

User Manual

SMARC™ Module

MSC SM2S-IMX8MINI

SMARC Rev. 2.0 Standard

16.08.2021 Rev. 1.5



Preface

Copyright Notice

Copyright © 2021 MSC Technologies GmbH. All rights reserved.

Copying of this document and providing to others and the use or communication of the contents thereof, is forbidden without express authority of MSC Technologies GmbH. Offenders are liable to the payment of damages.

All rights are reserved in the event of the grant of a patent or the registration of a utility model or design.

Important Information

This documentation is intended for qualified audiences only. The product described herein is not an end user product. It was developed and manufactured for further processing by trained personnel.

Disclaimer

Although this document has been generated with the utmost care no warranty or liability for correctness or suitability for any particular purpose is implied. The information in this document is provided “as is” and is subject to change without notice.

EMC Rules

This unit has to be installed in a shielded housing. If not installed in a properly shielded enclosure, and used in accordance with the instruction manual, this product may cause radio interference in which case the user may be required to take adequate measures at his or her own expense.

Trademarks

All used product names, logos or trademarks are property of their respective owners.

Certification

MSC Technologies GmbH is certified according to DIN EN ISO 9001:2000 standards.

Life-Cycle-Management

MSC products are developed and manufactured according to high quality standards. Our life-cycle-management assures long term availability through permanent product maintenance. Technically necessary changes and improvements are introduced if applicable. A product- change-notification and end-of-life management process assures early information of our customers.

Product Support

MSC engineers and technicians are committed to provide support to our customers whenever needed.

Before contacting Technical Support of MSC Technologies GmbH, please consult the respective pages on our web site at <https://www.msc-technologies.eu/support/boards.html> for the latest documentation, drivers and software downloads.

If the information provided there does not solve your problem, please contact our Avnet Embedded /MSC Technical Support:

Phone: +49 - 8165 906 - 200

Email: support.boards@avnet.eu

Contents

1	INTRODUCTION	11
1.1	Key Features	11
1.2	Block Diagram	16
1.3	Power Supply.....	17
1.4	Power Consumption	17
1.4.1	Use Cases.....	17
1.4.2	Hardware used	18
1.4.3	Measurement Results	18
1.5	Mechanical Dimensions.....	19
1.6	Mechanical Distortion of PCB.....	20
2	THERMAL SPECIFICATIONS.....	21
2.1	Thermal Definitions.....	21
3	MODULE CONNECTOR PINOUT	23
4	MODULE CONNECTOR SIGNAL DESCRIPTION	27
4.1	I ² S	27
4.2	Ethernet	28
4.3	PCI Express	30
4.4	USB.....	30
4.5	Camera	32
4.6	LVDS.....	33
4.7	SPI Bus	35
4.8	CAN	37
4.9	GPIO	38
4.10	SDIO	39
4.11	UART	40
4.12	I ² C Bus.....	41
4.13	Watchdog.....	43
4.14	System Management.....	43
4.15	Boot Options	45
5	FUNCTIONS ON MODULE	47
5.1	CPU Options.....	47
5.2	Power-Up Behaviour.....	47
5.2.1	Power-On Sequencing	48
5.2.2	Reset Sequencing	49
5.3	Memory	50

5.3.1	SDRAM	50
5.3.2	eMMC.....	50
5.3.3	EEPROM.....	51
5.4	Trusted Platform Module	51
5.5	WiFi/Bluetooth	52
5.6	MicroSD Card Socket	52
5.7	Debug Options	52
5.7.1	Debug Connector	52
5.7.2	JTAG Connector.....	54
6	BUS AND ADDRESS MAPPING.....	56
6.1	I ² C Devices	56
6.2	SPI Devices	57
7	BOARD SUPPORT PACKAGE (BSP)	58
7.1	General information	58
7.2	MSC-LDK (Yocto)	58
7.2.1	MSC-LDK Terms	58
7.2.2	Getting Started	58
7.2.3	Setup the MSC-LDK build environment.....	60
7.2.4	Generate images.....	68
7.2.5	Image Deployment	71
7.3	Running an Image	71
7.3.1	Bootimg SPL (secondary program loader)/U-Boot	71
7.3.2	Bootimg OS.....	74
7.3.3	Login to FS.....	80
7.3.4	SMARC GPIO access	80
7.3.5	Bug Reporting	81
7.4	Hotfixes and updating MSC-LDK.....	85
8	TROUBLESHOOTING	87
8.1	Known issues and limitations.....	87
8.1.1	Issue 1. SPI interfaces not available on MSC SM2S-IMX8MINIQC-14N0261I variant Both CAN transceivers drive SPI[0:1]_DIN signals low even though not selected by chip select signal.	87
8.2	Support	87

Figure 1-1: Block Diagram	16
Figure 1-2: Module Dimensions.....	19
Figure 1-3: Overall height without heat spreader of the SMARC™ Module.....	19
Figure 1-4: Distance between mounting holes	20
Figure 2-1: Defined Temperature Point	22
Figure 5-1: Start-up Sequence	48
Figure 5-2: Power-On Timings	48
Figure 5-3: Reset Sequencing.....	49
Figure 5-4: Reset Timings	49
Figure 5-5: Module top side with debug UART FFC connectors marked in red*	53
Figure 5-6: Module top side with MSC UART debug adapter	53
Figure 5-7: Module bottom side with JTAG FFC connectors marked in red*	55
Figure 5-8: Module bottom side with MSC JTAG debug adapter	55
Figure 7-1. RSA key generation	59
Figure 7-2. Clone base MSC-LDK repo.....	61
Figure 7-3. Initial content of the root MSC-LDK directory.....	61
Figure 7-4. Create build directory.	62
Figure 7-5. Base directory content after setup build directory.....	62
Figure 7-6. Enter build directory.	63
Figure 7-7. Prepare docker container for MSC-LDK. Part 1.....	64
Figure 7-8. Prepare docker container for MSC-LDK. Part 2.....	64
Figure 7-9. Prepare docker container for MSC-LDK. Part 3.....	65
Figure 7-10. Start and enter the MSC-LDK container.	65
Figure 7-11. Leave the MSC-LDK container.	66
Figure 7-12. Re-start and re-enter the MSC-LDK container.....	66
Figure 7-13. Stop the MSC-LDK container and release its resources.	67
Figure 7-14. Building msc-image-qt5 image.....	69
Figure 7-15. Content of 'version_layer' file.	70
Figure 7-16. SPL boot selector on EP1 carrier board (S2801).....	71
Figure 7-17. Forced SPL boot from carrier SD card.....	72
Figure 7-18. SPL boot selector on EP1 carrier board (S2801). eMMC flash boot mode (default).....	72
Figure 7-19. SPL boot from module eMMC flash.	73
Figure 7-20. Booting OS (linux) from on-carrier SD card.	74
Figure 7-21. Booting OS (linux) from on-module eMMC flash.	75
Figure 7-22. Preparing U-Boot environment for net boot.	76
Figure 7-23. Booting OS (linux) from network.	78
Figure 7-24. Booting OS (linux) from USB device (pen drive).....	79
Figure 7-25. Bug report. Main page.....	81
Figure 7-26. Bug report. User message editor.	82
Figure 7-27. Bug report. Viewer page.	83
Figure 7-28. Bug report. Zip archive content selector.	84

Figure 7-29. Bug report. Target partition selector.	85
Table 1: Module Power Inputs	17
Table 2: Modules used for Power Consumption Measurement	18
Table 3: Typical Power Consumption Measurement *	18
Table 4: Temperature Range	22
Table 5: Module Connector Pinout	23
Table 6: I ² S Signal Description	27
Table 7: Ethernet Signal Description	28
Table 8: PCIe Signal Description.....	30
Table 9: USB Signal Description	31
Table 10: HDMI Signal Description	32
Table 11: LVDS Signal Description	34
Table 12: SPI Signal Description	36
Table 13: CAN Signal Description *	37
Table 14: GPIO Signal Description.....	38
Table 15: SDIO Signal Description.....	39
Table 16: UART Signal Description.....	41
Table 17: I ² C Signal Description.....	42
Table 18: Watchdog Signal Description	43
Table 19: System Management Signal Description	43
Table 20: Boot Options Control Signal Description	45
Table 21: Boot Options.....	46
Table 22: Available SDRAM options	50
Table 23: Available eMMC devices	50
Table 24: I ² C Interfaces Overview	56
Table 25: SPI Interfaces Overview	57
Table 26: Available images	68
Table 27: Carrier SD Card Boot Mode	74
Table 28: eMMC Boot Mode.....	75
Table 29: Network Boot Mode	76
Table 30: USB Boot Mode	78
Table 31. Available DT-blobs.	79
Table 32. Available user accounts.....	80
Table 33. Available SMARC GPIOs	80

Revision History

Rev.	Date	Description
1.0	July 31, 2020	First Release
1.1	September 16, 2020	Added comment in section EEPROM
1.2	February 10, 2021	Bug fix
1.3	March 17, 2021	Fix USB Boot option and Section 2.1
1.4	April 30, 2021	Corrected Section 4.5 Camera
1.5	August 16, 2021	Changed Debug Adapter to 82479, Avnet CI

Reference Documents

- [1] SMARC™ Specification
Revision 2.0
Last update: June 2nd 2016
<http://www.sget.org>
- [2] IEEE Std. 802.3-2002
802.3-2002.pdf
<http://www.ieee.org>
- [3] i.MX8 Series of Application Processors
IMX8MMIEC.pdf
<http://www.nxp.com>
- [4] Module Datasheet
MSC-SM2S-IMX8MINI.pdf
<https://www.msc-technologies.eu/products-solutions/products/boards/smarc/msc-sm2s-imx8mini.html>
- [5] i.MX Yocto Project User's Guide
i.MX_Yocto_Project_User's_Guide.pdf
Rev. L4.19.35_1.1.0, 11/2019
<http://www.nxp.com>
- [6] i.MX Reference Manual.
i.MX_Reference_Manual.pdf
Rev. L4.19.35-1.1.0, 11/2019
<http://www.nxp.com>
- [7] i.MX Linux User's Guide
i.MX_Linux_User's_Guide.pdf
Rev. L4.19.35_1.1.0, 11/2019
<http://www.nxp.com>
- [8] [i.MX Porting Guide](#)
i.MX_Porting_Guide.pdf
Rev. L4.19.35_1.1.0, 11/2019
<http://www.nxp.com>

[9] Docker documentation
<https://docs.docker.com/>

1 Introduction

SMARC™ modules are compact, highly integrated Single Board Computers.

Typically a SMARC™ module consists of a CPU, chipset, memory, Ethernet controller and USB controller. Interface controllers or connectors (e.g. RJ45) are implemented on a base board on to which the SMARC™ module can be mounted.

In addition to the power supply PCIe, USB, etc. interfaces are present on the connector.

Due to the standardized mechanics and interfaces the system can be scaled arbitrarily. Despite the modular concept the system design is very flat and compact.

SMARC™ modules require a carrier board to build a working system. For evaluation purposes MSC recommends the MSC SM2-MB-EP1 Evaluation Board.

1.1 Key Features

SoC:

- NXP™ i.MX8M Mini ARM® CORTEX™-A53
Assembly options for i.MX8M Mini single, dual or quad-core

SDRAM:

- Up to 4GB LPDDR4

Video:

- Dual Channel LVDS 18-bit/24-bit (1920x1080 max.) or 1x Single Channel LVDS (1366x768 max.) or 1x MIPI-DSI 4 lane (1920x1080 max.)

Audio:

- 2x I²S links for audio codec connection

Camera Interface:

- 1x MIPI CSI-2 2 Lane or 1x MIPI CSI-2 4 Lane *

PCI Express Interface:

- Up to 1x PCIe x1 Gen.2 *

Network:

- Up to 2x 10/100/1000BASE-T Ethernet *
- Optional: H&D Wireless™ Module SPB209A with 802.11 ac/a/b/g/n and Bluetooth 4.2 with BLE support *

USB:

- 1x USB2.0 Host Port with device Interface capability and on-the-go (OTG) support
- 1x USB2.0 Host Port or 4x USB2.0 Host Ports (with additional USB hub populated on module)

GPIO:

- 12x GPIO configurable as input or output (push-pull or open-drain).

SPI:

- 2x SPI with 2 chip selects each *

I²C Bus:

- 1x I²C for power Management functions
- 1x I²C bus for general purposes
- 1x I²C bus for display interface
- 1x I²C bus for camera interface

UART:

- 2x UART without RTS/CTS support
- 2x UART with RTS/CTS support

Flash:

- Up to 64GByte eMMC NAND flash
- Optional: 32Mbit QSPI NOR Flash

Storage Interface:

- 1bit/4bit SD/SDIO/MMC Interface
- Optional: on module microSD Card Socket *

EEPROM:

- 64Kbit EEPROM for module information and user applications

CAN:

- Optional: 2x CAN 2.0B (up to 1Mbps) *

Real-time Clock:

- High accuracy RTC
- Optional RTC with temperature compensated DTCXO

Watchdog:

- Module provides a watchdog connected to the SMARC[™] connector

Security Device:

- Advanced Security, Safety, and Reliability integrated in the SOC
- Optional Trusted Platform Module 2.0 (TPM): Infineon[™] SLB9673 or STMicroelectronics[™] ST33TPHF20I2C

Environment Temperature:

- 0° ... 70°C (all components commercial temp. or better)
- -40° ... 85°C (all components industrial temp.)
- -40° ... 85°C (storage)

Environment Humidity:

- 5 ... 95% (operating)
- 5... 95% (storage)

*NOTES:

- the second Ethernet interfaces makes use of the PCIe interface, so the PCIe Lane is not available when second Ethernet is populated (mutually exclusive assembly option)

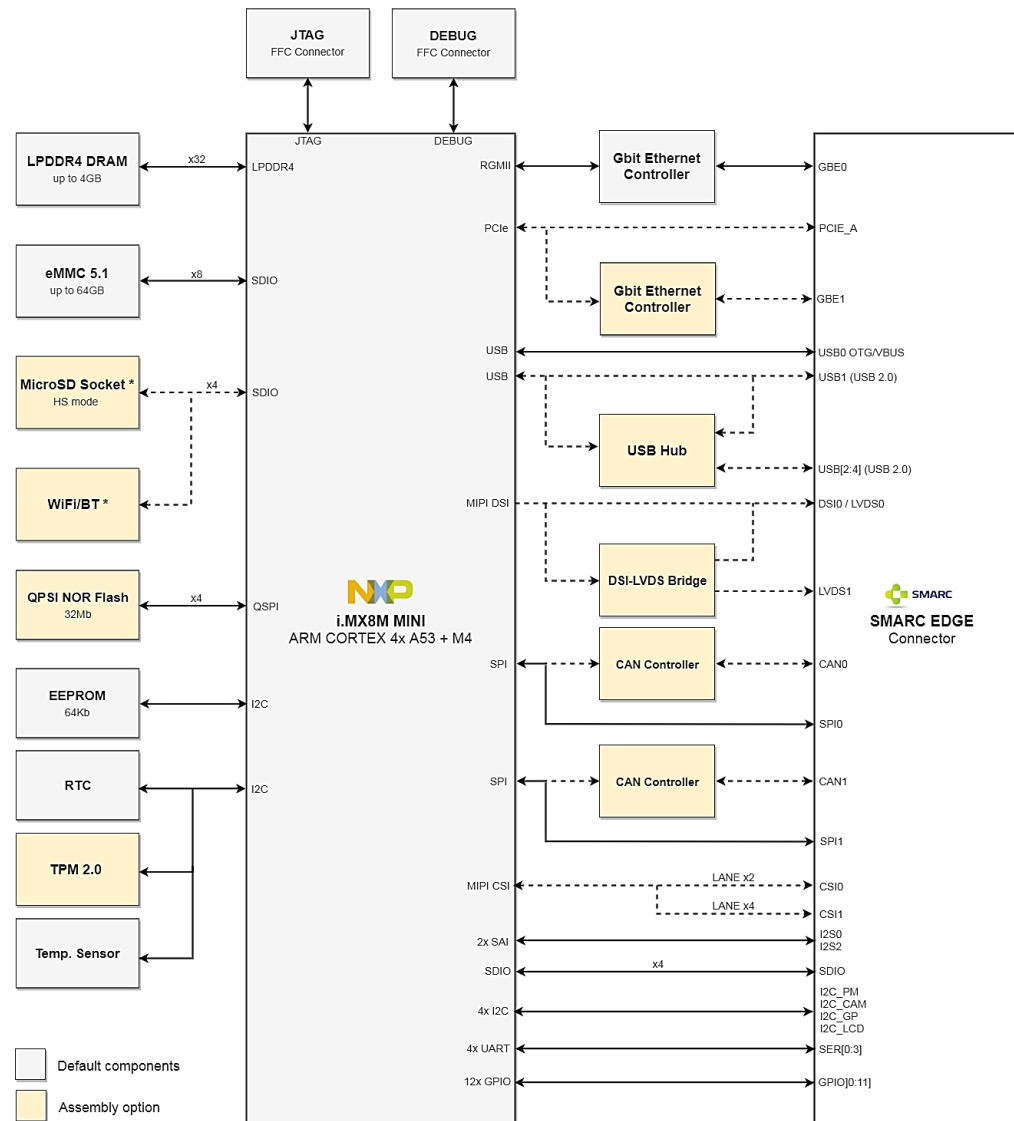
microSD card socket and Wireless/BT module are mutually exclusive assembly options.

CSI 2-Lane and CSI 4-Lane are mutual exclusive assembly options.

The SPI interface(s) are used to implement the optional CAN interface(s) and so are mutually exclusive options. If two CAN controllers are used, then the two SPI buses are not available.

1.2 Block Diagram

Figure 1-1: Block Diagram



1.3 Power Supply

Table 1: Module Power Inputs

Power Rail	Description		
VDD_IN	Primary power input	Nominal	+5V
		Voltage Range	+4.75V ... +5.25V
		Max. Input Ripple	±100mV
		Max. Rate of Voltage Rise	< 250V/s
VDD_RTC	May be sourced from a Lithium cell or a Super Cap.	Nominal	+3V
		Voltage Range	+1.5V ... +5.5V
		Max. Input Ripple	±20mV
		Current	0.18µA typ. @ VDD = 3V (1µA max.)
GND	Power and signal return path. All available GND connector pins shall be connected and tied to Carrier Board GND plane.		

1.4 Power Consumption

1.4.1 Use Cases

- Uboot Idle: Ethernet link established, no display used, no USB devices
- Linux Idle: Ethernet link established, no display used, no USB devices
- Linux Heavy Load: CPU load 100% on each core, memory tester, LVDS display, Ethernet traffic generated with iperf
- Deep Sleep: wake on power button press

1.4.2 Hardware used

Table 2: Modules used for Power Consumption Measurement

Order Number	Reference	CPU	RAM	Temp. Range
78402	MSC SM2S-IMX8MINIQC-14N0261I PCBFTX	i.MX8M Mini Quad, Quad-Core Cortex-A53 at 1.8GHz	2G LPDDR4	-40°C to +85°C
78370	MSC SM2S-IMX8MINIQC-13N4200I PCBFTX	i.MX8M Mini Quad, Quad-Core Cortex-A53 at 1.8GHz	2G LPDDR4	-40°C to +85°C
79649	MSC SM2S-IMX8MINIQC-03N0840E PCBFTX	i.MX8M Mini Quad, Quad-Core Cortex-A53 at 1.8GHz	1G LPDDR4	-25°C to +85°C
78368	MSC SM2S-IMX8MINIDC-03N4200I PCBFTX	i.MX8M Mini Dual, Dual-Core Cortex-A53 at 1.8GHz	1G LPDDR4	-40°C to +85°C
78404	MSC SM2S-IMX8MINISC-03N4210I PCBFTX	i.MX8M Mini Solo, Single-Core Cortex-A53 at 1.8GHz	1G LPDDR4	-40°C to +85°C
78406	MSC SM2S-IMX8MINISCL-03N0880I PCBFTX	i.MX8M Mini Solo Lite, Single-Core Cortex-A53 at 1.8GHz	1G LPDDR4	-40°C to +85°C

1.4.3 Measurement Results

Table 3: Typical Power Consumption Measurement *

Order Number	Reference	Uboot Idle [W]	Linux idle [W]	Linux Heavy Load [W]	Deep Sleep [W]
78402	MSC SM2S-IMX8MINIQC-14N0261I PCBFTX	2.045	1.790	4.650	0.315
78370	MSC SM2S-IMX8MINIQC-13N4200I PCBFTX	2.030	1.780	4.610	0.315
79649	MSC SM2S-IMX8MINIQC-03N0840E PCBFTX	1.950	1.710	4.320	0.140
78368	MSC SM2S-IMX8MINIDC-03N4200I PCBFTX	1.905	1.615	3.820	0.350
78404	MSC SM2S-IMX8MINISC-03N4210I PCBFTX	1.810	1.580	3.050	0.310
78406	MSC SM2S-IMX8MINISCL-03N0880I PCBFTX	1.925	1.660	3.046	0.140

*NOTE: Unless stated otherwise, all measurements were taken at room temperature approx 23°C.

1.5 Mechanical Dimensions

Figure 1-2: Module Dimensions

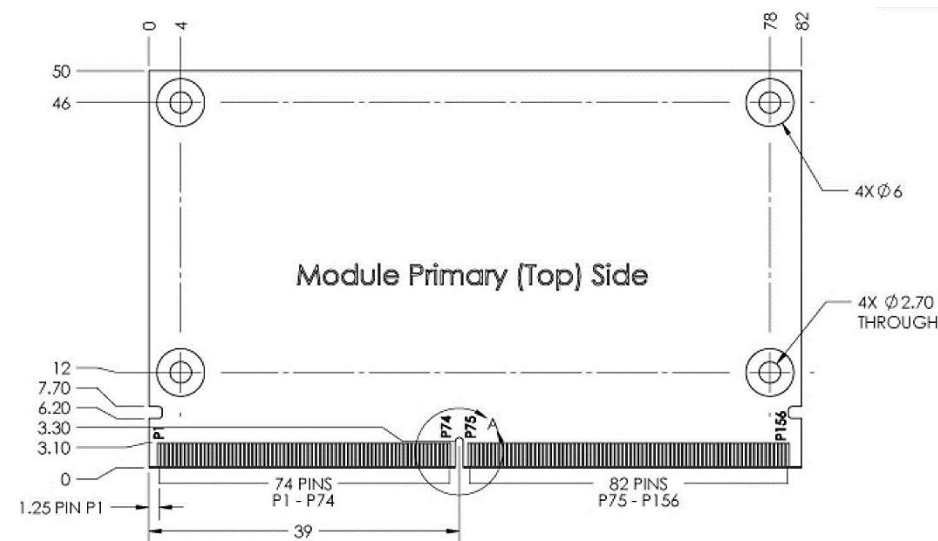
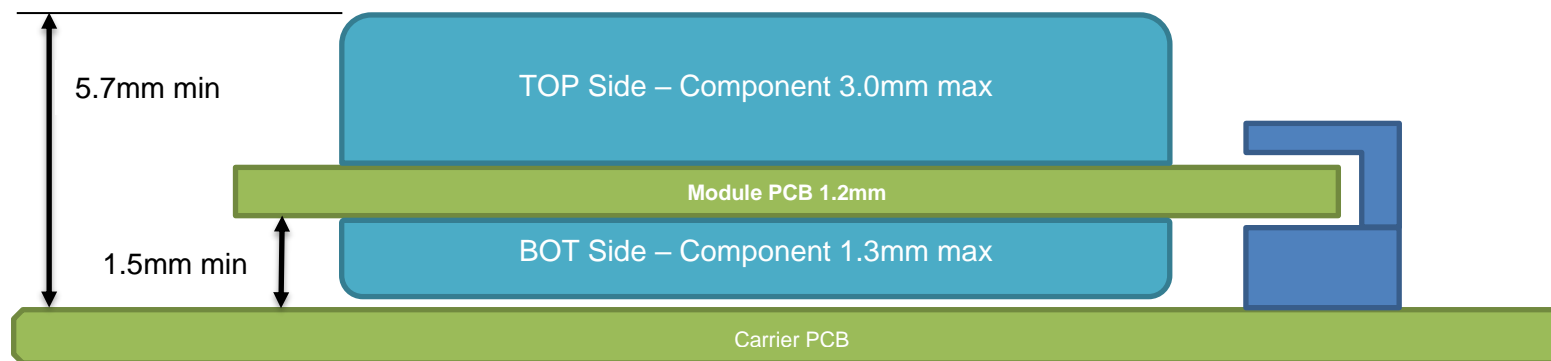


Figure 1-3: Overall height without heat spreader of the SMARC™ Module



The overall height is dependent on the MXM3 connector used on the baseboard.

1.6 Mechanical Distortion of PCB

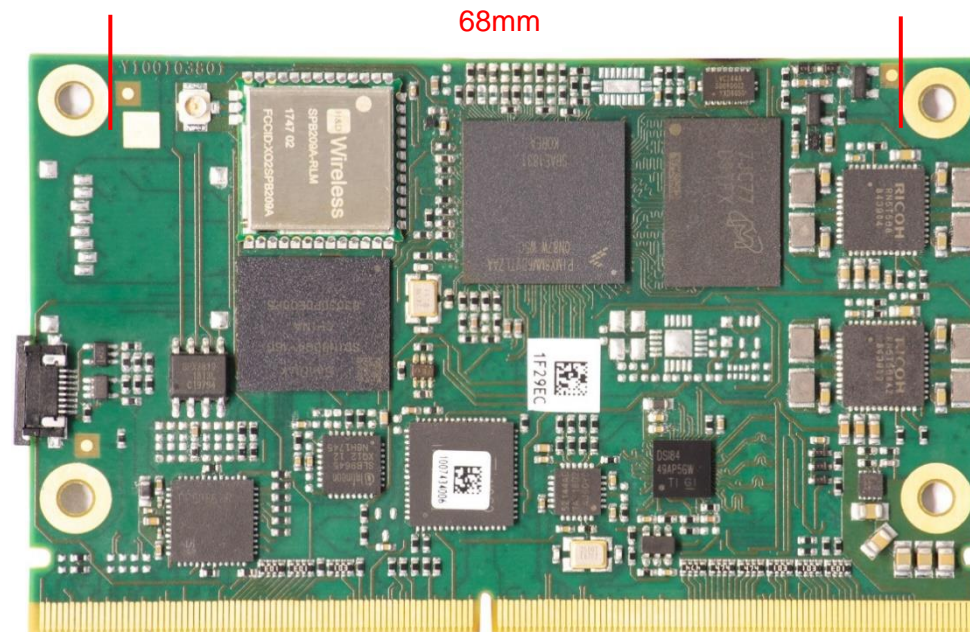
For thermal heat dissipation the heat sink needs to have a good mechanical contact to the CPU housing which means the heat sink should be mounted such that there is some mechanical stress applied. The higher the force applied the better the thermal resistance and consequently the better the thermal cooling. This pressure may result in a slight mechanical bending of the SMARC module PCB.

Production tolerance, material deviation and thermal expansion lead to a range of possible pressure range and bending. A negative pressure with an air gap between the heat spreader and the chip case needs to be avoided and likewise too much distortion.

Component types and their distance to the heat spreader mounting holes need to be considered.

Referring to data sheets of the relevant parts and referring to AEC-Q200 the bending needs to be less than 1mm over 90mm. $(1.11\%) \rightarrow 0.75\text{mm}$

Figure 1-4: Distance between mounting holes



2 Thermal Specifications

The cooling solution for a SMARC™ module is based on a heat spreader or heat-sink concept.

A heat spreader or heat sink is typically made of aluminum mounted on top of the module. The connection between this plate and the module components is made using thermal interface materials such as phase change foils, gap pads and copper or aluminum blocks. A very good thermal conductivity is required in order to transfer the heat from the SoC to the heat spreader plate. The heat sink concept maximizes the surface contact area with the cooling medium.

Heat spreader and heat sinks used by the MSC module are thermally attached using phase change material. Stand-alone heat sinks generally offer best thermal transfer characteristics. Please contact MSC Technologies support for a suitable heat-sink solution for the MSC SM2S-IMX8MINI SMARC™ Module.

The main goal for the thermal design of a system is that each device on the module is operated within its specified thermal limits. There may be system implementations where the heat spreader temperature could be higher. In such a case the cooling solution design should be validated such that the thermal specifications of all the components on the module are not violated across the system operating temperature range even under worst case conditions.

2.1 Thermal Definitions

- Tpcb** This is the temperature on the surface of the module PCB at point P (defined below).
- Tpcb_max** The maximum temperature allowed for the surface of the module PCB at point P (defined below).
- Tpcb_min** This is defined as the minimum temperature allowed for the surface of the module PCB.
- P** The point on the module PCB where the PCB temperature must be measured.

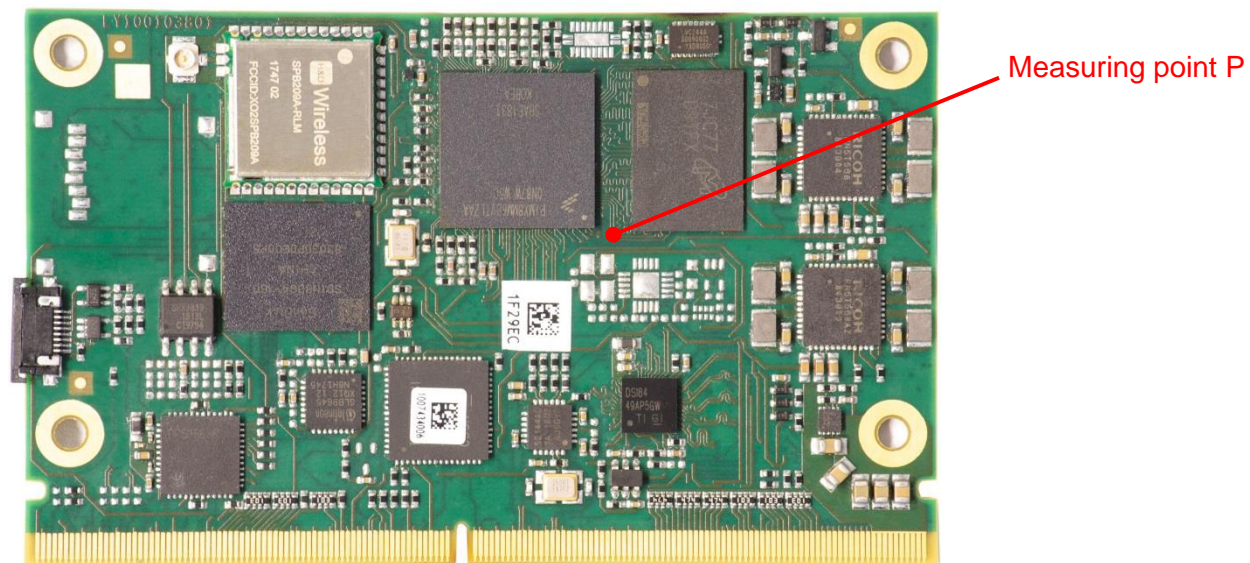
The stabilized temperature measured on the heat spreader and the module PCB during runtime mostly depends on the computing power demand from the application and the cooling solution implemented in the system. It is the responsibility of the system designer to provide a cooling solution in addition to the heat spreader that fulfils the requirements of the application.

The temperature at the defined point on the PCB shall not exceed the temperature range in the following table.

Table 4: Temperature Range

Module Variant	Tpcb_min	Tpcb_max
Module variants with commercial temperature components	0 °C	+70 °C
Module variants with extended temperature components	-25 °C	+ 85 °C
Module variants with industrial temperature components	- 40 °C	+ 85 °C

Figure 2-1: Defined Temperature Point



3 Module Connector Pinout

The pinout of the module connector is based on the SMARC™ specification[1].

Table 5: Module Connector Pinout

Primary (Top) Side		Secondary (Bottom) Side	
P1	SMB_ALERT_1V8#	S1	I2C_CAM1_CK
P2	GND	S2	I2C_CAM1_DAT
P3	CSI1_CK+	S3	GND
P4	CSI1_CK-	S4	NC
P5	NC	S5	I2C_CAM0_CK
P6	GBE0_SDP	S6	CAM_MCK
P7	CSI1_RX0+	S7	I2C_CAM0_DAT
P8	CSI1_RX0-	S8	CSI0_CK+
P9	GND	S9	CSI0_CK-
P10	CSI1_RX1+	S10	GND
P11	CSI1_RX1-	S11	CSI0_RX0+
P12	GND	S12	CSI0_RX0-
P13	CSI1_RX2+	S13	GND
P14	CSI1_RX2-	S14	CSI0_RX1+
P15	GND	S15	CSI0_RX1-
P16	CSI1_RX3+	S16	GND
P17	CSI1_RX3-	S17	GBE1_MDI0+

Primary (Top) Side		Secondary (Bottom) Side	
P18	GND	S18	GBE1_MDI0-
P19	GBE0_MDI3-	S19	GBE1_LINK100#
P20	GBE0_MDI3+	S20	GBE1_MDI1+
P21	GBE0_LINK100#	S21	GBE1_MDI1-
P22	GBE0_LINK1000#	S22	GBE1_LINK1000#
P23	GBE0_MDI2-	S23	GBE1_MDI2+
P24	GBE0_MDI2+	S24	GBE1_MDI2-
P25	GBE0_LINK_ACT#	S25	GND
P26	GBE0_MDI1-	S26	GBE1_MDI3+
P27	GBE0_MDI1+	S27	GBE1_MDI3-
P28	NC	S28	NC
P29	GBE0_MDI0-	S29	NC
P30	GBE0_MDI0+	S30	NC
P31	SPI0_CS1#	S31	GBE1_LINK_ACT#
P32	GND	S32	NC
P33	SDIO_WP	S33	NC
P34	SDIO_CMD	S34	GND

Primary (Top) Side		Secondary (Bottom) Side	
P35	SDIO_CD#	S35	USB4+
P36	SDIO_CK	S36	USB4-
P37	SDIO_PWR_EN	S37	NC
P38	GND	S38	AUDIO_MCK
P39	SDIO_D0	S39	I2S0_LRCK
P40	SDIO_D1	S40	I2S0_SDOOUT
P41	SDIO_D2	S41	I2S0_SDIN
P42	SDIO_D3	S42	I2S0_CK
P43	SPI0_CS0#	S43	NC
P44	SPI0_CK	S44	NC
P45	SPI0_DIN	S45	NC
P46	SPI0_DO	S46	NC
P47	GND	S47	GND
P48	NC	S48	I2C_GP_CK
P49	NC	S49	I2C_GP_DAT
P50	GND	S50	I2S2_LRCK
P51	NC	S51	I2S2_SDOOUT
P52	NC	S52	I2S2_SDIN
P53	GND	S53	I2S2_CK
P54	SPI1_CS0#	S54	NC
P55	SPI1_CS1#	S55	NC

Primary (Top) Side		Secondary (Bottom) Side	
P56	SPI1_CK	S56	NC
P57	SPI1_DIN	S57	NC
P58	SPI1_DO	S58	NC
P59	GND	S59	NC
P60	USB0+	S60	NC
P61	USB0-	S61	GND
P62	USB0_EN_OC#	S62	NC
P63	USB0_VBUS_DET	S63	NC
P64	USB0_OTG_ID	S64	GND
P65	USB1+	S65	NC
P66	USB1-	S66	NC
P67	USB1_EN_OC#	S67	GND
P68	GND	S68	USB3+
P69	USB2+	S69	USB3-
P70	USB2-	S70	GND
P71	USB2_EN_OC#	S71	NC
P72	NC	S72	NC
P73	NC	S73	GND
P74	USB3_EN_OC#	S74	NC
		S75	NC
KEY		KEY	

Primary (Top) Side		Secondary (Bottom) Side	
P75	PCIE_A_RST#	S76	NC
P76	USB4_EN_OC#	S77	NC
P77	NC	S78	NC
P78	NC	S79	NC
P79	GND	S80	GND
P80	NC	S81	NC
P81	NC	S82	NC
P82	GND	S83	GND
P83	PCIE_A_REFCK+	S84	NC
P84	PCIE_A_REFCK-	S85	NC
P85	GND	S86	GND
P86	PCIE_A_RX+	S87	NC
P87	PCIE_A_RX-	S88	NC
P88	GND	S89	GND
P89	PCIE_A_TX+	S90	NC
P90	PCIE_A_TX-	S91	NC
P91	GND	S92	GND
P92	NC	S93	NC
P93	NC	S94	NC
P94	GND	S95	NC
P95	NC	S96	NC

Primary (Top) Side		Secondary (Bottom) Side	
P96	NC	S97	NC
P97	GND	S98	NC
P98	NC	S99	NC
P99	NC	S100	NC
P100	GND	S101	GND
P101	NC	S102	NC
P102	NC	S103	NC
P103	GND	S104	NC
P104	NC	S105	NC
P105	NC	S106	NC
P106	NC	S107	LCD1_BKLT_EN
P107	NC	S108	LVDS1_CK+ / DSI1_CLK+
P108	GPIO0	S109	LVDS1_CK- / DSI1_CLK-
P109	GPIO1	S110	GND
P110	GPIO2	S111	LVDS1_0+ / DSI1_D0+
P111	GPIO3	S112	LVDS1_0- / DSI1_D0-
P112	GPIO4	S113	NC
P113	GPIO5 (PWM)	S114	LVDS1_1+ / DSI1_D1+
P114	GPIO6 (CLK)	S115	LVDS1_1- / DSI1_D1-
P115	GPIO7	S116	LCD1_VDD_EN
P116	GPIO8	S117	LVDS1_2+ / DSI1_D2+

Primary (Top) Side		Secondary (Bottom) Side	
P117	GPIO9	S118	LVDS1_2- / DSI1_D2-
P118	GPIO10	S119	GND
P119	GPIO11	S120	LVDS1_3+ / DSI1_D3+
P120	GND	S121	LVDS1_3- / DSI1_D3-
P121	I2C_PM_CK	S122	LCD1_BKLT_PWM
P122	I2C_PM_DAT	S123	NC
P123	BOOT_SEL0#	S124	GND
P124	BOOT_SEL1#	S125	LVDS0_0+ / DSI0_D0+
P125	BOOT_SEL2#	S126	LVDS0_0- / DSI0_D0-
P126	RESET_OUT#	S127	LCD0_BKLT_EN
P127	RESET_IN#	S128	LVDS0_1+ / DSI0_D1+
P128	POWER_BTN#	S129	LVDS0_1- / DSI0_D1-
P129	SER0_TX	S130	GND
P130	SER0_RX	S131	LVDS0_2+ / DSI0_D2+
P131	SER0_RTS#	S132	LVDS0_2- / DSI0_D2-
P132	SER0_CTS#	S133	LCD0_VDD_EN
P133	GND	S134	LVDS0_CK+ / DSI0_CLK+
P134	SER1_TX	S135	LVDS0_CK- / DSI0_CLK-
P135	SER1_RX	S136	GND
P136	SER2_TX	S137	LVDS0_3+ / DSI0_D3+
P137	SER2_RX	S138	LVDS0_3- / DSI0_D3-

Primary (Top) Side		Secondary (Bottom) Side	
P138	SER2_RTS#	S139	I2C_LCD_CK
P139	SER2_CTS#	S140	I2C_LCD_DAT
P140	SER3_TX	S141	LCD0_BKLT_PWM
P141	SER3_RX	S142	NC
P142	GND	S143	GND
P143	CAN0_TX	S144	NC
P144	CAN0_RX	S145	WDT_TIME_OUT#
P145	CAN1_TX	S146	PCIE_WAKE#
P146	CAN1_RX	S147	VDD_RTC
P147	VCC	S148	LID#
P148	VCC	S149	SLEEP#
P149	VCC	S150	VIN_PWR_BAD#
P150	VCC	S151	CHARGING#
P151	VCC	S152	CHARGER_PRSENT#
P152	VCC	S153	CARRIER_STBY#
P153	VCC	S154	CARRIER_PWR_ON
P154	VCC	S155	FORCE_RECOV#
P155	VCC	S156	BATLOW#
P156	VCC	S157	TEST#
		S158	GND

4 Module Connector Signal Description

In the following tables signals are marked with the power rail associated with the pin, and for input and I/O pins, with the input voltage tolerance. The pin power rail and the pin input voltage tolerance may be different.

Output pins are also classified as push pull (PP) or open drain (OD). The column “PU/PD” describes pull-up resistors (PU) or pull-down resistors (PD) implemented on the module.

4.1 I²S

The module provides two I²S Links for connecting I²S codecs on the carrier board. Driver support for I²S is only available for Linux.

Some features:

- Programmable data interface modes such as I2S, LSB or MSB-justified
- Programmable word length (16, 20, 24 or 28bits)
- AC97 and TDM support
- Time Slot Mask Registers for reduced ARM platform overhead (for both Transmit and Receive)
- 128-word Transmit FIFO and 128-word Receive FIFO

Table 6: I²S Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
I2S0_LRCK	O PP	1.8V CMOS	AC14	SAI5_RXD1	1.8V		Sample-synchronization signal to the codec(s).
I2S0_CK	O PP	1.8V CMOS	AD13	SAI5_RXD2	1.8V		Serial data clock
I2S0_SDOUT	O PP	1.8V CMOS	AC13	SAI5_RXD3	1.8V		Serial TDM data output to the codec.
I2S0_SDIN	I	1.8V CMOS	AD18	SAI5_RXD0	1.8V		Serial TDM data inputs from the codec.
I2S2_LRCK	O PP	1.8V CMOS	AB19	SAI1_TXFS	1.8V		Sample-synchronization signal to the codec(s).
I2S2_CK	O PP	1.8V CMOS	AC18	SAI1_TXC	1.8V		Serial data clock
I2S2_SDOUT	O PP	1.8V CMOS	AG20	SAI1_TXD0	1.8V		Serial TDM data output to the codec.

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
I2S2_SDIN	I	1.8V CMOS	AG15	SAI1_RXD0	1.8V		Serial TDM data inputs from the codec.
AUDIO_MCK	O PP	1.8V CMOS	H26	CLKOUT1	1.8V		Connected by default.

4.2 Ethernet

Based on Texas Instruments™ DP83867 Ethernet Controller the module provides 10/100/1000 Mbps Ethernet with MDI differential pairs on GBE0 Interface.

The module provides a second 10/100/1000 Mbps Ethernet option (GBE1 interface) based on Intel's™ I210 Ethernet Controller. In case GBE1 interface is implemented PCIe will no longer be available. PCIE_A and GBE1 are mutually exclusive assembly options.

Both Ethernet Controllers include a voltage mode line driver, so using an analog powered center tap is not allowed. Therefore Pin P28 GBE0_CTREF and S28 GBE1_CTREF specified in the SMARC™ specification 2.0 are not connected on the module and can be left unconnected.

Both Ethernet Controller have built in termination resistors. As a result, no external termination resistors should be used. Please refer to the DP83867EVM schematics for correct magnetics wiring. Each Center tap of the magnetics should be independently de-coupled to ground via a 0.1µF capacitor.

Table 7: Ethernet Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
GBE0_MDI0+ GBE0_MDI0-	I/O	Analog	n.a.	n.a.	3.3V		Media Dependent Interface Differential Pair 0 Used for the receive pair in 10/100 Mbit/s mode
GBE0_MDI1+ GBE0_MDI1-	I/O	Analog	n.a.	n.a.	3.3V		Media Dependent Interface Differential Pair 1 Used for the transmit pair in 10/100 Mbit/s mode
GBE0_MDI2+ GBE0_MDI2-	I/O	Analog	n.a.	n.a.	3.3V		Media Dependent Interface Differential Pair 2 This signal pair is only used for 1000Mbit/s mode.
GBE0_MDI3+ GBE0_MDI3-	I/O	Analog	n.a.	n.a.	3.3V		Media Dependent Interface Differential Pair 3 This signal pair is only used for 1000Mbit/s mode.
GBE0_LINK_ACT#	O OD	3.3V CMOS	n.a.	n.a.	3.3V		Link/Activity Indication, active low and 24mA sink capability.

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
GBE0_LINK100#	O OD	3.3V CMOS	n.a	n.a	3.3V		Link Speed Indication for 100Mbps, active low, and 24mA sink capability.
GBE0_LINK1000#	O OD	3.3V CMOS	n.a.	n.a.	3.3V		Link Speed Indication for 1000Mbps, active low, and 24mA sink capability.
GBE1_MDI0+ GBE1_MDI0-	I/O	Analog	n.a.	n.a.	3.3V		Media Dependent Interface Differential Pair 0 Used for the receive pair in 10/100 Mbit/s mode
GBE1_MDI1+ GBE1_MDI1-	I/O	Analog	n.a.	n.a.	3.3V		Media Dependent Interface Differential Pair 1 Used for the transmit pair in 10/100 Mbit/s mode
GBE1_MDI2+ GBE1_MDI2-	I/O	Analog	n.a.	n.a.	3.3V		Media Dependent Interface Differential Pair 2 This signal pair is only used for 1000Mbit/s mode.
GBE1_MDI3+ GBE1_MDI3-	I/O	Analog	n.a.	n.a.	3.3V		Media Dependent Interface Differential Pair 3 This signal pair is only used for 1000Mbit/s mode.
GBE1_LINK_ACT#	O OD	3.3V CMOS	n.a.	n.a.	3.3V		Link/Activity Indication, active low and 24mA sink capability.
GBE1_LINK100#	O OD	3.3V CMOS	n.a	n.a	3.3V		Link Speed Indication for 100Mbps, active low, and 24mA sink capability.
GBE1_LINK1000#	O OD	3.3V CMOS	n.a.	n.a.	3.3V		Link Speed Indication for 1000Mbps, active low, and 24mA sink capability.

4.3 PCI Express

The i.MX8M Mini SoC supports PCIe x1 Gen2 lane.*

Table 8: PCIe Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
PCIE_A_TX+ PCIE_A_TX-	O	LVDS PCIe	B20 A20	PCIE_TXN_P PCIE_TXN_N	According to PCIe spec		PCI Express Differential Transmit Pairs. AC coupled on module
PCIE_A_RX+ PCIE_A_RX-	I	LVDS PCIe	B19 A19	PCIE_RXN_P PCIE_RXN_N	According to PCIe spec		PCI Express Differential Receive Pairs
PCIE_A_REFCK+ PCIE_A_REFCK-	O	LVDS PCIe	B21 A21	PCIE_CLK_N PCIE_CLK_P	According to PCIe spec		PCI Express Reference Clock. AC coupled on module. Clock enabled by default.
PCIE_WAKE#	I	3.3V CMOS	N27	NAND_RE_B	3.3V	PU 10k	PCI Express Wake signal. Asserted by device when requesting wake up. (CPU GPIO3_IO15)
PCIE_A_RST#	O PP	3.3V CMOS	K27	NAND_CLE	3.3V		PCI Express Reset output signal.

*NOTE: In case GBE1 Interface is implemented on the module the PCIe Interface will no longer be available. Pins can be left unconnected

4.4 USB

The USB controller supports USB 2.0.

Depending on the module variant the following USB lane options are available:

- **Option 1** with USB 2.0 Hub: USB[0] = USB 2.0 host/device OTG compliant
USB[1:4] = USB 2.0 host
- **Option 2** without USB 2.0 Hub: USB[0] = USB 2.0 host/device OTG compliant
USB[1] = USB 2.0 host.

Table 9: USB Signal Description

Signal	Option Availability	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
USB0+ USB0-	1 & 2	I/O	USB	B22 A22	USB1_DP USB1_DN	3.3V		Differential USB 2.0 data pairs connected to SoC. Can be configured as host or device.
USB1+ USB1-	1	I/O	USB	n.a.	n.a.	3.3V		Differential USB 2.0 data pairs connected to USB hub. Can be configured as host only.
	2	I/O	USB	B23 A23	USB2_DP USB2_DN	3.3V		Differential USB 2.0 data pairs connected to SoC. Can be configured as host only.
USB[2:4]+ USB[2:4]-	1	I/O	USB	n.a.	n.a.	3.3V		Differential USB 2.0 data pairs connected to USB hub. Can be configured as host only.
USB0_VBUS_DET	1 & 2	I	Analog	F22	USB1_VBUS	5V		external VBUS detection pin
USB0_OTG_ID	1 & 2	I	3.3V CMOS	D22	USB1_ID	3.3V	PU 10k 3.3V	USB host/client control select pin for the USB controller on the module
USB0_EN_OC#	1 & 2	I/O OD	3.3V CMOS	AB10	GPIO1_IO12	3.3V	PU 10k 3.3V	Host/client dependent enable signal for USB power switch on the carrier board. *
USB1_EN_OC#	1	I/O OD	3.3V CMOS	n.a	n.a	3.3V	PU 10k 3.3V	Multi-function signal for enabling USB power and indicating an over-current event. *
	2	I/O OD	3.3V CMOS	P26	NAND_READY_B	3.3V	PU 10k 3.3V	Multi-function signal for enabling USB power and indicating an over-current event. *
USB2_EN_OC#	1	I/O OD	3.3V CMOS	n.a.	n.a.	3.3V	PU 10k 3.3V	Multi-function signal for enabling USB power and indicating an over-current event. *

Signal	Option Availability	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
USB3_EN_OC#	1	I/O OD	3.3V CMOS	n.a.	n.a.	3.3V	PU 10k 3.3V	Multi-function signal for enabling USB power and indicating an over-current event. *
USB4_EN_OC#	1	I/O OD	3.3V CMOS	n.a.	n.a.	3.3V	PU 10k 3.3V	Multi-function signal for enabling USB power and indicating an over-current event. *

*NOTE: Module pulls USB[0:4]_EN_OC# low to disable USB power delivery on carrier board. Carrier board pulls the signal low to indicate over-current situation. If over-current monitoring is desired, an OD driver should be implemented. In case no power switches are used, USB[0:4]_EN_OC# should be left unconnected.

4.5 Camera

MIPI CSI-2 interface is supported on CSI0 with 2 lanes or on CSI1 with 4 lanes, mutually exclusive assembly option.

Table 10: HDMI Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
CSI0_RX[0]- CSI0_RX[0]+	I	1.8V CMOS	A14 B14	MIPI_CSI_D0_N MIPI_CSI_D0_P	1.8V		CSI differential data inputs
CSI0_RX[1]- CSI0_RX[1]+	I	1.8V CMOS	A15 B15	MIPI_CSI_D1_N MIPI_CSI_D1_P	1.8V		CSI differential data inputs
CSI1_RX[0]- CSI1_RX[0]+	I	1.8V CMOS	A14 B14	MIPI_CSI_D0_N MIPI_CSI_D0_P	1.8V		CSI differential data inputs
CSI1_RX[1]- CSI1_RX[1]+	I	1.8V CMOS	A15 B15	MIPI_CSI_D1_N MIPI_CSI_D1_P	1.8V		CSI differential data inputs
CSI1_RX[2]- CSI1_RX[2]+	I	1.8V CMOS	A17 B17	MIPI_CSI_D2_N MIPI_CSI_D2_P	1.8V		CSI differential data inputs
CSI1_RX[3]- CSI1_RX[3]+	I	1.8V CMOS	A18 B18	MIPI_CSI_D3_N MIPI_CSI_D3_P	1.8V		CSI differential data inputs

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
CSI0_CK+ CSI0_CK-	I	1.8V CMOS	A22 B22	MIPI_CSI_CLK_N MIPI_CSI_CLK_P	1.8V		CSI differential clock inputs
CSI1_CK+ CSI1_CK-	I	1.8V CMOS	A19 B19	MIPI_CSI_CLK_N MIPI_CSI_CLK_P	1.8V		CSI differential clock inputs
CAM_MCK	O PP	1.8V CMOS	J26	CLKOUT2	1.8V		Master clock for camera
I2C_CAM0_CK	O OD	1.8V CMOS	F15	UART2_RXD	1.8V	PU 2.2k 1.8V	CAM0 DDC clock line (CPU GPIO5_IO24) *
I2C_CAM0_DAT	I/O OD	1.8V CMOS	E15	UART2_TXD	1.8V	PU 2.2k 1.8V	CAM0 DDC data line (CPU GPIO5_IO25) *
I2C_CAM1_CK	O OD	1.8V CMOS	F15	UART2_RXD	1.8V	PU 2.2k 1.8V	CAM0 DDC clock line (CPU GPIO5_IO24) *
I2C_CAM1_DAT	I/O OD	1.8V CMOS	E15	UART2_TXD	1.8V	PU 2.2k 1.8V	CAM1 DDC data line (CPU GPIO5_IO25) *
CAM0_PWR#	I/O OD	1.8V CMOS	AG14	GPIO1_IO00	1.8V	PU 470k 1.8V	CAM0 Power Enable, active low. GPIO0 alternate use
CAM1_PWR#	I/O OD	1.8V CMOS	AF14	GPIO1_IO01	1.8V	PU 470k 1.8V	CAM1 Power Enable, active low. GPIO1 alternate use
CAM0_RST#	I/O OD	1.8V CMOS	AF13	GPIO1_IO03	1.8V	PU 470k 1.8V	CAM0 Reset, active low. GPIO2 alternate use
CAM1_RST#	I/O OD	1.8V CMOS	AF12	GPIO1_IO05	1.8V	PU 470k 1.8V	CAM1 Reset, active low. GPIO3 alternate use

*NOTE: CSI0 and CSI1 share the same I²C bus.
CAM0 and CAM1 I²C drivers are implemented using bit-banged IO operation.

4.6 LVDS

LVDS channel 0 and 1 are available on the SMARC[™] module depending on module variant. An on-module DSI bridge converts the MIPI DSI data stream to Single-Link LVDS and Dual-link LVDS.

Features:

- Single-Link and Dual-Link with four data lanes per link
- Supports 18bpp and 24bpp
- Pixel clock up to 154 MHz

Table 11: LVDS Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
LVDS0_0+ / DSI0_D0+ LVDS0_0- / DSI0_D0-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 0 differential pair
LVDS0_1+ / DSI0_D1+ LVDS0_1- / DSI0_D1-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 0 differential pair
LVDS0_2+ / DSI0_D2+ LVDS0_2- / DSI0_D2-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 0 differential pair
LVDS0_3+ / DSI0_D3+ LVDS0_3- / DSI0_D3-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 0 differential pair
LVDS0_CLK+ / DSI0_CLK+ LVDS0_CLK- / DSI0_CLK-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 0 differential clock
LVDS1_0+ / DSI1_D0+ LVDS1_0- / DSI1_D0-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 1 differential pair
LVDS1_1+ / DSI1_D1+ LVDS1_1- / DSI1_D1-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 1 differential pair
LVDS1_2+ / DSI1_D2+ LVDS1_2- / DSI1_D2-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 1 differential pair
LVDS1_3+ / DSI1_D3+ LVDS1_3- / DSI1_D3-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 1 differential pair

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
LVDS1_CK+ / DSI1_CLK+ LVDS1_CK- / DSI1_CLK-	O	LVDS	n.a.	n.a.	2.8V		LVDS Channel 1 differential clock
LCD0_VDD_EN	O PP	1.8V CMOS	AF21	SAI1_TXD3	1.8V	PD 10k	LCD0 panel power enable (CPU GPIO4_IO15)
LCD0_BKLT_EN	O PP	1.8V CMOS	AF20	SAI1_TXD1	1.8V		LCD0 backlight enable (CPU GPIO4_IO13)
LCD0_BKLT_PWM	O PP	1.8V CMOS	AF8	SPDIF_EXT_CLK	1.8V	PD 10k	LCD0 backlight brightness control (PWM1_OUT)
LCD1_VDD_EN	O PP	1.8V CMOS	AG22	SAI1_TXD4	1.8V	PD 10k	LCD1 panel power enable (CPU GPIO4_IO16)*
LCD1_BKLT_EN	O PP	1.8V CMOS	AG21	SAI1_TXD2	1.8V		LCD1 backlight enable (CPU GPIO4_IO14)*
LCD1_BKLT_PWM	O PP	1.8V CMOS	AG9	SPDIF_RX	1.8V	PD 10k	LCD1 backlight brightness control (PWM2_OUT)*
I2C_LCD_CK	O PP	1.8V CMOS	D13	I2C4_SCL	1.8V	PU 2.2k 1.8V	I ² C clock output for LVDS display use
I2C_LCD_DAT	I/O OD	1.8V CMOS	E13	I2C4_SDA	1.8V	PU 2.2k 1.8V	I ² C data line for LVDS display use

*NOTE: LCD1_VDD_EN, LCD1_BKLT_EN and LCD1_BKLT_PWM can be left unconnected in case dual-link LVDS is used
The DSI-LVDS Bridge is only capable of a single-link output on channel 0 or a dual-link output on channel 0 and 1. A simultaneous output of the same source on LVDS channel 0 and channel 1 or independent usage on LVDS channel 0 and channel 1 is not supported.

4.7 SPI Bus

The i.MX8M Mini SMARC module offers two Enhanced Configurable SPI (ECSPI) busses with two slave select signals each.

Key features of the ECSPI include:

- Full-duplex synchronous serial interface

- Master/Slave configurable
- Two Chip Select (CS) signals to support multiple peripherals
- Transfer continuation function allows unlimited length data transfers
- 32-bit wide by 64-entry FIFO for both transmit and receive data
- Polarity and phase of the Chip Select (CS) and SPI Clock (SCLK) are configurable
- Direct Memory Access (DMA) support

Table 12: SPI Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
SPI0_DIN	I	1.8V CMOS	A7	ECSPI1_MISO	1.8V		Master Input Slave Output
SPI0_DO	O PP	1.8V CMOS	B7	ECSPI1_MOSI	1.8V		Master Output Slave Input
SPI0_CK	O PP	1.8V CMOS	D6	ECSPI1_SCLK	1.8V		Clock Output
SPI0_CS0#	O PP	1.8V CMOS	B6	ECSPI1_SS0	1.8V	PU 10k 1.8V	Chip-Select 0
SPI0_CS1#	O PP	1.8V CMOS	AD22	SAI2_TXC	1.8V	PU 10k 1.8V	Chip-Select 1 (GPIO4_IO25) *
SPI1_DIN	I	1.8V CMOS	A8	ECSPI2_MISO	1.8V		Master Input Slave Output
SPI1_DO	O PP	1.8V CMOS	B8	ECSPI2_MOSI	1.8V		Master Output Slave Input
SPI1_CK	O PP	1.8V CMOS	E6	ECSPI2_SCLK	1.8V		Clock Output
SPI1_CS0#	O PP	1.8V CMOS	A6	ECSPI2_SS0	1.8V	PU 10k 1.8V	Chip-Select 0
SPI1_CS1#	O PP	1.8V CMOS	AC22	SAI2_TXD0	1.8V	PU 10k 1.8V	Chip-Select 1 (GPIO4_IO26) *

*NOTE: SPI[0:1] are not available if CAN[0:1] interfaces are implemented on the module.

4.8 CAN

The i.MX8M Mini SMARC module optionally features two CAN interfaces based on the Microchip™ MCP2515 stand-alone controller. The controller implements CAN 2.0B protocol specification with up to 1Mbps bit rate. Standard data, extended data and remote frames are supported.

Several features are supported (time-triggered protocols, data byte filtering and one-shot mode).

Table 13: CAN Signal Description *

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
CAN0_TX	O	1.8V CMOS	n.a.	n.a.	1.8V		CAN Transmit output
CAN0_RX	I	1.8V CMOS	n.a.	n.a.	1.8V		CAN Receive input
CAN1_TX	O	1.8V CMOS	n.a.	n.a.	1.8V		CAN Transmit output
CAN1_RX	I	1.8V CMOS	n.a.	n.a.	1.8V		CAN Receive input

*NOTE: CAN0 and CAN1 Controllers are accessible via SPI0 and SPI1. Refer to Chapter [6.2](#) for further information

4.9 GPIO

The CPU GPIO can be used with default Linux GPIO SYSFS interface in user space.

Table 14: GPIO Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
GPIO0 / CAM0_PWR#	I/O	1.8V CMOS	AG14	GPIO1_IO00	1.8V	PU 470k 1.8V	CPU GPIO1_IO00
GPIO1 / CAM1_PWR#	I/O	1.8V CMOS	AF14	GPIO1_IO01	1.8V	PU 470k 1.8V	CPU GPIO1_IO01
GPIO2 / CAM0_RST#	I/O	1.8V CMOS	AF13	GPIO1_IO03	1.8V	PU 470k 1.8V	CPU GPIO1_IO03
GPIO3 / CAM1_RST#	I/O	1.8V CMOS	AF12	GPIO1_IO05	1.8V	PU 470k 1.8V	CPU GPIO1_IO05
GPIO4 / HDA_RST#	I/O	1.8V CMOS	AG11	GPIO1_IO05	1.8V	PU 470K 1.8V	CPU GPIO1_IO06
GPIO5 / PWM_OUT	I/O	1.8V CMOS	AD6	SAI3_MCLK	1.8V	PU 470K 1.8V	CPU GPIO5_IO02 / PWM4_OUT
GPIO6 / TACHIN	I/O	1.8V CMOS	AF6	SAI3_TXD	1.8V	PU 470K 1.8V	CPU GPIO5_IO01
GPIO7	I/O	1.8V CMOS	AB22	SAI2_RXC	1.8V	PU 470K 1.8V	CPU GPIO4_IO22
GPIO8	I/O	1.8V CMOS	AD15	SAI5_MCLK	1.8V	PU 470K 1.8V	CPU GPIO3_IO25
GPIO9	I/O	1.8V CMOS	AG8	SAI3_RXFS	1.8V	PU 470K 1.8V	CPU GPIO4_IO28
GPIO10	I/O	1.8V CMOS	AF10	GPIO1_IO09	1.8V	PU 470K 1.8V	CPU GPIO1_IO09
GPIO11	I/O	1.8V CMOS	AF11	GPIO1_IO07	1.8V	PU 470K 1.8V	CPU GPIO1_IO07

4.10 SDIO

The SDIO interface on the SMARC™ connector supports:

- 1-bit / 4-bit for SD/SDIO mode (standard up to version 3.0)
- 1-bit / 4-bit for MMC mode (standard up to version 5.0)
- SD/SDIO 1.8 V or 3.3 V operation with auto detection
- Default Speed, High-Speed and UHS-I (SDR50, DDR50 and SDR104)

Table 15: SDIO Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
SDIO_D0	I/O	3.3V / 1V8 CMOS	AB23	SD2_DATA0	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Data
SDIO_D1	I/O	3.3V / 1V8 CMOS	AB24	SD2_DATA1	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Data
SDIO_D2	I/O	3.3V / 1V8 CMOS	V24	SD2_DATA2	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Data
SDIO_D3	I/O	3.3V / 1V8 CMOS	V23	SD2_DATA3	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Data
SDIO_CMD	I/O	3.3V / 1V8 CMOS	W24	SD2_CMD	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Command
SDIO_CK	O	3.3V / 1V8 CMOS	W23	SD2_CLK	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Clock
SDIO_PWR_EN	O PP	3.3V CMOS	AB26	SD2_RESET_B	3.3V	PU 10k 3.3V	SDIO Controller Power enable
SDIO_CD#	I	3.3V / 1V8 CMOS	AA26	SD2_CD_B	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Card Detect
SDIO_WP	I	3.3V / 1V8 CMOS	AA27	SD2_WP	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Write Protect



Maximum bus speed for SDIO interface is set per default to High Speed Mode (HS) with a maximum frequency of 50MHz and 3.3V signal voltage.

To achieve higher bus speed mode (SDR50 and SDR104) it is highly recommend to keep SD signal trace length less then 100mm. A load switch with quick output discharge should be used for SD-Card power.

Please contact Avnet Embedded /MSC Technical Support if this feature is required.

4.11 UART

The i.MX8M Mini offers four separate UART interfaces that are linked to the SMARC™ connector, two of them with Hardware flow control support signals.

The UART interfaces supports Serial RS-232NRZ mode, 9-bit RS-485 mode or IrDA mode and includes the following features amongst others:

- High-speed TIA/EIA-232-F compatible, up to 5.0 Mbit/s
- Serial IR interface low-speed, IrDA-compatible (up to 115.2 Kbit/s)
- 9-bit or Multidrop mode (RS-485) support (automatic slave address detection)
- 7 or 8 data bits for RS-232 characters, or 9 bit RS-485 format
- 1 or 2 stop bits
- Programmable parity (even, odd, and no parity)
- Hardware flow control support for request to send (RTS_B) and clear to send (CTS_B) signals
- RS-485 driver direction control via CTS_B signal
- Edge-selectable RTS_B and edge-detect interrupts
- Status flags for various flow control and FIFO states
- Voting logic for improved noise immunity (16x oversampling)
- Transmitter FIFO empty interrupt suppression
- UART internal clocks enable/disable
- Auto baud rate detection (up to 115.2 Kbit/s)
- Receiver and transmitter enable/disable for power saving
- RX_DATA input and TX_DATA output can be inverted respectively in RS-232/RS-485 mode
- Maskable interrupts
- Two DMA Requests (TxFIFO DMA Request and RxFIFO DMA Request)
- Escape character sequence detection
- Software reset (SRST_B)
- Two independent, 32-entry FIFOs for transmit and receive

Table 16: UART Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
SER0_TX	O	1.8V CMOS	F13	UART1_TXD	1.8V	PU 10k 1.8V	UART transmit data (see Note 1 below)
SER0_RX	I	1.8V CMOS	E14	UART1_RXD	1.8V	PU 10k 1.8V	UART receive data (see Note 1 below)
SER0_RTS#	O	1.8V CMOS	AC24	SAI2_TXFS	1.8V	PU 10k 1.8V	UART handshake, ready to receive data
SER0_CTS#	I	1.8V CMOS	AD23	SAI2_RXD0	1.8V	PU 10k 1.8V	UART handshake, ready to send data
SER1_TX	O	1.8V CMOS	D18	UART3_TXD	1.8V	PU 10k 1.8V	UART transmit data
SER1_RX	I	1.8V CMOS	E18	UART3_RXD	1.8V	PU 10k 1.8V	UART receive data
SER2_TX	O	1.8V CMOS	AG6	SAI3_TXC	1.8V	PU 10k 1.8V	UART transmit data
SER2_RX	I	1.8V CMOS	AC6	SAI3_TXFS	1.8V	PU 10k 1.8V	UART receive data
SER2_RTS#	O	1.8V CMOS	AF7	SAI3_RXD	1.8V	PU 10k 1.8V	UART handshake, ready to receive data
SER2_CTS#	I	1.8V CMOS	AG7	SAI3_RXC	1.8V	PU 10k 1.8V	UART handshake, ready to send data
SER3_TX	O	1.8V CMOS	F18	UART4_TXD	1.8V	PU 10k 1.8V	UART transmit data
SER3_RX	I	1.8V CMOS	F19	UART4_RXD	1.8V	PU 10k 1.8V	UART receive data

Note 1: By default, SER0_RX/TX carries the Debug console output of U-Boot and Linux, typically at 115200Bd/8N1. The same serial signals (level-shifted to 3.3 V) also connect to the on-board Serial Debug connector, see also chapter 5.7. So for access to the serial Console of U-Boot / Linux, either one connection can be used as appropriate.

4.12 I²C Bus

I2C_GP on the SMARC™ connector is also linked to an on-module EEPROM at address 0x50.

For further I²C bus signals see also Camera, LCD/LVDS and System Management interfaces.

The I²C bus driven by CPU core function has the following key features:

- Compatible with I²C bus standard
- Multimaster operation
- Software programmability for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven, byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated Start signal generation
- Acknowledge bit generation/detection
- Bus-busy detection
- Data rates up to 100kbits/s in Standard mode and 400kbits/s in Fast mode

Table 17: I²C Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
I2C_GP_CK	O OD	1.8V CMOS	E10	I2C2_SCL	1.8V	PU 2.2k 1.8V	General Purpose SMB clock output
I2C_GP_DAT	I/O OD	1.8V CMOS	F10	I2C2_SDA	1.8V	PU 2.2k 1.8V	General Purpose SMB data I/O line
SMB_ALERT_1V8#	I OD	1.8V CMOS	AF9	SPDIF_TX	1.8V	PU 2.2k 1.8V	Interrupt Signal (GPIO5_IO03)

4.13 Watchdog

Table 18: Watchdog Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
WDT_TIME_OUT#	O PP	1.8V CMOS	AG13	GPIO1_IO02	1.8V		Watch-Dog-Timer Output from the SOC.

4.14 System Management

Table 19: System Management Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
VIN_PWR_BAD#	I OD	1.8V	n.a.	n.a.	5.0V	PU 18K Vin PD 10K	Pulled low at carrier until external power supply is ready. Pulled-up at module by voltage divider. Should be driven by OD part on carrier.
CARRIER_PWR_ON	O PP	1.8V CMOS	n.a.	n.a.	1.8V		Carrier board circuits should not be powered up until module asserts this signal.
CARRIER_STBY#	O PP	1.8V CMOS	n.a.	n.a.	1.8V		Module asserts this signal to indicate standby power state.
RESET_OUT#	O PP	1.8V CMOS	AD19	SAI2_MCLK	1.8V		General purpose reset for carrier board. (CPU GPIO4_IO27)
RESET_IN#	I OD	1.8V CMOS	n.a.	n.a.	1.8V	PU 18k Vin PD 10K	Reset input from Carrier board. Carrier drives low to force a Module reset, floats the line otherwise. Pulled up on module. Should be driven by OD part on carrier.
POWER_BTN#	I	1.8V CMOS	n.a.	n.a.	1.8V	PU 10k 1V8	Power button to bring system into a power state. Pulled up on module. Driven by OD part on carrier. Do not connect any Pull-down or Pull-up resistors on Carrier.

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
CHARGING#	I	1.8V CMOS	AC15	SAI5_RXC	1.8V	PU 10k 1V8	Held low by Carrier during battery charging. Carrier to float the line when charge is complete. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO3_IO20)
CHARGER_PRSENT#	I	1.8V CMOS	AF18	SAI1_RXD5	1.8V	PU 10k 1V8	Held low by Carrier if DC input for battery charger is present. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO4_IO07)
SLEEP#	I OD	1.8V CMOS	AF19	SAI1_RXD7	1.8V	PU 10k 1.8V	Sleep indicator from Carrier board. May be sourced from user Sleep button or Carrier logic. Carrier to float the line in in-active state. Driven by OD part on Carrier. Pulled up on module. (CPU GPIO4_IO09)
LID#	I OD	1.8V CMOS	AB18	SAI1_MCLK	1.8V	PU 10k 1.8V	Lid open/close indication to Module. Low indicates lid closure. Carrier to float the line in in-active state. Active low, level sensitive. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO4_IO20)
BATLOW#	I OD	1.8V CMOS	AB15	SAI5_RXFS	1.8V	PU 10k 1.8V	Battery low indication to Module. Carrier to float the line in in-active state. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO3_IO19)
I2C_PM_CK	O OD	1.8V CMOS	D10	I2C2_SCL	1.8V	PU 2.2k 1.8V	Power management I2C clock output
I2C_PM_DAT	I/O OD	1.8V CMOS	D9	I2C2_SDA	1.8V	PU 2.2k 1.8V	Power management I2C data I/O

4.15 Boot Options

Table 20: Boot Options Control Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M Mini	Pin name on i.MX8M Mini	Power Tolerance	PU/PD	Description
BOOT_SEL0#	I OD	1.8V CMOS	AF23	SAI1_TXD7	1.8V	PU 10k 1.8V	Input straps determine the module boot device. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO4_IO19)
BOOT_SEL1#	I OD	1.8V CMOS	AG23	SAI1_TXD6	1.8V	PU 10k 1.8V	Input straps determine the module boot device. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO4_IO18)
BOOT_SEL2#	I OD	1.8V CMOS	AF22	SAI1_TXD5	1.8V	PU 10k 1.8V	Input straps determine the module boot device. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO4_IO17)
FORCE_RECOV#	I OD	1.8V CMOS	n.a.	n.a.	1.8V	PU 10k 1.8V	Pulled up on Module. Driven by OD part on Carrier.
TEST#	I OD	1.8V CMOS	n.a.	n.a.	1.8V	PU 10k 1.8V	Active low signal for test mode activation. Pulled up on Module. Driven by OD part on Carrier.

If the SMARC module is powered up with VIN_PWR_BAD# left floating and RESET_IN# left floating, the module boots from selected boot device. The boot process consists of two parts:

- Primary boot Loader (includes Uboot)
- Operating System boot

The boot device for the primary boot loader is selected using the TEST# signal:

- If TEST# is pulled low the module uses carrier SD-Card as the primary boot device.
- If TEST# is left floating the module loads the boot loader from eMMC.

If the primary Boot device fails, module will always try to load the primary boot loader from the Carrier SD card. This is unintended behavior but cannot be changed as it is configured by the ROM code in the CPU.

Once the Uboot starts it will check the state of the BOOT_SELx# signals to decide where to boot the Operating System from. (See table below)

If FORCE_RECOV# signal is pulled low at carrier, the module boots via USB-Client Mode (this feature is only intended for recovery and requires dedicated software from NXP). For normal operation do not pull FORCE_RECOV# signal low.

Table 21: Boot Options

	BOOT_SEL2#	BOOT_SEL1#	BOOT_SEL0#	Boot Source
0	GND	GND	GND	Carrier SATA (not supported)
1	GND	GND	Float	Carrier SD Card
2	GND	Float	GND	Carrier SPI1 with CS0# (not sensible for ARM devices, not planned to be supported)
3	GND	Float	Float	Carrier SPI0 with CS0# (not sensible for ARM devices, not planned to be supported)
4	Float	GND	GND	Module SD Card
5	Float	GND	Float	Remote boot
6	Float	Float	GND	Module eMMC Flash
7	Float	Float	Float	USB Mass Storage

5 Functions on Module

5.1 CPU Options

The module can be ordered with several i.MX8M Mini CPU types. Detailed information is provided in the module datasheet which can be downloaded from <https://www.msc-technologies.eu/support/boards/smarc/msc-sm2s-imx8mini.html>.

For details regarding the i.MX8M Mini CPU please refer to the NXP™ website. For order information please contact Avnet Embedded /MSC.

5.2 Power-Up Behaviour

The module will behave in the following ways:

- When coming from complete power off (5V unpowered), the module will boot if VIN_PWR_BAD# or TEST# is not low and 5V is present.
- When OS is shut down and 5V is still powered, a power button press is required to restart the module.
- If the module does not come up in test mode or force recovery mode it fetches the OS and the file system from the boot source, defined by the BOOT_SEL strapping pins.
- On keeping the power button pressed for 8 seconds or longer, the module will shut down, and restart as soon as the power button is pressed again.

5.2.1 Power-On Sequencing

Figure 5-1: Start-up Sequence

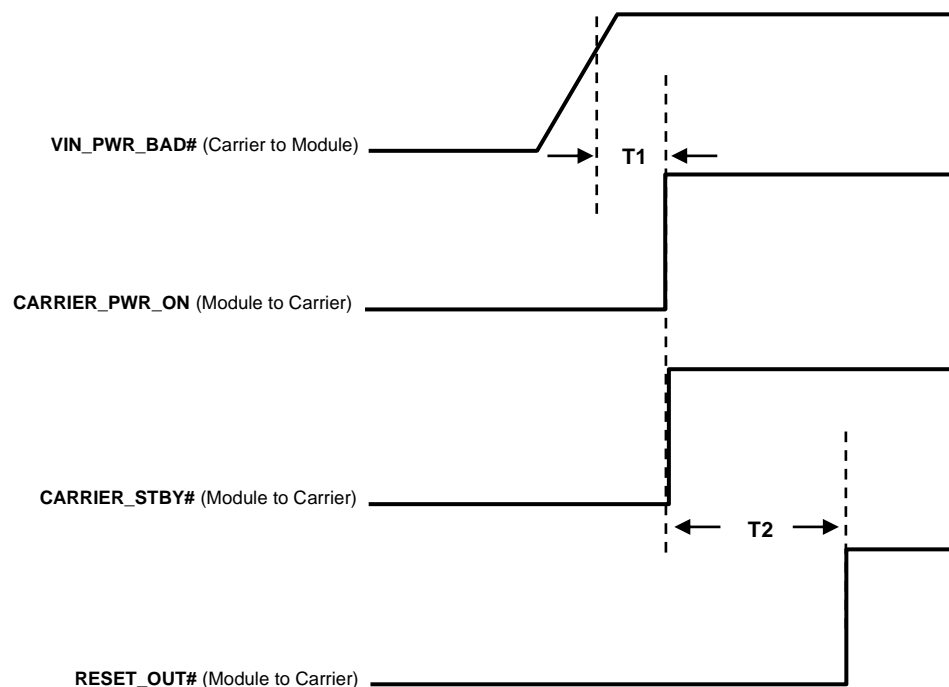


Figure 5-2: Power-On Timings

Time Slot	Description	Value
T1	CARRIER_PWR_ON to VIN_PWR_BAD# timing	10.8ms
T2	RESET_OUT# to CARRIER_PWR_ON	355ms

5.2.2 Reset Sequencing

IMX8M Mini can't be reset by internal reset and a power cycle is preferred. Some peripherals require a power cycle during system reset.

RESET_IN# assertion leads to a module power cycle as shown in the following figure:

Figure 5-3: Reset Sequencing

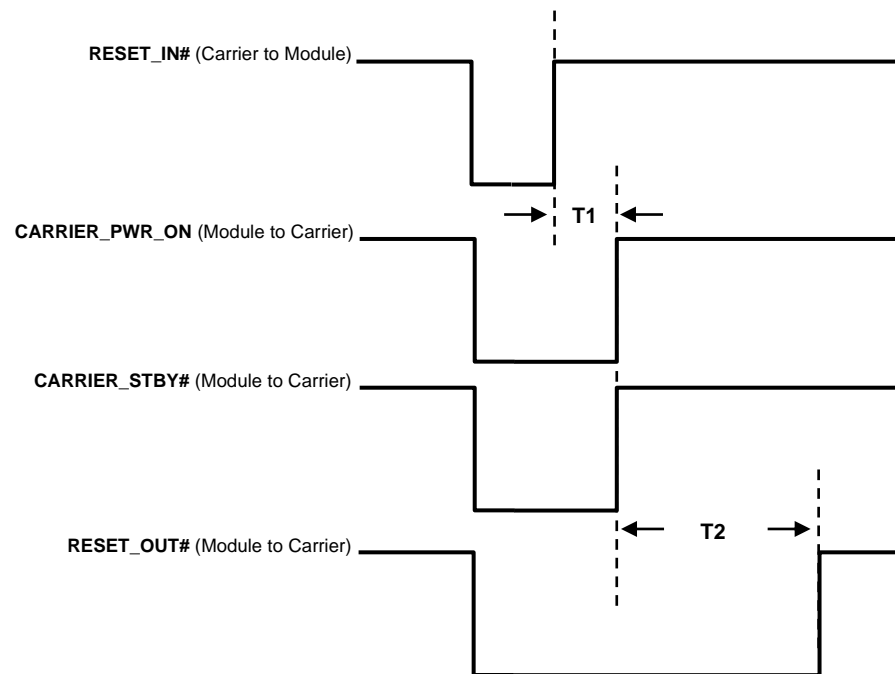


Figure 5-4: Reset Timings

Time Slot	Description	Value
T1	CARRIER_PWR_ON to RESET_IN# timing	10.8ms
T2	RESET_OUT# to CARRIER_PWR_ON	355ms

5.3 Memory

5.3.1 SDRAM

The DDR Controller supports 32/16-bit LPDDR4-3000.

MSC SM2S-IMX8MINI SMARC™ modules use one physical rank with up to 4GByte SDRAM.

Table 22: Available SDRAM options

CPU	Bus Width	Memory Size	Memory Organisation
i.MX8M Mini Solo, Dual and Quad	64-bit Interface	1 GB	2x 32Mx16x8B
	64-bit Interface	2 GB	4x 32Mx16x8B
	64-bit Interface	4 GB	4x 64Mx16x8B

5.3.2 eMMC

Up to 64GB eMMC are supported. The eMMC is used in 8 bit mode.

Table 23: Available eMMC devices

Memory Size	Technology	Operating Temperature	Chip Identification
Extended			
8 GB	15nm X2 eMLC	-25°C to +85°C	SDINBDG4-8G-I1
16 GB	15nm X2 eMLC	-25°C to +85°C	SDINBDG4-16G-I1
32 GB	15nm X2 eMLC	-25°C to +85°C	SDINBDG4-32G-I1
64 GB	15nm X2 eMLC	-25°C to +85°C	SDINBDG4-64G-I1
Industrial			

Memory Size	Technology	Operating Temperature	Chip Identification
8 GB	15nm X2 eMLC	-40°C to +85°C	SDINBDG4-8G-XI1
16 GB	15nm X2 eMLC	-40°C to +85°C	SDINBDG4-16G-XI1
32 GB	15nm X2 eMLC	-40°C to +85°C	SDINBDG4-32G-XI1
64 GB	15nm X2 eMLC	-40°C to +85°C	SDINBDG4-64G-XI1

5.3.3 EEPROM

64Kb EEPROM for board data connected to I2C3 bus at address 0x50.

The EEPROM on address 0x50 of the I2C_GP Bus holds the Board Information (boardinfo) structure, which is evaluated by U-Boot to determine the exact board variant and set necessary parameters.

Make sure to leave this EEPROM in place, and DO NOT block address 0x50 on the I2C_GP Bus with other devices, otherwise the module will be unable to boot!

The Board information structure occupies the first 0x80 bytes inside the EEPROM. The remaining upper range starting at offset 0x80 is freely available for customer purposes. When making use of this option, make sure to keep the lower 0x80 bytes intact, otherwise the board will also not boot any more.

5.4 Trusted Platform Module

The i.MX8M Mini SMARC™ module offers an optional Trusted Platform Module 2.0 (TPM): Infineon™ SLB9673 or STMicroelectronics™ ST33TPHF20I2C.

The TPM is connected to the I2C1 bus.

5.5 WiFi/Bluetooth

The i.MX8M Mini SMARC™ module offers an optional WiFi/Bluetooth module the SPB209A from H&D Wireless™. The module is connected via SDHC3 interface* and is equipped with an RF micro coaxial connector (U.FL receptacle).

Key features:

- IEEE 802.11ac compliant
- IEEE 802.11 PHY data rates of up to 433 Mbps
- Bluetooth 4.2
- Supports multiple SW features such as 802.11e/i (Security, QoS), Soft AP and Wi-Fi Direct
- Extensive DMA hardware support for data flow to reduce CPU load

*NOTE: microSD card socket and WiFi/BT module are mutually exclusive assembly options.

5.6 microSD Card Socket

An optional on-module microSD card socket is connected via SDHC3 interface. The interface only supports High Speed (HS) Mode at 50MHz frequency and data rates up to 25MB/s (3.3V operation only).

5.7 Debug Options

5.7.1 Debug Connector

Access to the Debug UART port is possible via an 8pin FFC connector.

Note: This connector may not be populated on all board variants. Contact Avnet Embedded /MSC Technical Support if this feature is required.

Figure 5-5: Module top side with debug UART FFC connectors marked in red*

Pinout:

- 1: Ground
- 2: NC
- 3: NC
- 4: NC
- 5: UART_TXD
- 6: UART_RXD
- 7: RESET_IN#
- 8: VCC_3V3

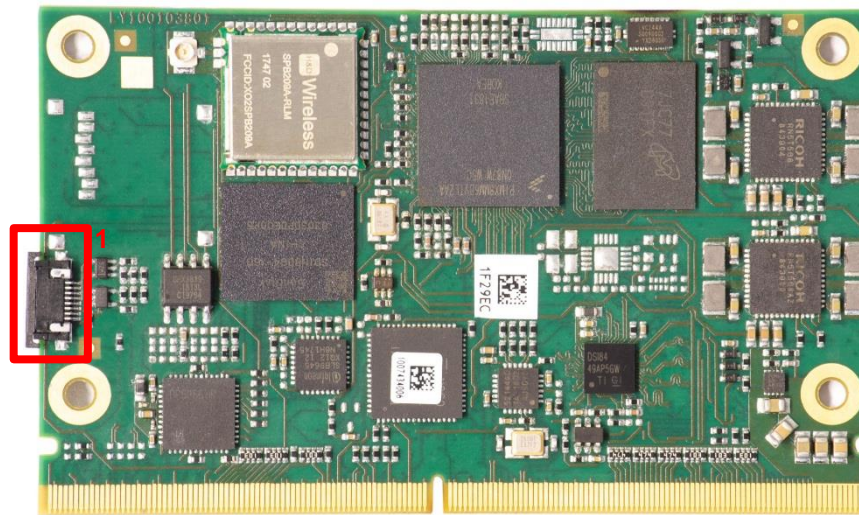
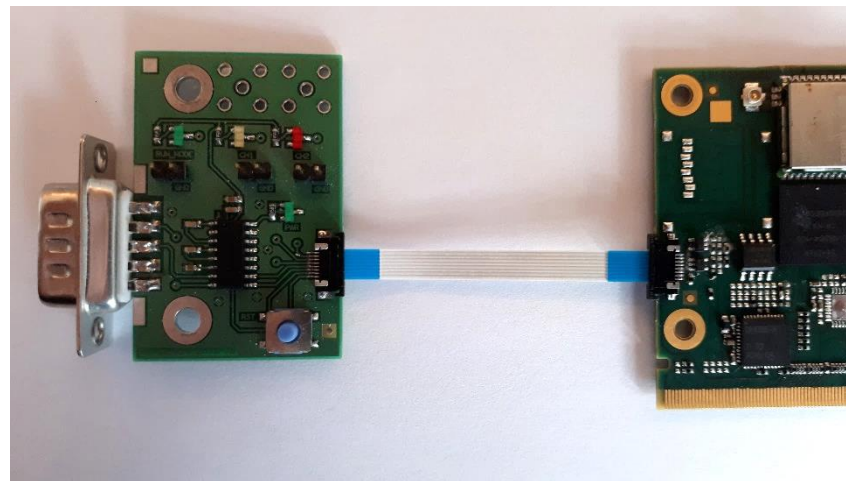


Figure 5-6: Module top side with MSC UART debug adapter



MSC Debug-Adaptor
82479

Debug UART Adapter for i.MX8-based SMARC, Qseven and nanoRISC modules, with 8-pin FFC cable to connect COM module to 9-pin D-Sub connector

Use top / top cables.

Serial signals on the Debug connector are at 3.3V TTL level.

For using this Debug connector, customers can obtain a small size debug board (including FCC cable) as an accessory with Order No. 82479. This board converts the Debug UART signals to RS-232 level and offers them on a standard DSUB9-M connector.

Additionally, this debug board has a soft-reset button and three LEDs on GPIOs for additional debug capabilities.

Serial Debug Console Output options

The Debug connector offers the same SER0_RX/TX signals which are also duplicated on the SMARC connector, pins P129/130 (there with 1.8V level, while on Debug connector with 3.3V TTL level). See also section 4.11 .

So for accessing the U-Boot / Linux console, depending on availability and usage of the port on the target Carrier board, either connection via the Carrier board + SMARC connector or via the Debug connector on the module can be chosen to the same effect.

5.7.2 JTAG Connector

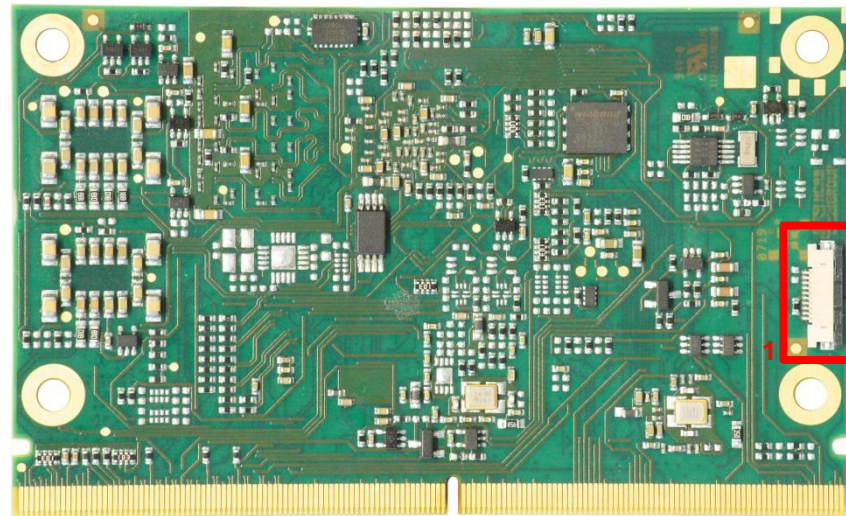
JTAG access to the IMX8M Mini CPU is possible via a 10pin FFC connector. The JTAG Chain only contains the CPU itself, so all suitable JTAG debuggers should work with their default configuration for the respective CPU.

The JTAG connector is not populated by default. Please contact Avnet Embedded /MSC Technical Support if this feature is required.

NOTE: JTAG_MODE has an on-module 10k pull down. If JTAG Mode is left open or pulled low, the CPU is in debug-JTAG mode

(JTAG Interface is connected to the CPU core for software debug). If pulled up, JTAG is connected to the boundary-scan chain of the CPU.

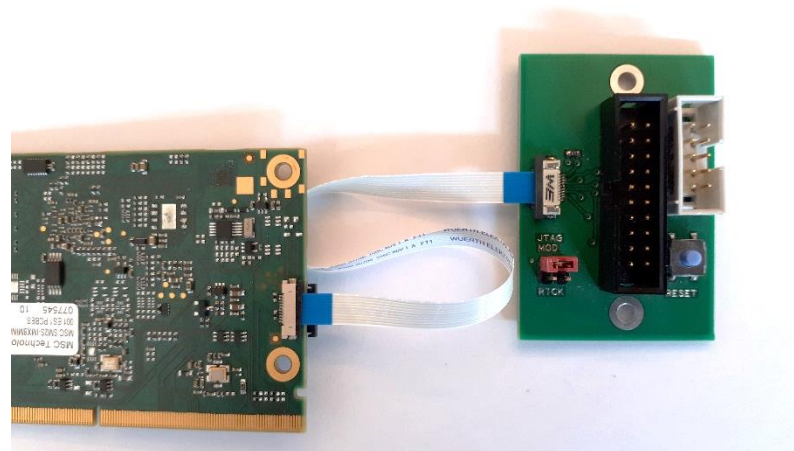
Figure 5-7: Module bottom side with JTAG FFC connectors marked in red*



Pinout:

10: VCC_1V8
9: JTAG_TRST#
8: JTAG_TDI
7: JTAG_TDO
6: JTAG_TMS
5: JTAG_TCK
4: JTAG_MOD
3: NC
2: RESET_IN#
1: Ground

Figure 5-8: Module bottom side with MSC JTAG debug adapter



MSC JTAG-Adaptor FFC 10pol
68948

Debug JTAG Adapter for i.MX8-based SMARC modules, with 10-pin FFC cable to connect COM module to connectors for JTAG connection to Lauterbach and/or Goepel debuggers

Use top / top cables.

6 Bus and Address Mapping

6.1 I²C Devices

Table 24: I²C Interfaces Overview

CPU Interface	Device	SMARC Connector	7-bit Address
I2C1	TPM		0x2E
	PMIC1		0x31
	PMIC2		0x33
	Temp. Sensor		0x71
	RTC		0x32
	PCIe Clock Generator		0x6B
I2C2		I2C_PM	
I2C3	Serial EEPROM	I2C_GP	0x50
I2C4		I2C_LCD	
GPIO		I2C_CAM0* I2C_CAM1*	

*NOTE: CAM0 and CAM1 are GPIO based (bit-banged) and share the same I2C interface.

6.2 SPI Devices

Table 25: SPI Interfaces Overview

CPU Interface	Chip Select	CPU Pin	CPU Pin Name	Device	SMARC Connector	Description
ECSPI1	CS0	B6	ECSPI1_SS0		SPI0_CS0#	
	CS1	AD22	SAI2_TXC		SPI0_CS1#	GPIO based chip select (CPU GPIO4_IO25)*
	CS1	AD22	SAI2_TXC	CAN0 Controller		Dedicated chip select for CAN0 Controller*
ECSPI2	CS0	A6	ECSPI1_SS0		SPI1_CS0#	
	CS1	AC22	SAI2_TXD0		SPI1_CS1#	GPIO based chip select (CPU GPIO4_IO26)*
	CS1	AC22	SAI2_TXD0	CAN1 Controller		Dedicated chip select for CAN1 Controller*
QSPI_A	CS0	N24	NAND_CE0_B	QSPI NOR Flash		Dedicated chip select for QSPI NOR Flash*

*NOTE: SPI[0:1]_CS1# are not available if the CAN interfaces are implemented on the module.
The on-module QSPI NOR Flash is an assembly option.

7 Board Support Package (BSP)

7.1 General information

MSC-LDK and the underlying NXP release are based on the Yocto build system (<https://yoctoproject.org>).

The current MSC-LDK and the msc-sm2s-imx8mm BSP base on NXP's Release L4.19.35_1.1.0.

7.2 MSC-LDK (Yocto)

7.2.1 MSC-LDK Terms

- The Yocto-based MSC-LDK uses a sophisticated approach to generate Linux images.
- A target is the hardware or CPU module on which the generated Linux software is to be run.
- An image contains all the files necessary for execution by the targeted hardware, e.g. the Linux kernel and the root filesystem.
- Software that is part of a Linux image is called a package.
- A package is generated from sources by a recipe, which is a description of where to download the sources and how to compile them within Yocto.
- A layer is a collection of recipes. They are stackable and can extend or modify recipes defined in other layers. A BSP provides the necessary layers to MSC-LDK to support the target's hardware.
- MSC-LDK is mainly an installer of Yocto, MSC specific layers and BSP layers.

7.2.2 Getting Started

7.2.2.1 System requirements

- 64bit Linux x86 development host with at least 8GB of RAM - 16GB or more recommended
- Ubuntu 16.04 (LTS) or newer, other distributions may also work.
- Internet access for downloading packages (HTTP, FTP, Git and SSH).
- Lots of free disk space for the initial build (>128 GB)
- Python3 with 'pip' installed (at least Python v3.3).

7.2.2.2 Registration on the MSC Git Server

Downloading files from the MSC Git server requires a registration on:

<http://www.msc-technologies.eu/register.html>.

Registered users may apply for specific Git repositories by sending an email with their public SSH key and desired project name to:

support.boards@avnet.eu

7.2.2.3 Creating SSH key

If there is no SSH key already available (/.ssh/id_rsa.pub), it can be generated with following command :

```
$ ssh-keygen -t rsa
```

Example:

```
waldemar@Workstation3~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/waldemar/.ssh/id_rsa):
Created directory '/home/waldemar/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/waldemar/.ssh/id_rsa.
Your public key has been saved in /home/waldemar/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:lUnkxE/cUg7ZmCv2zI9HpVq1YhNhc3ZrgFMqbo0mGRI waldemar@Workstation3
The key's randomart image is:
+---[RSA 2048]---+
|                 o+.+0. |
|                +.=0Boo. |
|   E            .+=+0oB o |
|                . o.+ o. oo |
|                . . oS= = o+. |
|                . = . . ++++ |
|                o      .*o   |
|                 o o       |
|                .         |
+-----[SHA256]-----+
waldemar@Workstation3~$
```

Figure 7-1. RSA key generation

Share the public key in /.ssh/id_rsa.pub with MSC during Git registration.



Make sure to keep the Private Key "id_rsa" well secured. It is generally possible to share one keypair within one and the same project, to allow several people access to the MSC sources. But then the validated user is responsible to keep track of the Private Key any time, and **MUST** inform MSC Support if the key has been compromised and access must be withdrawn.



The SSH key **must not** have a passphrase. It will be used in background communication and therefore there is no possibility to enter the passphrase. Trying to fetch repositories from the MSC Public GIT Server would fail with no hint that the passphrase is missing.

7.2.2.4 Configuring HTTP proxy

Some source files will be downloaded from HTTP and FTP servers. If a proxy must be used, these environment variables have to be set:

```
export http_proxy=http://my-proxy:3128
export https_proxy=http://my-proxy:3128
export ftp_proxy=http://my-proxy:3128
```

Note: Replace "my-proxy" with the appropriate address of your network's proxy, Port number "3128" may also vary depending on your network. Please consult your admin for proper settings.

On some networks, tunneling Git SSH access over HTTPS may be additionally necessary. Please contact Avnet Embedded /MSC Technical Support for a related App Note.

7.2.3 Setup the MSC-LDK build environment



The MSC-LDK must be installed on a partition with at least 128 GB free space. As a lot of source files will be accessed, it is recommended to use an EXT4 partition with the mount options "noatime,nodiratime".

7.2.3.1 The “classic” method

7.2.3.1.1 Step 1: Clone the base MSC-LDK repo

To clone and enter the base MSC-LDK repo, run the following command:

```
git clone ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/msc-ldk --branch v1.5.0 msc-ldk-v1.5.0
cd msc-ldk-v1.5.0
```

Example:

```
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk$git clone ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/msc-ldk --branch v1.5.0 msc-ldk-v1.5.0
Cloning into 'msc-ldk-v1.5.0'...
remote: Counting objects: 5473, done.
remote: Compressing objects: 100% (2027/2027), done.
remote: Total 5473 (delta 3709), reused 4834 (delta 3283)
Receiving objects: 100% (5473/5473), 691.87 KiB | 7.60 MiB/s, done.
Resolving deltas: 100% (3709/3709), done.
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk$cd msc-ldk-v1.5.0/
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0 (v1.5.0)$
```

Figure 7-2. Clone base MSC-LDK repo

Your current directory now contains the following sub directories:

```
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0 (v1.5.0)$tree -d -L 1
.
├── scripts
└── template

2 directories
```

Figure 7-3. Initial content of the root MSC-LDK directory.

7.2.3.1.2 Step 2: Create build directory

To create your build directory run the following command:

```
./setup.sh --bsp=0103801 --checkout-layers
```



The 0103801 refers to the MSC LDK for the SM2S-i.MX8MMini module.

Example:

```
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0 (v1.5.0)$ ./setup.sh --bsp=0103801 --checkout-layers
Switching libMscBoostPython to 'v1.1.2'
Executing 'git checkout master'
Executing 'git checkout v1.1.2'
NOTICE: libMscBoostPython is at <Branch: master, TAG: v1.1.2 [Detached HEAD]>
NOTICE: MSC-LDK git server: 'ssh://gitolite@msc-git02.msc-ge.com:9418/'
NOTICE: MSC-LDK root: /mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0
NOTICE: MSC-LDK is based on Yocto branch: warrior, MSC-LDK is at <Branch: v1.5.0 [LOL99_20191211_V1_5_0-2-g5dcf4d6]>
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0103801/msc-ldk-bsp-recipes'
NOTICE: MSC-LDK Configuration: BSP=0103801
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/yocto'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/meta-openembedded'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/meta-gt5'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/meta-secure-core'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0199/meta-msc-ldk-core-recipes'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0199/meta-msc-ldk-core'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0199/meta-msc-ldk-mscio'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/meta-freescale'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/meta-freescale-distro'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/meta-fsl-bsp-release'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/meta-browser'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/thirdparty/meta-gt5-extra'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0199/meta-msc-arm-extensions'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0199/meta-msc-ldk-marvell'
NOTICE: Created '/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0/build/0103801/conf/local.conf'
NOTICE: Created '/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0/build/0103801/conf/bblayers.conf'
INFO: You can now cd to /mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0/build/0103801 and run 'make' or './build.sh <image-name>'
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0 (v1.5.0)$
```

Figure 7-4. Create build directory.

Your current directory now contains the following sub directories:

```
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0 (v1.5.0)$ tree -d -L 1
.
├── build
├── scripts
├── sources
└── template

4 directories
```

Figure 7-5. Base directory content after setup build directory.

7.2.3.1.3 Step 3: Enter build directory

To enter the build directory execute:

```
cd build/0103801
```

Example:

```
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0/build/0103801 (v1.5.0)$tree -d -L 1
.
└─ conf

1 directory
```

Figure 7-6. Enter build directory.

7.2.3.2 The “docker” method

We assume that the docker packages (docker, docker.io, etc.) are already installed on your development system and your local user is a member of the docker group. Using "docker" may especially be helpful under newer versions of the host's OS, such as Ubuntu 18.04 LTS or higher.

For detailed information about docker installation, container handling and development under docker please take a look at [9].

7.2.3.2.1 Step 1: Create MSC-LDK container

Execute:

```
git clone ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/docker-msc-ldk
cd docker-msc-ldk
git checkout v1.3.0
mkdir src
mkdir -p rootfs/home/.ssh
cp ~/.ssh/id_rsa rootfs/home/.ssh
cp ~/.ssh/id_rsa.pub rootfs/home/.ssh
docker build -t=msc-ldk .
rm -rf rootfs/home/.ssh
```

Example:

```
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp$git clone ssh://gitolite@msc-git02.msc-ge.co
m:9418/msc_0199/docker-msc-ldk
Cloning into 'docker-msc-ldk'...
remote: Counting objects: 50, done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 50 (delta 22), reused 0 (delta 0)
Receiving objects: 100% (50/50), 7.58 KiB | 3.79 MiB/s, done.
Resolving deltas: 100% (22/22), done.
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp$cd docker-msc-ldk
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk (master)$git checkout v1.3.0
Note: checking out 'v1.3.0'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

    git checkout -b <new-branch-name>

HEAD is now at d93d2c9 removed default proxy configuration (MLDK-428)
```

Figure 7-7. Prepare docker container for MSC-LDK. Part 1.

```
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$mkdir src
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$mkdir -p rootfs/h
me/.ssh
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$cp ~/.ssh/id_rsa r
ootfs/home/.ssh
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$cp ~/.ssh/id_rsa.p
ub rootfs/home/.ssh
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$docker build -t=ms
c-ldk .
Sending build context to Docker daemon 76.8kB
Step 1/29 : FROM ubuntu:16.04
--> 657d80a6401d
Step 2/29 : MAINTAINER mpie
--> Using cache
--> 296dd372d9ea
Step 3/29 : ARG UID=1000
--> Using cache
--> e8c7fa8ac2af
Step 4/29 : ARG GID=1000
```

Figure 7-8. Prepare docker container for MSC-LDK. Part 2.


```
Step 26/29 : RUN mkdir -p /src /src/msc-ldk-downloads /src/msc-ldk-ssstate-cache && chown ${UID}:${GID} /src -R &&
chown ${UID}:${GID} ${HOME} -R && true
----> Using cache
----> cdf0e59b2dcc
Step 27/29 : USER ${UID}
----> Using cache
----> ebf5c3e99cf1
Step 28/29 : RUN echo "if [ -e /src/.bashrc ]; then source /src/.bashrc; fi" >> ${HOME}/.bashrc
----> Using cache
----> acd86641a26a
Step 29/29 : WORKDIR /src
----> Using cache
----> 91a6a285b92c
Successfully built 91a6a285b92c
Successfully tagged msc-ldk:latest
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$rm -rf rootfs/home
/.ssh
```

Figure 7-9. Prepare docker container for MSC-LDK. Part 3.

7.2.3.2.2 Step 2: Start and enter the MSC-LDK container for the first time

Execute:

```
docker run --privileged -t -i --dns $(nmcli -f 'IP4.DNS' \
-m multiline device show 2>&1 | sed -rn 's/IP4.DNS\[1\]: *(.*)/\1/p') \
--name msc-ldk -h docker -v `pwd`/src:/src msc-ldk /bin/bash
```

Example:

```
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$docker run --privi
leged -t -i --dns $(nmcli -f 'IP4.DNS' \
-m multiline device show 2>&1 | sed -rn 's/IP4.DNS\[1\]: *(.*)/\1/p') --name msc-ldk -h docker -v `pwd`/src:/src msc-ldk
/bin/bash
```

Figure 7-10. Start and enter the MSC-LDK container.

7.2.3.2.3 Step 3: Clone and enter the base MSC-LDK repo

See 7.2.3.1.1

7.2.3.2.4 Step 4: Create build directory

See 7.2.3.1.2

7.2.3.2.5 Step 5: Enter build directory

See 7.2.3.1.3

7.2.3.2.6 Leave the MSC-LDK container.

Execute:

```
exit
```

Example:

```
user@docker:/src$
user@docker:/src$ exit
exit
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$
```

Figure 7-11. Leave the MSC-LDK container.

7.2.3.2.7 Re-start and re-enter the MSC-LDK container.

Execute:

```
docker container start msc-ldk
```

```
docker container exec -ti msc-ldk /bin/bash
```

Example:

```
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$docker container s
tart msc-ldk
msc-ldk
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$docker container e
xec -ti msc-ldk /bin/bash
user@docker:/src$
```

Figure 7-12. Re-start and re-enter the MSC-LDK container.

7.2.3.2.8 Stop the MSC-LDK container and release its resources.

Execute:

```
docker stop msc-ldk
```

```
docker rm msc-ldk
```

Example:

```
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$docker stop msc-ldk
msc-ldk
waldemar@waldemar-HP-Z6-G4-Workstation/mnt/raid1/Develop/repos/msc-ldk/temp/docker-msc-ldk ((v1.3.0))$docker rm msc-ldk
msc-ldk
```

Figure 7-13. Stop the MSC-LDK container and release its resources.

7.2.4 Generate images

7.2.4.1 Choosing an MSC-LDK image

The MSC-LDK provides different images for the sms2-imx8mmi module. The following table lists all currently available images, their contents and sizes.

Table 26: Available images

dtb	approx size	comment
msc-image-base	612MiB	A small Wayland GUI image with full target device hardware support. Contains MSC features/tools.
msc-image-qt5	2,3GiB	A Wayland GUI image with full target device hardware support. Contains Qt5/Qml, gstreamer, opencv, and MSC features/tools.

7.2.4.2 Building an Image with MSC-LDK

The MSC-LDK build uses the:

```
./build.sh bitbake <component>
```

command, where <component> can be:

- image name (e.g. msc-image-qt5, etc.)
- package name (e.g. u-boot-imx, linux-imx, memtester, etc.)

Example:

```
waldemar@Workstation1/mnt/data2/Develop/repos/msc/msc-ldk/msc-ldk-v1.5.0/build/0103801 (v1.5.0)$ ./build.sh bitbake msc-i
mage-gt5
WARNING: You have included the meta-gnome layer, but 'x11' has not been enabled in your DISTRO_FEATURES. Some bbappend f
iles may not take effect. See the meta-gnome README for details on enabling meta-gnome support.
WARNING: Host distribution "linuxmint-19.1" has not been validated with this version of the build system; you may possib
ly experience unexpected failures. It is recommended that you use a tested distribution.
Loading cache: 100% |#####| Time: 0:00:00
Loaded 4177 entries from dependency cache.
: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION      = "1.42.0"
BUILD_SYS       = "x86_64-linux"
NATIVELSBSTRING = "universal"
TARGET_SYS      = "aarch64-poky-linux"
MACHINE         = "am2s-imx8mm-gc"
DISTRO          = "poky"
DISTRO_VERSION  = "2.7.1"
TUNE_FEATURES   = "aarch64"
TARGET_FPU      = ""
meta-poky
meta-yocto-bsp   = "warrior-4.19.35-1.1.0-msc:4175f1771245d5c0208994c9b0dcaa96dab30541"
meta            = "master:023a8dcd46bd2d038b01ff42afe97e926e430c4"
meta            = "warrior-4.19.35-1.1.0-msc:4175f1771245d5c0208994c9b0dcaa96dab30541"
meta-oe
meta-networking
meta-python      = "warrior-msc-v1.5.0:f1b18c0da7f247998c2269069d7b47ed1e62bf12"
meta-gt5.git     = "warrior-msc-v1.5.0:8586caf2f2f5c11fb2ddff29eefe5dd7fc08cf38"
meta
meta-integrity
meta-signing-key
meta-tpm
meta-tpm2        = "warrior-v1.5.0:0fbc3c26c5836b5a040d8799ffa70d95493930e3"
meta-msc-ldk-core-recipes.git = "warrior:dc2653c37d8e97da5fcd24b69ed289b5bbba281"
meta-msc-ldk-core.git = "warrior:d24dc70d92eb3525108f64f768752a651d61e3a"
meta-msc-ldk-macio.git = "warrior:e8418741adcefd5b442e338cfe366ec40d4fa77"
meta-freescale.git = "warrior-4.19.35-1.1.0-msc:c6fc58555ac9b2270944b5759c98f8188022864"
meta-freescale-distro.git = "warrior-4.19.35-1.1.0-msc:67465321a8038282eebab5cd40e57f204b6203be"
meta-bsp
meta-sdk         = "warrior-4.19.35-1.1.0-msc:301bcd8d9d48056366c58ae3fec2fc800f2bd194"
meta-browser.git = "warrior-4.19.35-1.1.0:5f365ef0f842ba4651efe88787cf9c63bc8b6cb3"
meta-gt5-extra.git = "warrior-msc-v1.5.0:2b32b3c0318c45b6a69614dde8e98968ae4d6e75"
meta-msc-arm-extensions.git = "warrior:7db7a75f1d03a1712505fadb476ae8d07b97ca5"
meta-multimedia
meta-gnome       = "warrior-msc-v1.5.0:f1b18c0da7f247998c2269069d7b47ed1e62bf12"
meta-msc-ldk-marvell.git = "v1.5.0:ed1c4212e979883aba7ba29440cf0f782182eb46"

Initialising tasks: 100% |#####| Time: 0:00:03
State summary: Wanted 0 Found 0 Missed 0 Current 2950 (0% match, 100% complete)
: Executing SetScene Tasks
: Executing RunQueue Tasks
: Tasks Summary: Attempted 7925 tasks of which 7925 didn't need to be rerun and all succeeded.
: No commit since BUILDHISTORY_COMMIT != '1'

Summary: There were 2 WARNING messages shown.
*** OK ***
```

Figure 7-14. Building msc-image-gt5 image.

For more details and further information see also [5], Chapter 5 “Building an image”

Some scripts of the recipes use an 'echo -e <somewhat>' command. bitbake calls the buildscripts with /bin/sh as shell. If your hosts system uses "bash" as "/bin/sh" everything works fine. But if a shell with less functionality like "dash" is used, it is necessary to setup "bash" as sh. This can be done on most debian derivated systems by:



```
user@devhost:$ sudo dpkg-reconfigure dash
```

The subsequent question has to be answered with "no"

Depending on the internet connection and the development host a first build may take several hours. To speed it up on further installations, share the directories downloads and sstate-cache. All generated images can be collected in a specific directory with:

```
user@devhost:msc-ldk$ make install_images DESTDIR=/tmp/msc-ldk-images
```

7.2.4.3 Reproduce Images with MSC-LDK

One of the key features of Yocto is the strong versioning of the resulting images. Each package uses a predefined version, e.g. busybox 1.32.0. When compiling an image, yocto also prints the used GIT layer versions (see **Fehler! Verweisquelle konnte nicht gefunden werden.**)

For further improvement, MSC-LDK has these additional features to recreate the image **after** it has been built and shipped:

- The used layers and the setup line how the BSP was configured is stored in the image's file /etc/version_layer. After compilation, the file can be also found in the build directory under:

```
tmp/work/sm2s_imx8mm_qc-poky-linux/msc-image-base/1.0-r0/rootfs/etc/
```

```
root@sm2s-imx8mm-qc:~#
root@sm2s-imx8mm-qc:~# cat /etc/version_layer
MSC-LDK LOL99_20191211_V1_5_0-2-g5dcf4d6 built on Mon Jul 13 10:15:26 UTC 2020 by waldemar@waldemar-HP-Z6-G4-Workstation
Poky (Yocto Project Reference Distro) 2.7.1 \n \l

LAYER meta-browser=LOL99_20190718_V1_4_0-72-g5f365ef
LAYER meta-freescale-distro=2.2-48-g6746532
LAYER meta-freescale=LOL99_20170627_V1_2_0-1231-gc6fc5855
LAYER meta-fsl-bsp-release=301bcd8d9
LAYER meta-msc-arm-extensions=LOL99_20191211_V1_5_0-4-g7db7a75
LAYER meta-msc-ldk-core=
LAYER meta-msc-ldk-core=LOL99_20191211_V1_5_0-8-gd24dc70
LAYER meta-msc-ldk-core-recipes=LOL99_20191211_V1_5_0-13-gdccc2653
LAYER meta-msc-ldk-marvell=LOL99_20191211_V1_5_0-1-ged1c421
LAYER meta-msc-ldk-mscio=LOL99_20191211_V1_5_0-6-ge841874
LAYER meta-openembedded=warrior-msc-LOL99_20191211_V1_5_0
LAYER meta-qt5-extra=warrior-msc-LOL99_20191211_V1_5_0
LAYER meta-qt5=warrior-msc-LOL99_20191211_V1_5_0
LAYER meta-secure-core=warrior-LOL99_20191211_V1_5_0
LAYER msc-ldk-bsp-recipes=LOL99_20190718_V1_4_0-78-g023a8dc
LAYER yocto=warrior-21.0.1-27-g4175f17712
SETUP --bsp=0103801 --checkout-layers
```

Figure 7-15. Content of 'version_layer' file.

- The setup tool allows to checkout exactly these layers and configure the BSP as before. To use it, call setup.py with only one argument --version-file, e.g.

```
./setup.py --version-file ~/version_layer
```



- Modifications of conf/local.conf are not traced.
- This will checkout exactly the versions used by version_layer. It is then no longer possible to use scripts/update.py to pull the latest changes on the branch. A fresh checkout of MSC-LDK is necessary. The directories downloads and sstate-cache can be moved or copied to improve build speed.
- Time stamps in the image will be updated, e.g. in /etc/issue.

7.2.5 Image Deployment

See [5], Chapter 6, “Image Deployment”

7.2.5.1 Flashing an SD card image

To flash an SD card image, run the following command:

```
sudo dd if=<image name>.sdcard of=/dev/sd<partition> bs=4MiB conv=fsync
```

7.3 Running an Image

7.3.1 Booting SPL (secondary program loader)/U-Boot

The TEST# pin is used to select one of the following boot schema.

7.3.1.1 TEST# = LOW: Forced booting SPL/U-Boot from carrier SD card

With TEST# = LOW (e.g. on MSC SM2-MB-EP1 Carrier Board: "Test Mode Select" DIP Switch = ON), the iMX8MMini Boot ROM code uses the carrier SD card as boot media regardless of whether the module eMMC flash contains a properly programmed system image or not.



Figure 7-16. SPL boot selector on EP1 carrier board (S2801).

Forced carrier SD card boot mode.

Example:

```
U-Boot SPL 2019.04-4.19.35-1.1.0+ged283a8930 (Jul 13 2020 - 07:35:13 +0000)
-----
company ..... msc
form factor ..... sm2s
platform ..... imx8mm
processor ..... qc
feature ..... 03N0840E
serial ..... 1009641171
revision (MES) ... B0
boot count ..... 156
-----
DDRINFO: start DRAM init
DRAM PHY: 1D image training for 3000MTS ... passed
DRAM PHY: 1D image training for 400MTS ... passed
DRAM PHY: 1D image training for 100MTS ... passed
DRAM PHY: 2D image training for 3000MTS ... passed
DDRINFO:ddrphy calibration done
DDRINFO: ddrmix config done
Normal Boot
Trying to boot from MMC2

U-Boot 2019.04-4.19.35-1.1.0+ged283a8930 (Jul 13 2020 - 07:35:13 +0000)
```

Figure 7-17. Forced SPL boot from carrier SD card

7.3.1.2 TEST# = HIGH: Booting SPL/U-Boot from module eMMC flash

With TEST# line = HIGH (e.g. on MSC SM2-MB-EP1 Carrier Board: "Test Mode Select" DIP Switch = OFF), the iMX8MMini Boot ROM code uses the module eMMC flash as **primary** and the carrier SD card as **secondary** (fallback) boot media. The fall back media is always selected, when booting from primary media is not possible (empty, corrupted, etc.)



Figure 7-18. SPL boot selector on EP1 carrier board (S2801). eMMC flash boot mode (default).

Example:


```

U-Boot SPL 2019.04-4.19.35-1.1.0+ged283a8930 (Jul 13 2020 - 07:35:13 +0000)
-----
company ..... msc
form factor ..... sm2s
platform ..... imx8mm
processor ..... qc
feature ..... 03N0840E
serial ..... 1009641171
revision (MES) ... B0
boot count ..... 157
-----
DDRINFO: start DRAM init
DRAM PHY: 1D image training for 3000MTS ... passed
DRAM PHY: 1D image training for 400MTS ... passed
DRAM PHY: 1D image training for 100MTS ... passed
DRAM PHY: 2D image training for 3000MTS ... passed
DDRINFO:ddrphy calibration done
DDRINFO: ddrmix config done
Normal Boot
Trying to boot from MMC1

U-Boot 2019.04-4.19.35-1.1.0+ged283a8930 (Jul 13 2020 - 07:35:13 +0000)

```

Figure 7-19. SPL boot from module eMMC flash.

7.3.1.3 Booting SPL from USB

Not supported yet.

7.3.2 Booting OS

According to chapter 4.15 “Boot Options”, the BOOT_SEL [0:2] pins are used to select one of the following OS boot schema.

7.3.2.1 Booting OS from carrier SD card


Carrier SD Card Boot Mode			
BOOT_SEL0#	BOOT_SEL1#	BOOT_SEL2#	Dip Switches on SM2-MB-EP1
HIGH	LOW	LOW	

Table 27: Carrier SD Card Boot Mode

In this configuration the Linux kernel image (Image) and the device tree blob are loaded from the first partition on the carrier SD card. The second partition contains the Linux file system (FS, ext4).

Example:

```
Hit any key to stop autoboot: 0
Boardinfo: OK, complete.
Using carrier SD card as boot device ...
switch to partitions #0, OK
mmc1 is current device
Loading image <Image> from mmc 1:1
24478208 bytes read in 1043 ms (22.4 MiB/s)
Loading fdt <msc-sm2s-imx8mm-03N0840E-headless.dtb> from mmc 1:1
46277 bytes read in 8 ms (5.5 MiB/s)
## Flattened Device Tree blob at 43000000
   Booting using the fdt blob at 0x43000000
   Using Device Tree in place at 0000000043000000, end 000000004300e4c4

Starting kernel ...

[ 0.000000] Booting Linux on physical CPU 0x0000000000 [0x410fd034]
[ 0.000000] Linux version 4.19.35-1.1.0+ga9eed949e290 (oe-user@oe-host) (gcc version 8.3.0 (GCC)) #1 SMP PREEMPT Mon
Jul 13 07:21:05 UTC 2020
[ 0.000000] Machine model: MSC SM2S-IMX8MM
```

Figure 7-20. Booting OS (linux) from on-carrier SD card.

7.3.2.2 Booting OS from module eMMC flash

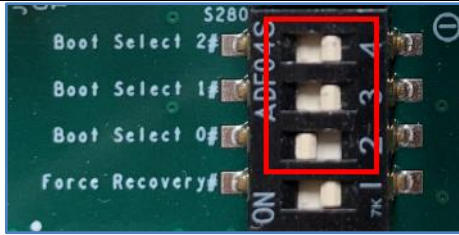
eMMC Boot Mode			
BOOT_SEL0#	BOOT_SEL1#	BOOT_SEL2#	Dip Switches on SM2-MB-EP1
LOW	HIGH	HIGH	

Table 28: eMMC Boot Mode

In this configuration the Linux kernel image (Image) and the device tree blob are loaded from the first partition on the module eMMC flash. The second partition contains the Linux file system (FS, ext4).

Example:

```
Hit any key to stop autoboot: 0
Boardinfo: OK, complete.
Using module eMMC flash as boot device ...
switch to partitions #0, OK
mmc0(part 0) is current device
Loading image <Image> from mmc 0:1
24478208 bytes read in 94 ms (248.3 MiB/s)
Loading fdt <msc-sm2s-imx8mm-03N0840E-headless.dtb> from mmc 0:1
46277 bytes read in 4 ms (11 MiB/s)
## Flattened Device Tree blob at 43000000
   Booting using the fdt blob at 0x43000000
   Using Device Tree in place at 0000000043000000, end 000000004300e4c4

Starting kernel ...

[ 0.000000] Booting Linux on physical CPU 0x000000000 [0x410fd034]
[ 0.000000] Linux version 4.19.35-1.1.0+ga9eed949e290 (oe-user@oe-host) (gcc version 8.3.0 (GCC)) #1 SMP PREEMPT Mon
Jul 13 07:21:05 UTC 2020
[ 0.000000] Machine model: MSC SM2S-IMX8MM
```

Figure 7-21. Booting OS (linux) from on-module eMMC flash.

7.3.2.3 Booting OS from Network (Ethernet)

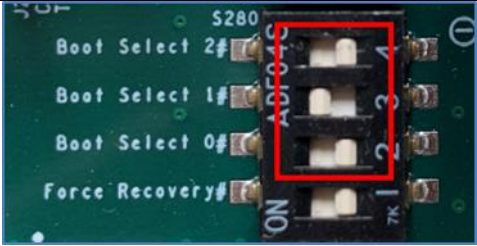
Network Boot Mode			
BOOT_SEL0#	BOOT_SEL1#	BOOT_SEL2#	Dip Switches on SM2-MB-EP1
HIGH	LOW	HIGH	

Table 29: Network Boot Mode

In this configuration the Linux kernel image (Image) and the device tree blob are loaded from the TFTP-, and the Linux file system is mounting on NFS-Server on LAN.

Depending on your local network infrastructure, set the following environment variables at the U-Boot prompt:

```
setenv serverip <TFTP/NFS server IP>
setenv nfsroot <nfs root path on NFS server>
saveenv
```

Example:

```
U-Boot>
u-boot> setenv nfsroot /home/public/0103801/rootfs
u-boot> setenv serverip 172.30.206.100
u-boot> saveenv
Saving Environment to MMC... Writing to MMC(0)... OK
u-boot>
```

Figure 7-22. Preparing U-Boot environment for net boot.

Then continue system boot with:

```
Boot
```

or simply reboot (power cycle) your system.

Example:

[illegible]

Figure 7-23. Booting OS (linux) from network.

7.3.2.4 Booting OS from USB

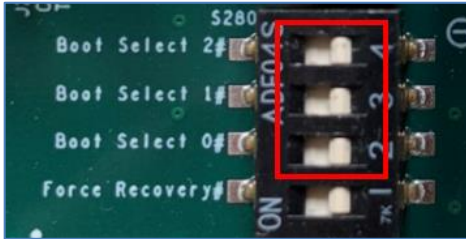
USB Boot Mode			
BOOT_SEL0#	BOOT_SEL1#	BOOT_SEL2#	Dip Switches on SM2-MB-EP1
HIGH	HIGH	HIGH	

Table 30: USB Boot Mode

In this configuration the Linux kernel image (Image) and the device tree blob are being loaded from the first partition on the USB. The second partition contains the Linux file system (FS, ext4).

Example:

```
Normal Boot
Hit any key to stop autoboot: 0
Boardinfo: OK, complete.
Using USB as boot device ...
starting USB...
USB0: Port not available.
USB1: USB EHCI 1.00
scanning bus 1 for devices... 3 USB Device(s) found
       scanning usb for storage devices... 1 Storage Device(s) found

Device 0: Vendor: Intenso Rev: 8.07 Prod: Rainbow Line
         Type: Removable Hard Disk
         Capacity: 3840.0 MB = 3.7 GB (7864320 x 512)
... is now current device
Loading image <Image> from usb 0:1
24478208 bytes read in 2004 ms (11.6 MiB/s)
Loading fdt <msc-sm2s-imx8mm-03N0840E-headless.dtb> from usb 0:1
46277 bytes read in 13 ms (3.4 MiB/s)
## Flattened Device Tree blob at 43000000
   Booting using the fdt blob at 0x43000000
   Using Device Tree in place at 0000000043000000, end 000000004300e4c4

Starting kernel ...

[ 0.000000] Booting Linux on physical CPU 0x00000000 [0x410fd034]
[ 0.000000] Linux version 4.19.35-1.1.0+ga9eed949e290 (oe-user@oe-host) (gcc version 8.3.0 (GCC)) #1 SMP PREEMPT Mon
Jul 13 07:21:05 UTC 2020
[ 0.000000] Machine model: MSC SM2S-IMX8MM
```

Figure 7-24. Booting OS (linux) from USB device (pen drive).

7.3.2.5 Available device tree blobs (DTBs)

Table 31. Available DT-blobs.

dtb	comment
msc-sm2s-imx8mm-03N0840E-headless	for 03N0840E variant based headless systems
msc-sm2s-imx8mm-03N0840E-dsi-rm67191	for 03N0840E variant based systems with Raydium RM67191 (1080x1920) MIPI/DSI panel
msc-sm2s-imx8mm-03N0840E-dsi-avd-tt78qt	for 03N0840E variant based systems with AV Display TT78QT (400x1280) MIPI/DSI panel
msc-sm2s-imx8mm-03N0880I-headless	for 03N0880I variant based headless systems
msc-sm2s-imx8mm-03N0880I-dsi-rm67191	for 03N0880I variant based systems with Raydium RM67191 (1080x1920) MIPI/DSI panel
msc-sm2s-imx8mm-03N4200I-headless	for 03N4200I variant based headless systems
msc-sm2s-imx8mm-03N4200I-lvds-am800480n7	for 03N4200I variant based systems with Ampire AMA-800480N7 (800x480) single channel LVDS panel
msc-sm2s-imx8mm-03N4200I-lvds-ama121a01	for 03N4200I variant based systems with Ampire AMA-121A01 (1280x800) single channel LVDS panel
msc-sm2s-imx8mm-03N4200I-lvds-p251hvn01	for 03N4200I variant based systems with AUO P215HVN01 (1920x1080) dual channel LVDS panel
msc-sm2s-imx8mm-03N4210I-headless	for 03N4210I variant based headless systems
msc-sm2s-imx8mm-03N4210I-lvds-am800480n7	for 03N4210I variant based systems with Ampire AMA-800480N7 (800x480) single channel LVDS panel
msc-sm2s-imx8mm-03N4210I-lvds-ama121a01	for 03N4210I variant based systems with Ampire AMA-121A01 (1280x800) single channel LVDS panel
msc-sm2s-imx8mm-03N4210I-lvds-p251hvn01	for 03N4210I variant based systems with AUO P215HVN01 (1920x1080) dual channel LVDS panel
msc-sm2s-imx8mm-13N4200I-headless	for 13N4200I variant based headless systems
msc-sm2s-imx8mm-13N4200I-dsi-tm055jvhg14	for 13N4200I variant based systems with Tianma TM055JVHG14 (720x1280) MIPI/DSI panel
msc-sm2s-imx8mm-13N4200I-lvds-am800480n7	for 13N4200I variant based systems with Ampire AMA-800480N7 (800x480) single channel LVDS panel
msc-sm2s-imx8mm-13N4200I-lvds-ama121a01	for 13N4200I variant based systems with Ampire AMA-121A01 (1280x800) single channel LVDS panel
msc-sm2s-imx8mm-13N4200I-lvds-p251hvn01	for 13N4200I variant based systems with AUO P215HVN01 (1920x1080) dual channel LVDS panel
msc-sm2s-imx8mm-14N0261I-headless	for 14N0261I variant based headless systems

msc-sm2s-imx8mm-14N0261l-lvds-ama070a04	for 14N0261l variant based systems with Ampire AMA-800480N7 (800x480) single channel LVDS panel
msc-sm2s-imx8mm-14N0261l-lvds-ama121a01	for 14N0261l variant based systems with Ampire AMA-121A01 (1280x800) single channel LVDS panel
msc-sm2s-imx8mm-14N0261l-lvds-p251hvn01	for 14N0261l variant based systems with AUO P215HVN01 (1920x1080) dual channel LVDS panel

7.3.3 Login to FS

Login is enabled via serial console (115200 baud/8 bits/no parity). All images also have telnet login enabled.

Table 32. Available user accounts.

account	password	comment
root	mscldk	root user
msc	msc	standard user with sudo permissions

7.3.4 SMARC GPIO access

According to [1] following GPIOs are available on sms2-imx8mmmini module:

Table 33. Available SMARC GPIOs

SMARC GPIO	IMX GPIO		Linux/U-Boot idx (bank-1)*32+id	mscio-cmd' alias	comment
	bank	id			
0	1	0	0	gpio0	EP1.CAM0_PWR
1	1	1	1	gpio1	available
2	1	3	3	gpio2	EP1.CAM0_RST
3	1	5	5	gpio3	available
4	1	6	6	gpio4	EP1.HDA_RST
5	5	2	130	gpio5	EP1.FAN_PWMOUT
6	5	1	129	gpio6	EP1.FAN_TACHIN
7	4	22	118	gpio7	available

8	3	25	89	gpio8	available
9	4	28	124	gpio9	available
10	1	9	9	gpio10	available
11	1	7	7	gpio11	available

For detailed information about GPIO hardware- and software-operation see [6]:

- chapter 2.1.5.6.1 "GPIO Hardware Operation"
- chapter 2.1.6 "General Purpose Input/Output (GPIO)"

7.3.5 Bug Reporting

To simplify collecting information necessary for effectively responding to bug reports, please use the `msc_bug_report.sh` tool to generate bug report message. It will collect all necessary information like hardware description/configuration, kernel logs etc.

- Run `msc_bug_report.sh`

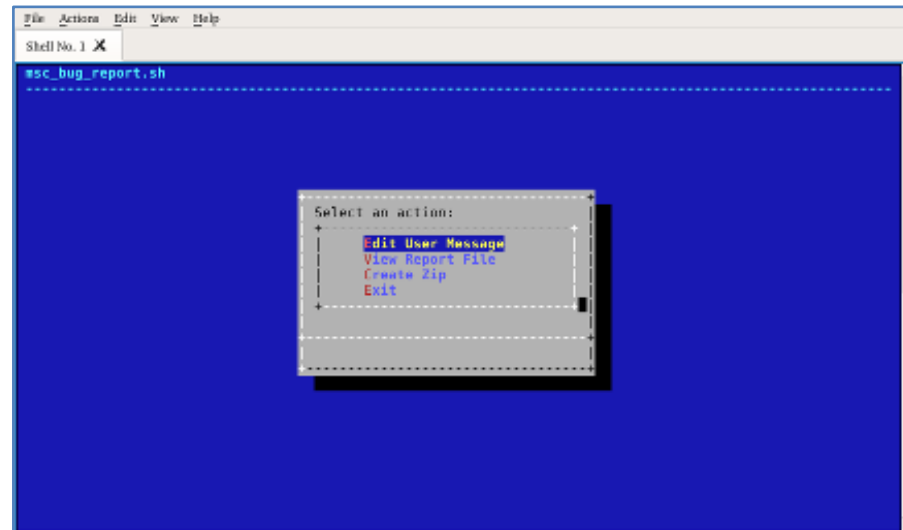


Figure 7-25. Bug report. Main page.

- Select “Edit User Message”.

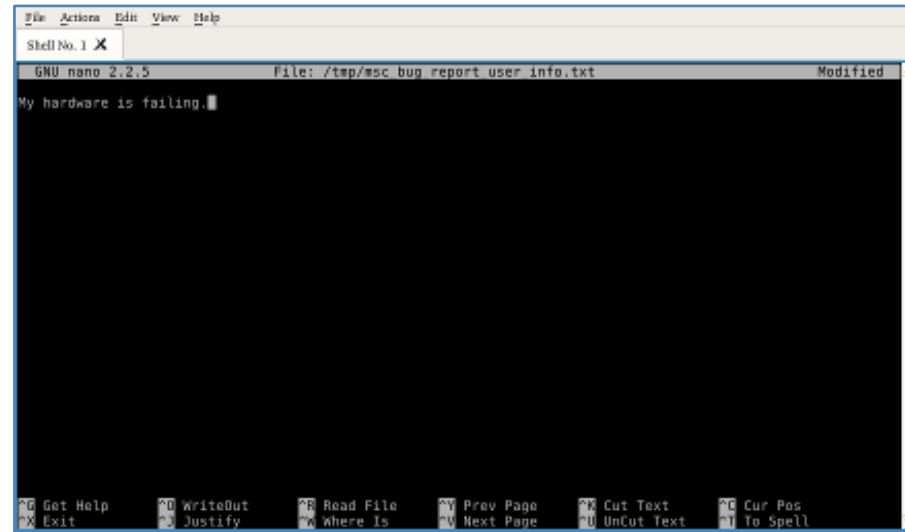
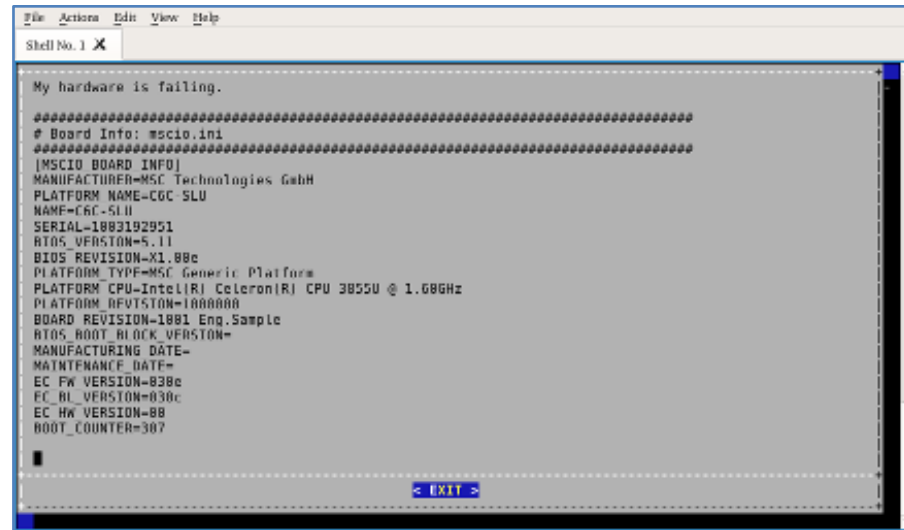


Figure 7-26. Bug report. User message editor.

- Enter bug report message and press Ctrl-O and Ctrl-X.

- Optionally you can then view the message with the board report (hardware information).



The screenshot shows a window titled "Shell No. 1" with a menu bar (File, Actions, Edit, View, Help). The main content area displays the following text:

```
My hardware is failing.

#####
# Board Info: mscio.ini
#####
[MSCIO BOARD INFO]
MANUFACTURER=MSC Technologies GmbH
PLATFORM_NAME=CGC SLU
NAME=CGC-SLU
SERIAL=1883192951
RTOS_VERSION=5.11
BIOS_REVISION=X1.08c
PLATFORM_TYPE=MSC Generic Platform
PLATFORM_CPU=Intel(R) Celeron(R) CPU 3855U @ 1.68GHz
PLATFORM_HWVERSION=1000000
BOARD_REVISION=1881 Eng.Sample
RTOS_ROOT_BLOCK_VERSION=
MANUFACTURING_DATE=
MAINTENANCE_DATE=
EC_FW_VERSION=838c
EC_HW_VERSION=838c
EC_HW_VERSION=88
ROOT_COUNTER=307

< EXIT >
```

Figure 7-27. Bug report. Viewer page.

- Press “Create a zip file” and select the components you want to send (e.g. bootlog, mscio.ini, last kernel logs (dmesg) or the installed hardware).

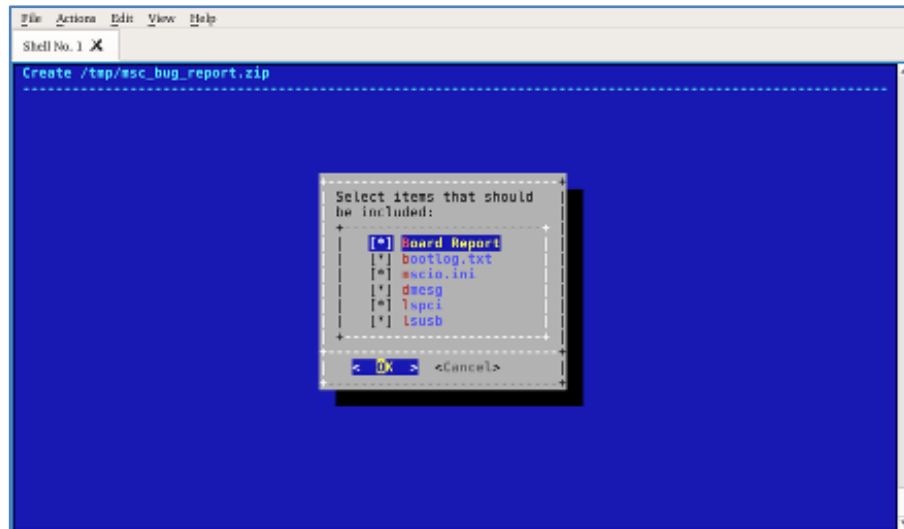


Figure 7-28. Bug report. Zip archive content selector.

- Press “Save ZIP to a disc” and select the filesystem where to store the zip file. It is recommended to use a USB stick.
- Send the files msc_bug_report_brief.txt and msc_bug_report.zip to Avnet Embedded /MSC Technical Support: support.boards@avnet.eu

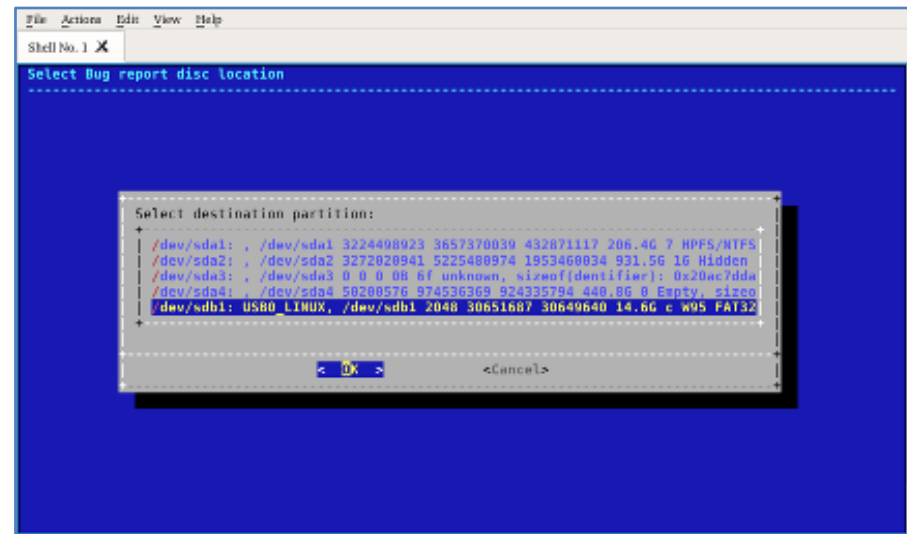


Figure 7-29. Bug report. Target partition selector.

7.4 Hotfixes and updating MSC-LDK

Typically twice a year a full MSC-LDK release is issued. A release may contain an updated Yocto or other updated layers as well as new supported boards. For each release an own branch is used (e.g. v1.0.0) which is tagged with the date encoded (e.g. LC984_20150421_V0_4_0, 21st April 2015), too. The release is checked out using the version syntax (e.g.: `git checkout v1.3.0`) as described above. Sometimes an intermediate hot-fix is necessary which doesn't modify the resulting image but fixes changed repository locations of third party software or similar minor changes. Hot-fixes are tagged with a newer date stamp (e.g. LC984_20160113_V0_4_0). A hot-fix can be checked out explicitly using these tags. When MSC-LDK is checked out the first time all hot-fixes are applied automatically.

To update an older checkout and to pull all the newer hot-fixes, run "scripts/update.py" from the MSC-LDK root directory. This will update MSC-LDK and all layers. Depending on the kind of hot-fixes applied in the meantime, running `setup.py` again might be necessary. When a hot-fix has been checked out explicitly, running update will not make sense and it will fail with an error.

After the first call of `setup.py`, no manual "git checkout" is required, as its layers will already be in synchronization with MSC-LDK. Either use `update.py` or clone MSC-LDK again. The subdirectories "download" and "sstate-cached" can be moved to other MSC-LDK installations or shared by symbolic links.

Customers who want to keep pace with development rather than building on a fixed status might want to use the "master" branch instead of a released branch. Here `update.py` must also be used. However, on the SM2S-IMX8MINI, currently the "master" branch is not applicable, instead all current changes are applied as hotfixes to the MSC-LDK release state of "v1.5.0" (State: March 2021).

This may change soon, as MSC is working on the next release already. Please ask Avnet Embedded /MSC Technical Support for respective last state at any given time.

8 Troubleshooting

8.1 Known issues and limitations

8.1.1 Issue 1. SPI interfaces not available on MSC SM2S-IMX8MINIQC-14N0261I variant
Both CAN transceivers drive SPI[0:1]_DIN signals low even though not selected by chip select signal.

Source: Hardware

Workaround: Not specified yet.

8.2 Support

For additional help please contact Avnet Embedded /MSC Technical Support:

Phone: +49 8165 906-200

WWW <https://www.msc-technologies.eu/support/boards.html>

Email: support.boards@avnet.eu