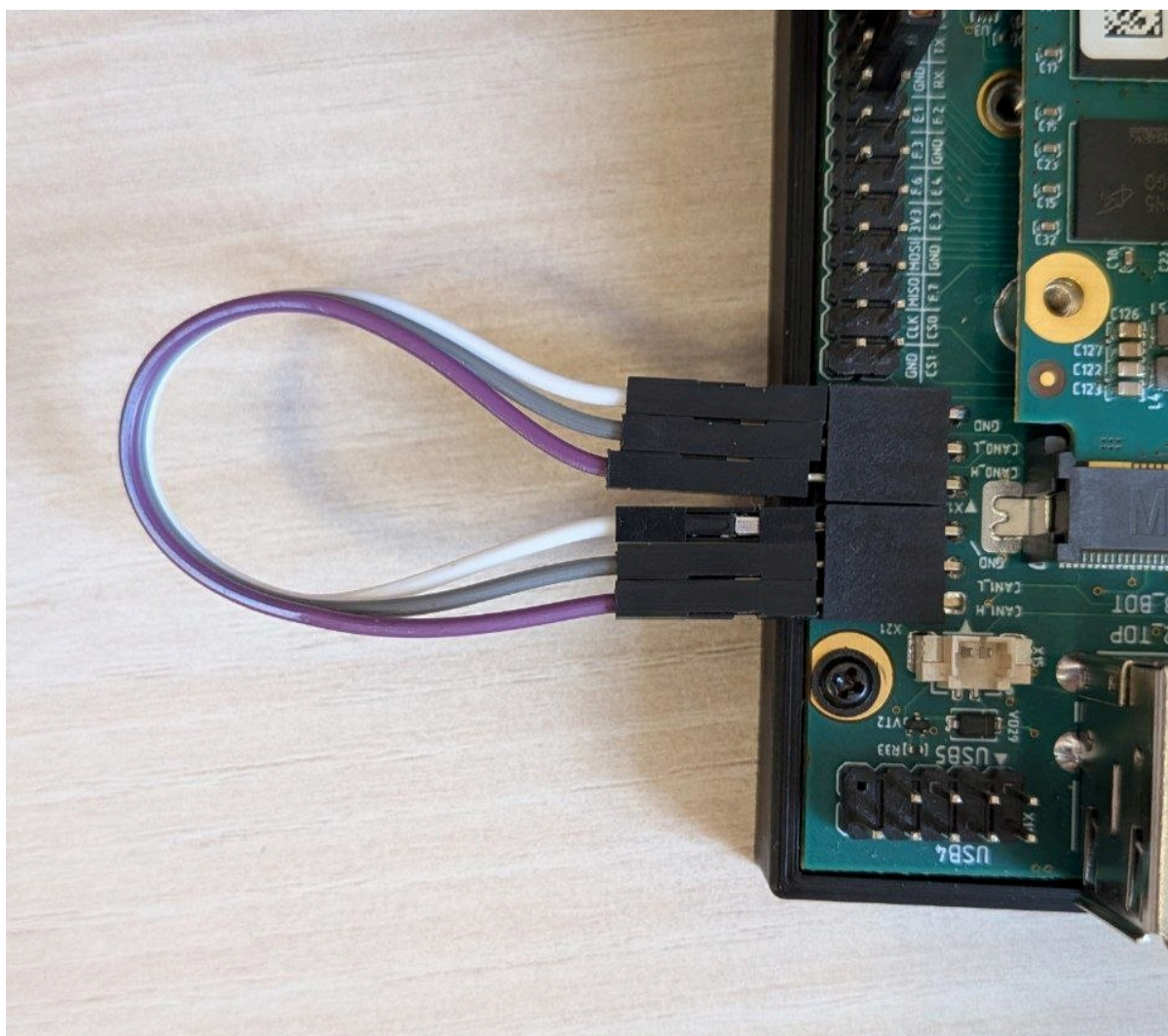
CAN

Пример CAN loopback:

Подключение

Соединить CAN интерфейсы следующим образом:

```
CAN0_H --- CAN1_H  
CAN0_L --- CAN1_L  
GND      --- GND
```



Настройка

Перед изменением скорости интерфейсы необходимо выключить:

```
ip link set can0 down  
ip link set can0 type can bitrate 1000000 restart-ms 100  
ip link set can0 up
```

```
ip link set can1 down  
ip link set can1 type can bitrate 1000000 restart-ms 100  
ip link set can1 up
```

Скорость на обоих интерфейсах должна совпадать.

Приём

Приём всех кадров:

```
candump can0
```

Приём только кадров с идентификатором 123:

```
candump 'can0,123:7FF'
```

Отправка

Отправка кадра через can1:

```
cansend can1 123#1122334455667788
```

Где 123 — CAN ID, после # идут данные.

Остановка

```
ip link set can0 down  
ip link set can1 down
```

Проверка состояния

```
ip -details -statistics link show can0
```

Нормальное состояние интерфейса:

```
state ERROR-ACTIVE bitrate 1000000
```

В статистике отображаются:

количество принятых пакетов — RX packets;

количество отправленных пакетов — TX packets;

ошибки приёма — RX errors;

ошибки передачи — TX errors.

UART

Подключение

Для loopback соедините:

```
TX --- RX
```

Настройка

```
stty -F /dev/ttyS1 115200 raw -echo
```

Приём

```
cat /dev/ttyS1
```

Отправка

В другом окне терминала:

```
printf 'UART test' > /dev/ttyS1
```

При loopback отправленная строка должна появиться в окне приёма.

I2C

I2C-шины по стандарту Q7 реализованы через 4х-канальный мультиплексор PCA9546APW.

Описание i2c-шин с точки зрения Linux:

```
ls -la /sys/class/i2c-adapter*
```

```
i2c-0 -> ../../devices/platform/lb400130.i2c/i2c-0 - SoC's I2C-Master
i2c-1 -> ../../devices/platform/lb400130.i2c/i2c-0/i2c-1 - I2C_SYS.
Шина на модуле, куда подключены PMIC(0x18), i2c eeprom, i2c-mux(0x74), rtc
(0x68)
i2c-2 -> ../../devices/platform/lb400130.i2c/i2c-0/i2c-2 - I2C0 по
стандарту Q7
i2c-3 -> ../../devices/platform/lb400130.i2c/i2c-0/i2c-3 - SMB_I2C по
стандарту Q7
i2c-4 -> ../../devices/platform/lb400130.i2c/i2c-0/i2c-4 - Not
Connected
```

Пример работы с VL53L0X, подключенном к I2C0 по Q7:

Просмотр шин

```
i2cdetect -l  
ls /dev/i2c-*
```

Поиск устройств

Для шины i2c-2:

```
i2cdetect -y 2
```

VL53L0X обычно доступен по адресу 0x29.

Чтение регистра

```
i2cget -y 2 0x29 0xC0
```

Где:

- 2 — номер I²C-шины;
- 0x29 — адрес устройства;
- 0xC0 — адрес регистра.

Запись регистра

```
i2cset -y 2 0x29 0x00 0x01
```

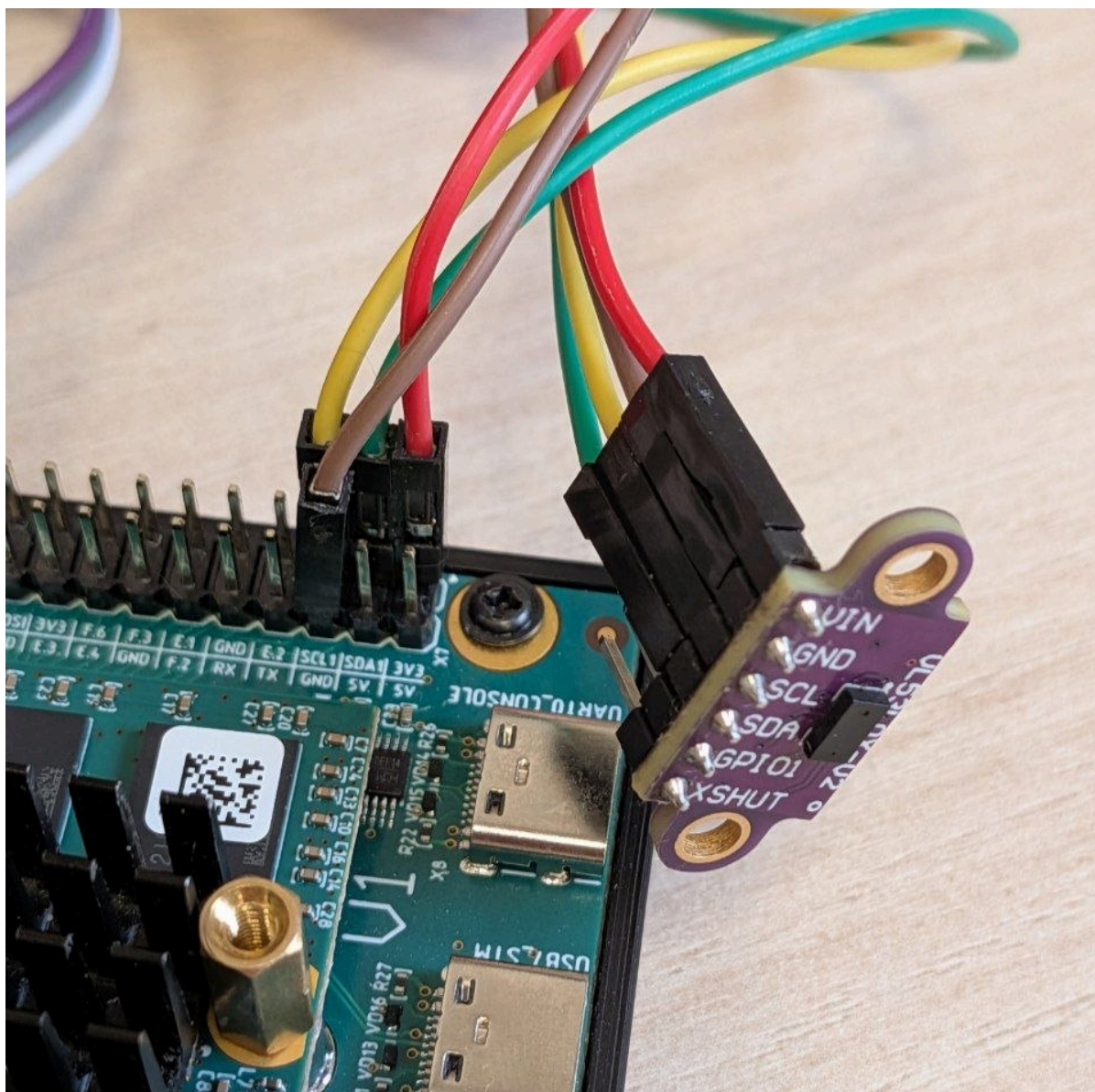
Чтение нескольких байтов

```
i2ctransfer -y 2 w1@0x29 0x14 r12
```

Команда указывает датчику начать чтение с регистра 0x14, после чего получает 12 последовательных байт данных.

В качестве примера используется лазерный дальномер VL53L0X.

Подключение VL53L0X



Чтение VL53L0X

```
chmod +x i2c-test.sh
./i2c-test.sh 2 20
```

Где:

- 2 — номер I²C-шины;
- 20 — количество измерений.

Вывод скрипта:

```
Distance 1: 87 mm
Distance 2: 65 mm
Distance 3: 71 mm
Distance 4: 68 mm
Distance 5: 74 mm
Distance 6: 65 mm
```

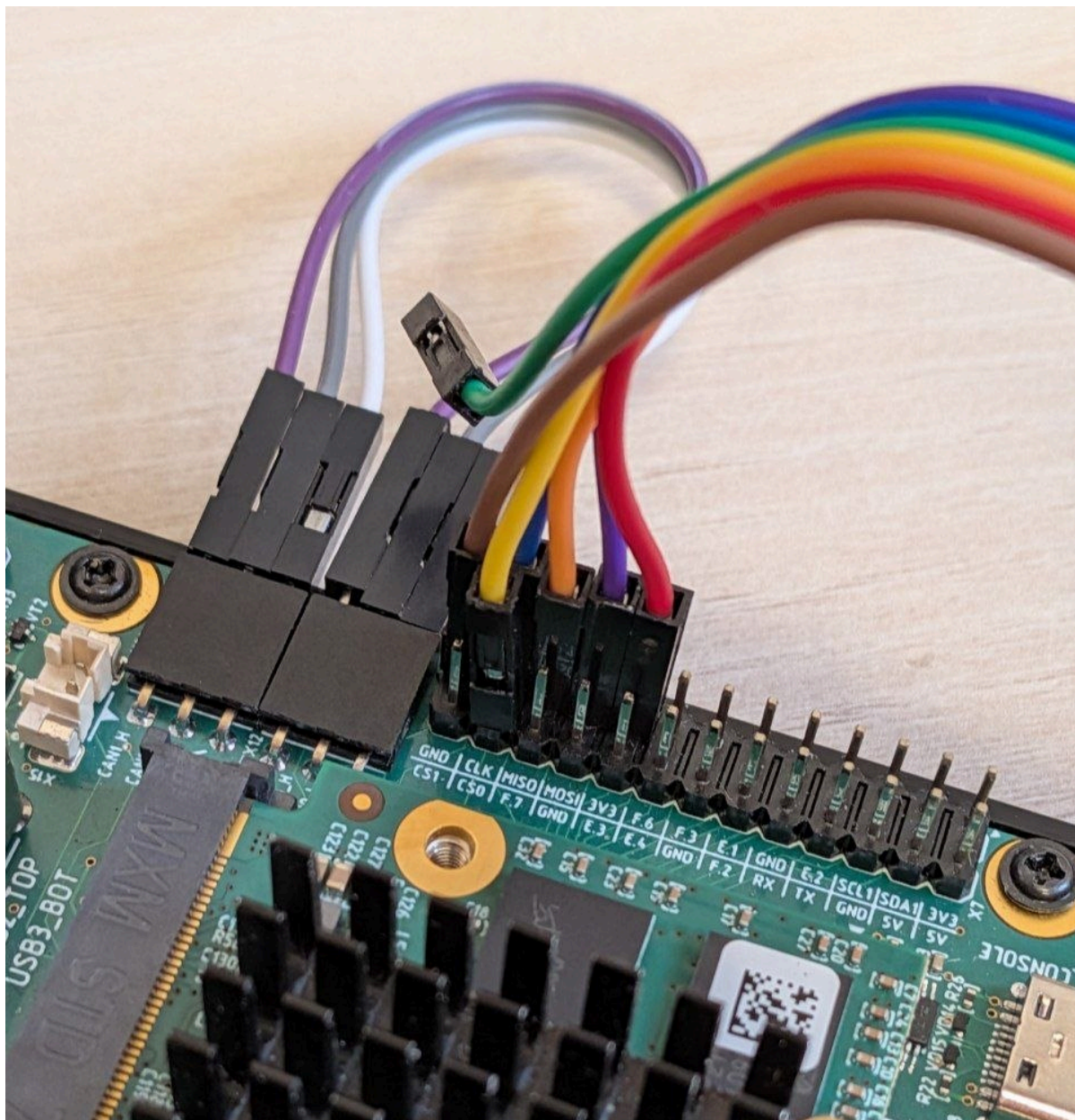
```
Distance 7: 74 mm
Distance 8: 104 mm
Distance 9: 122 mm
Distance 10: 169 mm
Distance 11: 281 mm
Distance 12: 364 mm
Distance 13: 298 mm
Distance 14: 185 mm
Distance 15: 112 mm
Distance 16: 89 mm
Distance 17: 76 mm
Distance 18: 69 mm
Distance 19: 65 mm
Distance 20: 70 mm
```

SPI

В качестве примера используется SPI Flash, подключённая к CS0. Ей соответствует устройство `/dev/spidev1.0`.

При подключении к CS1 используется устройство `/dev/spidev1.1`.

Подключение



Конфигурация

Для работы с устройствами SPI необходимо загрузить модуль ядра:

```
modprobe spidev
```

Определение SPI Flash

```
flashrom -p linux_spi:dev=/dev/spidev1.0,spispeed=1000
```

Пример успешного результата:

```
flashrom v1.5.1 on Linux 6.1.135 (mips64)
flashrom is free software, get the source code at https://flashrom.org

Found Macronix flash chip "MX66L51235F/MX25L51245G" (65536 kB, SPI) on
```

```
linux_spi.  
===  
This flash part has status UNTESTED for operations: WP  
The test status of this chip may have been updated in the latest  
development  
version of flashrom. If you are running the latest development version,  
please email a report to flashrom@flashrom.org if any of the above  
operations  
work correctly for you with this flash chip. Please include the  
flashrom log  
file for all operations you tested (see the man page for details), and  
mention  
which mainboard or programmer you tested in the subject line.  
You can also try to follow the instructions here:  
https://www.flashrom.org/contrib\_howtos/how\_to\_mark\_chip\_tested.html  
Thanks for your help!  
No operations were specified.
```

SATA

В модуле 1x SATA, который мультиплицируется между SATA0 и SATA1. По умолчанию выбран SATA1 по стандарту Q7 (на отладочной плате m2-разъем).

Выбор активного канала выполняется через GPIO с помощью скрипта `gpio.sh`.

Выбор канала 0:

```
./gpio.sh sata_select 0
```

Выбор канала 1:

```
./gpio.sh sata_select 1
```

После выбора канала проверить обнаружение накопителя можно командой:

```
cat /proc/partitions
```

Пример успешного обнаружения SATA-накопителя:

```
major minor #blocks name  
1        0      16384 ram0  
1        1      16384 ram1  
1        2      16384 ram2  
1        3      16384 ram3  
1        4      16384 ram4  
1        5      16384 ram5  
1        6      16384 ram6  
1        7      16384 ram7  
1        8      16384 ram8
```

```

1      9      16384 ram9
1     10      16384 ram10
1     11      16384 ram11
1     12      16384 ram12
1     13      16384 ram13
1     14      16384 ram14
1     15      16384 ram15
31     0        640 mtdblock0
31     1         64 mtdblock1
31     2         64 mtdblock2
31     3         64 mtdblock3
179    0     7634944 mmcblk0
179    1     1048576 mmcblk0p1
179    2     1048576 mmcblk0p2
179    3     1048576 mmcblk0p3
179    8         4096 mmcblk0boot0
179   16         4096 mmcblk0boot1
 8     16 125034840 sdb   # <-----/
 8     17  153600 sdb1  # <-----/
Обнаруженный SATA-накопитель
 8     18 124881223 sdb2  # <-----/

```

В данном примере накопитель определён как /dev/sdb и содержит два раздела.

Имя устройства может отличаться в зависимости от порядка обнаружения накопителей.

RTC

В дереве устройств определена I2C RTC.

При первом запуске системное время может быть установлено на 01.01.1970, а RTC ещё не будет содержать корректного значения.

Пример установки системного времени:

```
date -s "2026-07-29 14:48:00"
```

Запись системного времени в RTC:

```
hwclock -w
```

Просмотр текущего времени RTC:

```
hwclock -r
```

При наличии батарейки записанные дата и время сохраняются после отключения питания.