MX-V User Guide

Host Mobility, 2022-03-30

Copyright © 2021, 2022 Host Mobility AB

MX-V is a registered trademark in Sweden.

You are free to share and adapt this work under the terms of the CC BY-SA 4.0 license.

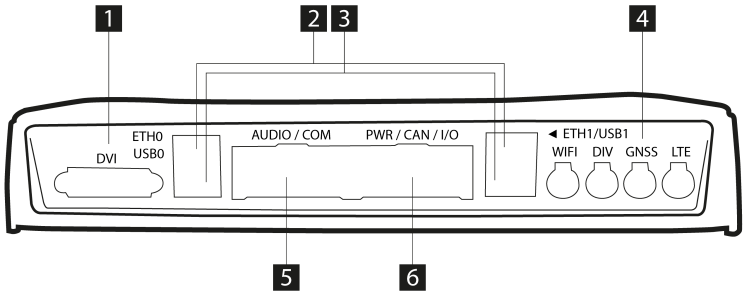
This product and any batteries it contains must not be disposed of with your household waste. Instead, it is your responsibility to hand this over to an applicable collection point for the recycling of batteries and electrical and electronic equipment.

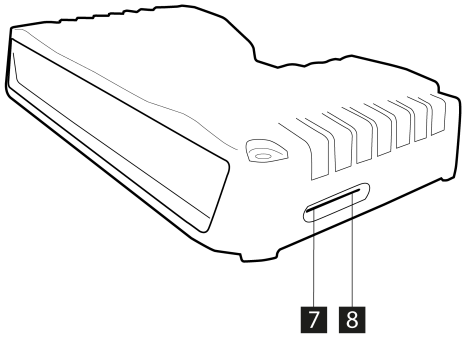
Table of Contents

Chapter 1. Overview and installation

1.1. Connectors

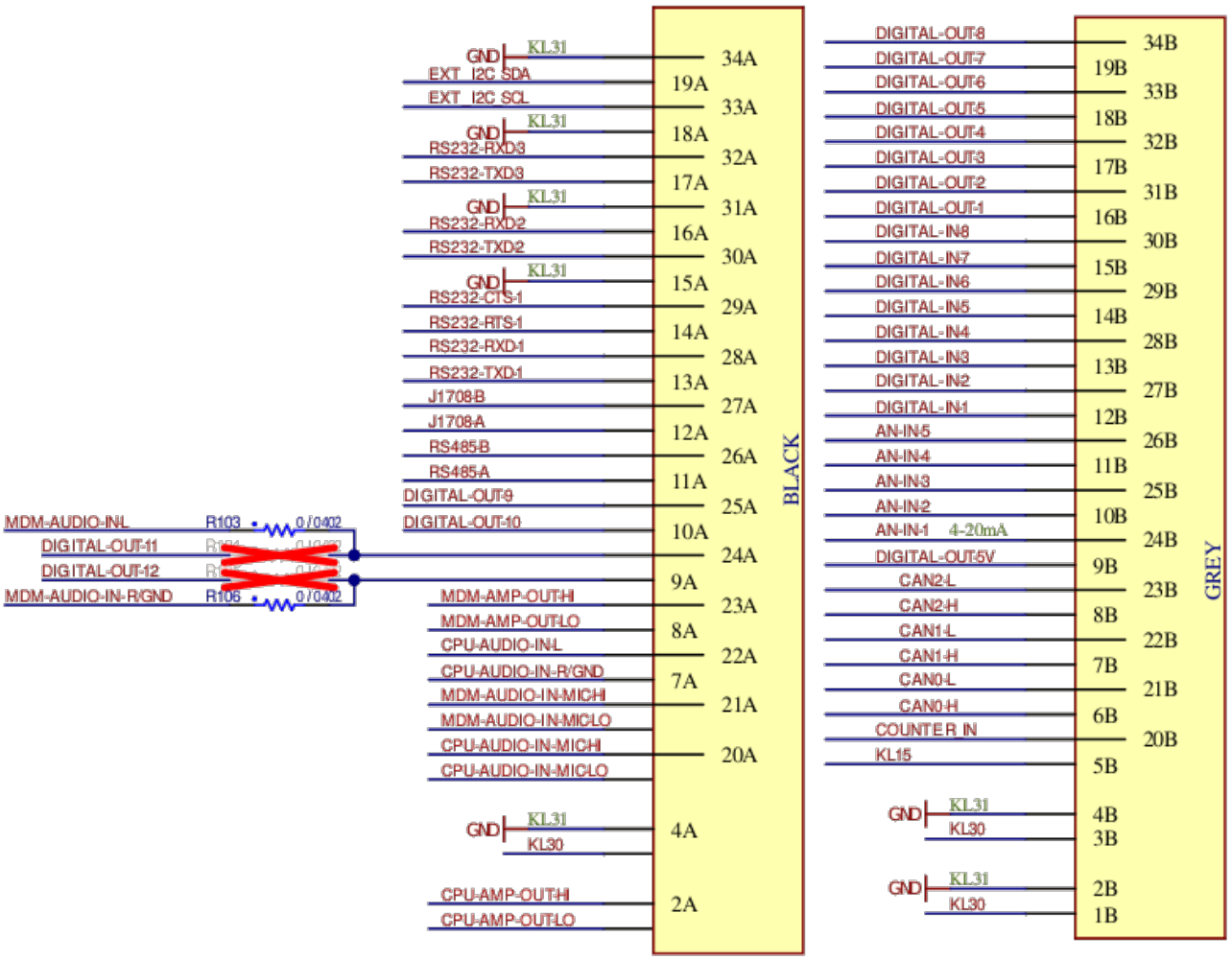
The MX-V has the following connectors accessible from the outside.





1. DVI connector for display
2. two RJ45 Ethernet ports
3. two USB type A connectors
4. antenna connectors for GPS, WiFi and mobile
5. 34-pin I/O connector, including serial interfaces, see below
6. 34-pin I/O connector, including power and CAN, see below
7. SDHC card slot
8. SIM card slot

1.2. I/O connectors



1.2.1. Power

KL30 is positive power terminal (red) and KL31 is negative (black). The units start when KL15 (yellow ignition input) is connected to positive input (e.g. battery voltage).

1.2.2. Digital out

The digital out channels can both source and sink a load. When sourcing, enabling the channel (output pin) connects it to **battery voltage**, while sinking connects the channel to ground.

1.2.3. Analog/Digital in

The (5+8) analog/digital in channels can measure voltages up to 32V. The difference lies in the hardware filtering where analog is filtered at 25Hz and digital at 250Hz.

1.3. Initial setup

- Connect either Ethernet connector to a router to get an IP address through DHCP. Alternatively, connect a male to male USB-A cable between the MX-V and your workstation for direct network access.
- You can also access the machine without a workstation by connecting a computer monitor along with a keyboard.
- Connect the MX-V harness to a power supply in the range 9 to 32V. Then nominal voltage is 24V.

The MX-V boots automatically when power and ignition are applied. The boot time depends on the configuration. For the current development build with a lot of things enabled it takes about 15 seconds.

1.4. Operating system

The Host Mobility MX-V platform runs a custom built GNU/Linux operating system. The current kernel version is 4.19. To load the kernel, the U-Boot bootloader is used. The operating system for the MX-V is installed on the on-board eMMC memory. It contains the U-boot, kernel image plus device tree as well as the root file system.

The MX-V is delivered with the operating system installed and you can login through a secure shell (SSH).

Chapter 2. Login

The MX-V has an SSH server enabled by default. For ethernet over USB0, the IP address is 192.168.250.1. For regular ethernet, the MX-V gets IP addresses from a DHCP server, e.g. a router. In case there is no DHCP server available, the IP addresses default to `192.168.1.200` (eth0) and `192.168.2.200` (eth1).

Log in by running `ssh root@IP-ADDRESS` and provide an empty password. The latter can be changed as usual with the `passwd` command.

Chapter 3. Networking

Networking is configured by the scripts in `/etc/network`.

Configuration can also be done manually with the iproute2 utilities, e.g. the `ip` command.

3.1. Ethernet

The MX-V has two Ethernet ports, `eth0` with 1 gigabit signalling speed and `eth1` at 100 megabit. In the default configuration the Ethernet interfaces are configured to request their IP addresses using DHCP with a fallback to static addresses (192.168.1.200 for eth0 and 192.168.2.200 for eth1). See scripts in `/etc/network/if-up.d` for details.

3.2. USB

Ethernet over USB is supported via RNDIS. Connect a USB-A cable to the USB0 port on the MX-V and a free USB port on e.g. a PC, then power on the MX-V or reboot. This will give the PC the IP address 192.168.250.2 and the MX-V 192.168.250.1.

3.3. WiFi

After accessing the MX-V, you can set up WiFi using e.g. wpa_supplicant. The interface will show up as mlan0.

```
wpa_passphrase ${WIFI_SSID} ${WIFI_PASSWORD}
wpa_supplicant -B -imlan0 -c /etc/wpa_supplicant.conf -Dnl80211,wext
```

Chapter 4. Storage

4.1. Overview

- The MX-V has internal eMMC flash with a boot partition at `/dev/mmcblk2p1` and system root at `/dev/mmcblk2p2`.
- The Micro-SD card shows up as `/dev/mmcblk1` with its partitions as `/dev/mmcblk1p[1..n]`.
- USB based storage connected to the USB host port shows up as `/dev/sd[a..z]`.
- All these can be mounted in the standard way, e.g. `mount /dev/mmcblk1p1 /mnt`.

Chapter 5. Accelerometer

5.1. Overview

The accelerometer is presented as an industrial i/o device(iio). It can be found at `/sys/bus/iio/devices/iio\:device0/in_accel_*`

5.2. Example

```
root@mxv-pt:~# cat /sys/bus/iio/devices/iio\:device0/in_accel_*
0.000000
0.250000 0.500000 1.000000 2.000000
4
4 32 2
50.000000
0.009577
0.038307 0.019154 0.009577
0
18
0
31
0
1037
```

Chapter 6. CAN

6.1. Overview

The CAN interfaces on the MX-V are accessed with the SocketCAN API. This means that a CAN interface is implemented as a type of network interface. They can utilize the Linux network stack and present a programming model similar to TCP/IP.

6.2. Configuration

A CAN device is typically configured with the iproute2 utilities.

6.2.1. List of supported switches

```
$ ip link set can0 type can help
Usage: ip link set DEVICE type can
        [ bitrate BITRATE [ sample-point SAMPLE-POINT] ] |
        [ tq TQ prop-seg PROP_SEG phase-seg1 PHASE-SEG1
          phase-seg2 PHASE-SEG2 [ sjw SJW ] ]

          dphase-seg2 PHASE-SEG2 [ dsjw SJW ] ]

        [ loopback { on | off } ]
        [ listen-only { on | off } ]
        [ triple-sampling { on | off } ]
        [ one-shot { on | off } ]
        [ berr-reporting { on | off } ]
        [ fd { on | off } ]
        [ fd-non-iso { on | off } ]
        [ presume-ack { on | off } ]

        [ restart-ms TIME-MS ]
        [ restart ]

        Where: BITRATE  := { 1..1000000 }
                  SAMPLE-POINT  := { 0.000..0.999 }
                  TQ            := { NUMBER }
                  PROP-SEG      := { 1..8 }
                  PHASE-SEG1    := { 1..8 }
                  PHASE-SEG2    := { 1..8 }
                  SJW           := { 1..4 }
                  RESTART-MS    := { 0 | NUMBER }
```

6.2.2. Configure bit rate of one specific CAN controller

```
$ ifconfig can0 down
$ ip link set can0 type can bitrate 250000
$ ifconfig can0 up
```

6.2.3. Query a device for its current configuration

```
$ ip -d link show can0
```

6.2.4. Query a device for statistics

```
$ ip -s link show can0
```

6.3. CAN-utils

can-utils is a collection of CAN-tools that can be used to debug CAN networks and applications. They can be very useful as a source of inspiration if you are to write your own application that interacts with a CAN network.

6.3.1. Dump all incoming data

```
$ candump -l any,0:0,#FFFFFFFF    (log error frames and also all data frames)
```

6.3.2. Send a single CAN frame

```
$ cansend can0 123#DEADBEEF
```

Chapter 7. Digital I/O

Digital I/O on the MX-V is handled through the Linux GPIO subsystem. This is a generic interface that organizes the GPIO pins by the hardware chips that input or output digital signals. The interface is generic and presents a large number of inputs and outputs in the system, mostly internal.

To use the digital I/O on the MX-V, the new (since Linux v4.8) GPIO API is recommended. This C API is implemented in libgpiod and comes with bindings also for C++ and Python.

The library comes with a set of tools that can be used from a shell script to set and get values.

- gpiodetect - list all gpiochips present on the system, their names, labels and number of GPIO lines
- gpioinfo - list all lines of specified gpiochips, their names, consumers, direction, active state and additional flags
- gpioget - read values of specified GPIO lines
- gpioset - set values of specified GPIO lines, potentially keep the lines exported and wait until timeout, user input or signal
- gpiofind - find the gpiochip name and line offset given the line name
- gpiomon - wait for events on GPIO lines, specify which events to watch, how many events to process before exiting or if the events should be reported to the console

By using gpiodetect, a list of the GPIO chips in the system is shown.

```
root@mxv-pt:~# gpiodetect
gpiochip0 [209c000.gpio] (32 lines)
gpiochip1 [20a0000.gpio] (32 lines)
gpiochip10 [digital_in] (13 lines)
gpiochip2 [20a4000.gpio] (32 lines)
gpiochip3 [20a8000.gpio] (32 lines)
gpiochip4 [20ac000.gpio] (32 lines)
gpiochip5 [20b0000.gpio] (32 lines)
gpiochip6 [20b4000.gpio] (32 lines)
gpiochip7 [modem_control] (4 lines)
gpiochip8 [ncv7751] (32 lines)
gpiochip9 [gpio_overlay] (25 lines)
```

7.1. Digital out

7.1.1. Overview

The digital out channels (lines) have been collected in an `overlay` virtual chip that also restricts the user from changing the direction of the signal (which can usually be done for these generic I/Os).

With the `gpioinfo` command, the available channels are listed.

```
root@mxv-pt:~# gpioinfo gpio_overlay
gpiochip9 - 25 lines:
        line   0: "digital-out-source-0" unused input active-high
        line   1: "digital-out-source-1" unused input active-high
        line   2: "digital-out-source-2" unused input active-high
        line   3: "digital-out-source-3" unused input active-high
        line   4: "digital-out-source-4" unused input active-high
        line   5: "digital-out-source-5" unused input active-high
        line   6: "digital-out-source-6" unused input active-high
        line   7: "digital-out-source-7" unused input active-high
        line   8: "digital-out-oc-0" unused input active-high
        line   9: "digital-out-oc-1" unused input active-high
        line  10: "digital-out-oc-2" unused input active-high
        line  11: "digital-out-oc-3" unused input active-high
        line  12: "digital-out-oc-4" unused input active-high
        line  13: "digital-out-oc-5" unused input active-high
        line  14: "digital-out-oc-6" unused input active-high
        line  15: "digital-out-oc-7" unused input active-high
        line  16: "digital-out-sink-0" unused input active-high
        line  17: "digital-out-sink-1" unused input active-high
        line  18: "digital-out-sink-2" unused input active-high
        line  19: "digital-out-sink-3" unused input active-high
        line  20: "digital-out-sink-4" unused input active-high
        line  21: "digital-out-sink-5" unused input active-high
        line  22: "digital-out-sink-6" unused input active-high
        line  23: "digital-out-sink-7" unused input active-high
        line  24: "digital-out-enable" unused input active-high
```

To find a specific signal, the `gpiofind` command can be used:

```
root@mxv-pt:~# gpiofind digital-out-enable
gpiochip9 24
```

7.1.2. Enable digital out

To enable digital out, the `digital-out-enable` line needs to be set to high.

We can use the `gpioset` command in combination with `gpiofind` in `bash`.

```
root@mxv-pt:~# gpioset $(gpiofind digital-out-enable)=1
```

7.1.3. Digital out high and low

The digital out drivers can both source and sink current. The 8 source drivers are controlled with `digital-out-source-0..7` and the sink drivers with `digital-out-sink-0..7`. For a specific channel, the source and sink drivers cannot be used at the same time.

```
root@mxv-pt:~# gpioset $(gpiofind digital-out-source-0)=1
```

7.1.4. Over-current sense

For each digital-out-source-n there is a digital-out-oc-n which can be read to detect short-circuits.

```
root@mxv-pt:~# gpioget $(gpiofind digital-out-oc-0)
```

7.1.5. Debugging

`/sys/kernel/debug/gpio` can be read to get the kernel's view of the GPIO status.

```
root@mxv-pt:~# cat /sys/kernel/debug/gpio | grep -i dig
 gpio-48  (DIG_OUT_LO_1        |DIG-OUT-LO          ) out hi
 gpio-49  (DIG_OUT_EN          |DIG-OUT-EN          ) out hi
 gpio-64  (DIG_OUT_HI_1        |DIG-OUT-HI          ) out hi
 gpio-65  (DIG_OUT_HI_2        |DIG-OUT-HI          ) out lo
 gpio-66  (DIG_OUT_HI_3        |DIG-OUT-HI          ) out lo
 gpio-67  (DIG_OUT_HI_4        |DIG-OUT-HI          ) out lo
 gpio-68  (DIG_OUT_HI_5        |DIG-OUT-HI          ) out lo
 gpio-69  (DIG_OUT_HI_6        |DIG-OUT-HI          ) out lo
 gpio-70  (DIG_OUT_HI_7        |DIG-OUT-HI          ) out lo
 gpio-71  (DIG_OUT_HI_8        |DIG-OUT-HI          ) out lo
 gpio-72  (DIG_OUT_HI_OC_1     |DIG-OUT-HI-OC       ) in  hi
 gpio-73  (DIG_OUT_HI_OC_2     |DIG-OUT-HI-OC       ) in  lo
 gpio-74  (DIG_OUT_HI_OC_3     |DIG-OUT-HI-OC       ) in  lo
 gpio-75  (DIG_OUT_HI_OC_4     |DIG-OUT-HI-OC       ) in  lo
 gpio-76  (DIG_OUT_HI_OC_5     |DIG-OUT-HI-OC       ) in  hi
 gpio-77  (DIG_OUT_HI_OC_6     |DIG-OUT-HI-OC       ) in  lo
 gpio-78  (DIG_OUT_HI_OC_7     |DIG-OUT-HI-OC       ) in  hi
 gpio-79  (DIG_OUT_HI_OC_8     |DIG-OUT-HI-OC       ) in  lo
 gpio-166 (DIG_OUT_LO_2        |DIG-OUT-LO          ) out hi
 gpio-493 (digital-out-source-0|gpioset             ) in  hi
 gpio-494 (digital-out-source-1)
 gpio-495 (digital-out-source-2)
 gpio-496 (digital-out-source-3)
 gpio-497 (digital-out-source-4)
 gpio-498 (digital-out-source-5)
 gpio-499 (digital-out-source-6)
 gpio-500 (digital-out-source-7)
 gpio-501 (digital-out-oc-0    )
 gpio-502 (digital-out-oc-1    )
 gpio-503 (digital-out-oc-2    )
 gpio-504 (digital-out-oc-3    )
 gpio-505 (digital-out-oc-4    )
 gpio-506 (digital-out-oc-5    )
 gpio-507 (digital-out-oc-6    )
 gpio-508 (digital-out-oc-7    )
 gpio-509 (digital-out-sink-0  )
 gpio-510 (digital-out-sink-1  )
 gpio-511 (digital-out-enable  )
```

Chapter 8. LEDS

8.1. Overview

The LEDs on the MX-V can be controlled using the `sysfs` in `/sys/class/leds`.

8.2. Examples

8.2.1. Testing all LEDs off

```
echo 0 > /sys/class/leds/wifi_led_red/brightness
echo 0 > /sys/class/leds/wifi_led_green/brightness
echo 0 > /sys/class/leds/gps_led_red/brightness
echo 0 > /sys/class/leds/gps_led_green/brightness
echo 0 > /sys/class/leds/func_led_green/brightness
echo 0 > /sys/class/leds/func_led_red/brightness
echo 0 > /sys/class/leds/modem_led_green/brightness
echo 0 > /sys/class/leds/modem_led_red/brightness
```

8.2.2. Testing all LEDs on

```
echo 255 > /sys/class/leds/wifi_led_red/brightness
echo 255 > /sys/class/leds/wifi_led_green/brightness
echo 255 > /sys/class/leds/gps_led_red/brightness
echo 255 > /sys/class/leds/gps_led_green/brightness
echo 255 > /sys/class/leds/func_led_green/brightness
echo 255 > /sys/class/leds/func_led_red/brightness
echo 255 > /sys/class/leds/modem_led_green/brightness
echo 255 > /sys/class/leds/modem_led_red/brightness
```

8.2.3. List trigger for LED

```
cat /sys/class/leds/modem_led_red/trigger
```

cat /sys/class/leds/X/trigger could result in none usb-host rfkill-none cpu cpu0 disk-write heartbeat

8.2.4. add trigger for LED

```
echo heartbeat > /sys/class/leds/modem_led_red/trigger
```

Chapter 9. Modem

9.1. Overview

The MX-V has a modem with a GPS receiver.

9.2. Starting/Enabling modem

The modem is turned of by default. To enable it, set the GPIO control line to 1.

```
root@mxv-pt:~# gpioinfo modem_control
gpiochip8 - 4 lines:
    line   0: "MODEM_ENABLE_ON" unused input active-high
    line   1: "MODEM_RESET" unused input active-high
    line   2: "MODEM_POWER_ENABLE" unused input active-high
    line   3: "MODEM_STATUS_ON" unused input active-high
```

To find out if the modem is on, read the `MODEM_STATUS_ON` GPIO.

```
root@mxv-pt:~# gpioget $(gpiofind MODEM_STATUS_ON)
0
```

Set `MODEM_ENABLE_ON` to 1:

```
gpioset $(gpiofind MODEM_ENABLE_ON)=1
```

- The modem takes about 10 seconds to start and register itself with the kernel.
- Check USB TTY device to see if it has shown up.

```
find /dev -name "ttyUSB*"
```

```
/dev/ttyUSB3
/dev/ttyUSB2
/dev/ttyUSB1
/dev/ttyUSB0
```

- The modem also shows up with the `lsusb` command.

```
root@mxv-pt:~# lsusb | grep Quectel
Bus 001 Device 006: ID 2c7c:0125 Quectel Wireless Solutions Co., Ltd. EC25 LTE modem
```

9.3. Controlling the modem with AT commands

- Use microcom to connect to the modem. (Exit microcom with `CTRL-X`)

```
microcom /dev/ttyUSB2
```

- To get more informative error messages, you can type:

```
AT+CMEE=2
```

Expected reply

```
OK
```

9.4. Test mobile network

- List all current network operators with

```
AT+COPS=?`
```

- After 20–30 seconds, you get something similar to:

```
+COPS: (1,"3 SE","3 SE","24002",7),(1,"Telenor
SE","TelenorS","24008",7),(1,"Tele2","Tele2 SE","24007",7),(1,"TELIA
S","TELIA","24001",7),(1,"TELIA S","TELIA)
```

9.5. Test SIM card

9.5.1. Check if SIM is detected

type

```
AT+QSIMSTAT?
```

expected reply

```
+QSIMSTAT: 0,1
```

or

```
+QSIMSTAT: 0,1
```

0,1 or 1,1 means that SIM is detected

9.5.2. Check status of pin code protection

type

```
AT+CPIN?
```

expected:

```
+CPIN: READY
```

error:

```
+CME ERROR: SIM not inserted
```

9.6. GPS

9.6.1. Enable

Connect to the modem with

```
microcom /dev/ttyUSB2
```

Enable GPS with the AT commands

```
AT+QGPSCFG="autogps",1
AT+QGPS=1
```

9.6.2. Reading values

- Wait until the GPS receiver has found enough satellites to get a fix.
- Get the GPS position with `AT+QGPSLOC?`
- If the GPS receiver has no fix, the Not fixed now error is shown.

```
AT+QGPSLOC?
+CME ERROR: Not fixed now
```

- GPS data can be continuously streamed from `/dev/ttyUSB1`

```
microcom /dev/ttyUSB1
```

Chapter 10. Audio

10.1. Overview

The MX-V has two audio devices that are accessed as standard ALSA sound cards. Using `aplay --list-devices` to list them, they shows up as follows:

```
root@mxv-pt:~# aplay --list-devices

**** List of PLAYBACK Hardware Devices ****
card 0: imxhdmisoc [imx-hdmi-soc], device 0: i.MX HDMI Audio Tx i2s-hifi-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 1: Audiotlv320aic3 [Audio-tlv320aic310x], device 0: 2028000.ssi-tlv320aic3x-hifi
tlv320aic3x-hifi-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

By using `aplay --list-pcms`, all PCMs can be listed:

```
root@mxv-pt:~# aplay -L
sysdefault
    Default Audio Device
surround21
    2.1 Surround output to Front and Subwoofer speakers
surround40
    4.0 Surround output to Front and Rear speakers
surround41
    4.1 Surround output to Front, Rear and Subwoofer speakers
surround50
    5.0 Surround output to Front, Center and Rear speakers
surround51
    5.1 Surround output to Front, Center, Rear and Subwoofer speakers
surround71
    7.1 Surround output to Front, Center, Side, Rear and Woofer speakers
null
    Discard all samples (playback) or generate zero samples (capture)
pulse
    PulseAudio Sound Server
sysdefault:CARD=imxhdmisoc
    imx-hdmi-soc,
    Default Audio Device
surround21:CARD=imxhdmisoc,DEV=0
    imx-hdmi-soc,
    2.1 Surround output to Front and Subwoofer speakers
surround40:CARD=imxhdmisoc,DEV=0
    imx-hdmi-soc,
    4.0 Surround output to Front and Rear speakers
surround41:CARD=imxhdmisoc,DEV=0
    imx-hdmi-soc,
    4.1 Surround output to Front, Rear and Subwoofer speakers
surround50:CARD=imxhdmisoc,DEV=0
    imx-hdmi-soc,
    5.0 Surround output to Front, Center and Rear speakers
surround51:CARD=imxhdmisoc,DEV=0
    imx-hdmi-soc,
    5.1 Surround output to Front, Center, Rear and Subwoofer speakers
surround71:CARD=imxhdmisoc,DEV=0
    imx-hdmi-soc,
    7.1 Surround output to Front, Center, Side, Rear and Woofer speakers
sysdefault:CARD=Audiotlv320aic3
    Audio-tlv320aic310x,
    Default Audio Device
```

10.2. Speaker-test

Testing of the sound can be done with the `speaker-test` ALSA application as follows

```
speaker-test -c 2 -D plughw:1,0
```

Chapter 11. Packages

11.1. Overview

Software packages can be installed into a running system using the `opkg` command.

Host Mobility hosts a public package server on http://hostmobility.org:8008. Individual packages can also be copied to the MX-V using `scp` and installed like this:

```
opkg install <path/to/ipk-package>
```

Chapter 12. RTC

12.1. Network synchronization

If the MX-V has access to Internet, it is synchronized with the NTP protocol.

12.2. Reading

```
hwclock --get
```

12.3. Writing

```
timedatectl set-time "2020-10-09 10:11:54"
hwclock --systohc
```

Chapter 13. GUI

A graphical user interface can be accessed via the DVI connector. Sound output is also available through this port.

The the `x11perf` tool can be used to test a variety of graphics functions:

```
`x11perf --all`
```

Chapter 14. UART, RS-232, RS-485

14.1. Summary

The MX-V has four RS-232 ports at `/dev/ttymxc0,1,2,4` and one RS-485 port at `/dev/ttymxc3`. The first of them, `/dev/ttymxc0`, is the default serial console. The ports is accessed as standard serial ports from Linux.

14.2. uart-test

To help testing, a tool is provided:

```
uart-test <send|receive> <port> <baudrate> <file> <byteCount>
```

The tool can send or receive data

14.2.1. uart-test example

```
# Test sending from one serial port the other and reverse
RS232_LOG1=test1.log
RS232_LOG2=test2.log
echo clean > $RS232_LOG1
echo clean > $RS232_LOG2
RS232_SEND_FILE_PATH1=/tmp/rs232_tx1.bin
RS232_SEND_FILE_PATH2=/tmp/rs232_tx2.bin
echo -n $'123456789' > $RS232_SEND_FILE_PATH1
echo -n $'987654321' > $RS232_SEND_FILE_PATH2
RS2321_RX_TEST_FILE=/tmp/RS2321_rx_file.bin
RS2322_RX_TEST_FILE=/tmp/RS2322_rx_file.bin
UART_RS2321=/dev/ttymxc4
UART_RS2322=/dev/ttymxc1
#UART_RS2320=/dev/ttymxc0 debugger
RS2322_RX_PID=0
RS2323_RX_PID=0


# SYNTAX: uart-test <send|receive> <port> <baudrate> <file> <byteCount>

uart-test receive $UART_RS2322 9600 $RS2321_RX_TEST_FILE 9 >> $RS232_LOG1 &
RS2322_RX_PID=$!
uart-test receive $UART_RS2321 9600 $RS2322_RX_TEST_FILE 9 >> $RS232_LOG2 &
RS2323_RX_PID=$!

uart-test send $UART_RS2322 9600 $RS232_SEND_FILE_PATH1 9 >> $RS232_LOG1
uart-test send $UART_RS2321 9600 $RS232_SEND_FILE_PATH2 9 >> $RS232_LOG2

sleep 1
pkill uart-test

grep "987654321" ${RS2321_RX_TEST_FILE}
if [ $? != 0 ]; then
    RESULT="FAIL"
    echo "$UART_RS2322 receive:FAIL: $(hexdump -C ${RS2321_RX_TEST_FILE})"
else
   echo OK!
fi

grep "123456789" ${RS2322_RX_TEST_FILE}
if [ $? != 0 ]; then
    RESULT="FAIL"
    echo "$UART_RS2321 receive:FAIL: $(hexdump -C ${RS2322_RX_TEST_FILE})"
else
   echo OK!
fi
```

Chapter 15. Upgrading the operating system

15.1. Upgrading from USB memory

To perform a clean installation of the operating system, you need an image of the type *wic.gz*. The image is built using the Yocto system. We currently provide the following images:

- console-hostmobility-image-mxv-pt-jenkins-Mx5-bringup-Mx5-PT-Poky-VERSION.wic.gz
- mobility-image-development-mxv-pt-jenkins-Mx5-bringup-Mx5-PT-Poky-VERSION.wic.gz
- mobility-image-mxv-pt-jenkins-Mx5-bringup-Mx5-PT-Poky-VERSION.wic.gz
- mobility-image-xfce-mxv-pt-jenkins-Mx5-bringup-Mx5-PT-Poky-VERSION.wic.gz

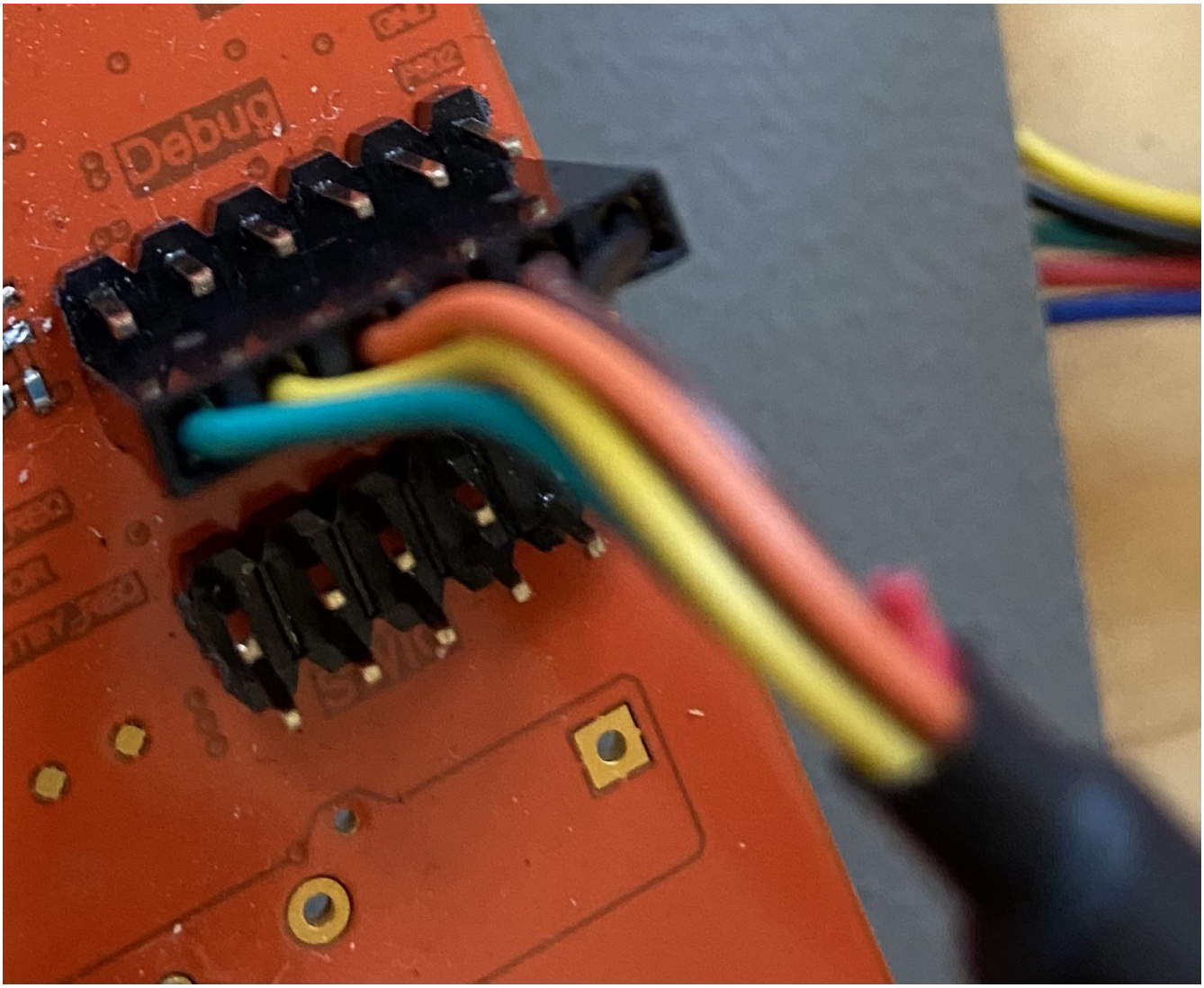
To upgrade

- place the wic.gz file on a FAT32 formatted USB storage device with the image renamed as *mxv-image.wic.gz*
- copy *flashmxv.scr* to the USB device
- insert the USB device into a USB port of the MX-V and press the reset button

It takes a couple of minutes to reprogram the operating system. *DO NOT* power off the system during this process.

15.2. Upgrading over a console

During development, system upgrades can be performed over the serial console using an FTDI serial and USB A-A cable. To to this, connect the FTDI serial cable to the internal pin header (see picture). Also, connect a USB A-A cable between your computer and the USB-OTG port on the MX-V.



A serial terminal program is then used to connect the MX-V terminal. A typical command on GNU/Linux would be e.g. `minicom -D /dev/ttyUSB2`.

Power-cycle the MX-V and hold any key on the keyboard to get a u-boot console.

Enter `mmc dev 2 && mmc partconf 2 1 0 0 && ums 0 mmc 2` and use balenaEtcher to program the MX-V.