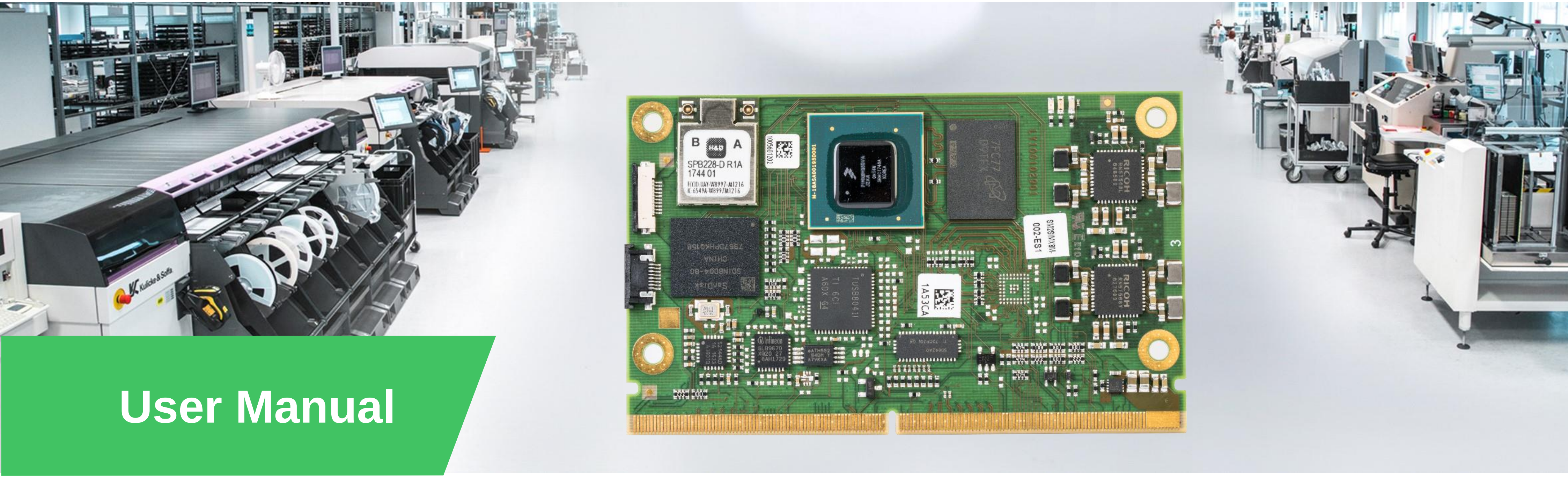


User Manual

SMARC™ Module

MSC SM2S-IMX8M

SMARC Rev. 2.0 Standard

Version 1.3   13.08.2021

Preface

Copyright Notice

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 owns 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 team:

Phone: +49 8165 906-200

Email: support.boards@avnet.eu

Contents

Revision History

Rev.	Date	Description
1.0	February 9th, 2021	First Release
1.1	March 18th, 2021	Several fixes
1.2	July 19th, 2021	Updated “Module Connector Pinout” chapter 3: Swapped SPI1_DIN (P57) and SPI1_DO (P58) correction
1.3	August 13th, 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] Universal Bus Specification
usb_20.pdf
Last update: April 27th, 2000
http://www.usb.org

[4] i.MX 8M Series of Application Processors
IMX8MDQLQIEC.pdf
http://www.nxp.com

[5] Module Datasheet
MSC-SM2S-IMX8M.pdf
https://www.msc-technologies.eu/support/boards/smarc/msc-sm2s-imx8m.html

[6] i.MX Reference Manual.
i.MX_Linux_Reference_Manual.pdf
Version: L4.9.51_imx8mq-ga, 03/2018
http://www.nxp.com

[7] i.MX Linux Reference Manual.
i.MX_Reference_Manual.pdf
Version: L4.14.98-2.0.0_ga, 04/2019
http://www.nxp.com

[8] i.MX Linux® User's Guide
i.MX_Linux_User's_Guide.pdf
Version: L4.14.98-2.0.0_ga, 04/2019
http://www.nxp.com

[9] i.MX Porting Guide
i.MX_Porting_Guide.pdf
Version: L4.14.98-2.0.0_ga, 04/2019
http://www.nxp.com

[10] i.MX Yocto Project User's Guide()
i.MX_Yocto_Project_User's_Guide.pdf
Version: L4.14.98-2.0.0_ga, 04/2019
http://www.nxp.com

[11] i.MX Linux® Release Notes
i.MX_Linux_Release_Notes.pdf.pdf
Version: L4.14.98-2.0.0_ga, 05/2019
http://www.nxp.com

[12] 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, BIOS flash, SATA- 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, SATA, USB, LPC 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 SMARC™ EP1 carrier board.

1.1 Key Features

SoC:

- NXP™ i.MX 8M ARM® CORTEX™-A53
  Assembly options for i.MX 8M Dual, i.MX 8M QuadLite and i.MX 8M Quad

SDRAM:

- up to 4GB LPDDR4 (soldered on module)

Video:

- Dual Channel LVDS 18-bit/24-bit (1920x1080 max.) or Single Channel LVDS (1366x768 max.) or MIPI-DSI 4 lane (1920x1080 max.)
- HDMI 2.0a (4096x2160 max.) or DisplayPort 1.3 (4096x2160 max.)

Audio:

- 2x I^2S links for audio codec connection

Camera Interface:

- 2x MIPI CSI-2 (4 Lane / 2 Lane)

PCI Express Interface:

- 2x PCIe x1 Gen.2 Lane

Network:

- 1x 10/100/1000BASE-T Ethernet
- Optional: HD Wireless Module SPB228, MU-MIMO 2x2 with 802.11 ac/a/b/g/n and Bluetooth/BLE support, soldered on module

USB:

- 1x USB2.0 host port with device Interface capability and on-the-go (OTG) support
- Up to 2x USB2.0 host ports
- Up to 2x USB3.0 host ports

SATA:

- The i.MX 8M does not support SATA

GPIO:

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

SPI:

- Up to 2x SPI with 2 chip selects each
  NOTE: CAN and SPI are mutual exclusive
- Optional QSPI NOR Flash *(population option on module only)*

I²C Bus:

- I²C and SM-Bus for Power Management functions
- I²C bus for general purpose
- I²C bus for LVDS display interface
- I²C bus for HDMI interface
- 2x I²C bus for camera interfaces

UART:

- 2x legacy UARTs without RTS/CTS support
- Up to 2x legacy UARTs with RTS/CTS support
  NOTE: Quantity of UART interfaces with RTS/CTS support is dependent on SPI/CAN, mutual exclusive option.

SDIO/eMMC:

- SD Host Controller Standard Specification 3.0 and MMC System Specification 5.1
- 1bit/4bit SDIO
- eMMC NAND flash with up to 64GByte (option)

EEPROM:

- 64Kb EEPROM on I2C bus for module information and user applications

CAN:

- Up to 2x CAN 2.0B (Controller Area Network) at 1Mbps
  NOTE: Quantity of UART interfaces with RTS/CTS support is dependent on SPI/CAN, mutual exclusive option
  NOTE: CAN and SPI are mutual exclusive

Real-time Clock:

- Module provides a high accuracy RTC
- Optional: RTC with temperature compensated DTCXO

Watchdog:

- Module provides a watchdog output to the SMARC ™ connector

Security Device:

- Advanced Security, Safety, and Reliability integrated in the SOC
- Optional: Infineon Trusted Platform Module (TPM) SLB9671 2.0

Environment Temperature:

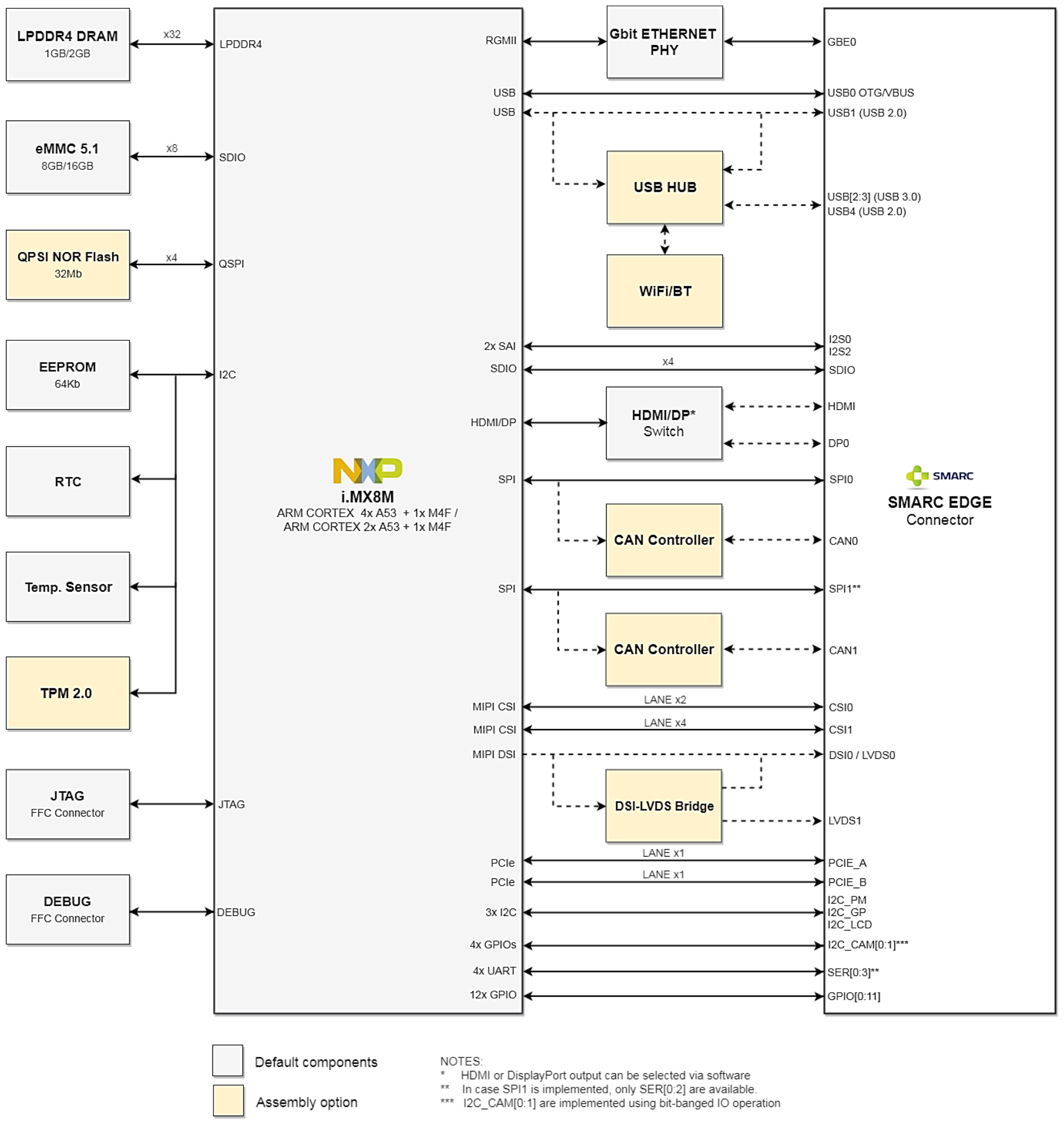
- 0° … 70°C (all components commercial temp. or better)
- -40° … 85°C (all components industrial temp.)
- -40° … 85°C (storage)

Environment Humidity:

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

1.2 Block Diagram

Figure 1-1: Block Diagram



1.3 Power Supply

Table 1-1: Module Power Inputs

Power Rail	Description		
VDD_IN	Primary power input	Nominal	+5V
		Voltage Range	+4.75V ... +5.25V
		Max. Input Ripple	±100mV
		Rate of Voltage Rising	< 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	1uA typical @ VDD = 3V
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, HDMI monitor used, no USB devices
- Linux Idle: Ethernet link established, HDMI monitor used, no USB devices
- Linux Heavy Load: CPU load 100% on each core, memory tester, HDMI monitor and LVDS display used, Ethernet traffic generated with iperf
- Sleep Mode: with active wake on serial port input (Serial 0)

1.4.2 Hardware used

Table 1-2: Modules Used for Power Consumption Measurement

Order Number	Reference	CPU	RAM	Temp. Range
73289	MSC SM2S-IMX8M-DC-03N0600I PCBFTX	8M Dual, Dual-Core Cortex-A53 at 1.3Ghz	1G LPDDR4	-40°C to +85°C
73301	MSC SM2S-IMX8M-QCL-13N0600I PCBFTX	8M QuadLite, Quad-Core Cortex-A53 at 1.3Ghz	2G LPDDR4	-40°C to +85°C
73303	MSC SM2S-IMX8M-QC-13N0600I PCBFTX	8M Quad, Quad-Core Cortex-A53 at 1.3Ghz	2G LPDDR4	-40°C to +85°C

1.4.3 Measurement Results

Table 1-3: Power Consumption Measurement

Referece	CPU	Uboot Idle [W]	Linux idle [W]	Linux Heavy Load [W]
MSC SM2S-IMX8M-DC-03N0600I PCBFTX	8M Dual-Core Cortex-A53 at 1.3Ghz	3.22	2.70	5.25
MSC SM2S-IMX8M-QC-13N0600I PCBFTX	8M Quad Quad-Core Cortex-A53 at 1.3Ghz	3.55	2.90	6.20

NOTE: Unless stated otherwise, all measurements were taken at room temperature approximately 25°C.

1.5 Mechanical Dimensions

Figure 1-2: Module Dimensions

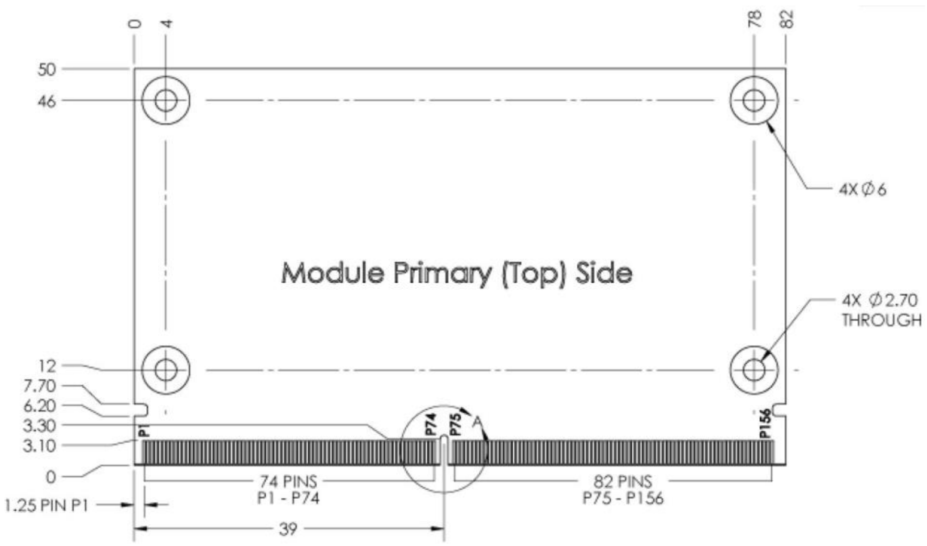
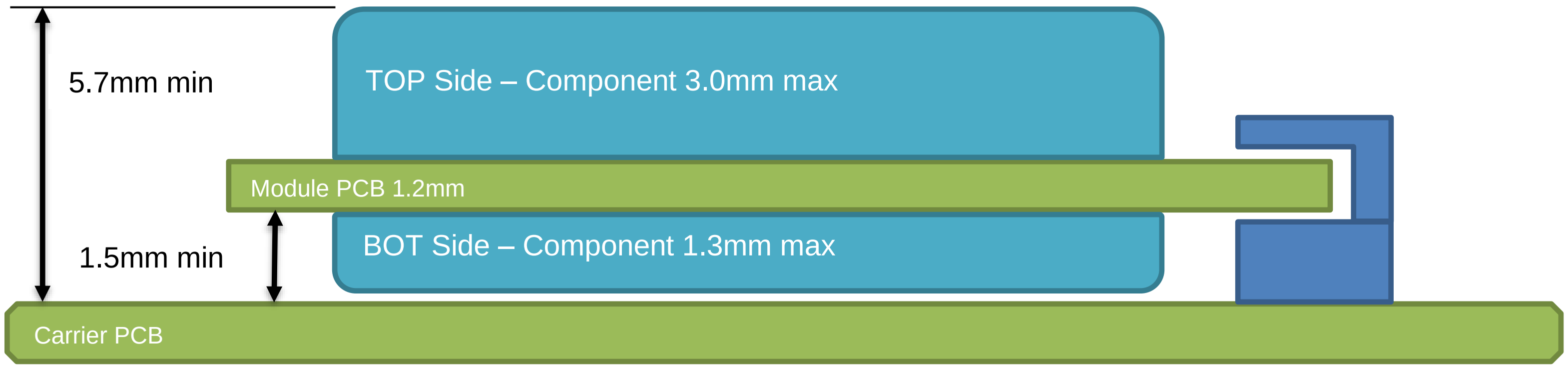


Figure 1-3: Overall Height without heat spreader of the SMARC™ Module



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

1.6 Mechanical Deflection of PCB

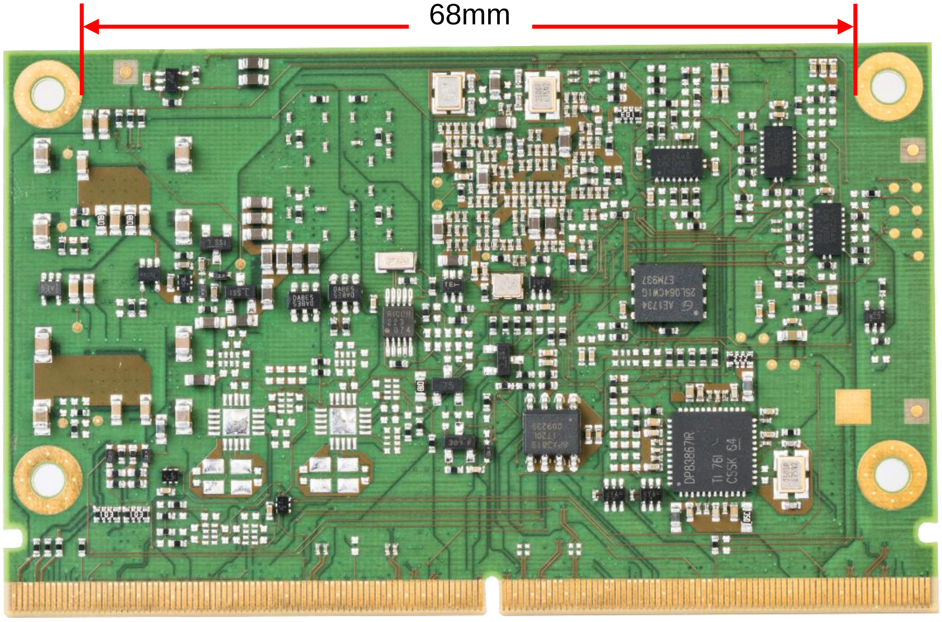
For thermal heat dissipation the heat sink needs to be pressed onto the CPU. The higher the pressure the lower is the thermal transition resistance and consequently the better the thermal cooling. This pressure may result in a slight mechanical bending of the SMARC module.

Production tolerance, material deviation and thermal expansion lead to a range of possible pressure and bending. No pressure with an air gap between the heat spreader and the chip case needs to be avoided and likewise too high a distortion.

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

Referring to data sheets of the relevant parts and AEC-Q200 the bending needs to be below 1mm over a length of 90mm. (1.11%) → 0.75mm

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 materials and small aluminum blocks filling the gap between CPU and chipset dies and heat spreader plate. Stand-alone heat sinks generally offer best thermal transfer characteristics. Contact MSC Technologies support for a suitable heat sink solution for the i.MX8M 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.

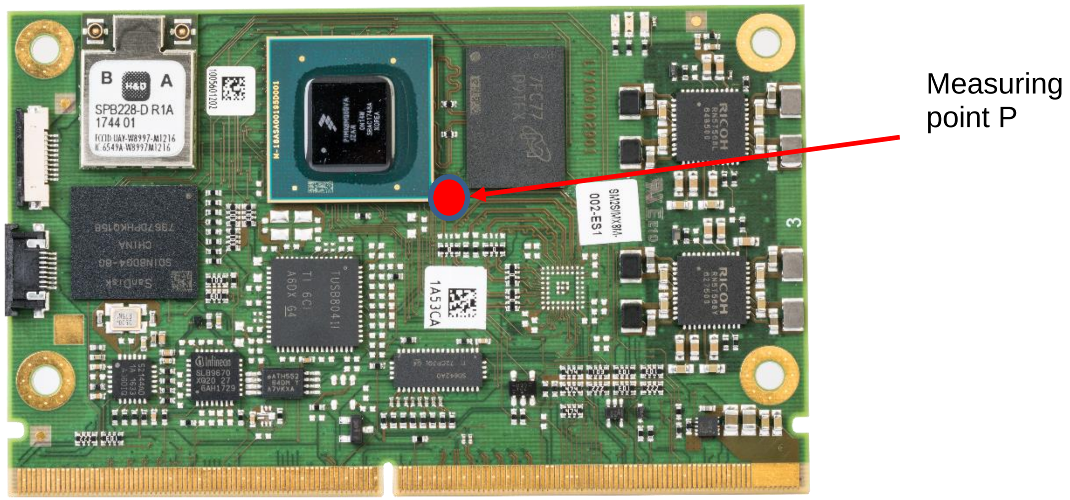
The stabilized temperature measured on the heat spreader and the module PCB during runtime mostly depends on the computing power demand from the application and the cooling solution implemented in the system. It is the responsibility of the system designer to provide a cooling solution in addition to 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 2-1: 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: Temperature measuring point on PCB



3 Module Connector Pinout

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

Table 3-1: 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	NC

Primary (Top) Side		Secondary (Bottom) Side	
P18	GND	S18	NC
P19	GBE0_MDI3-	S19	NC
P20	GBE0_MDI3+	S20	NC
P21	GBE0_LINK100#	S21	NC
P22	GBE0_LINK1000#	S22	NC
P23	GBE0_MDI2-	S23	NC
P24	GBE0_MDI2+	S24	NC
P25	GBE0_LINK_ACT#	S25	GND
P26	GBE0_MDI1-	S26	NC
P27	GBE0_MDI1+	S27	NC
P28	NC	S28	NC
P29	GBE0_MDI0-	S29	NC
P30	GBE0_MDI0+	S30	NC
P31	SPI0_CS1#	S31	NC
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_SDOUT
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_SDOUT
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 (See 8.1.6)	S57	NC
P58	SPI1_DO (See 8.1.6)	S58	NC
P59	GND	S59	NC
P60	USB0+	S60	NC
P61	USB0-	S61	GND
P62	USB0_EN_OC#	S62	USB3_SSTx+
P63	USB0_VBUS_DET	S63	USB3_SSTx-
P64	USB0_OTG_ID	S64	GND
P65	USB1+	S65	USB3_SSRx+
P66	USB1-	S66	USB3_SSRx-
P67	USB1_EN_OC#	S67	GND
P68	GND	S68	USB3+
P69	USB2+	S69	USB3-
P70	USB2-	S70	GND
P71	USB2_EN_OC#	S71	USB2_SSTx+
P72	CPU_TMS	S72	USB2_SSTx-
P73	CPU_TDI	S73	GND
P74	USB3_EN_OC#	S74	USB2_SSRx+
		S75	USB2_SSTx-
KEY		KEY	

Primary (Top) Side		Secondary (Bottom) Side	
P75	PCIE_A_RST#	S76	PCIE_B_RST#
P76	USB4_EN_OC#	S77	NC
P77	CPU_TCK	S78	NC
P78	CPU_TDO	S79	NC
P79	GND	S80	GND
P80	NC	S81	NC
P81	NC	S82	NC
P82	GND	S83	GND
P83	PCIE_A_REFCK+	S84	PCIE_B_REFCK+
P84	PCIE_A_REFCK-	S85	PCIE_B_REFCK-
P85	GND	S86	GND
P86	PCIE_A_RX+	S87	PCIE_B_RX+
P87	PCIE_A_RX-	S88	PCIE_B_RX-
P88	GND	S89	GND
P89	PCIE_A_TX+	S90	PCIE_B_TX+
P90	PCIE_A_TX-	S91	PCIE_B_TX-
P91	GND	S92	GND
P92	HDMI_D2+	S93	DP0_LANE0+
P93	HDMI_D2-	S94	DP0_LANE0-
P94	GND	S95	NC
P95	HDMI_D1+	S96	DP0_LANE1+

Primary (Top) Side		Secondary (Bottom) Side	
P96	HDMI_D1-	S97	DP0_LANE1-
P97	GND	S98	DP0_HPD
P98	HDMI_D0+	S99	DP0_LANE2+
P99	HDMI_D0-	S100	DP0_LANE2-
P100	GND	S101	GND
P101	HDMI_CK+	S102	DP0_LANE3+
P102	HDMI_CK-	S103	DP0_LANE3-
P103	GND	S104	NC
P104	HDMI_HPD	S105	DP0_AUX+
P105	HDMI_CTRL_CK	S106	DP0_AUX-
P106	HDMI_CTRL_DAT	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_PRSNT#
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 additional schematic action taken on the module pull-up resistor (PU) or pull-down resistor (PD).

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 4-1: I²S Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
I2S0_LRCK	O PP	1.8V CMOS	H4	SAI2_TXFS	1.8V		Sample-synchronization signal to the codec(s).
I2S0_CK	O PP	1.8V CMOS	J5	SAI2_TXC	1.8V		Serial data clock
I2S0_SDOUT	O PP	1.8V CMOS	G5	SAI2_TXD0	1.8V		Serial TDM data output to the codec.
I2S0_SDIN	I	1.8V CMOS	H6	SAI2_RXD0	1.8V		Serial TDM data inputs from the codec.
I2S2_LRCK	O PP	1.8V CMOS	G2	SAI3_TXFS	1.8V		Sample-synchronization signal to the codec(s).
I2S2_CK	O PP	1.8V CMOS	C4	SAI3_TXC	1.8V		Serial data clock
I2S2_SDOUT	O PP	1.8V CMOS	C3	SAI3_TXD	1.8V		Serial TDM data output to the codec.
I2S2_SDIN	I	1.8V CMOS	F3	SAI3_RXD	1.8V		Serial TDM data inputs from the codec.
AUDIO_MCK	O PP	1.8V CMOS	K7	GPIO1_IO14	1.8V		Clock Output (CCM_CLKO1)

4.2 Ethernet

Based on Texas Instruments™ DP83867 Ethernet controller the module provides 10/100/1000 Mbps Ethernet with MDI differential pairs for an external transformer.

The DP83867 includes a voltage mode line driver so it doesn't require an analog powered center tap. Therefore Pin P28 GBE0_CTREF specified in the SMARC™ specification 2.0 is not connected on the module.

The DP83867 has 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 4-2: Ethernet Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	pin name on i.MX8M	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 PP	3.3V CMOS	n.a.	n.a.	3.3V		Link/Activity Indication, active low, 24mA output drive.
GBE0_LINK100#	O PP	3.3V CMOS	n.a	n.a			Link Speed Indication for 100Mbps, active low, 24mA output drive.
GBE0_LINK1000#	O PP	3.3V CMOS	n.a.	n.a.	3.3V		Link Speed Indication for 1000Mbps, active low, 24mA output drive.

NOTE: If only 10/100 Mbit/s is required, the unused MDI Pins can be left unconnected.

4.3 PCI Express

The i.MX8M SoC supports two PCIe x1 Gen2 lanes. Please note, that only single channel (x1) interfaces are supported.

Table 4-3: PCIe Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
PCIE_A_TX+ PCIE_A_TX-	O	LVDS PCIe	J24 J25	PCIE1_TXN_P PCIE1_TXN_N	According to PCIe spec		PCI Express Differential Transmit Pairs. AC coupled on module
PCIE_A_RX+ PCIE_A_RX-	I	LVDS PCIe	H24 H25	PCIE1_RXN_P PCIE1_RXN_N	According to PCIe spec		PCI Express Differential Receive Pairs
PCIE_A_REFCK+ PCIE_A_REFCK-	O	LVDS PCIe	K24 K25	PCIE1_REF_PAD_CLK_P PCIE1_REF_PAD_CLK_N	According to PCIe spec		PCI Express Reference Clock. AC coupled on module. Clock enabled by default. Please contact support, if spread spectrum support is required.
PCIE_B_TX+ PCIE_B_TX-	O	LVDS PCIe	E24 E25	PCIE2_TXN_P PCIE2_TXN_N	According to PCIe spec		PCI Express Differential Transmit Pairs. AC coupled on module
PCIE_B_RX+ PCIE_B_RX-	I	LVDS PCIe	D24 D25	PCIE2_RXN_P PCIE2_RXN_N	According to PCIe spec		PCI Express Differential Receive Pairs
PCIE_B_REFCK+ PCIE_B_REFCK-	O	LVDS PCIe	F24 F25	PCIE2_REF_PAD_CLK_P PCIE2_REF_PAD_CLK_N	According to PCIe spec		PCI Express Reference Clock. AC coupled on module. Clock enabled by default. Please contact support, if spread spectrum support is required.
PCIE_A_RST#	O PP	3.3V CMOS	K2	SAI1_RXD0	3.3V		PCI Express Reset signal (active low) to connected device. (CPU GPIO4_IO02)
PCIE_B_RST#	O PP	3.3V CMOS	K2	SAI1_RXD0	3.3V		PCI Express Reset signal (active low) to connected device. (CPU GPIO4_IO02)

Signal	Pin Type	Signal Level	Pin on i.MX8	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
PCIE_WAKE#	I	3.3V CMOS	A3	SAI1_MCLK	3.3V	PU 10k	PCI Express Wake signal. Asserted by device when requesting wake up. (CPU GPIO4_IO20)

NOTE: PCIE_A_RST# and PCIE_B_RST# share same CPU pin.

4.4 USB

The USB controller supports USB 3.0 and USB 2.0.

Depending on the module variant different number of USB lanes are available:

- **Option 1** with USB 3.0 Hub:
    - USB[0] = USB 2.0 host/device OTG compliant
    - USB[1] = USB 2.0 host
    - USB[2:3] = USB 3.0 host
    - USB[4] = USB 2.0 host.
- **Option 2** without USB 3.0 Hub:
    - USB[0] = USB 2.0 host/device OTG compliant
    - USB[1] = USB 2.0 host.

Table 4-4: USB Signal Description

Signal	Option Availability	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
USB0+ USB0-	1 & 2	I/O	USB	A14 B14	USB1_DP USB1_DN	3.3V		Differential USB 2.0 data pairs connected to SoC. Can be configured as host or device.

Signal	Option Availability	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Toleranc e	PU/PD	Description
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	A10 B10	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.
USB[2:3]_SSTX+ USB[2:3]_SSTX-	1	O	USB	n.a	n.a	3.3V		USB 3.0 transmit signal differential pairs connected to USB hub. AC coupled on module.
USB[2:3]_SSRX+ USB[2:3]_SSRX-	1	I	USB	n.a	n.a	3.3V		USB 3.0 receive signal differential pairs connected to USB hub.
USB0_VBUS_DET	1 & 2	I	Analog	D14	USB1_VBUS	5V		external VBUS detection pin
USB0_OTG_ID	1 & 2	I	3.3V CMOS	M7	GPIO1_IO10	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	L7 K6	GPIO1_IO12 GPIO1_IO13	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	n.a	n.a	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	Pin name on i.MX8M	Power Toleranc e	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 USB power switches are used, USB[0:4]_EN_OC# pins may be left unconnected.

USB1 overcurrent detection and power enable function are not available in Option 2.

USB1 power is always enabled in Option 2.

4.5 Camera

MIPI CSI-2 interface is supported on CSI0 with 2 lanes and on CSI1 with 4 lanes.

Table 4-5: HDMI Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pi name on i.MX8M	Power Tolerance	PU/PD	Description
CSI0_D[0]- CSI0_D[0]+	I	1.8V CMOS	A23 B23	MIPI_CSI1_D0_N MIPI_CSI1_D0_P	1.8V		CSI differential data inputs
CSI0_D[1]- CSI0_D[1]+	I	1.8V CMOS	C22 D22	MIPI_CSI1_D1_N MIPI_CSI1_D1_P	1.8V		CSI differential data inputs
CSI1_D[0]- CSI1_D[0]+	I	1.8V CMOS	C20 D20	MIPI_CSI2_D0_N MIPI_CSI2_D0_P	1.8V		CSI differential data inputs
CSI1_D[1]- CSI1_D[1]+	I	1.8V CMOS	A20 B20	MIPI_CSI2_D1_N MIPI_CSI2_D1_P	1.8V		CSI differential data inputs

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pi name on i.MX8M	Power Tolerance	PU/PD	Description
CSI1_D[2]- CSI1_D[2]+	I	1.8V CMOS	A21 B21	MIPI_CSI2_D2_N MIPI_CSI2_D2_P	1.8V		CSI differential data inputs
CSI1_D[3]- CSI1_D[3]+	I	1.8V CMOS	C19 D19	MIPI_CSI2_D3_N MIPI_CSI2_D3_P	1.8V		CSI differential data inputs
CSI0_CK+ CSI0_CK-	I	1.8V CMOS	A22 B22	MIPI_CSI1_CLK_N MIPI_CSI1_CLK_P	1.8V		CSI differential clock inputs
CSI1_CK+ CSI1_CK-	I	1.8V CMOS	A19 B19	MIPI_CSI2_CLK_N MIPI_CSI2_CLK_P	1.8V		CSI differential clock inputs
CAM_MCK	O PP	1.8V CMOS	J6	GPIO1_IO15	1.8V		Master clock for camera (CCM_CLKO2)
I2C_CAM0_CK	O OD	1.8V CMOS	M19	NAND_DATA06	1.8V	PU 2.2k 1.8V	CAM0 DDC clock line (CPU GPIO3_IO13)
I2C_CAM0_DAT	I/O OD	1.8V CMOS	L19	NAND_DATA06	1.8V	PU 2.2k 1.8V	CAM0 DDC data line (CPU GPIO3_IO12)
I2C_CAM1_CK	O OD	1.8V CMOS	H1	SAI1_TXFS	1.8V	PU 2.2k 1.8V	CAM1 DDC clock line (CPU GPIO4_IO10)
I2C_CAM1_DAT	I/O OD	1.8V CMOS	E1	SAI1_TXC	1.8V	PU 2.2k 1.8V	CAM1 DDC data line (CPU GPIO4_IO11)
CAM0_PWR#	I/O OD	1.8V CMOS	T6	GPIO1_IO00	1.8V	PU 470k 1.8V	CAM0 Power Enable, active low. (CPU GPIO1_IO00)
CAM1_PWR#	I/O OD	1.8V CMOS	T7	GPIO1_IO01	1.8V	PU 470k 1.8V	CAM1 Power Enable, active low. (CPU GPIO1_IO01)
CAM0_RST#	I/O OD	1.8V CMOS	P4	GPIO1_IO03	1.8V	PU 470k 1.8V	CAM0 Reset, active low. (CPU GPIO1_IO03)
CAM1_RST#	I/O OD	1.8V CMOS	G21	NAND_CE1_B	1.8V	PU 470k 1.8V	CAM1 Reset, active low. (CPU GPIO3_IO02)

NOTE: CAM0 and CAM1 I²C drivers are implemented using bit-banged IO operation.

4.6 LVDS

LVDS is optionally available and is mutually exclusive with the DSI option.

LVDS channels 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 one Single-Link LVDS or one 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 4-6: LVDS Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	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_CK+ / DSI0_CLK+ LVDS0_CK- / 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

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
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
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	H2	SAI1_RXD2	1.8V	PD 10k	LCD0 panel power enable (CPU GPIO4_IO4)
LCD0_BKLT_EN	O PP	1.8V CMOS	E2	SAI1_TXD1	1.8V		LCD0 backlight enable (CPU GPIO4_IO13)
LCD0_BKLT_PWM	O PP	1.8V CMOS	F6	SPDIF_TX	1.8V	PD 10k	LCD0 backlight brightness control (PWM3_OUT)
LCD1_VDD_EN	O PP	1.8V CMOS	J2	SAI1_RXD3	1.8V	PD 10k	LCD1 panel power enable (CPU GPIO4_IO5)
LCD1_BKLT_EN	O PP	1.8V CMOS	C1	SAI1_TXD7	1.8V		LCD1 backlight enable (CPU GPIO4_IO19)
LCD1_BKLT_PWM	O PP	1.8V CMOS	G6	SPDIF_RX	1.8V	PD 10k	LCD1 backlight brightness control (PWM2_OUT)
I2C_LCD_CK	O PP	1.8V CMOS	G8	I2C3_SCL	1.8V	PU 2.2k 1.8V	I²C clock output for LVDS display use (I2C3)
I2C_LCD_DAT	I/O OD	1.8V CMOS	E9	I2C3_SDA	1.8V	PU 2.2k 1.8V	I²C data line for LVDS display use (I2C3)

NOTE: The DSI 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.

For support of customer specific LVDS displays (timing, resolution, etc) please contact your local support team.

4.7 HDMI/DP

An on-module MUX/DEMUX provides HDMI and DisplayPort signal switching. Both interfaces are available on the SMARC™ module and can be selected via software. The two options are mutually exclusive.

Supported interfaces:

- HDMI 2.0a up to 4096x2160
- DisplayPort 1.3 up to 4096x2160

Table 4-7: HDMI/DP Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
HDMI_D[0:2]+ HDMI_D[0:2]-	O	CML 3.3V	n.a.	n.a.	CML 3.3V spec.		HDMI differential pair signals
HDMI_CK+ HDMI_CK-	O	CML 3.3V	n.a.	n.a.	CML 3.3V spec.		HDMI differential clock
HDMI_HPD	I	3.3V CMOS	W2	HDMI_HPD	5.5V	PD 100k	HDMI Hot Plug Detect
HDMI_CTRL_CK	O OD	1.8V CMOS	R3	HDMI_DDC_SCL	1.8V	PU 100k 1.8V	HDMI DDC clock line
HDMI_CTRL_DAT	I/O OD	1.8V CMOS	P3	HDMI_DDC_SDA	1.8V	PU 100k 1.8V	HDMI DDC data line
DP0_LANE[0:3]+ DP0_LANE[0:3]-	O	CML 3.3V	n.a.	n.a.	CML 3.3V spec.		DisplayPort differential pair signals
DP0_AUX+ DP0_AUX -	O	CML 3.3V	n.a.	n.a.	CML 3.3V spec.		Auxiliary channel differential pair signals
DP0_HPD	I	3.3V CMOS	W2	HDMI_HPD	5.5V	PD 100k	DisplayPort Hot Plug Detect
DP0_AUX_SEL	I	1.8V CMOS	n.a.	n.a.	1.8V		Not used
DP1_AUX_SEL	I	1.8V CMOS	n.a.	n.a.	1.8V		Not used

4.8 SPI Bus

The i.MX8M 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
- 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 4-8: SPI Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
SPI0_DIN	I	1.8V CMOS	B5	ECSPI2_MISO	1.8V		Master Input Slave Output
SPI0_DO	O PP	1.8V CMOS	E5	ECSPI2_MOSI	1.8V		Master Output Slave Input
SPI0_CK	O PP	1.8V CMOS	C5	ECSPI2_SCLK	1.8V		Clock Output
SPI0_CS0#	O PP	1.8V CMOS	J22	NAND_DATA05	1.8V	PU 10k 1.8V	Chip-Select0, available for baseboard usage (GPIO3_IO11)
SPI0_CS1#	O PP	1.8V CMOS	K1	SAI1_RXC	1.8V	PU 10k 1.8V	Chip-Select1, available for baseboard usage (GPIO4_IO01)
SPI1_DIN	I	1.8V CMOS	B4	ECSPI1_MISO	1.8V		Master Input Slave Output
SPI1_DO	O PP	1.8V CMOS	A4	ECSPI1_MOSI	1.8V		Master Output Slave Input
SPI1_CK	O PP	1.8V CMOS	D5	ECSPI1_SCLK	1.8V		Clock Output
SPI1_CS0#	O PP	1.8V CMOS	L20	NAND_DATA04	1.8V	PU 10k 1.8V	Chip-Select 0, available for baseboard usage (GPIO3_IO10)
SPI1_CS1#	O PP	1.8V CMOS	J1	SAI1_RXD4	1.8V	PU 10k 1.8V	Chip-Select 1, available for baseboard usage (GPIO4_IO06)

NOTE: GPIOs are used for chip select pins SPI[0:1]_CS[0:1]#.

In case SPI1 is implemented, only SER[0:2] are available.

4.9 CAN

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

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

Table 4-9: CAN Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
CAN0_TX	O	1.8V CMOS	n.a.	n.a.	1.8V		CAN0 Transmit output
CAN1_TX	O	1.8V CMOS	n.a.	n.a.	1.8V		CAN1 Transmit output
CAN0_RX	I	1.8V CMOS	n.a.	n.a.	1.8V		CAN0 Receive input
CAN1_RX	I	1.8V CMOS	n.a.	n.a.	1.8V		CAN1 Receive input
CAN0_INT#	I	1.8V CMOS	H5	SAI2_MCLK	1.8V	PU 10K 1.8V	CAN0 multipurpose interrupt
CAN1_INT#	I	1.8V CMOS	F4	SAI3_RXC	1.8V	PU 10K 1.8V	CAN1 multipurpose interrupt

NOTE: Both CAN0 and CAN1 Controllers are accessible via SPI0 and SPI1. Refer to Chapter 6.2 for further information

4.10 GPIO

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

Table 4-10: GPIO Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
GPIO0 / CAM0_PWR#	I/O	1.8V CMOS	T6	GPIO1_IO00	1.8V	PU 470K 1.8V	CPU GPIO1_IO00
GPIO1 / CAM1_PWR#	I/O	1.8V CMOS	K22	NAND_RE_B	1.8V	PU 470K 1.8V	CPU GPIO3_IO17
GPIO2 / CAM0_RST#	I/O	1.8V CMOS	P4	GPIO1_IO03	1.8V	PU 470K 1.8V	CPU GPIO1_IO03
GPIO3 / CAM1_RST#	I/O	1.8V CMOS	G21	NAND_CE1_B	1.8V	PU 470K 1.8V	CPU GPIO3_IO02
GPIO4 / HDA_RST#	I/O	1.8V CMOS	P7	GPIO1_IO05	1.8V	PU 470K 1.8V	CPU GPIO1_IO05
GPIO5 / PWM_OUT	I/O	1.8V CMOS	T7	GPIO1_IO01	1.8V	PU 470K 1.8V	CPU GPIO1_IO01 / PWM1_OUT
GPIO6 / TACHIN	I/O	1.8V CMOS	N5	GPIO1_IO06	1.8V	PU 470K 1.8V	CPU GPIO1_IO06
GPIO7	I/O	1.8V CMOS	N6	GPIO1_IO07	1.8V	PU 470K 1.8V	CPU GPIO1_IO07
GPIO8	I/O	1.8V CMOS	N7	GPIO1_IO08	1.8V	PU 470K 1.8V	CPU GPIO1_IO08
GPIO9	I/O	1.8V CMOS	M6	GPIO1_IO09	1.8V	PU 470K 1.8V	CPU GPIO1_IO09
GPIO10	I/O	1.8V CMOS	F21	NAND_CE2_B	1.8V	PU 470K 1.8V	CPU GPIO3_IO03
GPIO11	I/O	1.8V CMOS	L6	GPIO1_IO011	1.8V	PU 470K 1.8V	CPU GPIO1_IO11

4.11 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 5.1)
- SD/SDIO 1.8 V (UHS-I) or 3.3 V operation with auto detection
- Normal-speed, high-speed, SDR50 and SDR104 mode

Table 4-11: SDIO Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
SDIO_D0	I/O	3.3V / 1V8 CMOS	N22	SD2_DATA0	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Data
SDIO_D1	I/O	3.3V / 1V8 CMOS	N21	SD2_DATA1	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Data
SDIO_D2	I/O	3.3V / 1V8 CMOS	P22	SD2_DATA2	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Data
SDIO_D3	I/O	3.3V / 1V8 CMOS	P21	SD2_DATA3	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Data
SDIO_CMD	I/O	3.3V / 1V8 CMOS	M22	SD2_CMD	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Command
SDIO_CK	O	3.3V / 1V8 CMOS	L22	SD2_CLK	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Clock
SDIO_PWR_EN	O PP	3.3V CMOS	K1	SAI1_RXC	3.3V / 1.8V	PD 1k	SDIO Controller Power enable (CPU GPIO4_IO01)
SDIO_CD#	I	3.3V / 1V8 CMOS	L21	SD2_CD_B	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Card Detect
SDIO_WP	I	3.3V / 1V8 CMOS	M21	SD2_WP	3.3V / 1.8V	PU 10k 3.3V / 1.8V	SDIO Controller Write Protect

4.12 UART

The i.MX8M offers several UART interfaces. Depending on module variant up to four separate interfaces 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 4-12: UART Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
SER0_TX	O	1.8V CMOS	A7	UART1_RXD	1.8V	PU 10k 1.8V	UART transmit data (also see Note 1 below!)
SER0_RX	I	1.8V CMOS	C7	UART1_TXD	1.8V	PU 10k 1.8V	UART receive data (also see Note 1 below!)

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
SER0_RTS#	O	1.8V CMOS	B7	UART3_RXD	1.8V	PU 10k 1.8V	UART handshake, ready to receive data
SER0_CTS#	I	1.8V CMOS	A6	UART3_TXD	1.8V	PU 10k 1.8V	UART handshake, ready to send data
SER1_TX	O	1.8V CMOS	D6	UART2_TXD	1.8V	PU 10k 1.8V	UART transmit data
SER1_RX	I	1.8V CMOS	B6	UART2_RXD	1.8V	PU 10k 1.8V	UART receive data
SER2_TX	O	1.8V CMOS	A4	ECSPI1_MOSI	1.8V	PU 10k 1.8V	UART transmit data
SER2_RX	I	1.8V CMOS	G22	EIM_D25	1.8V	PU 10k 1.8V	UART receive data
SER2_RTS#	O	1.8V CMOS	H21	EIM_D31	1.8V	PU 10k 1.8V	UART handshake, ready to receive data (See Note 2 below - not available if SPI1 is implemented)
SER2_CTS#	I	1.8V CMOS	D25	EIM_D23	1.8V	PU 10k 1.8V	UART handshake, ready to send data (See Note 2 below - not available if SPI1 is implemented)
SER3_TX	O	1.8V CMOS	W5	KEY_COL0	1.8V	PU 10k 1.8V	UART transmit data (See Note 2 below - not available if SPI1 is implemented)
SER3_RX	I	1.8V CMOS	V6	KEQ_ROW0	1.8V	PU 10k 1.8V	UART receive data (See Note 2 below - not available if SPI1 is implemented)

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 5.6.2. So for access to the serial Console of U-Boot / Linux, either one connection can be used as appropriate.

Note 2: In case SPI1 is implemented on the module, only SER0 (4-Wire), SER1 (2-Wire) and SER2 (2-Wire) are available. The unused pins may be left unconnected.

4.13 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, HDMI, LCD/LVDS and System Management interfaces.

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

- Compatibility 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 4-13: I²C Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
I2C_GP_CK	O OD	1.8V CMOS	G7	I2C2_SCL	1.8V	PU 2.2k 1.8V	General Purpose SMB clock output
I2C_GP_DAT	I/O OD	1.8V CMOS	F7	I2C2_SDA	1.8V	PU 2.2k 1.8V	General Purpose SMB data I/O line
SMB_ALERT_1V8#	I OD	1.8V CMOS	K19	NAND_RE_B	1.8V	PU 2.2k 1.8V	(CPU GPIO3_IO15)

4.14 Watchdog

Table 4-14: Watchdog Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
WDT_TIME_OUT#	O PP	1.8V CMOS	R4	GPIO1_IO02	1.8V		Watch-Dog-Timer Output from the SOC.

4.15 System Management

Table 4-15: System Management Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	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.
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	K4	SAI5_MCLK	1.8V	PD 10k	General purpose reset for carrier board. (CPU GPIO3_IO25)
RESET_IN#	I OD	1.8V CMOS	n.a.	n.a.	1.8V	PU 10k Vin	Reset input from Carrier board. Carrier drives low to force a Module reset, floats the line otherwise. Pulled up on module. Driven by OD part on carrier.
POWER_BTN#	I	1V8 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	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
CHARGING#	I	1V8 CMOS			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_IO14)
CHARGER_PRSNT#	I	1V8 CMOS			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 GPIO3_IO16)
SLEEP#	I OD	1.8V CMOS	M5	SAI5_RXD0	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 GPIO3_IO21)
LID#	I OD	1.8V CMOS	L4	SAI5_RXD1	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 GPIO3_IO22)
BATLOW#	I OD	1.8V CMOS	L5	SAI5_RXC	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_IO20)
I2C_PM_CK	O OD	1.8V CMOS	E7	I2C1_SCL	1.8V	PU 2.2k 1.8V	Power management I²C clock output
I2C_PM_DAT	I/O OD	1.8V CMOS	E8	I2C1_SDA	1.8V	PU 2.2k 1.8V	Power management I²C data I/O

4.16 Boot-Options

Table 4-16: Boot Options Control Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
BOOT_SEL0#	I OD	1.8V CMOS	M4	SAI5_RXD2	1.8V	PU 10k 1.8V	Input straps determine the module boot device. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO3_IO23)
BOOT_SEL1#	I OD	1.8V CMOS	K5	SAI5_RXD3	1.8V	PU 10k 1.8V	Input straps determine the module boot device. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO3_IO24)
BOOT_SEL2#	I OD	1.8V CMOS	D3	SAI3_MCLK	1.8V	PU 10k 1.8V	Input straps determine the module boot device. Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO5_IO02)
FORCE_RECOV#	I OD	1.8V CMOS	N4	SAI5_RXFS	1.8V	PU 10k 1.8V always on	Pulled up on Module. Driven by OD part on Carrier. (CPU GPIO3_IO19)
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.

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.

TEST# signal is checked for primary boot device selection (primary boot device selection defines the device in which ROM code expects SPL and U-BOOT code). With inactive test mode the module loads the boot loader from eMMC and checks the BOOT_SEL signals for operating system location.

The i.MX8M SoC supports recovery devices. If primary boot device fails, the module will always try to boot from SD-Card. This means in detail, that if the TEST# signal is not pulled low and eMMC does not contain valid boot code, the carrier SD-Card is booted. This is unintended behaviour (but cannot be changed as it is configured by the ROM code in the CPU). When a system is booted in this way, HDMI and DP display interfaces are not operational, and the board is in an invalid condition.

If TEST# signal is pulled low at carrier, the module boots from carrier SD-Card directly.

Table 4-17: 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 eSPI with CS0# (not supported)
3	GND	Float	Float	Carrier SPI with CS0# (not supported)
4	Float	GND	GND	Module SD Card (not supported)
5	Float	GND	Float	Remote boot (LAN0)
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 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-imx8m.html.

For details regarding the i.MX8M CPU please refer to the NXP website (see[4]). For order information please contact Avnet Embedded /MSC.

Figure 5-1: CPU Options

i.MX 8M FAMILY—DIFFERENTIATED FEATURES

Feature	i.MX 8M Dual/i.MX 8M Quad	i.MX 8M QuadLite
ARM® Core	2 or 4 x Cortex-A53	4 x Cortex-A53
ARM Core	1 x Cortex-M4F	1 x Cortex-M4F
Audio	20 channels in/out; 32-bit up to 384 KHz, with DSD512 support	
GPU	GC7000Lite	GC7000Lite
Video Acceleration	4Kp60, h.265 and VP9	
Camera	2 x MIPI-CSI	2 x MIPI-CSI

*2-lane PCIe can act as 2 x 1-lane PCIe

5.2 Start-Up and Power-Down 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# is high 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 pressing the Power button shorter than 8 seconds when powered, the module will initiate a shutdown and go off after ~5 seconds.
- 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 released.

5.3 Memory

5.3.1 SDRAM

The DDR Controller supports 32/16-bit LPDDR4-3200.

MSC SM2S-IMX8M SMARC™ modules use one physical rank with up to 4GByte SDRAM.

Table 5-1: Available SDRAM options

CPU	Bus Width	Memory Size	Memory Organisation
i.MX8M Dual, Quad and QuadLite	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 interface mode.

Table 5-2: 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			
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 I2C2 bus at address 0x50.

NOTE: **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 SMARC™ module offers an optional Trusted Platform Module (TPM) from Infineon.

The Infineon TPM SLB9671 2.0 is connected to the I2C4 bus at address 0x20.

NOTE: Current revision of the module integrates the Infineon TPM SLB9645 1.2.

5.5 WiFi Module

The i.MX8M SMARC™ module offers an optional on-board low power WiFi 2x2 MU-MIMO and Bluetooth 5.0. The SPB228 provides up to 867Mbps data rate with 802.11ac on 80MHz channel and dual spatial streams.

The module is equipped with dual RF micro connectors (U.FL receptacles)

5.6 Debug Options

5.6.1 LEDs

The module features three LEDs which display the module status.

The Software-LED (yellow colour) is connected to one pin of the i.MX8M CPU. ThisLED is switched off when boot loader boot process has finished successfully. Other usage options are possible, .

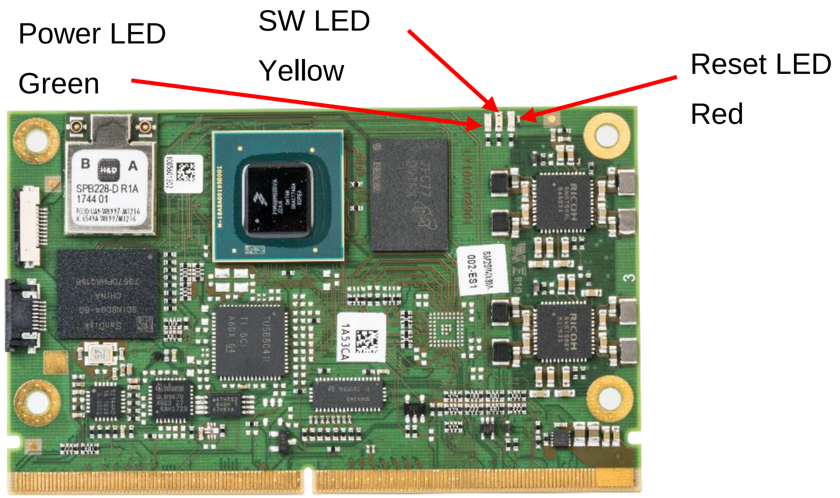
Table 5-3: SW LED Signal Description

Signal	Pin Type	Signal Level	Pin on i.MX8M	Pin name on i.MX8M	Power Tolerance	PU/PD	Description
SW_LED#	I/O	3.3V CMOS	B3	SAI1_TXD6	3.3V		Low active signal for LED control (CPU GPIO4_IO18)

The Power LED (green colour) is on, if module power is completely ok.

The Reset LED (red colour) is on, if module is held in reset.

Figure 5-2: Module top side with debug LEDs



5.6.2 Debug Connector

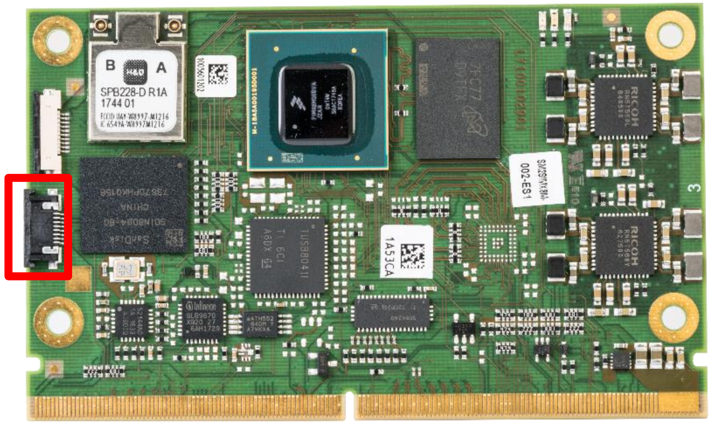
Access to the Debug UART port is possible via an 8pin FFC connector.

Note: This connector may not be populated on all board variants.

Figure 5-3: Module top side with debug UART FFC connectors marked in red

Pinout:

1: Ground
2: Debug_2
3: Debug_1
4: Debug_0
5: UART_TXD
6: UART_RXD
7: RESET_IN#
8: VCC_3V3

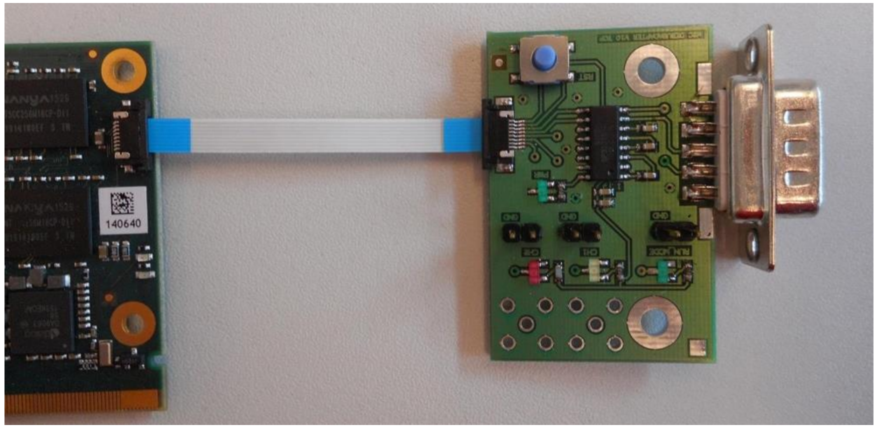


Serial signals on the Debug connector are at 3.3V TTL level.

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.

Figure 5-4: Module top side with MSC UART debug adapter



MSC Debug Adapter Order Part No. 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 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 4.12 .

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.

5.6.3 JTAG Connector

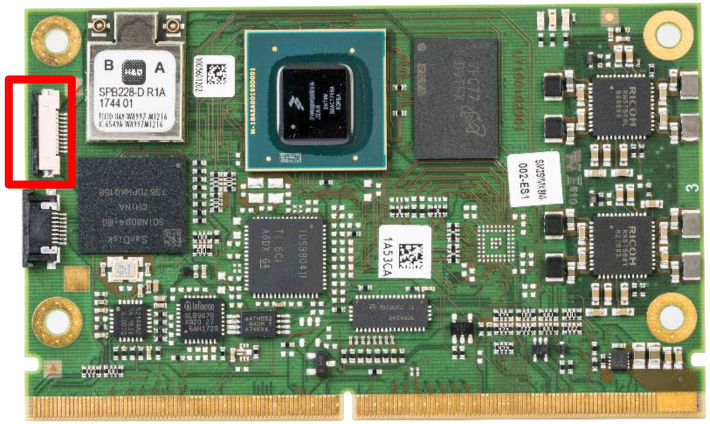
JTAG access to the i.MX 8M CPU is possible via a 10pin FFC connector. The JTAG Chain only contains the CPU itself, so all suitable JTAG debuggers should work with their default configuration for the respective CPU.

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

Figure 5-5: Module top side with JTAG FFC connectors marked in red

Pinout:

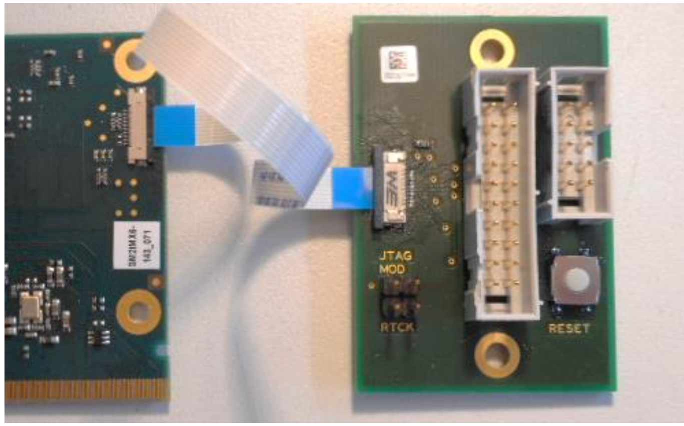
1:	Ground
2:	RESET_IN#
3:	NC
4:	CPU_JTAG_MOD
5:	JTAG_TCK
6:	JTAG_TMS
7:	JTAG_TDO
8:	JTAG_TDI
9:	JTAG_TRST#
10:	VCC_3V3



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 5-6: Module bottom side with MSC JTAG debug adapter



MSC JTAG-Adaptor FFC 10pol
Order Part No. 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 6-1: I²C Interfaces Overview

CPU Interface	Device	SMARC Connector	7 bit Address
I2C1		I2C_PM	
I2C2	EEPROM	I2C_GP	0x50
I2C3		I2C_LCD	
I2C4	TPM		0x20
	PMIC1		0x30
	PMIC2		0x31
	Temp. Sensor		0x71
	RTC		0x32
	PCIe Clock Generator		0x6B
	DSI to LVDS Bridge		0x2D
GPIO based		CAM0	
GPIO based		CAM1	
HDMI		HDMI_CTRL	

NOTE: CAM0 and CAM1 are GPIO based (bit-banged). Data transfer rates up to 100kbips are possible.

6.2 SPI Devices

Table 6-2: SPI Interfaces Overview

CPU Interface	Chip Select	CPU Pin	CPU Pin Name	Device	SMARC Connector	Description
ECSPI1	CS0	D4	ECSPI1_SS0	CAN1 Controller		Dedicated chip select for CAN1 Controller
	CS0	L20	NAND_DATA04		SP1_CS0#	GPIO based chip select (CPU GPIO4_IO10)
	CS1	J1	SAI1_RXD4		SP1_CS1#	GPIO based chip select (CPU GPIO4_IO06)
ECSPI2	CS0	A5	ECSPI2_SS0	CAN0 Controller		Dedicated chip select for CAN0 Controller
	CS0	J22	NAND_DATA05		SPI0_CS0#	GPIO based chip select (CPU GPIO3_IO11)
	CS1	K1	SAI1_RXC		SPI0_CS1#	GPIO based chip select (CPU GPIO4_IO01)
QSPI_A	CS0	H19	NAND_CE0_B	QSPI NOR Flash		Dedicated chip select for QSPI NOR Flash

NOTE: The dedicated chip select lines of ECSPI1 and ECSPI2 are reserved for both CAN controllers. GPIOs are used for chip select pins SPI[0:1]_CS[0:1]# on the SMARC connector.

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).

7.2 The current MSC-LDK and the msc-sm2s-imx8m BSP base on NXP's release L4.14.98-2.0.0_ga.MSC-LDK (Yocto)

This chapter describes how to build an image for an msc-sm2s-imx8m module by using the MSC-LDK.

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

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).

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

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~$ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/waldemar/.ssh/id_rsa):
Created directory '/home/waldemar/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/waldemar/.ssh/id_rsa.
Your public key has been saved in /home/waldemar/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:lUnkxE/cUg7ZmCv2zI9HpVq1YhNhc3ZrgFMqboOmGRI waldemar@Workstation3
The key's randomart image is:
+---[RSA 2048]----+
|         o+.+O.  |
|         +.=OBoo.|
|   E     .=+ooB o|
|    .   o.+ o. oo|
|   . . oS= =  o+.|
|    . = . . +++. |
|     o      .*o  |
|            o o  |
|             .   |
+----[SHA256]-----+
waldemar@Workstation3~$
```

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.

Configuring HTTP proxy

Some source files will be downloaded from HTTP and FTP servers. If a proxy must be used, theseenvironment 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 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".

The “classic” method

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-v.1.5.0

cd msc-ldk-v.1.5.0
```

Note: The subsequent examples have been recorded with earlier release states of the BSP, but the same methods and procedures still apply respectively. Just the readings of "v1.x.x" will change.

Example:

```
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk$git clone ssh://gitolite@msc-git02.msc-ge.com:9418/m
sc_ol99/msc-ldk --branch v1.4.0 msc-ldk-v1.4.0
Cloning into 'msc-ldk-v1.4.0'...
remote: Counting objects: 5337, done.
remote: Compressing objects: 100% (1893/1893), done.
remote: Total 5337 (delta 3620), reused 4834 (delta 3283)
Receiving objects: 100% (5337/5337), 677.67 KiB | 0 bytes/s, done.
Resolving deltas: 100% (3620/3620), done.
Checking connectivity... done.
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk$cd msc-ldk-v1.4.0/
```

Figure 7-2: Clone base MSC-LDK repo

Your current directory now contains the following sub directories:

```
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0 (v1.4.0)$tree -d
.
├── scripts
└── template

2 directories
```

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

Step 2: Create build directory

To create your build directory run the following command:

```
./setup.sh --bsp=0102901
```

where the "0102901" is the internal project number of the MSC SM2S-IMX8M module.

Example:

```
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0 (v1.4.0)$./setup.sh --bsp=0102901
Cloning libMscBoostPython (version: v1.1.2)
Executing 'git clone -q ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0000/libmscboostpython libmscboostpython.git
'
NOTICE: libMscBoostPython is at <Branch: master, TAG: v1.2.1>
NOTICE: MSC-LDK git server: 'ssh://gitolite@msc-git02.msc-ge.com:9418/'
NOTICE: MSC-LDK root: /data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0
NOTICE: MSC-LDK is based on Yocto branch: sumo, MSC-LDK is at <Branch: v1.4.0 [LOL99_20190718_V1_4_0-9-gd663eb7]>
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0102901/msc-ldk-bsp-recipes'
NOTICE: MSC-LDK Configuration: BSP=0102901
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-qt5'
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_ol99/meta-msc-ldk-core-recipes'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/meta-msc-ldk-core'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/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-qt5-extra'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/meta-msc-arm-extensions'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/meta-spb228-usb'
NOTICE: Created '/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0/build/0102901/conf/local.conf'
NOTICE: Created '/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0/build/0102901/conf/bblayers.conf'
INFO: Building an Azure enabled image requires git v2.11 or later (installed: git v2.7.4)
INFO: You can now cd to /data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0/build/0102901 and run 'make' or './build.s
h <image-name>'
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0 (v1.4.0)$
```

Figure 7-4: Create build directory

Your current directory now looks like this:

```
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0 (v1.4.0)$tree -d -L 1
.
├── build
├── scripts
├── sources
└── template

4 directories
```

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

Step 3: Enter build directory

To enter the build directory execute:

```
cd build/0102901
```

Example:

```
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0 (v1.4.0)$cd build/0102901/
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0/build/0102901 (v1.4.0)$
```

Figure 7-6: Enter build directory

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 [12].

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 v1.5.0
cd docker-msc-ldk
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~/msc-ldk$git clone ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/docker-msc-ldk
Cloning into 'docker-msc-ldk'...
remote: Counting objects: 40, done.
remote: Compressing objects: 100% (31/31), done.
remote: Total 40 (delta 16), reused 0 (delta 0)
Receiving objects: 100% (40/40), 6.64 KiB | 6.64 MiB/s, done.
Resolving deltas: 100% (16/16), done.
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk$cd docker-msc-ldk/
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/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)
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$mkdir src
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$mkdir -p rootfs/home/.ssh
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$cp ~/.ssh/id_rsa rootfs/home/.ssh
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$cp ~/.ssh/id_rsa.pub rootfs/home/.ssh
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$docker build -t=msc-ldk .
Sending build context to Docker daemon  75.78kB
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
 ---> Using cache
 ---> ac726a17c5d8
Step 5/29 : ARG USER=user
```

Figure 7-7: Create docker container for MSC-LDK. Part 1

```
Removing intermediate container f685d7052702
 ---> 1b4a55d8578e
Step 28/29 : RUN echo "if [ -e /src/.bashrc ]; then source /src/.bashrc; fi" >> ${HOME}/.bashrc
 ---> Running in 74957865afe8
Removing intermediate container 74957865afe8
 ---> 18132373d182
Step 29/29 : WORKDIR /src
 ---> Running in 0603826e8f35
Removing intermediate container 0603826e8f35
 ---> e32180a747cc
Successfully built e32180a747cc
Successfully tagged msc-ldk:latest
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$rm -rf rootfs/home/.ssh
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
```

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

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

Execute (on Debian, Ubuntu, Mint, etc.):

```
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~/msc-ldk/docker-msc-ldk ((v1.3.0))$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
user@docker:/src$
user@docker:/src$
```

Figure 7-9: Start and enter the MSC-LDK container

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

Execute:

```
git clone ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/msc-ldk --branch v1.5.0
        msc-ldk-v.1.5.0

cd msc-ldk-v.1.5.0
```

Example: (Note: Example Screenshots still show v1.4.0)

```
user@docker:/src$
user@docker:/src$ git clone ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/msc-ldk --branch v1.4.0 msc-ldk-v.1.4.0
Cloning into 'msc-ldk-v.1.4.0'...
remote: Counting objects: 5349, done.
remote: Compressing objects: 100% (1905/1905), done.
remote: Total 5349 (delta 3629), reused 4834 (delta 3283)
Receiving objects: 100% (5349/5349), 679.21 KiB | 0 bytes/s, done.
Resolving deltas: 100% (3629/3629), done.
Checking connectivity... done.
user@docker:/src$
user@docker:/src$ cd msc-ldk-v.1.4.0/
user@docker:/src/msc-ldk-v.1.4.0$
```

Figure 7-10: Clone and enter the base MSC-LDK repo

Step 4: Create build directory

Execute:

./setup.sh --bsp=0102901

Example:

```
user@docker:/src/msc-ldk-v.1.4.0$
user@docker:/src/msc-ldk-v.1.4.0$ ./setup.sh --bsp=0102901
Cloning libMscBoostPython (version: v1.1.2)
Executing 'git clone -q ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0000/libmscboostpython libmscboostpython.git'
INFO: Added host 'ftp4.ebv.com' to /home/user/.ssh/known_hosts
NOTICE: libMscBoostPython is at <Branch: master, TAG: v1.2.1>
NOTICE: MSC-LDK git server: 'ssh://gitolite@msc-git02.msc-ge.com:9418/'
NOTICE: MSC-LDK root: /src/msc-ldk-v.1.4.0
NOTICE: MSC-LDK is based on Yocto branch: sumo, MSC-LDK is at <Branch: v1.4.0 [LOL99_20190718_V1_4_0-9-gd663eb7]>
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_0102901/msc-ldk-bsp-recipes'
NOTICE: MSC-LDK Configuration: BSP=0102901
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-qt5'
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_ol99/meta-msc-ldk-core-recipes'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/meta-msc-ldk-core'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/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-qt5-extra'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/meta-msc-arm-extensions'
NOTICE: Installing repository 'ssh://gitolite@msc-git02.msc-ge.com:9418/msc_ol99/meta-spb228-usb'
NOTICE: Created '/src/msc-ldk-v.1.4.0/build/0102901/conf/local.conf'
NOTICE: Created '/src/msc-ldk-v.1.4.0/build/0102901/conf/bblayers.conf'
INFO: Building an Azure enabled image requires git v2.11 or later (installed: git v2.7.4)
INFO: You can now cd to /src/msc-ldk-v.1.4.0/build/0102901 and run 'make' or './build.sh <image-name>'
user@docker:/src/msc-ldk-v.1.4.0$
```

Figure 7-11: Create build directory

Step 5: Enter build directory

Execute:

cd build/0102901

Example:

```
user@docker:/src/msc-ldk-v.1.4.0$
user@docker:/src/msc-ldk-v.1.4.0$ cd build/0102901/
user@docker:/src/msc-ldk-v.1.4.0/build/0102901$
user@docker:/src/msc-ldk-v.1.4.0/build/0102901$
```

Figure 7-12: Enter build directory

Leave the MSC-LDK container.

Execute:

exit

Example:

```
user@docker:/src$
user@docker:/src$ exit
exit
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
```

Figure 7-13: Leave the MSC-LDK container

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~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$docker container start msc-ldk
msc-ldk
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$docker container ls
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
39e5798ff7e1        msc-ldk             "/bin/bash"         4 minutes ago       Up 12 seconds                           msc-ldk
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$docker container exec -ti msc-ldk /bin/bash
user@docker:/src$
user@docker:/src$
user@docker:/src$
```

Figure 7-14: Re-start and re-enter the MSC-LDK container

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~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$docker stop msc-ldk
msc-ldk
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$docker rm msc-ldk
msc-ldk
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
waldemar@waldemar-HP-Z6-G4-Workstation~/msc-ldk/docker-msc-ldk ((v1.3.0))$
```

Figure 7-15: Stop the MSC-LDK container and release its resources.

7.2.4 Generate images

Choosing an MSC-LDK image

The MSC-LDK provides different images for the sms2-imx8m module. The following table lists all currently available images, their contents and sizes.

Table 7-1: Available images

Image name	Content	Approx size
msc-image-minimal	A small image that only allows a device to boot. Contains MSC features.	352 MiB
msc-image-base	A small Wayland GUI image that fully supports the target device hardware. Contains MSC features.	440 MiB
msc-image-qt5	A Wayland GUI and opensource Qt 5 image that fully supports the target device hardware. Contains MSC features.	2,0 GiB
Yocto core images:		
core-image-minimal	A small image that only allows a device to boot.	216 MiB
core-image-base	A console-only image that fully supports the target device hardware.	272 MiB
Community images		
fsl-image-validation-imx	An i.MX image with a GUI without any Qt content.	1,7 GiB
fsl-image-machine-test	An FSL Community i.MX core image with console environment and no GUI interface	1,6 GiB

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.)
- software package name (e.g. u-boot-imx, linux-imx, memtester, etc.)

Example:

```
waldemar@Workstation3/data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0/build/0102901 (v1.4.0)$make msc-image-base
NOTE: Your conf/bblayers.conf has been automatically updated.
WARNING: Host distribution "linuxmint-18" has not been validated with this version of the build system; you may p
ossibly experience unexpected failures. It is recommended that you use a tested distribution.
WARNING: /data/Develop/repos/msc/msc-ldk/msc-ldk-v1.4.0/sources/yocto.git/meta/recipes-kernel/linux-firmware/linu
x-firmware_git.bb: LICENSE_linux-firmware-bcm4359-pcie includes licenses (Firmware-cypress) that are not listed i
n LICENSE
Parsing recipes: 100% |##########################################################################| Time: 0:00:59
Parsing of 2976 .bb files complete (0 cached, 2976 parsed). 3907 targets, 448 skipped, 2 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION           = "1.38.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "linuxmint-18"
TARGET_SYS           = "aarch64-poky-linux"
MACHINE              = "sm2s-imx8m-qc"
DISTRO               = "poky"
DISTRO_VERSION       = "2.5"
TUNE_FEATURES        = "aarch64"
TARGET_FPU           = ""
meta-poky
meta-yocto-bsp       = "sumo-msc:5b9648484d800c5faeec6619203dad2e0ff400b9"
meta                 = "master:be4066817326273a8b69750ddaaf72790af19a3d"
meta                 = "sumo-msc:5b9648484d800c5faeec6619203dad2e0ff400b9"
meta-oe
meta-networking
meta-python          = "sumo-msc:f08c9d7544c464d09325a16e274d57ddbed4b069"
meta-qt5.git         = "sumo-msc:74451d7524e0d470a5b17791e98eea54b18a3885"
meta
meta-integrity
meta-signing-key
meta-tpm
meta-tpm2            = "sumo-msc:860d0ff84270987e6e310f8983179cd7ec16949c"
meta-msc-ldk-core-recipes.git = "v1.4.0:a5ec49a5056002bf39bf7f2603cf572807e75ec2"
meta-msc-ldk-core.git = "v1.4.0:02e1091442b673977518a61abbc28643dd64e238"
meta-msc-ldk-mscio.git = "v1.4.0:a42539cdf2775fe799a3478816b31c48939d0d10"
meta-freescale.git   = "sumo-4.14.98_2.0.0_ga:cc883ea06f83b014c50e62ba0227da4a20915904"
meta-freescale-distro.git = "sumo-4.14.98_2.0.0_ga:f7e2216e93aff14ac32728a13637a48df436b7f4"
meta-bsp
meta-sdk             = "sumo-4.14.98-2.0.0_ga:ec7b004b44ba03277f60c2b675c34fa835c5610e"
meta-browser.git     = "sumo-4.14.98_2.0.0_ga:75640e14e325479c076b6272b646be7a239c18aa"
meta-qt5-extra.git   = "sumo-msc:eb5a6f603c351f80f40ca2dd817737aff9e8d7df"
meta-msc-arm-extensions.git = "v1.4.0:4158ad87727053d08b1396805f3e68f42ade15b8"
meta-spb228-usb.git  = "sumo-msc:4fc468f8daa1e923488d7c3bf838451d96f8836c"
meta-multimedia      = "sumo-msc:f08c9d7544c464d09325a16e274d57ddbed4b069"

NOTE: Fetching uninative binary shim from http://downloads.yoctoproject.org/releases/uninative/1.9/x86_64-natives
dk-libc.tar.bz2;sha256sum=c26622a1f27dbf5b25de986b11584b5c5b2f322d9eb367f705a744f58a5561ec
Initialising tasks: 100% |#######################################################################| Time: 0:00:02
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
```

Figure 7-16: Building msc-image-base

For more details and further information see also [10], Chapter 5 *“Image build”* .

Some scripts of the recipes use an 'echo -e <somewhat>' command. bitbake calls the buildscripts with /bin/sh as shell. If your hostsystem 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
```

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 Figure 7-16: Building msc-image-base)

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_imx8m_qc-poky-linux/msc-image-base/1.0-r0/rootfs/etc/

```
root@sm2s-imx8m-qc:~# cat /etc/version_layer
MSC-LDK LOL99_20190718_V1_4_0-9-gd663eb7 built on Thu Oct  3 09:00:58 UTC 2019 by waldemar@Workstation3
Poky (Yocto Project Reference Distro) 2.5 \n \l

LAYER meta-browser=LOL99_20190718_V1_4_0
LAYER meta-freescale-distro=LOL99_20190718_V1_4_0
LAYER meta-freescale=LOL99_20190718_V1_4_0-4-gcc883ea
LAYER meta-fsl-bsp-release=LOL99_20190718_V1_4_0
LAYER meta-msc-arm-extensions=LOL99_20190718_V1_4_0-1-g4158ad8
LAYER meta-msc-ldk-core=LOL99_20190718_V1_4_0
LAYER meta-msc-ldk-core-recipes=LOL99_20190718_V1_4_0-24-ga5ec49a
LAYER meta-msc-ldk-mscio=LOL99_20190718_V1_4_0-10-ga42539c
LAYER meta-openembedded=LOL99_20190718_V1_4_0
LAYER meta-qt5-extra=LOL99_20190718_V1_4_0
LAYER meta-qt5=LOL99_20190718_V1_4_0
LAYER meta-secure-core=LOL99_20190718_V1_4_0
LAYER meta-spb228-usb=LOL99_20190718_V1_4_0
LAYER msc-ldk-bsp-recipes=LOL99_20190718_V1_4_0-17-gbe40668
LAYER yocto=LOL99_20190718_V1_4_0
SETUP --bsp=0102901
root@sm2s-imx8m-qc:~#
```

Figure 7-17: 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 [10], Chapter 6, “Image Deployment”

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.

Forced booting SPL/U-Boot from carrier SD card

The i.MX 8M 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.

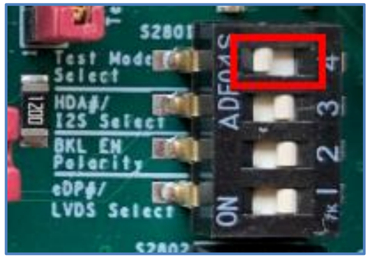


Figure 7-18: SPL boot selector on EP1 carrier board (S2801). Forced carrier SD card boot mode

Example:

```
U-Boot SPL 2018.03-imx_v2018.03_4.14.98_2.0.0_ga+gdadf96b (Oct 03 2019 - 08:52:00 +0000)
------------------------------
company .......... msc
form factor ...... sm2s
platform ......... imx8m
processor ........ qc
feature .......... 13N0600I
serial ........... 1008305407
revision (MES) ... A0
boot count ....... 176
------------------------------
DRAM PHY: 1D image training for 3200MTS ... passed
DRAM PHY: 1D image training for  400MTS ... passed
DRAM PHY: 1D image training for  100MTS ... passed
DRAM PHY: 2D image training for 3200MTS ... passed
DDRINFO: ddrphy calibration done
DDRINFO: ddrmix config done
Normal Boot
Trying to boot from MMC2


U-Boot 2018.03-imx_v2018.03_4.14.98_2.0.0_ga+gdadf96b (Oct 03 2019 - 08:52:00 +0000)

CPU:   Freescale i.MX8MQ rev2.1 1300 MHz (running at 800 MHz)
```

Figure 7-19: Forced SPL boot from carrier SD card

Booting SPL/U-Boot from module eMMC flash

The i.MX 8M 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.)

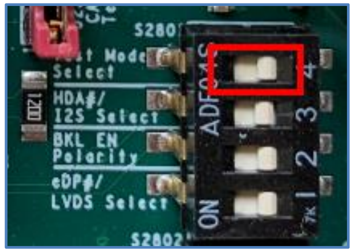


**Figure 7-20: SPL boot selector on EP1 carrier board (S2801).
eMMC flash boot mode (default)**

Example:

```
U-Boot SPL 2018.03-00024-gb45d827-dirty (Sep 03 2019 - 16:01:32 +0200)
------------------------------
company .......... msc
form factor ...... sm2s
platform ......... imx8m
processor ........ qc
feature .......... 13N0600I
serial ........... 1008305407
revision (MES) ... A0
boot count ....... 177
------------------------------
DRAM PHY: training for 3200MTS ... passed
DRAM PHY: training for  667MTS ... passed
DRAM PHY: training for 3200MTS ... passed
DDRINFO:ddrphy calibration done
DDRINFO: ddrmix config done
Normal Boot
Trying to boot from MMC1


U-Boot 2018.03-00024-gb45d827-dirty (Sep 03 2019 - 16:01:32 +0200)

CPU:   Freescale i.MX8MQ rev2.1 1300 MHz (running at 800 MHz)
```

Figure 7-21: SPL boot from module eMMC flash

Booting SPL from USB

Not supported yet (see section 8.1.3)

7.3.2 Booting OS

According to [1], chapter 4.17 “boot select” the BOOT_SEL [0:2] pins are used to select one of the following OS boot schema.

Booting OS from carrier SD card

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).

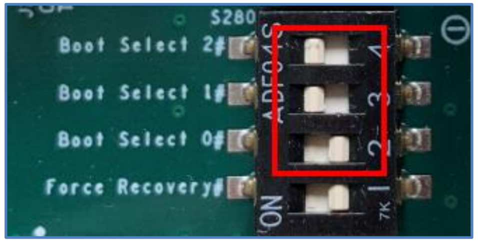


Figure 7-22: OS boot selector on EP1 carrier board (S2802). Carrier SD card boot mode

Example:

```
Normal Boot
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
** Unable to read file boot.scr **
Loading image <Image> from 1:1
23228928 bytes read in 999 ms (22.2 MiB/s)
Loading fdt <msc-sm2s-imx8m-qc-13N0600I.dtb> from 1:1
46690 bytes read in 21 ms (2.1 MiB/s)
## Flattened Device Tree blob at 43000000
   Booting using the fdt blob at 0x43000000
   Using Device Tree in place at 0000000043000000, end 000000004300e661

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.14.98-imx_4.14.98_2.0.0_ga+gdd03928 (oe-user@oe-host) (gcc version 7.3.0 (GCC)) #1 9
[    0.000000] Boot CPU: AArch64 Processor [410fd034]
[    0.000000] Machine model: MSC-SM2S-IMX8M
[    0.000000] earlycon: ec_imx6q0 at MMIO 0x0000000030860000 (options '115200')
[    0.000000] bootconsole [ec_imx6q0] enabled
```

Figure 7-23: OS boot from carrier SD card

Booting OS from module eMMC flash

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).

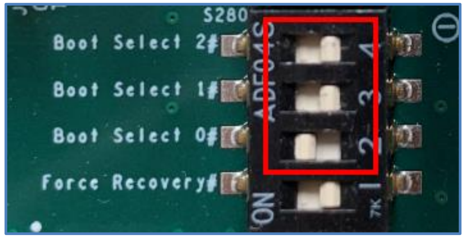


**Figure 7-24: OS boot selector on EP1 carrier board (S2802).
Module eMMC flash boot mode.**

Example:

```
Normal Boot
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
** Unable to read file boot.scr **
Loading image <Image> from 0:1
23228928 bytes read in 285 ms (77.7 MiB/s)
Loading fdt <msc-sm2s-imx8m-qc-13N0600I.dtb> from 0:1
46690 bytes read in 11 ms (4 MiB/s)
## Flattened Device Tree blob at 43000000
   Booting using the fdt blob at 0x43000000
   Using Device Tree in place at 0000000043000000, end 000000004300e661

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.14.98-imx_4.14.98_2.0.0_ga+g0e32dc0 (oe-user@oe-host) (gcc version 7.3.0 (GCC)) #1 9
[    0.000000] Boot CPU: AArch64 Processor [410fd034]
[    0.000000] Machine model: MSC-SM2S-IMX8M
[    0.000000] earlycon: ec_imx6q0 at MMIO 0x0000000030860000 (options '115200')
```

Figure 7-25: OS boot from module eMMC flash

Booting OS from Net (Ethernet)

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.

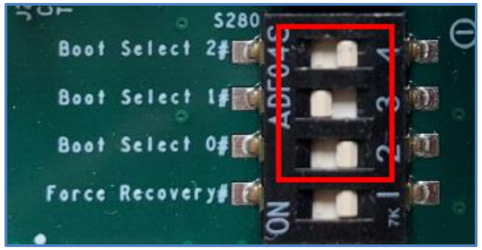


Figure 7-26: OS boot selector on EP1 carrier board (S2802). Network/Ethernet boot mode.

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/0102901/rootfs
u-boot=> setenv serverip 172.30.206.183
u-boot=> saveenv
Saving Environment to MMC... Writing to MMC(1)... OK
u-boot=>
u-boot=>
```

Figure 7-27: Setting U-Boot environment for net boot.

Then continue system boot with:

boot

or simply reboot your system.

Example:

```
u-boot=>
u-boot=> boot
Boardinfo: OK, complete.
Using network/ethernet as boot device ...
BOOTP broadcast 1
DHCP client bound to address 172.30.206.101 (39 ms)
Using ethernet@30be0000 device
TFTP from server 172.30.206.183; our IP address is 172.30.206.101
Filename 'Image'.
Load address: 0x40480000
Loading: #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #######################
         1.5 MiB/s
done
Bytes transferred = 23228928 (1627200 hex)
BOOTP broadcast 1
DHCP client bound to address 172.30.206.101 (37 ms)
Using ethernet@30be0000 device
TFTP from server 172.30.206.183; our IP address is 172.30.206.101
Filename 'msc-sm2s-imx8m-qc-13N0600I.dtb'.
Load address: 0x43000000
Loading: ####
         1.1 MiB/s
done
Bytes transferred = 46690 (b662 hex)
## Flattened Device Tree blob at 43000000
   Booting using the fdt blob at 0x43000000
   Using Device Tree in place at 0000000043000000, end 000000004300e661

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.14.98-imx_4.14.98_2.0.0_ga+gdd03928d7327 (oe-user@oe-host) (gcc version 7.3.0 (GCC)9
[    0.000000] Boot CPU: AArch64 Processor [410fd034]
```

Figure 7-28: OS boot from network

Booting OS from USB

Not support yet (see 8.1.3)

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).

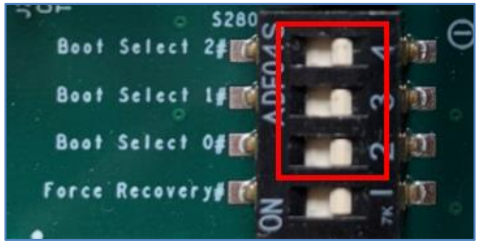


Figure 7-29: OS boot selector on EP1 carrier board (S2802).
USB boot mode.

Example:

```
Normal Boot
Hit any key to stop autoboot:  0
Boardinfo: OK, complete.
Using USB as boot device ...
USB boot not supported yet!
u-boot=>
u-boot=>
u-boot=>
```

Figure 7-30: OS boot from USB

Available device tree blobs (DTBs)

Table 7-2: Available DT blobs.

DTB	Comment
msc-sm2s-imx8m-dc-13N0600I.dtb	i.MX 8M dual core, headless
msc-sm2s-imx8m-dc-13N0600I-hdmi.dtb	i.MX 8M dual core, HDMI up to 4K support
msc-sm2s-imx8m-dc-13N0600I-lcdif-lvds-ama121a1.dtb	i.MX 8M dual core, LVDS (bridge) amd AMA121A1 (1280x800) panel support
msc-sm2s-imx8m-qc-13N0600I.dtb	i.MX 8M quad core, headless
msc-sm2s-imx8m-qc-13N0600I-hdmi.dtb	i.MX 8M quad core, HDMI up to 4K support
msc-sm2s-imx8m-qc-13N0600I-lcdif-lvds-ama121a1.dtb	i.MX 8M quad core, LVDS (bridge) and AMA121A1 (1280x800) panel support
msc-sm2s-imx8m-qc-001-dp.dts	Develop, i.MX 8M q/d core, display port up to 4K support
msc-sm2s-imx8m-qc-001-dual-head.dts	Develop, i.MX 8M q/d core, HDMI and LVDS (bridge) support
msc-sm2s-imx8m-qc-001-dual-head-single-cam.dts	Develop, i.MX 8M q/d core, HDMI, LVDS (bridge) and single camera sensor adapter support
msc-sm2s-imx8m-qc-001-dual-head-twin-cam.dts	Develop, i.MX 8M q/d core, HDMI, LVDS (bridge) and twin camera sensor adapter support
msc-sm2s-imx8m-qc-001-ep1-verification.dts	Develop, i.MX 8M q/d core, EP1 verification DT

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 7-3: 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-imx8m module:

Table 7-4: Available SMARC GPIOs

SMARC GPIO	IMX GPIO		Linux/U-Boot idx (Bank-1)*32+Id	'mscio-cmd' alias	Comment
	Bank	Id			
GPIO0	1	0	0	GPIO0	Not available if CAM0 populated
GPIO1	3	17	81	GPIO1	Not available if CAM1 populated
GPIO2	1	3	3	GPIO2	Not available if CAM0 populated
GPIO3	3	2	66	GPIO3	Not available if CAM1 populated
GPIO4	1	5	5	GPIO4	Available
GPIO5	1	1	1	GPIO5	Available
GPIO6	1	6	6	GPIO6	Available
GPIO7	1	7	7	GPIO7	Available
GPIO8	1	8	8	GPIO8	Available
GPIO9	1	9	9	GPIO9	Available
GPIO10	3	3	67	GPIO10	Available
GPIO11	1	11	11	GPIO11	Available

For detailed information about GPIO hardware and softwareoperation 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.

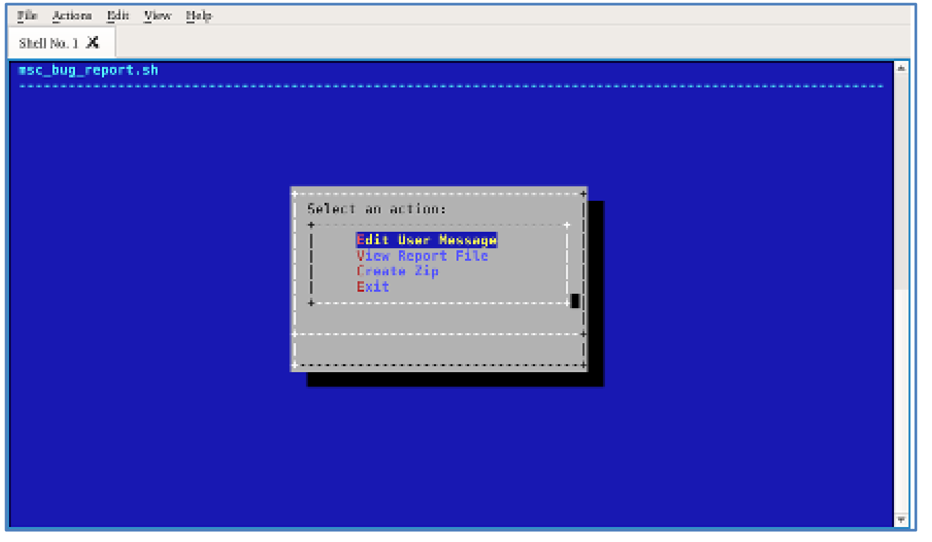


Figure 7-31: Bug report. Main page

- Select “Edit User Message”.

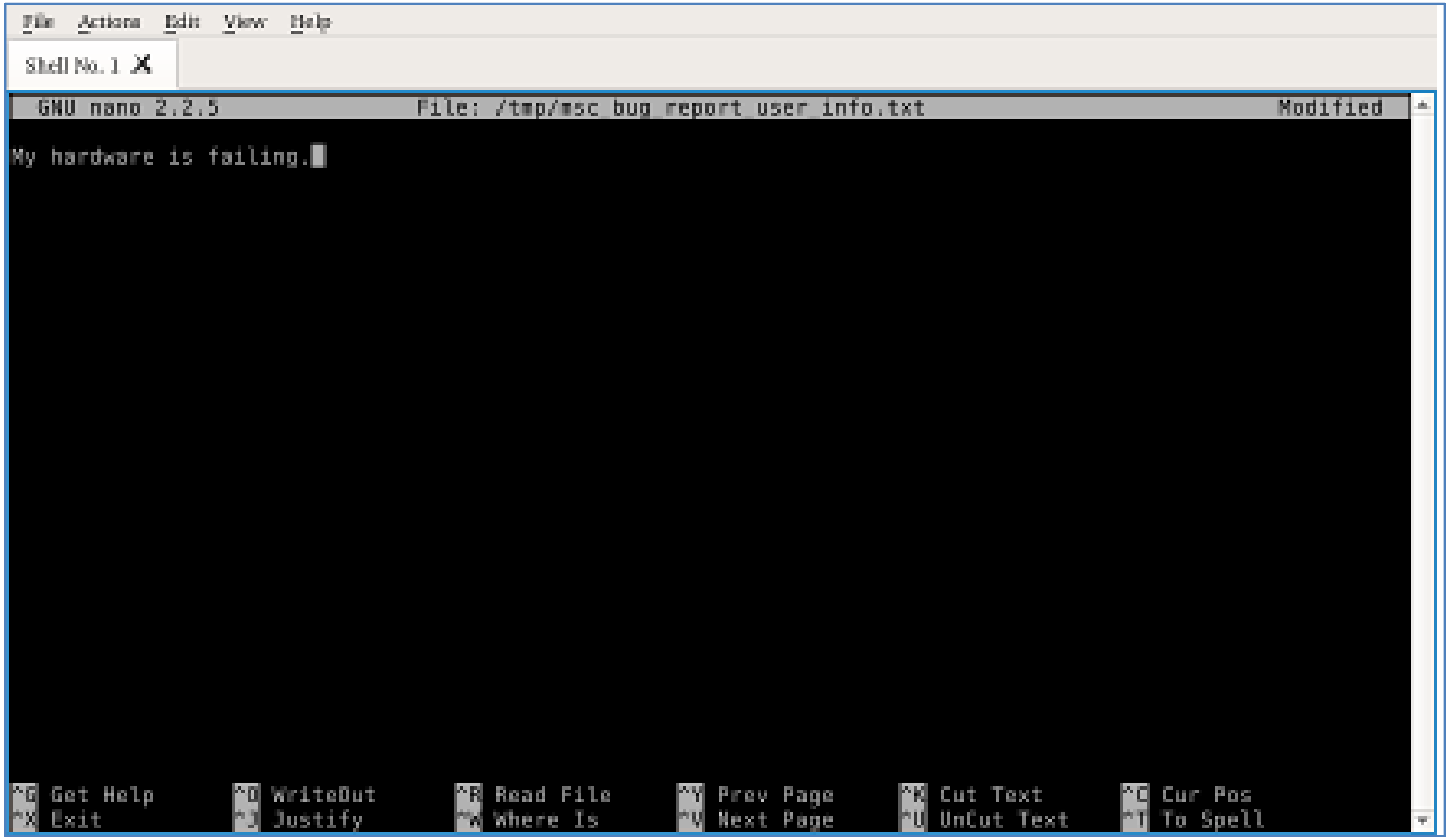


Figure 7-32: Bug report. User message editor

- Enter bug report message and press Ctrl-O and Ctrl-X.
- Optionally you can then view the message with the board report (hardware information).

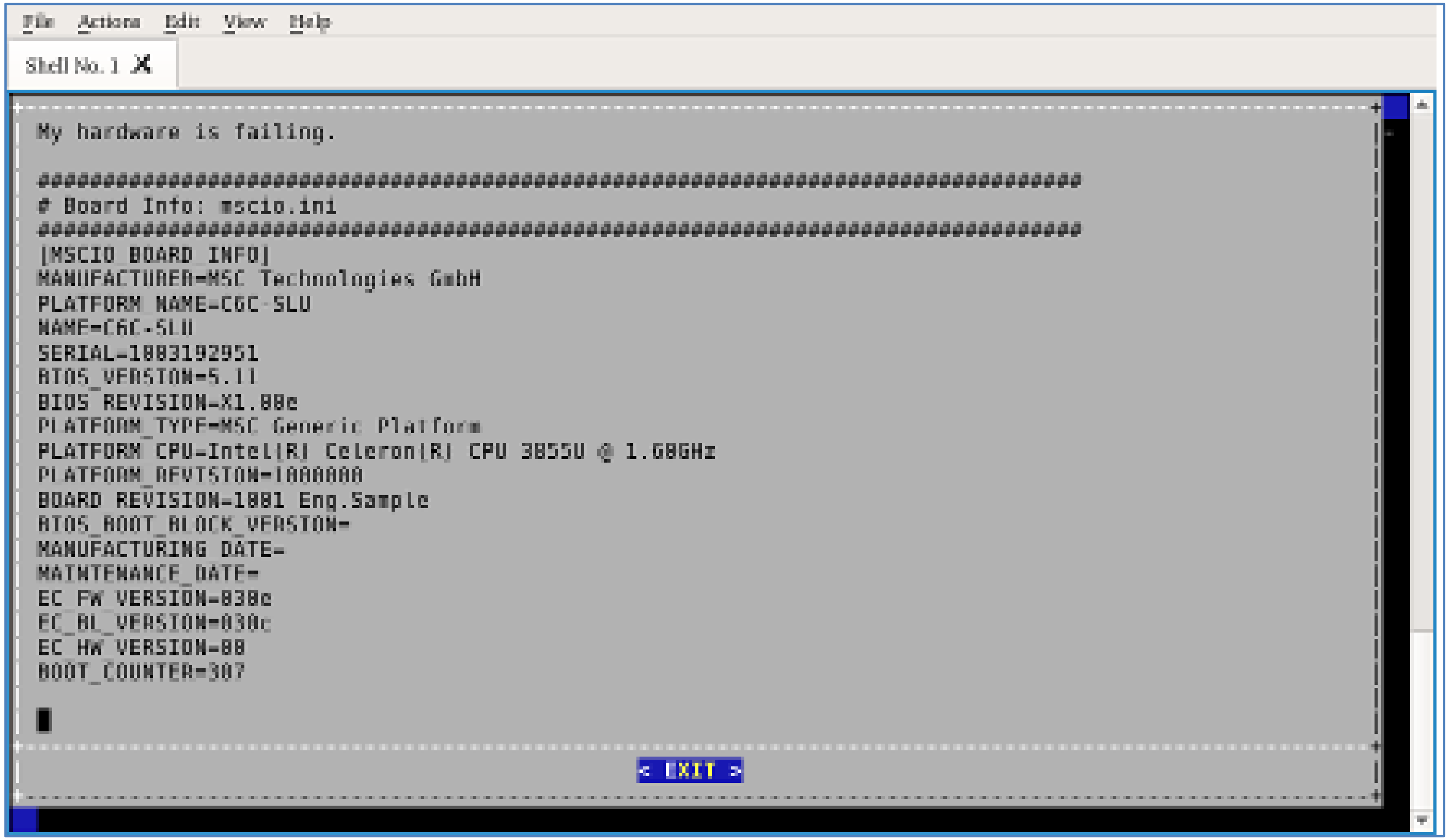


Figure 7-33: Bug report. Viewer page

- 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).

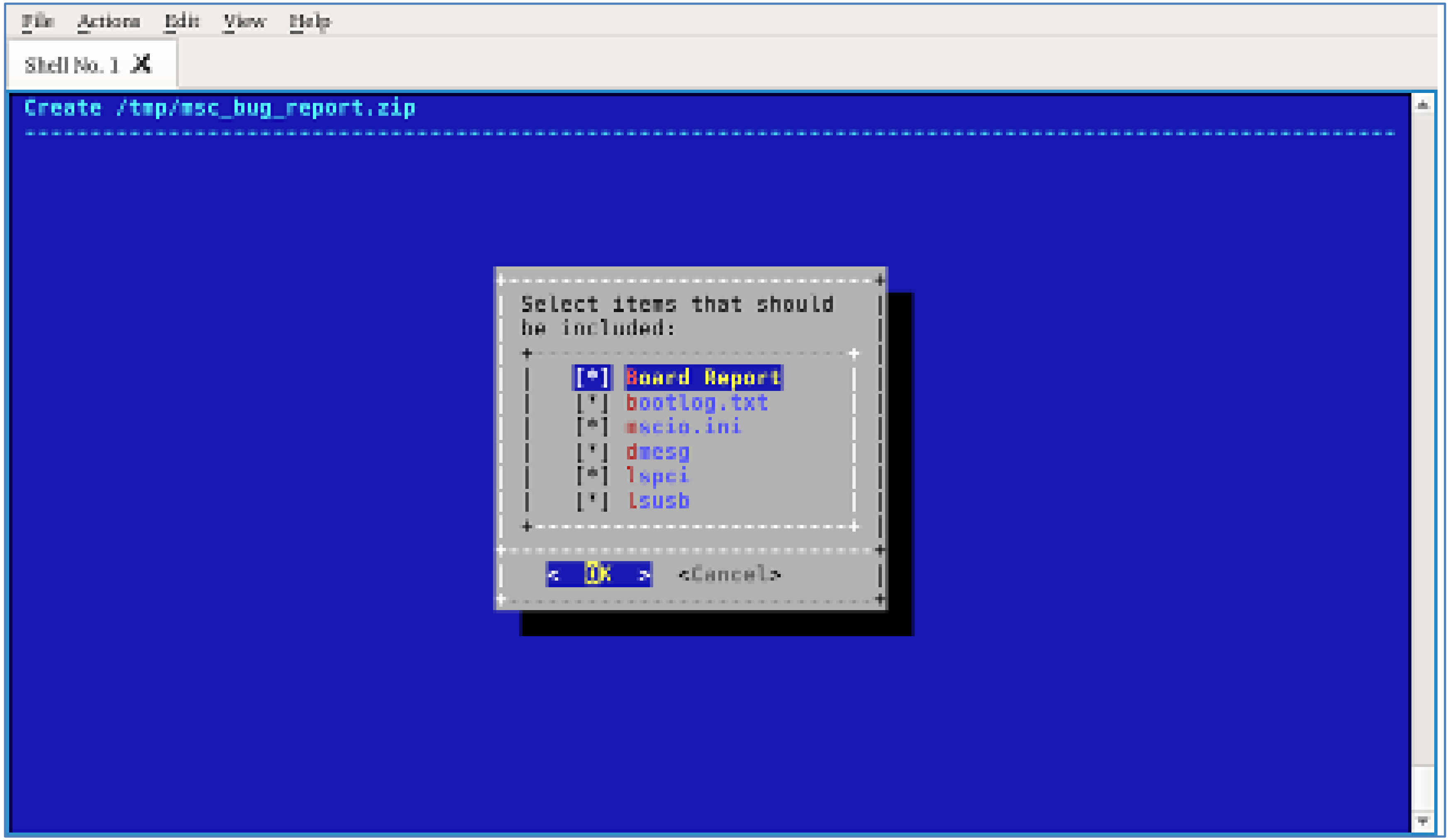


Figure 7-34: Bug report. Content selector

- Press “Save ZIP to a disc” and select the filesystem where to store the zip file. It is recommended to use a USB stick.

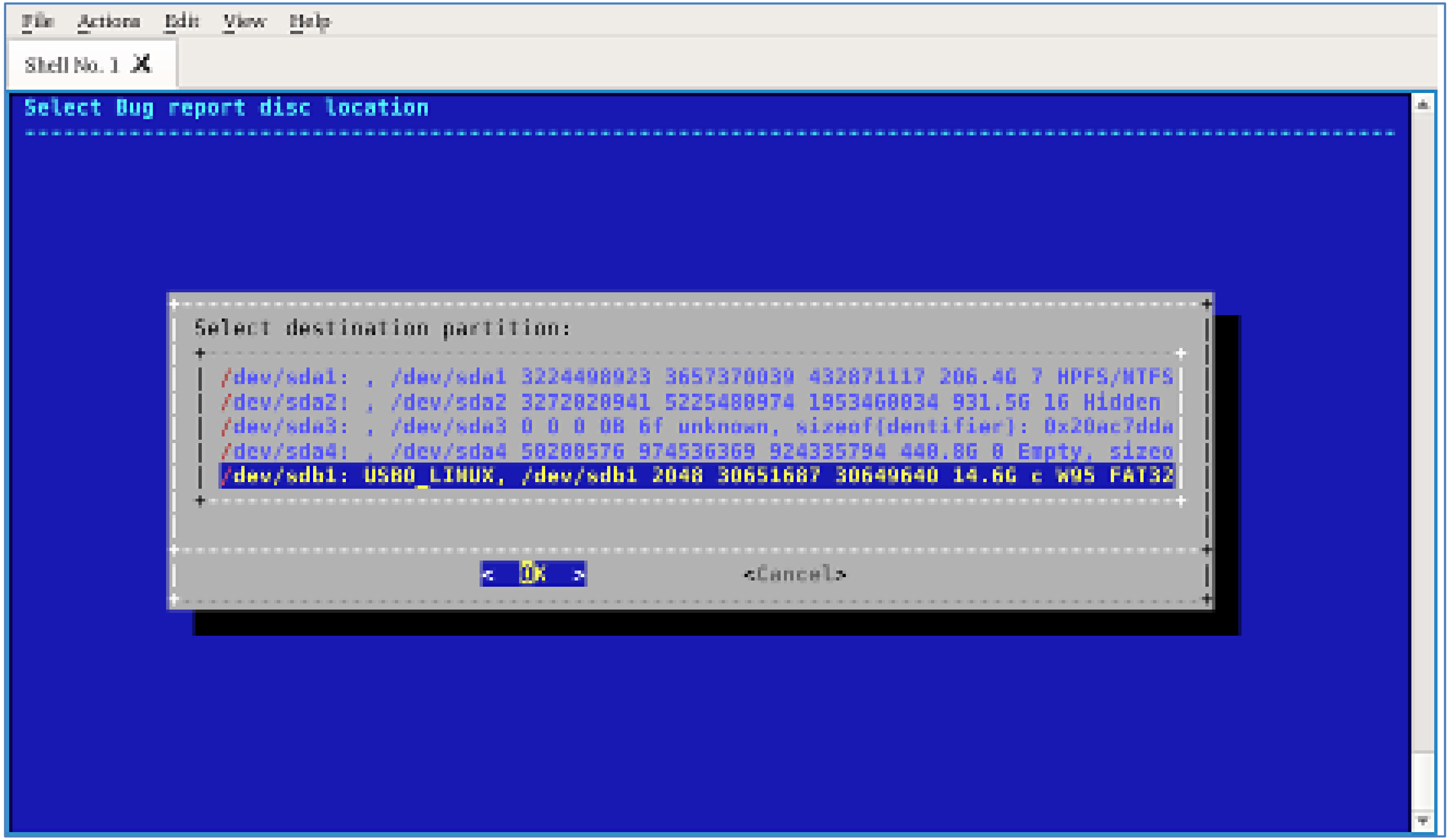


Figure 7-35: Bug report. Partition selector.

- Send the files msc_bug_report_brief.txt and msc_bug_report.zip to MSC: support.boards@avnet.eu

7.3.6 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 v1.5.0) 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" while in the MSC-LDK root directory. This will update MSC-LDK and all layers. Depending on the kind of hot-fix running "./setup.py" again might also 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-cache" can be moved to other MSC-LDK installations or shared by symbolic links.

NOTE: On the SM2S-IMX8M module, currently (as of March 2021) the "master" branch is not usable, instead all current changes are applied as hotfixes to the MSC-LDK release state of "v1.5.0". This is however subject to change in due time.

8 Troubleshooting

8.1 Known issues and limitations

8.1.1 Issue 1. Thermal management for i.MX 8M CPU.

Source: Hardware

Workaround: It is highly recommended to use a suitable cooling concept (for example the heat sink offered by MSC Technologies). The i.MX 8M CPU will tend to consume more power with rising temperature.

8.1.2 Issue 2. HDMI interface. Some ACER monitors are not supported in 1080p mode.

Source: Hardware/Software

Workaround: Solved with latest software release.

8.1.3 Issue 4. USB 2.0 interface. Not operable under U-Boot.

Source: Software

Workaround: Solved with latest software release.

8.1.4 Issue 5. USB 3.0 interface. Super speed not operable.

Source: Hardware/Software

Solution: Solved with latest software release.

8.1.5 Issue 6. Temperature Management Unit (TMU). Sensors report wrong values for temperatures below 0°C

Source: Hardware

Solution: Solved with latest software release

For further issues and limitations see also [11], Table 14, *“Known issues and workarounds for i.MX 8M Family SoC”*

8.1.6 Issue 7. ESPI interface is not available (P57 and P58 are crossed)

Source: Hardware

Solution: Solved with latest hardware release (4^{th} layout revision, DV4) P57=SPI1_DIN and P58=SPI1_DO.

8.1.7 Issue 8. SDIO_PWR_EN signal is currently not supported in ROM code

Source: Hardware

Solution: Solved with latest hardware release (4^{th} layout revision, DV4) and latest software release.

For further issues and limitations see also [11], Table 14, “Known issues and workarounds for i.MX 8M Family SoC”.

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