USER GUIDE

Kontron SMARC-sAMX8 User Guides

Doc. Rev. 1.2 Preliminary

Doc. ID: [To be Determined]

POSSIBILITIES START HERE kontron S&T Group

www.kontron.com

This page has been intentionally left blank

KONTRON SMARC-SAMX8 - USER GUIDE

Disclaimer

Kontron would like to point out that the information contained in this user guide may be subject to alteration, particularly as a result of the constant upgrading of Kontron products. This document does not entail any guarantee on the part of Kontron with respect to technical processes described in the user guide or any product characteristics set out in the user guide. Kontron assumes no responsibility or liability for the use of the described product(s), conveys no license or title under any patent, copyright or mask work rights to these products and makes no representations or warranties that these products are free from patent, copyright or mask work right infringement unless otherwise specified. Applications that are described in this user guide are for illustration purposes only. Kontron makes no representation or warranty that such application will be suitable for the specified use without further testing or modification. Kontron expressly informs the user that this user guide only contains a general description of processes and instructions which may not be applicable in every individual case. In cases of doubt, please contact Kontron.

Kontron S&T AG

Lise-Meitner-Str. 3-5
86156 Augsburg
Germany
www.kontron.com

Intended Use

THIS DEVICE AND ASSOCIATED SOFTWARE ARE NOT DESIGNED, MANUFACTURED OR INTENDED FOR USE OR RESALE FOR THE OPERATION OF NUCLEAR FACILITIES, THE NAVIGATION, CONTROL OR COMMUNICATION SYSTEMS FOR AIRCRAFT OR OTHER TRANSPORTATION, AIR TRAFFIC CONTROL, LIFE SUPPORT OR LIFE SUSTAINING APPLICATIONS, WEAPONS SYSTEMS, OR ANY OTHER APPLICATION IN A HAZARDOUS ENVIRONMENT, OR REQUIRING FAIL-SAFE PERFORMANCE, OR IN WHICH THE FAILURE OF PRODUCTS COULD LEAD DIRECTLY TO DEATH, PERSONAL INJURY, OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE (COLLECTIVELY, "HIGH RISK APPLICATIONS").

You understand and agree that your use of Kontron devices as a component in High Risk Applications is entirely at your risk. To minimize the risks associated with your products and applications, you should provide adequate design and operating safeguards. You are solely responsible for compliance with all legal, regulatory, safety, and security related requirements concerning your products. You are responsible to ensure that your systems (and any Kontron hardware or software components incorporated in your systems) meet all applicable requirements. Unless otherwise stated in the product documentation, the Kontron device is not provided with error-tolerance capabilities and cannot therefore be deemed as being engineered, manufactured or setup to be compliant for implementation or for resale as device in High Risk Applications. All application and safety related information in this document (including application descriptions, suggested safety measures, suggested Kontron products, and other materials) is provided for reference only.

⚠CAUTION

Handling and operation of the product is permitted only for trained personnel within a work place that is access controlled. Please follow the “General Safety Instrctions for IT Equipment” supplied with the system.

Revision History

Revision	Brief Description of Changes	Date of Issue	Author/ Editor
1.0	Initial Issue	2020-01-07	ML
1.1	Update Section 8.2 Boot from SD-Card	2020-05-15	ML
1.2	General update after board revision B	2022-02-23	SD

Terms and Conditions

Kontron warrants products in accordance with defined regional warranty periods. For more information about warranty compliance and conformity, and the warranty period in your region, visit http://www.kontron.com/terms-and-conditions.

Kontron sells products worldwide and declares regional General Terms & Conditions of Sale, and Purchase Order Terms & Conditions. Visit http://www.kontron.com/terms-and-conditions.

For contact information, refer to the corporate offices contact information on the last page of this user guide or visit our website CONTACT US.

Customer Support

Find Kontron contacts by visiting: http://www.kontron.com/support.

Customer Service

As a trusted technology innovator and global solutions provider, Kontron extends its embedded market strengths into a services portfolio allowing companies to break the barriers of traditional product lifecycles. Proven product expertise coupled with collaborative and highly-experienced support enables Kontron to provide exceptional peace of mind to build and maintain successful products.

For more details on Kontron's service offerings such as: enhanced repair services, extended warranty, Kontron training academy, and more visit http://www.kontron.com/support-and-services/services.

Customer Comments

If you have any difficulties using this user guide, discover an error, or just want to provide some feedback, contact Kontron support. Detail any errors you find. We will correct the errors or problems as soon as possible and post the revised user guide on our website.

Symbols

The following symbols may be used in this user guide

⚠DANGER	DANGER indicates a hazardous situation which, if not avoided, will result in death or serious injury.
⚠WARNING	WARNING indicates a hazardous situation which, if not avoided, could result in death or serious injury.
NOTICE	NOTICE indicates a property damage message.
⚠CAUTION	CAUTION indicates a hazardous situation which, if not avoided, may result in minor or moderate injury.
	**Electric Shock!** This symbol and title warn of hazards due to electrical shocks (> 60 V) when touching products or parts of products. Failure to observe the precautions indicated and/or prescribed by the law may endanger your life/health and/or result in damage to your material.
	**ESD Sensitive Device!** This symbol and title inform that the electronic boards and their components are sensitive to static electricity. Care must therefore be taken during all handling operations and inspections of this product in order to ensure product integrity at all times.
	**HOT Surface!** Do NOT touch! Allow to cool before servicing.
	**Laser!** This symbol inform of the risk of exposure to laser beam and light emitting devices (LEDs) from an electrical device. Eye protection per manufacturer notice shall review before servicing.
	This symbol indicates general information about the product and the user guide. This symbol also indicates detail information about the specific product configuration.
	This symbol precedes helpful hints and tips for daily use.

For Your Safety

Your new Kontron product was developed and tested carefully to provide all features necessary to ensure its compliance with electrical safety requirements. It was also designed for a long fault-free life. However, the life expectancy of your product can be drastically reduced by improper treatment during unpacking and installation. Therefore, in the interest of your own safety and of the correct operation of your new Kontron product, you are requested to conform with the following guidelines.

High Voltage Safety Instructions

As a precaution and in case of danger, the power connector must be easily accessible. The power connector is the product's main disconnect device.

⚠CAUTION

Warning

All operations on this product must be carried out by sufficiently skilled personnel only.

⚠CAUTION

Electric Shock!

Before installing a non hot-swappable Kontron product into a system always ensure that your mains power is switched off. This also applies to the installation of piggybacks. Serious electrical shock hazards can exist during all installation, repair, and maintenance operations on this product. Therefore, always unplug the power cable and any other cables which provide external voltages before performing any work on this product.

Earth ground connection to vehicle's chassis or a central grounding point shall remain connected. The earth ground cable shall be the last cable to be disconnected or the first cable to be connected when performing installation or removal procedures on this product.

Special Handling and Unpacking Instruction

NOTICE

ESD Sensitive Device!

Electronic boards and their components are sensitive to static electricity. Therefore, care must be taken during all handling operations and inspections of this product, in order to ensure product integrity at all times.

Do not handle this product out of its protective enclosure while it is not used for operational purposes unless it is otherwise protected.

Whenever possible, unpack or pack this product only at EOS/ESD safe work stations. Where a safe work station is not guaranteed, it is important for the user to be electrically discharged before touching the product with his/her hands or tools. This is most easily done by touching a metal part of your system housing.

It is particularly important to observe standard anti-static precautions when changing piggybacks, ROM devices, jumper settings etc. If the product contains batteries for RTC or memory backup, ensure that the product is not placed on conductive surfaces, including anti-static plastics or sponges. They can cause short circuits and damage the batteries or conductive circuits on the product.

Lithium Battery Precautions

If your product is equipped with a lithium battery, take the following precautions when replacing the battery.

⚠CAUTION

Danger of explosion if the battery is replaced incorrectly.

- Replace only with same or equivalent battery type recommended by the manufacturer.
- Dispose of used batteries according to the manufacturer's instructions.

General Instructions on Usage

In order to maintain Kontron's product warranty, this product must not be altered or modified in any way. Changes or modifications to the product, that are not explicitly approved by Kontron and described in this user guide or received from Kontron Support as a special handling instruction, will void your warranty.

This product should only be installed in or connected to systems that fulfill all necessary technical and specific environmental requirements. This also applies to the operational temperature range of the specific board version that must not be exceeded. If batteries are present, their temperature restrictions must be taken into account.

In performing all necessary installation and application operations, only follow the instructions supplied by the present user guide.

Keep all the original packaging material for future storage or warranty shipments. If it is necessary to store or ship the product then re-pack it in the same manner as it was delivered.

Special care is necessary when handling or unpacking the product. See Special Handling and Unpacking Instruction.

Quality and Environmental Management

Kontron aims to deliver reliable high-end products designed and built for quality, and aims to complying with environmental laws, regulations, and other environmentally oriented requirements. For more information regarding Kontron's quality and environmental responsibilities, visit http://www.kontron.com/about-kontron/corporate-responsibility/quality-management.

Disposal and Recycling

Kontron's products are manufactured to satisfy environmental protection requirements where possible. Many of the components used are capable of being recycled. Final disposal of this product after its service life must be accomplished in accordance with applicable country, state, or local laws or regulations.

WEEE Compliance

The Waste Electrical and Electronic Equipment (WEEE) Directive aims to:

- Reduce waste arising from electrical and electronic equipment (EEE)
- Make producers of EEE responsible for the environmental impact of their products, especially when the product become waste
- Encourage separate collection and subsequent treatment, reuse, recovery, recycling and sound environmental disposal of EEE
- Improve the environmental performance of all those involved during the lifecycle of EEE

Environmental protection is a high priority with Kontron.

Kontron follows the WEEE directive

You are encouraged to return our products for proper disposal.

Table of Contents

List of Tables

List of Figures

1/ Introduction

This manual describes the Smart Mobility ARChitecture (SMARC) sAMX8 (SMX8) board. The Advanced RISC Machines (ARM) based module is equipped with the NXP i.MX8 processor QuadMax

The SMARC-sAMX8 module is intended to complement the current SMARC product family by adding features and performance. Specifically, the iMX8 SMARC module is a performance upgrade from the iMX8X product in that it delivers higher CPU and graphics processing performance.

NOTICE

Some of the information contained within this product user guide applies only to certain product revisions. If certain information applies to specific product revisions it will be stated. Please check the product revision of your module to see if this information is applicable.

2 / Description

The SMARC-sAMX8 is a SMARC half-size module using the NXPs i.MX8 processor with quad core ARM. It is designed on the latest SMARC 2.0 specification. The SMARC-sAMX8 is a highly integrated, embedded computer board.

Figure 1: Half-size Card with SMARC interface

2.1. SMARC™ Computer-on-Modules

The SMARC™ standard was developed especially for new modules with ARM- and SoC-processors. Boards with this interfaces are characterized by the extremely flat form factor. The SMARC or MXM 3.0 connector comes with 314 pins and a construction height of just 4.3 millimeters. The connector is also available in a shock- and vibration-resistant version for rough environmental conditions.

Furthermore, the standard integrates dedicated interfaces for the latest ARM, x86 and SoC processors like LVDS, Gigabit Ethernet and HDMI support. In addition, dedicated camera interfaces are being incorporated. OEMs profit from minimized design effort and low Bill of Material (BoM) costs. SMARC™ defines two different module sizes in order to offer a high level of flexibility regarding different mechanical requirements.

2.2. Main characteristics

Main characteristics of the SMARC-sAMX8 are:

- Quad-Core Cortex A-53 with additional M4 and A72 Core on SMARC short size form factor
- Based on i.MX8 Series from NXP – three Pin compatible, scalable SKUs (1x QuadPlus, 2x QuadMax)
- Up to 8 GB LPDDR4 memory down
- LVDS support
- HDMI/DP
- eDP (via MIPI-DSI Bridge)
- 2x Gigabit Ethernet with internal MAC and PHY
- Optional PCIe WIFI/ USB Bluetooth M.2 1216
- SATA
- 2x CAN
- 6x USB2.0 (4x via USB Hub) and 1x USB3.0 SS (shared with one USB2.0)
- Support for Audio output and common features (SPI, I2C, SMB etc.)
- Optional eMMC flash onboard
- Full industrial grade temp. range E2 (-40°C up to +85°C) for standard SKUs, commercial version possible

2.3. Product Variants and Accessories

Following variants are planned:

Table 1: Product Variants of SMARC-sAMX8

Board	Description
P1-00544-001	SMARC-sAMX8 QuadPlus + 6GB RAM / 8GB eMMC
P1-00544-002	SMARC-sAMX8 QuadMax + 6GB RAM / 64GB eMMC + WIFI/BT module
P1-00544-003	Custom SMARC-sAMX8 QuadMax + 2GB RAM / 8GB eMMC + WIFI/BT module
P1-00544-004	SMARC-sAMX8 QuadMax + 8GB RAM / 64GB eMMC + WIFI/BT module

Following accessories are available:

- SMARC 2.0 Evaluation Carrier
- SMARC Starter Kit

2.4. SMARC-sAMX8 Feature Set

Table 2: SMARC-sAMX8 Feature Set

SMARC™ Feature specification	SMARC™ Specification Maximum Number Possible	SMARC-sAMX8 Feature support	Description
LVDS Display support	1 (2-channel)	1x (2-channel)	2 x LVDS Channel (Channel1 shared with DSI)
CSI Camera support	2	2x	1 x MIPI CSI with 2 lanes 1 x MIPI CSI with 4 lanes
DSI Display support	2	1x	1x MIPI DSI (shared with LVDS Channel 1)
HDMI/DP	1	1x	1x HDMI/DP Combo PHY
eDP	1	1x	1x eDP/DP via bridge chip
USB Interface	6 x USB 2.0 with 2 x USB 3.0 included	4x USB 2.0 via USB Hub 1x USB 3.0 2x USB2 OTG (one shared with USB3.0)	2x USB OTG ports, 4x USB host ports, 1x USB 3.0 SS (If WIFI/BT module is installed, USB5 is not accessible)
PCIe Interface	4	3x	3x PCIe 3.0 lane (2 root ports)[1] (PCIe C is shared with SATA port)
SATA Interface	1	1x	1x SATA (SATA is shared with PCIe C)
GbE Interface	2	2x GbE	2x GB Ethernet
SDIO Interface	1	1x	1x SDIO interface on SMARC connector (There is also one microSD connector on the module)
SPI Interface	2	2x	1x SPI, 1x QuadSPI (3 additional SPI instead of LVDS on Custom SMARC module)
I2S Interface	2	2x	Default lead to HDMI
I2C Interface	5	5x	External Fast I2C_GP from Embedded Controller or CPU I2C_PM for Power Management
CAN	2	2x	2x CAN with CAN-FD support (1 additional CAN instead of LVDS on Custom SMARC module)
UART	4	4x	
GPIOs		12x GPIOs	

1. If all the PCIe port are used, one of the root port will be in x2 (Port A and B) and the one from the PCIe port C in PCIe x1 for a total of “3 lanes” with a maximum of 2 devices.

3/ System Specifications

3.1. Component Main Data

The table below summarizes the features of the motherboard.

Table 3: Component Main Data

SMARC-sAMX8	
Form factor	Short size Smart Mobility ARChitecture (SMARC) Hardware with 82 mm x 50 mm
Processor	NXP i.MX8 29mm x 29mm BGA package with 1313 balls in 0.75 mm pitch. Used processor types are i.MX 8 QuadPlus and i.MX 8 QuadMax.
Memory	LPDDR4 @ 1600 MHz (no ECC)
Boot Flash	64 MB SPI NOR flash
Bootloader/BIOS	U-Boot Bootloader, Flash for Bootloader
embedded Multimedia Card (eMMC)	8 to 64 GB eMMC with pSLC option available
EEPROM	▶ Type: 24C32, 4k x 8 (32 kbit) ▶ Connected at I2C_GP bus at address 0x50 (7-bit)
Display LVDS	▶ 18/24-bit LVDS RGB (True Color) ▶ Resolution: up to 1920x1080/60Hz ▶ Single/Dual Channel
Display DP/HDMI	▶ HDMI/DisplayPort from SoC
PCIe / SATA	▶ 3x PCIe lanes / 1 SATA (Shared with 1x PCIe)
Onboard Controllers	
Ethernet Controller	2x GBE PHY AR8031
Wifi/Bluetooth	Optional Wifi/Bluetooth module M.2 1216
Watchdog Timer	CPU internal watchdog, configurable timeout counter with timeout periods from 0.5 to 128 seconds
USB HUB	USB HUB USB4604 for 4x USB 2.0 ports on SMARC Connector
Display bridge	MIPI DSI to eDP/DP SN65DSI86
Real Time Clock (RTC)	High accuracy (+/-3 ppm), low power, RV-8803
System Management Controller	No dedicated System Management Controller on module System settings can be arranged in U-Boot environment variables
Storage	8 to 64 GB eMMC 5.1 Flash (option)
H/W Status Monitor	Voltage, Temperature monitoring
Security	iMX8 features
Operating System Support	Linux Yocto, other Operating Systems only on customer request

Interfaces via Smarc I/O	
I2C	External Fast I2C_GP from Embedded Controller or CPU I2C_PM for Power Management
LAN, USB	2x Gb-Ethernet 1x USB 2.0/3.0 OTG on USB#3 (shared), 1x USB Client/OTG on USB#0, 4x USB 2.0, via USB Hub
PCIe	3x PCIe Gen3.0 (2 root port) PCIe B is shared with WIFI/BT module PCIe C is shared with SATAIII
Display	Dual channel LVDS 1/2x18/24bit (openLDI and VESA) up to 1920x1080
Camera	MIPI CSI camera support default: 2 lane interface and 4 lane interface
SD-Card	1x SDIO
Serial Peripheral Interface (SPI)	2 x SPI interface on defined pins for general purpose devices
SER	4x serial ports with full function according to SMARC 2.0.
GPIO	12x General Purpose Inputs/Outputs (GPIO)
Other Connectivity	2x CAN
Power	
Consumption	Maximum Power consumption of the board is measured to 12 W (Quad Core)
Input Voltage	VCC 3.3V to 5.25V
Power Supply Limits	Voltage Ripple maximum 200mV peak to peak at 0 to 20 MHz. 0.1 to 20ms rise time from input voltage ≤10% to nominal VCC inrush current peak limit. G3/S5 to S0 acc. SFX Design Guide
Power Features	▶ Module shall power on automatically in single supply operation when VCC is connected (Uboot setting) ▶ Module shall be possibly reconfigured to start after PowerButton, when VCC is connected
Security	
	▶ HAB, SRTC, SJTAG, TrustZone® ▶ AES256, RSA4096, SHA-256 ▶ 3DES, ARC4, MD-5 ▶ Flashless SHE, ECC ▶ Tamper, Inline Enc Engine

3.2. Environmental Conditions

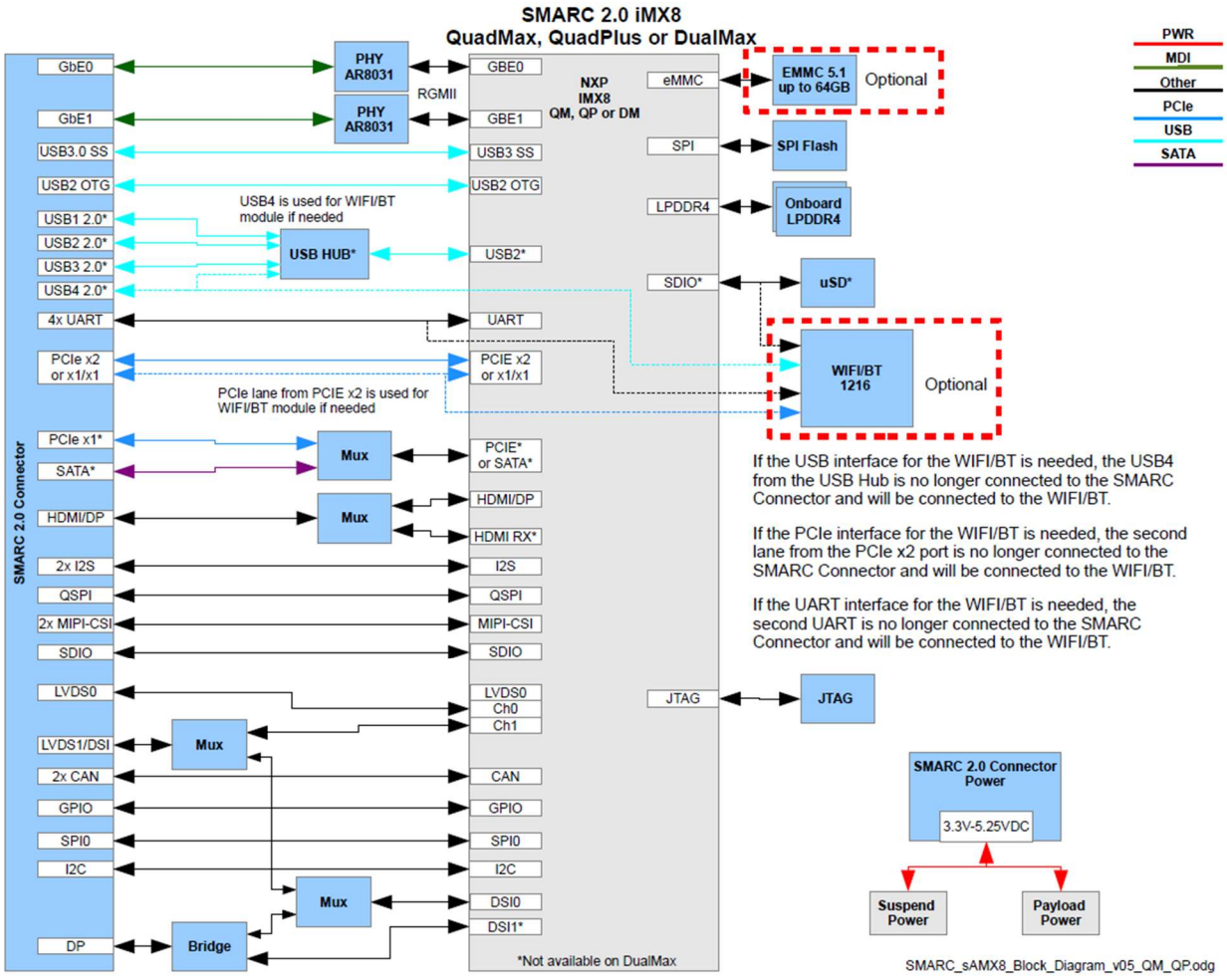
Table 4: Environmental Conditions

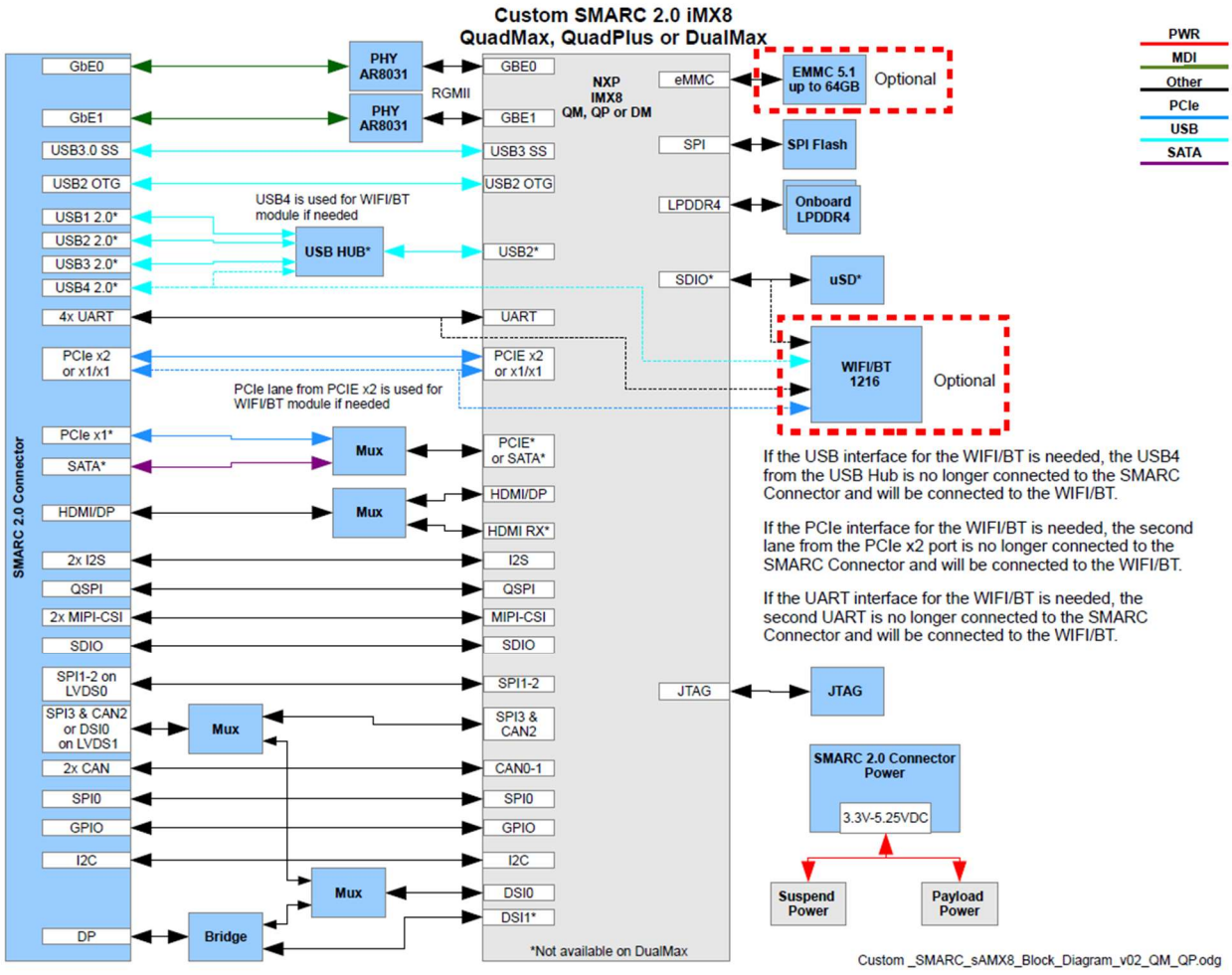
Operating	▶ 0°C to 60°C (on request for cost optimization) ▶ -40°C to 85°C (by design)
Storage	▶ -40°C to +85°C
Relative Humidity	▶ non-condensing 10 % to 93 % at 40°C acc. to IEC 60068-2-78
Electromagnetic Compatibility (EMC)	According to EN55022 (Class B), EN61000-6-2 and EN61000-6-4
CE	EN 62368-1:2014 - Safety for audio/video and information technology equipment
UL	Component Recognition to UL60950-1 - Information Technology Equipment Including Electrical Business Equipment
REACH	REACH compliant (Regulation (EC) No 1907/2006)
WEEE	WEEE compliant (Directive 2012/19/EU)
RoHS II Compliance	The product is RoHS II compliant (Directive 2011/65/EU)

3.3. Functional Block Diagram

The block diagram shows all available interfaces on the sAMX8 module.

Figure 2: Block Diagram




Custom SMARC 2.0 iMX8
QuadMax, QuadPlus or DualMax
PWR
MDI
Other
PCIe
USB
SATA
SMARC 2.0 Connector
GbE0
GbE1
USB3.0 SS
USB2 OTG
USB1 2.0*
USB2 2.0*
USB3 2.0*
USB4 2.0*
4x UART
PCIe x2 or x1/x1
PCIe x1*
SATA*
HDMI/DP
2x I2S
QSPI
2x MIPI-CSI
SDIO
SPI1-2 on LVDS0
SPI3 & CAN2 or DSI0 on LVDS1
2x CAN
SPI0
GPIO
I2C
DP
PHY AR8031
PHY AR8031
RGMII
USB4 is used for WIFI/BT module if needed
USB HUB*
PCIe lane from PCIE x2 is used for WIFI/BT module if needed
Mux
Mux
Mux
Bridge
GBE0
GBE1
USB3 SS
USB2 OTG
USB2*
UART
PCIE x2 or x1/x1
PCIE* or SATA*
HDMI/DP
HDMI RX*
I2S
QSPI
MIPI-CSI
SDIO
SPI1-2
SPI3 & CAN2
CAN0-1
SPI0
GPIO
I2C
DSI0
DSI1*
*Not available on DualMax
NXP IMX8 QM, QP or DM
eMMC
SPI
LPDDR4
SDIO*
JTAG
EMMC 5.1 up to 64GB
Optional
SPI Flash
Onboard LPDDR4
uSD*
WIFI/BT 1216
Optional
JTAG
If the USB interface for the WIFI/BT is needed, the USB4 from the USB Hub is no longer connected to the SMARC Connector and will be connected to the WIFI/BT.
If the PCIe interface for the WIFI/BT is needed, the second lane from the PCIe x2 port is no longer connected to the SMARC Connector and will be connected to the WIFI/BT.
If the UART interface for the WIFI/BT is needed, the second UART is no longer connected to the SMARC Connector and will be connected to the WIFI/BT.
SMARC 2.0 Connector Power
3.3V-5.25VDC
Suspend Power
Payload Power
Custom _SMARC_sAMX8_Block_Diagram_v02_QM_QP.odg

4/ Board and Connectors

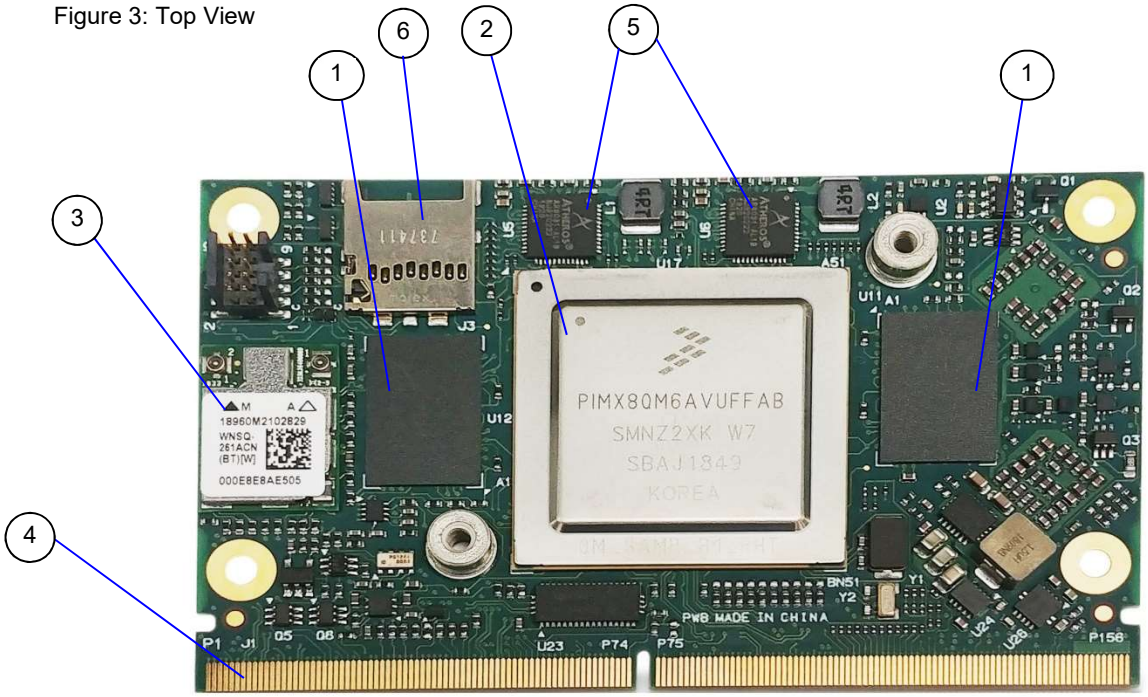
4.1.1. Connectors

Table 5: Connectors of SMARC-sAMX8X

Connector	Function	Remark
SMARC	Central Interface	Mating connector: SMARC 2.0 (MXM3)

4.2. Mainboard view and I/O locations

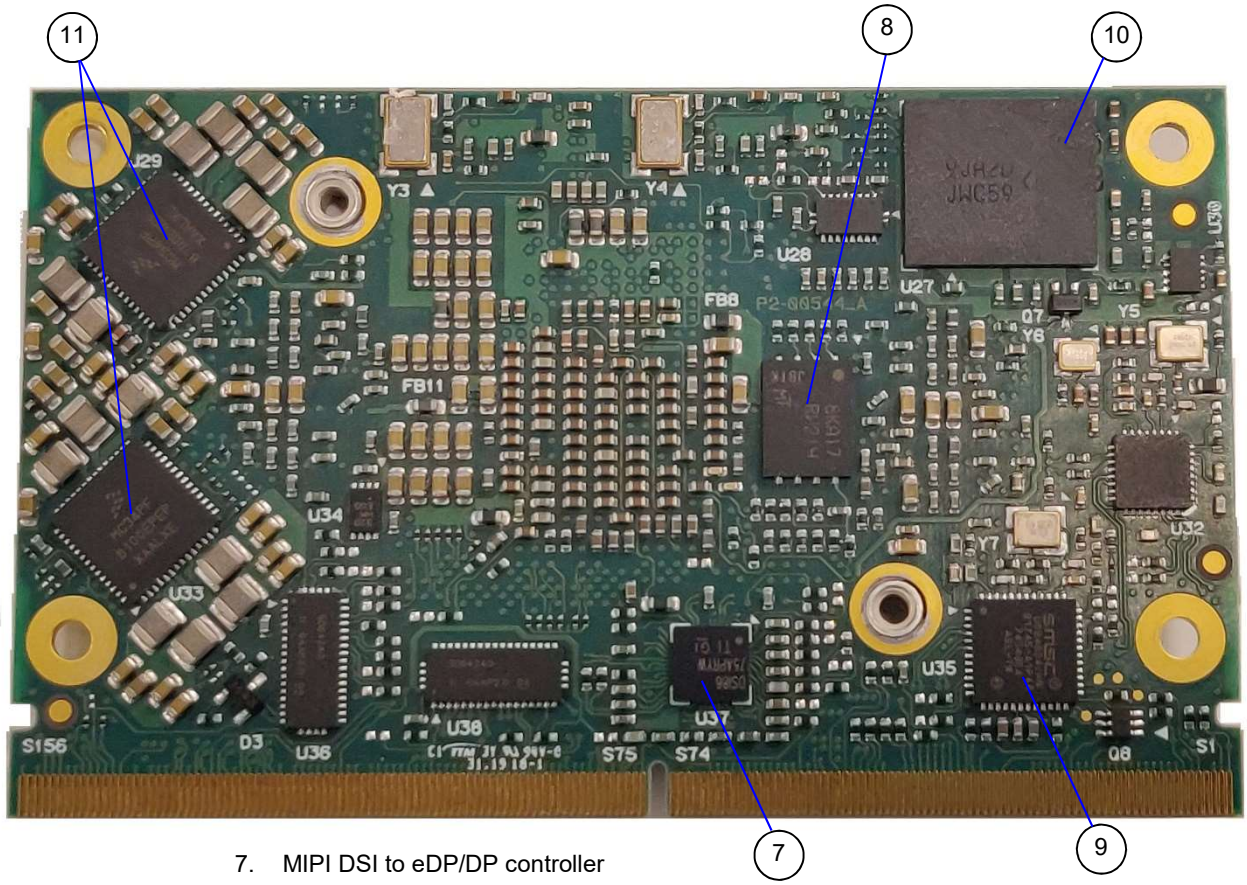
Figure 3: Top View



1. LPDDR4 memory
2. Freescale Processor i.MX8
3. WIFI/Bluetooth M.2 1216 module (optional)
4. SMARC Interface
5. GbE 0 / 1
6. uSD connector (not available on DualMax)

4.3. Bottom Side

Figure 4: Bottom Side from SMARC-sAMX8



7. MIPI DSI to eDP/DP controller
8. SPI Flash
9. USB Hub
10. eMMC
11. PMICs

4.4. Mechanical Drawings

Figure 5: Dimensions of SMARC-sAMX8

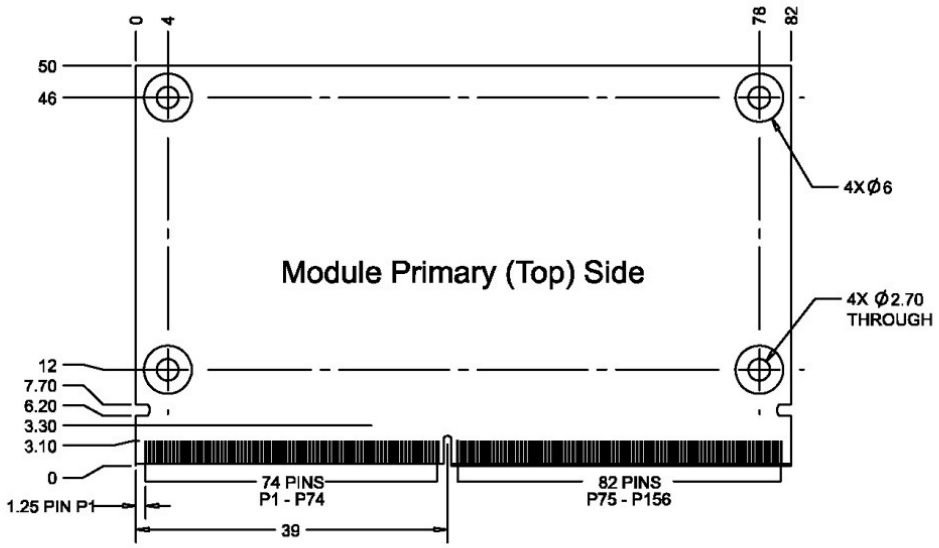
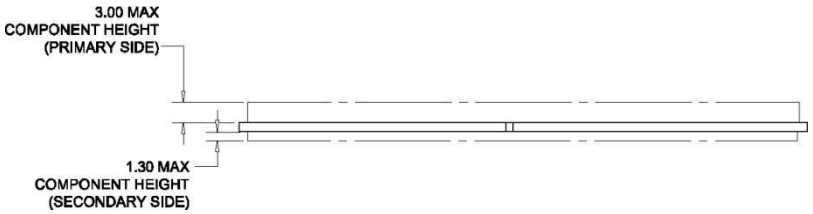


Figure 6: Thickness from side view



4.5. Thermal Considerations

The Cooling concept is based on a standard cooler for SMARC modules with mounting holes for iMX8 module.

Figure 7: Heatspreader Top View with screw holes

Figure 8: Heatspreader Bottom View

NOTICE Heat spreader mech. data are available on customer section

5/ Pin Definitions

5.1. Processor Support

Kontron uses a NXP i.MX8 chip with with 29 mm x 29 mm BGA package in 0.75 mm pitch available.

Table 6: Processor Support

Name	Core	RAM	Cache	Tj
QuadPlus	A-53 (4x 1.26 GHz) A-72 (1x 1.6GHz) M4F (2x 266MHz) 2x GPU (GC7000Lite/XSVX)	LPDDR4 (no ECC)	1MB with ECC	-40°C to 125ºC
QuadMax	A-53 (4x 1.2 GHz) A-72 (2x 1.6GHz) M4F (2x 264MHz) 2x GPU (GC7000XSVX)	LPDDR4 (no ECC)	1MB with ECC	-40°C to 125ºC

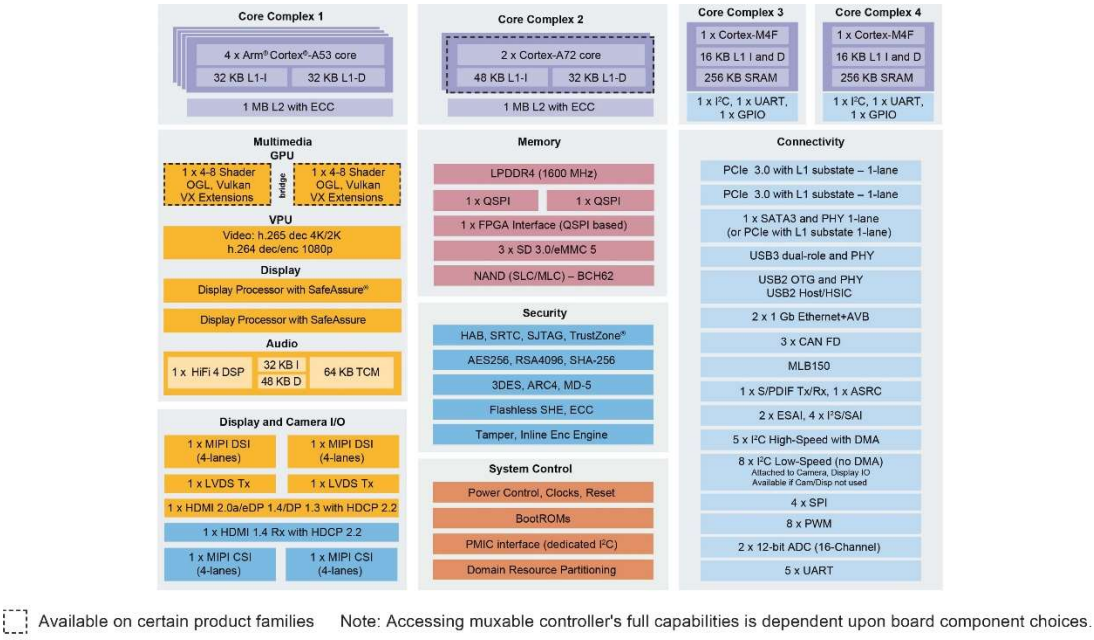


Figure 9: Processor Block Diagram (Source: NXP)

5.2. System Memory Support

The system supports the following memory features:

- LPDDR4 @ 1600 MHz (no ECC)

Table 7: Memory Options

Processor type	RAM
QuadPlus, QuadMax	LPDDR4 (no ECC) up to 8GB

5.3. eMMC Flash Memory

An optional embedded Multimedia Flash Card (eMMC) complying with the eMMC 5.1 specification can be permanently attached to the module, allowing for a capacity of up to 64 GByte NAND Flash.

Specific eMMC Flash memory features are:

- Up to 64 GByte
- eMMC 5.1 specification
- Class 0 (basic); class 2 (block read); class 4 (block write); class 5 (erase); class 6 (write protection); class 7 (lock card)
- HS200/HS400 modes
- DDR modes up to 52 MHz clock speed
- ECC and block management
- Boot operation (High-speed boot)
- Sleep mode
- Permanent and power-on write protection
- Replay-protected memory block (RPMB)
- Secure erase and secure trim

5.4. WIFI/Bluetooth 1216

WIFI/Bluetooth 1216 option may be install onboard. Allow to connect SMARC-sAMX8 to WIFI and Bluetooth network.

If module installed, following features might be unavailable:

-SD card onboard is unavailable if a module with SDIO interface is used

-USB5 on SMARC Connector is unavailable if a module with USB interface is used

-PCIe B is unavailable if a module with PCIe interface is used

-SER2 on SMARC Connector is unavailable if a module with a serial interface is used

5.5. Multiplexer

There are 4 multiplexers on board to support various configurations possibilities. These multiplexers are configured in the BSP. Below you will find more information on the different multiplexers on the SMARC module:

Table 8: Memory Options

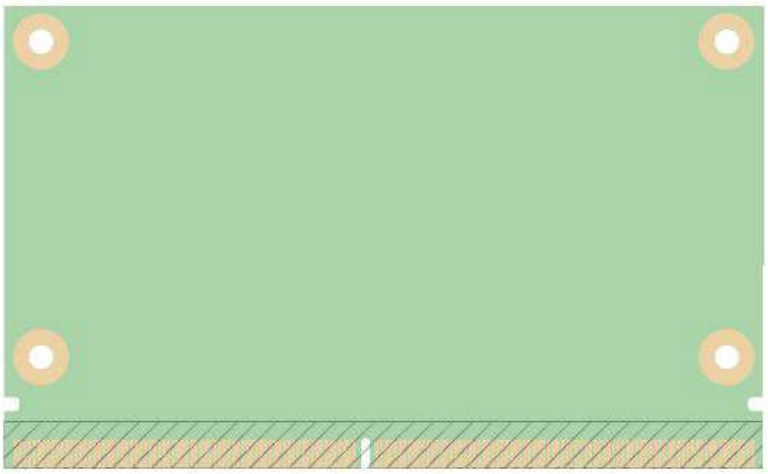
Multiplexer	Description
SATA/PCIe	Allow to choose between an additional PCIe port or a SATA port. Do note that there is only two roots port for PCIe device.
DSI1 / eDP DP	Allow to choose between the eDP/DP bridge chip or DSI0 from the SoC on the SMARC connector. DSI0 is going through a second mux since DSI0 is shared with LVDS Channel1. **Note: DSI0 from SoC is connected to DSI1 on SMARC connector.**
DSI1 / LVDS0 Channel1 **(Standard SMARC 2.0 module pinout)**	Allow to choose between the DSI0 from the SoC or the LVDS0 Channel1. Do note that the multiplexer DS1 / eDP DP should be set to DSI for the DSI option to work.
DS1 / (SPI3 and CAN3) **(Custom SMARC module pinout)**	Allow to choose between the DSI0 from the SoC or SPI3 and CAN3. Do note that the multiplexer DS1 / eDP DP should be set to DSI for the DSI option to work.
HDMI and DisplayPort / HDMI RX	Allow to choose between the HDMI/DP TX from the SoC or HDMI RX from the SoC

5.6. SMARC Connector

The SMARC connector has different pins on both sides:

- Top side: 74 pins are on the left side, 82 pins on the right side
- Bottom side: 75 pins are on the left side, 83 pins on the right side

Figure 10: 314-pin SMARC Connector,



5.7. Pinout of SMARC sAMX8 Connector

There is two pinouts available for this module:

1- SMARC 2.0 pinout

2- Custom SMARC pinout that removes the LVDS interface. The LVDS interface is replaced by 1x CAN and 3x SPI

5.7.1. Pinout of SMARC sAMX8 Topside Connector **(SMARC 2.0 pinout)**

Note: Pin Type/Tolerance definition is according to SMARC Specification 2.0

Table 9: Pinout of SMARC sAMX8 Topside Connector

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
P1	SMB_ALERT_1V8#	In	-	CMOS 1.8V	iMX8	
P2	GND	-	-	-	-	GND
P3	CSI1_CK+	In	-	LVDS D-PHY	iMX8	-
P4	CSI1_CK-	In	-	LVDS D-PHY	iMX8	-
P5	GBE1_SDP	Bi-Dir	-	CMOS 3.3V	AR8031 (CLK_25M)	-
P6	GBE0_SDP	Bi-Dir	-	CMOS 3.3V	AR8031 (CLK_25M)	-
P7	CSI1_RX0+	In	-	LVDS D-PHY	iMX8	-
P8	CSI1_RX0-	In	-	LVDS D-PHY	iMX8	-
P9	GND	-	-	-	-	GND
P10	CSI1_RX1+	In	-	LVDS D-PHY	iMX8	-
P11	CSI1_RX1-	In	-	LVDS D-PHY	iMX8	-
P12	GND	-	-	-	-	GND
P13	CSI1_RX2+	In	-	LVDS D-PHY	iMX8	-
P14	CSI1_RX2-	In	-	LVDS D-PHY	iMX8	-
P15	GND	-	-	-	-	GND
P16	CSI1_RX3+	In	-	LVDS D-PHY	iMX8	-
P17	CSI1_RX3-	In	-	LVDS D-PHY	iMX8	-
P18	GND	-	-	-	-	GND
P19	GBE0_MDI3-	Bi-Dir	-	GBE MDI	AR8031	-
P20	GBE0_MDI3+	Bi-Dir	-	GBE MDI	AR8031	-
P21	GBE0_LINK100#	Out/OD	-	CMOS 3.3V	AR8031	-
P22	GBE0_LINK1000#	Out/OD	-	CMOS 3.3V	AR8031	-
P23	GBE0_MDI2-	Bi-Dir	-	GBE MDI	AR8031	-
P24	GBE0_MDI2+	Bi-Dir	-	GBE MDI	AR8031	-
P25	GBE0_LINK_ACT#	Out/OD	-	CMOS 3.3V	AR8031	-
P26	GBE0_MDI1-	Bi-Dir	-	GBE MDI	AR8031	-
P27	GBE0_MDI1+	Bi-Dir	-	GBE MDI	AR8031	-
P28	GBE0_CTREF	Out	NC	-	-	-
P29	GBE0_MDI0-	Bi-Dir	-	GBE MDI	AR8031	-
P30	GBE0_MDI0+	Bi-Dir	-	GBE MDI	AR8031	-
P31	SPI0_CS1#	Out	Serial–33R	CMOS 1.8V	iMX8	-
P32	GND	-	-	-	-	GND
P33	SDIO_WP	In	PU-10k	CMOS 3.3V	iMX8	V_3V3
P34	SDIO_CMD	Bi-Dir	-	CMOS 3.3V	iMX8	-
P35	SDIO_CD#	In	PU-10k	CMOS 3.3V	iMX8	V_3V3

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
P36	SDIO_CK	Out	-	CMOS 3.3V	iMX8	-
P37	SDIO_PWR_EN	Out	-	CMOS 3.3V	iMX8	-
P38	GND	-	-	-	-	GND
P39	SDIO_D0	Bi-Dir	-	CMOS 3.3V	iMX8	-
P40	SDIO_D1	Bi-Dir	-	CMOS 3.3V	iMX8	-
P41	SDIO_D2	Bi-Dir	-	CMOS 3.3V	iMX8	-
P42	SDIO_D3	Bi-Dir	-	CMOS 3.3V	iMX8	-
P43	SPI0_CS0#	Out	Serial-33R	CMOS 1.8V	iMX8	-
P44	SPI0_CK	Out	Serial-33R	CMOS 1.8V	iMX8	-
P45	SPI0_DIN	In	Serial-33R	CMOS 1.8V	iMX8	-
P46	SPI0_DO	Out	Serial-33R	CMOS 1.8V	iMX8	-
P47	GND	-	-	-	-	GND
P48	SATA_TX+	Out	Serial-10n	SATA	iMX8	-
P49	SATA_TX-	Out	Serial-10n	SATA	iMX8	-
P50	GND	-	-	-	-	GND
P51	SATA_RX+	In	Serial-10n	SATA	iMX8	-
P52	SATA_RX-	In	Serial-10n	SATA	iMX8	-
P53	GND	-	-	-	-	GND
P54	ESPI_CS0#	Out	Serial-33R	CMOS 1.8V	iMX8	-
P55	ESPI_CS1#	Out	Serial-33R	CMOS 1.8V	iMX8	-
P56	ESPI_CK	Out	Serial-33R	CMOS 1.8V	iMX8	-
P57	ESPI_IO_1	Bi-Dir	Serial-33R	CMOS 1.8V	iMX8	-
P58	ESPI_IO_0	Bi-Dir	Serial-33R	CMOS 1.8V	iMX8	-
P59	GND	-	-	-	-	GND
P60	USB0+	Bi-Dir	-	USB	iMX8	-
P61	USB0-	Bi-Dir	-	USB	iMX8	-
P62	USB0_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	iMX8	V_3V3
P63	USB0_VBUS_DET	In	PD-4k7	USB VBUS 5V	iMX8	-
P64	USB0_OTG_ID	In	-	CMOS 3.3V	iMX8	-
P65	USB1+	Bi-Dir	-	USB	USB4604	-
P66	USB1-	Bi-Dir	-	USB	USB4604	-
P67	USB1_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	USB4604	V_3V3
P68	GND	-	-	-	-	GND
P69	USB2+	Bi-Dir	-	USB	USB4604	-
P70	USB2-	Bi-Dir	-	USB	USB4604	-
P71	USB2_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	USB4604	V_3V3
P72	RSVD	-	NC	-	-	-
P73	RSVD	-	NC	-	-	-
P74	USB3_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	iMX8	V_3V3
P75	PCIE_A_RST#	Out	-	CMOS 3.3V	iMX8	-
P76	USB4_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	USB4604	V_3V3
P77	RSVD	-	NC	-	-	-
P78	RSVD	-	NC	-	-	-
P79	GND	-	-	-	-	GND
P80	PCIE_C_REFCK+	Out	-	LVDS PCIe	9FGV0441AKILF	-
P81	PCIE_C_REFCK-	Out	-	LVDS PCIe	9FGV0441AKILF	-
P82	GND	-	-	-	-	GND

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
P83	PCIE_A_REFCK+	Out	-	LVDS PCIe	9FGV0441AKILF	-
P84	PCIE_A_REFCK-	Out	-	LVDS PCIe	9FGV0441AKILF	-
P85	GND	-	-	-	-	GND
P86	PCIE_A_RX+	In	-	LVDS PCIe	iMX8	-
P87	PCIE_A_RX-	In	-	LVDS PCIe	iMX8	-
P88	GND	-	-	-	-	GND
P89	PCIE_A_TX+	Out	Serial-220n	LVDS PCIe	iMX8	-
P90	PCIE_A_TX-	Out	Serial-220n	LVDS PCIe	iMX8	-
P91	GND	-	-	-	-	GND
P92	HDMI_D2+/ DP1_LANE0+	Out	-	TMDS	iMX8	-
P93	HDMI_D2/ DP1_LANE0-	Out	-	TMDS	iMX8	-
P94	GND	-	-	-	-	GND
P95	HDMI_D1+/ DP1_LANE1+	Out	-	TMDS	iMX8	-
P96	HDMI_D1- /DP1_LANE1-	Out	-	TMDS	iMX8	-
P97	GND	-	-	-	-	GND
P98	HDMI_D0+/ DP1_LANE2+	Out	-	TMDS	iMX8	-
P99	HDMI_D0/ DP1_LANE2-	Out	-	TMDS	iMX8	-
P100	GND	-	-	-	-	GND
P101	HDMI_CK+ /DP1_LANE3+	Out	-	TMDS	iMX8	-
P102	HDMI_CK /DP1_LANE3-	Out	-	TMDS	iMX8	-
P103	GND	-	-	-	-	GND
P104	HDMI_HPD/DP1_HPD	In	PD-100k	CMOS 1.8V	iMX8	-
P105	HDMI_CTRL_CK/ DP1_AUX+	Bi-Dir	PU-2k2	CMOS 1.8V	iMX8	V_1V8
P106	HDMI_CTRL_DAT/ DP1_AUX-	Bi-Dir	PU-2k2	CMOS 1.8V	iMX8	V_1V8
P107	DP1_AUX_SEL	In	PD-1Meg	CMOS 1.8V	-	-
P108	GPIO0/CAM0_PWR#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P109	GPIO1/CAM1_PWR#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P110	GPIO2/CAM0_RST#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P111	GPIO3/CAM1_RST#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P112	GPIO4/HDA_RST#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P113	GPIO5/PWM_OUT	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P114	GPIO6/TACHIN	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P115	GPIO7	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P116	GPIO8	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P117	GPIO9	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P118	GPIO10	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P119	GPIO11	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P120	GND	-	-	-	-	GND
P121	I2C_PM_CK	Bi-Dir	PU-4k7	CMOS 1.8V	iMX8	V_1V8

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
P122	I2C_PM_DAT	Bi-Dir	PU-4k7	CMOS 1.8V	iMX8	V_1V8
P123	BOOT_SEL0#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
P124	BOOT_SEL1#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
P125	BOOT_SEL2#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
P126	RESET_OUT#	Out –OD	-	CMOS 1.8V	SN74AUP1G125DRY	-
P127	RESET_IN#	In	PU-10k + Buffer	CMOS 1.8V	Buffer	V_1V8_SCU
P128	POWER_BTN#	In	PU-100k	CMOS 1.8V	iMX8	V_1V8_SNVS
P129	SER0_TX	Out	-	CMOS 1.8V	iMX8	-
P130	SER0_RX	In	-	CMOS 1.8V	iMX8	-
P131	SER0_RTS#	Out	-	CMOS 1.8V	iMX8	-
P132	SER0_CTS#	In	-	CMOS 1.8V	iMX8	-
P133	GND	-	-	-	-	GND
P134	SER1_TX	Out	-	CMOS 1.8V	iMX8	-
P135	SER1_RX	In	-	CMOS 1.8V	iMX8	-
P136	SER2_TX	Out	-	CMOS 1.8V	iMX8	-
P137	SER2_RX	In	-	CMOS 1.8V	iMX8	-
P138	SER2_RTS#	Out		CMOS 1.8V	iMX8	
P139	SER2_CTS#	In	-	CMOS 1.8V	iMX8	-
P140	SER3_TX	Out	-	CMOS 1.8V	iMX8	-
P141	SER3_RX	In	-	CMOS 1.8V	iMX8	-
P142	GND	-	-	-	-	GND
P143	CAN0_TX	Out	-	CMOS 1.8V	iMX8	-
P144	CAN0_RX	In	-	CMOS 1.8V	iMX8	-
P145	CAN1_TX	Out	-	CMOS 1.8V	iMX8	-
P146	CAN1_RX	In	-	CMOS 1.8V	iMX8	-
P147	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P148	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P149	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P150	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P151	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P152	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P153	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P154	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P155	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P156	VDD_IN	PWR	-	-	-	3.3V - 5.25V

5.7.2. Pinout of SMARC sAMX8 Bottom Side Connector **(SMARC 2.0 pinout)**

Table 10: Pinout of SMARC sAMX8 Bottom Side Connector

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
S1	I2C_CAM1_CK	Bi-Dir	PU-4K7	CMOS 1.8V	iMX8	V_1V8
S2	I2C_CAM1_DAT	Bi-Dir	PU-4K7	CMOS 1.8V	iMX8	V_1V8
S3	GND	-	-	-	-	GND
S4	RSVD	-	NC	-	-	-
S5	I2C_CAM0_CK	Bi-Dir	PU-4K7	CMOS 1.8V	iMX8	V_1V8
S6	CAM_MCK	Out	-	CMOS 1.8V	iMX8	-
S7	I2C_CAM0_DAT	Bi-Dir	PU-4K7	CMOS 1.8V	iMX8	V_1V8
S8	CSI0_CK+	In	-	LVDS D-PHY	iMX8	-
S9	CSI0_CK-	In	-	LVDS D-PHY	iMX8	-
S10	GND	-	-	-	-	GND
S11	CSI0_RX0+	In	-	LVDS D-PHY	iMX8	-
S12	CSI0_RX0-	In	-	LVDS D-PHY	iMX8	-
S13	GND	-	-	-	-	GND
S14	CSI0_RX1+	In	-	LVDS D-PHY	iMX8	-
S15	CSI0_RX1-	In	-	LVDS D-PHY	iMX8	-
S16	GND	-	-	-	-	GND
S17	GBE1_MDI0+	Bi-Dir	-	GBE MDI	AR8031	-
S18	GBE1_MDI0-	Bi-Dir	-	GBE MDI	AR8031	-
S19	GBE1_LINK100#	Out/OD	-	CMOS 3.3V	AR8031	-
S20	GBE1_MDI1+	Bi-Dir	-	GBE MDI	AR8031	-
S21	GBE1_MDI1-	Bi-Dir	-	GBE MDI	AR8031	-
S22	GBE1_LINK1000#	Out/OD	-	CMOS 3.3V	AR8031	-
S23	GBE1_MDI2+	Bi-Dir	-	GBE MDI	AR8031	-
S24	GBE1_MDI2-	Bi-Dir	-	GBE MDI	AR8031	-
S25	GND	-	-	-	-	GND
S26	GBE1_MDI3+	Bi-Dir	-	GBE MDI	AR8031	-
S27	GBE1_MDI3-	Bi-Dir	-	GBE MDI	AR8031	-
S28	GBE1_CTREF	Out	NC	GBE MDI	-	-
S29	PCIE_D_TX+	Out	NC	LVDS PCIe	-	-
S30	PCIE_D_TX-	Out	NC	LVDS PCIe	-	-
S31	GBE1_LINK_ACT#	Out/OD	-	CMOS 3.3V	AR8031	-
S32	PCIE_D_RX+	In	NC	LVDS PCIe	-	-
S33	PCIE_D_RX-	In	NC	LVDS PCIe	-	-
S34	GND	-	-	-	-	GND
S35	USB4+	Bi-Dir	-	USB	USB4604	-
S36	USB4-	Bi-Dir	-	USB	USB4604	
S37	USB3_VBUS_DET	In	-	USB VBUS 5V	iMX8	-
S38	AUDIO_MCK	Out	-	CMOS 1.8V	iMX8	
S39	I2S0_LRCK	Bi-Dir	-	CMOS 1.8V	iMX8	-
S40	I2S0_SDOUT	Out	-	CMOS 1.8V	iMX8	-
S41	I2S0_SDIN	In	PD-10k	CMOS 1.8V	iMX8	-
S42	I2S0_CK	Bi-Dir	-	CMOS 1.8V	iMX8	-
S43	ESPI_ALERT0#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
S44	ESPI_ALERT1#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
S45	RSVD	-	-	-	-	-
S46	RSVD	-	-	-	-	-
S47	GND	-	-	-	-	GND
S48	I2C_GP_CK	Out	PU-4k7	CMOS 1.8V	iMX8	V_1V8
S49	I2C_GP_DAT	Bi-Dir	PU-4k7	CMOS 1.8V	iMX8	V_1V8
S50	I2S2_LRCK	Bi-Dir	-	CMOS 1.8V	iMX8	-
S51	I2S2_SDOUT	Out	-	CMOS 1.8V	iMX8	-
S52	I2S2_SDIN	In	PD-10k	CMOS 1.8V	iMX8	-
S53	I2S2_CK	Bi-Dir	-	CMOS 1.8V	iMX8	-
S54	SATA_ACT#	Out/OD	NC	CMOS 3.3V	-	-
S55	USB5_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	USB4604	V_3V3
S56	ESPI_IO_2	Bi-Dir	Serial-33R	CMOS 1.8V	iMX8	-
S57	ESPI_IO_3	Bi-Dir	Serial-33R	CMOS 1.8V	iMX8	-
S58	ESPI_RESET#	Out	Serial-0R	CMOS 1.8V	SN74AUP1G08DRYR	-
S59	USB5+	Bi-Dir	-	USB	USB4604	-
S60	USB5-	Bi-Dir	-	USB	USB4604	-
S61	GND	-	-	-	-	GND
S62	USB3_SSTX+	Out	Serial-100n	USB SS	iMX8	-
S63	USB3_SSTX-	Out	Serial-100n	USB SS	iMX8	-
S64	GND	-	-	-	-	GND
S65	USB3_SSRX+	In	-	USB SS	iMX8	-
S66	USB3_SSRX-	In	-	USB SS	iMX8	-
S67	GND	-	-	-	-	GND
S68	USB3+	Bi-Dir	-	USB	iMX8	-
S69	USB3-	Bi-Dir	-	USB	iMX8	-
S70	GND	-	-	-	-	GND
S71	USB2_SSTX+	Out	NC	USB SS	-	-
S72	USB2_SSTX-	Out	NC	USB SS	-	-
S73	GND	-	-	-	-	GND
S74	USB2_SSRX+	In	NC	USB SS	-	-
S75	USB2_SSRX-	In	NC	USB SS	-	-
S76	PCIE_B_RST#	Out	-	CMOS 3.3V	iMX8	-
S77	PCIE_C_RST#	Out	-	CMOS 3.3V	iMX8	-
S78	PCIE_C_RX+	In	-	LVDS PCIe	iMX8	-
S79	PCIE_C_RX-	In	-	LVDS PCIe	iMX8	-
S80	GND	-	-	-	-	GND
S81	PCIE_C_TX+	Out	Serial-220n	LVDS PCIe	iMX8	-
S82	PCIE_C_TX-	Out	Serial-220n	LVDS PCIe	iMX8	-
S83	GND	-	-	-	-	GND
S84	PCIE_B_REFCK+	Out	Serial-0R	LVDS PCIe	9FGV0441AKILF	-
S85	PCIE_B_REFCK-	Out	Serial-0R	LVDS PCIe	9FGV0441AKILF	-
S86	GND	-	-	-	-	GND
S87	PCIE_B_RX+	In	-	LVDS PCIe	iMX8	-
S88	PCIE_B_RX-	In	-	LVDS PCIe	iMX8	-
S89	GND	-	-	-	-	GND
S90	PCIE_B_TX+	Out	Serial-220n	LVDS PCIe	iMX8	-
S91	PCIE_B_TX-	Out	Serial-220n	LVDS PCIe	iMX8	-

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
S92	GND	-	-	-	-	GND
S93	DP0_LANE0+	Out	-	LVDS PCIE	SN65DSI86	-
S94	DP0_LANE0-	Out	-	LVDS PCIE	SN65DSI86	-
S95	DP0_AUX_SEL	In	PD-1Meg	CMOS 1.8V	-	-
S96	DP0_LANE1+	Out	-	LVDS PCIE	SN65DSI86	-
S97	DP0_LANE1-	Out	-	LVDS PCIE	SN65DSI86	-
S98	DP0_HPD	In	-	CMOS 1.8V	SN65DSI86	-
S99	DP0_LANE2+	Out	-	LVDS PCIE	SN65DSI86	-
S100	DP0_LANE2-	Out	-	LVDS PCIE	SN65DSI86	-
S101	GND	-	-	-	-	GND
S102	DP0_LANE3+	Out	-	LVDS PCIE	SN65DSI86	-
S103	DP0_LANE3-	Out	-	LVDS PCIE	SN65DSI86	-
S104	USB3_OTG_ID	In	PD-10K	CMOS 3.3V	i.MX8	-
S105	DP0_AUX+	Bi-Dir	Serial-100n PD-100k	LVDS PCIE	SN65DSI86	-
S106	DP0_AUX-	Bi-Dir	Serial-100n PU-100k	LVDS PCIE	SN65DSI86	-
S107	LCD1_BKLT_EN	Out	-	CMOS 1.8V	iMX8	-
S108	LVDS1_CK+/ DSI1_CLK+	Out	Serial-0R	LVDS LCD	iMX8	-
S109	LVDS1_CK/ DSI1_CLK-	Out	Serial-0R	LVDS LCD	iMX8	-
S110	GND	-	-	-	-	GND
S111	LVDS1_0+/ DSI1_D0+	Out	Serial-0R	LVDS LCD	iMX8	-
S112	LVDS1_0-/ DSI1_D0-	Out	Serial-0R	LVDS LCD	iMX8	-
S113	eDP1_HPD	In	NC	CMOS 1.8V	NC	-
S114	LVDS1_1+/ DSI1_D1+	Out	Serial-0R	LVDS LCD	iMX8	
S115	LVDS1_1-/ DSI1_D1-	Out	Serial-0R	LVDS LCD	iMX8	-
S116	LCD1_VDD_EN	Out	-	CMOS 1.8V	iMX8	-
S117	LVDS1_2+/ DSI1_D2+	Out	Serial-0R	LVDS LCD	iMX8	-
S118	LVDS1_2-/ DSI1_D2-	Out	Serial-0R	LVDS LCD	iMX8	-
S119	GND	-	-	-	-	GND
S120	LVDS1_3+/ DSI1_D3+	Out	Serial-0R	LVDS LCD	iMX8	-
S121	LVDS1_3-/ DSI1_D3-	Out	Serial-0R	LVDS LCD	iMX8	-
S122	LCD1_BKLT_PWM	Out	-	CMOS 1.8V	iMX8	-
S123	RSVD	-	NC	-	NC	-
S124	GND	-	-	-	-	GND
S125	LVDS0_0+	Out	Serial-0R	LVDS LCD	iMX8	-
S126	LVDS0_0-	Out	Serial-0R	LVDS LCD	iMX8	-
S127	LCD0_BKLT_EN	Out	-	CMOS 1.8V	iMX8	-
S128	LVDS0_1+	Out	Serial-0R	LVDS LCD	iMX8	-
S129	LVDS0_1-	Out	Serial-0R	LVDS LCD	iMX8	-
S130	GND	-	-	-	-	GND

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
S131	LVDS0_2+	Out	Serial-0R	LVDS LCD	iMX8	-
S132	LVDS0_2-	Out	Serial-0R	LVDS LCD	iMX8	-
S133	LCD0_VDD_EN	Out	-	CMOS 1.8V	iMX8	-
S134	LVDS0_CK+	Out	Serial-0R	LVDS LCD	iMX8	-
S135	LVDS0_CK-	Out	Serial-0R	LVDS LCD	iMX8	-
S136	GND	-	-	-	-	GND
S137	LVDS0_3+	Out	Serial-0R	LVDS LCD	iMX8	-
S138	LVDS0_3-	Out	Serial-0R	LVDS LCD	iMX8	-
S139	I2C_LCD_CK	Out	PU-4k7	CMOS 1.8V	iMX8	V_1V8
S140	I2C_LCD_DAT	Bi-Dir	PU-4k7	CMOS 1.8V	iMX8	V_1V8
S141	LCD0_BKLT_PWM	Out	-	CMOS 1.8V	iMX8	-
S142	RSVD	-	NC	-	-	-
S143	GND	-	-	-	-	GND
S144	EDP0_HPD	In	NC	CMOS 1.8V	-	-
S145	WDT_TIME_OUT#	Out	-	CMOS 1.8V	Buffers	-
S146	PCIE_WAKE#	In	PU-10k	CMOS 3.3V	iMX8	V_3V3
S147	VDD_RTC	-	Diode and Measurement circuit	PWR	RV-8803 and iMX8	V_3V3_RTC (2.0V-3.25V)
S148	LID#	In	Serial-0R PU-10k	CMOS 1.8V	iMX8	V_1V8_SCU
S149	SLEEP#	In	Serial-0R PU-10k	CMOS 1.8V	iMX8	V_1V8_SCU
S150	VIN_PWR_BAD#	In	PU-10k	CMOS VDD_IN	TPS3808	Input power V_3V0-5V25_IN
S151	CHARGING#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
S152	CHARGER_PRSNT#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
S153	CARRIER_STBY#	Out	PU-10k	CMOS 1.8V	iMX8	V_1V8
S154	CARRIER_PWR_ON	Out	PD-100k	CMOS 1.8V	SN74AUP2G125	-
S155	FORCE_RECOV#	In	PU-4k7	CMOS 1.8V	iMX8	V_1V8_SCU
S156	BATLOW#	In	PU_10k	CMOS 1.8V	iMX8	V_1V8
S157	TEST#	In	NC	CMOS 1.8V	NC	-
S158	GND	-	-	-	-	GND

5.7.3. Pinout of SMARC sAMX8 Topside Connector **(Custom SMARC pinout)**

Note: Pin Type/Tolerance definition is according to SMARC Specification 2.0

Table 11: Pinout of SMARC sAMX8 Topside Connector

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
P1	SMB_ALERT_1V8#	In	-	CMOS 1.8V	iMX8	
P2	GND	-	-	-	-	GND
P3	CSI1_CK+	In	-	LVDS D-PHY	iMX8	-
P4	CSI1_CK-	In	-	LVDS D-PHY	iMX8	-
P5	GBE1_SDP	Bi-Dir	-	CMOS 3.3V	AR8031 (CLK_25M)	-
P6	GBE0_SDP	Bi-Dir	-	CMOS 3.3V	AR8031 (CLK_25M)	-
P7	CSI1_RX0+	In	-	LVDS D-PHY	iMX8	-
P8	CSI1_RX0-	In	-	LVDS D-PHY	iMX8	-
P9	GND	-	-	-	-	GND
P10	CSI1_RX1+	In	-	LVDS D-PHY	iMX8	-
P11	CSI1_RX1-	In	-	LVDS D-PHY	iMX8	-
P12	GND	-	-	-	-	GND
P13	CSI1_RX2+	In	-	LVDS D-PHY	iMX8	-
P14	CSI1_RX2-	In	-	LVDS D-PHY	iMX8	-
P15	GND	-	-	-	-	GND
P16	CSI1_RX3+	In	-	LVDS D-PHY	iMX8	-
P17	CSI1_RX3-	In	-	LVDS D-PHY	iMX8	-
P18	GND	-	-	-	-	GND
P19	GBE0_MDI3-	Bi-Dir	-	GBE MDI	AR8031	-
P20	GBE0_MDI3+	Bi-Dir	-	GBE MDI	AR8031	-
P21	GBE0_LINK100#	Out/OD	-	CMOS 3.3V	AR8031	-
P22	GBE0_LINK1000#	Out/OD	-	CMOS 3.3V	AR8031	-
P23	GBE0_MDI2-	Bi-Dir	-	GBE MDI	AR8031	-
P24	GBE0_MDI2+	Bi-Dir	-	GBE MDI	AR8031	-
P25	GBE0_LINK_ACT#	Out/OD	-	CMOS 3.3V	AR8031	-
P26	GBE0_MDI1-	Bi-Dir	-	GBE MDI	AR8031	-
P27	GBE0_MDI1+	Bi-Dir	-	GBE MDI	AR8031	-
P28	GBE0_CTREF	Out	NC	-	-	-
P29	GBE0_MDI0-	Bi-Dir	-	GBE MDI	AR8031	-
P30	GBE0_MDI0+	Bi-Dir	-	GBE MDI	AR8031	-
P31	SPI0_CS1#	Out	Serial–33R	CMOS 1.8V	iMX8	-
P32	GND	-	-	-	-	GND
P33	SDIO_WP	In	PU-10k	CMOS 3.3V	iMX8	V_3V3
P34	SDIO_CMD	Bi-Dir	-	CMOS 3.3V	iMX8	-
P35	SDIO_CD#	In	PU-10k	CMOS 3.3V	iMX8	V_3V3
P36	SDIO_CK	Out	-	CMOS 3.3V	iMX8	-
P37	SDIO_PWR_EN	Out	-	CMOS 3.3V	iMX8	-
P38	GND	-	-	-	-	GND
P39	SDIO_D0	Bi-Dir	-	CMOS 3.3V	iMX8	-
P40	SDIO_D1	Bi-Dir	-	CMOS 3.3V	iMX8	-
P41	SDIO_D2	Bi-Dir	-	CMOS 3.3V	iMX8	-
P42	SDIO_D3	Bi-Dir	-	CMOS 3.3V	iMX8	-
P43	SPI0_CS0#	Out	Serial-33R	CMOS 1.8V	iMX8	-

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
P44	SPI0_CK	Out	Serial-33R	CMOS 1.8V	iMX8	-
P45	SPI0_DIN	In	Serial-33R	CMOS 1.8V	iMX8	-
P46	SPI0_DO	Out	Serial-33R	CMOS 1.8V	iMX8	-
P47	GND	-	-	-	-	GND
P48	SATA_TX+	Out	Serial-10n	SATA	iMX8	-
P49	SATA_TX-	Out	Serial-10n	SATA	iMX8	-
P50	GND	-	-	-	-	GND
P51	SATA_RX+	In	Serial-10n	SATA	iMX8	-
P52	SATA_RX-	In	Serial-10n	SATA	iMX8	-
P53	GND	-	-	-	-	GND
P54	ESPI_CS0#	Out	Serial-33R	CMOS 1.8V	iMX8	-
P55	ESPI_CS1#	Out	Serial-33R	CMOS 1.8V	iMX8	-
P56	ESPI_CK	Out	Serial-33R	CMOS 1.8V	iMX8	-
P57	ESPI_IO_1	Bi-Dir	Serial-33R	CMOS 1.8V	iMX8	-
P58	ESPI_IO_0	Bi-Dir	Serial-33R	CMOS 1.8V	iMX8	-
P59	GND	-	-	-	-	GND
P60	USB0+	Bi-Dir	-	USB	iMX8	-
P61	USB0-	Bi-Dir	-	USB	iMX8	-
P62	USB0_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	iMX8	V_3V3
P63	USB0_VBUS_DET	In	PD-4k7	USB VBUS 5V	iMX8	-
P64	USB0_OTG_ID	In	-	CMOS 3.3V	iMX8	-
P65	USB1+	Bi-Dir	-	USB	USB4604	-
P66	USB1-	Bi-Dir	-	USB	USB4604	-
P67	USB1_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	USB4604	V_3V3
P68	GND	-	-	-	-	GND
P69	USB2+	Bi-Dir	-	USB	USB4604	-
P70	USB2-	Bi-Dir	-	USB	USB4604	-
P71	USB2_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	USB4604	V_3V3
P72	RSVD	-	NC	-	-	-
P73	RSVD	-	NC	-	-	-
P74	USB3_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	iMX8	V_3V3
P75	PCIE_A_RST#	Out	-	CMOS 3.3V	iMX8	-
P76	USB4_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	USB4604	V_3V3
P77	RSVD	-	NC	-	-	-
P78	RSVD	-	NC	-	-	-
P79	GND	-	-	-	-	GND
P80	PCIE_C_REFCK+	Out	-	LVDS PCIe	9FGV0441AKILF	-
P81	PCIE_C_REFCK-	Out	-	LVDS PCIe	9FGV0441AKILF	-
P82	GND	-	-	-	-	GND
P83	PCIE_A_REFCK+	Out	-	LVDS PCIe	9FGV0441AKILF	-
P84	PCIE_A_REFCK-	Out	-	LVDS PCIe	9FGV0441AKILF	-
P85	GND	-	-	-	-	GND
P86	PCIE_A_RX+	In	-	LVDS PCIe	iMX8	-
P87	PCIE_A_RX-	In	-	LVDS PCIe	iMX8	-
P88	GND	-	-	-	-	GND

P89	PCIE_A_TX+	Out	Serial-220n	LVDS PCIe	iMX8	-
P90	PCIE_A_TX-	Out	Serial-220n	LVDS PCIe	iMX8	-
P91	GND	-	-	-	-	GND
P92	HDMI_D2+/ DP1_LANE0+	Out	-	TMDS	iMX8	-
P93	HDMI_D2/ DP1_LANE0-	Out	-	TMDS	iMX8	-
P94	GND	-	-	-	-	GND
P95	HDMI_D1+/ DP1_LANE1+	Out	-	TMDS	iMX8	-
P96	HDMI_D1- /DP1_LANE1-	Out	-	TMDS	iMX8	-
P97	GND	-	-	-	-	GND
P98	HDMI_D0+/ DP1_LANE2+	Out	-	TMDS	iMX8	-
P99	HDMI_D0/ DP1_LANE2-	Out	-	TMDS	iMX8	-
P100	GND	-	-	-	-	GND
P101	HDMI_CK+ /DP1_LANE3+	Out	-	TMDS	iMX8	-
P102	HDMI_CK /DP1_LANE3-	Out	-	TMDS	iMX8	-
P103	GND	-	-	-	-	GND
P104	HDMI_HPD/DP1_HPD	In	PD-100k	CMOS 1.8V	iMX8	-
P105	HDMI_CTRL_CK/ DP1_AUX+	Bi-Dir	PU-2k2	CMOS 1.8V	iMX8	V_1V8
P106	HDMI_CTRL_DAT/ DP1_AUX-	Bi-Dir	PU-2k2	CMOS 1.8V	iMX8	V_1V8
P107	DP1_AUX_SEL	In	PD-1Meg	CMOS 1.8V	-	-
P108	GPIO0/CAM0_PWR#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P109	GPIO1/CAM1_PWR#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P110	GPIO2/CAM0_RST#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P111	GPIO3/CAM1_RST#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P112	GPIO4/HDA_RST#	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P113	GPIO5/PWM_OUT	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P114	GPIO6/TACHIN	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P115	GPIO7	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P116	GPIO8	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P117	GPIO9	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P118	GPIO10	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P119	GPIO11	Bi-Dir	PU-100k	CMOS 1.8V	iMX8	V_1V8
P120	GND	-	-	-	-	GND
P121	I2C_PM_CK	Bi-Dir	PU-4k7	CMOS 1.8V	iMX8	V_1V8
P122	I2C_PM_DAT	Bi-Dir	PU-4k7	CMOS 1.8V	iMX8	V_1V8
P123	BOOT_SEL0#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
P124	BOOT_SEL1#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
P125	BOOT_SEL2#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
P126	RESET_OUT#	Out –OD	-	CMOS 1.8V	SN74AUP1G125DRY	-
P127	RESET_IN#	In	PU-10k + Buffer	CMOS 1.8V	Buffer	V_1V8_SCU
P128	POWER_BTN#	In	PU-100k	CMOS 1.8V	iMX8	V_1V8_SNVS
P129	SER0_TX	Out	-	CMOS 1.8V	iMX8	-
P130	SER0_RX	In	-	CMOS 1.8V	iMX8	-
P131	SER0_RTS#	Out	-	CMOS 1.8V	iMX8	-

P132	SER0_CTS#	In	-	CMOS 1.8V	iMX8	-
P133	GND	-	-	-	-	GND
P134	SER1_TX	Out	-	CMOS 1.8V	iMX8	-
P135	SER1_RX	In	-	CMOS 1.8V	iMX8	-
P136	SER2_TX	Out	-	CMOS 1.8V	iMX8	-
P137	SER2_RX	In	-	CMOS 1.8V	iMX8	-
P138	SER2_RTS#	Out		CMOS 1.8V	iMX8	
P139	SER2_CTS#	In	-	CMOS 1.8V	iMX8	-
P140	SER3_TX	Out	-	CMOS 1.8V	iMX8	-
P141	SER3_RX	In	-	CMOS 1.8V	iMX8	-
P142	GND	-	-	-	-	GND
P143	CAN0_TX	Out	-	CMOS 1.8V	iMX8	-
P144	CAN0_RX	In	-	CMOS 1.8V	iMX8	-
P145	CAN1_TX	Out	-	CMOS 1.8V	iMX8	-
P146	CAN1_RX	In	-	CMOS 1.8V	iMX8	-
P147	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P148	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P149	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P150	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P151	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P152	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P153	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P154	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P155	VDD_IN	PWR	-	-	-	3.3V - 5.25V
P156	VDD_IN	PWR	-	-	-	3.3V - 5.25V

5.7.4. Pinout of SMARC sAMX8 Bottom Side Connector **(Custom SMARC pinout)**

Table 12: Pinout of SMARC sAMX8 Bottom Side Connector

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
S1	I2C_CAM1_CK	Bi-Dir	PU-4K7	CMOS 1.8V	iMX8	V_1V8
S2	I2C_CAM1_DAT	Bi-Dir	PU-4K7	CMOS 1.8V	iMX8	V_1V8
S3	GND	-	-	-	-	GND
S4	RSVD	-	NC	-	-	-
S5	I2C_CAM0_CK	Bi-Dir	PU-4K7	CMOS 1.8V	iMX8	V_1V8
S6	CAM_MCK	Out	-	CMOS 1.8V	iMX8	-
S7	I2C_CAM0_DAT	Bi-Dir	PU-4K7	CMOS 1.8V	iMX8	V_1V8
S8	CSI0_CK+	In	-	LVDS D-PHY	iMX8	-
S9	CSI0_CK-	In	-	LVDS D-PHY	iMX8	-
S10	GND	-	-	-	-	GND
S11	CSI0_RX0+	In	-	LVDS D-PHY	iMX8	-
S12	CSI0_RX0-	In	-	LVDS D-PHY	iMX8	-
S13	GND	-	-	-	-	GND
S14	CSI0_RX1+	In	-	LVDS D-PHY	iMX8	-
S15	CSI0_RX1-	In	-	LVDS D-PHY	iMX8	-
S16	GND	-	-	-	-	GND
S17	GBE1_MDI0+	Bi-Dir	-	GBE MDI	AR8031	-
S18	GBE1_MDI0-	Bi-Dir	-	GBE MDI	AR8031	-
S19	GBE1_LINK100#	Out/OD	-	CMOS 3.3V	AR8031	-
S20	GBE1_MDI1+	Bi-Dir	-	GBE MDI	AR8031	-
S21	GBE1_MDI1-	Bi-Dir	-	GBE MDI	AR8031	-
S22	GBE1_LINK1000#	Out/OD	-	CMOS 3.3V	AR8031	-
S23	GBE1_MDI2+	Bi-Dir	-	GBE MDI	AR8031	-
S24	GBE1_MDI2-	Bi-Dir	-	GBE MDI	AR8031	-
S25	GND	-	-	-	-	GND
S26	GBE1_MDI3+	Bi-Dir	-	GBE MDI	AR8031	-
S27	GBE1_MDI3-	Bi-Dir	-	GBE MDI	AR8031	-
S28	GBE1_CTREF	Out	NC	GBE MDI	-	-
S29	PCIE_D_TX+	Out	NC	LVDS PCIe	-	-
S30	PCIE_D_TX-	Out	NC	LVDS PCIe	-	-
S31	GBE1_LINK_ACT#	Out/OD	-	CMOS 3.3V	AR8031	-
S32	PCIE_D_RX+	In	NC	LVDS PCIe	-	-
S33	PCIE_D_RX-	In	NC	LVDS PCIe	-	-
S34	GND	-	-	-	-	GND
S35	USB4+	Bi-Dir	-	USB	USB4604	-
S36	USB4-	Bi-Dir	-	USB	USB4604	
S37	USB3_VBUS_DET	In	-	USB VBUS 5V	iMX8	-
S38	AUDIO_MCK	Out	-	CMOS 1.8V	iMX8	
S39	I2S0_LRCK	Bi-Dir	-	CMOS 1.8V	iMX8	-
S40	I2S0_SDOUT	Out	-	CMOS 1.8V	iMX8	-
S41	I2S0_SDIN	In	PD-10k	CMOS 1.8V	iMX8	-
S42	I2S0_CK	Bi-Dir	-	CMOS 1.8V	iMX8	-
S43	ESPI_ALERT0#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
S44	ESPI_ALERT1#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
S45	RSVD	-	-	-	-	-
S46	RSVD	-	-	-	-	-
S47	GND	-	-	-	-	GND
S48	I2C_GP_CK	Out	PU-4k7	CMOS 1.8V	iMX8	V_1V8
S49	I2C_GP_DAT	Bi-Dir	PU-4k7	CMOS 1.8V	iMX8	V_1V8
S50	I2S2_LRCK	Bi-Dir	-	CMOS 1.8V	iMX8	-
S51	I2S2_SDOUT	Out	-	CMOS 1.8V	iMX8	-
S52	I2S2_SDIN	In	PD-10k	CMOS 1.8V	iMX8	-
S53	I2S2_CK	Bi-Dir	-	CMOS 1.8V	iMX8	-
S54	SATA_ACT#	Out/OD	NC	CMOS 3.3V	-	-
S55	USB5_EN_OC#	Bi-Dir OD	PU-10k	CMOS 3.3V	USB4604	V_3V3
S56	ESPI_IO_2	Bi-Dir	Serial-33R	CMOS 1.8V	iMX8	-
S57	ESPI_IO_3	Bi-Dir	Serial-33R	CMOS 1.8V	iMX8	-
S58	ESPI_RESET#	Out	Serial-0R	CMOS 1.8V	SN74AUP1G08DRYR	-
S59	USB5+	Bi-Dir	-	USB	USB4604	-
S60	USB5-	Bi-Dir	-	USB	USB4604	-
S61	GND	-	-	-	-	GND
S62	USB3_SSTX+	Out	Serial-100n	USB SS	iMX8	-
S63	USB3_SSTX-	Out	Serial-100n	USB SS	iMX8	-
S64	GND	-	-	-	-	GND
S65	USB3_SSRX+	In	-	USB SS	iMX8	-
S66	USB3_SSRX-	In	-	USB SS	iMX8	-
S67	GND	-	-	-	-	GND
S68	USB3+	Bi-Dir	-	USB	iMX8	-
S69	USB3-	Bi-Dir	-	USB	iMX8	-
S70	GND	-	-	-	-	GND
S71	USB2_SSTX+	Out	NC	USB SS	-	-
S72	USB2_SSTX-	Out	NC	USB SS	-	-
S73	GND	-	-	-	-	GND
S74	USB2_SSRX+	In	NC	USB SS	-	-
S75	USB2_SSRX-	In	NC	USB SS	-	-
S76	PCIE_B_RST#	Out	-	CMOS 3.3V	iMX8	-
S77	PCIE_C_RST#	Out	-	CMOS 3.3V	iMX8	-
S78	PCIE_C_RX+	In	-	LVDS PCIe	iMX8	-
S79	PCIE_C_RX-	In	-	LVDS PCIe	iMX8	-
S80	GND	-	-	-	-	GND
S81	PCIE_C_TX+	Out	Serial-220n	LVDS PCIe	iMX8	-
S82	PCIE_C_TX-	Out	Serial-220n	LVDS PCIe	iMX8	-
S83	GND	-	-	-	-	GND
S84	PCIE_B_REFCK+	Out	Serial-0R	LVDS PCIe	9FGV0441AKILF	-
S85	PCIE_B_REFCK-	Out	Serial-0R	LVDS PCIe	9FGV0441AKILF	-
S86	GND	-	-	-	-	GND
S87	PCIE_B_RX+	In	-	LVDS PCIe	iMX8	-
S88	PCIE_B_RX-	In	-	LVDS PCIe	iMX8	-
S89	GND	-	-	-	-	GND
S90	PCIE_B_TX+	Out	Serial-220n	LVDS PCIe	iMX8	-
S91	PCIE_B_TX-	Out	Serial-220n	LVDS PCIe	iMX8	-

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
S92	GND	-	-	-	-	GND
S93	DP0_LANE0+	Out	-	LVDS PCIE	SN65DSI86	-
S94	DP0_LANE0-	Out	-	LVDS PCIE	SN65DSI86	-
S95	DP0_AUX_SEL	In	PD-1Meg	CMOS 1.8V	-	-
S96	DP0_LANE1+	Out	-	LVDS PCIE	SN65DSI86	-
S97	DP0_LANE1-	Out	-	LVDS PCIE	SN65DSI86	-
S98	DP0_HPD	In	-	CMOS 1.8V	SN65DSI86	-
S99	DP0_LANE2+	Out	-	LVDS PCIE	SN65DSI86	-
S100	DP0_LANE2-	Out	-	LVDS PCIE	SN65DSI86	-
S101	GND	-	-	-	-	GND
S102	DP0_LANE3+	Out	-	LVDS PCIE	SN65DSI86	-
S103	DP0_LANE3-	Out	-	LVDS PCIE	SN65DSI86	-
S104	USB3_OTG_ID	In	PD-10K	CMOS 3.3V	i.MX8	-
S105	DP0_AUX+	Bi-Dir	Serial-100n PD-100k	LVDS PCIE	SN65DSI86	-
S106	DP0_AUX-	Bi-Dir	Serial-100n PU-100k	LVDS PCIE	SN65DSI86	-
S107	LCD1_BKLT_EN	Out	-	CMOS 1.8V	iMX8	-
S108	SPI3_SCK / DSI1_CLK+	Out	Serial-33R	LVDS LCD	iMX8	-
S109	SPI3_CS0/ DSI1_CLK-	Out	Serial-33R	LVDS LCD	iMX8	-
S110	GND	-	-	-	-	GND
S111	SPI3_SDO/ DSI1_D0+	Out	Serial-33R	LVDS LCD	iMX8	-
S112	SPI3_SDI/ DSI1_D0-	Out	Serial-33R	LVDS LCD	iMX8	-
S113	eDP1_HPD	In	NC	CMOS 1.8V	NC	-
S114	SPI3_CS1/ DSI1_D1+	Out	Serial-33R	LVDS LCD	iMX8	
S115	NC/ DSI1_D1-	Out	-	LVDS LCD	iMX8	-
S116	LCD1_VDD_EN	Out	-	CMOS 1.8V	iMX8	-
S117	NC/ DSI1_D2+	Out	-	LVDS LCD	iMX8	-
S118	NC/ DSI1_D2-	Out	-	LVDS LCD	iMX8	-
S119	GND	-	-	-	-	GND
S120	CAN2_TX/ DSI1_D3+	Out	Serial-0R	LVDS LCD	iMX8	-
S121	CAN2_RX/ DSI1_D3-	Out	Serial-0R	LVDS LCD	iMX8	-
S122	LCD1_BKLT_PWM	Out	-	CMOS 1.8V	iMX8	-
S123	RSVD	-	NC	-	NC	-
S124	GND	-	-	-	-	GND
S125	SPI1_SDO	Out	Serial-33R	LVDS LCD	iMX8	-
S126	SPI1_SDI	Out	Serial-33R	LVDS LCD	iMX8	-
S127	LCD0_BKLT_EN	Out	-	CMOS 1.8V	iMX8	-
S128	SPI1_CS1	Out	Serial-33R	LVDS LCD	iMX8	-
S129	SPI2_CS1	Out	Serial-33R	LVDS LCD	iMX8	-
S130	GND	-	-	-	-	GND

Pin	Signal	Module Direction	Module Termination	Type/ Tolerance	Controller	Power Rail
S131	SPI2_SCK	Out	Serial-33R	LVDS LCD	iMX8	-
S132	SPI2_CS0	Out	Serial-33R	LVDS LCD	iMX8	-
S133	LCD0_VDD_EN	Out	-	CMOS 1.8V	iMX8	-
S134	SPI1_SCK	Out	Serial-33R	LVDS LCD	iMX8	-
S135	SPI1_CS0	Out	Serial-33R	LVDS LCD	iMX8	-
S136	GND	-	-	-	-	GND
S137	SPI2_SDO	Out	Serial-33R	LVDS LCD	iMX8	-
S138	SPI2_SDI	Out	Serial-33R	LVDS LCD	iMX8	-
S139	I2C_LCD_CK	Out	PU-4k7	CMOS 1.8V	iMX8	V_1V8
S140	I2C_LCD_DAT	Bi-Dir	PU-4k7	CMOS 1.8V	iMX8	V_1V8
S141	LCD0_BKLT_PWM	Out	-	CMOS 1.8V	iMX8	-
S142	RSVD	-	NC	-	-	-
S143	GND	-	-	-	-	GND
S144	EDP0_HPD	In	NC	CMOS 1.8V	-	-
S145	WDT_TIME_OUT#	Out	-	CMOS 1.8V	Buffers	-
S146	PCIE_WAKE#	In	PU-10k	CMOS 3.3V	iMX8	V_3V3
S147	VDD_RTC	-	Diode and Measurement circuit	PWR	RV-8803 and iMX8	V_3V3_RTC (2.0V-3.25V)
S148	LID#	In	Serial-0R PU-10k	CMOS 1.8V	iMX8	V_1V8_SCU
S149	SLEEP#	In	Serial-0R PU-10k	CMOS 1.8V	iMX8	V_1V8_SCU
S150	VIN_PWR_BAD#	In	PU-10k	CMOS VDD_IN	TPS3808	Input power V_3V0-5V25_IN
S151	CHARGING#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
S152	CHARGER_PRSNT#	In	PU-10k	CMOS 1.8V	iMX8	V_1V8
S153	CARRIER_STBY#	Out	PU-10k	CMOS 1.8V	iMX8	V_1V8
S154	CARRIER_PWR_ON	Out	PD-100k	CMOS 1.8V	SN74AUP2G125	-
S155	FORCE_RECOV#	In	PU-4k7	CMOS 1.8V	iMX8	V_1V8_SCU
S156	BATLOW#	In	PU_10k	CMOS 1.8V	iMX8	V_1V8
S157	TEST#	In	NC	CMOS 1.8V	NC	-
S158	GND	-	-	-	-	GND

6/ Installation

6.1. Boot Mode

The following table shows the possible boot sources on the carrier board defined in SMARC 2.0 spec:

Table 8: Boot Options on the carrier board

	Carrier Connection			Boot Source	Supported on sAMX8
	BOOT_SEL2#	BOOT_SEL1#	BOOT_SEL0#		
0	GND	GND	GND	Carrier SATA	No
1	GND	GND	Float	Carrier SD Card	From U-Boot
2	GND	Float	GND	Carrier eSPI (CS0#)	No
3	GND	Float	Float	Carrier SPI (CS0#)	No
4	Float	GND	GND	Module device (NAND, NOR)	No
5	Float	GND	Float	Remote boot (GBE, serial)	From U-Boot
6	Float	Float	GND	Module eMMC	From U-Boot
7	Float	Float	Float	Module SPI	From U-Boot

U-Boot only supports booting from QSPI NOR flash because "boot from fuses" is defined per default.

6.2. RTC Current Consumption

The module RTC (RV-8803) can operate down to 1.5 V, the i.MX8 RTC minimum supply voltage is 2.4 V. Due to the higher current consumption of the i-MX8 RTC this part is not powered with the V_RTC voltage rail.

Table 9: Minimum Supply Voltage of the RTC Module

Ambient temperature	Current in V_BAT_R	Minimum voltage for module RTC (RV-8803) V_BAT_R/V_RTC
25°C	40 µA (Max) / 240 nA (typical)	2.02V/1.69 V
85°C		2.08V/1.71 V

RV-8803 is powered through Schottky diode BAS70 from V_VDD_RTC pin S147 of SMARC connector.

6.3. UART Interfaces

Use following UART interfaces with control signals of i.MX8.

Table 10: Mapping of SMARC SER interfaces to i.MX8 UARTs

SER	UART
SER0	UART0
SER1	M40 UART
SER2	UART1
SER3	M41 UART

There are additional UART available but it will require modification to the bootloader and BSP. The pinout from SMARC will not be compatible with regular SMARC 2.0 carrier as they used pins that are defined for different functionality.

- UART2 shares UART0_RTS_B and UART0_CTS_B
- UART3 shares GPIO2/CAM0_RST# and GPIO3/CAM1_RST#
- UART4 shares LCD1_VDD_EN and LCD1_BKLT_EN

Table 11: UART connections between CPU and SMARC 2.0 connector

CPU Interface	CPU pins	SMARC 2.0 connector
UART0	UART0_TX	SER0_TX
	UART0_RX	SER0_RX
	UART0_RTS_B	SER0_RTS#
	UART0_CTS_B	SER0_CTS#
UART1	UART1_TX	SER2_TX
	UART1_RX	SER2_RX
	UART1_RTS_B	SER2_RTS#
	UART1_CTS_B	SER2_CTS#
UART2	UART0_RTS_B/UART2.RX	SER0_RTS#
	UART0_CTS_B/UART2.TX	SER0_CTS#
UART3	M41_GPIO0_00/UART3.RX	GPIO2/CAM0_RST#
	M41_GPIO0_01/UART3.TX	GPIO3/CAM1_RST#
UART4	MIPI_CSI1_GPIO0_00/UART4.RX	LCD1_VDD_EN
	MIPI_CSI1_GPIO0_01/UART4.TX	LCD1_BKLT_EN
UART M40	M40_I2C0_SCL/M40.UART0.RX	SER1_RX
	M40_I2C0_SDA/M40.UART0.TX	SER1_TX
UART M41	M41_I2C0_SLC/M41.UART0.RX	SER3_RX
	M41_I2C0_SDA/M41.UART0.TX	SER3_TX

6.4. Power Control

6.4.1. Power Supply

The SMARC-sAMX8 supports a power input from 3.3 to 5.25V. The supply voltage is applied through the VCC pins (VCC) of the module connector. Considered current rating of protective device is part of End-Equipment.

The following parameters should be delivered from the carrier board:

- Voltage Ripple maximum 200 mV peak to peak 0-20 MHz in 0 ms to 20 ms rise time from input voltage <10% to nominal VCC
- Max allowed inrush current: connector limit (15 W @ 3.3 V)

6.4.2. Power Button (POWER_BTN#)

The power button (Pin P128) is available through the module connector described in the pinout list. To start the module via Power Button the PWRBTN# signal must be at least 50 ms (50 ms ≤ t < 4 s, typical 400 ms) at low level (Power Button Event).

Pressing the power button for at least 4 seconds will turn off power to the module (Power Button Override).

6.4.3. Power Bad Signal (VIN_POWER_BAD#)

The SMARC-sAMX8 provides an external input for a Carrier Board Power Bad signal (Pin S150). The implementation of this subsystem complies with the SMARC Specification. VIN_POWER_BAD# is internally pulled up to module input voltage and must be high level (open drain) to power on the module.

6.4.4. Reset Button (RESET_IN#)

The reset button (Pin P127) is available through the module connector described in the pinout list. The module will stay in reset as long as RESET_IN# is grounded.

7/ Bootloader Operation

7.1. Copyrights and Licensing of U-Boot

U-Boot is free Software. It is copyrighted by Wolfgang Denk and many others who contributed code. U-Boot can be redistributed and modified under the terms of version 2 of the GNU General Public (GPL V2) License as published by the Free Software Foundation.

Actual source code of mainline U-Boot and authors of the source can be obtained from the git repository at

- git://git.denx.de/u-boot.git

SMARC-sAMX8 bootloader sources are derived work from a dedicated version of mainline U-Boot, e.g v2018.03. As bootloader evolves, the root of the derived work might change to a later version.

NOTICE The source code of U-Boot will be delivered with the standard software package.

7.2. Bootloader Quickstart

The SMARC-sAMX8 board comes with U-Boot preinstalled on the QSPI flash device. Follow the steps below to gain access to the bootloader command line (CLI) on your host PC.

- Connect your host machine to the carrier port connected with the edge connector SER0 port of the module. On Kontron SMARC 2.0 carrier this port is named SER_0.
- Start a suitable terminal program on your host and attach it to the port connected with the board's serial interface. Configure the serial line using **115200 baud, 8 data bits, 1 stop bit, no parity**.
- Connect power supply to the carrier and power up.
- When boot messages appear, press any key to stop automatic boot sequence.

After power on, bootloader boot messages will appear as shown below. There is a 3 second boot delay counter that will try to boot linux OS automatically after expiration. Pressing any key will stop the boot delay counter and enter the bootloader CLI

.

```
U-Boot <Version> (<Date-code>)

CPU:   Freescale i.MX8QM revB A53 at 1200 MHz at 18C
Model: Kontron SMARC-sAMX8 QuadMax
Board: Kontron sMX8 SMARC 2.0 Module
Boot:  FLEXSPI
DRAM:  6 GiB
start sata init
TX PLL is not locked.
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
Loading Environment from SPI Flash... SF: Detected mt35xu512g with page size 256 Bytes, erase size 128 KiB,
total 64 MiB
OK
TX PLL is not locked.
In:    serial
Out:   serial
Err:   serial

 BuildInfo:
  - SCFW f0226b37, SECO-FW 5ea79733, IMX-MKIMAGE 2cf091c0, ATF d6451cc
  - U-Boot 2018.03-imx_v2018.03_4.14.78_1.0.0_ga+g7ade5b4
```

```
SCSI:  Net:
Warning: ethernet@5b040000 using MAC address from ROM
eth0: ethernet@5b040000 [PRIME]
Warning: ethernet@5b050000 using MAC address from ROM
, eth1: ethernet@5b050000
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  0
=>
```

7.3. Bootloader Commands

The bootloader CLI provides a bunch of powerful commands to control the board, which can be grouped into

- Information Commands
- Memory Commands
- Flash Memory Commands
- Execution Control Commands
- Download Commands
- Environment Control Commands
- Flattened Device Tree Support Commands
- Storage Device Control Commands
- File System Support Commands
- Kontron Command Extensions

Typing “help” at the bootloader command line prompt will show up a list of the commands available. Typing “help <command>” will show specific command help. Further help can be found under https://www.denx.de/wiki/view/DULG/UBoot

On the SMARC_sAMX8 bootloader, the powerful hush shell is enabled, which is similar to Bourne shell and provides features similar to a linux shell:

- Control structures (if … then … else … fi etc.)
- Command line completion
- Command line editing
- Command line history up to 20 entries
- Local environment variables

7.4. Kontron Bootloader Command Extensions

Kontron’s implementation of U-Boot includes certain enhancements to provide board specific functions. They are not part of standard U-Boot as maintained by DENX. The following table provides a complete listing of all Kontroncommand extensions on the SMARC-sAMX8.

Table 12: Bootloader Command Extensions

Command	Description
kboardinfo	Kontron Board Information - Displays a summary of board and configuration information
md5sum	Creates or checks the md5 message digest over a memory area

7.4.1. kboardinfo - Kontron Board Information

The "kboardinfo" command shows a summary of board serialization data gathered from the system EEPROM.

```
=> kboardinfo
Manufacturer:        Kontron America Inc.
Product name:        SMARC-sAMX8
Material number:     51009-0208-10-2
Serial number:       9DP9340003
MAC0 (ethaddr):      00:20:0C:7C:C4:EC
MAC1 (eth1addr):     00:20:0C:7C:C4:ED
Manufacturer Date:   11/15/2019
Revision:            A00
Boot Counter:        10
CPU:                 Freescale i.MX8QM rev0.1 at 1200 MHz
==>
```

7.4.2. md5sum – MD5 Message Digest

The "md5sum" command is already part of standard U-Boot implementation. However Kontron provides the "-a" extension (ASCII) that allows to check MD5 checksum of a given memory area (e.g. a binary image copied into memory) against the checksum that has been copied into an ASCII file on an external linux host. To achieve this, the ASCII string in the file representing the checksum is converted into hexadecimal values and compared against the calculated one.

Syntax:

```
=> help md5sum
md5sum - compute MD5 message digest

Usage:
md5sum address count [[*]sum]
    - compute MD5 message digest [save to sum]
md5sum -v address count [*]sum
    - verify md5sum of memory area
m address count [*]sum
    - verify md5sum given in ASCII format
```

Example:

Calculate MD5 checksum of a given binary using the md5sum command on linux host and redirect output messages into a file:

```
# md5sum image.bin >image.md5
```

Copy both image file and checksum file to USB thumb device with FAT partition. Connect the USB device to the module carrier.

The following sequence shows how to load both image file and image checksum file into SMARC_sAMX8X memory and compare them. In case of success, the "md5sum -a" command will have no output messages as this extensions is meant to be used in automatic update scripts to check the binary images against their MD5 checksum.

```
usb start
fatload usb 0:1 88000000 <image.bin>
fatload usb 0:1 8a000000 <image.md5>
if md5sum -a 88000000 $ubootsz *8a000000; then
    echo  'CRC check passed'
else
    echo  'CRC check failed'
fi
```

7.5. Bootloader Environment

The bootloader environment is used to control bootloader and OS startup behavior. Environment variables can be used to control boot timing (e.g. bootdelay), interface properties (e.g. baudrate, ethact) or they define memory locations where OS images are stored before boot (e.g. loadaddr, fdt_addr). In addition, bootloader shell commands can be combined to environment scripts.

The bootloader environment is permanently stored in the QSPI flash device at offset 0x400000. During bootloader operation, the environment is held in RAM memory and can be modified and written back to persistent storage.

Bootloader commands to modify the environment are summed up under the "env" command group:

- env default [-f] –a [forcibly] reset default environment
- env default [-f] var [...] [forcibly] reset variable(s) to their default values
- env delete [-f] var [...] [forcibly] delete variable(s)
- env edit name edit environment variable
- env exists name tests for existence of variable
- env print [-a | name ...] print environment
- env run var [...] run commands in an environment variable
- env save save environment
- env set [-f] name [arg ...]

However, the legacy commands for environment handling are still available:

- "setenv",
- "editenv",
- "printenv"
- "saveenv".

U-Boot standard environment variables are set up for the SMARC_sAMX8 module as shown below.

Table 13: Standard Environment Variables

Variable	Value	Description
baudrate	115200	Serial line baudrate
bootcmd	run module_mmc_boot || run carrier_sd_boot || run netboot	Try booting (in this order) from module eMMC, carrier uSD card, network
bootdelay	3	Wait 3 seconds before executing bootcmd
ethprime	FEC0	Use Ethernet port FEC0 as default
loadaddr	0x80280000	Default memory location for OS boot

A typical user modification would be to set the variable "bootcmd" to change OS boot commands.

7.6. Kontron Bootloader Environment Extensions

To support SMARC_sAMX8 board properly, Kontron adds some environment variables to the standard set of variables provided by mainline U-Boot. These variables are shown below.

Table 14: Bootloader Environment Extensions

Variable	Value	Description
boot_sel	Depends on BOOT_SEL lines	BOOT_SEL lines from carrier are evaluated during startup and boot_sel is set appropriately. See chapter 0 Bootloader Boot Source for more detail
eth1addr	From EEPROM	Ethernet port FEC1 MAC address
ethaddr	From EEPROM	Ethernet port FEC0 MAC address
fdt_addr	0x83000000	Memory location for device tree blob
serial#	From EEPROM	Module serial number

Variables from the table above marked in bold are set automatically each time U-Boot starts. They will override different settings possibly stored in persistent environment.

7.7. Bootloader Mass Storage Support

U-Boot provides support to read and write from mass storage devices like

- QSPI flash
- eMMC device
- SD card
- USB thumb device

7.7.1. QSPI flash

QSPI flash is accessed using the "sf" command

Example: Load one sector (64K) from SPI flash

```
=> sf probe 0
=> sf read ${loadaddr} 0 10000
```

7.7.2. SD Card and eMMC Devices

eMMC and SD card are accessed using the "mmc" command

Example: Load 256 blocks from eMMC

```
=> mmc dev 1
=> mmc read ${loadaddr} 0 100
```

7.7.3. USB Storage Device

USB storage devices are accessed using "usb" command

Example: Load bootloader update file from USB thumb device

```
=> usb start
=> usb dev 0
=> fatload usb 0:1 u-boot-sMX8X_spl.bin
```

7.8. Bootloader File System Support

U-Boot for the SMARC_sAMX8 provides support for FAT file systems. There are file system specific commands available to list file system contents (fatls) and load a given file into board memory (fatload).

Example: Show/boot folder contents from SD card on carrier file system

```
=> fatls mmc 1:1 /boot
```

7.9. Bootloader Network Support

U-Boot provides support for both onboard Ethernet interfaces. The current interface can be selected by setting "ethact" environment variable to either “FEC0” or “FEC1”.

Board specific MAC addresses are read from EEPROM during startup and environment variables are set automatically. In case EEPROM contents is missing or corrupted, a "random" MAC address will be set to "ethaddr"

In case that the current network interface is attached to a network providing a DHCP server, an IP address can be gathered using "bootp" ord "dhcp" commands.

After that, a file from a tftp server can be copied to memory using the "tftpboot" command.

Example:

```
=> bootp
=> tftpboot ${loadaddr} <filename>
```

7.10. Bootloader Boot Source Support

The SMARC v2.0 Specification defines three boot select signals BOOT_SEL[0:2]# that allows the user to select from eight possible boot devices. On the SMARC_sAMX8X, U-Boot detects the BOOT_SEL signals from the carrier and sets the environment variable "boot_sel" as shown below.

Table 15: Environment Variables for "boot_sel"

BOOT_SEL2#	BOOT_SEL1#	BOOT_SEL0#	boot_sel	Boot Source
GND	GND	GND	carrier_sata (1)	Carrier SATA
GND	GND	Float	carrier_sd	Carrier SD Card
GND	Float	GND	carrier_mmc (1)	Carrier eSPI (CS0#)
GND	Float	Float	carrier_spi (1)	Carrier SPI (CS0#)
Float	GND	GND	module_device (1)	Module Device (NAND, NOR) – vendor specific
Float	GND	Float	remote	Remote boot (GBE, serial) – vendor specific
Float	Float	GND	module_mmc	Module eMMC Flash
Float	Float	Float	module_spi	Module SPI

(1) The SMARC sAMX8 does not support these modes

Bootloader environment scripts can use the boot_sel environment variable to select the source where the dedicated OS image can be loaded. As an example, the bootloader environment of the SMARC_sAMX8 implements some small scripts that will load a linux system from the boot source as defined by the BOOT_SEL pins.

```
bootsel_boot=echo BOOT_SEL ${boot_sel} selected && run ${boot_sel}_boot
module_spi_boot=run module_mmc_boot
module_mmc_boot=setenv mmcdev 0; run mmcboot
carrier_sd_boot=setenv mmcdev 1; run mmcboot
remote_boot=run netboot
```

Running the "run bootsel_boot" script will load a linux system from MMC if BOOT_SEL pins define the module MMC Flash or the Module SPI as boot source. Depending on application, the script variables in environment can be adapted.

The bootloader is always booted from the boot source defined by fuse settings, which is normally the QSPI flash device. The BOOT_SEL pins only define boot source for the OS.

7.11. Bootloader Boot Counter

U-Boot on the SMARC_sAMX8 module will read the current boot counter value and increment it on every boot cycle. Current boot counter is shown as part of the information shown by the "kboardinfo" command (see description of kboardinfo).

7.12. Bootloader Update

Bootloader update on SMARC_sAMX8 can be update from Uboot prompt

```
/* Erase the QSPI */
=> sf probe 0
=> sf erase 0 200000           (The SPI image should be loaded with 0 offset into FlexSPI)

/* Load the bootloader image to memory address 0x88000000 */

from a usb key:
=> usb start
=> usb dev 0
=> fatload usb 0:1 88000000 imx-boot-kontron-samx8-qm-fspi.bin

from Ethernet:
=> bootp 88000000 imx-boot-kontron-samx8-qm-fspi

/* Write back to QSPI */
=> sf write 88000000 0 <filesize>
```

8/ Linux OS

8.1. Software Images

SMARC sAMX8QM Linux BSP is derived from the IMX8 Yocto Linux Framework. The sAMX8QM BSP images consists of Uboot based bootloader, kernel image, device tree binary and a root filesystem image. To boot Linux on sAMX8 board the below images are needed:

- Bootloader which includes U-Boot, Arm Trusted Firmware, DCD file, System controller firmware and the SECO firmware since B0
- Linux kernel image (Image)
- A device tree file (.dtb) for the board being used
- A root file system (rootfs) for the Linux image

The graphical backend for the sAMX8QM Linux OS is the XWayland

8.2. Boot from SD

This section describes the steps to copy the BSP images to the boot media and deploy it on the sAMX8QM board. The BSP images are built from the Yocto build framework that generates combined as well as individual BSP images

8.2.1. Preparing SD card

This section describes the steps to copy the image when all images U-boot, kernel and the Root filesystem are loaded from the SD card. The Yocto project build generates an integrated SD card image (.sdcard) that can be flashed directly onto the SD card. The .sdcard image contains all images (U-boot, kernel and RootFS) properly configured for an SD card. Below are the steps to prepare the SD card using a Linux Host PC

- Insert a 4GB SD card to the Linux Host PC using a SD card reader
- Check the mount point by giving the command

```
=> mount
          (or)
=> cat /proc/partitions
     major minor #blocks name
  8     0     78125000 sda
  8     1     75095811 sda1
  8     2     1        sda2
  8     5     3028221 sda5
  8     16    3921920 sdc
  8     18    3905535 sdc1
```

- Execute the following command to copy the SD card image to the SD/MMC card. Change `sdx` below to match the one used by the SD card sdc or sdb

```
=> sudo dd if=<image name>.sdcard of=/dev/sdx bs=1M && sync
```

Example image name can be either kci-image-qt5-validation-smx-kci_smx8qm.sdcard or kci-image-validation-smx-kci_smx8qm.sdcard

Note: The Yocto build environment generates the compressed. sdcard image and it needs to be uncompressed by using the below command before copying to the SD card

```
=> bunzip2 -dk -f <image_name>. sdcard.bz2
```

8.2.2. Running Linux from SD

- Insert the SD card after all images are copied as per the previous steps in the designated SD card slot on board
- Make sure the Boot switch is configured for SD boot

Note: If board tries to boot from other media then perform the following steps

Stop at Uboot prompt and enter the following commands to set the U-boot environment variables to boot kernel and rootfs from SD card

- Command to set boot argument variables for serial console port for boot logs, baud rate of console port and root file system path is as given below:

```
=> setenv bootargs console=ttyLP0,115200 earlycon=lpuart32,0x5a060000,115200 root=/dev/mmcblk1p2 rootwait
```

- Commands to set the load address of FDT (Flattened Device Tree) file and Kernel Image from SD partition mmcblk1p1 is as below

```
=> setenv loadfdt 'fatload mmc ${mmcdev}:${mmcpart} ${fdt_addr} ${fdt_file}'
=> setenv loadkernel 'fatload mmc ${mmcdev}:${mmcpart} ${loadaddr} Image'
```

- Command to set and save the Boot Command environment variable for Booting kernel and DTB from `loadaddr` and `fdt_addr` is as given below

```
=> setenv bootcmd 'mmc dev ${mmcdev}; run loadkernel; run mmcargs; run loadfdt; booti ${loadaddr} -
  ${fdt_addr};'
=> saveenv
=> boot
```

8.3. Boot from eMMC

8.3.1. Preparing eMMC

Using UUU Tool (Download from https://github.com/NXPmicro/mfgtools/releases)

Follow these instructions to use the Universal Update Utility (UUU)

1. Connect a USB cable from a computer to USB port3 (bottom) on the carrier

2. Connect a serial cable from the UART0 port tot he computer for console output

3. Open terminal emulator program (115200,8,0,1,None)

4. Assert signal FORCE_RECOV# (pin# S155) on your SMARC carrier. The board will then be in Serial download mode for the UUU Tool

Execute the command line tools from the computer (Windows or Linux)

- Burn uboot into emmc

```
uuu -b emmc bootloader
```

- Burn rootfs image into emmc

```
uuu -b emmc_all bootloader rootfs.sdcard
```

- Decompress rootfs image and burn into emmc

```
uuu -b emmc_all bootloader rootfs.sdcard.bz2/*
```

8.3.2. Running Linux from eMMC

- Make sure the Boot switch is configured for eMMC boot
- Stop at Uboot prompt and give the following commands to set the boot environment variables

- Command to set boot argument variables for serial console port for boot logs, baud rate of console port and root file system path MMC0 Partition 2 is as given below:

```
=> setenv bootargs console=ttyLP0,115200 earlycon=lpuart32,0x5a060000,115200 root=/dev/mmcblk0p2 rootwait
```

- Commands to set the load address of FDT (Flattened Device Tree) file and Kernel Image from MMC partition mmcblk0p1 is as below

```
=> setenv mmcdev 0
=> setenv mmcpart 1
=> setenv loadfdt 'fatload mmc ${mmcdev}:${mmcpart} ${fdt_addr} ${fdt_file}'
=> setenv loadkernel 'fatload mmc ${mmcdev}:${mmcpart} ${loadaddr} Image'
```

- Command to set and save the Boot Command environment variable for Booting kernel and DTB from `loadaddr` and `fdt_addr` is as given below

```
=> setenv bootcmd 'mmc dev ${mmcdev}; run loadkernel; run mmcargs; run loadfdt; booti ${loadaddr} -
${fdt_addr};'
=> saveenv
=> boot
```

8.4. Boot from Network – TFTP and NFS

Network Boot is supported on the sAMX8 board from kernel images onwards. The Kernel image, dtb and other firmware files can be transferred via TFTP to the board in Uboot environment and once the kernel starts it can mount the root filesystem over NFS

Note: In this mode Boot Firmware and Uboot shall be loaded from boot media like SD card, eMMC or SPI and thus the bootloader image needs to be flashed in one of these sources. Once Uboot starts stop at Uboot prompt to setup the TFTP environment and proceed with TFTP boot

8.4.1. Preparing TFTP and NFS setup

This section describes the steps to configure the Trivial File Transfer Protocol (TFTP) server and Network File System (NFS) server on the host system. U-Boot shall download the Linux kernel image, HDMI firmware binaries and dtb file using TFTP and then the kernel will mount the root file system from the host system NFS Root path

- Preparing the TFTP Service

  Install the required software in Host PC to setup TFTP.

```
=> sudo apt-get install tftpd-hpa
```

  Start the tftpd-hpa service automatically by adding a command to /etc/rc.local file

```
=> vi /etc/rc.local
```

  Before the exit 0 line edit below command then Save and Exit

```
=> service tftpd-hpa start
```

To control and check the status of the TFTP service from the command line use the below commands:

```
=> service tftpd-hpa restart
=> service tftpd-hpa status
```

- Setup the TFTP Directories

  Create the directory which will contain the kernel image and the device tree file.

  Example:

```
=> mkdir -p /samx8-boot/kontron_smx8qm/tftp
```

  Copy the kernel image, HDMI firmware binaries and the device tree blob file in this directory

```
=> cp Image /samx8-boot/kontron_smx8qm/tftp
=> cp kontron-samx8-qm.dtb /samx8-boot/kontron_smx8qm/tftp
=> cp hdmirxfw.bin /samx8-boot/kontron_smx8qm/tftp
=> cp hdmitxfw.bin /samx8-boot/kontron_smx8qm/tftp
```

  Edit default tftp directory

```
=> vi /etc/default/tftpd-hpa
```

Change the directory defined as TFTP_DIRECTORY with the host system directory which contains kernel and device tree blob file

TFTP_DIRECTORY="/samx8-boot/kontron_smx8qm/tftp"

Restart the TFTP service if required

```
=> service tftpd-hpa restart
```

- Preparing the NFS Service

Install the required software package on Host PC

```
=> sudo apt-get install nfs-kernel-server
```

The NFS service starts automatically. To control and check the status of NFS services use the below commands

```
=> service nfs-kernel-server restart
=> service nfs-kernel-server status
```

- Setup the NFS Directories

Create the directory which will contain the root file system

```
=> mkdir -p /samx8-boot/kontron_smx8qm/nfs
```

Copy the rootfs in this directory

```
=> sudo tar -jxvf kci-image-qt5-validation-smx-kci_smx8qm.tar.bz2 -C /samx8-boot/kontron_smx8qm/nfs
```

The NFS server requires /etc/exports to be configured correctly to access NFS filesystem directory to specific hosts

```
=> vi /etc/exports
```

Then edit below line into the opened file

<" NFS DIRECTORY"> < BOARD IP>(rw,sync,no_root_squash,no_subtree_check)

Example: /samx8-boot/kontron_smx8qm/nfs 10.16.82.170(rw,sync,no_root_squash,no_subtree_check)

Now restart the NFS service

```
=> service nfs-kernel-server restart
```

8.4.2. Running Linux from NFS

Power on the board and hit a key to stop the U-Boot from continuing. To boot from NFS, set the following environment variables at the U-Boot prompt the below commands

```
=> setenv ipaddr 10.16.82.170
=> setenv serverip 10.16.82.57
=> setenv nfsroot /samx8-boot/kontron_smx8qm/nfs
=> setenv image < zImage name >
=> setenv fdt_file<dtb file name on host>
=> setenv netargs 'setenv bootargs console=ttyLP0,115200 root=/dev/nfs ip=dhcp
   nfsroot=${serverip}:${nfsroot},v3,tcp'
=> setenv bootcmd run netboot
=> saveenv
```

Example-

Board IP address is 10.16.82.170
Server IP address is 10.16.82.57
zImage name is Image
dtb file name is kontron-samx8-qm.dtb

Restart the board it should now boot from the network

8.5. Selecting a Device Tree Binary

sAMX8QM Linux BSP provides option for the user to select a specific DTB file based on the PIN configuration and GPIO mux selection to enable a particular interface. The below list provides the different DTB files available to the user to load before booting of the kernel. To select the DTB file stop at Uboot prompt and enter the below command

```
# setenv fdt_file <DTB filename>
# saveenv
```

Example

```
# setenv fdt_file kontron-samx8-qm-enet2.dtb
```

DTB File Name	BSP Feature Supported
kontron-samx8-qm.dtb	Default DTB file with most interfaces enabled
kontron-samx8-qm-enet2.dtb	DTB file with ENET2 interface enabled
kontron-samx8-qm-usbotg3-peripheral.dtb	DTB file with USB3.0 defined in peripheral or device mode
kontron-samx8-qm-edp.dtb	DTB with eDP support

8.6. Kernel Boot Log

Linux OS boot logs can be viewed on a Serial console in Host PC connected to the sAMX8QM board on the SER_0 port of the SMARC carrier board. Common serial communication programs such as HyperTerminal, Tera Term, or PuTTY can be used. Configure the serial line using **115200 baud, 8 data bits, 1 stop bit, no parity**. Once the kernel starts the below messages shall appear on the console

```
Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.14.78-imx_4.14.78_1.0.0_ga+g94da7bd (oe-user@oe-host) (gcc version 7.3.0 (GCC)) #1
           SMP PREEMPT Th9
[    0.000000] Boot CPU: AArch64 Processor [410fd034]
[    0.000000] Machine model: Kontron SMX8QM i.MX8QM
[    0.000000] earlycon: lpuart32 at MMIO 0x000000005a060000 (options '115200')
```

8.7. Running ARM Cortex-M4 Images

SMARC sAMX8QM Uboot supports booting of Cortex-M4 cores using the bootaux commands. U-Boot supports loading the Arm Cortex-M4 image from the FAT partitions of the SD card.

- Copying M4 Binaries to SD card

Refer to section 9.2.2 for partitioning of SD card

Copy m4_0_image.bin and m4_1_image to <vfat> partition of the SD Card using the below commands by inserting the SD card to a Host PC using a USB card reader

```
=> sudo mkdir /media/<vfat_mountpoint>
=> sudo mount /dev/sdx1 /media/<vfat_mountpoint>
```

where sdx1 is the mount point and partition number (can also be sdc1 or sdb1 depending on the mount point detected in Host PC)

```
=> cp m4_0_image.bin /media/<vfat_mountpoint>
=> cp m4_1_image.bin /media/<vfat_mountpoint>
=> sync
=> sudo umount /dev/sdx1
```

- Loading M4 images from Uboot

On the target board insert the SD card in the designated slot and boot till Uboot and hit the prompt

Enter the below command to run FreeRTOS application from M4 core 0

```
=> fatload mmc 1:1 0x80280000 m4_0_image.bin
=> bootaux 0x80280000 0
```

Enter the below command to run FreeRTOS application from M4 core 1

```
=>fatload mmc 1:1 0x80280000 m4_1_image.bin
=>bootaux 0x80280000 1
```

Alternatively M4 binaries can also be copied into target uboot environment from TFTP setup instead of SD card
Follow the below procedure for loading the M4 binaries from TFTP server and running them after stopping at uboot

On the Host PC copy the M4 binaries to the TFTP directory as explained in section 9.4.1 section

```
=> cp m4_0_image.bin /samx8-boot/kontron_smx8qm/tftp
=> cp m4_1_image.bin /samx8-boot/kontron_smx8qm/tftp
```

On the sAMX8 target board enter the below commands to run FreeRTOS application from M4 core 0

```
=> tftp 0x80280000 m4_0_image.bin
=> bootaux 0x80280000 0
```

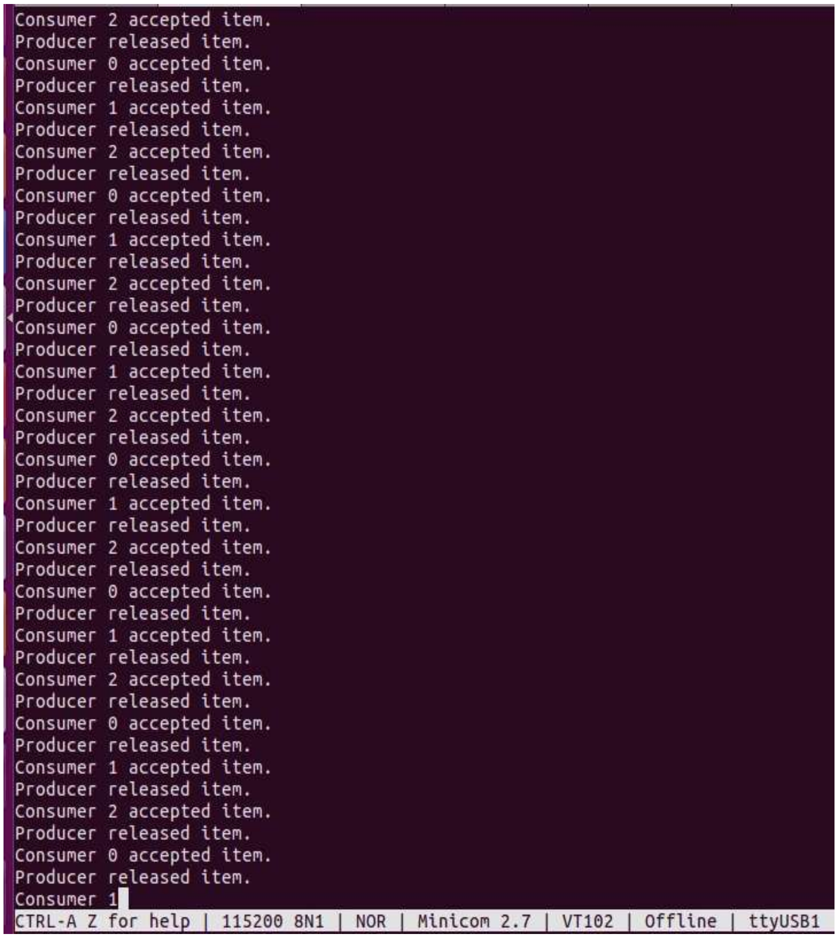
Enter the below commands to run FreeRTOS application from M4 core 1

```
=> tftp 0x80280000 m4_1_image.bin
=> bootaux 0x80280000 1
```

- Viewing M4 FreeRTOS Application Logs

Connect the DB9 cable from Host PC to M4 Serial port. M4_0 Serial Port is - J29 Top connector and M4_1 Serial Port is - J30 Top connector on the SMARC carrier board. Common serial communication programs such as HyperTerminal, Tera Term, or PuTTY can be used. Configure the serial line using **115200 baud, 8 data bits, 1 stop bit, no parity**.

Below is a sample log from FreeRTOS Semaphore example from NXP MCUXpresso SDK running on M4 0 core



```
Consumer 2 accepted item.
Producer released item.
Consumer 0 accepted item.
Producer released item.
Consumer 1 accepted item.
Producer released item.
Consumer 2 accepted item.
Producer released item.
Consumer 0 accepted item.
Producer released item.
Consumer 1 accepted item.
Producer released item.
Consumer 2 accepted item.
Producer released item.
Consumer 0 accepted item.
Producer released item.
Consumer 1 accepted item.
Producer released item.
Consumer 2 accepted item.
Producer released item.
Consumer 0 accepted item.
Producer released item.
Consumer 1 accepted item.
Producer released item.
Consumer 2 accepted item.
Producer released item.
Consumer 0 accepted item.
Producer released item.
Consumer 1 accepted item.
Producer released item.
Consumer 2 accepted item.
Producer released item.
Consumer 0 accepted item.
Producer released item.
Consumer 1 accepted item.
Producer released item.
Consumer 2 accepted item.
Producer released item.
Consumer 0 accepted item.
Producer released item.
Consumer 1
CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7 | VT102 | Offline | ttyUSB1
```

8.8. File System Support

SMARC sAMX8QM BSP supports the EXT2/EXT3 or EXT4 Root filesystem type in eMMC or SD. An image build for sAMX8QM from the Yocto build system generates a U-Boot, kernel, and an image type based on the "IMAGE_FSTYPES" defined in the machine configuration file. The default machine configuration file for sAMX8QM generates an SD card image (.sdcard), an ext4 and tar.bz2. The ext4 is the root file system only. The .sdcard image contains U-Boot, the kernel and the rootfs completely set up for use on an SD card

8.9. Watchdog Support

Linux kernel can reset the system via a hardware watchdog or a software watchdog when serious problems or failures are detected. This can be achieved through a watchdog service or daemon which opens the /dev/watchdog and refreshes it periodically to keep the kernel from resetting. When the daemon stops refreshing the watchdog the watchdog times out and issues a system reset

SMARC sAMX8 BSP enables the watchdog daemon in background such that it keeps refreshing the hardware watchdog /dev/watchdog periodically. The status of this service cab be checked using the below command

```
=> systemctl status watchdog
  watchdog.service - watchdog daemon
  Loaded: loaded (/lib/systemd/system/watchdog.service; enabled; vendor preset>
  Active: active (running) since Fri 2019-11-08 08:52:49 UTC; 1min 39s ago
```

The Linux watchdog daemon not only refreshes the watchdog hardware periodically but also performs other system checks as per the configurations in the watchdog.conf file and triggers a system reset when a failure in any of these checks is encountered

- Watchdog Daemon Test

The Watchdog daemon reset behavior can be tested on the SMARC board using the below sequence of commands

Stop the Watchdog service using the below commands

```
=> systemctl stop watchdog
=> systemctl status watchdog
   Active: inactive (dead) since Fri 2019-11-08 08:55:45 UTC; 7s ago
```

Edit the watchdog conf file in location /etc/watchdog.conf
Enable ping to an IP which doesn't exist in the board's network (uncomment the below line in the file)

```
ping 10.1.x.x
```

Restart the watchdog service using the below commands

```
=> systemctl start watchdog
=> systemctl status watchdog
    watchdog.service - watchdog daemon
    Loaded: loaded (/lib/systemd/system/watchdog.service; enabled; vendor preset>
    Active: active (running) since Fri 2019-11-08 08:52:49 UTC; 1min 39s ago
```

In approximately 60secs the watchdog daemon will trigger a system reset due to failure of the ping command

Alternatively SMARC sAMX8 BSP also includes two example user space applications to demonstrate the device reset behaviour and the dual stage watchdog behaviour upon a watchdog pretimeout and timeout event. To run these sample applications on the sAMX8 target requires the user to stop the watchdog deamon service so that the /dev/watchdog is released by the service and acquired by the user applications given in the below directory

- Watchdog Reset Test

The sample user space application is located in the file system path given below. The sample application uses the IOCTL calls to set the Watchdog timeout using the `WDIOC_SETTIMEOUT` parameter

Note: It is important to make sure the watchdog service is inactive before trying to perform watchdog reset test. When you perform watchdog reset test and the watchdog service is active already then you get and error message " *device busy error* " There are two possible scenarios, each one is explained below.

- **Scenario 1 – Running the test applications before the watchdog service starts**

1. This scenario is applicable in SMARC SAMX8 BSP case before 1 minute of FS mounting and OS services start when the watchdog daemon has still not started
2. Enter the below command and ensure that the watchdog service is inactive
   - # *systemctl status watchdog*
   - *Loaded: loaded (/lib/systemd/system/watchdog.service; enabled; vendor preset*>
   - *Active:* ***inactive*** *(dead)*
3. Now run the watchdog tests as below and confirm that you are not getting device busy error and device resets as expected
   - # *cd /unit_tests/Watchdog*
   - # *./wdt_driver_test.out 5 10 0 &*
   - Note: Here timeout=5 secs, Refresh Rate=10secs, 0-signifies IOCTL call
   - *(or)*
   - # *cd /unit_tests/kontron/watchdog/*
   - # *./wdt_pretimeout_test.out 5 2 10 0*
   - Note: Here timeout=5 secs, Pre-timeout =2secs, Refresh Rate=10secs, 0-signifies IOCTL call

- **Scenario 2 – Running the test applications after the watchdog service starts**

1. This scenario is applicable if you want to run the watchdog application tests after the watchdog service has started. This is approx. after a minute of system booting

2. Check the daemon status as active by giving the below command
   - # *systemctl status watchdog*
   - *watchdog.service - watchdog daemon*
   - *Loaded: loaded (/lib/systemd/system/watchdog.service; enabled; vendor preset>*
   - *Active:* ***active*** *(running) since Fri 2019-11-08 08:52:49 UTC; 1min 39s ago*
3. Active state implies the daemon has acquired the hardware watchdog and hence the test applications will not be able to acquire this node
4. Stop the watchdog service by giving the below commands
   - # *systemctl stop watchdog*
   - # *systemctl status watchdog*
   - *Active:* ***inactive*** *(dead) since Fri 2019-11-08 08:55:45 UTC; 7s ago*
5. Once the watchdog service is stopped the hardware watchdog is released and hence after that you should be able to perform the user application tests as per point 3 given in scenario 1 section

Here the actual output of a watchdog command:

```
root@kci_smx8qm:~# cd /unit_tests/Watchdog
root@kci_smx8qm:/unit_tests/Watchdog# ./wdt_driver_test.out 5 10 0 &
[1] 3994

---- Running < ./wdt_driver_test.out > test ----

Starting wdt_driver (timeout: 5, sleep: 10, test: ioctl)
Trying to set timeout value=5 seconds
The actual timeout was set to 5 seconds
Now reading back -- The timeout is 5 seconds
root@kci_smx8qm:/unit_tests/Watchdog#
```

- Watchdog Dual Stage Test

Note: It is important to make sure the watchdog service is inactive before trying to perform watchdog reset test. Please refer to the precedent section *Watchdog Reset test* and follow either scenario 1 or 2 depending on your situation.

IMX8 Watchdog supports the Dual stage watchdog along with the Linux Kernel configuration to enable pre-timeout governor

```
< >   Berkshire Products USB-PC Watchdog
      *** Watchdog Pretimeout Governors ***
[*]   Enable watchdog pretimeout governors
        Default Watchdog Pretimeout Governor (panic)  --->
<*>     Noop watchdog pretimeout governor
-*-     Panic watchdog pretimeout governor

        <Select>    < Exit >    < Help >    < Save >    < Load >
```

SMARC SAMX8 BSP enables this configuration by default

Check watchdog pretimeout governor on the sAMX8 board by giving the below command:

```
=> cat /sys/class/watchdog/watchdog0/pretimeout_governor
   panic
```

The sample user space application is located in the file system path given below. The sample application uses the IOCTL calls to set the Watchdog pre-timeout using the `WDIOC_SETPRETIMEOUT` parameter

To test watchdog pre timeout issue the following command. After 3 seconds (timeout - pretimeout), it will indicate a pretimeout message with kernel panic message and then after 2 seconds, it will reset the board.

```
=> cd /unit_tests/kontron/watchdog/
=> ./wdt_pretimeout_test.out 5 2 6 0
---- Running < ./wdt_pretimeout_test.out > test ----
```

```
Starting wdt_driver (timeout: 5, pre-timeout: 2, sleep: 6, test: ioctl)
Trying to set timeout value = 5 seconds
The actual timeout was set to 5 seconds
Now reading back -- The timeout is 5 seconds
Trying to set pre_timeout value = 2 seconds
[ 262.416085] Kernel panic - not syncing: watchdog pretimeout event
[ 262.416085]
[ 262.423677] CPU: 0 PID: 105 Comm: kworker/0:1 Tainted: G O 4.14.78-imx_4.14.78_1.0.0_ga+g94da7bd #1
[ 262.433856] Hardware name: Kontron SMX8QM i.MX8QM (DT)
[ 262.439018] Workqueue: events scu_mu_work_handler
[ 262.443725] Call trace:
[ 262.446184] [<ffff000008089c48>] dump_backtrace+0x0/0x3c8
[ 262.451593] [<ffff00000808a024>] show_stack+0x14/0x20
[ 262.456649] [<ffff000008d7c240>] dump_stack+0x9c/0xbc
[ 262.461705] [<ffff0000080cecb8>] panic+0x11c/0x28c
[ 262.466502] [<ffff000008a05cf4>] pretimeout_panic+0x14/0x18
[ 262.472076] [<ffff000008a05694>] watchdog_notify_pretimeout+0x3c/0x58
[ 262.478523] [<ffff000008a07044>] imx8_wdt_notify+0x2c/0x38
[ 262.484014] [<ffff0000080ef720>] notifier_call_chain+0x50/0x90
[ 262.489850] [<ffff0000080f0010>] blocking_notifier_call_chain+0x48/0x70
[ 262.496472] [<ffff0000085b5058>] scu_mu_work_handler+0x50/0x78
[ 262.502310] [<ffff0000080e7c94>] process_one_work+0x1d4/0x348
[ 262.508059] [<ffff0000080e7e50>] worker_thread+0x48/0x470
[ 262.513465] [<ffff0000080ee154>] kthread+0x12c/0x130
[ 262.518433] [<ffff000008084ed8>] ret_from_fork+0x10/0x18
[ 262.523753] SMP: stopping secondary CPUs
[ 262.527690] Kernel Offset: disabled
[ 262.531182] CPU features: 0x180200c
[ 262.534674] Memory Limit: none
[ 262.537742] ---[ end Kernel panic - not syncing: watchdog pretimeout event
[ 262.537742]

U-Boot 2018.03-imx_v2018.03_4.14.78_1.0.0_ga+g7ade5b4 (Oct 11 2019 - 13:11:44 +0000)
CPU: Freescale i.MX8QM revB A53 at 1200 MHz at 33C
Model: Freescale i.MX8QM MEK
Board: Kontron sMX8 SMARC 2.0 Module
Boot: SD1
DRAM: 6 GiB
Invalid fsl,pins property in node hoggrp-1
start sata init
SATA link 0 timeout.
MMC: FSL_SDHC: 0, FSL_SDHC: 1
Loading Environment from MMC... OK
[pcie_ctrla_init_rc] LNK DOWN 8600000
In: serial
Out: serial
Err: serial
```

9/ Technical Support

For technical support contact our Support Department:

- E-mail: support@kontron.com
- Phone: +49-821-4086-888

Make sure you have the following information available when you call:

- Product ID Number (PN),
- Serial Number (SN)

The serial number can be found on the Type Label, located on the product's rear side.

Be ready to explain the nature of your problem to the service technician.

9.1. Warranty

Due to their limited service life, parts that by their nature are subject to a particularly high degree of wear (wearing parts) are excluded from the warranty beyond that provided by law. This applies to the CMOS battery, for example.

If there is a protection label on your product, then the warranty is lost if the product is opened.

9.2. Returning Defective Merchandise

All equipment returned to Kontron must have a Return of Material Authorization (RMA) number assigned exclusively by Kontron. Kontron cannot be held responsible for any loss or damage caused to the equipment received without an RMA number. The buyer accepts responsibility for all freight charges for the return of goods to Kontron's designated facility. Kontron will pay the return freight charges back to the buyer's location in the event that the equipment is repaired or replaced within the stipulated warranty period. Follow these steps before returning any product to Kontron.

1. Visit the RMA Information website:
http://www.kontron.com/support-and-services/support/rma-information

Download the RMA Request sheet for Kontron Europe GmbH and fill out the form. Take care to include a short detailed description of the observed problem or failure and to include the product identification Information (Name of product, Product number and Serial number). If a delivery includes more than one product, fill out the above information in the RMA Request form for each product.

2. Send the completed RMA-Request form to the fax or email address given below at Kontron Europe GmbH. Kontron will provide an RMA-Number.

   Kontron Europe GmbH
   RMA Support
   Phone: +49 (0) 821 4086-0
   Fax: +49 (0) 821 4086 111
   Email: service@kontron.com

3. The goods for repair must be packed properly for shipping, considering shock and ESD protection.

Goods returned to Kontron Europe GmbH in non-proper packaging will be considered as customer caused faults and cannot be accepted as warranty repairs.

4. Include the RMA-Number with the shipping paperwork and send the product to the delivery address provided in the RMA form or received from Kontron RMA Support.

Appendix A: List of Acronyms

Table 16: List of Acronyms (Example)

API	Application Programming Interface
BMC	Base Management Controller
CLI	Command-Line Interface
COM	Computer-on-Module
ECC	Error Checking and Correction
FRU	Field Replaceable Unit
GPU	Graphics Processing Unit
HD/HDD	Hard Disk /Drive
HPM	PICMG Hardware Platform Management specification family
IOL	IPMI-Over-LAN
IOT	Internet of Things
IPMI	Intelligent Platform Management Interface
KCS	Keyboard Controller Style
KVM	Keyboard Video Mouse
MEI	Management Engine Interface
NCSI	Network Communications Services Interface
PCIe	PCI-Express
PECI	Platform Environment Control Interface
PICMG®	PCI Industrial Computer Manufacturers Group
RTC	Real Time Clock
SEL	System Event Log
ShMC	Shelf Management Controller
SMBus	System Management Bus
SMWI	System Monitor Web Interface
SOL	Serial Over LAN
SSH	Secure Shell
TPM	Trusted Platform Module
UEFI	Unified Extensible Firmware Interface
VLP	Very Low Profile

0.32	Front Panel chapter with colors; Safety note added	2019-10-21

About Kontron

Kontron is a global leader in Embedded Computing Technology (ECT). As a part of technology group S&T, Kontron offers a combined portfolio of secure hardware, middleware and services for Internet of Things (IoT) and Industry 4.0 applications. With its standard products and tailor-made solutions based on highly reliable state-of-the-art embedded technologies, Kontron provides secure and innovative applications for a variety of industries. As a result, customers benefit from accelerated time-to-market, reduced total cost of ownership, product longevity and the best fully integrated applications overall. For more information, please visit: **www.kontron.com**

Global Headquarters

Kontron S&T AG

Lise-Meitner-Str. 3-5
86156 Augsburg
Germany
Tel.: + 49 821 4086-0
Fax: + 49 821 4086-111
info@kontron.com