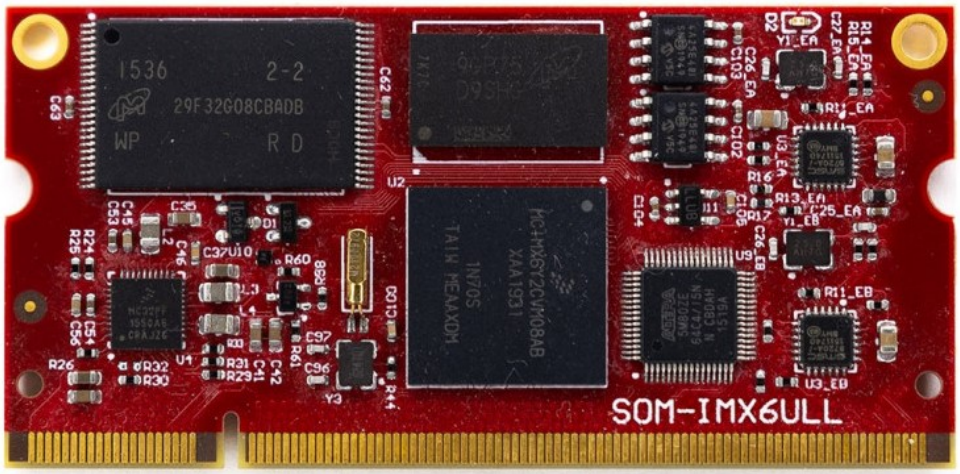
Содержание

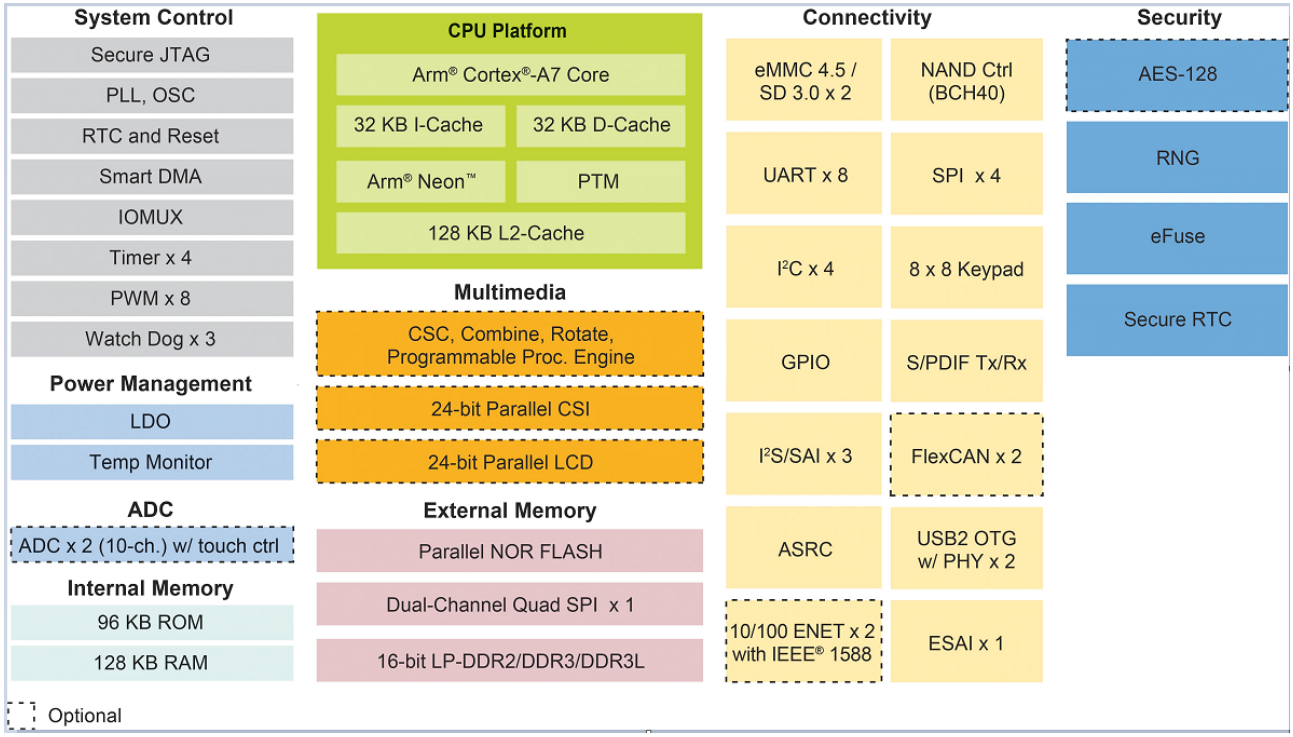
NMS-SDM-IMX6ULL v1 ds-ru 3
Структурная процессора 3
Структурная схема модуля 3
Назначение выводов модуля 4
RMII1/RGII2 выводы процессора 9
I2C интерфейс 9
NAND интерфейс 9
Мультиплексирование UART и CAN 10
Отладочная плата 10
Сборка Linux для imx6ull 11
Вступление 11
Состав bsp 11
Сборка uboot, kernal и rootfs 12
Загрузка с SD карты 12
Загрузка с NAND 13
Проверка работы CAN 13
Проверка работы UART 15
Настройки Ethernet 15

NMS-SDM-IMX6ULL v1 ds-ru

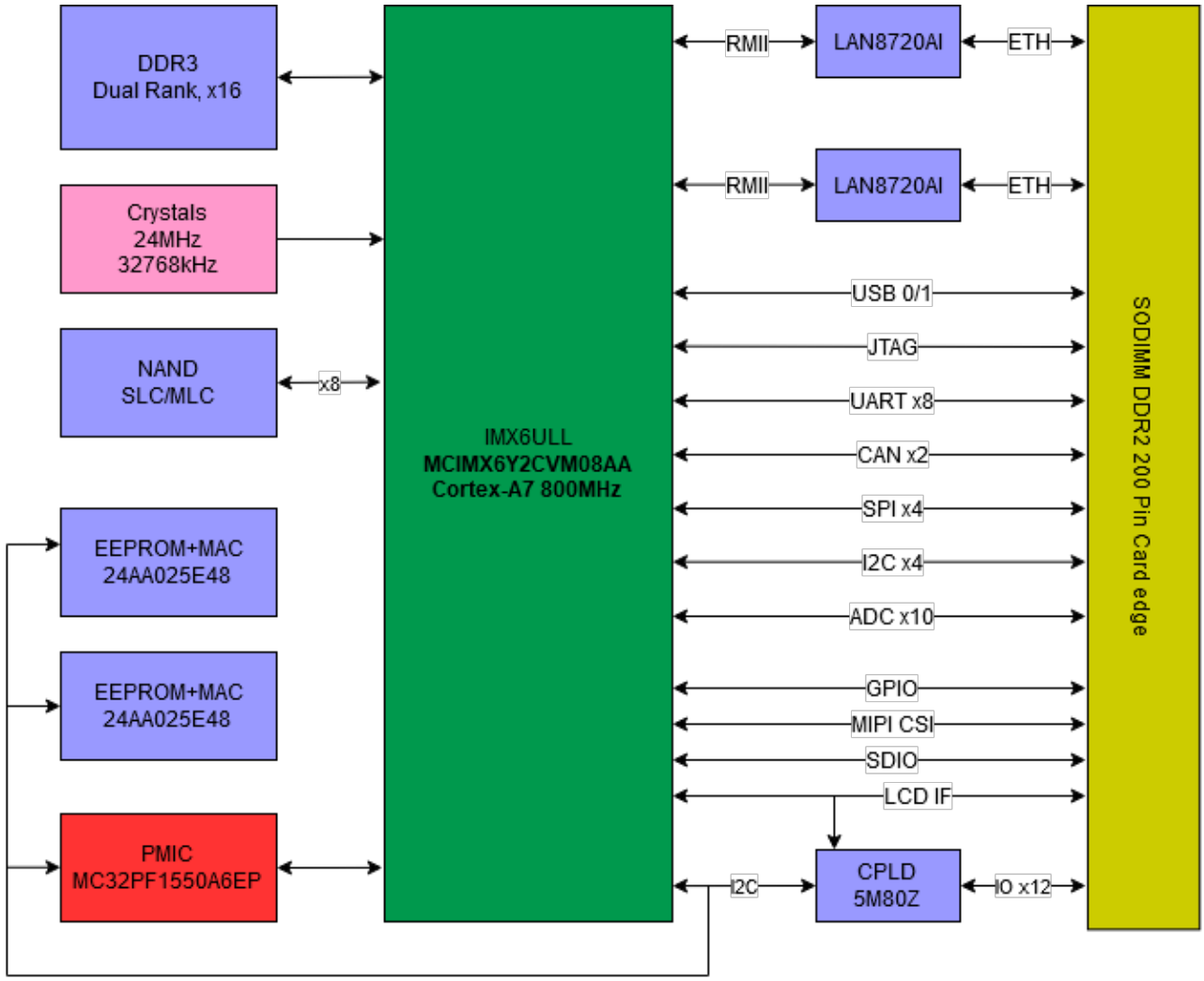
(SOM-IMX6ULL-E2)



Структурная процессора



Структурная схема модуля



Назначение выводов модуля

Вывод	Обозначение
1	VBUSIN
2	VDDCOIN
3	VBUSIN
4	PORB
5	VBUSIN
6	VDD_SNVS_3V3
7	VBUSIN
8	SYS_4V4
9	
10	SYS_4V4
11	GND
12	SYS_4V4
13	GND
14	VCC_3V3
15	GND
16	VCC_3V3

Вывод	Обозначение
17	GND
18	VCC_3V3
19	GND
20	VBAT
21	GND
22	VBAT
23	PMIC_ONREQ
24	VBAT
25	PMIC_STBY_REQ
26	NVCC_CSI
27	MX6_RESETB
28	NVCC_SD1
29	NWDOG
30	VLDO_1V8
31	ONOFF
32	VLDO2_3V3
33	ONKEY
34	VLDO3_3V3
35	GND
36	LED_CHARGE
37	I2C1_SDA
38	WDI
39	I2C1_SCL
40	NTC_THERMISTOR
41	SNVS_TAMPER9
42	USB2_DP
43	SNVS_TAMPER8
44	USB2_DN
45	SNVS_TAMPER7
46	GND
47	SNVS_TAMPER6
48	USB2_CHD_B
49	SNVS_TAMPER5
50	USB2_VBUS
51	SNVS_TAMPER4
52	GND
53	SNVS_TAMPER3
54	USB1_DP
55	SNVS_TAMPER2
56	USB1_DN
57	SNVS_TAMPER1
58	GND
59	SNVS_TAMPER0
60	USB1_CHD_B

Вывод	Обозначение
61	GND
62	USB1_VBUS
63	GPIO1_0
64	GND
65	GPIO1_1
66	GND
67	GPIO1_2
68	TRST
69	GPIO1_3
70	TDO
71	GPIO1_4
72	TDI
73	GPIO1_5
74	TCK
75	GPIO1_6
76	TMS
77	GPIO1_7
78	MOD
79	GPIO1_8
80	GND
81	GPIO1_9
82	UART1_TXD
83	GND
84	UART1_RXD
85	UART3_TXD
86	UART1_CTS
87	UART3_RXD
88	UART1_RTS
89	UART3_CTS
90	GND
91	UART3_RST
92	UART2_TXD
93	GND
94	UART2_RXD
95	UART4_TXD
96	UART2_CTS
97	UART4_RXD
98	UART2_RTS
99	GND
100	GND
101	UART5_TXD
102	CLK1_N
103	UART5_RXD
104	CLK1_P

Вывод	Обозначение
105	GND
106	NC
107	NC
108	NC
109	LCD_RESET
110	LCD_ENABLE
111	LCD_CLK
112	LCD_VSYNC
113	LCD_HSYNC
114	LCD_DATA_0
115	LCD_DATA_1
116	LCD_DATA_2
117	LCD_DATA_3
118	LCD_DATA_4
119	LCD_DATA_5
120	LCD_DATA_6
121	LCD_DATA_7
122	LCD_DATA_8
123	LCD_DATA_9
124	LCD_DATA_10
125	LCD_DATA_11
126	LCD_DATA_12
127	LCD_DATA_13
128	LCD_DATA_14
129	LCD_DATA_15
130	LCD_DATA_16
131	LCD_DATA_17
132	LCD_DATA_18
133	LCD_DATA_19
134	LCD_DATA_20
135	LCD_DATA_21
136	LCD_DATA_22
137	LCD_DATA_23
138	GND
139	GND
140	CPLD_IO_10
141	CPLD_TMS
142	CPLD_IO_9
143	CPLD_TDI
144	CPLD_IO_8
145	CPLD_TCK
146	CPLD_IO_7
147	CPLD_TDO
148	CPLD_IO_6

Вывод	Обозначение
149	GND
150	CPLD_IO_5
151	CSI_MCLK
152	CPLD_IO_4
153	CSI_PIXCLK
154	CPLD_IO_3
155	CSI_VSYNC
156	CPLD_IO_2
157	CSI_HSYNC
158	CPLD_IO_1
159	NC
160	CPLD_IO_0
161	CSI_D7
162	NC
163	CSI_D6
164	BOOT_0
165	CSI_D5
166	BOOT_1
167	CSI_D4
168	GND
169	CSI_D3
170	SD_CLK
171	CSI_D2
172	SD_CMD
173	CSI_D1
174	SD_D3
175	CSI_D0
176	SD_D2
177	NC
178	SD_D1
179	NC
180	SD_D0
181	GND
182	GND
183	LED_SPD_1
184	LED_SPD_0
185	LED_LINK_1
186	LED_LINK_0
187	GND
188	GND
189	ETH_0_TX_P
190	ETH_1_TX_P
191	ETH_0_TX_N
192	ETH_1_TX_N

Вывод	Обозначение
193	GND
194	GND
195	ETH_1_RX_P
196	ETH_0_RX_P
197	ETH_1_RX_N
198	ETH_0_RX_N
199	GND
200	GND

RMII1/RGII2 выводы процессора

Вывод	Режим
E17	ENET1_RDATA1
F16	ENET1_RDATA0
D15	ENET1_RX_ER
E16	ENET1_RX_EN
E14	ENET1_TDATA1
E15	ENET1_TDATA0
F14	ENET1_REF_CLK
F15	ENET1_TX_EN
C16	ENET2_RDATA1
C17	ENET2_RDATA0
D16	ENET2_RX_ER
B17	ENET2_RX_EN
A16	ENET2_TDATA1
A15	ENET2_TDATA1
D17	ENET2_REF_CLK
B15	ENET2_TX_EN
K17	ENET1_MDIO
L16	ENET1_MDC

I2C интерфейс

Вывод	Режим
G17	I2C_SCL
G16	I2C_SDA

NAND интерфейс

Вывод	Режим
E6	NAND_DQS
A3	NAND_READY
A4	NAND CLE
D5	NAND_WP
C8	NAND_WE
D8	NAND_RE

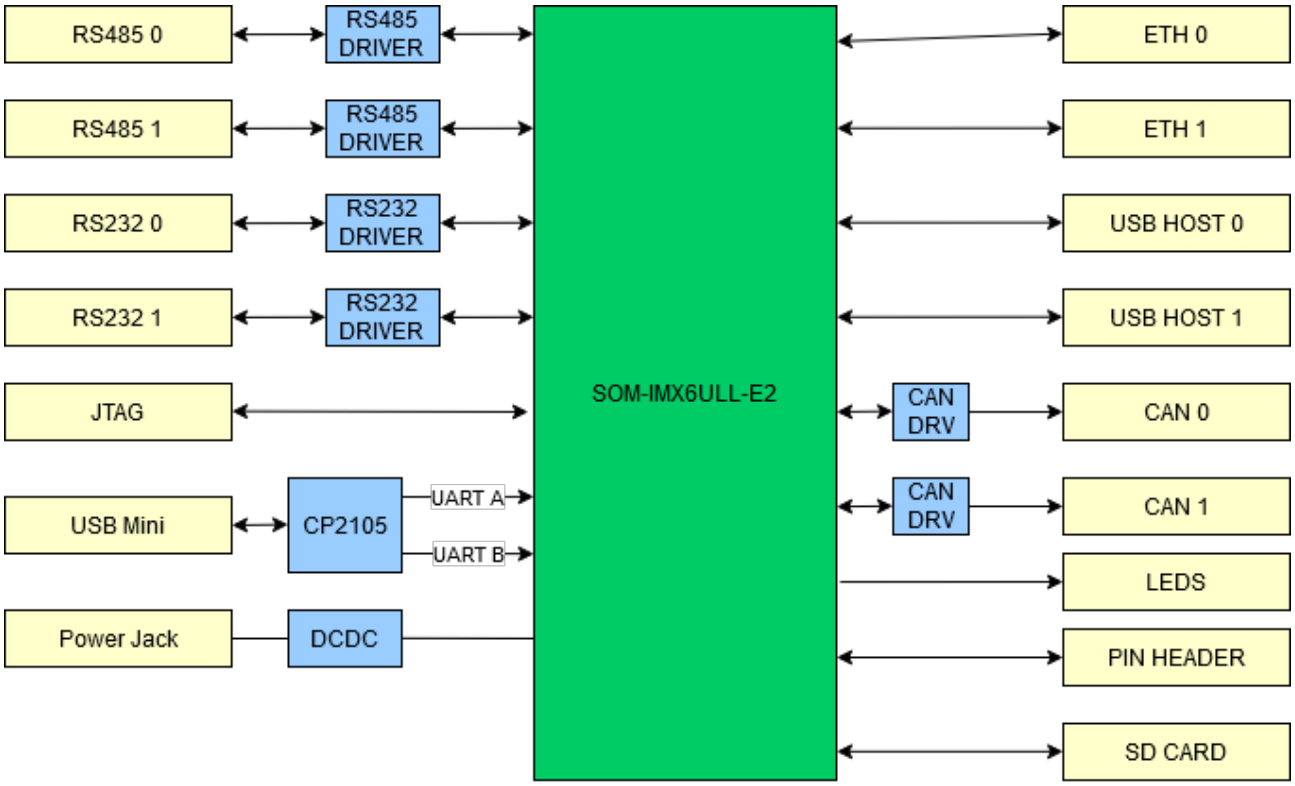
Вывод	Режим
B4	NAND_ALE
A5	NAND_D7
A6	NAND_D6
B6	NAND_D5
C6	NAND_D4
D6	NAND_D3
A7	NAND_D2
B7	NAND_D1
D7	NAND_D0
B5	NAND_CS1
C5	NAND_CS0

Мультиплексирование UART и CAN

Вариант задействования 8-ми портов UART и 2 порта CAN

Контакт разъема	Вывод процессора	Обозначение
84	K16	UART1_RX
82	K14	UART1_TX
94	J16	UART2_RX
92	J17	UART2_TX
87	H16	UART3_RX
85	H17	UART3_TX
110	B8	UART4_RX
111	A8	UART4_TX
130	G13	UART5_RX
101	F17	UART5_TX
153	E5	UART6_RX
151	F5	UART6_TX
131	B13	UART7_RX
130	C13	UART7_TX
135	B14	UART8_RX
134	C14	UART8_TX
123	A11	CAN1_RX
122	B11	CAN1_TX
125	D12	CAN2_RX
124	E12	CAN2_TX

Отладочная плата



Сборка Linux для imx6ull

Вступление

В качестве системы сборки для платы imx6_ull используется buildroot. Данная система сборки позваляет гибко собрать дистрибутив Embedded Linux для многих платформ. Официальный сайт системы сборки расположен по адресу https://buildroot.org/ . Для сборки проекта Вам понадобится ПК с дистрибутивом Linux (рекомендуем установить Ubuntu 18.04 или старше) или виртуальную машину с дистрибутивом Linux. Все дальнейшие действия по распоковки и сборки проекта будут производится на ПК с дистрибутивом Ubuntu 18.04.

Состав bsp

В домашней папке пользователя создайте папку **inmys_board**. Скачайте архив проекта (файл

imx6ull.tar.gz

) в папку inmys_board. Распакуйте архив imx6_ull.tar.gz в данную папку. После распаковки должна появится папка **imx6_ull**. Перейдите к папке imx6_ull (cd ~/inmys_board/imx6_ull/). Состав BSP:

- Архив **buildroot-2019.05.1.tar.gz** (система сборки buildroot)
- Архив **buildroot-external-inmys.tar.gz** с кофигурационными скриптами, настройками и патчами для сборки.
- Скрипт **install_board.sh** для настройки системы сборки.

Сборка uboot, kernal и rootfs

Перейдите в папке **~/inmys_board/imx6_ull/** и выполните скрипт **~/inmys_board/imx6_ull/install_board.sh imx6ull_board**, скрипт настроит систему сборки. По завершению работы скрипта будут распакованы и созданы ряд папок buildroot-2019.05.1 - система сборки buildroot-external-inmys - BSP dl - папка куда будут скачиватся программы необходимые для сборки. imx6ull_board_output - папка с выходными файлами.

В BSP используется:

- Linux kernel 4.19.35 **git** https://source.codeaurora.org/external/imx/linux-imx branch **imx_4.19.35_1.1.0**
- U-boot v2019.04 **git** https://source.codeaurora.org/external/imx/uboot-imx branch **imx_v2019.04_4.19.35_1.1.0**
- Buildroot 2019.05.1 https://buildroot.org/downloads/buildroot-2019.05.1.tar.gz
- Патчи ядра и его конфигурация **~/inmys_board/imx6_ull/buildroot-external-inmys/board/inmys/imx6ull/linux**
- Патчи загрузчика и его конфигурация **~/inmys_board/imx6_ull/buildroot-external-inmys/board/inmys/imx6ull/uboot**
- Папка overlay **~/inmys_board/imx6_ull/buildroot-external-inmys/board/inmys/imx6ull/rootfs_overlay**

Перейдите в папку **~/inmys_board/imx6_ull/imx6ull_board_output** и выполните команду **make**. После этого начнется сборка системы. По завершению сборки в папке **~/inmys_board/imx6_ull/imx6ull_board_output/images** появятся файлы

- imx6ull-inmys-som.dtb - файл конфигурации периферии для linux
- rootfs.tar - архив с файловой системой
- rootfs.ubi - архив с файловой системой ubifs
- rootfs.ubifs - архив с файловой системой ubifs
- u-boot-dtb.imx - загрузчик
- uImage - ядро linux
- sdcard_imx6ull_board.img - образ SD карты

Пароль и логин для входа **root root**.

Загрузка с SD карты

Для загрузки som модуля с sd карты, необходимо записать загрузочный образ на sd карту. Подключите sd карту к компьютер.

перейдите

```
cd ~/inmys_board/imx6_ull/imx6ull_board_output
```

Замените название SD устройства **sdX** на тот который у вас в системе.

```
dd if=./images/sdcard_imx6ull_board.img of=/dev/sdX
```

Загрузка с NAND

Воспользуйтесь сознанной ранее SD картой и скопируйте на нее в папку /boot файлы u-boot-dtb.imx и rootfs.ubi

Загрузитесь с SD карты и после загрузки системы выполните команды:

```
# cd /boot

# flash_erase /dev/mtd0 0 0
# mount -t debugfs none  /sys/kernel/debug/
# kobs-ng init -x -w --chip_0_device_path=/dev/mtd0 ./u-boot-dtb.imx

# flash_erase /dev/mtd1 0 0
# nandwrite -p /dev/mtd1 ./uImage

# flash_erase /dev/mtd2 0 0
# nandwrite -p /dev/mtd2 ./imx6ull-inmys-som.dtb

# flash_erase /dev/mtd4 0 0
# ubiformat /dev/mtd4 -f rootfs.ubi
```

После копирования выключите питания модуля и переведите модуль для загрузки с NAND.

Проверка работы CAN

Проверка наличия доступных CAN интерфейсов

```
# ifconfig
can0      Link encap:UNSPEC  HWaddr
00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          UP RUNNING NOARP  MTU:16  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:26

can1      Link encap:UNSPEC  HWaddr
00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          UP RUNNING NOARP  MTU:16  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:27
```

```
eth0      Link encap:Ethernet  HWaddr 04:91:62:BC:83:17
          inet addr:192.168.1.199  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::691:62ff:febc:8317/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2686 errors:0 dropped:0 overruns:0 frame:0
          TX packets:27 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:177780 (173.6 KiB)  TX bytes:1566 (1.5 KiB)

eth1      Link encap:Ethernet  HWaddr 04:91:62:BC:91:7E
          inet addr:192.168.0.199  Bcast:0.0.0.0  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Для прослушивания данных передаваемых по can можно воспользоваться консольной утилитой **candump** Пример для can0

```
# candump can0
  can0  123   [8]  11 22 33 44 55 66 77 88
  can0  123   [8]  11 22 33 44 55 66 77 88
  can0  123   [8]  11 22 33 44 55 66 77 88
  can0  123   [8]  11 22 33 44 55 66 77 88
  can0  123   [8]  11 22 33 44 55 66 77 88
  can0  123   [8]  11 22 33 44 55 66 77 88
```

Для проверки отладки передачи воспользуйтесь консольной программой **cansend** Пример для can0

```
# cansend can0 123#1122334455667788
```

Для временного изменения скорости работы can порта

```
# ifconfig can0 down
# ip link set can0 up type can bitrate 125000
```

Для установки скорости по умолчанию отредактируйте файл /etc/network/interfaces

```
# vi /etc/network/interfaces
```

Проверка двух CAN (необходимо соединить can0 и can1)

```
# candump can0 &
# cansend can1 123#1122334455667788
  can0  123   [8]  11 22 33 44 55 66 77 88
# cansend can1 123#1122334455667788
  can0  123   [8]  11 22 33 44 55 66 77 88
# cansend can1 123#1122334455667788
  can0  123   [8]  11 22 33 44 55 66 77 88
# killall candump
```

Проверка работы UART

На плате доступны Последовательные порты:

- ttymxc0 - консольный порт на USB
- ttymxc1 - порт на USB
- ttymxc2 - rs485 X7_E
- ttymxc4 - ttl X17
- ttymxc5 - rs232 X11
- ttymxc6 - rs232 X12
- ttymxc7 - ttl X17

По умолчанию в сборке включена утилита **picocom** с помощью которой можно по передавать данные.

```
picocom -b 115200 /dev/ttymxc1
```

Настройки Ethernet

По умолчанию установлены статические IP адреса для обоих интерфейсов. Для изменения поведения(изменения IP адресов и под сетей или для выключения DHCP) необходимо отредактировать файл **/etc/network/interfaces**.

```
auto lo
iface lo inet loopback

auto can0
iface can0 inet manual
    bitrate 125000
    up /sbin/ip link set $IFACE down
    up /sbin/ifconfig $IFACE txqueuelen 10
    up /sbin/ip link set $IFACE type can bitrate 125000 restart-ms 10
    up /sbin/ip link set $IFACE up
```

```
auto can1
iface can1 inet manual
    bitrate 125000
    up /sbin/ip link set $IFACE down
    up /sbin/ifconfig $IFACE txqueuelen 10
    up /sbin/ip link set $IFACE type can bitrate 125000 restart-ms 10
    up /sbin/ip link set $IFACE up


# Config static ip
auto eth0
iface eth0 inet static
        address 192.168.1.199
        netmask 255.255.255.0
# Configure eth0 with dhcp IP
# auto eth0
# iface eth0 inet dhcp



# Config static ip
auto eth1
iface eth1 inet static
        address 192.168.0.199
        netmask 255.255.255.0

# Configure eth0 with dhcp IP
# auto eth1
# iface eth1 inet dhcp
```

Пример получения сетевых настроек по dhcp на eth0

```
auto lo
iface lo inet loopback

auto can0
iface can0 inet manual
    bitrate 125000
    up /sbin/ip link set $IFACE down
    up /sbin/ifconfig $IFACE txqueuelen 10
    up /sbin/ip link set $IFACE type can bitrate 125000 restart-ms 10
    up /sbin/ip link set $IFACE up


auto can1
iface can1 inet manual
    bitrate 125000
    up /sbin/ip link set $IFACE down
    up /sbin/ifconfig $IFACE txqueuelen 10
```

```
    up /sbin/ip link set $IFACE type can bitrate 125000 restart-ms 10
    up /sbin/ip link set $IFACE up


# Config static ip
#auto eth0
#iface eth0 inet static
#        address 192.168.1.199
#        netmask 255.255.255.0
# Configure eth0 with dhcp IP
 auto eth0
 iface eth0 inet dhcp



# Config static ip
auto eth1
iface eth1 inet static
        address 192.168.0.199
        netmask 255.255.255.0

# Configure eth0 with dhcp IP
# auto eth1
# iface eth1 inet dhcp
```