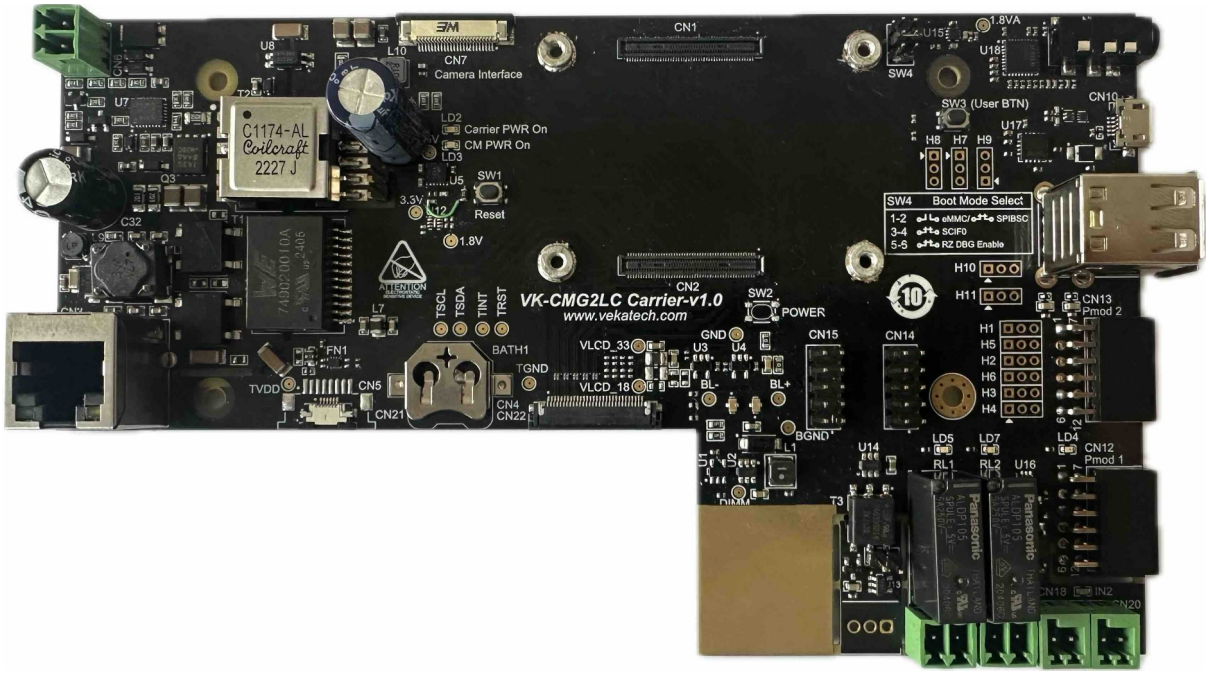
VK-CMG2LC Development Carrier Board How To



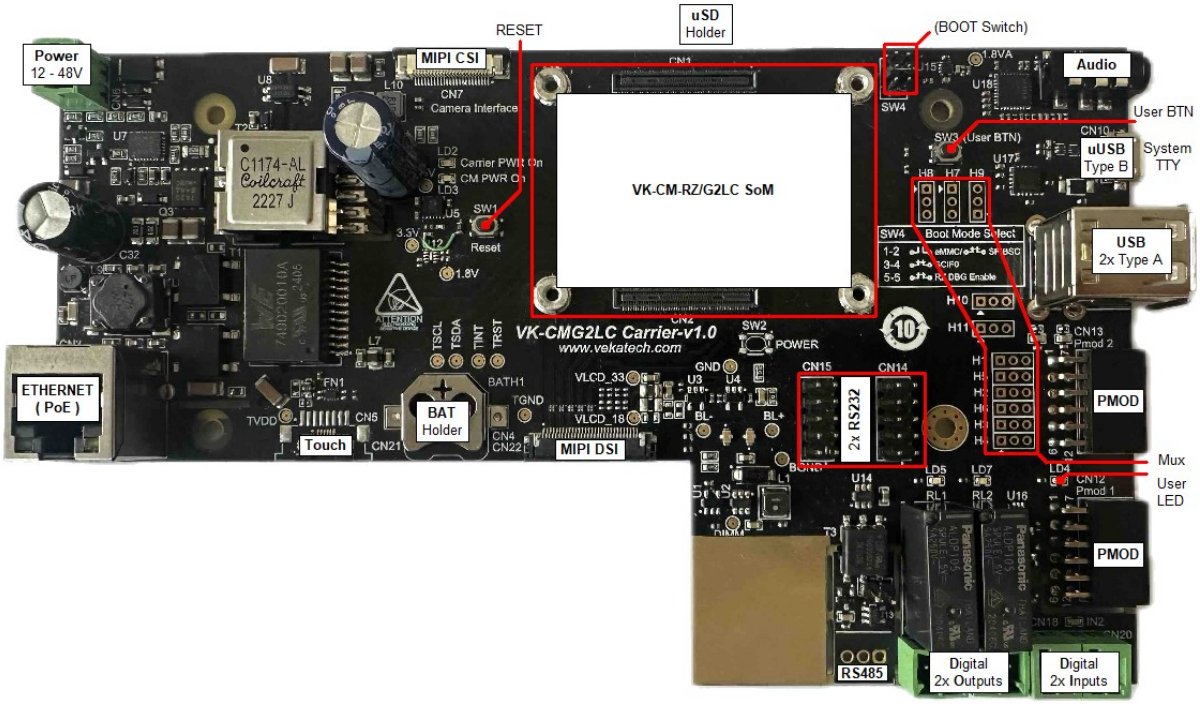
VK-CMG2LC v1.0 Development Carrier Board

Content:

1. Introduction

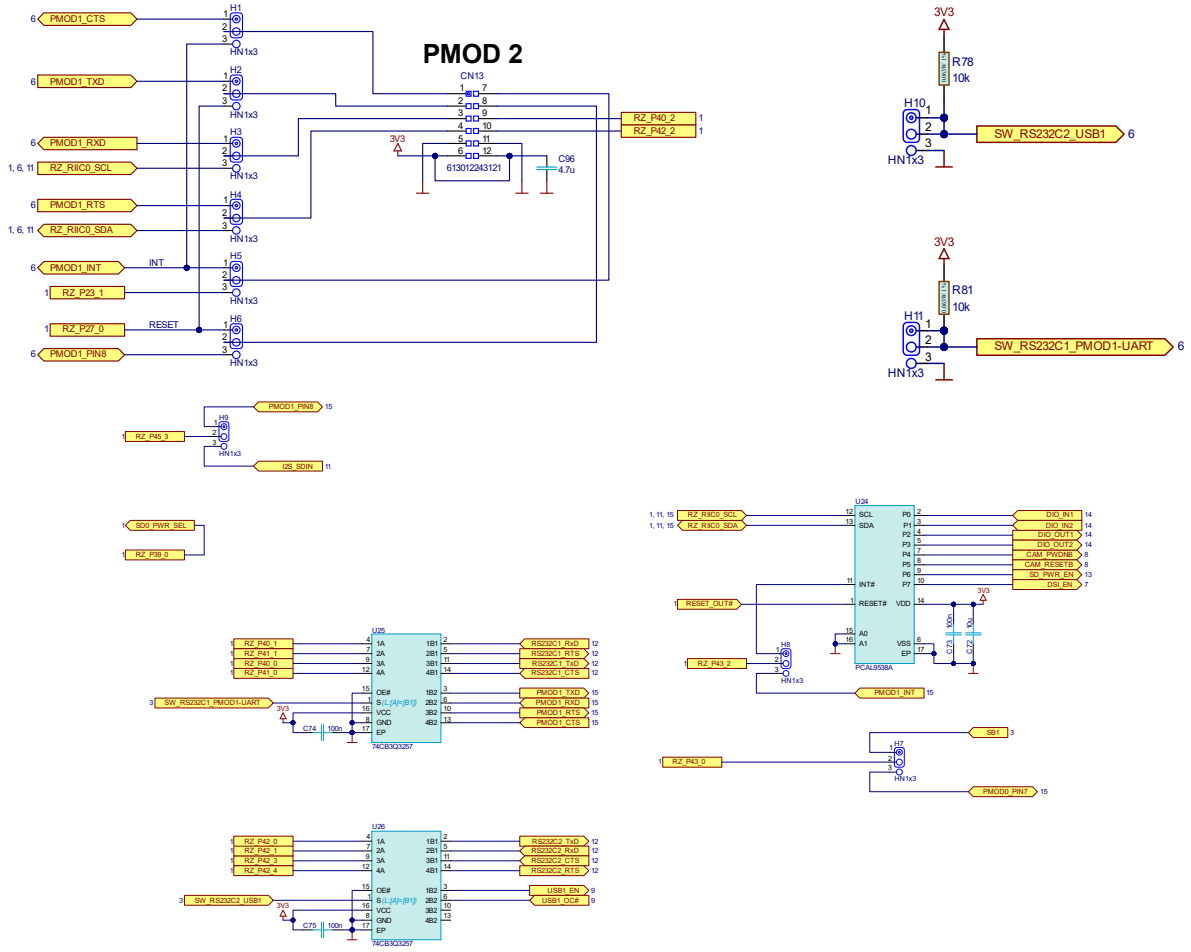
VK-CMG2LC is development carrier board for VK-CM-RZ/G2LC **S**ystem **o**n **M**odule. It is based on Renesas **R9A07G044C22GBG, Dual ARM Cortex-A55 + Cortex-M33 MCU**. The main purpose of this manual is to show how to get started with the module onboard.

1.1 Connectors



Connectors & Signals

1.2 Mux

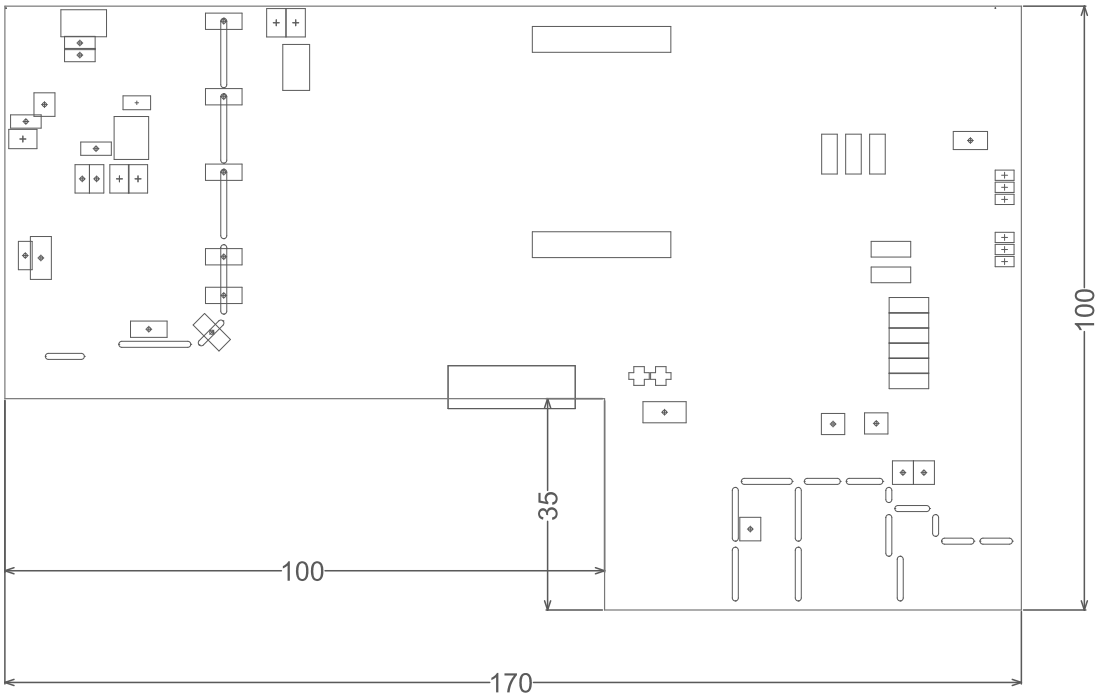


Mux

Some of the periphery is available only when Mux is configured correctly:

- **H 1** → Connects **PMOD2_PIN1** to PMOD1_CTS or PMOD1_INT.
- **H 2** → Connects **PMOD2_PIN2** to PMOD1_TXD or P27_0.
- **H 3** → Connects **PMOD2_PIN3** to PMOD1_RXD or I^2C_SCL0.
- **H 4** → Connects **PMOD2_PIN4** to PMOD1_RTS or I^2C_SDA0.
- **H 5** → Connects **PMOD2_PIN7** to PMOD1_INT or P23_1.
- **H 6** → Connects **PMOD2_PIN8** to P27_0 or PMOD1_PIN8.
- **H 7** → Connects **P43_0** to UserBTN or PMOD1_PIN7
- **H 8** → Connects **P43_2** to PCAL9538A_INT or PMOD1_INT
- **H 9** → Connects **P45_3** to PMOD1_PIN8 or I^2S_SIN
- **H 10** → Connects **P42_0 – P42_4** to RS232C2 or USB1
- **H 11** → Connects **P40_0 – P40_1 & P41_0 – P41_2** to RS232C1 or PMOD1-UART

1.3 Dimensions



Board size & Mounting Holes

2. Power Up

The board can be powered with from 2 sources:

- POWER connector (24 – 48 V)
- PoE

In idle state (without plugged periphery) the board consumes ~ **300 mA**. The consumption however, can jump up to ~ **1,2 A** (if Camera, Display, Ethernet, USB Flash, USB Keyboard, Audio jack & SD card are plugged in).

3. Install U-boot (v2021.10)

You can get the U-boot firmware from our site or compile it yourself from our repository, if you prefer manually building it from scratch.

- Connect **VK-CMG2LC** to the PC (through **USB Type B micro**) & see what COM port is assigned by the OS in the Device Manager.
- Place a jumper in position 3-4 of the **SW4** (middle) to boot from the **SCIF0**.
- Download **vkPyFlasher** rescue tool and launch it.
- Download the U-boot **firmware** and unpack it in: **vkPyFlasher/images**.

3.1 into SPI Flash

- Run the following commands:

```
cd vkPyFlasher.
flash_boot.py --board=vkcmg2lc --serial_port=COM<n> --qspi.
```

- Press **reset** button on the **VK-CMG2LC**
- Wait the flashing to complete.

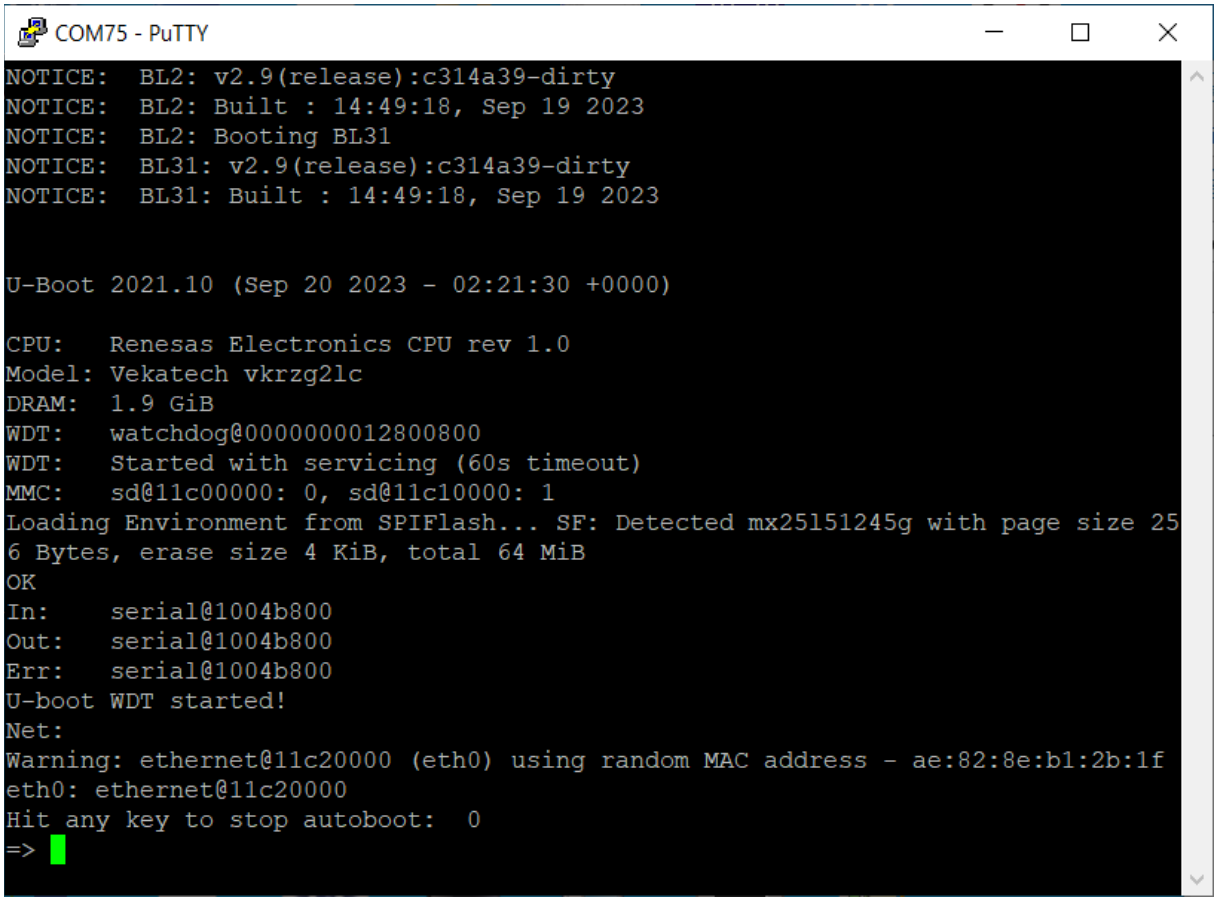
3.2 into eMMC SSD

- Run the following commands:

```
cd vkPyFlasher.
flash_boot.py --board=vkcmg2lc --serial_port=COM<n>.
```

- Press **reset** button on the **VK-CMG2LC**
- Wait the flashing to complete.

After download is ready, remove the jumper from **SW4** to boot from **SPIBSC** (by default) and press **reset** => you should now be able to see the boot log of the U-boot.



U-boot log

4. Boot logic

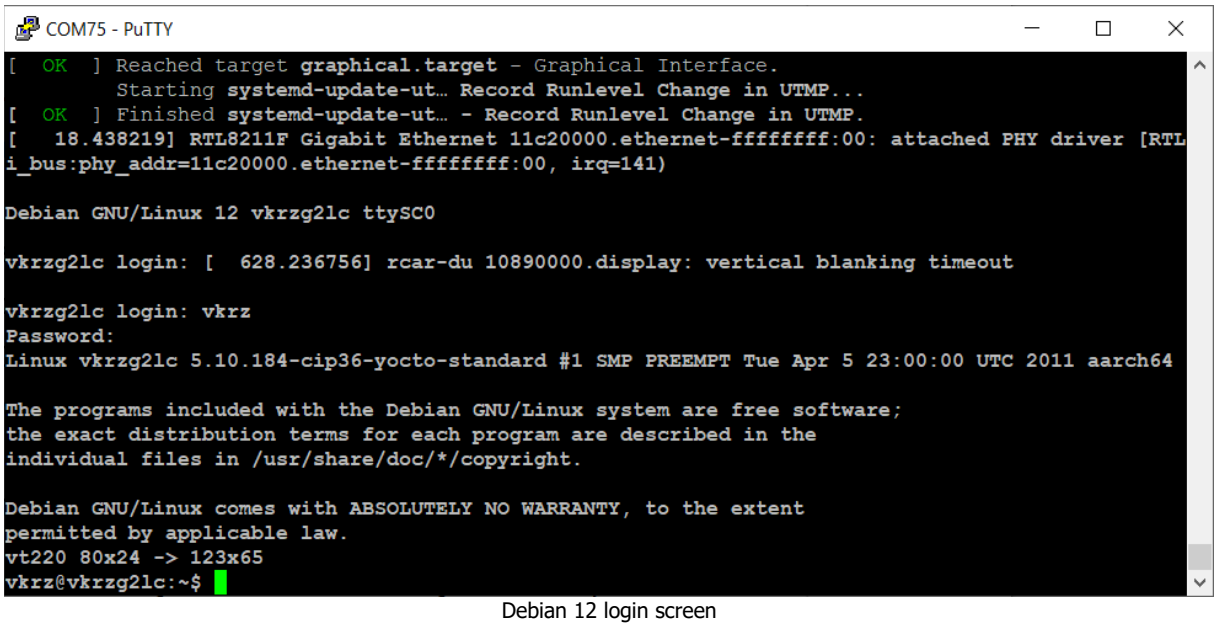
Understanding boot logic is crucial. As you see there are **2** copies of **U-boot** (1 in eMMC & 1 in SPI). Environment parameters of the ones are slightly different from the others. They are configured in such way, that when **SW4** is set to boot from eMMC (**no jumper**) it's U-boot searches Linux image in the same that **eMMC**, but when SW4 is set to boot from SPI (**1-2**) it's U-boot searches Linux image in **µSD** card. That's why default U-boot environment parameters differs on purpose and you don't need to edit them every time you want to boot from one or other media (you just set SW4). That actually explains why booting from SPI, leads to booting from µSD. You can always change those parameters & boot from wherever you want.

5. Install Linux (Kernel v5.10.184)

Now that you have 2 workable U-boot instances, up & running, it is time to load something bigger. You can get Debian Linux image from our site or compile Yocto yourself from our repository, (if you prefer building it from scratch):

5.1 into SD card → Debian v12.4 (Bookworm)

- Get a **µSD** card with **min 4 G** capacity and plug it into the PC.
- Download tool, named Rufus from here.
- Unpack **debian-bookworm-vkcmg2lc.img.xz**.
- Launch Rufus, for **Device** select µSD card drive, for **boot section** select desired Linux image file (in this case **debian-bookworm-vkcmg2lc.img**).
- Hit **START** & wait completion (Status **READY**), and eject µSD card from the PC.
- Plug µSD card into VK-CMG2LC's holder, set **SW4** to connect (1-2) & press reset.
- You should now be able to see login screen, use user: **`vkrz`** & password: **`vkcmg2lc`**.



Debian 12 login screen

- You can check for available package updates: `sudo apt-get update`.

If there are available upgrades such as in this case:

```
34 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

- You can install them: `sudo apt-get upgrade`.
- If you want to extend the root partition to use the full capacity of the µSD:

`sudo growpart /dev/mmcblk1 2 && sudo resize2fs /dev/mmcblk1p2`.

- If you want to extend the root partition to a fixed size and form a new separate partition you should first install partition editor `sudo apt-get install parted`.

Extend the root partition to **N** GB size: `sudo parted /dev/mmcblk1 resizepart 2 NG`.

Fix filesystem block size: `sudo resize2fs /dev/mmcblk1p2`.

Make new partition: `sudo parted /dev/mmcblk1 "mkpart primary fat32 NG -1"`.

Format the new partition: `sudo mkfs.vfat -F 32 /dev/mmcblk1p3`.

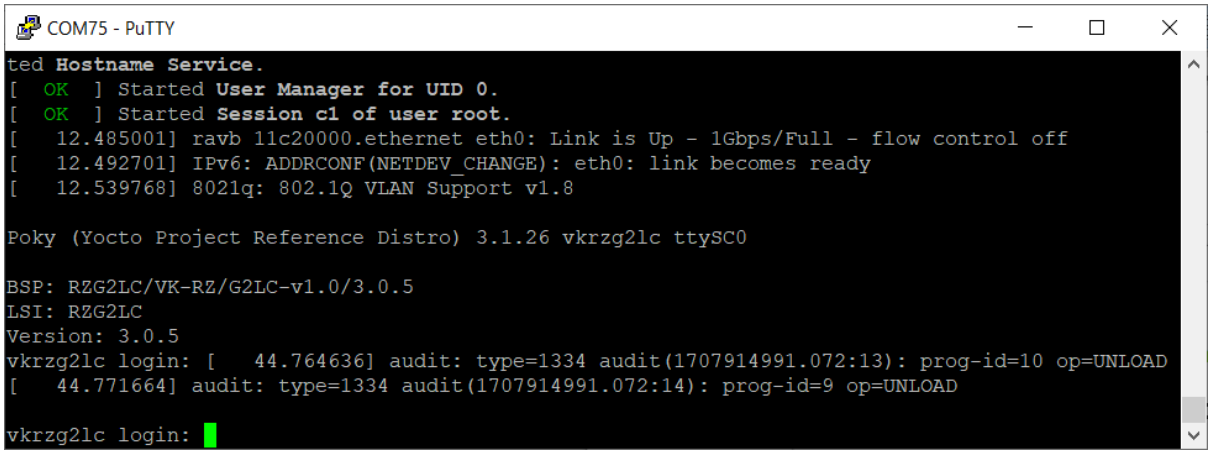
5.2 into eMMC SSD → Yocto v3.1.26 (Dunfell)

Get the **vkPyFlasher** prepared:

- Connect **VK-CMG2LC** to the PC (through **USB Type B micro**) & see what COM port is assigned by the OS in the Device Manager.
- Connect **VK-CMG2LC** to the PC (through the **lower USB Type A**)
- Remove any jumper from **SW4** to boot from the **eMMC**.
- Download the desired Linux image (Yocto / Debian), and place it at the location: **vkPyFlasher/images/vkcmg2lc**.

Flash the image:

- `cd vkPyFlasher`.
- In case you want Yocto:
  `flash_img.py --board=vkcmg2lc --serial_port=COM<n> --image_rootfs=core-image-weston-vkcmg2lc.simg`.
- In case you want Debian:
  `flash_img.py --board=vkcmg2lc --serial_port=COM<n> --image_rootfs=debian-bookworm-vkcmg2lc.simg`.
- Press **reset** button on the **VK-CMG2LC**.
- Wait the flashing to complete.
- Open the COM port assigned by the OS for the VK-CMG2LC with (115200|B|N|1) settings.
- Press **reset** once again.
- Login to Yocto (user: **root**) or Debian (user: **vkrz** & password: **vkcmg2lc**).
- In case of Debian, you can setup the partition on the fly:
  → Extend the root partition to use the full capacity of the eMMC:
  `sudo growpart /dev/mmcblk0 2 && sudo resize2fs /dev/mmcblk0p2`.
- In case of Yocto, you should now be able to see login screen:



Yocto Login screen

6. Launch the installed Linux images

Thanks to the convenient boot logic, launching is easy, it depends on correct setting of **SW4**

- To boot Linux from µSD: place a jumper to **SPIBSC (1-2)**.
- To boot Linux from eMMC: remove the jumper from **eMMC (1-2)**.
- To boot Linux from Ethernet: set the switch to **SPIBSC** and make sure µSD slot is Empty. When U-boot can't find SD card, it will try to boot from its **serverip**, **tftpdir** & **netrootfs** environment parameters. (serverip is the address, where U-boot will look for NFS & TFTP servers, tftpdir shows where the boot directory is on the server and netrootfs → where root directory is on the server) Same server can be used to boot multiple boards, every board can have its own boot and root directories and that's why tftpdir and netrootfs parameters should be filled with the correct paths on the server.

7. Apply Overlays

Overlays are way to tell Linux to use or not a given periphery. They can even point a specific hardware for a same periphery and form configurations for different version of a board. For example in version 1, the board can use **Realtek** audio driver, in other **someone's else**, in third no audio at all. Management of the overlays is done through **uEnv.txt**, located in **/boot** directory of the particular linux image. This file is analyzed during boot of the Linux image, so every change in that file takes effect after boot. To apply overlay, you need to line it up in the following row:

```
fdt_extra_overlays= vkrz-dsi-kd070hdfia030.dtbo
```

All available overlays that can be applied are listed in **/boot/overlays**, but simultaneously only these can be activated at the same time.

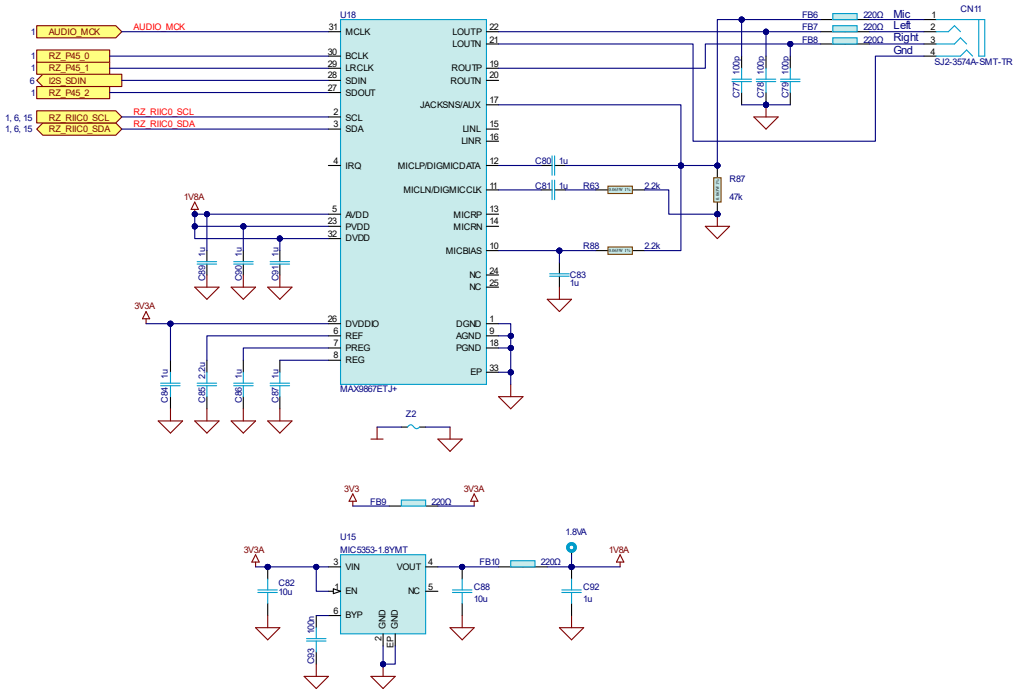
- vkrz-cm33.dtbo
- vkrz-cm-scif2.dtbo / vkrz-cm-usb1.dtbo
- vkrz-csi-ov5645.dtbo
- vkrz-dsi-kd070hdfia030.dtbo / vkrz-dsi-kd101wxfid045.dtbo
- vkrz-qspi-mx25l51245g.dtbo
- vkrz-udma.dtbo

8. Use various periphery in Linux

To demonstrate using different periphery, we are going to use the Debian image, for other images (Yocto for example) the commands could be not available or could be slightly different, so don't worry if some of them do not work, just google how to do it for your particular distribution.

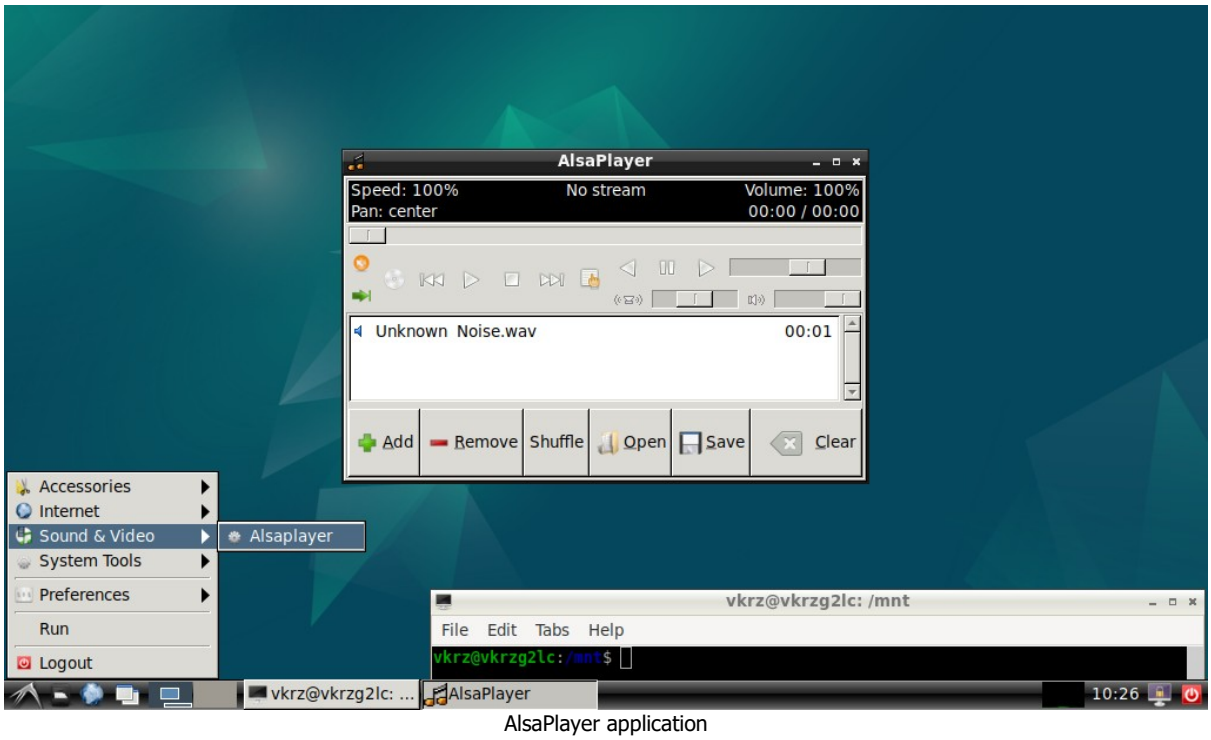
8.1 Audio

Audio is enabled by default and does not need overlay.



3,5mm CTIA Mini Jack

- Plug headphones in the jack & type: `aplay /usr/share/sounds/alsa/Noise.wav`.
- You can also play it through the GUI: just go to **Start/Sound & Video /Alsaplayer**. press Add, browse to /usr/share/sounds/alsa/Noise.wav, press Play.
- You should now be able to hear white noise sample. If for some reason you can't here anything, open alsamixer and check if **Card** (in the upper left corner) is **audio-da7212**. If it is different, press **F6** (Select sound card) and choose **audio-da7212**, also find **Mixout Left DAC Left** & **Mixout Right DAC Right** and make sure they are not **Off**.

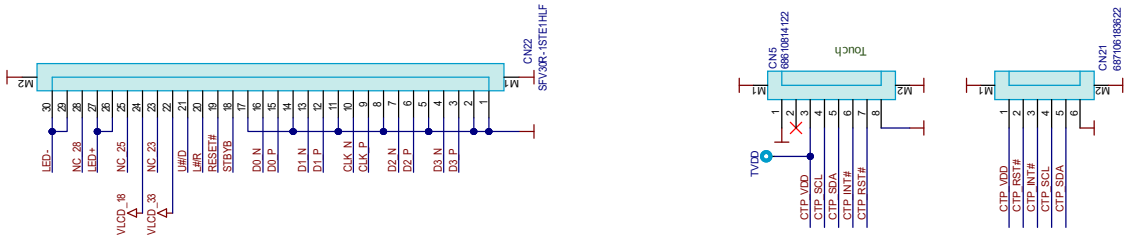


AlsaPlayer application

8.2 MIPI DSI Display

Make sure you are applied one of these overlays in **/boot/uEnv.txt**.

```
fdt_extra_overlays=vkrz-dsi-kd070hdfia030.dtbo/vkrz-dsi-kd101wxfid045.dtbo
```

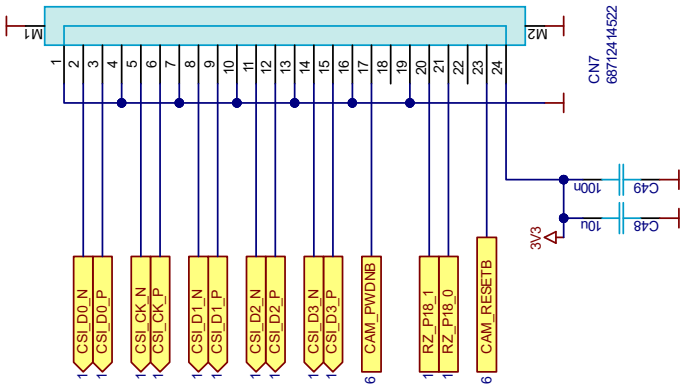


30 pin FPC MIPI DSI Connector

8.3 MIPI CSI Camera

Make sure you are applied **vkrz-csi-ov5645.dtbo** overlay in **/boot/uEnv.txt**.

```
fdt_extra_overlays=vkrz-csi-ov5645.dtbo
```



22 pin FPC MIPI CSI Connector

- Install tool to catch the video stream: `sudo apt-get install vlc`.
- Download the CSI v4l2-init script and put it in the home directory (`~`).

If the hardware is available, the script will set default settings through these rows:

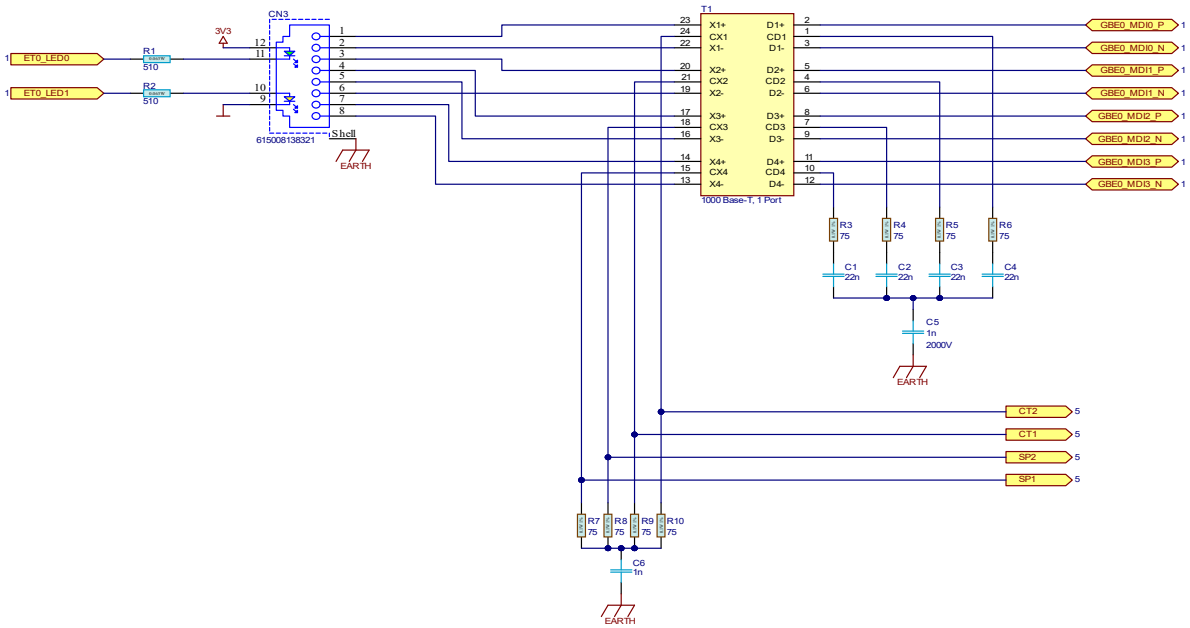
```
media-ctl -d /dev/media0 -l "'rzg2l_csi2 10830400.csi2':1 -> 'CRU output':0 [1]"
media-ctl -d /dev/media0 -V "'rzg2l_csi2 10830400.csi2':1 [fmt:UYVY8_2X8/$imx219_res field:none]"
media-ctl -d /dev/media0 -V "'imx219 0-0010':0 [fmt:UYVY8_2X8/$imx219_res field:none]"
```

If default format (UYVY8_2X8) does not suit your needs, you can always change it.

- When you are happy with settings in the script, execute it: `~/v4l2-init.sh`.
- Open VLC through the GUI: go to **Start/Sound & Video/VLC media player**.
- Go to **Media** → **Open Capture Device …** → **Capture Device** Tab
  for **Video device name** select **/dev/video0** and press **Play**.
- You should now be able to see the video input stream.

8.4 Ethernet

Ethernet is enabled by default and does not need overlay. Plug the **cat** cable into **RJ45** connector and in a couple of seconds the board should get an **IP** from the local **DHCP** server.



RJ45 connector

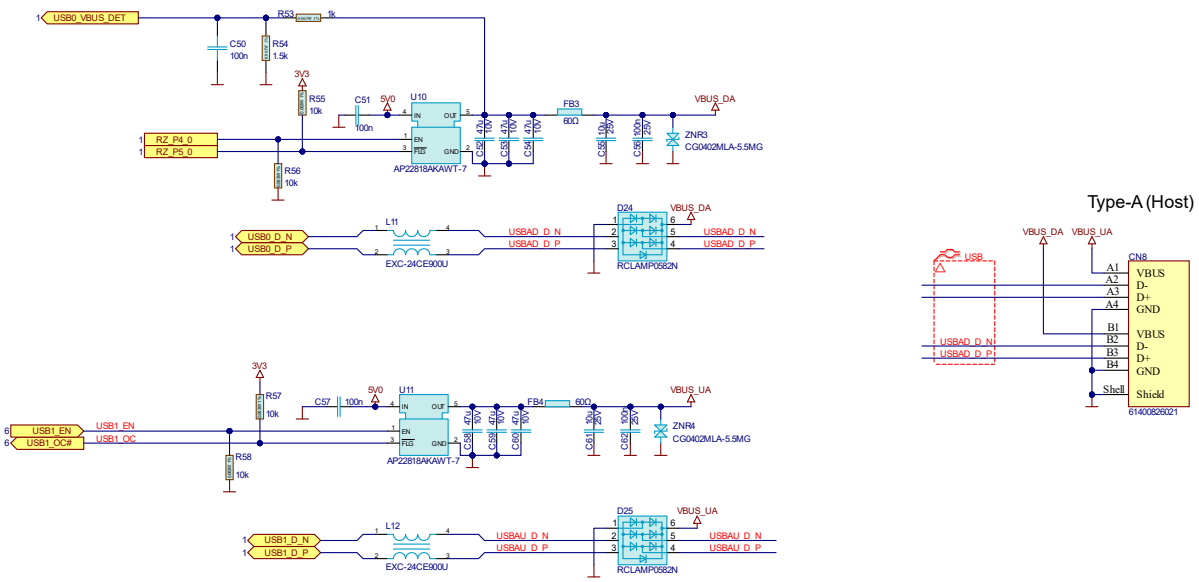
- You can check what IP is assigned: `sudo ifconfig`.
- You can probe if Host is reachable: `ping vekatech.com`.

8.5 USB

USB0 (the lower) is enabled by default and does not need overlay, but if you want to use USB1 (the upper), remove any jumper on H10 and make sure you are applied **vkrz-cm-usb1.dtbo** overlay in **/boot/uEnv.txt**.

```
fdt_extra_overlays=vkrz-cm-usb1.dtbo.
```

Make sure `vkrz-cm-scif2.dtbo` is excluded, from `fdt_extra_overlays`, because it uses same pins as USB1. To test the USB, plug a device such as: Mouse, Keyboard, USB Flash, USB Camera & whatever other USB device you can think of.



USB Type A connectors

- Mouse: It is Plug & Play device, you don't need to do anything.
- Keyboard: Also Plug & Play device, you don't need to do anything.
- Flash Drive: most of the latest Linux distributions mounts the drive automatically, (Debian 12 is not an exception), so it is again another Plug & Play device.
- Camera: usually you need software to see the stream, the example below is with VLC, but ffmpeg or others are also a great choice.

  → Open VLC through the GUI: go to **Start/Sound & Video/VLC media player**.

  → Go to **Media** → **Open Capture Device …** → **Capture Devie** Tab

  for **Video device name** select **/dev/video0** and press **Play**.

  → You should now be able to see the USB video stream.
- GSM Modem: Also Plug & Play device, but to set a connection it needs some setup:

→ Make sure `sudo apt-get install network-manager modemmanager` are installed

→ Start their services: `systemctl start NetworkManager ModemManager`.

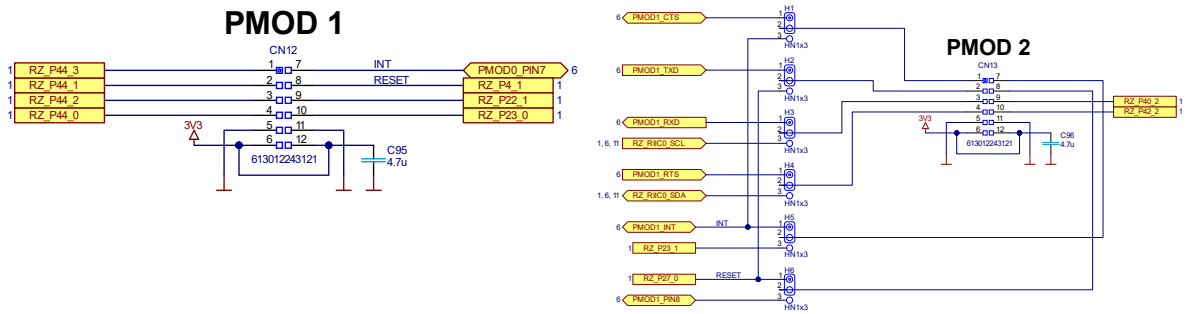
→ If autostart on boot is a must: `systemctl enable NetworkManager ModemManager`.

→ Configure GSM connection : `sudo nmcli connection add type gsm ifname '*' con-name '<name>' apn '<AP>' connection.autoconnect yes`.

→ A new network interface, type **gsm** should appear: `nmcli device status`.

8.6 PMOD (GPIO)

GPIOs are enabled by default and do not need overlay.



PMOD connectors

Pay attention that **RZ_P23_0, RZ_P23_1 & RZ_P27_0** are on the **Low Voltage Power Domain** (i.e. their logical **HI** levels are **1,8V**) and take that in consideration when using these GPIOs.

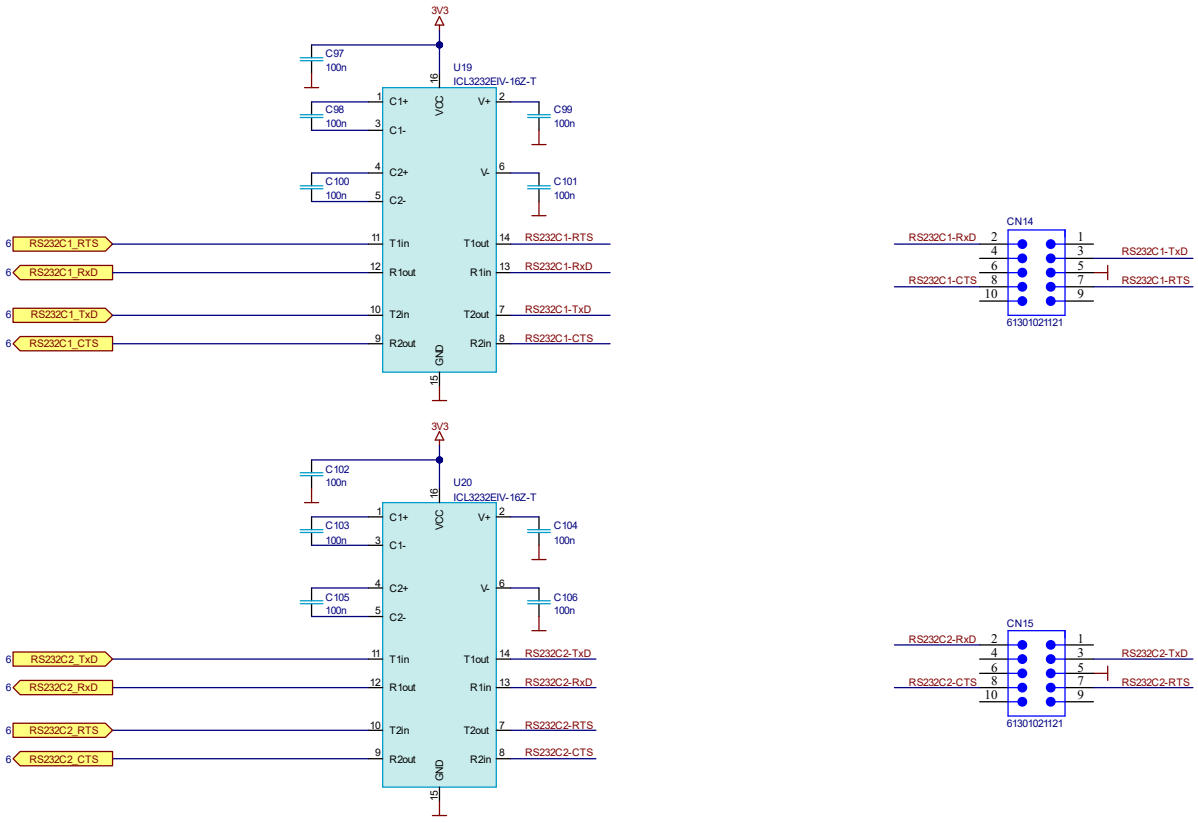
- You have to tell the system which pin you want to manipulate. Let's say the pin is "**P**port_pin". You have to calculate the internal pin number: (**port** x **8**) + **pin** + **120**, if **P23_0** is desired to be controlled, the internal number is: (**23** x 8) + **0** + 120 = **304**.
  The way to tell the system is with these commands: `sudo su`
  `echo 304 > /sys/class/gpio/export`.
- If you want to get the value of **P23_0** pin, you can type this:
  `echo in > /sys/class/gpio/P23_0/direction`.
  `cat /sys/class/gpio/P23_0/value`.
- If you want to set **P23_0** pin to **HI**, you can type this:
  `echo out > /sys/class/gpio/P23_0/direction`.
  `echo 1 > /sys/class/gpio/P23_0/value`.
- If you don't want to use the pin anymore you can release the resource:
  `echo 304 > /sys/class/gpio/unexport`.

8.7 RS232 (UART)

UART1 is enabled by default and does not need overlay, but if you want to redirect it to RS232 interface put a jumper on H11 between (2-3). If you want to use UART2, put another jumper on H10 between (2-3) to also redirect it to RS232 and make sure you are applied **vkrz-cm-scif2.dtbo** overlay in **/boot/uEnv.txt**.

```
fdt_extra_overlays=vkrz-cm-scif2.dtbo.
```

Make sure `vkrz-cm-usb1.dtbo` is excluded, from `fdt_extra_overlays`, because it uses same pins as UART2.

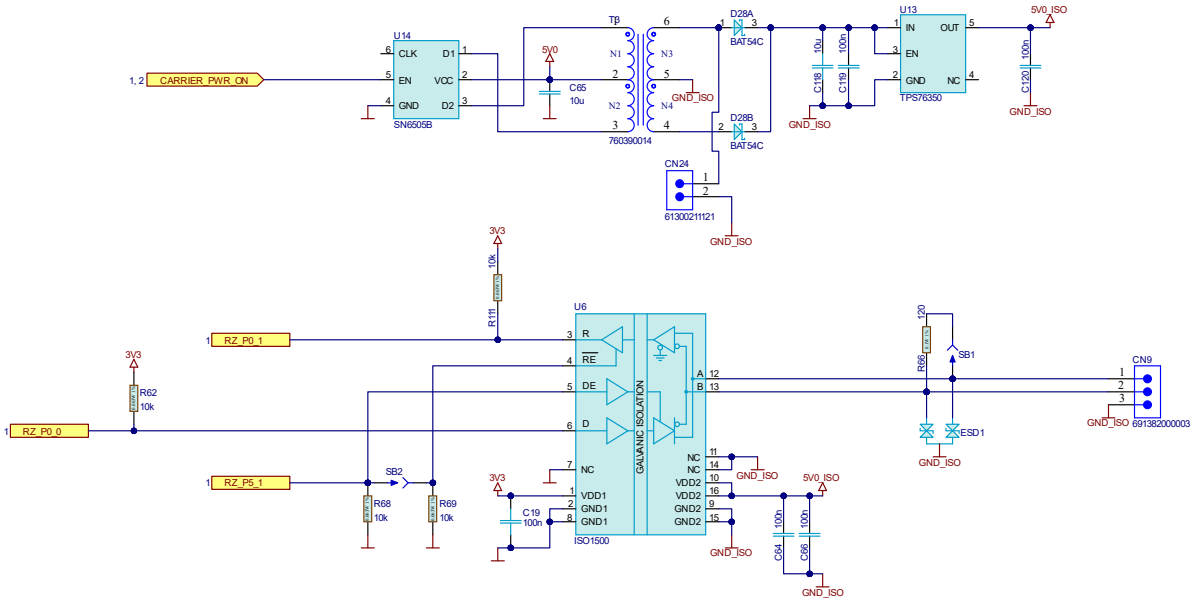


RS232 headers

- Install port software: `sudo apt-get install screen`.
- Short TX & RX together, so a Loopback test can be run.
- Execute: `screen /dev/ttySC2` or `screen -f /dev/ttySC2` (if flow control overlay)
- Type something and you should now be able to see what you are typing.
- If you remove the short, you won't see what you are typing.
- Press **Ctrl+A** and then **K** to exit, (you may need to apply with **y**)

8.8 RS485 (UART)

RS485 is enabled by default and does not need overlay.



RS485 connector

Set RS485 module to 19200 8 N 1 → `stty -F /dev/ttySC<n> 19200 cs8 -cstopb -parenb`.

- Init DE pin. RS485 is actually UART with **D**ata **E**nable pin, which goes **Hi** before **TX** transaction and goes back to **Low** after the transaction, so you need to manually control that pin, this means the internal pin should be **P5_1**, which is (**5** x 8) + **1** + 120 = **161**.

  `sudo su` && `echo 161 > /sys/class/gpio/export`.

  `echo out > /sys/class/gpio/P5_1/direction`.

  `echo 0 > /sys/class/gpio/P5_1/value`.

- TX some data: `echo 1 > /sys/class/gpio/P5_1/value`.

  `echo "Helow world" > /dev/ttySC3`.

  Wait data to be fully transmited & exec. `echo 0 > /sys/class/gpio/P5_1/value`.

- RX some data: `cat /dev/ttySC3`.

  Note that the receiving terminal can have buffering enabled, so the data read might not be displayed immediately. The buffer is flushed once enough data is entered, or when enough newlines are encountered.

- Release DE pin resource: `echo 161 > /sys/class/gpio/unexport`.

9. Install Mali GPU driver on Debian

Renesas RZ/G2LC MCU is equipped with **Mali-G31** GPU on chip. Debian has a built in support for that GPU through Mesa/X.org project. The productivity of that driver, however, is quite disappointing, especially when you try to use the browser. You can use the evaluation version of the driver, optimized for that GPU, from Renesas. The downside of it, however, is it's evaluation, i.e. it only works for a couple of hours. For the full version you have to contact with Renesas. The driver supplied by Renesas is only compatible with the Wayland display server protocol, keep that in mind if you consider using it.

- You can **Install** the driver directly by executing the script:
  In short the script installs **Weston**, configures a Weston startup service, and finally installs the driver itself.
  → On the target board, get the script for installation:
  `wget https://vekatech.com/os/debian/get_mali.sh`.
  → On the target board, make it executable:
  `chmod +x get_mali.sh`.
  → On the target board, Install the driver:
  `./get_mali.sh`.

10. Launch App on OS start up

The way to do it, is to create a user service, i.e write a text file, with **.service** extension, located at **/usr/lib/systemd/user**.

Let's say the application that needs to be launched is the **Chromium** browser, and the service is named **browser.service** . The contents of that file should look like this:

```
[Unit]
Description=Chromium (Kiosk  Mode)
After=graphical-session.target

[Service]
Restart=on-failure
RestartSec=5
Environment="WAYLAND_DISPLAY=wayland-1"

ExecStart=/usr/bin/chromium -enable-features=UseOzonePlatform
--ozone-platform=wayland --kiosk https://vekatech.com

[Install]
WantedBy=default.target
```

- To allow the service to be launched on OS start up execute this:
  `systemctl --user enable browser.service`.

- To try if the service works as expected, reboot the system:
  `sudo reboot`.

Revision overview list

Revision number	Description changes
0.1	Initial

Vekatech Ltd.

63, Nestor Abadzhiev st.
4023 Plovdiv
Bulgaria
Tel.: +359 (0) 32 262362
info@vekatech.com

www.vekatech.com