MRS MConn7/MConn10

Operating Instructions:

MRS MConn7/MConn10

Contact data

MRS Electronic GmbH & Co. KG
Klaus-Gutsch-Str. 7
78628 Rottweil

Telephone: +49 741 28070
Internet: https://www.mrs-electronic.com
E-Mail: info@mrs-electronic.com

Product

Product designation: MRS MConn7/MConn10

Serial number: See back cover.

CE

Document

Name: MRS MConn7/MConn10
Version: 2.6.1
Date: 05/2023

Our products are developed according to European norms and standards. Therefore, the use of these products is currently limited to the area of the European Economic Area (EEA). If products are to be used in another area, market access research must be carried out beforehand. You can do this yourself as the market introducer or you are welcome to contact us, and we will discuss how to proceed together.

1 General Information

This chapter includes information related to the product and its usage.

1.1 Introduction

The MConn7 and MConn10 are the new generation of Connected Displays of MRS Electronic. They come with a powerful 32-bit multi-core ARM Cortex-A9 processor, 2D, 3D, and Vector Graphics hardware acceleration. Featuring a multi-touch PCAP touchscreen, operators can take advantage of many gestures found on tablets today, including pinch-to-zoom, rotation, flick, and many more.

1.2 Product Details

Following are the details of the product:

Name: MConn7 / MConn10

Version: 2.6.1

1.3 Reference App

The Reference App is an example Qt application from MRS Electronic that comes with the MConn7 and MConn10 Displays. Using Reference App, users can understand how to build applications that use CAN bus, LIN bus, streams from analog and IP cameras, play music files, view PDF files, and so on.

This document refers to the Reference App to better understand the components, interactions, and customization abilities of the system.

1.4 Intended Users

These instructions address trained experts who are familiar with the programming and technical understanding of the software. Trained experts are people who can assess the tasks assigned to them and recognize possible dangers due to their expert training, knowledge, experience, and knowledge of the relevant standards and regulations.

1.5 Document Conventions

This document contains prompt boxes for the users to consider before certain calls to action. The measures for averting risks described in the instructions must be implemented to avoid the risk of property damage or personal injury.

Important	
	A notice of this sort denotes an important message. This prompt indicates any important step or a pre-requisite to be performed prior to a step.

Information	
	A notice of this sort denotes an additional piece of information that can be helpful for the user.

Danger	
	A notice of this sort denotes a danger that may arise.

Warning	
	A notice of this sort denotes a warning message.

Disclaimer	
	A notice of this sort denotes a disclaimer.

1.6 Validity

The validity of these instructions goes into effect with the transfer of the product from MRS Electronic to the customer. Changes to these operating instructions are possible at any time and that too without the specification of any reasons. The current version of this document replaces all previous versions of user manuals ever released.

1.7 Copyright Information

1.8 Technical Support

This section provides you with the information related to technical support from the manufacturer.

For Existing Users:

If you are already using MConn and have an issue with it, contact our Service and Support Team using the details below:

Manufacturer Name	MRS Electronic GmbH & Co. KG
Address	Klaus-Gutsch-Str. 7, 78628 Rottweil, Germany
Telephone	+49 741 28070
Website	www.mrs-electronic.de/en/service-support/support
E-mail	support@mrs-electronic.com

For New Users:

If you are interested in learning more about MConn, contact our team using the details below:

Manufacturer Name	MRS Electronic GmbH & Co. KG
Address	Klaus-Gutsch-Str. 7, 78628 Rottweil, Germany
Telephone	+49 741 28070
Website	www.mrs-electronic.de
E-mail	info@mrs-electronic.de

1.9 Warranty Conditions

See the General Terms and Conditions of MRS Electronic GmbH & Co. KG at www.mrs-electronic.de/en/terms.

2 User Product and Safety

This chapter includes information related to the operation of the product.

2.1 Hazards

The MConn has been programmed with the newest technology and recognized safety-related regulations.

Danger	
	Danger for users and/or property may arise in case of improper use. Lack of compliance with the work safety rules may result in damages.

Some functions of the application are intended for analysis, control, and manipulation of operating electronic systems. Faulty software or parameter settings may cause unforeseen reactions and malfunctions in the system resulting in serious operational disturbance, property damage, and personal injury.

2.2 Staff Qualifications

For a risk-free experience, only skilled or authorized persons should use the product.

Disclaimer	
	MRS Electronic shall not be liable for damages arising from the operation of the products by non-qualified users.

The two user groups entrusted to perform operation tasks are:

Skilled Personnel: Personnel having sufficient knowledge of the subject in question due to their special training. Such persons are familiar with the accident prevention regulations, guidelines, and generally recognized rules of technology. Skilled persons must be capable of securely assessing the results of their work and familiarize themselves with the contents of this user manual.

Authorized Personnel: Users who are permitted to perform the assigned work due to legal regulations or special permission granted by MRS Electronic.

Warning	
	This product should not be used by children and people who are mentally or physically disabled or are inexperienced. Those without enough experience or sufficient knowledge of the product should only use the product under the supervision of skilled and authorized personnel or after attending a detailed training regarding the use of the control unit.

2.3 Consumer Rights and Responsibilities

It is the right of consumers that the manufacturer delivers the product to them in a state where the programming of the control unit does not lead to any safety-related problem of the system in case of a failure or malfunction.

The user must also consider the fact that faulty programming or parameter settings of the control unit may lead to unforeseen signals to the outputs of the control unit or other hazards during operation of the complete system.

3 System Requirements

This chapter includes information related to software and hardware requirements of the MConn device.

3.1 Compatible Software

Following table enlists the compatible software for MConn Display:

Software	Version
Reference App	2.6
Operating System	2.6.1
Kernel	4.19.35
Qt	5.12.9

4 Setting up Development Environment

This section covers the details of setting up your development environment. You can either setup the development environment yourself or get a pre-configured virtual machine from MRS Electronic.

Important
The document solely focuses on the development on Linux host. However, it can be setup on Windows host as well. For more details, contact MRS Electronic.

Information
There are multiple virtualization techniques, but the document focuses on setting up an Ubuntu development environment using VirtualBox.

Download the latest version of VirtualBox according to your device's architecture.

4.1 Pre-Configured Machine

If you are using a pre-configured virtual machine from MRS, you can simply install it and jump to **Getting Started** (Section **5**). The virtual machine is already setup for the development and cross-compiling.

4.2 Manual Configuration

To configure your machine manually, follow the steps below:

1. Install the latest version of Ubuntu.
2. Follow this tutorial to configure your machine.
3. Setup Qt development environment by downloading the latest version of Qt. At the time of writing this document, the MRS Display has Qt 5.12.9 installed. Navigate to this link to find the steps to install Qt5 in Ubuntu.

Information
Depending on your deployment strategy, you can either purchase Qt Commercial or begin development using the Open-source version of Qt, which is free. Consult your lawyers regarding the suitable version for your application.

4. Run the installer to begin the installation process.

Important
To attain the steps to set up a cross-compile toolchain for Qt to allow the deployment of application to the Display, contact MRS Electronic.

The development environment is now all set.

5 Getting Started

This section covers the details of some components that help you in getting started with the Display.

5.1 Connector Pins

The Display comes equipped with an **AMPSEAL 776164-1** mating connector.

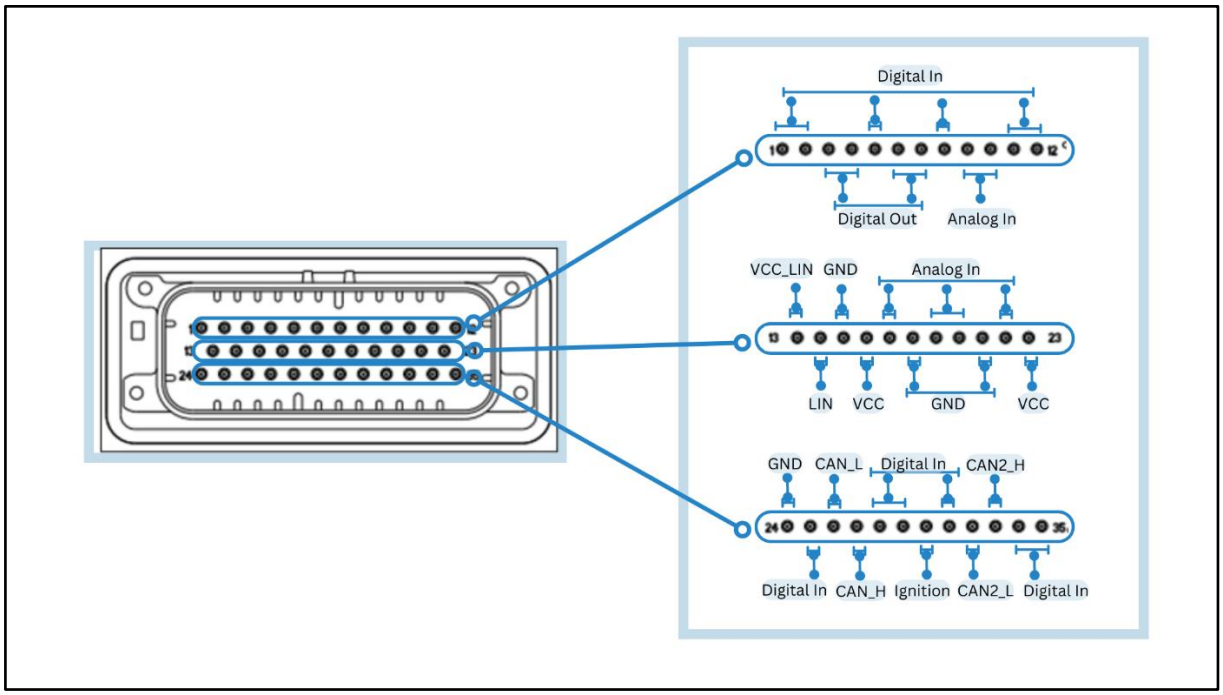

Figure 1: Mating Connector: AMPSEAL 776164-1 with Pin Configurations

Each of the pins hold a specified connection that is described in the table below:

Pin	Function	Pin	Function	Pin	Function
1	Digital In	**13**	VCC_LIN	**25**	Digital In
2	Digital In	**14**	LIN	**26**	CAN1_L
3	Digital Out	**15**	GND	**27**	CAN1_H
4	Digital Out	**16**	VCC	**28**	Digital In
5	Digital In	**17**	Analog In	**29**	Digital In
6	Digital Out	**18**	GND	**30**	Ignition
7	Digital Out	**19**	Analog In	**31**	Digital In
8	Digital In	**20**	Analog In	**32**	CAN2_L
9	Analog In	**21**	GND	**33**	CAN2_H
10	Analog In	**22**	Analog In	**34**	Digital In
11	Digital In	**23**	VCC	**35**	Digital In

12	Digital In	**24**	GND		

5.2 Power Supply

To power up the Display, it is important to ensure that you have the proper connections in your mating connector. Make the connection in the following manner:

1. Connect pins 16, 23, and 30 to +12V.
2. Connect pins 15, 18, 21, and 24 to GND.

5.3 Connection to Display

This section covers the details of the methods to connect to the Display such as Wi-Fi and Ethernet.

5.3.1 Wi-Fi Connection

By default, the Display operates like an Access Point. To connect via Wi-Fi, follow the steps below:

1. Search for a network with SSID: *MRS-Display-xxx*. The last section of the name could be any randomly-generated string.

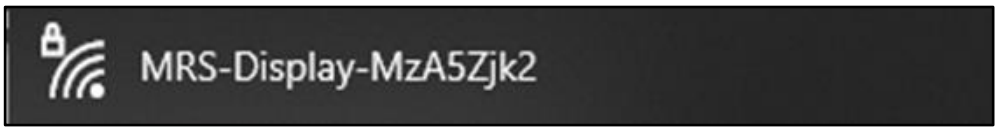

Figure 2: Connecting to Display via Wi-Fi

2. Connect to the network using password: *12345678*.

5.3.2 Ethernet Connection

To connect via Ethernet, simply plug the **RJ45** Ethernet plug at the back of **RJ45** receptacle.

5.3.3 SSH/SFTP Connection

To access the terminal or perform actions like sharing of files, the connection to Display via **SSH/SFTP** is required. The details of that are shown in the table below:

Field	Value
IP Address (Wi-Fi Connection)	192.168.87.1
IP Address (Ethernet Connection)	Router DHCP server
Port	22
Username	root

Password	mrsroot

5.4 File System

The main file system operates in *read-only* mode as it is safer and faster. However, for actions like copying the user application or enabling a *systemd service*, the *read/write* mode is required. To switch to that mode, follow these steps:

1. Type the following terminal command:

```
$ mount -o remount,rw /
```

2. After writing, issue the *sync* command to complete the process of writing to the main file and switch back to the *read-only* mode.

```
$ sync

$ mount -o remount,ro /
```

5.5 Read/Write Partition

The main file system operates in *read-only* mode. However, there is a dedicated partition for user data and operates in *read/write* mode. This partition also contains some important configuration files that you can change and is 1.2GB in size.

To navigate to this location, enter the following commands:

```
$ ls /

$ rw_data
```

Important	
	After writing files to the *read/write* partition, ensure to issue the **sync** command to complete the process of writing to the partition.

5.6 Operating System

This section covers the steps of finding the current version of the Display's Operating System (OS) and updating it.

5.6.1 Current Version

Enter the following commands in the terminal to find the current version of the OS:

```
$ get-os-version

$ System version:  2.6.1
```

To access the information about OS version in the **Reference App**, navigate to the **About** tab as shown below:

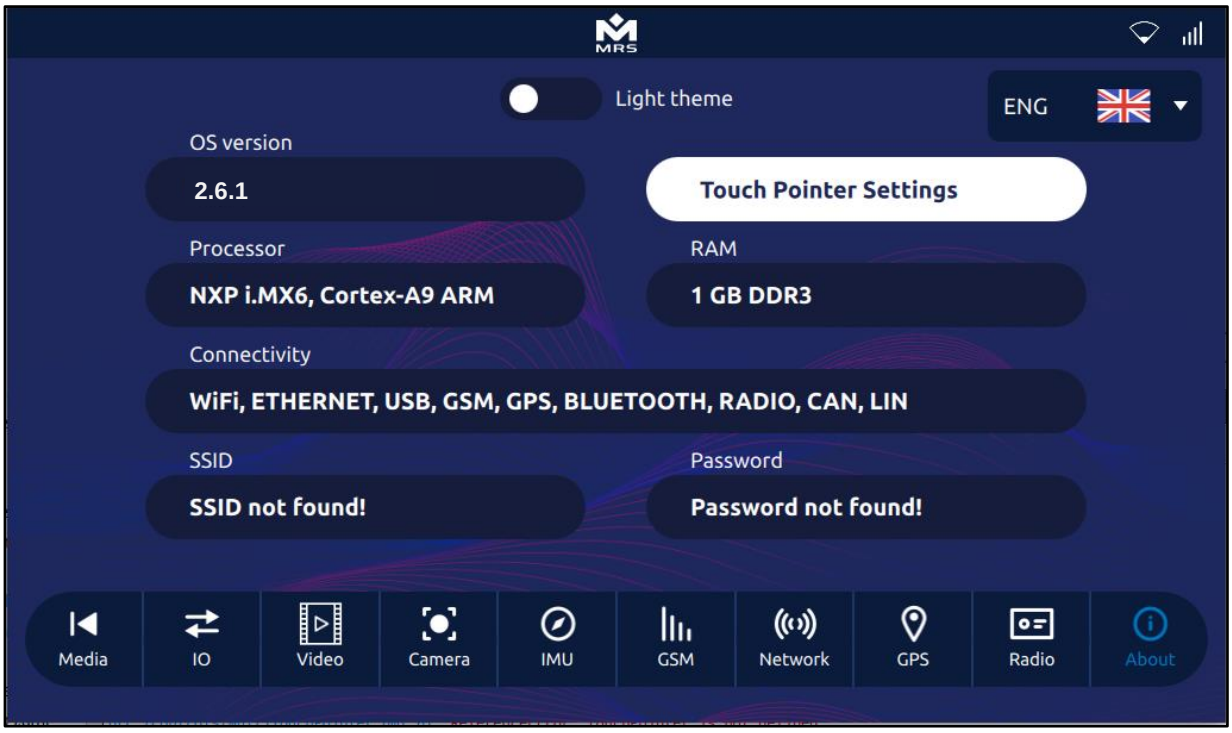

Figure 3: OS Version information in the Reference App

5.6.2 OS Updates

The OS Update feature has been introduced from OS 2.5.0 version. You can update to the latest versions using a USB flash drive. For further information, contact MRS Electronic.

5.7 Screen Setup

This section covers the details of setting up your screen such as touch screen calibration and backlight control.

5.7.1 Touch Screen Calibration

To calibrate the touch screen, run the **ts_calibrate** utility from the terminal as shown below:

```
$ mount -o remount,rw /
$ ts_calibrate
$ mount -o remount,ro /
```

Once done, the calibration utility shows on the display and prompts you to touch the crosshair on the touch screen. When all the touch points are satisfied, the utility terminates and saves the calibration file.

5.7.2 Backlight Control

The brightness of the backlight can be controlled programmatically or from the terminal. The backlight has 100 steps of brightness (0-100), with 0 being the lowest and 100 being the highest.

To control the backlight from the terminal, use the following command:

```
$ echo 100 > /sys/class/backlight/backlight/brightness
```

To control the backlight in the **Reference App** via C++, add the following to the code:

```
io->set_backlight(100); // Set backlight to 100%

io->set_backlight(50);  // Set backlight to 50%
```

To control the backlight of the device from the **Reference App**, use the slider bar in the left as shown below:

Figure 4: Backlight Control in the Reference App

6 Inputs and Outputs

This section covers the details of inputs and outputs of the Display.

6.1 Inputs

The Display handles two types of inputs such as digital and analog inputs.

6.1.1 Digital Inputs

There are twelve **0-12V** digital inputs on the Display. The table below shows the General-Purpose Input/Outputs (GPIOs) assigned to the digital inputs on **AMPSEAL** connector.

Pin	GPIO	Pin	GPIO
1	91	**25**	140
2	117	**28**	2
5	124	**29**	145
8	134	**31**	116
11	139	**34**	122
12	144	**35**	136

The state of the digital inputs can be read through C++ code or terminal commands. To read the state of digital inputs from terminal, enter the following commands:

```
$ echo 91 > /sys/class/gpio/export

$ echo in > /sys/class/gpio/gpio91/direction

$ cat /sys/class/gpio/gpio91/value // Read the value of the input
```

To read the state of digital inputs in the **Reference App** via C++, use the following code:

```
// Reference App I/O Module

int input = io->get_channel_value(“INPUT1”); // Read Digital Input
```

6.1.2 Analog Inputs

There are six **0-12V** analog inputs on the Display. The states of analog inputs are read by the Coprocessor (Read more about it in **Coprocessor** (Section **8.5**)) and then passed to the main processor via UART. The assigned pins for analog inputs on AMPSEAL connector are 9,10, 17, 19, 20, and 22.

The state of the analog input can only be read through C++ code in the **Reference App** by reading and parsing the data from UART. Use the code below to read analog inputs via C++:

```
analog_input = new adc("/dev/ttymxc4");

connect(analog_input, SIGNAL(adc_ready(int, int)), this, SLOT(adc_ready(int, int)));
```

```
void demoApp::adc_ready(int channel, int value)
{
   qDebug() << “adc” << channel << “ = “ << value;
}
```

6.2 Digital Outputs

There are four digital high-side drivers on the Display capable of sourcing up to 2A each. The table below shows the General-Purpose Input/Outputs (GPIOs) assigned to the digital outputs on the **AMPSEAL** connector.

Pin	GPIO	Pin	GPIO
3	162	**6**	167
4	163	**7**	200

The digital outputs can be controlled through C++ code or terminal commands.

To control it from terminal, enter the following commands:

```
$ echo 198 > /sys/class/gpio/export
$ echo out > /sys/class/gpio/gpio162/direction
$ echo 1 > /sys/class/gpio/gpio162/value      // Turn output on
$ echo 0 > /sys/class/gpio/gpio162/value      // Turn output off
```

Use the code below to control the digital outputs in the **Reference App** via C++:

```
analog_input = new adc("/dev/ttymxc4");
io->set_channel_value(“OUTPUT3”, true);       // Turn output on
io->set_channel_value(“OUTPUT3”, false);      // Turn output off
```

7 Sensors

The MConn is equipped with several sensors that can be calibrated in your software application.

7.1 IMU – Accelerometer

The **LSM9DS1** IMU is embedded on the Display's PCB. It is a 9DOF chip and comes with the following:

- 3D digital linear acceleration sensors
- 3D digital angular rate sensors
- 3D digital magnetometer sensors

It uses **I2C-1** serial bus interface to communicate with the Display processor. In Qt application, the **RTIMULib** is used to communicate with the device. The library requires a settings file to run properly. This file includes general IMU settings and those related to the IMU chip (**LSM9DS1**) in use.

The library contains a calibration document that includes calibration values, which can be used to remove IMU soft and hard iron disturbances.

The C++ implementation of **RTIMULib** in the **Reference App** is as follows:

```
RTIMUSettings *settings = new RTIMUSettings("RTIMULib");

RTIMU *imu = RTIMU::createIMU(settings);

imu->IMUInit();

imu->setSleepPower(0.02);

imu->setGyroEnable(true);

imu->setAccelEnable(true);

imu->setCompassEnable(true);

while (imu->IMURead())

{

            RTIMU_DATA imuData = imu->getIMUData();

}
```

To find the output of IMU Sensor in the **Reference App**, navigate to the **IMU** tab:

Figure 5: IMU Sensor's Output in the Reference App

7.2 Real-Time Clock

The Display comes equipped with a Real-Time Clock (RTC) that is powered by a 3V rechargeable battery. The RTC keeps track of the current time when there is no power to the Display and re-syncs with the Linux OS when the Display is powered on.

To set up the RTC, enter the following commands in the terminal:

```
$ date -s "2021-07-22 17:00:00"

$ hwclock -w
```

8 Embedded Devices

The Display is equipped with several embedded devices, which add multiple functionalities to it. This section covers the description of each of those embedded devices.

8.1 Buzzer

The Display comes with a programmable buzzer. It is typically used to indicate any fault or warning from the system to the user. The buzzer can be configured via C++ code or from the terminal.

Enter the following commands to configure the buzzer from the terminal:

```
$ echo 0 > /sys/class/pwm/pwmchip1/export

$ echo 1000000 > /sys/class/pwm/pwmchip1/pwm0/period

$ echo 500000 > /sys/class/pwm/pwmchip1/pwm0/duty_cycle

$ echo 1 > /sys/class/pwm/pwmchip1/pwm0/enable // Turn buzzer on

$ echo 0 > /sys/class/pwm/pwmchip1/pwm0/enable // Turn buzzer off
```

Use the following code to configure the buzzer in the **Reference App** via C++:

```
io = new digital_io();

io->set_buzzer_freq(frequency)

io->set_buzzer(true);    // Turn buzzer on

io->set_buzzer(false);   // Turn buzzer off
```

To control the volume of the buzzer from the **Reference App**, use the slide bar on the right side:

Figure 6: Buzzer Volume Control in the Reference App

8.2 Wi-Fi Module

In some configurations, the Display comes equipped with an **802.11b/g/n** module that can be set up as an Access Point (AP) or a client. By default, the Wi-Fi module is set up as an AP, broadcasting the SSID: *MRS-Display-xxx* with a passphrase: *12345678*.

To switch between AP and Station mode in the **Reference App**, tap the toggle button on **Wi-Fi Settings** screen:

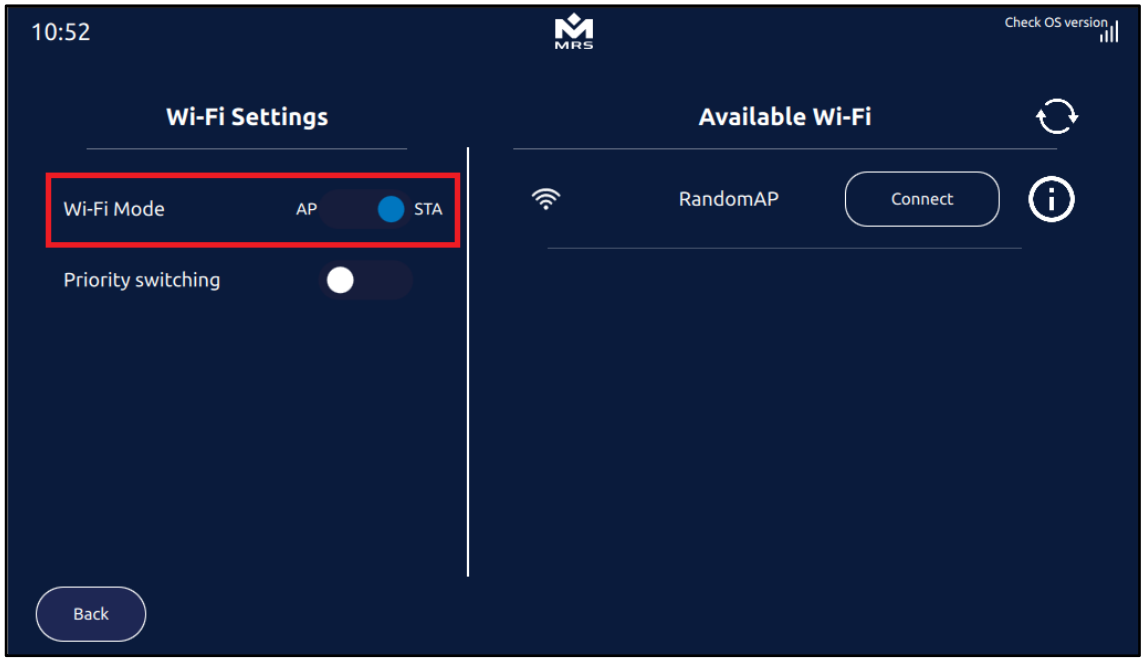

Figure 7: Switching Between AP and STA Mode

8.2.1 Access Point Mode

To setup the Wi-Fi in the Access Point (AP) mode, follow the steps below:

1. Edit the configuration file for the AP mode:

```
$ vi /etc/wpa_supplicant.conf
```

2. Enter the following terminal commands to enable the AP mode:

```
$ onebox-set-mode ap
$ systemctl restart onebox-protocol@3
$ systemctl restart network-wireless
```

Now, you will be able to connect a client to the Display.

8.2.2 Station Mode

To setup the Wi-Fi in the Station (STA) mode, follow the steps below:

1. Edit the configuration file for the STA mode:

```
$ vi /etc/sta_settings.conf
```

2. Enter the following terminal commands to enable the STA mode:

```
$ onebox-set-mode sta
$ systemctl restart onebox-protocol@3
$ systemctl restart network-wireless
```

3. Use the command *ifconfig* to ensure the connection of Display to the Access Point.

Information	
	There are numerous settings that can be configured in the config files. For a full datasheet of the Wi-Fi module, contact MRS Electronic.

8.3 Radio Tuner

In some configurations, the Display is equipped with a high-performance AM/FM tuner. MConn tunes to the station within the threshold of field strength when the scan for FM/AM station is made. The **Alsa** library in the **Reference App** controls the selection between audio output from Linux OS and the AM/FM radio tuner.

To control it via C++ in the **Reference App**, use the following code:

```
//Reference App tuner module
Tuner *tuner = new Tuner();
engine.rootContext()->setContextProperty("tuner", tuner);
```

To access the radio from the **Reference App**, navigate to the **Radio** tab:

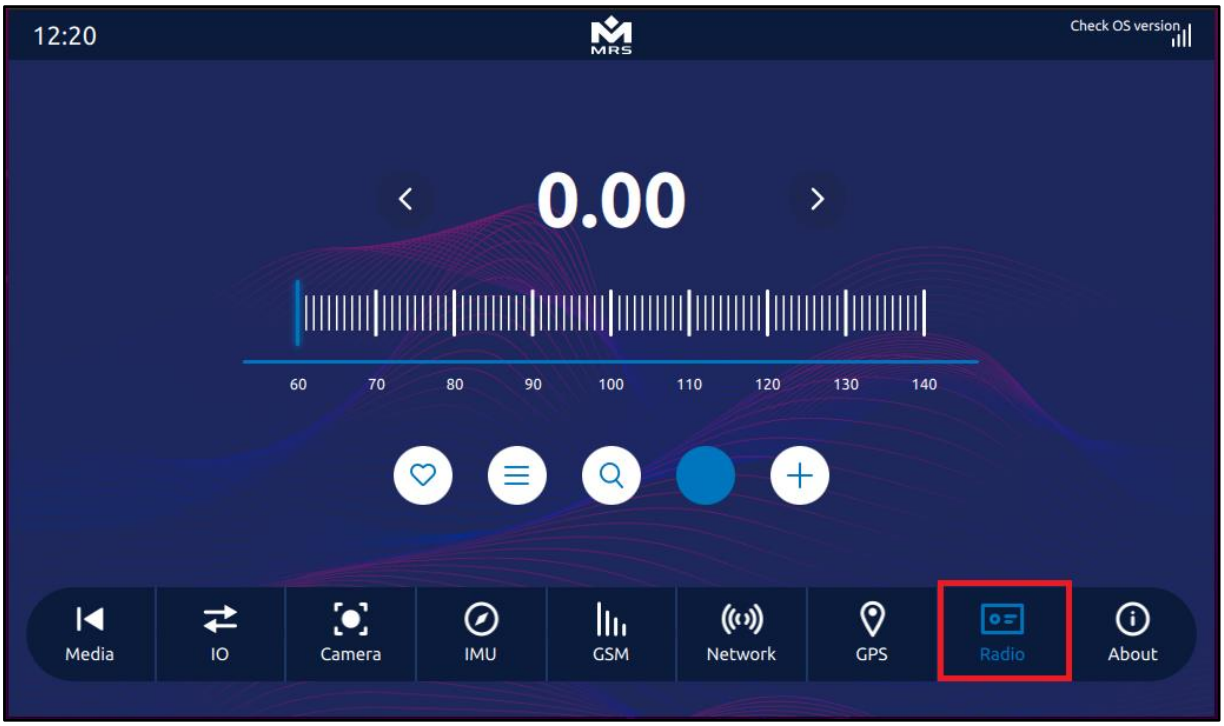

Figure 8: Radio Tab in the Reference App

8.4 Coprocessor

The MConn Display is equipped with a coprocessor that performs multiple functions, including the following:

1. Reads analog inputs.
2. Serves as a watchdog to ensure that the main processor does not lock up.
3. Keeps the main processor alive when the ignition signal goes low, and it can be done by turning on **KEEP_ALIVE** output on coprocessor. It allows shutting down of the main processor in a delayed and controlled manner.
4. Sends/Receives messages on the CAN bus.
5. Asserts specific commands/data on CAN bus immediately at power on. The main display can take at least 5 seconds to boot. Whereas the coprocessor is instant-on and can perform CAN functions at startup if necessary. Contact MRS Electronic for more details.
6. Holds a CAN bootloader and can be reprogrammed through the CAN bus.

Information	
	You can update the software on the coprocessor by using the **MRS Developers Studio**, which allows you to write the software in C or using Graphical Programming. For the source code of the application that currently resides on the Display, contact MRS Electronic.

8.5 GPS

In some configurations, the Display comes equipped with a GPS receiver that can support concurrent reception of up to 3 GNSS (**GPS, Galileo, GLONASS, BeiDou**). The GPS receiver provides the location of the Display using **NMEA** data (UART interface on the Display) that is available on the path: */dev/ttymxc1*.

Use the **QtSerialPort** Library to read the GPS data in the **Reference App** using C++ code:

```
//Reference App gps module

gps *m_gps = new gps(m_version, &app);

m_gps->start();
```

To access the GPS information in the **Reference App**, navigate to the **GPS** tab:

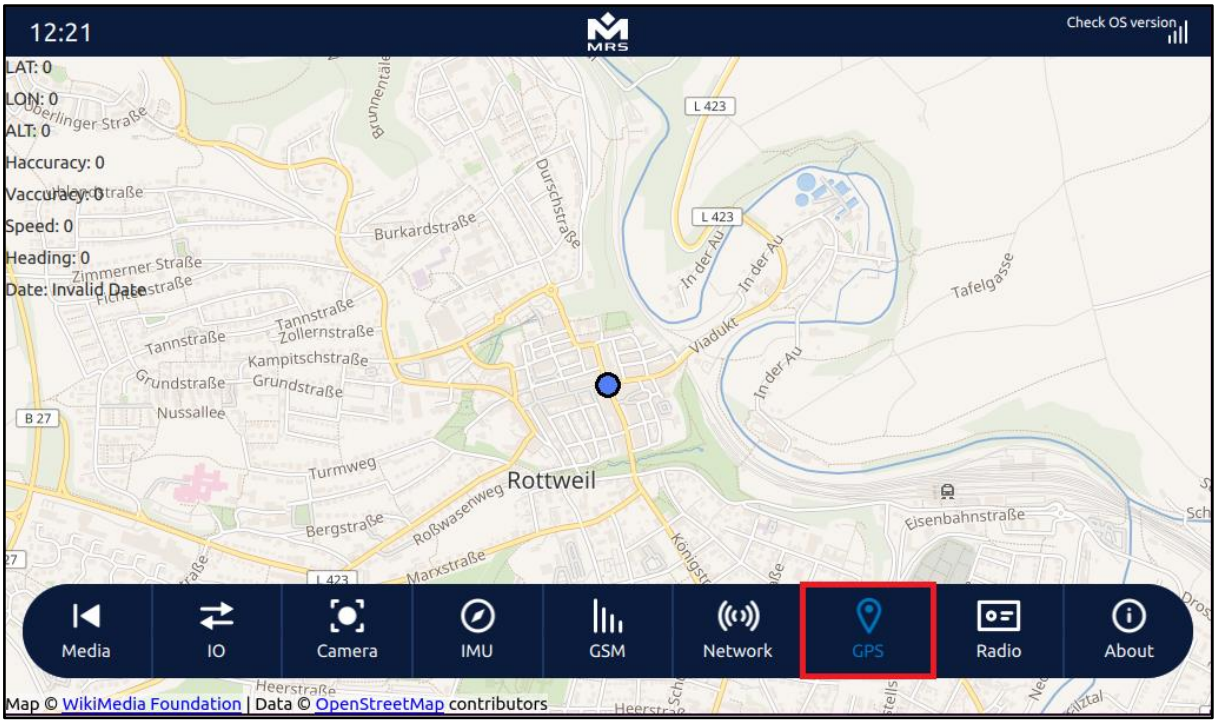

Figure 9: GPS Tab in the Reference App

8.6 Bluetooth

The display comes with **Bluetooth 4.0** and **BlueZ** software. The Bluetooth can be controlled via C++ code and from the terminal.

However, for using Bluetooth in the **Reference App**, follow the steps below:

1. Pull down the notification menu from the top.
2. Long press the **Bluetooth** card.

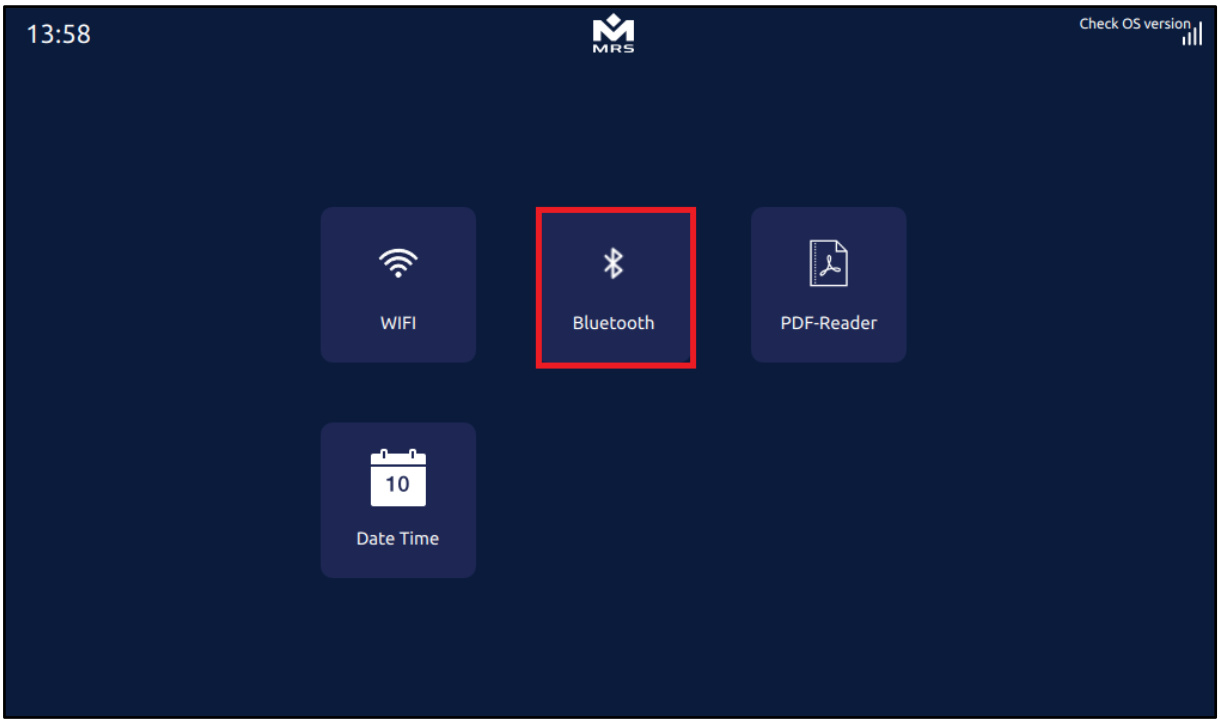

Figure 10: Bluetooth card in the Reference App

It will navigate you to the Bluetooth tab.

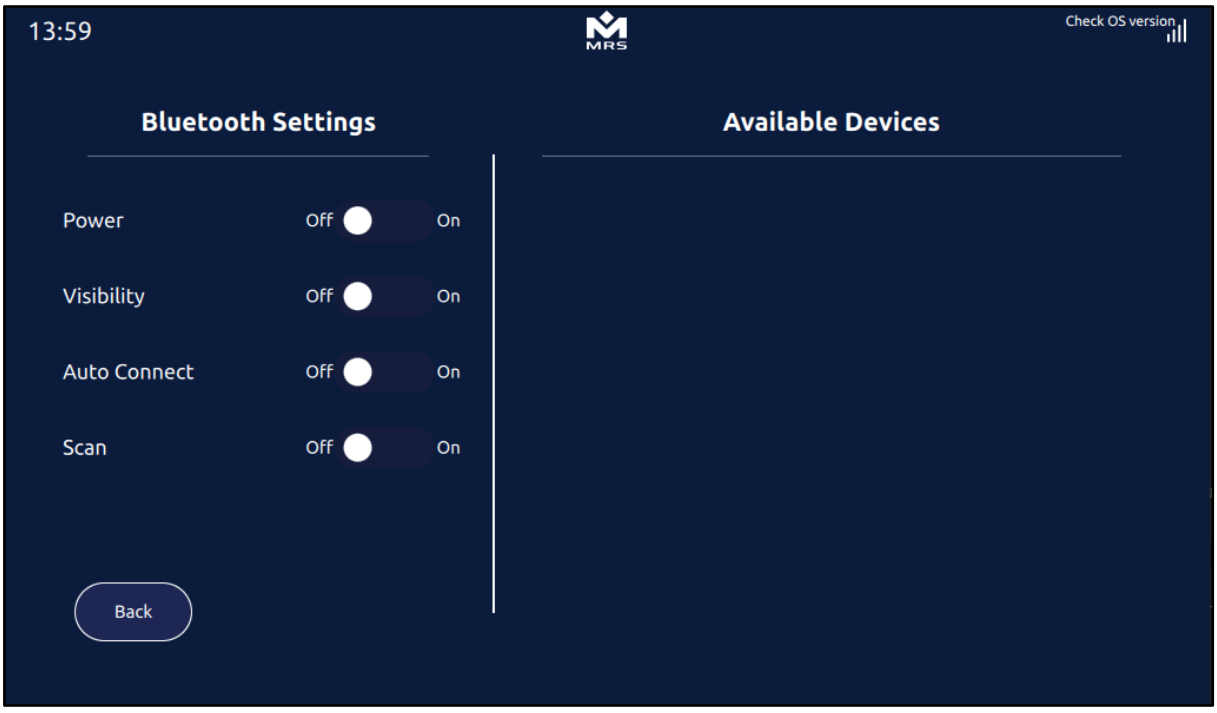

Figure 11: Bluetooth Settings Tab

This menu allows you to set the **Bluetooth Settings**. The description of the available toggles is as follows:

Power

The **Power** toggle allows the MConn to connect with the available Bluetooth devices. To do so, tap on the **Power** toggle to turn it On.

Doing so scans for the Bluetooth devices ready to accept the incoming connection requests within the vicinity of MConn. The available devices list on the right side of the tab.

To pair with any of the available devices, tap on the Pair icon on the right side of device name. This initiates a pairing request. When the request is accepted by the desired device, the Bluetooth pairing process completes.

Visibility

The **Visibility** toggle determines if the device should be visible to the other Bluetooth devices. Turn on the **Visibility** toggle to allow the nearby devices sending a pairing request to the MConn.

Auto Connect

The **Auto Connect** toggle allows the MConn to automatically connect to previously paired devices, once they are within the range.

Scan

The **Scan** toggle forces the Bluetooth module to re-scan for nearby devices.

The Bluetooth connection allows to stream audio data to MConn over the **A2DP** profile. For this purpose, a dedicated Bluetooth media player is utilized. Read more in **Media Playback** (Section **12.9**).

9 Peripheral Interfaces

The MConn Display is equipped with several peripheral interfaces that allow connection to various devices and components. This section covers the details of all those interfaces.

Following diagram shows all the peripheral interfaces of MConn Display (full configuration):

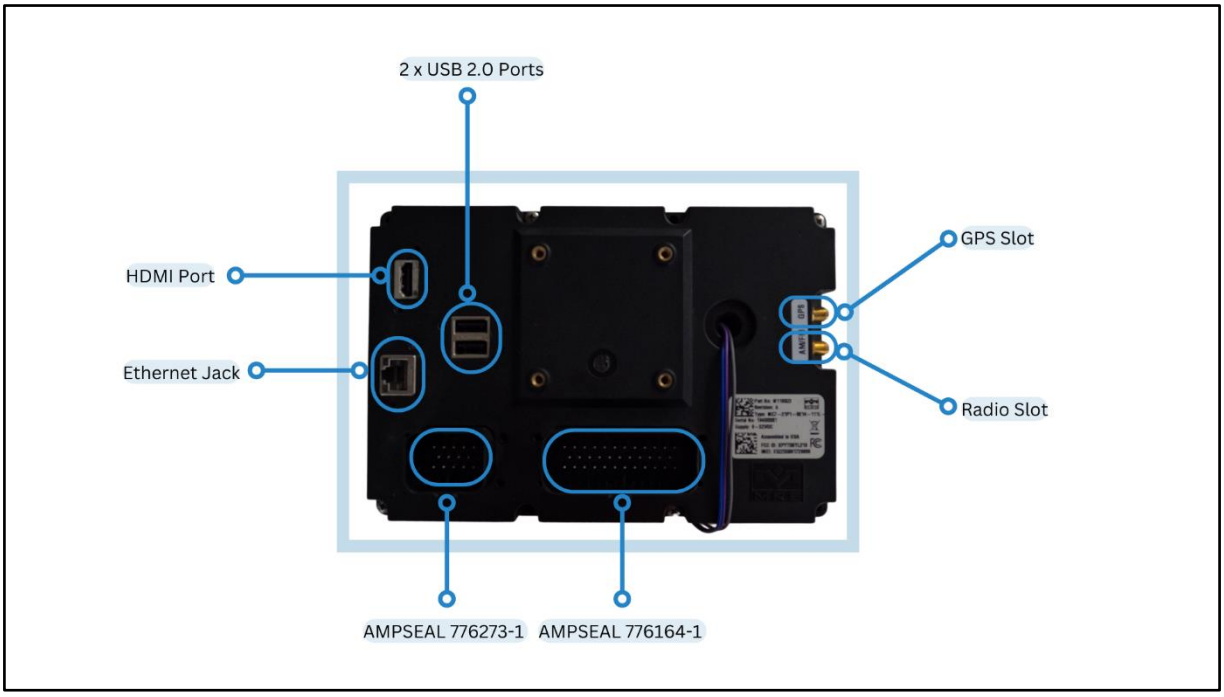

Figure 12: Peripheral Interfaces of MConn Display

9.1 SD Card

The Display holds an SD card slot that serves as an additional space to store user data. If the SD card is pre-installed, the system mounts the SD card automatically. The steps to write files on the SD card are as follows:

1. Find the SD card at the following address:

```
$ ls /

$ sdcard
```

2. Enable the read/write mode. The details of this have been covered in the **File System** (Section **5.4**).

3. Write the files to the SD card as per the requirement.

4. Enter the *sync* command to complete the writing process.

9.2 USB

The Display comes with two **USB 2.0** host ports that serve as an interface for a wide variety of devices; for example, storage devices. When the USB connects, the system mounts the USB storage automatically. The steps to write files on USB storage are as follows:

1. Find the USB storage at the following address:

```
$ ls /run/media/

$ sda1
```

2. Enable the read/write mode. The details of this have been covered in the **File System** (Section **5.4**).

3. Write the files to the USB storage as per the requirement.

4. Enter the *sync* command to complete the writing process.

9.3 CAN

The Display comes equipped with two CAN bus interfaces. By default, the baud rate of both CAN bus interfaces is configured at 250Kbit/s. The baud rate settings for CAN bus interfaces are as follows:

- The baud rate of the CAN-1 bus interface cannot be changed directly.
- The CAN-2 bus interface can be configured to the baud rates of up to 1Mbit/s.

To change the baud rate of interfaces, follow the steps below:

1. Enter the following terminal commands:

```
$ ifconfig can0 down

$ ip link set can0 type can bitrate <your-baudrate> triple-sampling on restart-ms 500

$ ifconfig can0 up
```

The changes made by terminal commands will be lost after reboot. For permanent changes, update the files in */etc/systemd/network* directory.

A code example of interaction with the CAN bus in C++ can be found in the *can.cpp* class of the **Reference App**. This class sends and receives data from both CAN bus interfaces using the **SocketCAN API**. Navigate to the kernel documentation to find more information about the **SocketCAN API**.

Information	
	In addition to the **SocketCAN API**, there is also a **Qt Serial Bus API** that implements the CAN Bus functionality. The **Reference App** does not use this method, but you can find more information about it at this link.

To access the CAN output in the **Reference App**, navigate to the **Network** tab:

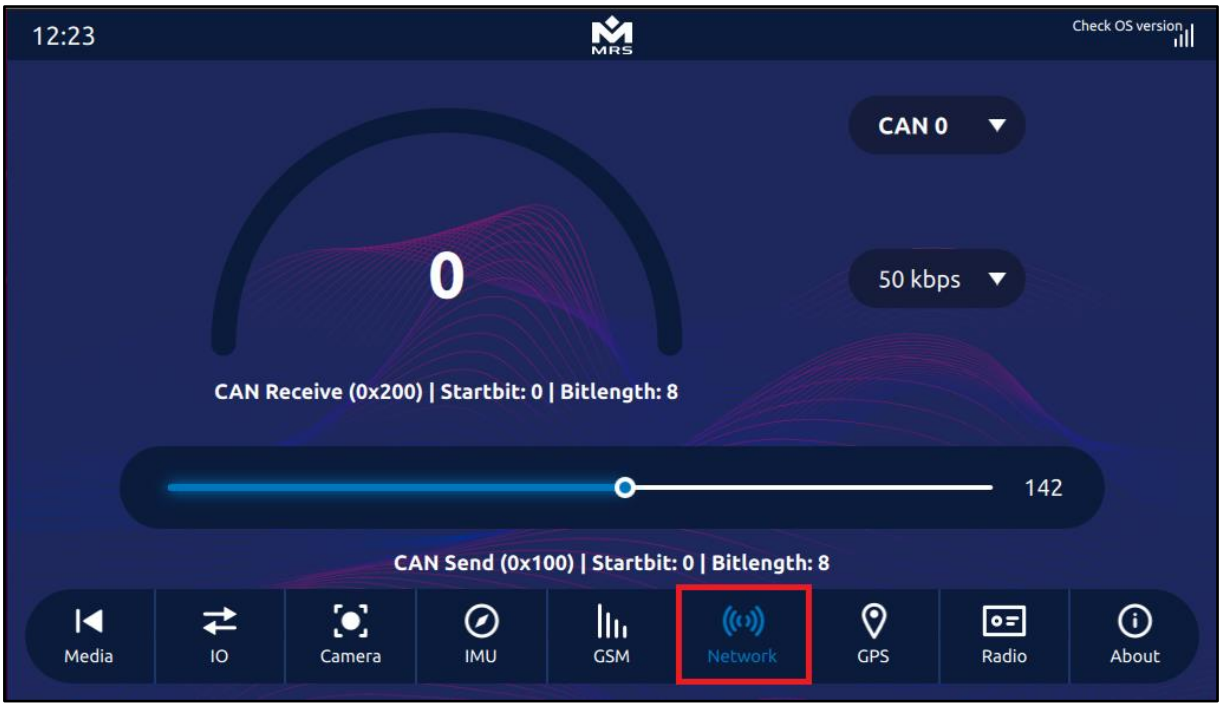

Figure 13: Display of CAN Output in the Reference App

9.4 LIN

The Display is equipped with **TJA1021 LIN** transceiver. One channel LIN bus is located at Pin-14 of the Display mating connector **AMPSEAL 776164-1**.

Information	
	To configure the LIN bus, you can program the coprocessor by using **MRS Developers Studio**, which allows you to write the software in C or using Graphical Programming. For this purpose, contact MRS Electronic.

9.5 Ethernet

The Display holds one **Gigabit Ethernet** interface that allows connection to the LAN network. The ethernet interface can be configured in either DHCP client or DHCP server mode.

Important	
	By default, the Ethernet interface is configured in DHCP Server mode. It implies that you can access the MConn device via the default IP address of DHCP Server: *192.168.57.1*

To configure the ethernet, use the following terminal commands:

```
// For help and possible modes

$ eth-conf.sh -h

// configuring as DHCP server

$ eth-conf.sh -s
```

```
// configuring as DHCP client

$ eth-conf.sh -c
```

9.6 Camera

This section covers the details of the camera interfaces on the Display.

9.6.1 Analog Camera

Certain models of the Display come with four analog camera inputs that can be used to display the view all around the vehicle and save the recorded video to the disk. Using **GStreamer**, the live video from all four cameras can be displayed on the screen.

It is important to ensure that the correct video pin is connected to the mating connector **AMPSEAL 776273-1** before executing any command.

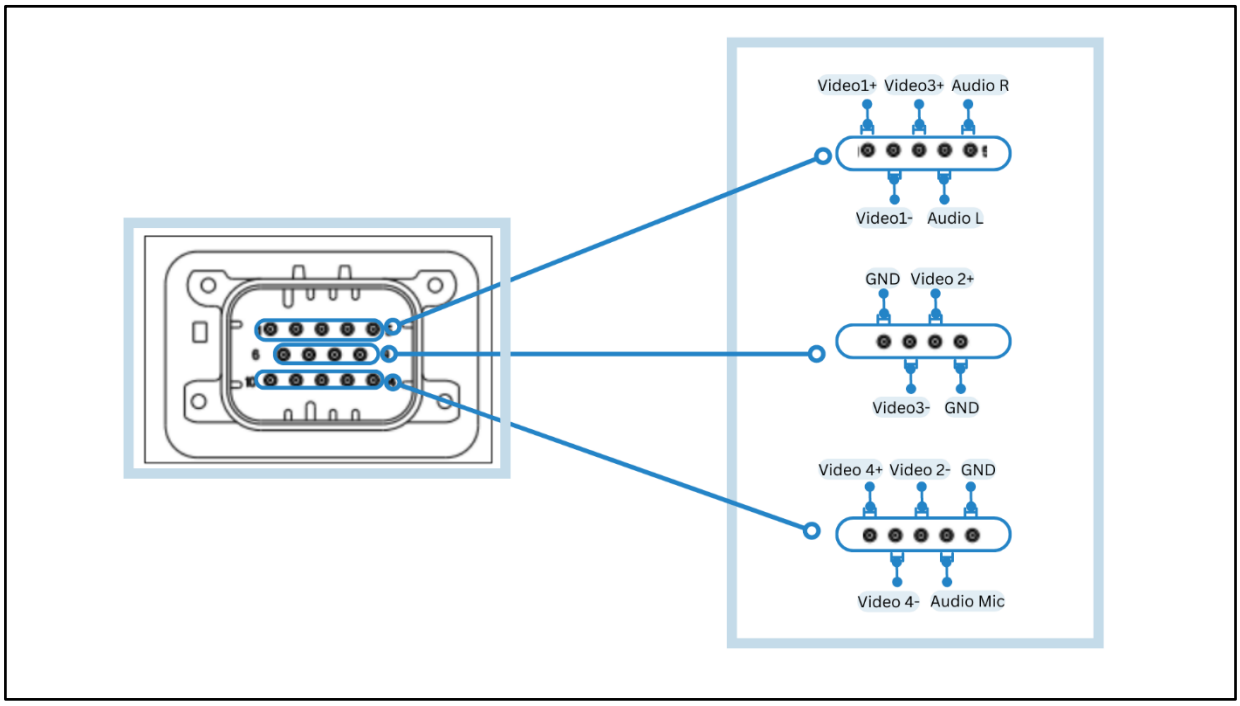

Figure 14: Mating Connector: AMPSEAL 776273-1 with Configurations

The table below maps out the pins for camera connection:

Pin	Function	Pin	Function
1	Video1+	**8**	Video2+
2	Video1-	**10**	Video4+
3	Video3+	**11**	Video4-
7	Video3-	**12**	Video2-

To show the live view of cameras on the screen, use the following commands:

```
$ gst-launch-1.0 imxv4l2videosrc device=/dev/video0 ! imxg2dvideosink window-width=512
window-height=300 window-x-coord=0 window-y-coord=0 & > /dev/null 2>&1 &
```

```
$ gst-launch-1.0 imxv4l2videosrc device=/dev/video1 ! imxg2dvideosink window-width=512
window-height=300 window-x-coord=512 window-y-coord=0 & > /dev/null 2>&1 &

$ gst-launch-1.0 imxv4l2videosrc device=/dev/video2 ! imxg2dvideosink window-width=512
window-height=300 window-x-coord=0 window-y-coord=300 & > /dev/null 2>&1 &

$ gst-launch-1.0 imxv4l2videosrc device=/dev/video3 ! imxg2dvideosink window-width=512
window-height=300 window-x-coord=512 window-y-coord=300 & > /dev/null 2>&1 &
```

To access the analog camera in the **Reference App**, navigate to the **Analog** view in the **Camera** tab:

Figure 15: Display of the Analog Camera View in the Reference App

9.6.2 Ethernet Camera

The Display supports ethernet camera with H.264 video compression. To view the video stream, use the **GStreamer** pipeline. However, before launching the pipeline, ensure that the camera is sending the UDP/RTP stream and is connected to the Display. For detailed information and C++ implementation, refer to the *CameraItem* module of the **Reference App**.

To view the video stream via **GStreamer** pipeline, use the following commands:

```
$ gst-launch-1.0 -v udpsrc port=5014 caps="application/x-rtp, media=(string)video, encoding-
name=(string)H264 , payload=(int)96" ! rtpjitterbuffer ! rtph264depay ! h264parse ! imxvpudec
! videoconvert ! imxg2dvideosink
```

To access the ethernet camera in the **Reference App**, navigate to **Ethernet** view in the **Camera** tab:

Figure 16: Display of the Ethernet Camera View in the Reference App

9.7 In Audio

Certain models of the Display are equipped with a stereo audio output and a microphone input. There is also an audio switch available to select between the audio output from the Linux OS and the AM/FM radio tuner (available only for certain models).

The table below defines the pins for audio input and output on the mating connector **AMPSEAL 776273-1:**

Pin	Function	Pin	Function
4	Audio L	**13**	Audio Mic
5	Audio R		

There are several ways to play an audio file on the Display, but the simplest method is to play a *.wav* file using **aplay** (available in the **ALSA** utility library).

Use the following commands to play a *.wav* file from the terminal:

```
$ aplay test.wav
```

To make a ten second recording, use the following terminal command:

```
$ arecord -d 10 /tmp/test-mic.wav
```

The MConn comes with ***alsaloop*** tool. It is a command-line tool that allows you to create loopback connections between ALSA devices. These devices include the microphone input and a speaker output. It enables you to listen on the connected speakers what is being picked up by the microphone.

To use ***alsaloop***, enter the following terminal command:

```
bash

alsaloop -t 500000
```

The above command configures the routing of output from default audio capture device to the default audio playback device. The ***-t*** flag is used to define the lag parameter. It is necessary to define this parameter as the operating system can require some time to process the audio frames.

In case of the availability of more than one audio capture or playback devices, it is possible to provide the ***alsaloop*** utility with additional parameters. It helps you specify which devices to use.

Important	
	It is important to ensure that the capture and playback devices are compatible in terms of sample rate, number of channels, and audio format. Otherwise, issues may arise with the loopback connection.

10 Installation of Cellular Modem

Important
Cellular modem's is an optional installation and can be provided on request.

If the Display comes equipped with a cellular modem, such as a **3G** or **LTE** modem, it can act as a telematics device to transmit **CAN** and sensor data to the cloud. The cellular module also enables the Display to upload log and event-capture files and share geolocation information with the cloud through the internet.

Information
MRS Electronic has a cloud solution that can be used by end users as a portal to stream real-time data from the Display and visualize it using dashboards. For further information, contact MRS Electronic.

10.1 Structure

The Display handles the communication and setup with the different GSM modules. Depending on the service provider, distinctive settings for the SIM card's usage apply. Display supports **Micro SIM** card (15x12mm) and allows you to create the necessary scenarios for the SIM card setup.

The standard scenario, which is already active, contains the following fields:

Feature	Description
SIM authentication	PIN and PUK
PDP context configuration	APN, Address, Type, etc.
Enable PDP context	Enable the Packet data connection over which a device and the mobile network can exchange IP packets.

10.2 Configuration

To apply the configurations, use the */etc/wan-config.conf* configuration file:

```
# General

SCENARIO="default"

SIM_PIN=6879

SIM_PIN2=9988

SIM_PUK=68514769

SIM_PUK2=57391908
```

```
# PDP

PDP_SET=false

PDP_ENABLE=true

PDP_CONTEXT=1

PDP_TYPE="IP"

PDP_APN="iot.1nce.net"

PDP_ADDR="0.0.0.0"

PDP_D_COMP=0

PDP_H_COMP=0
```

Technical Fields and Functions Information

The following table lists the variables and their description used in the config file above:

Field/Function	Description
SCENARIO	Custom scenario which should be executed. Read more about it in **Scenario** (Section **10.5**)
SIM_PIN	PIN1 of a SIM card
SIM_PIN2	PIN2 of a SIM card
SIM_PUK	PUK of a SIM card
SIM_PUK2	PUK2 of a SIM card
PDP_SET	Describes whether the PDP setting should be set (Default: false)
PDP_ENABLE	Describes whether the PDP context should be activated (Default: true)
PDP_CONTEXT	Packet data connection over which a device and the mobile network can exchange IP packets
PDP_TYPE	Type of the IP address assigned to the PDP (IPv4 or IPv6)
PDP_APN	Identifies the packet data network (PDN) that the user of mobile data wants to communicate with.

10.3 GSM Tools

The Display is equipped with essential tools that facilitate communication with the GSM module, help to get information, and to make settings of the module.

10.4 GSM Status

The GSM status script is used to attain information about the network. The following table defines the flags to use in the script along with their respective descriptions:

GET

Flag	Description
-r	Status of network registration
-q	Signal quality
-s	Information of the serving cell
-h	Attain additional information about the script

Following is the method of using flags in the script:

```
$ wan-status.sh -r -q -s
```

For additional information about the script, add the *-h* flag:

```
$ wan-status.sh -h
```

To use it in the **Reference App** via C++, refer to the code below:

```
#include "wan.h"

int main(int argc, char *argv[])

{

   // Initialize GSM library. Optional parameters are described in ReferenceApp

          wan *m_wan = new wan();

   // Status of network registration

   QString networkRegistration = m_wan-> statusNetworkRegistration();

   // Status of signal quality

   QString signalQuality = m_wan-> statusSignalQuality();

   // Status serving cell

   QString servingCell = m_wan-> statusServingCell();

}
```

10.4.1 GSM PDP

With the GSM PDP script, it is possible to configure the PDP context. The following table defines the flags to use in the script along with their respective descriptions:

SET

Flag	Description
-i	PDP context identifier
-t	PDP type
-a	Access point name
-d	PDP address
-c	Data compression

-o	Header compression
-h	Attain additional information about the script

Following is the method of using flags in the script:

```
$ wan-pdp.sh -i 1 -t IP -a iot.1nce.net
```

For additional information about the script, add the *-h* flag:

```
$ wan-pdp.sh -h
```

10.4.2 GSM Authentication

GSM authentication script allows to get the status of SIM authentication or to enter the SIM pin. The following tables define the options and the flags to use in the script along with their respective descriptions:

GET

Flag	Description
-i	SIM PIN authentication status
-h	Attain additional information about the script

SET

Flag	Description
-s	Set required authentication value, such as PIN/PUK
-h	Attain additional information about the script

Following is the method of using flags in the script:

```
$ wan-auth.sh -i

$ wan-auth.sh -s 1234
```

For additional information about the script, add the *-h* flag:

```
$ wan-auth.sh -h
```

To use it in the **Reference App** via C++, refer to the code below:

```
#include "wan.h"

int main(int argc, char *argv[])

{

   // Initialize GSM library. Optional parameters are described in ReferenceApp

          wan *m_wan = new wan();

   // Status SIM authentication

   QString statusSimAuth = m_wan-> simAuth();
```

```
   // Set SIM authentication
   m_wan->simSetPIN("1234");
}
```

10.4.3 GSM Communication

GSM communication script allows to send direct commands to the GSM device. The following table defines the flags to use in the script along with their respective descriptions:

SET

Flag	Description
-c	AT command
-h	Attain additional information about the script

Following is the method of using flags in the script:

```
$ wan-at-command.sh -c AT+COPS?
```

For additional information about the script, add the *-h* flag:

```
$ wan-at-command.sh -h
```

10.4.4 GSM Restart

GSM restart script allows the restarting of the GSM device. To do so, use the following command:

```
$ wan-dev-restart.sh
```

To execute it in the **Reference App** via C++, use the following code:

```
#include "wan.h"
int main(int argc, char *argv[])
{
   // Initialize GSM library. Optional parameters are described in ReferenceApp
          wan *m_wan = new wan();
   // Restart the device
   m_wan->deviceRestart();
}
```

10.5 Scenario

The scenario describes a configuration process for the GSM module. When certain processes such as registration of a SIM card with extended parameters are required, *scenario* can help in its implementation.

10.5.1 Create a Scenario

There are certain rules to create a scenario that are as follows:

1. Scenario must be a shell script.
2. Scenario must follow the naming format of **{scenario-name}.sh**, where *scenario-name* is the name or description of the scenario.
3. The created scenario must be placed at the location: */etc/wan.conf.d/{scenario-name}.sh*

10.5.2 Example Scenario

This section defines an example of scenario for better understanding. Follow the steps below to create an example scenario:

1. The scenario enters the PIN for the SIM card.
2. It is located at */etc/wan.conf.d/enter-pin.sh*.
3. The scenario uses four parameters that can be changed in */etc/wan-config.conf*. The example values for the parameters are described in the table below:

Parameter	Example Value
Device	/dev/ttyACM0
Baud rate	115200
APN context number	1
Interface number of wan	1

Now, enter the following commands to *create* the scenario:

```
#!/bin/sh

dev=${1}

baud=${2}

apn_number=${3}

wan_interface=${4}
```

Execute the scenario using the following commands:

```
# Execute command

status=$(wl_command ${dev} ${baud} "AT+CPIN="\"1234\" "Auth")

echo "${status}"
```

Information	
	Note that the *wl_command* function is part of the *wan-libcmd.sh* script and is already included in the scenario script.

10.5.3 Scenario Configuration

To activate the specific scenario, change the value of the *SCENARIO* parameter in the */etc/wan-config.conf* file. For example, **SCENARIO="enter-pin".**

11 Starting User Application

This section covers the details of the two methods used to start the user application i.e., manually and automatically.

11.1 Starting User Application Manually

The user application can be started manually. This can be done on the primary and secondary screens.

11.1.1 Primary Screen

To start the user application manually for the primary screen, follow the steps below:

1. Transfer the *your-app* executable to the */opt/bin* location on the Display, using the USB.

```
$ mount -o remount,rw /

$ cp /run/media/sda1/your-app /opt/bin

$ sync

$ mount -o remount,ro /
```

2. Add the permission to execute the application with the following commands:

```
$ cd /opt/bin

$ chmod +x your-app

$ ./your-app
```

11.1.2 Secondary Screen

To start the user application manually for the secondary screen, follow the steps below:

```
$QT_QPA_EGLFS_FB=/dev/<screen>
QT_QPA_EVDEV_TOUCHSCREEN_PARAMETERS=/dev/input/<touch>:rotate=<touch_rotation>

//Example:

QT_QPA_EGLFS_FB=/dev/fb2
QT_QPA_EVDEV_TOUCHSCREEN_PARAMETERS=/dev/input/touchscreen1:rotate=180
```

Technical Fields and Functions Information

The following table lists the variables and their description used in the code above:

Field/Function	Description
<screen>	Select **fb0** for Main Screen Select **fb2** for Secondary Screen
<touch>	Select **touchscreen0** for Main Screen touch Select **touchscreen1** for Secondary Screen touch

<touch_rotation>	Select **0**,**90**, or **180** as per the screen

11.2 Starting User Application Automatically

To start the user application automatically, choosing a startup mode is essential. It is important to choose only one startup mode at a time.

11.2.1 Normal Startup Mode

In the Normal Startup mode, the user application starts when most of the elements are initialized. Due to this, the system boots a bit slowly.

To start the application in the Normal Startup mode, follow the steps below:

1. Transfer the *your-app* executable to the */opt/bin* location on the Display, using the USB.

```
$ mount -o remount,rw /

$ cp /run/media/sda1/your-app /opt/bin
```

2. Create a link file of the user application.

```
$ cd /opt/bin

$ ln -sf your-app normal.app.<screen> //Select <screen> as 0 for Main Screen and 1 for secondary screen
```

3. Enable the *systemd* service.

```
$ systemctl enable startup-app@<screen>

$ sync

$ mount -o remount,ro /
```

The application starts in Normal Startup mode.

11.2.2 Early Startup Mode

The Early Startup mode is an optional startup method to start the user application faster. Upon the selection of this mode, the *your-app* starts in less than ten seconds.

Warning	
	In Early Startup mode, some components are not initialized within this time such as CAN bus, GPS, and more.

To enable the Early Startup mode, follow the steps below:

1. Transfer the *your-app* executable to the */opt/bin* location on the Display, using the USB.

```
$ mount -o remount,rw /

$ cp /run/media/sda1/your-app /opt/bin
```

2. Create a link file for the user application.

```
$ cd /opt/bin

$ ln -sf your-app early.app.<screen> //Select <screen> as 0 for Main Screen and 1 for secondary
screen

$ sync

$ mount -o remount,ro /
```

The application starts in Early Startup mode.

Important	
	Display reboot is required for the automatic startup. To enable the automatic app startup, it is important to have soft link of the application in the location: */opt/bin* with the name *early.app* or *normal.app*.

12 Reference Application Components

This section covers the details of the components on the **Reference App**'s User Interface (UI). These components are Camera Recording, Wi-Fi, Missing Interface Detection, Touch Pointer, PDF Viewer, Video Player, Date Time Settings, and Theme.

The following image represents the tabs of the **Reference App** and the relative description sections of each:

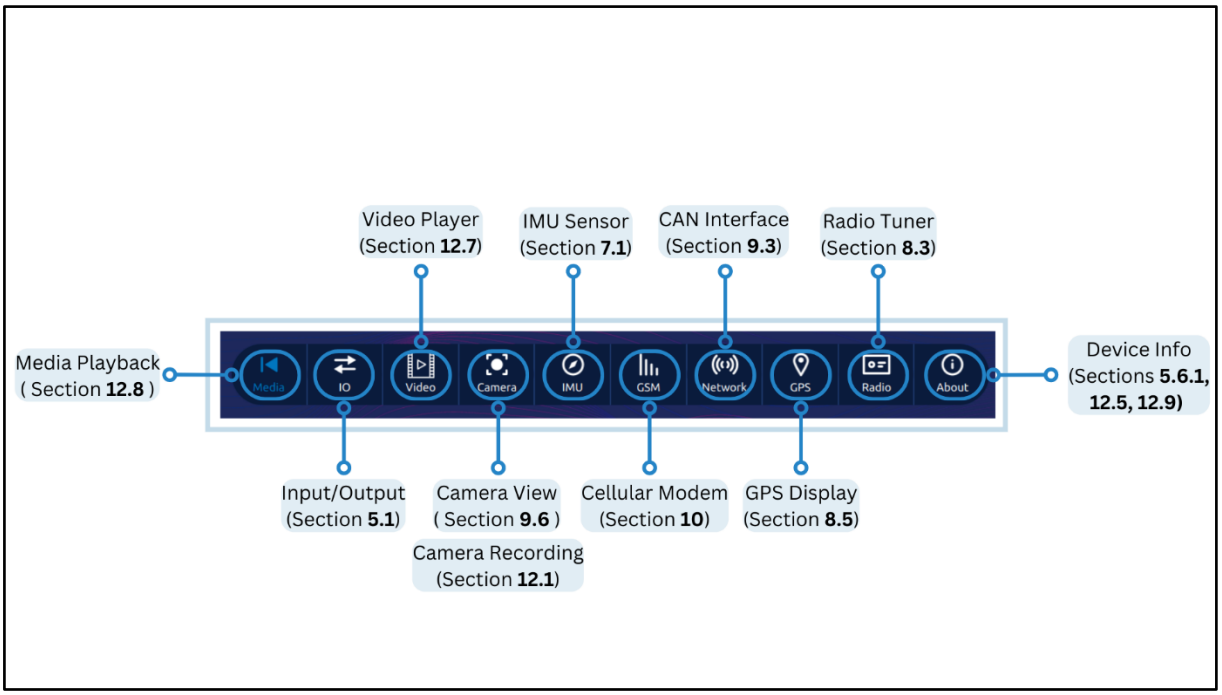

Figure 17: Tabs of the Reference App and their Relevant Description Sections

12.1 Camera Recording

The Display is capable of recording video streams of IP cameras (The recording feature for analog cameras will be included in the upcoming versions). The **Reference App**'s camera module has two methods for starting and stopping the recording, which can be used to record the camera feed dynamically. These functions attach and detach **GStreamer's** recording pipeline from the main pipeline at runtime.

Add the following in C++ code to control the camera recording:

```
Q_INVOKABLE void startRecording (QString startby);

Q_INVOKABLE void stopRecording (QString stopby);
```

Important	
	The *startby* argument must match with *stopby* argument. It imposes the condition that only the interface that starts the recording can stop it.

Using these methods, recording can be controlled through multiple interfaces, including CAN, IMU, and Analog input. Also, the Display holds the functionality of detecting error in the stream of IP camera.

To enable the camera recording in the **Reference App**, tap on the toggle button in the **Camera** tab as highlighted below:

Figure 18: Camera Recording in the Reference App

12.1.1 Camera Recording Based on CAN

The recording can start and stop through CAN message. The first two bytes of the CAN message represent the camera number which needs to be controlled. The next two bytes represent state for that camera (*1* for *On* and *0* for *OFF*). The CAN message for recording control is received at *0x201* CAN ID.

Following flags of *canvideorecording* module keep track of the camera's recording state based on the CAN message:

```
int m_recordingCam1;

int m_recordingCam2;

int m_recordingCam3;

int m_recordingCam4;
```

12.1.2 Camera Recording Based on IMU

The recording can start or stop based on the roll value of IMU. If roll value crosses upper or lower threshold of ±10, the recording starts for ten seconds.

Following flag of *imuvideorecording* module keeps track of the camera's recording state based on the IMU value:

```
bool m_recordingCam1;
```

12.1.3 Camera Recording Based on Analog Input

The recording can start or stop through an analog input. For this functionality, analog pin 1 is programmed. If the voltage on analog pin 1 exceeds upper threshold (10.8V), the recording starts. Similarly, if the voltage on analog pin 1 drops below lower threshold (10V), the recording stops.

Following flag of *iovideorecording* module keeps track of the camera's recording state based on analog input:

```
bool m_recordingCam1;
```

Information	
	The upper and lower threshold for voltage on analog pin can be changed as per the user preferences. For more details, contact MRS Electronic.

12.1.4 Error Handling for Video Stream

The Display comes with the functionality to detect error in the stream of the IP camera. In case, it detects an error, an error message displays on the black screen.

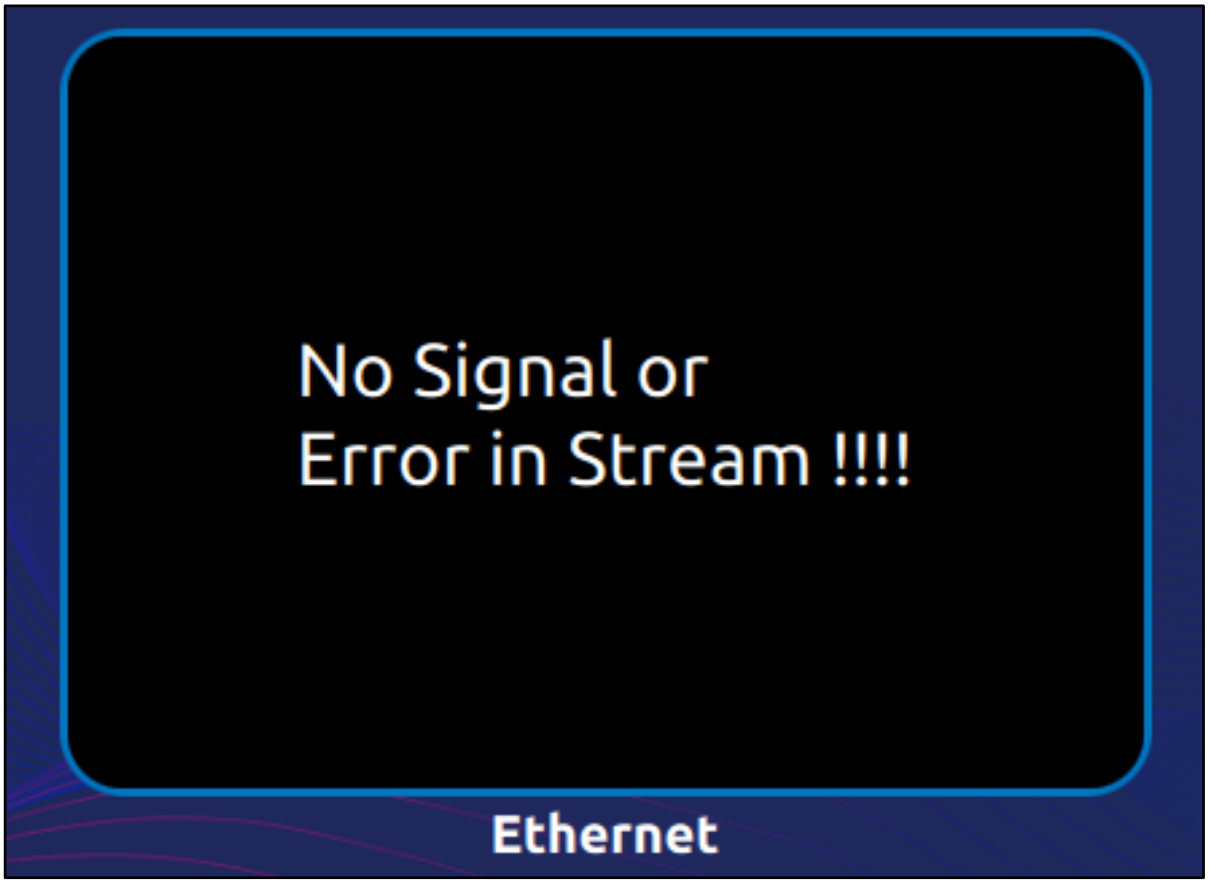

Figure 19: Error Display Screen for Error Detection in the IP Camera Stream

Upon resolution of error, the Display reinitiates **GStreamer** pipeline automatically. On backend, it uses *startErrorHandling()* function of the *camera* class. This function runs a repetitive timer at intervals of ten seconds on a separate thread that checks error message on the *gst_bus*. The *camera* class provides *m_isStreaming* flag to indicate the status of the stream.

12.2 Wi-Fi

The **Reference App** comes with a Wi-Fi component. To access it, follow the steps below:

1. Open the Control Panel by swiping down from the top.
2. Tap-and-hold on the **Wi-Fi** card.
3. The **Wi-Fi Settings** menu opens and displays several options, including **Wi-Fi Mode**, **Priority switching**, Wi-Fi scanning, and settings of the saved Access Points as shown in the image below:

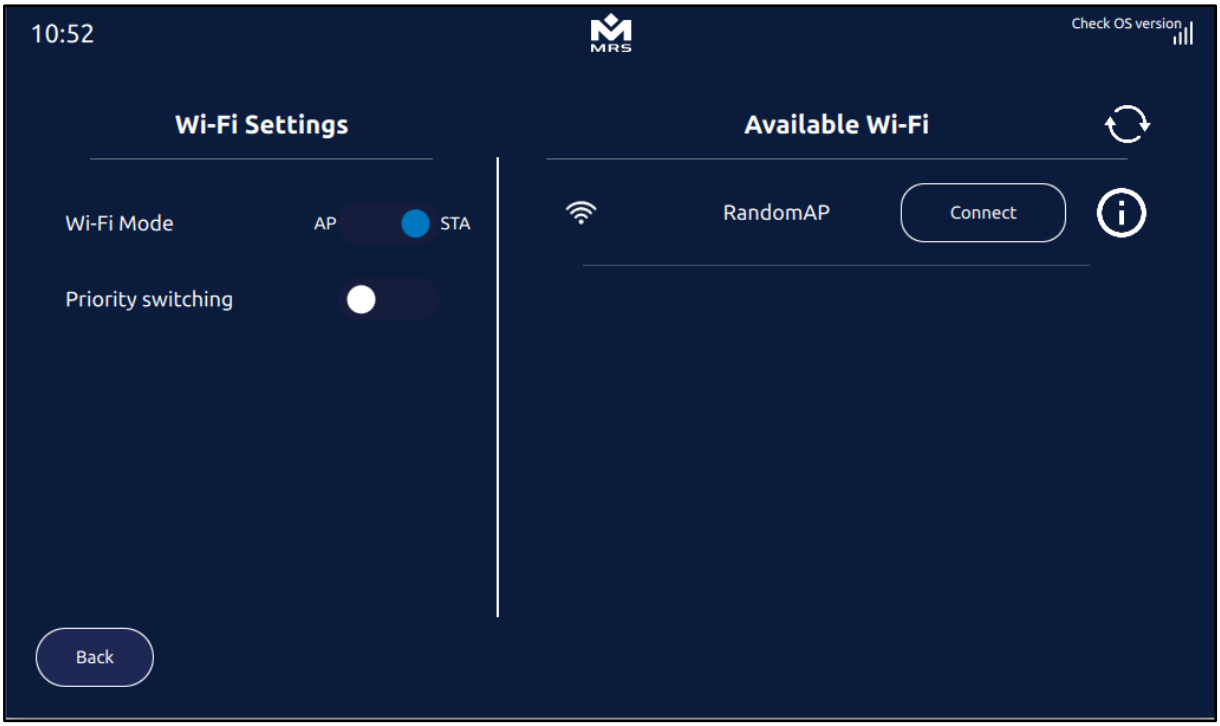

Figure 20: Options in the Wi-Fi Settings Menu

12.2.1 Wi-Fi Mode Switching

The **Wi-Fi Mode** switch allows switching between the AP and STA mode. On backend, it utilizes same the steps as defined in the **Wi-Fi Module** (Section **8.2**). By default, the MConn Display that has an inbuilt Wi-Fi module comes with the AP mode. On switching to STA mode, it performs the auto scan for available Wi-Fi Access Points and displays the list. On backend, it uses two *Q_INVOKABLE* functions to switch between the Wi-Fi modes.

Use the following QML functions to perform the Wi-Fi mode switching:

```
wifi.set_ap_mode()  // To switch to AP mode
```

```
wifi.set_sta_mode() // To switch to STA mode
```

12.2.2 Wi-Fi Priority Switching

To turn on Wi-Fi priority switching, press the toggle button. In this mode, MConn makes passive Wi-Fi scan using *wpa_cli* supplicant at regular intervals of ten seconds for the available Wi-Fi Access Points. When the MConn finds the access point with higher priority than the current one, it switches to the new access point. On backend, it uses two *Q_INVOKABLE* functions to start and stop priority Wi-Fi switching.

Use the following QML functions to perform Wi-Fi priority switching:

```
wifi.startPriorityWifiSwitching()//Start priority switching

wifi.endPriorityWifiSwitching() //Stop priority switching
```

12.2.3 Wi-Fi Scanning

To scan for the available Wi-Fi Access Points, tap on the **Refresh** (🔄) icon. On backend, it uses *wpa_cli* to scan for nearby Access Points. A list of available Access Points displays with options to connect to a specific point or to change the setting of the saved Access Points.

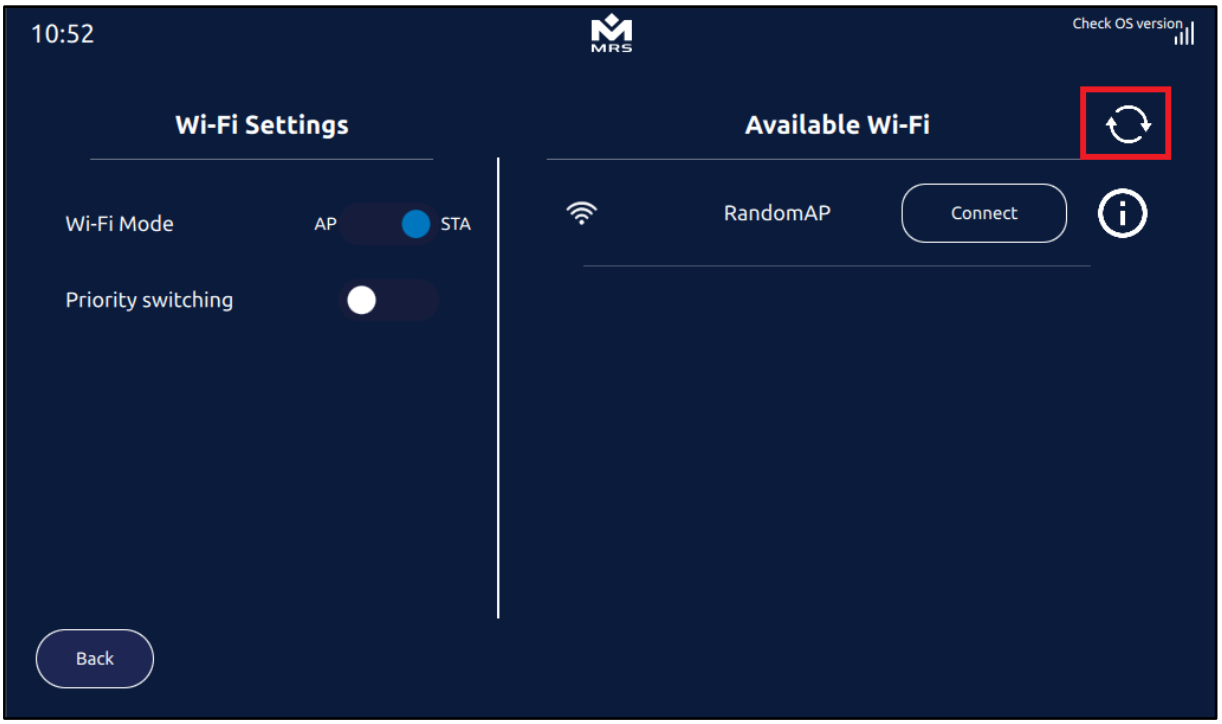

Figure 21: Wi-Fi Scanning Option in Wi-Fi Settings

Following QML function performs the Wi-Fi scanning:

```
if(wifi.wifiMode == "stamode")

{

wifiScanListModel.clear() // Clear model

wifiListBussyIndicator.running = true // For feedback

loadlistTimer.running = true //To read WiFi after some specific time
```

```
wifi.scan()  // start scanning
}
        Timer {
            id: loadlistTimer
            running: false
            interval: 3500
            onTriggered: {
                wifiScanListModel.clear()
                for (let i = 0; i < wifi.ssid_list.length; i++)
                {
                    wifiScanListModel.append({name: wifi.ssid_list[i]})//populate list model
                }
                wifiListBussyIndicator.running = false
                wifi.readRegisteredAccessPoints()
                wifi.get_status()
            }}
```

12.2.4 Settings of Saved Access Points

To change the settings of any of the saved Access Point(s) in the scanned list, press the Information (ⓘ) icon. It opens the settings window for that specific Access Point where the options of updating its priority, background scanning threshold, and password exist.

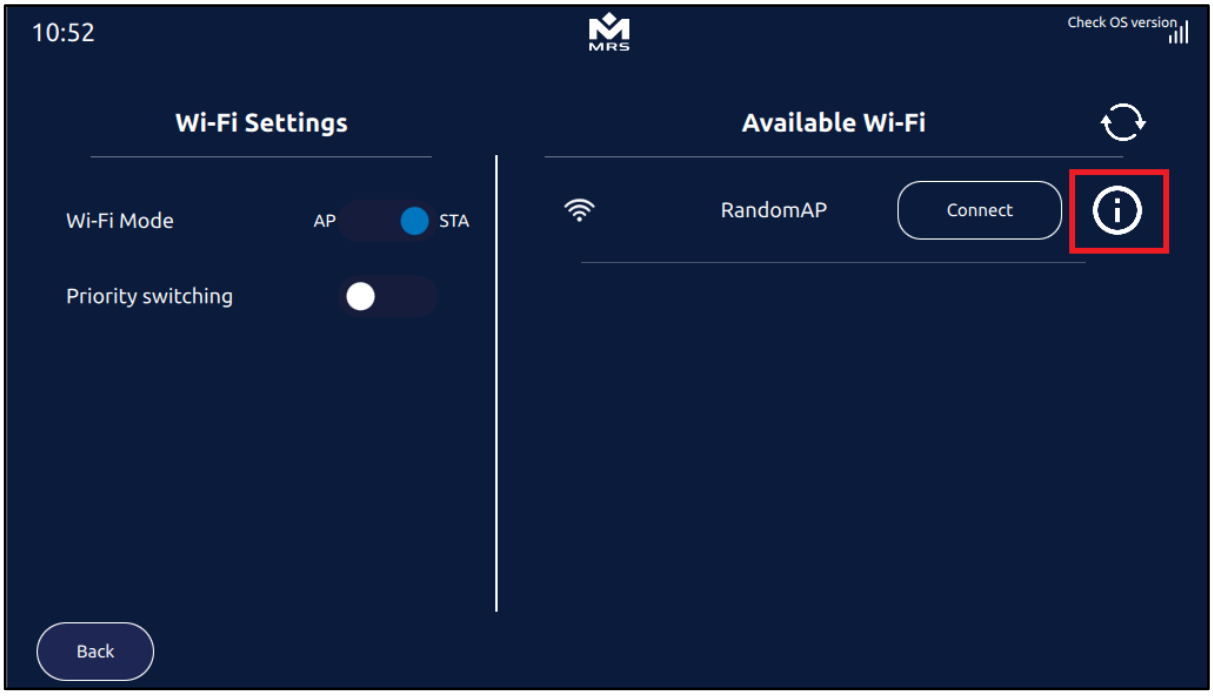

Figure 22: Settings of Saved Access Points

12.3 CAN-Flasher

CAN-Flasher allows users to flash **.s19** files onto the controllers or coprocessor of MConn through the CAN interface. It is a standalone tool that provides the functionality of installing and compiling **.s19** files for different MRS products.

Information
If you have any questions about CAN Flasher, contact MRS Electronic.

To install the relevant file(s), follow the steps below:

1. Pull down the notification pane in the **Reference App**.
2. Tap on the **CAN-Flasher** card.

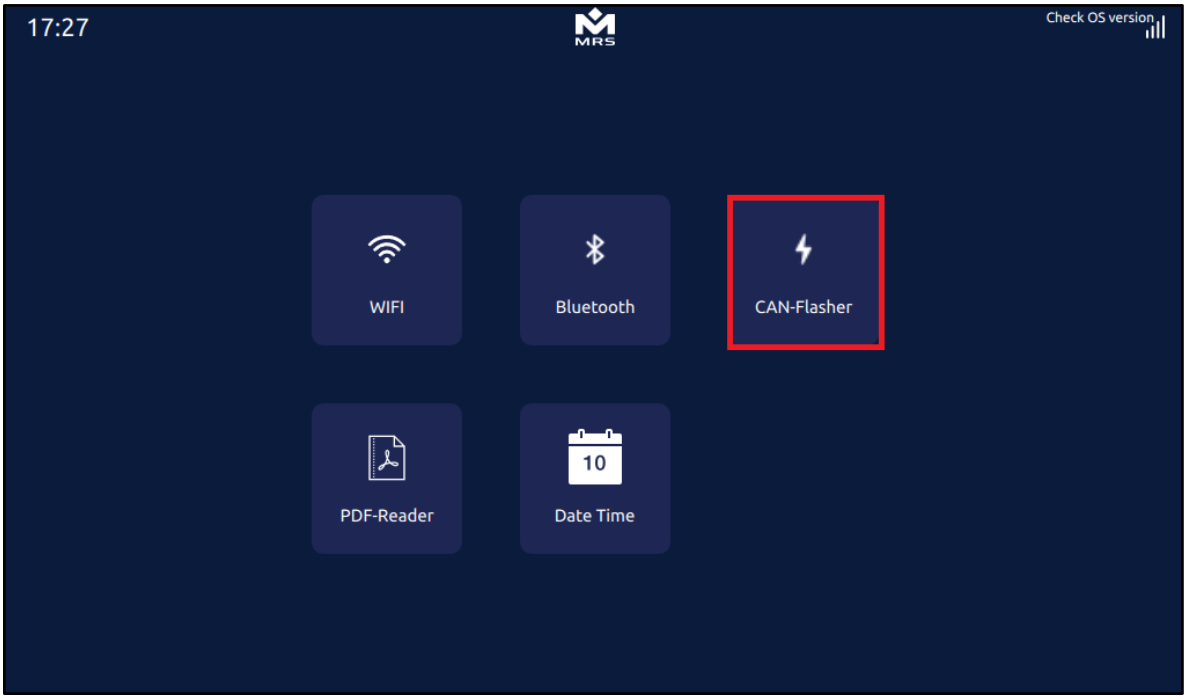

Figure 23: CAN-Flasher Card on the Reference App

3. Select the configured **Baud Rate** for the device from the dropdown.
4. Select the relevant CAN Bus from the dropdown. This displays the associated device in the **Devices** dropdown.

Information
It is advised to use CAN1 bus for flashing purposes.

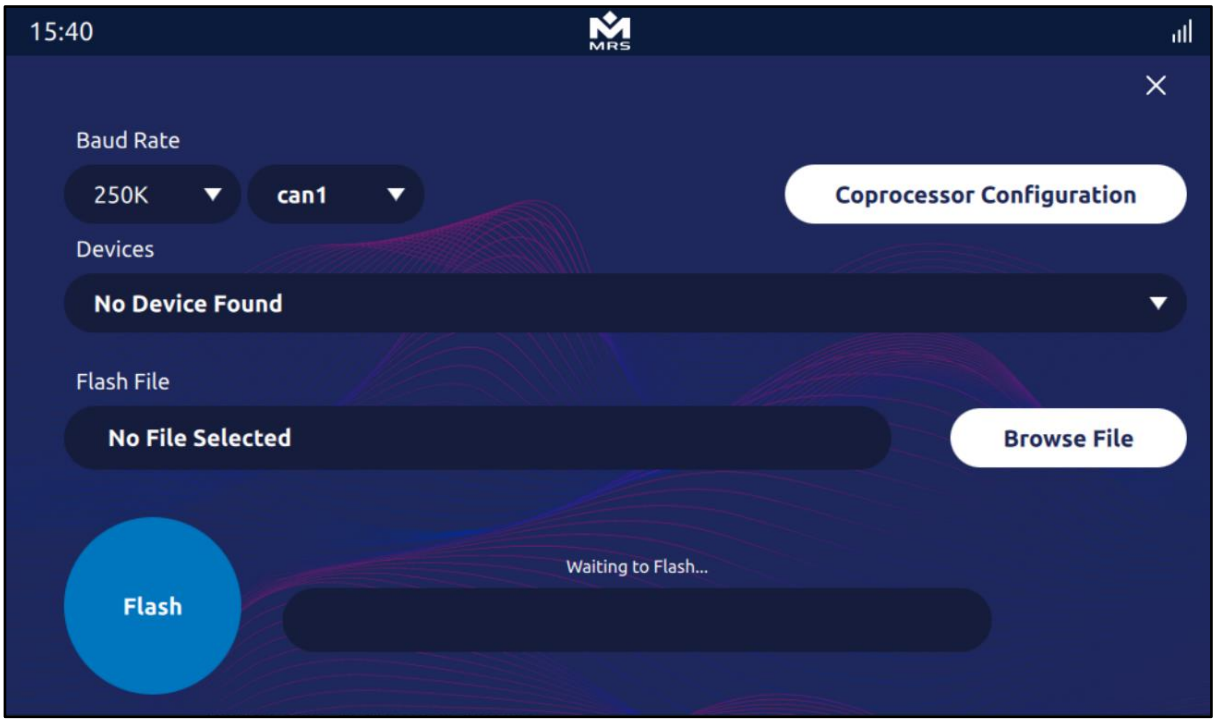

Figure 24: CAN-Flasher Settings

5. If required, follow these steps to change the coprocessor settings:
   a. Tap on the **Coprocessor Configuration** button on the right. It navigates you to the **Coprocessor Configuration** tab.
   b. Select the required **Baud Rate** from the dropdown.
   c. Upload the pre-defined **Flash File**.
      Tap on the **Okay** button. This completes the Coprocessor CAN settings.

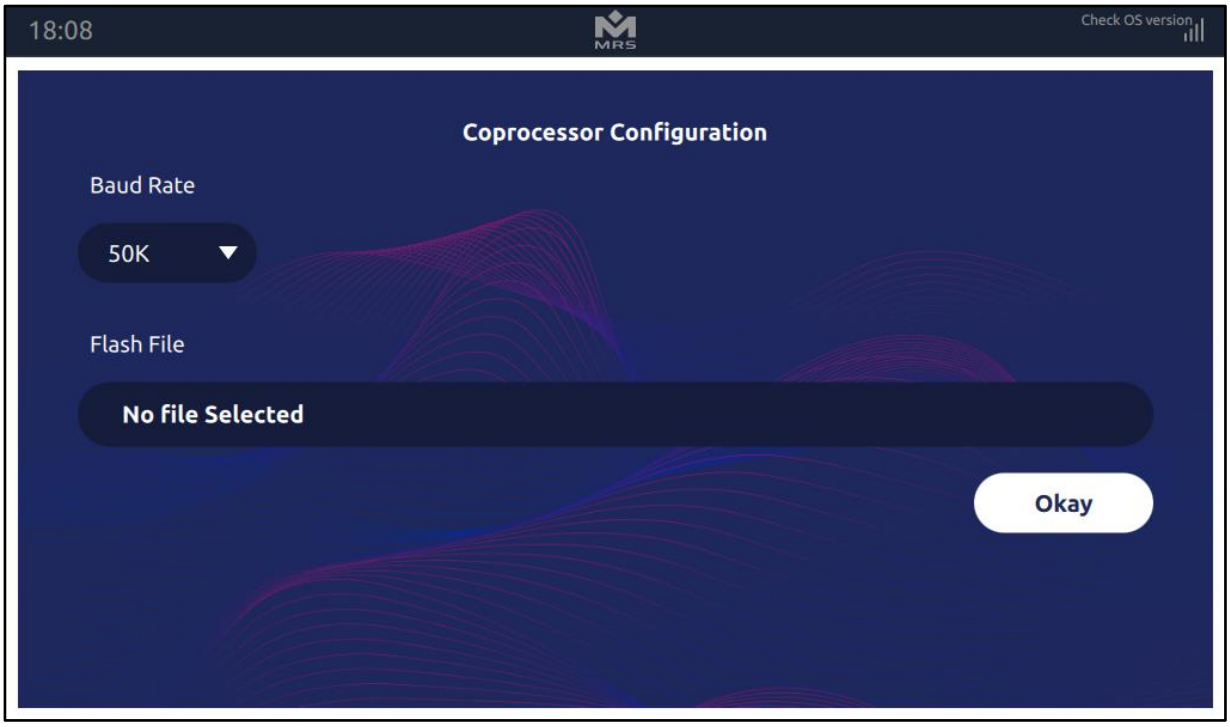

Figure 25: Coprocessor Configuration

6. If required, choose a custom file for flashing. To do so, tap on the **Browse File** button in the CAN-Flasher Settings tab and select the file to upload.
7. Tap on the **Flash** button to begin the flashing process and upload the files to device.

12.4 Missing Interface Detection

The Display holds the functionality to detect availability of an interface and provide feedback if a certain interface is missing. *InterfaceStatus* module in the **Reference App** keeps track of all the interfaces and shows their availability status through the following flags:

- Missing Network Interfaces
- Missing Interfaces Associated to the Menu Tab

12.4.1 Missing Network Interfaces

When the Display starts and any of the networking interfaces like Wi-Fi, Ethernet, or Bluetooth is missing, the popup notification appears for two seconds to show their unavailability. Tap on the close button present in the right of the popup to close it manually.

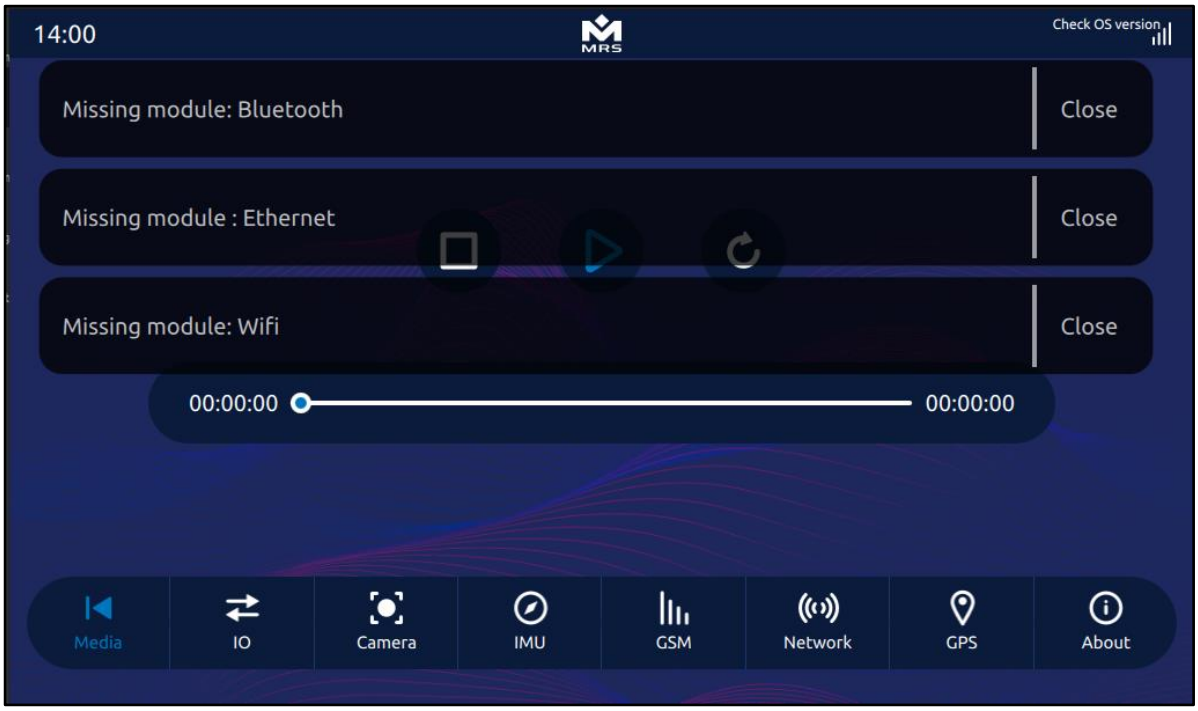

Figure 26: Missing Network Interface Detection

12.4.2 Missing Interfaces Associated to Menu Tab

If any of the interfaces associated with the menu tab of the **Reference App** is missing, a black screen with the unavailability message appears on the respective tab, which blocks all user interactions to avoid any unwanted behavior.

Figure 27: Missing Interface Associated with the Menu Detection

12.5 Touch Pointer

The MConn Display is equipped with the functionality to provide visual feedback upon a touch event. Every time you tap on the screen, a colored circular pointer appears. There is an option to enable or disable the touch pointer. Also, the Display allows the customization of touch pointer in term of its color, size, and opacity.

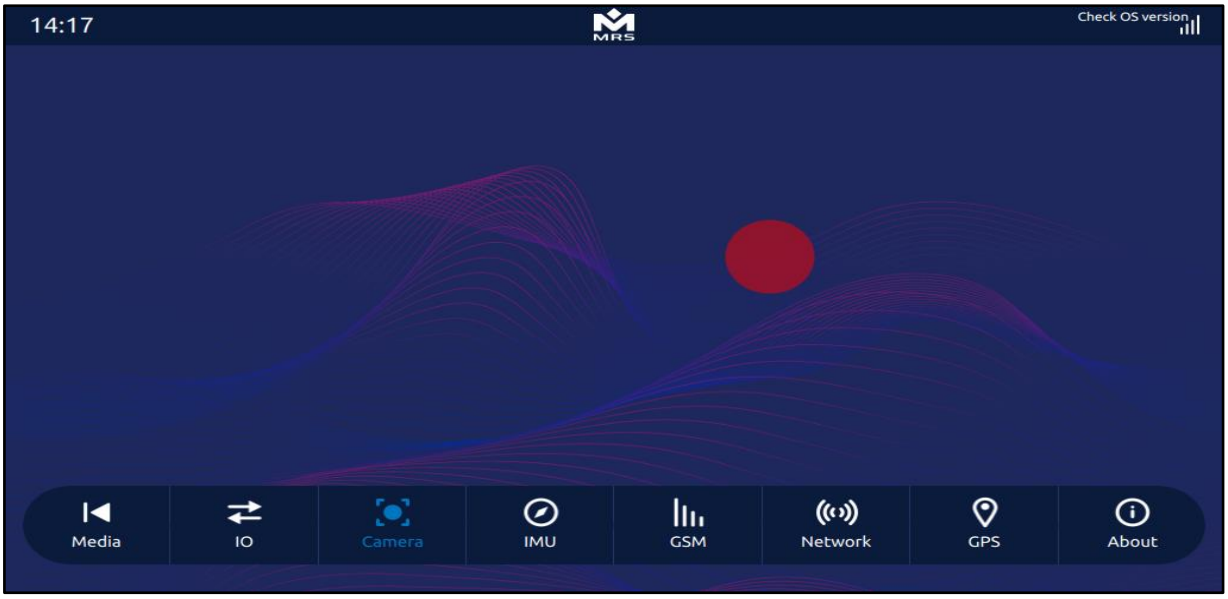

Figure 28: MConn Touch Pointer

To access the customization options of touch pointer, follow the steps below:

1. Navigate to the **About** tab of the Reference App.
2. Tap on the **Touch Pointer Settings** button.

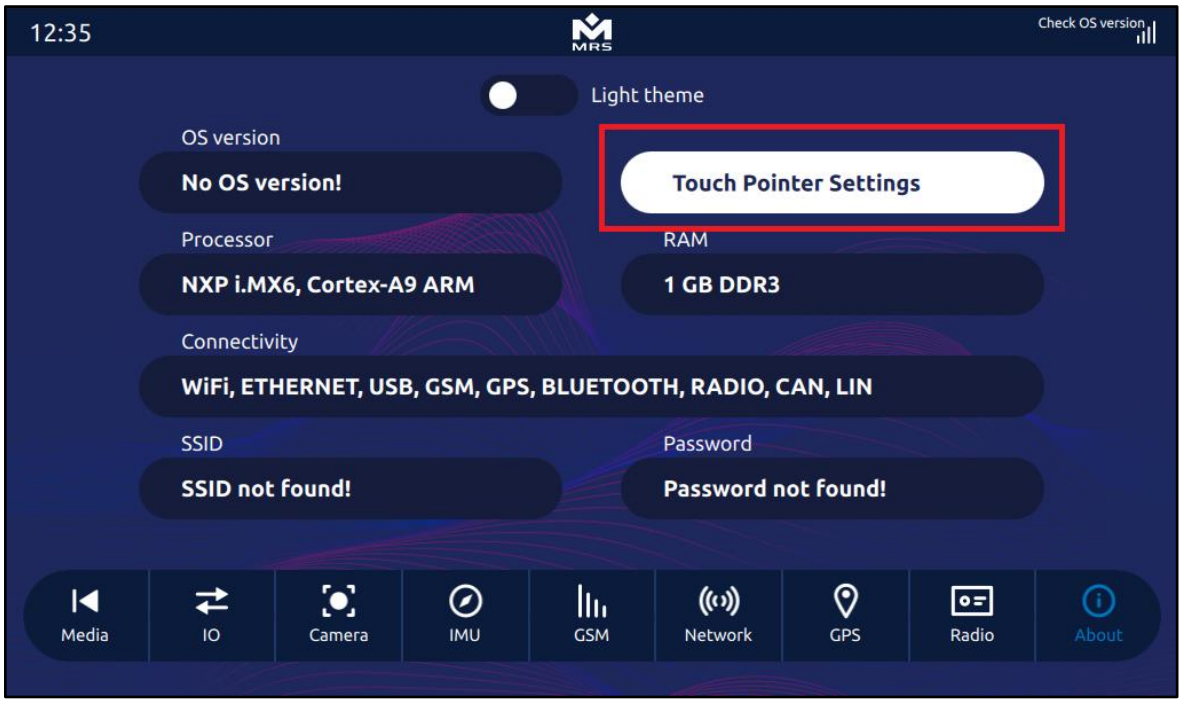

Figure 29: Touch Pointer Settings button in the About tab

3. Update the touch pointer settings as per requirements.

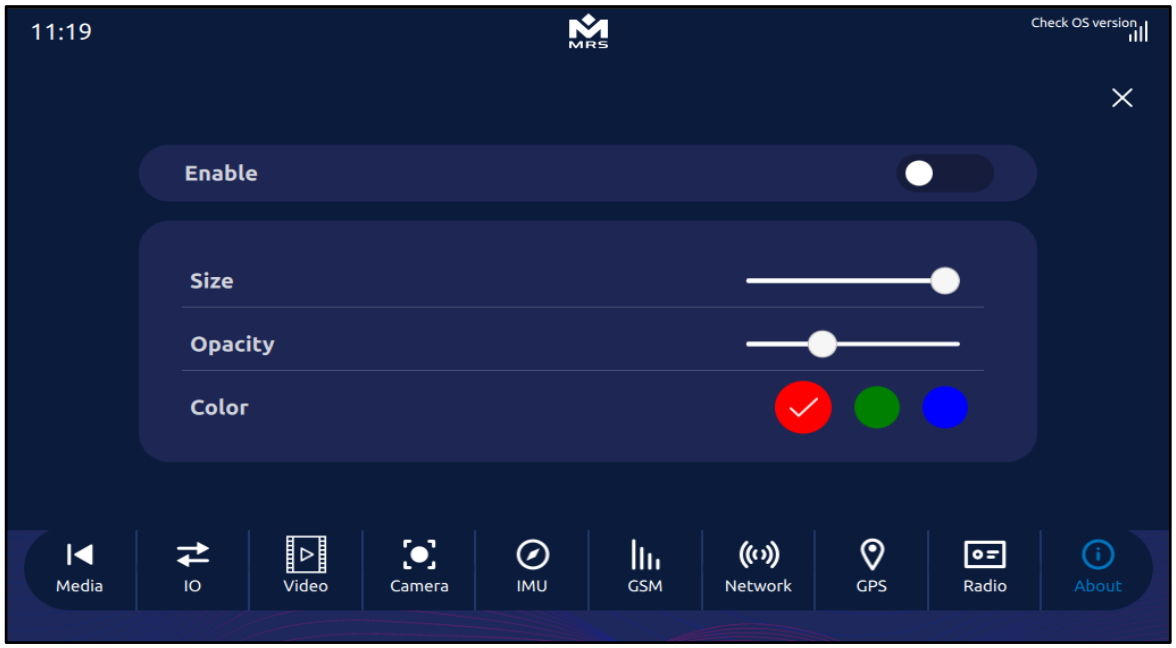

Figure 30: Touch Pointer Settings

Information	
	To retain the user preferences for the Touch Pointer Settings, all settings are stored in the *config* file.

12.6 PDF Viewer

The **Reference App** comes with the PDF Viewer tool for displaying contents of PDF files on the screen. The PDF Viewer consists of a tool bar with Zoom In/Out options and a Page Number entry box. The PDF Viewer can be configured using C++ code and via QML.

Use the following C++ code to configure the PDF Viewer in the **Reference App**:

```
#include "pdfModel.h"
int main(int argc, char *argv[])
{
pdfFileProvider * pdfFile = new pdfFileProvider(m_app);
 engine->rootContext()->setContextProperty("fileExplorer", m_pdfFile);
qmlRegisterType<PdfModel>("MRSComponents", 1, 0, "PdfReader");
}
```

Use the following QML commands to configure the PDF Viewer:

```
PdfReader{
            id: pdfReader
   path = file:///Path/to/File
}
ListView{
id: view
anchors.fill: parent
model: pdfReader.numPages
spacing : 10
delegate:  Image{
id: imagePdf
width: parent.width
source:pdfReader.loaded?"image://pdfReader/page/"+(modelData+1) : ""
sourceSize.width: width
visible:  pdfReader.loaded?1:0 }}
spacing : 10
delegate:  Image{
id: imagePdf
width: parent.width
source:pdfReader.loaded?"image://pdfReader/page/"+(modelData+1) : ""
sourceSize.width: width
visible:  pdfReader.loaded?1:0 } }
```

12.7 Video Player

The MConn comes with a video player. The **Reference App** video player holds play/pause and fast forward/rewind video functionalities. The video player supports the following video formats:

- Mp4
- Flv
- Mkv
- 3gp

Following are the commands to use the video player in QML:

```
Video {
      id: video
      anchors.fill: parent
      source: "file:///rw_data/samplevideo.mp4"
      autoLoad: true
      autoPlay: false
      fillMode: VideoOutput.Stretch }
```

12.8 Media Playback

The MConn Display comes with two different media players to accommodate playback for USB devices and Bluetooth. To play media files in a USB, follow the steps below:

1. Connect the USB to MConn via USB port.
2. Navigate to the **Media** tab of the **Reference App**.

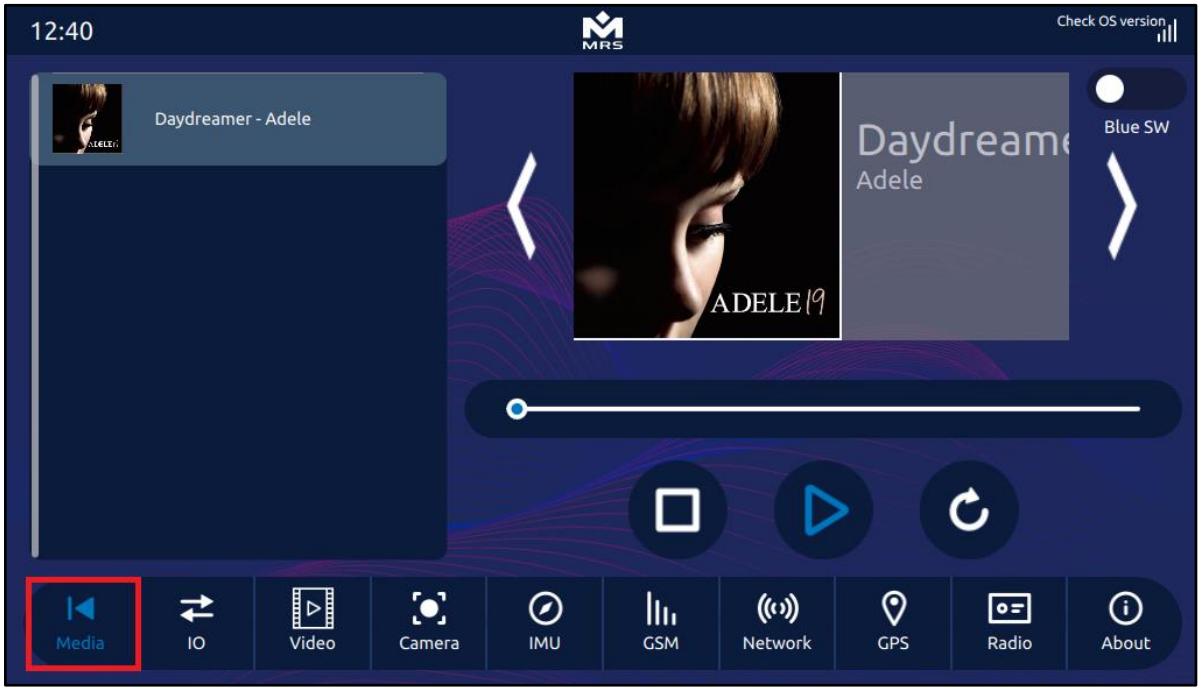

Figure 31: Media Tab in the Reference App

The application loads all the compatible files in the USB automatically. The files are listed in the form of a scrollable list in the left side of the tab. The currently selected file is displayed on the right side of the tab. If the thumbnail and title are available, they are retrieved automatically.

3. Tap on the Play/Pause, Stop, and Repeat icons below the audio file to execute the desired operation.
4. Select a new audio file by tapping on either of the arrows displaying around the thumbnail. You can also select it from the displayed list on the left.

To the top right of **Media** tab lies **Blue SW** toggle. This toggle allows you to play Bluetooth media by switching to a different media player. In this case, the audio data is streamed over the **A2DP** Bluetooth profile.

Important	
	The Bluetooth device must be connected with the MConn to stream audio data over the **A2DP** Bluetooth profile.

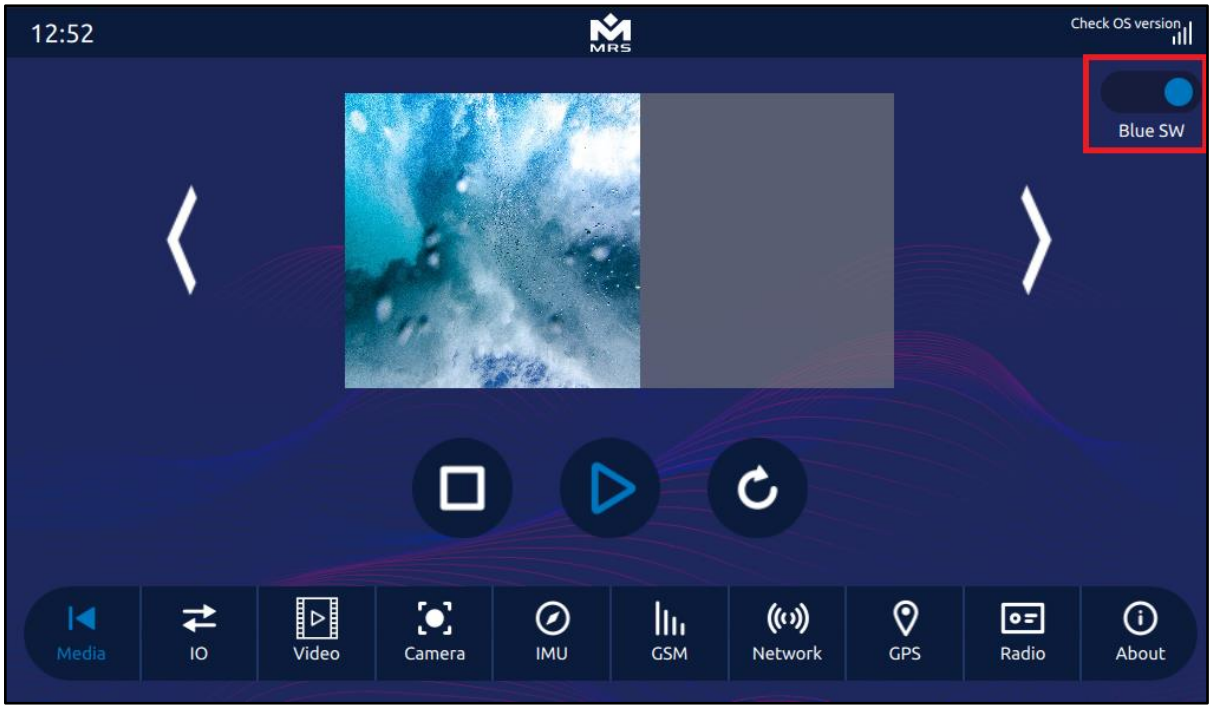

Figure 32: Blue SW Toggle in the Reference App

The audio's title and album details of the currently played audio are automatically retrieved and displayed in the center. The Play/Pause, Stop and Repeat icons display below the audio in the same manner. You can switch to the previous or new tracks by tapping on either of the arrows displaying around the thumbnail.

12.9 Date Time Settings

The Display comes with the functionality to set date and time. To access date and time setting component, follow the steps below:

1. Open Control Panel by swiping down from the top.
2. Tap on the **Date time** card to open the settings panel.
3. Set the date and time as per requirements.
4. Tap on the **Set** button to save the changes.

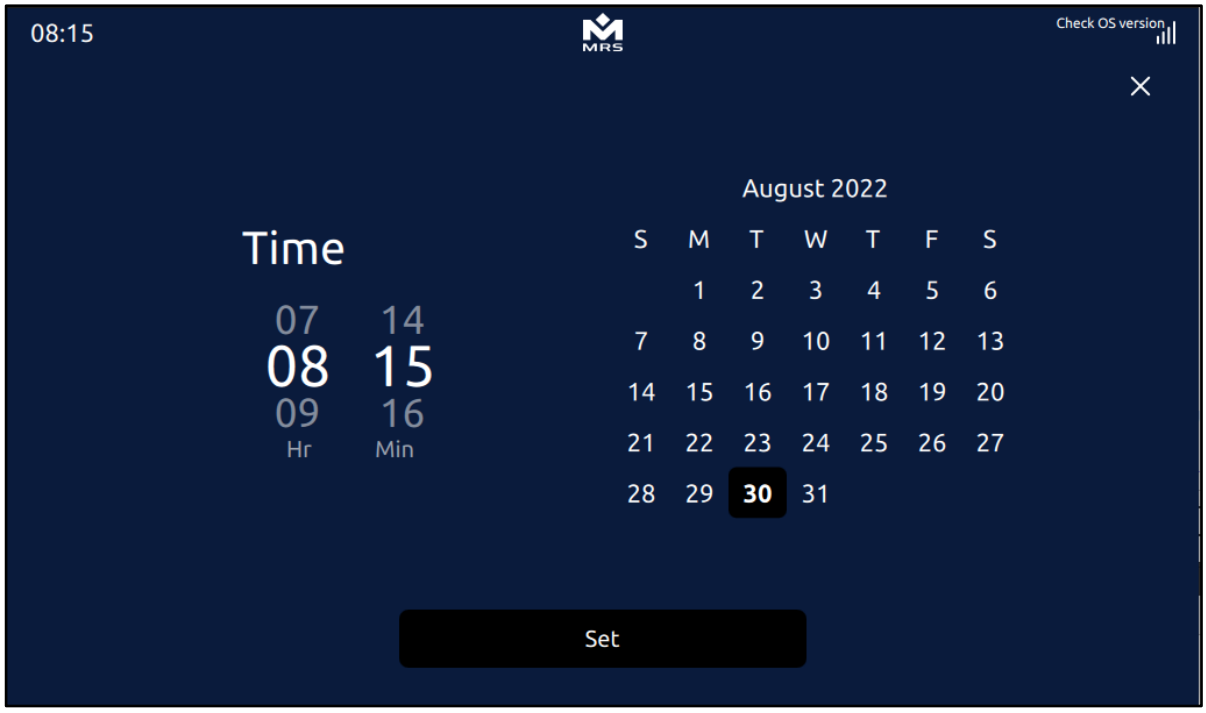

Figure 33: Date and Time Component of MConn

On backend, it uses simple *Q_INVOKABLE setTimeDate* function that uses the *date* command of Linux to set user-defined time and date.

Important	
	The date must follow the MM/DD/YYYY format.

12.10 Theme

The MConn **Reference App** comes with two in-built contrasting themes, which by default, is the Dark Theme. This information, along with some other configurations, is kept in the form of key value pairs loaded at startup from the following file:

```
/rw_data/app-config.ini
```

The variable *REF_APP_THEME* when set to 0 or 1 corresponds to dark or light theme for the **Reference App**, respectively.

To change the theme of the **Reference App**, tap on the toggle button in the **About** tab:

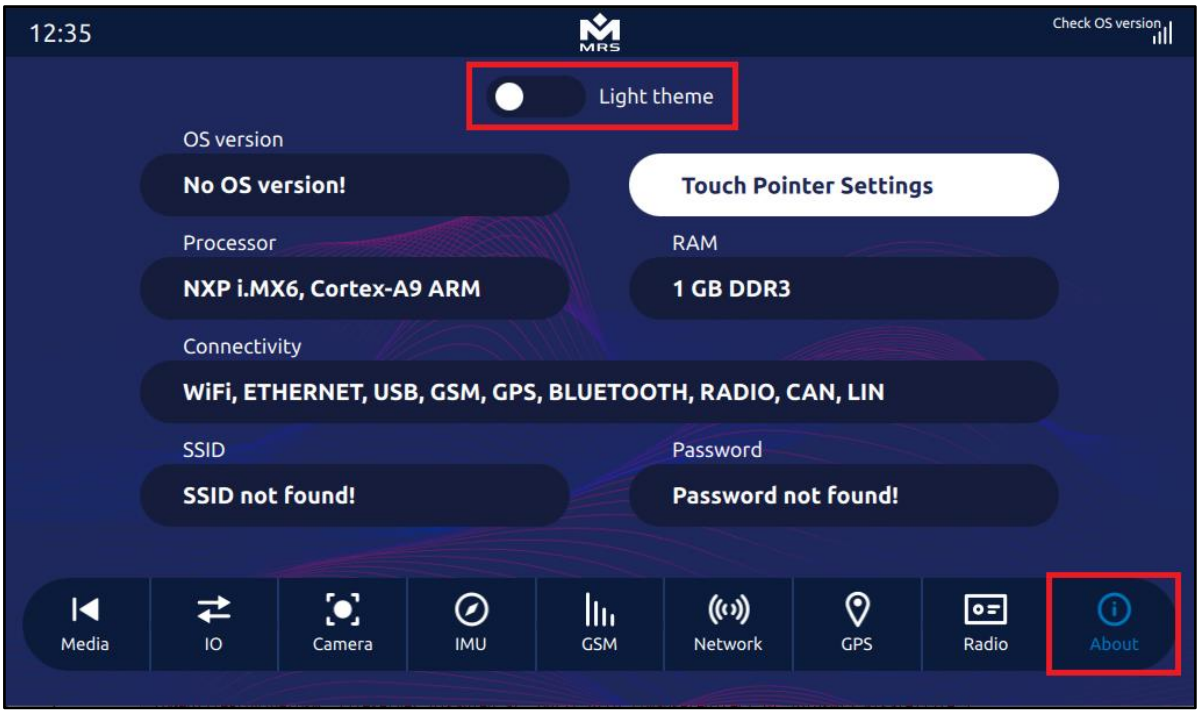

Figure 34: Theme Toggle in the Reference App

The light theme of the **Reference App** looks as follows:

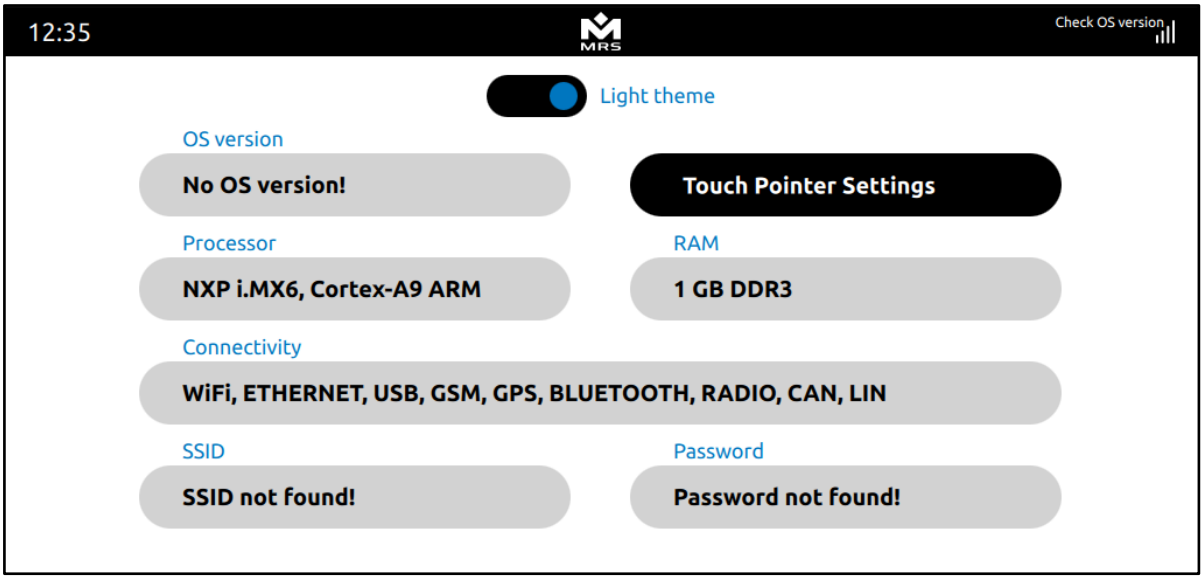

Figure 35: Light Theme of the Reference App

13 I2C Explorer

I2C (Inter-Integrated Circuit) is a communication protocol used for short-distance communication. This communication can be between microcontrollers, sensors, and other integrated circuits. The communication is initiated by a master device, which generates the clock signal and sends/receives data from one or more of the slave devices.

The **I2C-Explorer** utility allows you to scan for all the available **I2C** devices and provides you with some details regarding them. These details can include the name of the device and address at which the device can be found. It then sorts this list according to the **I2C** buses available for communication.

The image below shows the typical output when different devices are connected to multiple buses:

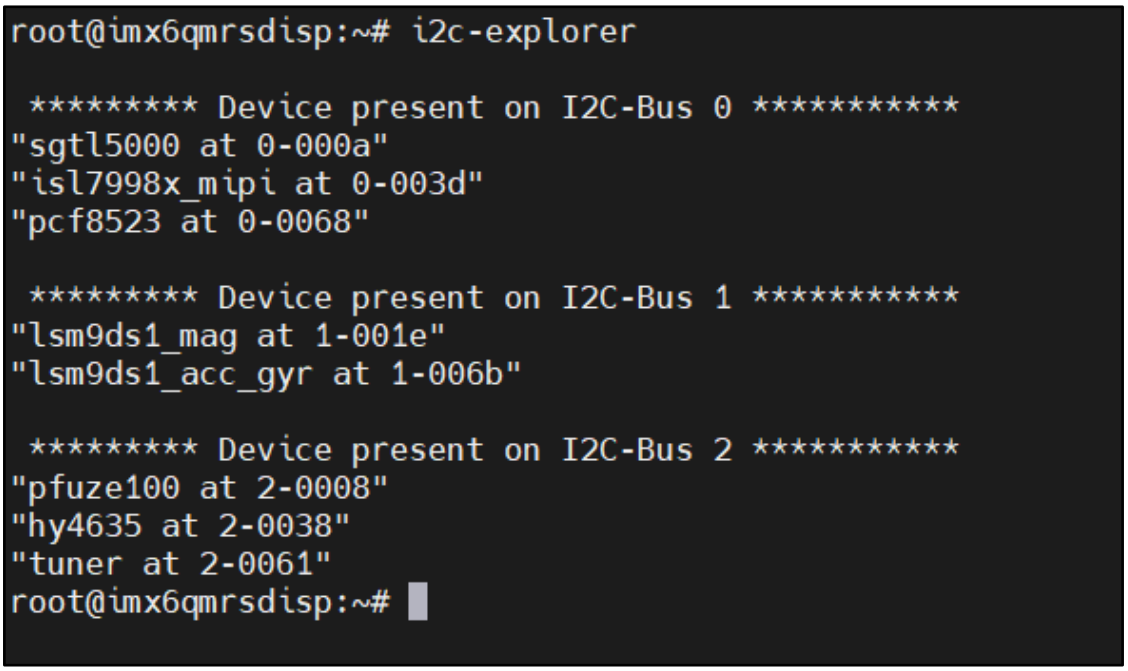

Figure 36: Output of I2

14 Service Mode

The **Reference App** comes with a mode known as Service mode. This mode starts with the login page and allows only authenticated users to access the MConn. The user's information is stored in the database. Following are the features of the Service mode:

Feature	Description
Multi-User Role	**Reference App** comes with the definition of two roles such as Admin and User. The admin can edit privileges of the user to access MConn components.
Dynamic Swipe Views	When the user successfully logs in, the dynamic swipe views are created based on the user privileges.
SQL Injection Prevention	All user input fields pass through sanitization function before executing SQL query to prevent SQL-Injection attack.
Secure Password	To secure password, *salt* is added to the password and passed through irreversible cryptographic function (**SHA256**) before storing it in the database.
Password Recovery Method	To recover password, the Forget Password option exists where the user is asked two security questions. If the answer is correct, the change password option appears. These security questions are set by the Admin at the time of account creation. Also, Admin can change password of any user through **Administrator** tab in the application.
Add/Remove User	Admin holds the privilege of adding and removing the user to/from the database.

In Service mode, the application starts with following login page:

Figure 37: Login Page of Application in Service Mode

14.1 Admin Role

When the Admin logs into the device, the **Administration** tab opens in the navigation bar. In that tab, the Admin can execute the following tasks:

- Add a new user
- Reset password of user
- Edit user privileges
- Delete user

To add a new user, the Admin must follow these conditions:

1. Username must be unique and alphanumeric with at least five characters.
2. Password must have at least four characters, including one letter, one number, and one special character.
3. The answers to the security questions must be alphanumeric with at least five characters.

14.2 User Role

After the user's registration on the device, he or she can log in using the *username* and *password* provided by the Admin. The user has the access to the components as assigned by the Admin.

15 Technical Support and FAQs

This chapter covers all the information related to technical support and some frequently asked questions.

15.1 Technical Support

If the instructions for the MConn have been followed, no difficulties should be encountered. However, if there is a problem, contact our customer service.

Information	
	To solve the problem, the customer service requires a clear and concise description. Ensure that you convey the issue with as many details as possible.

Manufacturer: MRS Electronic GmbH & Co. KG

Address: Klaus-Gutsch-Str. 7, 78628 Rottweil, Germany

Telephone: +49 741 28070

Internet: www.mrs-electronic.de/en/service-support/support

E-mail (Service & Support): support@mrs-electronic.com

15.2 Frequently Asked Questions

Question: When to lodge a complaint to the technical support team?

Answer: If a problem occurs, verify that it does not relate to a fault outside the MConn Display. Ensure the following:

1. The MConn device is working correctly.
2. The connection pins to the mating connector are connected properly.
3. The machine configuration is done correctly (in the case of manual configuration).

If none of the above-mentioned points holds true, contact MRS Electronic.

Question: I am unable to deploy the application to the MConn from development environment. How to resolve the issue?

Answer: Make sure the MConn device is connected and accessible through SSH in the development environment to perform the needful.

Question: How to deal with the analog camera decoder chip failing to start properly?

Answer: Reset the camera by toggling GPIO 146. It is recommended that *your-app* resets the camera upon initialization so that the camera is ready to use, when needed.

16 Revision History

Version Number	Reason for change	Date	Author
2.5.0	Updated the content and the UI/UX images of the application.	10/05/2022	Kainaat Arshad
2.6.0	Added the **CAN-Flasher**, **I2C Explorer**, and **Media Playback** sections. Updated the **In Audio** and **Bluetooth** sections.	18/04/2023	Kainaat Arshad
2.6.1	Updated the **Ethernet** section for new details.	05/05/2023	Kainaat Arshad