Toradex
Swiss. Embedded. Computing.

Lino iMX91

Datasheet

Preliminary – Subject to Change

Revision History

Document Revisions

Date	Doc. Revision	Product Version	Changes
19-Feb-2026	Rev. 0.1	V1.0	Initial documentation

Contents

List of Tables

List of Figures

Abbreviations

Abbreviations

Abbreviation	Explanation
ADC	Analog to Digital Converter
BTB	Board To Board
CAN	Controller Area Network, a bus that is mainly used in the automotive and industrial environment
CAN FD	Controller Area Network Flexible Data-Rate, an extension to the original CAN bus protocol which allows higher data rates and larger message sizes.
CEC	Consumer Electronic Control, HDMI feature that allows controlling CEC compatible devices
CPU	Central Processor Unit
CSI	Camera Serial Interface
DAC	Digital to Analog Converter
DDC	Display Data Channel, interface for reading out the capability of a monitor. In this document DDC2B (based on I2C) is always meant.
DFP	Downstream Facing Port, USB Type-C port that acts as a host
DRP	Dual-Role Port, USB Type-C port that can operate as power sink and source
DSI	Display Serial Interface
DVI	Digital Visual Interface, digital signals are electrically compatible with HDMI
EDID	Extended Display Identification Data, timing setting information provided by the display in a PROM
EMI	Electromagnetic Interference, high-frequency disturbances
ESD	Electrostatic Discharge, high voltage spike or spark that can damage electrostatic-sensitive devices
FPD-Link	Flat Panel Display Link, high-speed serial interface for liquid crystal displays. In this document is also called the LVDS interface.
GBE	Gigabit Ethernet, Ethernet interface with a maximum data rate of 1000Mbit/s
GND	Ground
GND_CHASSIS	Chassis Ground
GPIO	General Purpose Input/Output, pin that can be configured as an input or output
GSM	Global System for Mobile Communications
HDA	High-Definition Audio (HD Audio), the digital audio interface between CPU and audio codec
I2C	Inter-Integrated Circuit, the two-wire interface for connecting low-speed peripherals
I2S	Integrated Interchip Sound, serial bus for connecting PCM audio data between two devices
I/O	Input-Output
JTAG	Joint Test Action Group, widely used debug interface
LCD	Liquid Crystal Display
LSB	Least Significant Bit
LVDS	Low-Voltage Differential Signaling, electrical interface standard that can transport high-speed signals over twisted-pair cables. Many interfaces like PCIe or SATA use this interface. Since the first successful application was the Flat Panel Display Link, LVDS became a synonymous for this interface. In this document, the term LVDS is used for the FPD-Link interface.
MAC	Medium Access Control is part of the second layer (data link layer) in the Ethernet stack
MIPI	Mobile Industry Processor Interface Alliance
MDI	Medium Dependent Interface, the physical interface between Ethernet PHY and cable connector
MDIO	Management Data Input/Output, an interface that is used for controlling the Ethernet PHY. The bus consists of the MDC clock and the MDIO bidirectional data signal.

Continued on next page

Abbreviations (Continued)

Abbreviation	Explanation
mini PCIe	PCI Express Mini Card, the card form factor for internal peripherals. The interface features PCIe and USB 2.0 connectivity
MMC	MultiMediaCard, flash memory card
MSB	Most Significant Bit
NC	Not Connected
OD	Open-Drain
OTG	USB On-The-Go, a USB host interface that can also act as USB client when connected to another host interface
PCB	Printed Circuit Board
PCI	Peripheral Component Interconnect, parallel computer expansion bus for connecting peripherals
PCIe	PCI Express, a high-speed serial computer expansion bus, replaces the PCI bus
PCM	Pulse-Code Modulation, digitally representation of analog signals, standard interface for digital audio
PD	Pull-Down Resistor
PHY	The physical layer of the OSI model
PU	Pull-up Resistor
PWM	Pulse-Width Modulation
PWR	Power
QSPI	Quad SPI, SPI interface with four bidirectional data signals
RGMII	Reduced Gigabit Media-Independent Interface, the interface between Ethernet MAC and PHY for up to 1Gb/s
RJ45	Registered Jack, common name for the 8P8C modular connector that is used for Ethernet wiring
RS232	The single-ended serial port interface
RS485	Differential signaling serial port interface, half-duplex, multi-drop configuration possible
R-UIM	Removable User Identity Module, identifications card for CDMA phones and networks, an extension of the GSM SIM card
SD	Secure Digital, flash memory card
SDIO	Secure Digital Input Output, an external bus for peripherals that uses the SD interface
SIM	Subscriber Identification Module, an identification card for GSM phones
SMBus	System Management Bus (SMB), a two-wire bus based on the I^2C specifications, is used in x86 designs for system management.
SoC	System on a Chip, IC which integrates the main component of a computer on a single chip
SoM	System on a Module, PCB which integrates the main component of a computer on a single board
SPI	Serial Peripheral Interface Bus, synchronous four-wire full-duplex bus for peripherals
TIM	Thermal Interface Material, thermally conductive material between CPU and heat spreader or heat sink
TMDS	Transition-Minimized Differential Signaling, serial high-speed transmitting technology that is used by DVI and HDMI
TVS Diode	Transient-Voltage-Suppression Diode, a diode that is used to protect interfaces against voltage spikes
UFP	Upstream Facing Port, USB Type-C port that acts as a client
UART	Universal Asynchronous Receiver/Transmitter, serial interface, in combination with a transceiver an RS232, RS422, RS485, IrDA or similar interface can be achieved
USB	Universal Serial Bus, serial interface for internal and external peripherals

1 Introduction

1.1 Purpose of the Datasheet

This document describes the hardware characteristics, features, and capabilities of the **Lino iMX91** System-on-Module (SoM). It is intended to serve as a technical reference for hardware designers, system integrators, and software developers designing products based on this module.

For up-to-date information regarding supported software, operating systems, and integration guidelines, refer to the Toradex Developer Center:

https://developer.toradex.com/hardware/lino-som-family/modules/lino-imx91/

1.2 Lino SoM Family

The **Lino System-on-Module family** is designed as a **compact, cost-efficient, and high-volume solution** for embedded applications with constrained space and budget requirements. By integrating the application processor, memory, power management, and high-speed signal routing on a production-ready module, the Lino family significantly reduces carrier board complexity and accelerates time-to-market.

The Lino iMX91 targets a broad range of applications including:

- Industrial automation controllers
- Medical devices
- IoT gateways and edge computing systems
- Security and surveillance systems
- Energy management and smart factory applications

With its small **30 mm x 30 mm form factor**, the Lino iMX91 is optimized for designs requiring reliable computing performance in compact and thermally constrained environments.

1.2.1 NXP i.MX 91 SoC

The Lino iMX91 SoM is based on the NXP i.MX 91 family of application processors which features single core architecture optimized for power efficiency, security, and edge intelligence. It integrates:

- **One Arm® Cortex®-A55 core**, operating at **up to 1.4 GHz**, providing 64-bit Arm®v8-A performance for Linux-based applications
- **Arm® TrustZone®** architecture, **EdgeLock® secure enclave**, and **Battery Backed Security Module (BBSM)** support for robust protection
- machine-learning acceleration via **Arm® Cortex®-A55 NEON (SIMD) engine**

1.3 Main Features Overview

The Lino iMX91 is based on the **NXP i.MX 9121** or **i.MX 9131** applications processors, integrating **one Arm® Cortex®-A55** application core running at **up to 1.4 GHz**. Table 1 shows the overview of the i.MX 91 processor.

Table 1: CPU core overview

Parameter	i.MX 9131	i.MX 9121
Application cores	1× Cortex-A55	1× Cortex-A55
CPU Segment[1]	Full	Reduced
Max clock	1.4 GHz	800 MHz

[1] Part-number segments identifying configuration variants of the same SoC (CPU, peripherals, package, temperature, security).

1.3.1 Multimedia and Acceleration

The i.MX 91 does **not** integrate advanced multimedia subsystems such as GPU acceleration, camera interfaces, or dedicated display pipelines. It provides a single **LCDIF display controller**, which is exposed through multiplexed I/O pins and intended for simple parallel display interfaces rather than high-performance graphics applications.

Audio capability is limited to **one digital I²S interface**, suitable for basic audio input/output or voice-oriented applications.

This reduced multimedia feature set reflects the i.MX91's focus on control, communication, and security rather than human-machine interfaces or vision processing.

Table 2: Multimedia and acceleration

Feature	Description/Availability
LCDIF LCD Interface Controller	Parallel display resolution up to 1366x768p60 or 1280x800p60
NPU	No
2D acceleration	No
Digital Audio	1× I²S interface

1.3.2 Interfaces

The Lino iMX91 exposes the interface counts listed in Table 3. The maximum possible values are subject to pin multiplexing and system configuration.

Table 3: Maximum available interfaces

Interface	Count
ADC inputs	4 2x always compatible + 2x vendor defined
CAN FD	2
Ethernet (RMII)	2
GPIO	10 dedicated[1] plus additional GPIOs via alternate functions
I²C[2]	3
I³C[2]	2
JTAG	1

Continued on next page

Table 3: Maximum available interfaces (Continued)

Interface	Count
PWM outputs	3 dedicated plus alternate functions
SD/SDIO	2 plus one dedicated to the on-board eMMC
SPI	2
UART (2-wire)	4
UART (CTS/RTS)	2
USB 2.0	2 1x OTG and 1x Host

[1] Sixteen of which come from the GPIO expander on the module.
[2] General-purpose I^2C/I^3C interfaces, excluding the dedicated I^2C bus used for the PMIC and other on-module devices.

1.3.3 Memory and Storage

The device features a **16-bit DRAM** interface supporting data rates **up to 2400 MT/s**. LPDDR4 memory with inline **ECC is supported**, providing **up to 2 GB** of DDR memory space.

Three **Ultra Secure Digital Host Controller (uSDHC)** interfaces are available:

- one **eMMC 5.1-compliant** interface with an 8-bit data bus supporting HS400 DDR signaling at **up to 400 MB/s**,
- one **4-bit SDXC** interface supporting extended-capacity SD cards, and
- one **4-bit SDIO** interface compliant with SD/SDIO 3.01, supporting **200 MHz SDR signaling at up to 100 MB/s**.

Table 4: Memory and storage overview

Parameter	Value
eMMC	up tp 256 GB
RAM (LPDDR4 x16)	up to 2 GB

1.3.4 Physical and Mechanical

The **Lino iMX91** features a compact **30.0 mm × 30.0 mm × 6.0mm** footprint. All interfaces are routed through **two 100-pin board-to-board connectors** (0.4 mm pitch), enabling a high interface density while maintaining a small mechanical envelope.

The module supports an **industrial temperature range of -40 °C to +85 °C** and is engineered for robust operation in demanding industrial and embedded environments, including applications subject to vibration, thermal stress, and continuous-duty operation.

Table 5: Physical and mechanical overview

Parameter	Value
Size	30.0 mm x 30.0 mm x 6.0mm
Temperature	-40 °C to +85 °C
Connectors	2x 100-pin board-to-board
Pitch	0.4 mm
Stacking height	1.5 mm or 4.0 mm[1]

[1] The stacking height depends on the mating connector on the carrier board.

1.3.5 Security Features

The Lino iMX91 leverages **NXP EdgeLock® security technology**, providing hardware-assisted secure boot, cryptographic acceleration, secure key storage, and lifecycle management capabilities.

Optional **TPM 2.0** support further enhances platform security, enabling secure device identity, measured boot, and integration with industrial security infrastructures.

Table 6: Security overview

Feature	Support
Secure Boot	Yes
Hardware Crypto	Yes
EdgeLock® Secure Enclave	Yes
TPM 2.0	Optional

1.4 Interface Overview

The Lino family features are grouped into **Always Compatible**, **Reserved**, and **Module-Specific** signals. The interface overview table classifies each interface by category and provides “up to” interface counts (subject to change with SoM configuration).

- *Always Compatible* interfaces are features that must be present on every SoM within the family, supporting long-term scalability and module interchangeability.
- *Reserved* interfaces are defined interfaces that may be absent on certain module variants due to SoC limitations or assembly options. Replacement functions must remain electrically compatible to avoid damage when mixing modules and carrier boards.
- *Module-Specific* interfaces are not guaranteed to be functionally or electrically compatible across different modules. Designs relying on these signals may limit module interchangeability and may cause functional issues or hardware damage if another module assigns a different function to the same pins.

Module-Specific Signals and Compatibility
Using *Module-Specific* signals may reduce compatibility across Lino variants. Whenever possible, prioritize *Always Compatible* interfaces to maximize portability and simplify upgrades.

Alternate functions are additional SoC multiplexed functions available on pins already assigned to *Always Compatible*, *Reserved*, or *Module-Specific* signals. **Alternate functions can be used only when the primary pin function is not required.**

Table 7 summarizes the interfaces available on the Lino iMX91, classifying each as *Always Compatible*,

Reserved, or *Module-Specific*, and listing the corresponding maximum interface counts. These values represent upper limits and are subject to pin multiplexing, SoC capabilities, and module configuration.

Table 7: Interfaces classification summary

Feature	Total	Always Compatible	Reserved	Module-specific
Analog				
ADC inputs	4	0	4	0
Audio				
I^2S	1	0	1	0
Display				
LCDIF	1	0	0	1[2]
Low Speed				
CAN FD	2	0	2	0
GPIO	10	10	0	TBD[1]
I^2C	3	3	0	0
I^3C	2	0	0	2[2]
PWM	3	3	0	0
SPI	2	1	1	0
UART (2-wire)	4	3	1	0
UART (RTS/CTS)	2	0	2	0
Network				
RGMII 10/100/1000 Mbps Ethernet	2	1	1	0
Storage				
SD/SDIO/MMC	2[3]	1	1	0
USB				
USB 2.0 Host	1	1	0	0
USB 2.0 OTG	1	1	0	0

[1] More pins may be used as GPIOs when other interfaces are not in use. Refer to the function multiplexing section for more information about the pins that may be used as GPIOs as alternate functions.
[2] Available as alternate functions.
[3] One additional interface is always used for the onboard eMMC.

1.5 Reference Documents

The following documents provide additional technical information required to design, validate, and maintain carrier boards and end products based on the Lino iMX91.

1.5.1 Toradex Developer Center

The Toradex Developer Center is the primary source for module documentation, software enablement, and integration guidelines. Verify that the selected content applies specifically to the **Lino family** and the **Lino iMX91** module variant.

https://developer.toradex.com/

1.5.2 Lino iMX91 Product Page

The product page provides a consolidated entry point for hardware and software collateral, ordering information, and ecosystem resources for the Lino iMX91.

https://www.toradex.com/computer-on-modules/lino-arm-family/nxp-iMX91

1.5.3 Carrier Board Design Guides

Toradex provides carrier board design guides and supporting references (including layout guidance, checklists, and best practices) intended to reduce risk during custom carrier board development.

https://developer.toradex.com/carrier-board-design/

1.5.4 Toradex Pinout Designer

The Toradex Pinout Designer is used to configure and review pin multiplexing and to compare interface availability across Toradex modules. It is recommended during schematic capture to confirm pin function selections and compatibility.

https://developer.toradex.com/carrier-board-design/pinout-designer/

1.5.5 NXP i.MX 91 Documentation

For detailed SoC-level electrical characteristics, programming model, and peripheral descriptions, refer to the NXP i.MX 91 documentation.

https://www.nxp.com/products/i.MX91

1.6 Naming Convention

This document follows a consistent naming convention to avoid ambiguity between **SoM-level** and **SoC-level** signals and features.

- **Lino signal names** refer to the module-level naming used in pin tables and interface descriptions
- **SoC ball names** refer to NXP i.MX 91 package ball identifiers
- **Alternate functions** refer to i.MX 91 `IOMUX` selections (`ALT` modes) associated with each SoC pad

Pay close attention to punctuation and spacing in names. Do not confuse the NXP i.MX 91 SoC with the Toradex Lino iMX91 SoM. Table 8 shows the differences in naming convention between the NXP and Toradex products.

Table 8: Toradex naming conventions

Name	Description
i.MX 91	NXP i.MX 91 System-on-Chip family
i.MX 9132	Specific variant of the NXP i.MX 91 family featuring a 1.4 GHz Cortex®-A55 core
Lino iMX91	Lino module based on the i.MX 91 SoC the term Lino iMX91 refers to all versions of the module
Lino iMX9132 2GB IT	Lino module based on the full featured i.MX 91 SoC with one 1.4 GHz core, 2 GB of RAM, and support for the industrial temperature range

1.7 Part Number Nomenclature

The part number nomenclature consists of a structured sequence of seven fields, each representing a specific configurable attribute of the device within the family. These fields encode key parameters such as

- family name
- i.MX 91 segment
- A-Core quantity
- Ram density
- eMMC storage capacity
- temperature sensor inclusion
- TPM security module inclusion
- operating temperature range

This arrangement enables precise identification and differentiation of variants based on their feature sets and functional capabilities. For clarification and detailed visualization, refer to Tables 9 and 10, which provides an illustrative example of the part number formation and the significance of each individual field.

Table 9: Part number nomenclature options

Field	Options
Family name	
LIMX91	Lino IMX91 system-on-module
i.MX 91 segment	
3	1.4G Hz, 2400 MT/s DDR speed
2	800 MHz, 1866 MT/s DDR speed
Cortex®-A core quantity	
1	Single Arm Cortex®-A55 core
RAM density	
-256M	256 MB LPDDR4
-512M	512 MB LPDDR4
-1G	1 GB LPDDR4
-2G	2 GB LPDDR4
eMMC capacity	
-4G	4 GB eMMC
-8G	8 GB eMMC
-16G	16 GB eMMC
-32G	32 GB eMMC
-64G	64 GB eMMC
-128G	128 GB eMMC
-256G	256 GB eMMC

Continued on next page

Table 9: Part number nomenclature options (Continued)

Field	Options
Temperature Sensor	
-T	Temperature sensor assembled
-N	Temperature sensor not assembled
TPM	
-T	TPM 2.0 assembled
-N	TPM 2.0 not assembled
Temperature	
-IT	Industrial temperature range (-40 °C to +85 °C)
-ET	Extended temperature range (-25 °C to +85 °C)
-CT	Commercial temperature range (0 °C to +70 °C)

Table 10: Part number examples

NPI Part Number	i.MX 91 Segment	CPU Core Count	Max. Freq.	RAM Density	eMMC Density	TPM	Temp. Range
LIMX9131-2G-32G-T-IT	3 (Full Featured)	1	1.4GHz	2 GB	32 GB	Yes	Industrial -40°C…85°C
LIMX9121-1G-8G-N-ET	2 (Reduced)	1	1.4GHz	1 GB	8 GB	No	Extended -25°C…85°C
LIMX9121-256M-4G-T-CT	2 (Reduced)	1	800MHz	256 MB	4 GB	Yes	Commercial 0°C…70°C

If your desired configuration is not available please get in touch with your Toradex sales representative mentioning the NPI part number desired.

Please note that the NPI part number nomenclature shown below does not reflect volume or sellable part numbers from Toradex. It should only be used to communicate your desired configuration.

1.8 Concept Configurations

Table 11 provides an overview of representative concept configurations for the Lino iMX91. These configurations illustrate possible feature combinations and interface usage, but do not necessarily reflect all interfaces available simultaneously. When designing a carrier board, we recommend using the Toradex Pinout Designer Tool [1] when designing your system.

Interfaces Availability
- Interface availability on the module depends on **pin multiplexing and configuration choices**.
- Use the **Toradex Pinout Designer** to validate interface usage for your design.
- Concept configurations are intended for **guidance only** and may not become off-the-shelf products.

[1] https://developer.toradex.com/carrier-board-design/pinout-designer/

Table 11: Concept configurations

Parameter	Lino iMX91 1GB IT	Lino iMX91 1GB CT	Lino iMX91 512MB IT	Lino iMX91 512MB CT
CPU Details				
CPU Name	i.MX 9131	i.MX 9131	i.MX 9121	i.MX 9121
CPU Type	1x Arm® Cortex®-A55	1x Arm® Cortex®-A55	1x Arm® Cortex®-A55	1x Arm® Cortex®-A55
CPU Clock	A55: 1.4GHz	A55: 1.4GHz	A55: 800MHz	A55: 800MHz
Memory				
RAM	1GB LPDDR4 x16	1GB LPDDR4 x16	512MB LPDDR4 x16	512MB LPDDR4 x16
Flash	8GB eMMC	8GB eMMC	4GB eMMC	4GB eMMC
Connectivity				
Analog Input contacts	2x	2x	2x	2x
CAN	2x CAN FD	2x CAN FD	2x CAN FD	2x CAN FD
Ethernet / LAN (S)(R)(G)MII	2x RGMII	2x RGMII	2x RGMII	2x RGMII
GPIO contacts	10x	10x	10x	10x
JTAG	1x	1x	1x	1x
I^2C general purpose[1]	2x I^3C	2x I^3C	2x I^3C	2x I^3C
PWM signals	3x	3x	3x	3x
SD/SDIO	2x	2x	2x	2x
SPI	2x	2x	2x	2x
UART	2x with handshake 2x without handshake	2x with handshake 2x without handshake	2x with handshake 2x without handshake	2x with handshake 2x without handshake
UART Console	1x	1x	1x	1x
USB 2.0	1x USB 2.0 OTG 1x USB 2.0 Host	1x USB 2.0 OTG 1x USB 2.0 Host	1x USB 2.0 OTG 1x USB 2.0 Host	1x USB 2.0 OTG 1x USB 2.0 Host
Multimedia				
Analog Audio	1x I^2S / PDM	1x I^2S / PDM	1x I^2S / PDM	1x I^2S / PDM
LCDIF	1x RGB up to 24-bit[2]	1x RGB up to 24-bit[2]	1x RGB up to 24-bit[2]	1x RGB up to 24-bit[2]
Operating System				
Torizon	Coming Soon	Coming Soon	Coming Soon	Coming Soon
Embedded Linux	Coming Soon	Coming Soon	Coming Soon	Coming Soon
Preinstalled OS	Toradex Easy Installer	Toradex Easy Installer	Toradex Easy Installer	Toradex Easy Installer
Android	Contact Us	Contact Us	Contact Us	Contact Us
QNX	Contact Us	Contact Us	Contact Us	Contact Us
FreeRTOS	Contact Us	Contact Us	Contact Us	Contact Us
Physical				
Size	30.0 x 30.0 mm	30.0 x 30.0 mm	30.0 x 30.0 mm	30.0 x 30.0 mm
Temperature	-40°C to 85°C	0°C to 70°C	-40°C to 85°C	0°C to 70°C
Shock / Vibration	TBD	TBD	TBD	TBD
Power Dissipation	TBD	TBD	TBD	TBD

[1] Besides dedicated interfaces for on-board components.
[2] Using GPIO[0..23] ALT3 alternate functions.

Concept Configurations
The configurations shown in Table 11 are merely concepts. They may or may not make it to be off the shelf products.

1.9 Configure-To-Order (CTO) Options

In addition to standard products, configuration variants may be available through Toradex Configure-To-Order (CTO) programs depending on platform maturity and productization status. Refer to the Toradex sales channel for feasibility and ordering details.

2 Architecture Overview

The **Lino iMX91 System-on-Module** integrates the NXP i.MX 91 SoC memory, storage, connectivity subsystems, and supporting components on a compact PCB. This high level of integration reduces carrier board design complexity while ensuring robust signal integrity, thermal stability, and compliance with relevant industry standards.

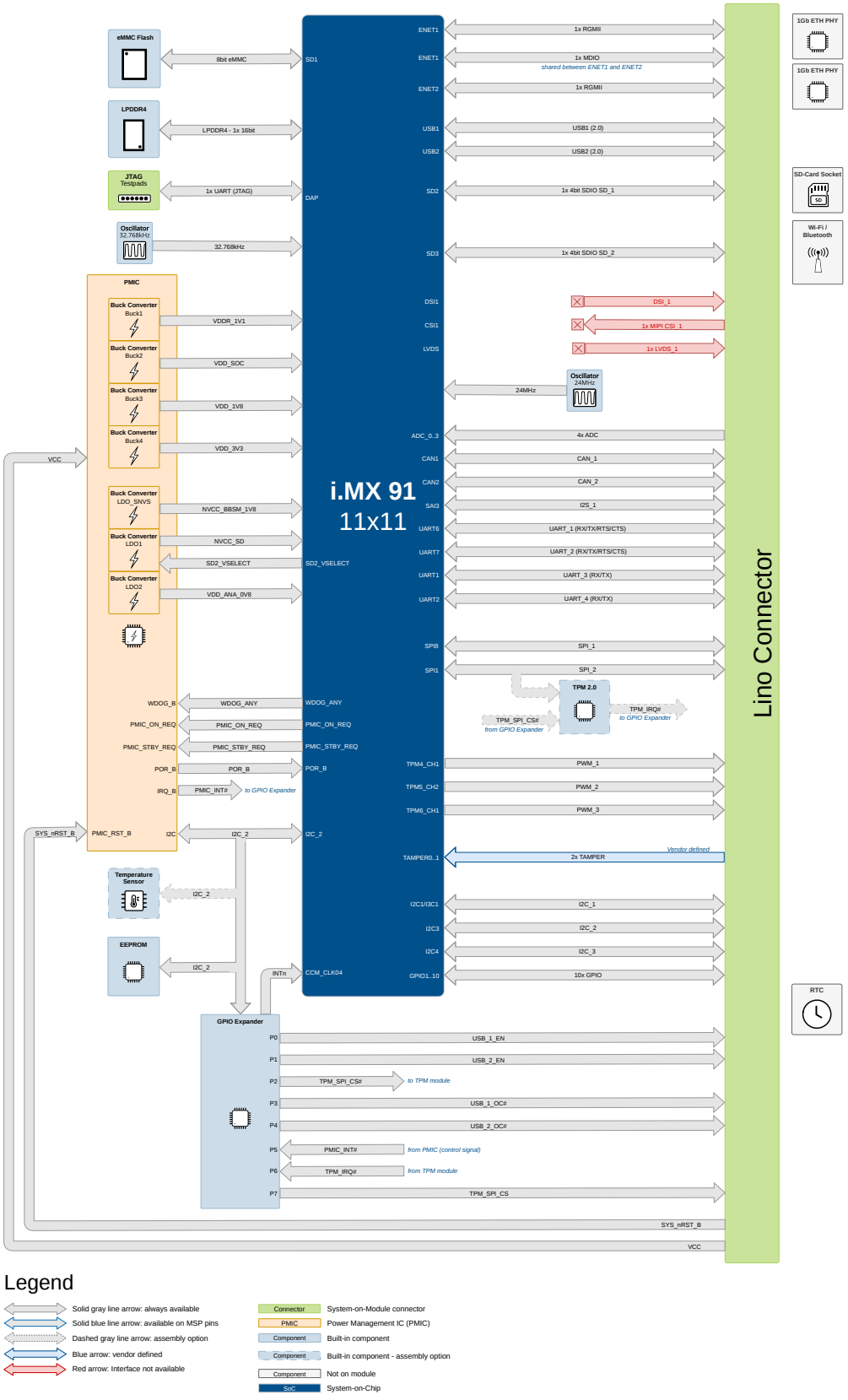
The overall architecture combines:

- **Application Processing:** Arm® Cortex®-A55 cores optimized for efficient application execution
- **Connectivity:** Integrated Ethernet MAC, USB, CAN FD, and SD/SDIO interfaces, with support for external wireless connectivity
- **Memmory and Storage:** up to 2 GB DDR4 DRAM and up to 32 GB eMMC on-module
- **Security:** EdgeLock® security subsystem with support for secure boot, cryptographic acceleration, and TPM 2.0 options
- **System Infrastructure:** EEPROM, I^2C-based peripherals, Power management, RTC, and on-module thermal monitoring

The functional block diagram of the Lino iMX91 SoM is shown in Figure 1.

2.1 Block Diagram

Figure 1: Block diagram



Legend

Solid gray line arrow: always available
Solid blue line arrow: available on MSP pins
Dashed gray line arrow: assembly option
Blue arrow: vendor defined
Red arrow: Interface not available

Connector — System-on-Module connector
PMIC — Power Management IC (PMIC)
Component — Built-in component
Component — Built-in component - assembly option
Component — Not on module
SoC — System-on-Chip

3 Connectors

The Lino iMX91 system-on-module uses **two Amphenol BergStak® board-to-board connectors** (refer to part number `10164228-1001A1RLF`) to form the mechanical and electrical interface between the module and the carrier board. Together, these connectors provide a robust, high-density interconnect capable of exposing the full feature set of the NXP i.MX 91 system-on-chip, including high-speed interfaces, low-speed control signals, and multiple power domains.

Each connector provides **100 signal positions with a 0.40 mm pitch**, resulting in a **total of 200 interconnect pins** across the module. The fine-pitch BergStak® connector family is optimized for compact layouts and supports controlled-impedance routing required by high-speed interfaces. The connector system is designed for high reliability and repeatable mating, making it suitable for both development and production deployments.

3.1 Mating Connector Options (Carrier Board)

Mating connectors for the carrier board are available with **two stacking height options**, allowing designers to select the appropriate board-to-board spacing based on mechanical clearance, thermal considerations, and enclosure constraints:

- **1.5 mm stacking height**
- **4.0 mm stacking height**

Selecting the correct mating connector variant and following the manufacturer's recommended footprints, keep-out zones, and routing guidelines is essential to ensure mechanical compatibility, signal integrity, and long-term reliability of the Lino iMX91 module-to-carrier board interface.

For more information, refer to the Section 11.

3.2 Pin Assignment

Tables 12 to 15 describe the main connectors (`X1` and `X2`) pinout and highlight the compatibility of each pin's function with the Lino Family Specification. A detailed explanation of the compatibility groups defined in the specification is available in Section 1.4.

Naming Convention
On the Lino iMX91 system-on-module, signals routed to connector **X1** are identified by pin names starting with **P** (for example, `P10` or `P12`), while signals routed to connector **X2** are identified by pin names starting with **S** (for example, `S28` or `S30`).

Table 12: X1 pin assignment odd connections

X1 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
P1	PWM_1	PWM NVCC_GPIO	GPIO_IO21 (T21)		Input with PD	–
P3	PWM_2		GPIO_IO18 (R18)		Input with PD	–
P5	PWM_3		GPIO_IO23 (U20)		Input with PD	–
P7	GND	Power	GND		–	–
P9	UART_1_RXD	UART NVCC_GPIO	GPIO_IO05 (L18)		Input with PD	–
P11	UART_1_TXD		GPIO_IO04 (L17)		Input with PD	–
P13	UART_2_RXD		GPIO_IO09 (M21)		Input with PD	–
P15	UART_2_TXD		GPIO_IO08 (M20)		Input with PD	–
P17	UART_3_RXD	Debug UART NVCC_AON	UART1_RXD (E20)		Input with PD	–
P19	UART_3_TXD		UART1_TXD (E21)		Input with PD	–
P21	GND	Power	GND		–	–
P23	UART_4_RXD	Debug UART NVCC_AON	UART2_RXD (F20)		Input with PD	–
P25	UART_4_TXD		UART2_TXD (F21)		Input with PD	–
P27	USB_1_EN	USB_1 USB1_VDD33	–	GPIO Expander (P0)	–	–
P29	USB_1_OC#		–	GPIO Expander (P3)	–	–
P31	USB_1_VBUS		USB1_VBUS (F12)		–	30kΩ series resistor 5V output
P33	USB_1_D_N		USB1_D_N (A14)		–	Differential pair
P35	USB_1_D_P		USB1_D_P (B14)		–	Differential pair
P37	GND	Power	GND		–	–
P39	MSP_31	Module-Specific pins	–		–	Not connected
P41	MSP_32				–	Not connected
P43	GND	Power			–	–

Continued on next page

Table 12: X1 pin assignment odd connections (Continued)

X1 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
P45	MSP_33	Module-Specific pins			–	Not connected
P47	MSP_34				–	Not connected
P49	GND	Power			–	–
P51	MSP_35	Module-Specific pins			–	Not connected
P53	MSP_36				–	Not connected
P55	GND	Power	GND		–	–
P57	USB_2_D_N	USB_2 USB2_VDD33	USB2_D_N (A15)		–	Differential pair
P59	USB_2_D_P		USB2_D_P (B15)		–	Differential pair
P61	USB_2_EN		–	GPIO Expander (P1)	–	–
P63	USB_2_OC#		–	GPIO Expander (P4)	–	–
P65	ETH_1_RGMII_INT#	RGMII NVCC_WAKEUP	PDM_BIT_STREAM1 (G18)		Input with PD	–
P67	ETH_MDIO		ENET1_MDIO (AA10)		Input with PD	–
P69	ETH_MDC		ENET1_MDC (AA11)		Input with PD	–
P71	GND	Power	GND		–	–
P73	ETH_1_RGMII_RXC	RGMII NVCC_WAKEUP	ENET1_RXC (AA7)		Input with PD	–
P75	ETH_1_RGMII_RX_CTL		ENET1_RX_CTL (Y8)		Input with PD	–
P77	ETH_1_RGMII_RXD_0		ENET1_RD0 (AA8)		Input with PD	–
P79	ETH_1_RGMII_RXD_1		ENET1_RD1 (Y9)		Input with PD	–
P81	ETH_1_RGMII_RXD_2		ENET1_RD2 (AA9)		Input with PD	–
P83	ETH_1_RGMII_RXD_3		ENET1_RD3 (Y10)		Input with PD	–
P85	ETH_1_RGMII_TX_CTL		ENET1_TX_CTL (V10)		Input with PD	–
P87	GND	Power	GND		–	–

Continued on next page

Table 12: X1 pin assignment odd connections (Continued)

X1 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
P89	ETH_1_RGMII_TXC	RGMII NVCC_WAKEUP	ENET1_TXC (U10)		Input with PD	–
P91	ETH_1_RGMII_TXD_3		ENET1_TD3 (V12)		Input with PD	–
P93	ETH_1_RGMII_TXD_2		ENET1_TD2 (U12)		Input with PD	–
P95	ETH_1_RGMII_TXD_1		ENET1_TD1 (T12)		Input with PD	–
P97	ETH_1_RGMII_TXD_0		ENET1_TD0 (W11)		Input with PD	–
P99	GND	Power	GND		–	–

Table 13: X1 pin assignment even connections

X1 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
P2	ADC_1	ADC AVDD_1P8_ADC	ADC_IN0 (B19)	–	Input without PU/PD	Resolution: 12 bits
P4	ADC_2		ADC_IN1 (A20)	–	Input without PU/PD	Resolution: 12 bits
P6	UART_1_RTS	UART NVCC_AON	GPIO_IO07 (L21)	–	Input with PD	–
P8	UART_1_CTS		GPIO_IO06 (L20)	–	Input with PD	–
P10	CAN_1_TX	CAN NVCC_AON	PDM_CLK (G17)	–	Input with PD	–
P12	CAN_1_RX		PDM_BIT_STREAM0 (J17)	–	Input with PD	–
P14	GND	Power	GND	–	–	–
P16	I2C_1_SDA	I^2C NVCC_AON or NVCC_GPIO	I2C1_SDA (C21)	–	Input with PD	–
P18	I2C_1_SCL		I2C1_SCL (C20)	–	Input with PD	–
P20	I2C_2_SDA		GPIO_IO28 (W20)	–	Input with PD	–
P22	I2C_2_SCL		GPIO_O29 (Y21)	–	Input with PD	–
P24	I2C_3_SDA		GPIO_IO02 (K20)	–	Input with PD	–
P26	I2C_3_SCL		GPIO_IO03 (K21)	–	Input with PD	–
P28	GND	Power	GND	–	–	–
P30	GPIO_1	GPIO NVCC_GPIO or NVCC_WAKEUP	GPIO_IO00 (J21)	–	Input with PD	–
P32	GPIO_2		GPIO_IO01 (J20)	–	Input with PD	–
P34	GPIO_3		GPIO_IO17 (R20)	–	Input with PD	–
P36	GPIO_4		GPIO_IO22 (U18)	–	Input with PD	–
P38	GPIO_5		GPIO_IO24 (U21)	–	Input with PD	–

Continued on next page

Table 13: X1 pin assignment even connections (Continued)

X1 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
P40	GPIO_6	GPIO NVCC_GPIO or NVCC_WAKEUP	CCM_CLKO1 (AA2)	–	Input with PU	–
P42	GPIO_7		CCM_CLKO2 (Y3)	–	Input with PU	–
P44	GPIO_8		CCM_CLKO3 (U4)	–	Input with PU	–
P46	GPIO_9		ENET2_MDC (Y7)	–	Input with PD	–
P48	GPIO_10		ENET2_MDIO (AA6)	–	Input with PD	–
P50	GND	Power	GND	–	–	–
P52	SPI_1_CLK	SPI NVCC_GPIO	GPIO_IO15 (P21)	–	Input with PD	–
P54	SPI_1_MISO		GPIO_IO13 (N21)	–	Input with PD	–
P56	SPI_1_MOSI		GPIO_IO14 (P20)	–	Input with PD	–
P58	SPI_1_CS		GPIO_IO12 (N20)	–	Input with PD	–
P60	SD_1_D2	SDIO NVCC_SD2	SD2_DATA2 (Y20)	–	Input with PD	–
P62	GND	Power	GND	–	–	–
P64	SD_1_D3	SDIO (continued) NVCC_SD2	SD2_DATA3 (AA20)	–	Input with PD	–
P66	SD_1_CMD		SD2_CMD (Y19)	–	Input with PD	–
P68	SD_1_CLK		SD2_CLK (AA19)	–	Input with PD	–
P70	GND		GND	–	–	–
P72	SD_1_PWR_EN		SD2_RESET_B (AA17)	–	Input with PD	–
P74	SD_1_D0		SD2_DATA0 (Y18)	–	Input with PD	–
P76	SD_1_D1		SD2_DATA1 (AA18)	–	Input with PD	–
P78	SD_1_CD#		SD2_CD_B (Y17)	–	Input with PD	10kΩ Pull-Up to 1.8V or 3.3V voltage dependent on VSELECT (V18)

Continued on next page

Table 13: X1 pin assignment even connections (Continued)

X1 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
P80	CTRL_RECOVERY_MICO#	System control NVCC_BBSM	–	–	–	10kΩ Pull-Up to 1.8V
P82	CTRL_PWR_BTN_MICO#		ONOFF (A19)	–	Input without PU/PD	100kΩ Pull-Up to 1.8V
P84	CTRL_PWR_EN_MOCI		–	–	–	–
P86	CTRL_RESET_MOCI#		–	–	–	–
P88	CTRL_RESET_MICO#		–	–	–	10kΩ Pull-Up to 1.8V
P90	VCC	Power	–	–	–	–
P92	VCC		–	–	–	–
P94	VCC		–	–	–	–
P96	VCC		–	–	–	–
P98	VCC		–	–	–	–
P100	VCC		–	–	–	–

Table 14: X2 pin assignment odd connections

X2 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
S1	ADC_3	ADC AVDD_1P8_ADC	ADC_IN2 (B20)	–	–	Resolution: 12 bits
S3	ADC_4		ADC_IN3 (B21)	–	–	Resolution: 12 bits
S5	UART_2_RTS	UART NVCC_AON	GPIO_IO11 (N18)	–	–	–
S7	UART_2_CTS		GPIO_IO10 (N17)	–	–	–
S9	GND	Power	–	–	–	–
S11	MSP_1	Module-Specific pins		–	–	Not connected
S13	MSP_2			–	–	Not connected
S15	MSP_3		TAMPER0 (B16)	–	–	–
S17	MSP_4			–	–	Not connected
S19	MSP_5			–	–	Not connected
S21	GND	Power	–	–	–	–
S23	MSP_6	Module-Specific pins		–	–	Not connected
S25	MSP_7			–	–	Not connected
S27	MSP_8		TAMPER1 (F14)	–	–	–
S29	MSP_9			–	–	Not connected
S31	MSP_10			–	–	Not connected
S33	GND	Power	–	–	–	–
S35	MSP_11	Module-Specific pins		–	–	Not connected
S37	MSP_12			–	–	Not connected
S39	MSP_13		–	–	–	Not connected
S41	MSP_14		–	–	–	Not connected
S43	MSP_15		–	–	–	Not connected
S45	GND	Power	–	–	–	–

Continued on next page

Table 14: X2 pin assignment odd connections (Continued)

X2 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
S47	MSP_16	Module-Specific pins	+V1.1_DDR	–	–	–
S49	MSP_17		USB2_VBUS (E14)	–	–	30kΩ series resistor 5V output
S51	MSP_18		+V0.8_VDD_SOC	–	–	–
S53	MSP_19		+V0.8_ANA	–	–	–
S55	MSP_20		–	–	–	Not connected
S57	GND	Power	–	–	–	–
S59	JTAG_TDI MSP_21	Module-Specific pins	DAP_TDI (W1)	–	–	–
S61	JTAG_TDO MSP_22		DAP_TDO_TRACESWO (Y2)	–	–	–
S63	MSP_23		+V1.8	–	–	–
S65	JTAG_TMS MSP_24		DAP_TMS_SWDIO (W2)	–	–	–
S67	JTAG_TCK MSP_25		DAP_TCLK_SWCLK (Y1)	–	–	–
S69	GND	Power	–	–	–	–
S71	ETH_2_RGMII_RXC	RGMII NVCC_WAKEUP	ENET2_RXC (AA3)	–	–	–
S73	ETH_2_RGMII_RX_CTL		ENET2_RX_CTL (Y4)	–	–	–
S75	GND	Power	–	–	–	–
S77	ETH_2_RGMII_RXD_0	RGMII NVCC_WAKEUP	ENET2_RD0 (AA4)	–	–	–
S79	ETH_2_RGMII_RXD_1		ENET2_RD1 (Y5)	–	–	–
S81	ETH_2_RGMII_RXD_2		ENET2_RD2 (AA5)	–	–	–
S83	ETH_2_RGMII_RXD_3		ENET2_RD3 (Y6)	–	–	–
S85	GND	Power	–	–	–	–

Continued on next page

Table 14: X2 pin assignment odd connections (Continued)

X2 pin odd pins	Lino specification signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
S87	ETH_2_RGMII_TXC	RGMII NVCC_WAKEUP	ENET2_TXC (U6)	–	–	–
S89	ETH_2_RGMII_TX_CTL		ENET2_TX_CTL (V6)	–	–	–
S91	ETH_2_RGMII_TXD_3		ENET2_TD3 (T10)	–	–	–
S93	ETH_2_RGMII_TXD_2		ENET2_TD2 (V8)	–	–	–
S95	GND	Power	–	–	–	–
S97	ETH_2_RGMII_TXD_1	RGMII NVCC_WAKEUP	ENET2_TD1 (U8)	–	–	–
S99	ETH_2_RGMII_TXD_0		ENET2_TD0 (T8)	–	–	–

Table 15: X2 pin assignment even connections

X2 pin even pins	Lino specification// signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
S2	SD_2_D2	SDIO NVCC_SD2	SD3_DATA2 (U14)	–	–	–
S4	SD_2_D3		SD3_DATA3 (T14)	–	–	–
S6	GND	Power	–	–	–	–
S8	SD_2_CMD	SDIO NVCC_SD2	SD3_CMD (U16)	–	–	–
S10	SD_2_CLK		SD3_CLK (V16)	–	–	–
S12	SD_2_D0		SD3_DATA0 (T16)	–	–	–
S14	GND	Power	–	–	–	–
S16	SD_2_D1	SDIO NVCC_SD2	SD3_DATA1 (V14)	–	–	–
S18	SPI_2_CS	SPI NVCC_WAKEUP	SAI1_TXFS (G21)	–	–	–
S20	SPI_2_CLK		SAI1_TXD0 (H21)	–	–	–
S22	SPI_2_MOSI		SAI1_RXD0 (H20)	–	–	–
S24	SPI_2_MISO		SAI1_TXC (G20)	–	–	–
S26	GND	Power	–	–	–	–
S28	CAN_2_TX	CAN NVCC_GPIO	GPIO_IO25 (V21)	–	–	–
S30	CAN_2_RX		GPIO_IO27 (W21)	–	–	–
S32	GND	Power	–	–	–	–
S34	DSI_1_D3_N	DSI MIPI_DSI1_VPH		–	–	Not connected
S36	DSI_1_D3_P			–	–	Not connected
S38	GND	Power	–	–	–	–
S40	DSI_1_D2_N	DSI MIPI_DSI1_VPH		–	–	Not connected
S42	DSI_1_D2_P			–	–	Not connected
S44	GND	Power	–	–	–	–

Continued on next page

Table 15: X2 pin assignment even connections (Continued)

X2 pin even pins	Lino specification// signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
S46	DSI_1_CLK_N	DSI MIPI_DSI1_VPH		–	–	Not connected
S48	DSI_1_CLK_P			–	–	Not connected
S50	GND	Power	–	–	–	–
S52	DSI_1_D1_N	DSI MIPI_DSI1_VPH		–	–	Not connected
S54	DSI_1_D1_P			–	–	Not connected
S56	GND	Power	–	–	–	–
S58	DSI_1_D0_N	DSI MIPI_DSI1_VPH		–	–	Not connected
S60	DSI_1_D0_P			–	–	Not connected
S62	GND	Power	–	–	–	–
S64	MSP_26	Module-Specific pins	–	–	–	Not connected
S66	MSP_27 P7 (GPIO)[1]		–	GPIO Expander (P7)	–	–
S68	MSP_28		–	–	–	Not connected
S70	MSP_29		–	–	–	Not connected
S72	MSP_30		–	–	–	Not connected
S74	GND	Power	–	–	–	–
S76	CSI_1_CLK_P	CSI MIPI_CSI1_VPH		–	–	Not connected
S78	CSI_1_CLK_N			–	–	Not connected
S80	GND	Power	–	–	–	–
S82	CSI_1_D1_P	CSI MIPI_CSI1_VPH		–	–	Not connected
S84	CSI_1_D1_N			–	–	Not connected
S86	GND	Power	–	–	–	–
S88	CSI_1_D0_P	CSI MIPI_CSI1_VPH		–	–	Not connected
S90	CSI_1_D0_N			–	–	Not connected

Continued on next page

Table 15: X2 pin assignment even connections (Continued)

X2 pin even pins	Lino specification// signal name	Signal group or power group	SoC function (ball number)	Non-Soc ball name	Reset state when asserted	Remarks
S92	GND	Power	–	–	–	–
S94	I2S_1_BCLK	I2S NVCC_GPIO	GPIO_IO16 (R21)	–	–	–
S96	I2S_1_SYNC		GPIO_IO26 (V20)	–	–	–
S98	I2S_1_D_OUT		GPIO_IO19 (R17)	–	–	–
S100	I2S_1_D_IN		GPIO_IO20 (T20)	–	–	–

[1] GPIO Expander PCAL6408AHKX (I^2C address 0x21, using PMIC_I2C interface) pin P7.

4 I/O Pins

4.1 Function Multiplexing

Low-speed I/O pins on the NXP i.MX 91 SoC can be configured for up to seven alternate functions. Most of these pins can also be used as GPIOs (general-purpose I/O, sometimes referred to as digital I/O). For example, the i.MX 91 signal connected to the board-to-board connector X1 on pin P52 exposes the SoC alternate function spi8.SCK (ALT4), which corresponds to the Lino standard function SPI_A_CLK. In addition to this SPI function, the pin can also be configured as:

- GPIO_IO15 (general-purpose input/output – ALT0)
- uart3.RX (UART3 receive signal – ALT1)
- isi.D[7] (parallel camera data input bit 7 – ALT2)
- lcdif.D[11] (parallel display data bus bit 11 – ALT3)
- uart8.RTS_B (UART8 request-to-send signal – ALT5)
- uart4.RX (UART4 receive signal – ALT6)
- flexio1.FLEXIO[15] (flexible I/O module channel 1 – ALT7)

Whenever possible, **it is strongly recommended to use functions that are compatible across all Lino modules**. This ensures maximum compatibility with standard software and other modules in the Lino Family.

Multiplexing Conflics
Some alternate functions are available on more than one pin. **Care must be taken to avoid assigning the same function to multiple pins simultaneously**, as this can result in system instability or undefined behavior.

Table 16 lists all pins that support alternate functions, along with the alternate functions available for each pin (sorted by the standard Lino function). Alternate functions highlighted in bold indicate the primary interfaces selected for optimal Lino family compatibility.

Table 16: Alternate functions (X1 odd pins, X1 even pins, X2 odd pins, X2 even pins)

X1/X2 pin	Lino function	SoC ball name	SoC ball ID	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	Reset Status
P12	CAN_1_RX	PDM_BIT_STREAM0	J17	pdm.BIT_STREAM[0]	mqs1.RIGHT	spi1.PCS1	tpm1.EXTCLK	lptmr1.ALT2	gpio1.IO[9]	**can1.RX**	–	Input with PD
P10	CAN_1_TX	PDM_CLK	G17	pdm.CLK	mqs1.LEFT	–	–	lptmr1.ALT1	gpio1.IO[8]	**can1.TX**	–	Input with PD
S30	CAN_2_RX	GPIO_IO27	W21	gpio2.IO[27]	usdhc3.DATA3	**can2.RX**	lcdif.D[23]	tpm6.CH3	dap.TMS_SWDIO	spi5.PCS1	flexio1.FLEXIO[27]	Input with PD
S28	CAN_2_TX	GPIO_IO25	V21	gpio2.IO[25]	usdhc3.DATA1	**can2.TX**	lcdif.D[21]	tpm4.CH3	dap.TCLK_SWCLK	spi7.PCS1	flexio1.FLEXIO[25]	Input with PD
–	CLKIN1 connected to ground	CLKIN1	B17	**anamix.CLKIN1**	anamix.esd_diode	–	–	–	–	–	–	Input without PU/PD
–	CLKIN2 connected to ground	CLKIN2	A18	**anamix.CLKIN2**	anamix.atx	–	–	–	–	–	–	Input without PU/PD
P82	CTRL_PWR_BTN_MICO#	ONOFF	A19	**bbsmmix.ONOFF**	–	–	–	–	–	–	–	Input without PU/PD
–	eMMC internal to the module	SD1_CLK	Y11	**usdhc1.CLK**	–	–	–	flexio1.FLEXIO[8]	gpio3.IO[8]	–	–	Input with PD
–	eMMC internal to the module	SD1_CMD	AA12	**usdhc1.CMD**	–	–	–	flexio1.FLEXIO[9]	gpio3.IO[9]	–	–	Input with PD
–	eMMC internal to the module	SD1_DATA0	AA14	**usdhc1.DATA0**	–	–	–	flexio1.FLEXIO[10]	gpio3.IO[10]	–	–	Input with PD
–	eMMC internal to the module	SD1_DATA1	AA15	**usdhc1.DATA1**	–	–	–	flexio1.FLEXIO[11]	gpio3.IO[11]	–	–	Input with PD
–	eMMC internal to the module	SD1_DATA2	AA16	**usdhc1.DATA2**	–	–	–	flexio1.FLEXIO[12]	gpio3.IO[12]	–	–	Input with PD
–	eMMC internal to the module	SD1_DATA3	AA13	**usdhc1.DATA3**	flexspi.A_SS1_B	–	–	flexio1.FLEXIO[13]	gpio3.IO[13]	–	–	Input with PD
–	eMMC internal to the module	SD1_DATA4	Y13	**usdhc1.DATA4**	flexspi.A_DATA[4]	–	–	flexio1.FLEXIO[14]	gpio3.IO[14]	–	–	Input with PD
–	eMMC internal to the module	SD1_DATA5	Y14	**usdhc1.DATA5**	flexspi.A_DATA[5]	usdhc1.RESET_B	–	flexio1.FLEXIO[15]	gpio3.IO[15]	–	–	Input with PD
–	eMMC internal to the module	SD1_DATA6	Y15	**usdhc1.DATA6**	flexspi.A_DATA[6]	usdhc1.CD_B	–	flexio1.FLEXIO[16]	gpio3.IO[16]	–	–	Input with PD
–	eMMC internal to the module	SD1_DATA7	Y16	**usdhc1.DATA7**	flexspi.A_DATA[7]	usdhc1.WP	–	flexio1.FLEXIO[17]	gpio3.IO[17]	–	–	Input with PD
–	eMMC internal to the module	SD1_STROBE	Y12	**usdhc1.STROBE**	flexspi.A_DQS	–	–	flexio1.FLEXIO[18]	gpio3.IO[18]	–	–	Input without PU/PD
P65	ETH_1_RGMII_INT#	PDM_BIT_STREAM1	G18	pdm.BIT_STREAM[1]	m33.NMI	spi2.PCS1	tpm2.EXTCLK	lptmr1.ALT3	**gpio1.IO[10]**	–	–	Input with PD

Continued on next page

Table 16: Alternate functions (X1 odd pins, X1 even pins, X2 odd pins, X2 even pins) (Continued)

X1/X2 pin	Lino function	SoC ball name	SoC ball ID	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	Reset Status
P75	ETH_1_RGMII_RX_CTL	ENET1_RX_CTL	Y8	**enet_qos.RGMII_RX_CTL**	uart3.DSR_B	–	usb2.OTG_PWR	flexio2.FLEXIO[8]	gpio4.IO[8]	–	–	Input with PD
P73	ETH_1_RGMII_RXC	ENET1_RXC	AA7	**enet_qos.RGMII_RXC**	enet_qos.RX_ER	–	–	flexio2.FLEXIO[9]	gpio4.IO[9]	–	–	Input with PD
P77	ETH_1_RGMII_RXD_0	ENET1_RD0	AA8	**enet_qos.RGMII_RD0**	uart3.RX	–	–	flexio2.FLEXIO[10]	gpio4.IO[10]	–	–	Input with PD
P79	ETH_1_RGMII_RXD_1	ENET1_RD1	Y9	**enet_qos.RGMII_RD1**	uart3.CTS_B	–	lptmr2.ALT1	flexio2.FLEXIO[11]	gpio4.IO[11]	–	–	Input with PD
P81	ETH_1_RGMII_RXD_2	ENET1_RD2	AA9	**enet_qos.RGMII_RD2**	–	–	lptmr2.ALT2	flexio2.FLEXIO[12]	gpio4.IO[12]	–	–	Input with PD
P83	ETH_1_RGMII_RXD_3	ENET1_RD3	Y10	**enet_qos.RGMII_RD3**	–	–	lptmr2.ALT3	flexio2.FLEXIO[13]	gpio4.IO[13]	–	–	Input with PD
P85	ETH_1_RGMII_TX_CTL	ENET1_TX_CTL	V10	**enet_qos.RGMII_TX_CTL**	uart3.DTR_B	–	–	flexio2.FLEXIO[6]	gpio4.IO[6]	–	–	Input with PD
P89	ETH_1_RGMII_TXC	ENET1_TXC	U10	**enet_qos.RGMII_TXC**	enet_qos.TX_ER	–	–	flexio2.FLEXIO[7]	gpio4.IO[7]	–	–	Input with PD
P97	ETH_1_RGMII_TXD_0	ENET1_TD0	W11	**enet_qos.RGMII_TD0**	uart3.TX	–	–	flexio2.FLEXIO[5]	gpio4.IO[5]	–	–	Input with PD
P95	ETH_1_RGMII_TXD_1	ENET1_TD1	T12	**enet_qos.RGMII_TD1**	uart3.RTS_B	i3c2.PUR	usb1.OTG_OC	flexio2.FLEXIO[4]	gpio4.IO[4]	i3c2.PUR_B	–	Input with PD
P93	ETH_1_RGMII_TXD_2	ENET1_TD2	U12	**enet_qos.RGMII_TD2**	I/O[1]	can2.RX	usb2.OTG_OC	flexio2.FLEXIO[3]	gpio4.IO[3]	–	–	Input with PD
P91	ETH_1_RGMII_TXD_3	ENET1_TD3	V12	**enet_qos.RGMII_TD3**	–	can2.TX	usb2.OTG_ID	flexio2.FLEXIO[2]	gpio4.IO[2]	–	–	Input with PD
S73	ETH_2_RGMII_RX_CTL	ENET2_RX_CTL	Y4	**enet2.RGMII_RX_CTL**	uart4.DSR_B	sai2.TX_DATA[0]	–	flexio2.FLEXIO[22]	gpio4.IO[22]	–	–	Input with PD
S71	ETH_2_RGMII_RXC	ENET2_RXC	AA3	**enet2.RGMII_RXC**	enet2.RX_ER	sai2.TX_DATA[1]	–	flexio2.FLEXIO[23]	gpio4.IO[23]	–	–	Input with PD
S77	ETH_2_RGMII_RXD_0	ENET2_RD0	AA4	**enet2.RGMII_RD0**	uart4.RX	sai2.TX_DATA[2]	–	flexio2.FLEXIO[24]	gpio4.IO[24]	–	–	Input with PD
S79	ETH_2_RGMII_RXD_1	ENET2_RD1	Y5	**enet2.RGMII_RD1**	spdif1.IN	sai2.TX_DATA[3]	–	flexio2.FLEXIO[25]	gpio4.IO[25]	–	–	Input with PD
S81	ETH_2_RGMII_RXD_2	ENET2_RD2	AA5	**enet2.RGMII_RD2**	uart4.CTS_B	sai2.MCLK	mqs2.RIGHT	flexio2.FLEXIO[26]	gpio4.IO[26]	–	–	Input with PD
S83	ETH_2_RGMII_RXD_3	ENET2_RD3	Y6	**enet2.RGMII_RD3**	spdif1.OUT	spdif1.IN	mqs2.LEFT	flexio2.FLEXIO[27]	gpio4.IO[27]	–	–	Input with PD
S89	ETH_2_RGMII_TX_CTL	ENET2_TX_CTL	V6	**enet2.RGMII_TX_CTL**	uart4.DTR_B	sai2.TX_SYNC	–	flexio2.FLEXIO[20]	gpio4.IO[20]	–	–	Input with PD
S87	ETH_2_RGMII_TXC	ENET2_TXC	U6	**enet2.RGMII_TXC**	enet2.TX_ER	sai2.TX_BCLK	–	flexio2.FLEXIO[21]	gpio4.IO[21]	–	–	Input with PD
S99	ETH_2_RGMII_TXD_0	ENET2_TD0	T8	**enet2.RGMII_TD0**	uart4.TX	sai2.RX_DATA[3]	–	flexio2.FLEXIO[19]	gpio4.IO[19]	–	–	Input with PD
S97	ETH_2_RGMII_TXD_1	ENET2_TD1	U8	**enet2.RGMII_TD1**	uart4.RTS_B	sai2.RX_DATA[2]	–	flexio2.FLEXIO[18]	gpio4.IO[18]	–	–	Input with PD
S97	ETH_2_RGMII_TXD_1	ENET2_TD1	U8	**enet2.RGMII_TD1**	uart4.RTS_B	sai2.RX_DATA[2]	–	flexio2.FLEXIO[18]	gpio4.IO[18]	–	–	Input with PD
S93	ETH_2_RGMII_TXD_2	ENET2_TD2	V8	**enet2.RGMII_TD2**	I/O[2]	sai2.RX_DATA[1]	–	flexio2.FLEXIO[17]	gpio4.IO[17]	–	–	Input with PD

Continued on next page

Table 16: Alternate functions (X1 odd pins, X1 even pins, X2 odd pins, X2 even pins) (Continued)

X1/X2 pin	Lino function	SoC ball name	SoC ball ID	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	Reset Status
S91	ETH_2_RGMII_TXD_3	ENET2_TD3	T10	**enet2.RGMII_TD3**	–	sai2.RX_DATA[0]	–	flexio2.FLEXIO[16]	gpio4.IO[16]	–	–	Input with PD
P69	ETH_MDC	ENET1_MDC	AA11	**enet_qos.MDC**	uart3.DCB_B	i3c2.SCL	usb1.OTG_ID	flexio2.FLEXIO[0]	gpio4.IO[0]	–	–	Input with PD
P67	ETH_MDIO	ENET1_MDIO	AA10	**enet_qos.MDIO**	uart3.RIN_B	i3c2.SDA	usb1.OTG_PWR	flexio2.FLEXIO[1]	gpio4.IO[1]	–	–	Input with PD
P30	GPIO_1	GPIO_IO00	J21	**gpio2.IO[0]**	i2c3.SDA	isi.PCLK	lcdif.PCLK	spi6.PCS0	uart5.TX	i2c5.SDA	flexio1.FLEXIO[0]	Input with PD
P32	GPIO_2	GPIO_IO01	J20	**gpio2.IO[1]**	i2c3.SCL	isi.D[0]	lcdif.DE	spi6.SIN	uart5.RX	i2c5.SCL	flexio1.FLEXIO[1]	Input with PD
P34	GPIO_3	GPIO_IO17	R20	**gpio2.IO[17]**	sai3.MCLK	isi.D[8]	lcdif.D[13]	uart3.RTS_B	spi4.PCS1	uart4.RTS_B	flexio1.FLEXIO[17]	Input with PD
P36	GPIO_4	GPIO_IO22	U18	**pio2.IO[22]**	usdhc3.CLK	spdif1.IN	lcdif.D[18]	tpm5.CH1	tpm6.EXTCLK	i2c5.SDA	flexio1.FLEXIO[22]	Input with PD
P38	GPIO_5	GPIO_IO24	U21	**gpio2.IO[24]**	usdhc3.DATA0	–	lcdif.D[20]	tpm3.CH3	dap.TDO_TRACESWO	spi6.PCS1	flexio1.FLEXIO[24]	Input with PD
P40	GPIO_6	CCM_CLKO1	AA2	ccmsrcgpcmix.CLKO1	–	–	–	flexio1.FLEXIO[26]	**gpio3.IO[26]**	–	–	Output low
P42	GPIO_7	CCM_CLKO2	Y3	ccmsrcgpcmix.CLKO2	–	–	–	flexio1.FLEXIO[27]	**gpio3.IO[27]**	–	–	Output low
P44	GPIO_8	CCM_CLKO3	U4	ccmsrcgpcmix.CLKO3	–	–	–	flexio2.FLEXIO[28]	**gpio4.IO[28]**	–	–	Input with PD
P46	GPIO_9	ENET2_MDC	Y7	enet2.MDC	uart4.DCB_B	sai2.RX_SYNC	–	flexio2.FLEXIO[14]	**gpio4.IO[14]**	–	–	Input with PD
P48	GPIO_10	ENET2_MDIO	AA6	enet2.MDIO	uart4.RIN_B	sai2.RX_BCLK	–	flexio2.FLEXIO[15]	**gpio4.IO[15]**	–	–	Input with PD
P18	I2C_1_SCL	I2C1_SCL	C20	**i2c1.SCL**	i3c1.SCL	uart1.DCB_B	tpm2.CH0	–	gpio1.IO[0]	–	–	Input with PD
P16	I2C_1_SDA	I2C1_SDA	C21	**i2c1.SDA**	i3c1.SDA	uart1.RIN_B	tpm2.CH1	–	gpio1.IO[1]	–	–	Input with PD
P22	I2C_2_SCL	GPIO_IO29	Y21	gpio2.IO[29]	**i2c3.SCL**	–	–	–	–	–	flexio1.FLEXIO[29]	Input with PD
P20	I2C_2_SDA	GPIO_IO28	W20	gpio2.IO[28]	**i2c3.SDA**	–	–	–	–	–	flexio1.FLEXIO[28]	Input with PD
P26	I2C_3_SCL	GPIO_IO03	K21	gpio2.IO[3]	**i2c4.SCL**	isi.LINE_VALID	lcdif.HSYNC	spi6.SCK	uart5.RTS_B	i2c6.SCL	flexio1.FLEXIO[3]	Input with PD
P24	I2C_3_SDA	GPIO_IO02	K20	gpio2.IO[2]	**i2c4.SDA**	isi.FRAME_VALID	lcdif.VSYNC	spi6.SOUT	uart5.CTS_B	i2c6.SDA	flexio1.FLEXIO[2]	Input with PD
S94	I2S_1_BCLK	GPIO_IO16	R21	gpio2.IO[16]	**sai3.TX_BCLK**	pdm.BIT_STREAM[2]	lcdif.D[12]	uart3.CTS_B	spi4.PCS2	uart4.CTS_B	flexio1.FLEXIO[16]	Input with PD
S100	I2S_1_D_IN	GPIO_IO20	T20	gpio2.IO[20]	**sai3.RX_DATA[0]**	pdm.BIT_STREAM[0]	lcdif.D[16]	spi5.SOUT	spi4.SOUT	tpm3.CH1	flexio1.FLEXIO[20]	Input with PD
S98	I2S_1_D_OUT	GPIO_IO19	R17	gpio2.IO[19]	**sai3.RX_SYNC**	pdm.BIT_STREAM[3]	lcdif.D[15]	spi5.SIN	spi4.SIN	tpm6.CH2	sai3.TX_DATA[0]	Input with PD
S96	I2S_1_SYNC	GPIO_IO26	V20	gpio2.IO[26]	usdhc3.DATA2	pdm.BIT_STREAM[1]	lcdif.D[22]	tpm5.CH3	dap.TDI	spi8.PCS1	**sai3.TX_SYNC**	Input with PD

Continued on next page

Table 16: Alternate functions (X1 odd pins, X1 even pins, X2 odd pins, X2 even pins) (Continued)

X1/X2 pin	Lino function	SoC ball name	SoC ball ID	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	Reset Status
–	Internal I2C on-board I2C and PMIC	I2C2_SCL	D20	**i2c2.SCL**	i3c1.PUR	uart2.DCB_B	tpm2.CH2	sai1.RX_SYNC	gpio1.IO[2]	i3c1.PUR_B	–	Input with PD
–	Internal I2C on-board I2C and PMIC	I2C2_SDA	D21	**i2c2.SDA**	–	uart2.RIN_B	tpm2.CH3	sai1.RX_BCLK	gpio1.IO[3]	–	–	Input with PD
–	Interrupt GPIO Expander	CCM_CLKO4	V4	ccmsrcgpcmix.CLKO4	–	–	–	flexio2.FLEXIO[29]	**gpio4.IO[29]**	–	–	Input with PD
S59	MSP_21 JTAG interface	DAP_TCLK_SWCLK	Y1	**dap.TCLK_SWCLK**	–	–	–	flexio1.FLEXIO[30]	gpio3.IO[30]	uart5.CTS_B	–	Input with PD
S61	MSP_22 JTAG interface	DAP_TDO_TRACESWO	Y2	**dap.TDO_TRACESWO**	mqs2.RIGHT	–	can2.RX	flexio1.FLEXIO[31]	gpio3.IO[31]	uart5.TX	–	Input without PU/PD
S65	MSP_24 JTAG interface	DAP_TMS_SWDIO	W2	**dap.TMS_SWDIO**	–	–	–	flexio2.FLEXIO[31]	gpio3.IO[29]	uart5.RTS_B	–	Input with PU
S67	MSP_25 JTAG interface	DAP_TDI	W1	**dap.TDI**	mqs2.LEFT	–	can2.TX	flexio2.FLEXIO[30]	gpio3.IO[28]	uart5.RX	–	Input with PU
S15	MSP_3 module-specific pin	TAMPER0	B16	**bbsmmix.TAMPER0**	–	–	–	–	–	–	–	Input with PD
S27	MSP_8 module-specific pin	TAMPER1	F14	**bbsmmix.TAMPER1**	–	–	–	–	–	–	–	Input with PD
–	PMIC_ON_REQ PMIC_ON_REQ[3] – PMIC	PMIC_ON_REQ	A17	**bbsmmix.PMIC_ON_REQ**	–	–	–	–	–	–	–	Output high without PU/PD
–	PMIC_STBY_REQ PMIC_STBY_REQ[3] – PMIC	PMIC_STBY_REQ	B18	**bbsmmix.PMIC_STBY_REQ**	–	–	–	–	–	–	–	Output low withput PU/PD
–	POR_B POR_B – PMIC	POR_B	A16	**bbsmmix.POR_B**	–	–	–	–	–	–	–	Input without PU/PD
P1	PWM_1	GPIO_IO21	T21	gpio2.IO[21]	sai3.TX_DATA[0]	pdm.CLK	lcdif.D[17]	spi5.SCK	spi4.SCK	**tpm4.CH1**	sai3.RX_BCLK	Input with PD
P3	PWM_2	GPIO_IO18	R18	gpio2.IO[18]	sai3.RX_BCLK	isi.D[9]	lcdif.D[14]	spi5.PCS0	spi4.PCS0	**tpm5.CH2**	flexio1.FLEXIO[18]	Input with PD
P5	PWM_3	GPIO_IO23	U20	gpio2.IO[23]	usdhc3.CMD	spdif1.OUT	lcdif.D[19]	**tpm6.CH1**	–	i2c5.SCL	flexio1.FLEXIO[23]	Input with PD
P78	SD_1_CD#	SD2_CD_B	Y17	**usdhc2.CD_B**	enet_qos.1588_EVENT0_IN	i3c2.SCL	–	flexio1.FLEXIO[0]	gpio3.IO[0]	–	–	Input with PD
P68	SD_1_CLK	SD2_CLK	AA19	**usdhc2.CLK**	enet_qos.1588_EVENT0_OUT	i3c2.SDA	–	flexio1.FLEXIO[1]	gpio3.IO[1]	–	–	Input with PD
P66	SD_1_CMD	SD2_CMD	Y19	**usdhc2.CMD**	enet2.1588_EVENT0_IN	i3c2.PUR	i3c2.PUR_B	flexio1.FLEXIO[2]	gpio3.IO[2]	–	–	Input with PD

Continued on next page

Table 16: Alternate functions (X1 odd pins, X1 even pins, X2 odd pins, X2 even pins) (Continued)

X1/X2 pin	Lino function	SoC ball name	SoC ball ID	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	Reset Status
P74	SD_1_D0	SD2_DATA0	Y18	**usdhc2.DATA0**	enet2.1588_EVENT0_OUT	can2.TX	–	flexio1.FLEXIO[3]	gpio3.IO[3]	–	–	Input with PD
P76	SD_1_D1	SD2_DATA1	AA18	**usdhc2.DATA1**	enet2.1588_EVENT1_IN	can2.RX	–	flexio1.FLEXIO[4]	gpio3.IO[4]	–	–	Input with PD
P60	SD_1_D2	SD2_DATA2	Y20	**usdhc2.DATA2**	enet2.1588_EVENT1_OUT	mqs2.RIGHT	–	flexio1.FLEXIO[5]	gpio3.IO[5]	–	–	Input with PD
P64	SD_1_D3	SD2_DATA3	AA20	**usdhc2.DATA3**	lptmr2.ALT1	mqs2.LEFT	–	flexio1.FLEXIO[6]	gpio3.IO[6]	–	–	Input with PD
P72	SD_1_PWR_EN	SD2_RESET_B	AA17	**usdhc2.RESET_B**	lptmr2.ALT2	–	–	flexio1.FLEXIO[7]	gpio3.IO[7]	–	–	Input with PD
S10	SD_2_CLK	SD3_CLK	V16	**usdhc3.CLK**	flexspi.A_SCLK	–	–	flexio1.FLEXIO[20]	gpio3.IO[20]	–	–	Input with PD
S8	SD_2_CMD	SD3_CMD	U16	**usdhc3.CMD**	flexspi.A_SS0_B	–	–	flexio1.FLEXIO[21]	gpio3.IO[21]	–	–	Input with PD
S12	SD_2_D0	SD3_DATA0	T16	**usdhc3.DATA0**	flexspi.A_DATA[0]	–	–	flexio1.FLEXIO[22]	gpio3.IO[22]	–	–	Input with PD
S16	SD_2_D1	SD3_DATA1	V14	**usdhc3.DATA1**	flexspi.A_DATA[1]	–	–	flexio1.FLEXIO[23]	gpio3.IO[23]	–	–	Input with PD
S2	SD_2_D2	SD3_DATA2	U14	**usdhc3.DATA2**	flexspi.A_DATA[2]	–	–	flexio1.FLEXIO[24]	gpio3.IO[24]	–	–	Input with PD
S4	SD_2_D3	SD3_DATA3	T14	**usdhc3.DATA3**	flexspi.A_DATA[3]	–	–	flexio1.FLEXIO[25]	gpio3.IO[25]	–	–	Input with PD
–	SD2_VSELECT SD_VSEL – PMIC	SD2_VSELECT	V18	**usdhc2.VSELECT**	usdhc2.WP	lptmr2.ALT3	–	flexio1.FLEXIO[19]	gpio3.IO[19]	–	–	Input with PD
P52	SPI_1_CLK	GPIO_IO15	P21	gpio2.IO[15]	uart3.RX	isi.D[7]	lcdif.D[11]	**spi8.SCK**	uart8.RTS_B	uart4.RX	flexio1.FLEXIO[15]	Input with PD
P58	SPI_1_CS	GPIO_IO12	N20	gpio2.IO[12]	tpm3.CH2	pdm.BIT_STREAM[2]	lcdif.D[8]	**spi8.PCS0**	uart8.TX	i2c8.SDA	sai3.RX_SYNC	Input with PD
P54	SPI_1_MISO	GPIO_IO13	N21	gpio2.IO[13]	tpm4.CH2	pdm.BIT_STREAM[3]	lcdif.D[9]	**spi8.SIN**	uart8.RX	i2c8.SCL	flexio1.FLEXIO[13]	Input with PD
P56	SPI_1_MOSI	GPIO_IO14	P20	gpio2.IO[14]	uart3.TX	isi.D[6]	lcdif.D[10]	**spi8.SOUT**	uart8.CTS_B	uart4.TX	flexio1.FLEXIO[14]	Input with PD
S20	SPI_2_CLK	SAI1_TXD0	H21	sai1.TX_DATA[0]	uart2.RTS_B	**spi1.SCK**	uart1.DTR_B	can1.TX	gpio1.IO[13]/ccmsrcgpcmix.BOOT_MODE[3]	–	–	Input with PD
S18	SPI_2_CS	SAI1_TXFS	G21	sai1.TX_SYNC	sai1.TX_DATA[1]	**spi1.PCS0**	uart2.DTR_B	mqs1.LEFT	gpio1.IO[11]/ccmsrcgpcmix.BOOT_MODE[2]	–	–	Input with PD
S24	SPI_2_MISO	SAI1_TXC	G20	sai1.TX_BCLK	uart2.CTS_B	**spi1.SIN**	uart1.DSR_B	can1.RX	gpio1.IO[12]	–	–	Input with PD
S22	SPI_2_MOSI	SAI1_RXD0	H20	sai1.RX_DATA[0]	sai1.MCLK	**spi1.SOUT**	uart2.DSR_B	mqs1.RIGHT	gpio1.IO[14]	–	–	Input with PD
P8	UART_1_CTS	GPIO_IO06	L20	gpio2.IO[6]	tpm5.CH0	pdm.BIT_STREAM[1]	lcdif.D[2]	spi7.SOUT	**uart6.CTS_B**	i2c7.SDA	flexio1.FLEXIO[6]	Input with PD
P6	UART_1_RTS	GPIO_IO07	L21	gpio2.IO[7]	spi3.PCS1	isi.D[1]	lcdif.D[3]	spi7.SCK	**uart6.RTS_B**	i2c7.SCL	flexio1.FLEXIO[7]	Input with PD
P9	UART_1_RXD	GPIO_IO05	L18	gpio2.IO[5]	tpm4.CH0	pdm.BIT_STREAM[0]	lcdif.D[1]	spi7.SIN	**uart6.RX**	i2c6.SCL	flexio1.FLEXIO[5]	Input with PD

Continued on next page

Table 16: Alternate functions (X1 odd pins, X1 even pins, X2 odd pins, X2 even pins) (Continued)

X1/X2 pin	Lino function	SoC ball name	SoC ball ID	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	Reset Status
P11	UART_1_TXD	GPIO_IO04	L17	gpio2.IO[4]	tpm3.CH0	pdm.CLK	lcdif.D[0]	spi7.PCS0	**uart6.TX**	i2c6.SDA	flexio1.FLEXIO[4]	Input with PD
S7	UART_2_CTS	GPIO_IO10	N17	gpio2.IO[10]	spi3.SOUT	isi.D[4]	lcdif.D[6]	tpm4.EXTCLK	**uart7.CTS_B**	i2c8.SDA	flexio1.FLEXIO[10]	Input with PD
S5	UART_2_RTS	GPIO_IO11	N18	gpio2.IO[11]	spi3.SCK	isi.D[5]	lcdif.D[7]	tpm5.EXTCLK	**uart7.RTS_B**	i2c8.SCL	flexio1.FLEXIO[11]	Input with PD
P13	UART_2_RXD	GPIO_IO09	M21	gpio2.IO[9]	spi3.SIN	isi.D[3]	lcdif.D[5]	tpm3.EXTCLK	**uart7.RX**	i2c7.SCL	flexio1.FLEXIO[9]	Input with PD
P15	UART_2_TXD	GPIO_IO08	M20	gpio2.IO[8]	spi3.PCS0	isi.D[2]	lcdif.D[4]	tpm6.CH0	**uart7.TX**	i2c7.SDA	flexio1.FLEXIO[8]	Input with PD
P17	UART_3_RXD	UART1_RXD	E20	**uart1.RX**	seco.RX	spi2.SIN	tpm1.CH0	–	gpio1.IO[4]	–	–	Input with PD
P19	UART_3_TXD	UART1_TXD	E21	**uart1.TX**	seco.TX	spi2.PCS0	tpm1.CH1	–	gpio1.IO[5]/ccmsrcgpcmix.BOOT_MODE[0]	–	–	Input with PD
P23	UART_4_RXD	UART2_RXD	F20	**uart2.RX**	uart1.CTS_B	spi2.SOUT	tpm1.CH2	sai1.MCLK	gpio1.IO[6]	–	–	Input with PD
P25	UART_4_TXD	UART2_TXD	F21	**uart2.TX**	uart1.RTS_B	spi2.SCK	tpm1.CH3	–	gpio1.IO[7]/ccmsrcgpcmix.BOOT_MODE[1]	–	–	Input with PD
–	WDOB_ANY WDOG_B – PMIC	WDOG_ANY	J18	**wdog1.WDOG_ANY**	–	–	–	–	gpio1.IO[15]	–	–	Input with PU
–	XTALI_24M FOUT – RTC	RTC_XTALI	E16	**bbsmmix.RTC**	–	–	–	–	–	–	–	–
–	24MHz Oscillator internal signal	XTALI_24M	D18	**anamix.xtali_24M**	–	–	–	–	–	–	–	–
–	24MHz Oscillator internal signal	XTALO_24M	E18	**anamix.xtalo_24M**	–	–	–	–	–	–	–	–

[1] INPUT=enet_qos.TX_CLK or OUTPUT=ccmsrcgpcmix.ENET_CLK_ROOT
[2] INPUT=enet2.TX_CLK or OUTPUT=ccmsrcgpcmix.ENET_REF_CLK_ROOT
[3] 1 MΩ pull-down to GND.
***Bold** text*: Alternate function that provides maximum compatibility between modules of the Lino family.

Rows in : Module-specific interface.

Rows in : Pin used to determine boot mode at power-on reset. It is output-only because it is isolated from the SoM's board-to-board connector by a 3-state buffer. After reset, it operates according to the selected alternate function.
Reset status of pins: PD: pull-down resistor; PU: pull-up resistor.

4.2 Pin Multiplexing and Control

The alternate function of each pin can be configured independently. Each pin features a **multiplexing control register** named `SW_MUX_CTL_PAD_<PAD_NAME>` that selects one of the available alternate functions through the **`MUX_MODE` field**. While the architecture supports up to **12 alternate modes**, in this implementation each pin uses **ALT0 to ALT7**. The register also includes the **SION bit**, allowing the input path of the pad to be forced regardless of the configured function direction.

In addition, each pin has an associated **pad control register** named `SW_PAD_CTL_PAD_<PAD_NAME>`. This register configures the **electrical characteristics** of the pad, including **pull-up and pull-down resistors, drive strength, slew rate, hysteresis, and open-drain operation**, allowing fine control of signal behavior.

IOMUX Register Implementation
Only the **minimum number of registers required by software** are implemented by hardware. For example, if only **ALT0 and ALT1** modes are used on a pad, the hardware may generate only a **single-bit `MUX_MODE` field** in the corresponding software mux control register. The software mux control registers may also allow **forcing a pad to act as an input** regardless of the functional direction driven. This behavior is controlled through the **SION (Software Input On) bit**, which can be useful for **loopback testing and GPIO data capture**.

In addition, each pin has an associated **control register** that allows configuration of electrical characteristics such as **pull-up and pull-down resistors, drive strength, slew rate, hysteresis, and open-drain operation**. This register is named `SW_PAD_CTL_PAD_<PAD_NAME>` to configure specific pad settings of each pad.

Control Fields Availability
Not all control fields are available for every pad.

Input functions that can be routed to more than one physical pin require an additional **input multiplexer**. This multiplexer is configured using a register named `IOMUXC_x_SELECT_INPUT`, where where x corresponds to the name of the input function.

For a complete description of the available register fields and configuration options, refer to the **NXP i.MX 91 Reference Manual** [2].

4.3 Pin Reset Status

After a reset, i.MX 91 pins can be set in different modes. Most are pulled low, while some are in a high-impedance state or pulled high. Refer to Table 16 for a complete list of the reset states of each pin. Once the bootloader is running, pins and their states can be reconfigured.

Reset States
Pin reset states are only guaranteed during the release of the reset signal. During the power-up sequence, pin states may be undefined until the corresponding I/O bank voltage is enabled on the module.

[2] https://www.nxp.com/webapp/Download?colCode=IMX91RM

5 Interfaces Description

5.1 ADC – Analog to Digital Converter

The Lino iMX91 integrates a single **successive-approximation register (SAR)** Analog-to-Digital Converter (ADC) module that provides high-speed, multi-channel analog signal acquisition. The ADC supports **12-bit resolution** across **four input channels** with a **maximum sampling rate of 1 MS/s**.

The ADC operates from the **1.8 V analog supply** (`VDD_ANAx_1P8`), which also powers other sensitive analog and mixed-signal blocks. For applications that require improved Effective Number Of Bits (ENOB) and enhanced conversion accuracy, the use of an external voltage reference on `VDD_ANAx_1P8` is recommended when the ADC is enabled.

Best Practice for Unused ADC Inputs
If the ADC is not used, it is recommended to tie the ADC input pins to ground to minimize noise coupling and power consumption.

Table 17 lists the pins available for ADC functionality, while the SoC datasheet provides the complete electrical specifications for the ADC interface.

Table 17: ADC interface

Connector pin	Lino signal name	SoC ball name	SoC alternate function	I/O	Description/Remarks
X1 connector					
P2	ADC_1	ADC_IN0	ALT0 anamix.adc_in0	I	Analog Input 1
P4	ADC_2	ADC_IN1	ALT0 anamix.adc_in1	I	Analog Input 2
X2 connector					
S1	ADC_3	ADC_IN2	ALT0 anamix.adc_in2	I	Analog Input 3
S3	ADC_4	ADC_IN3	ALT0 anamix.adc_in3	I	Analog Input 4

5.2 CAN – Controller Area Network

The Lino iMX91 system-on-module provides **Controller Area Network (CAN)** connectivity through the FlexCAN controllers integrated in the NXP i.MX 91 system-on-chip. The CAN subsystem is designed for robust, real-time communication in automotive, industrial, and embedded control applications, offering reliable operation in electrically noisy environments.

The i.MX 91 integrates **two independent CAN controllers** (`CAN1` and `CAN2`), both of which are exposed on the Lino iMX91 board-to-board connector. Each controller provides dedicated `Tx` and `Rx` signals and is intended to be connected to an external CAN transceiver on the carrier board. The CAN interfaces support both **Classical CAN** and **CAN FD (Flexible Data-rate)** operation, enabling higher data throughput while maintaining backward compatibility with legacy CAN networks.

5.2.1 Key Features and Supported Capabilities

- **Number of controllers:** 2x CAN instances (`CAN1` and `CAN2`)
- **Protocol support:** classical CAN (`ISO 11898-1`) and CAN FD (`ISO 11898-1:2015`)
- **Bit rates:**
  - up to **1 Mbps** nominal bit rate (arbitration phase)
  - up to **5 Mbps** data bit rate (CAN FD data phase), depending on transceiver and bus conditions

- **Frame formats:**
  - standard (11-bit) and Extended (29-bit) identifiers
  - CAN FD frames with payloads up to 64 bytes
- **Reliability features:** error detection and fault confinement, with automatic retransmission and bus-off recovery support
- **System integration:** interrupt and DMA-capable operation

Table 18 lists the available CAN interface signals on the Lino iMX91 board-to-board connectors, including the corresponding SoC ball names, alternate functions, signal direction, and functional descriptions for both `CAN1` and `CAN2`.

Table 18: CAN interfaces

X1/X2 pin	Lino signal name	SoC ball name	SoC alternate function	SoC alternate function name	I/O	Description/Remarks
CAN interface 1						
P10	CAN_1_TX	PDM_CLK	ALT6	can1.TX	O	CAN port 1 transmit pin
P12	CAN_1_RX	PDM_BIT_STREAM0	ALT6	can1.RX	I	CAN port 1 receive pin
CAN interface 2						
S28	CAN_2_TX	GPIO_IO25	ALT2	can2.TX	O	CAN port 2 transmit pin
S30	CAN_2_RX	GPIO_IO27	ALT2	can2.RX	I	CAN port 2 receive pin

5.3 Digital Audio

5.3.1 I²S – Inter-IC Sound

The Lino iMX91 system-on-module provides digital audio connectivity through the **I²S (Inter-IC Sound)** interface integrated in the NXP i.MX 91 system-on-chip. This interface enables high-quality serial audio communication with external audio codecs, DACs, ADCs, and other digital audio devices.

Key Features and Supported Capabilities

- **Interface standard:** I²S-compatible serial audio interface (`I2S_1`)
- **Audio resolution:** supports audio sample widths **up to 32 bits**, depending on configuration and connected device
- **Sample rates:** supports common audio sample rates **from 8 kHz up to 192 kHz**, subject to clock configuration
- **Operation modes:** master or slave operation, **full-duplex audio streaming**
- **System integration:** DMA support for audio data transfer

Table 19 lists the `I2S_1` interface signals exposed on the Lino iMX91 board-to-board connector `X2`, including their descriptions and usage notes.

Table 19: I²S interface

X2 pin	Lino signal name	SoC ball name	SoC alternate function	SoC alternate function name	I/O	Description/Remarks
S94	I2S_1_BITCLK	GPIO_IO16	ALT1	sai3.TX_BCLK	O[1]/I[2]	Transmit bit clock receive for peripheral/slave mode
S96	I2S_1_SYNC	GPIO_IO26	ALT7	sai3.TX_SYNC	O[1]/I[2]	Transmit frame sync receive for peripheral/slave mode
S98	I2S_1_DATA_OUT	GPIO_IO19	ALT7	sai3.TX_DATA[0]	O	Data output
S100	I2S_1_DATA_IN	GPIO_IO20	ALT1	sai3.RX_DATA[0]	I	Data input

[1] Controller/master mode.
[2] Peripheral/slave mode.

5.4 Ethernet

The Lino iMX91 system-on-module integrates Ethernet connectivity based on the **Ethernet Media Access Controller (MAC)** provided by the NXP i.MX 91 system-on-chip. The Ethernet subsystem is designed to support reliable, high-performance wired networking.

The i.MX 91 Ethernet MAC supports **10/100/1000 Mbps Ethernet operation** and is compliant with the **IEEE 802.3** standard. It interfaces with an external Ethernet PHY through a standard **RGMII interface**, enabling reduced pin count and simplified board routing. The MAC includes support for full and half-duplex operation, automatic CRC generation and checking, frame filtering, and programmable MAC addressing. Hardware-assisted checksum offloading is provided to reduce CPU load during packet processing.

The Ethernet controller supports **interrupt and DMA-based** data transfer, allowing efficient movement of frame data between system memory and the MAC. This enables high throughput while minimizing processor intervention. The interface is suitable for use with real-time operating systems and Linux-based environments supported on the Lino iMX91 platform. Overall, the Ethernet implementation on the Lino iMX91 SoM provides a robust and standards-compliant solution for wired network connectivity.

Key features:

- **ENET Audio Video Bridging (AVB)** support for time-sensitive audio and video data streams
- **IEEE 1588 Precision Time Protocol (PTP)** support for accurate time synchronization across the network
- **Energy Efficient Ethernet (EEE)** support to reduce power consumption during periods of low network activity
- **Flexible I/O voltage operation**, supporting **1.8 V and 3.3 V RMII**, and **1.8 V RGMII** interfaces

5.4.1 Ethernet interface 1

The Ethernet interface 1, called `ENET1` and described in Table 20, is an **Ethernet controller with Quality of Service (QoS) and Time-Sensitive Networking (TSN) support**. It is designed for deterministic and real-time Ethernet applications and supports advanced features such as traffic shaping, frame preemption, time-aware scheduling, and precise time synchronization (**IEEE 802.1AS**). These capabilities make `ENET1` suitable for industrial networking, real-time control, and applications that require guaranteed latency and bandwidth.

Table 20: Ethernet interface 1

X1 pin	Lino signal name	SoC ball name	I/O	Description/Remarks
Control Signals				
P67	ETH_MDIO	ENET1_MDIO	I/O	Management data input/output[1]
P68	ETH_MDC	ENET1_MDC	O	Management data clock
P73	ETH_1_RGMII_RXC	ENET1_RXC	I	Receive clock provided by the PHY in RGMII mode
P75	ETH_1_RGMII_RX_CTL	ENET1_RX_CTL	I	Receive control[2]
P89	ETH_1_RGMII_TXC	ENET1_TXC	O	Transmit clock driven by the MAC in RGMII mode
P85	ETH_1_RGMII_TX_CTL	ENET1_TX_CTL	O	Transmit control[3]
Data signals				
P77	ETH_1_RGMII_RXD_0	ENET1_RX0	I	Receive Data [3:0] four-bit data bus carrying received Ethernet frame data from the PHY to the MAC
P79	ETH_1_RGMII_RXD_1	ENET1_RX1	I	
P81	ETH_1_RGMII_RXD_2	ENET1_RX2	I	
P83	ETH_1_RGMII_RXD_3	ENET1_RX3	I	
P97	ETH_1_RGMII_TXD_0	ENET1_TX0	O	Transmit Data [3:0] four-bit data bus carrying transmit Ethernet frame data from the MAC to the PHY
P95	ETH_1_RGMII_TXD_1	ENET1_TX1	O	
P93	ETH_1_RGMII_TXD_2	ENET1_TX2	O	
P91	ETH_1_RGMII_TXD_3	ENET1_TX3	O	

[1] Bidirectional serial data line used to configure and monitor the Ethernet PHY via the MDIO management interface.
[2] Encodes RX_DV (receive data valid) and RX_ER (receive error) information for the receive path.
[3] Encodes TX_EN (transmit enable) and TX_ER (transmit error) information for the transmit path.

5.4.2 Ethernet interface 2

The Ethernet interface 2, called ENET2 and described in Table 21, in contrast, is a **standard Ethernet Media Access Controller (MAC) without TSN capabilities**. It provides conventional Ethernet functionality for general-purpose networking applications, offering reliable 10/100/1000 Mbps communication with lower complexity. ENET2 is typically used for non-real-time data traffic, such as standard network connectivity, diagnostics, or management interfaces.

Table 21: Ethernet interface 2

X2 pin	Lino signal name	SoC ball name	I/O	Description/Remarks
Control Signals				
P67[1]	ETH_MDIO	ENET1_MDIO	I/O	Management data input/output[2] shared with Ethernet 1
P68[1]	ETH_MDC	ENET1_MDC	O	Management data clock shared with Ethernet 1
S71	ETH_2_RGMII_RXC	ENET2_RXC	I	Receive clock provided by the PHY in RGMII mode
S73	ETH_2_RGMII_RX_CTL	ENET2_RX_CTL	I	Receive control[3]
S87	ETH_2_RGMII_TXC	ENET2_TXC	O	Transmit clock driven by the MAC in RGMII mode
S89	ETH_2_RGMII_TX_CTL	ENET2_TX_CTL	O	Transmit control[4]

Continued on next page

Table 21: Ethernet interface 2 (Continued)

X2 pin	Lino signal name	SoC ball name	I/O	Description/Remarks
Data signals				
S77	ETH_2_RGMII_RXD_0	ENET2_RX0	I	Receive Data [3:0] Four-bit data bus carrying received Ethernet frame data from the PHY to the MAC
S79	ETH_2_RGMII_RXD_1	ENET2_RX1	I	
S81	ETH_2_RGMII_RXD_2	ENET2_RX2	I	
S83	ETH_2_RGMII_RXD_3	ENET2_RX3	I	
S99	ETH_2_RGMII_TXD_0	ENET2_TX0	O	Transmit Data [3:0] Four-bit data bus carrying transmit Ethernet frame data from the MAC to the PHY
S97	ETH_2_RGMII_TXD_1	ENET2_TX1	O	
S93	ETH_2_RGMII_TXD_2	ENET2_TX2	O	
S91	ETH_2_RGMII_TXD_3	ENET2_TX3	O	

[1] These control pins are exposed on X1 connector, as they are shared between both ethernet interfaces.
[2] Bidirectional serial data line used to configure and monitor the Ethernet PHY via the MDIO management interface.
[3] Encodes RX_DV (receive data valid) and RX_ER (receive error) information for the receive path.
[4] Encodes TX_EN (transmit enable) and TX_ER (transmit error) information for the transmit path.

Use Cases
- `ENET1` is intended for **real-time, time-critical** applications, leveraging its TSN capabilities
- `ENET2` provides a **general-purpose** Ethernet MAC interface for standard networking applications

5.5 GPIO – General-Purpose Input/Output

The Lino iMX91 integrates a flexible and low-latency **General-Purpose Input/Output (GPIO)** subsystem designed to support a wide range of control, monitoring, and event-driven applications. The module features **ten dedicated GPIO pins**, each of which can be independently configured for digital input, digital output, interrupt generation, or DMA request functionality.

When configured for GPIO operation, pin behavior is controlled through memory-mapped registers that independently manage pin direction and data. The **Port Data Direction Registers (PDDR)** define whether each pin operates as an input or an output. For pins configured as outputs, the **Port Data Output Registers (PDOR)** determine the driven logic level, while the **Port Data Input Registers (PDIR)** always reflect the real-time logic level observed at the pin, regardless of direction. Dedicated set, clear, and toggle registers enable atomic bit manipulation, allowing multiple pins within a port to be updated with a single write operation and eliminating the need for read-modify-write sequences.

Beyond basic I/O functionality, the Lino iMX91 GPIO implementation includes **interrupt and DMA support**. In **all digital pin multiplexing modes**, each GPIO pin supports independent configuration for external interrupt detection based on defined signal conditions, as shown in Table 22. Interrupts support rising, falling, or both edges, as well as high or low-level sensitivity, and can be used to asynchronously wake the system from Low-Power modes.

GPO in Debug Mode
GPIO functionality remains **fully operational in Debug mode**.

Table 22: Available pin configurations for external interrupts

Signal conditions	Software polling using flags	Interrupts	DMA requests
Rising-edge	Yes	Yes	Yes
Falling-edge	Yes	Yes	Yes
Rising and falling edge	–	Yes	–
High-level	–	Yes	–
Low-level	–	Yes	–

When the configured signal condition is detected on a pin, the corresponding Interrupt Status Flag (ISF) is set. In normal operating modes, the pin input is synchronized to the system clock prior to evaluation of the selected edge or level to ensure reliable detection.

An interrupt request is generated when any enabled **Interrupt Status Flag (ISF)** is set and remains asserted until all enabled interrupt status flags are cleared by software by writing a logic `1` to the corresponding ISF clear register. A **DMA request** is generated when any enabled DMA-related ISF is set and is automatically deasserted upon completion of the associated DMA transfer, which clears the corresponding status flags.

In Low-Power mode, enabled interrupt conditions are detected asynchronously. Detection of the configured edge or level sets the ISF and generates an asynchronous wake-up signal, allowing the system to exit Low-Power mode.

The GPIO subsystem also provides robust protection and access control features, allowing each pin, interrupt, and DMA request domain to be configured for secure or nonsecure, and privileged or nonprivileged access. A single clock and reset source are used for register access and synchronization with external pin inputs, with no special clocking or reset considerations required.

Table 23 shows the Lino iMX91 GPIO pin mapping and corresponding SoC signal assignments.

Table 23: GPIO pin mapping

X1 pin	Lino specification name	SoC ball name	SoC alternate function	SoC alternate function name	Reset status	Description/Remarks
P30	GPIO_1	GPIO_IO00	ALT0	gpio2.IO[0]	Input with Pull-down	General-Purpose Input/Output
P32	GPIO_2	GPIO_IO01	ALT0	gpio2.IO[1]	Input with Pull-down	General-Purpose Input/Output
P34	GPIO_3	GPIO_IO17	ALT0	gpio2.IO[17]	Input with Pull-down	General-Purpose Input/Output
P36	GPIO_4	GPIO_IO22	ALT0	gpio2.IO[22]	Input with Pull-down	General-Purpose Input/Output
P38	GPIO_5	GPIO_IO24	ALT0	gpio2.IO[24]	Input with Pull-down	General-Purpose Input/Output
P40	GPIO_6	CCM_CLK01	ALT5	gpio3.IO[26]	Output low	General-Purpose Input/Output
P42	GPIO_7	CCM_CLK02	ALT5	gpio3.IO[27]	Output low	General-Purpose Input/Output
P44	GPIO_8	CCM_CLK03	ALT5	gpio4.IO[28]	Input with Pull-down	General-Purpose Input/Output
P46	GPIO_9	ENET2_MDC	ALT5	gpio4.IO[14]	Input with Pull-down	General-Purpose Input/Output
P48	GPIO_10	ENET2_MDIO	ALT5	gpio4.IO[15]	Input with Pull-down	General-Purpose Input/Output

Reset State
As soon as the boot loader is running, it is possible to reconfigure the pins and their states.

5.5.1 System GPIO Expander

The Lino iMX91 system-on-module integrates the **PCAL6408AHKX GPIO expander** to provide additional general-purpose input/output capability beyond the native GPIOs of the i.MX 91 SoC. The PCAL6408AHKX is an **8-bit I²C-controlled** GPIO expander that enables flexible control and monitoring of board-level signals while minimizing SoC pin usage.

On the Lino iMX91, the GPIO expander is used to manage several system control and status signals, including **USB enable and overcurrent indications**, **PMIC interrupt signaling**, and auxiliary control lines. The device operates from a **1.8 V supply** and communicates with the SoC via the I²C bus, providing programmable direction control, input polarity inversion, and interrupt generation for each GPIO pin.

Table 24 summarizes the mapping between the PCAL6408AHKX GPIO pins and the corresponding system signals as implemented on the Lino iMX91 SoM. This table provides the reference needed to configure and use the GPIO expander within software and to understand its role in system-level signal management.

Table 24: System GPIO expander

X1/X2 pin[1]	PCAL6408AHKX pin	Lino specification function name	Description/Remarks
P27	P0	USB_1_EN	Enable USB_1 power switch
P61	P1	USB_2_EN	Enable USB_2 power switch
–	P2	TPM_SPI_CS#	Chip select signal for the TPM device active low
P29	P3	USB_1_OC#	Overcurrent indication for USB1 active low
P63	P4	USB_2_OC#	Overcurrent indication for USB2 active low
–	P5	PMIC_INT#	Interrupt signal from the PMIC active low with 10kΩ pull-up
–	P6	TPM_IRQ#	Request from the TPM device active low
S66	P7	MSP7	General-Purpose Input/Output

[1] The X1 pins start with the letter P while the pins on X2 start with the letter S. Refer to Section 3.2 for details on the naming convention for the main connector pins.

5.6 I²C – Inter-Integrated Circuit

The Lino iMX91 system-on-module provides multiple **I²C (Inter-Integrated Circuit)** interfaces through the controllers integrated in the NXP i.MX 91 system-on-chip. The I²C subsystem enables low-speed, bidirectional serial communication with a wide range of peripheral devices.

The I²C controllers on the i.MX 91 support **multi-controller and target operation** and are compliant with the I²C-bus specification. Each interface uses the standard two-wire bus architecture, consisting of a serial clock line (SCL) and a bidirectional serial data line (SDA), with open-drain signaling and external pull-up resistors provided on the module or carrier board as required.

5.6.1 I²C Interfaces

Table 25 lists the I²C interfaces on the Lino iMX91, including their pin assignments, function multiplexing, and functional descriptions.

Table 25: I²C interfaces

X1 pin	Lino specification signal name	SoC ball name	SoC alternate function	SoC alternate function name	Description/Remarks
I²C 1					
P18	I2C_1_SCL	I2C1_SCL	ALT0	i2c1.SCL	Generic I²C *Always Compatible*
P16	I2C_1_SDA	I2C1_SDA	ALT0	i2c1.SDA	
I²C 2					
P22	I2C_2_SCL	GPIO_IO29	ALT1	i2c3.SCL	Generic I²C *Always Compatible*
P20	I2C_2_SDA	GPIO_IO28	ALT1	i2c3.SDA	
I²C 3					
P26	I2C_3_SCL	GPIO_IO03	ALT1	i2c4.SCL	Generic I²C *Always Compatible*
P24	I2C_3_SDA	GPIO_IO02	ALT1	i2c4.SDA	
PMIC I²C[1]					
–	PMIC_I2C_SCL	I2C2_SCL	ALT0	i2c2.SCL	Dedicated I²C interfac for the PMIC and on-board components
–	PMIC_I2C_SDA	I2C2_SCL	ALT0	i2c2.SDA	

[1] Interface not exposed on the board-to-board connectors.

5.6.2 Connected Devices and Addresses

Table 26 lists the on-board devices connected to the I²C bus of the Lino iMX91 system-on-module, including the corresponding part numbers, associated I²C interface, and assigned 7-bit slave addresses.

Table 26: On-Board I²C devices

Device	Part number	I²C interface	Address (7-bit)
EEPROM	`M24C02-FMC6TG`	PMIC_I2C	`0x50`
GPIO Expander	`PCAL6408AHKX`	PMIC_I2C	`0x21`
Power Management IC (PMIC)	`MPF9453AVMA1HN`	PMIC_I2C	`0x32`
Temperature Sensor	`TMP1075`	PMIC_I2C	`0x48`

EEPROM

The Lino iMX91 system-on-module integrates the `M24C02-FMC6TG` EEPROM, a **2 kB (256 x 8-bit)** electrically erasable programmable memory device. The EEPROM is connected via a standard I²C interface, providing non-volatile storage for system configuration data, calibration parameters, and other persistent information.

The device is qualified for industrial temperature operation **from -40 °C to +85 °C**.

Key electrical characteristics of the EEPROM include:

- **Supply voltage:** operated at **3.3 V**, within the admissible range of 1.7 V to 5.5 V
- **Active current consumption:** typically around **4 mA**
- **Standby current:** below **1 µA**, enabling low-power operation

The `M24C02-FMC6TG` provides high endurance and long data retention, ensuring reliable long-term storage:

- **Write endurance:** up to **4,000,000 write cycles**
- **Data retention:** greater than **200 years**

System GPIO Expander

Refer to Section 5.5.1 for more information about the on-board GPIO expander.

PMIC

Refer to Section 11.1.1 for more information about the PMIC usage and characteristics.

Temperature Sensor

The module integrates a TMP1075 digital temperature sensor, providing accurate local temperature measurements for system monitoring and thermal management. The device communicates with the host processor via a standard I^2C-compatible interface. A programmable alert output allows implementation of temperature threshold monitoring and over-temperature protection.

Key characteristics of the TMP1075 include:

- **Temperature measurement range:** from **-55 °C to +125 °C**
- **Accuracy:** up to $\pm$**0.5 °C** (typical over the normal operating range)
- **Resolution:** up to **12-bit**, configurable
- **Programmable thresholds:** high and low temperature limits with alert functionality

5.7 JTAG – Joint Test Action Group

The Lino iMX91 system-on-module integrates a **JTAGC (Joint Test Action Group Controller)** that provides standardized access for device test, debug, and boundary-scan operations. The JTAG interface is fully compliant with the **IEEE 1149.1-2001** standard and enables reliable control and observation of internal device states during manufacturing, board-level testing, and system debugging.

The JTAGC implements a standard **four-pin Test Access Port (TAP)** interface consisting of TDI, TMS, TCK, and TDO. It includes a dedicated instruction register supporting both IEEE 1149.1 defined instructions and additional public and private device-specific instructions. Core JTAG data registers, including the bypass register, boundary-scan register, and device identification register, are provided to support efficient test access and device identification. Operation of these registers and associated circuitry is managed by an IEEE-compliant TAP controller state machine, ensuring predictable and standards-based JTAG behavior across all supported use cases.

Table 27 lists the JTAG interface signals exposed on the Lino iMX91 board-to-board connector.

Table 27: JTAG interface

X1 pin	Lino signal name	SoC ball name	SoC alternate function	SoC alternate function name	I/O	Description/Remarks
S59	JTAG_1_TDI (MSP_21)	DAP_TDI	ALT0	dap.TDI	I	Test data in
S61	JTAG_1_TDO (MSP_22)	DAP_TDO_TRACESWO	ALT0	dap.TDO_TRACESWO	O	Test data out
S67	JTAG_1_TCK (MSP_25)	DAP_TCLK_SWCLK	ALT0	dap.TCLK_SWCLK	I/O	Test Clock
S65	JTAG_1_TMS (MSP_24)	DAP_TMS_SWDIO	ALT0	dap.TMS_SWDIO	I[1]	Test Mode Select

[1] Input in JTAG mode; bidirectional in SWD mode.

5.8 LCDIF – LCD Interface

The Lino iMX91 exposes the **LCD Interface (LCDIF)** of the NXP i.MX 91 SoC, providing a parallel display interface for connecting external RGB displays. The LCDIF supports a configurable pixel bus width and timing signals required for common TFT panels, enabling direct connection to display controllers without additional bridge devices.

LCDIF Pin Multiplexing and Availability
The LCDIF signals are routed through GPIO pads and are multiplexed with other SoC functions. To enable the LCDIF interface, the corresponding GPIO pins must be configured for the LCDIF alternate function (ALT3), as defined in the NXP i.MX 91 Reference Manual. All LCDIF signals operate as **outputs from the SoC**.
The availability of individual LCDIF signals and the usable pixel data bus width (e.g. RGB565, RGB666, RGB888) depend on the selected pin-mux configuration and overall system design. Designers must ensure that LCDIF pin assignments do not conflict with other required interfaces sharing the same pins.

Table 28 lists the LCDIF interface signals exposed on the Lino iMX91 board-to-board connector.

Table 28: LCDIF interface description

X1/X2 pin	Lino signal name	SoC ball name	SoC alternate function	SoC alternate function name	I/O	Description/Remarks
Control Signals						
P30	GPIO_1	GPIO_IO00	ALT3	lcdif.PCLK	O	Pixel clock generated by the SoC drives data sampling on the display interface
P32	GPIO_2	GPIO_IO01	ALT3	lcdif.DE	O	Horizontal synchronization signal indicates the start of a display line
P24	I2C_3_SDA	GPIO_IO02	ALT3	lcdif.VSYNC	O	Vertical synchronization signal indicates the start of a display frame
P26	I2C_3_SCL	GPIO_IO03	ALT3	lcdif.HSYNC	O	Data enable signal indicates valid pixel data on the data bus
Data Signals						
P11	UART_1_TX	GPIO_IO04	ALT3	lcdif.D[0]	O	Pixel data bit 0 least significant bit
P9	UART_1_RX	GPIO_IO05	ALT3	lcdif.D[1]	O	Pixel data bit 1
P8	UART_1_CTS	GPIO_IO06	ALT3	lcdif.D[2]	O	Pixel data bit 2
P6	UART_1_RTS	GPIO_IO07	ALT3	lcdif.D[3]	O	Pixel data bit 3
P15	UART_2_TX	GPIO_IO08	ALT3	lcdif.D[4]	O	Pixel data bit 4
P13	UART_2_RX	GPIO_IO09	ALT3	lcdif.D[5]	O	Pixel data bit 5
S7	UART_2_CTS	GPIO_IO10	ALT3	lcdif.D[6]	O	Pixel data bit 6
S5	UART_2_RTS	GPIO_IO11	ALT3	lcdif.D[7]	O	Pixel data bit 7
P58	SPI_1_CS	GPIO_IO12	ALT3	lcdif.D[8]	O	Pixel data bit 8
P54	SPI_1_MISO	GPIO_IO13	ALT3	lcdif.D[9]	O	Pixel data bit 9
P56	SPI_1_MOSI	GPIO_IO14	ALT3	lcdif.D[10]	O	Pixel data bit 10
P52	SPI_1_CLK	GPIO_IO15	ALT3	lcdif.D[11]	O	Pixel data bit 11
S94	I2S_1_BCLK	GPIO_IO16	ALT3	lcdif.D[12]	O	Pixel data bit 12
P34	GPIO_3	GPIO_IO17	ALT3	lcdif.D[13]	O	Pixel data bit 13
P3	PWM_2	GPIO_IO18	ALT3	lcdif.D[14]	O	Pixel data bit 14

Continued on next page

Table 28: LCDIF interface description (Continued)

X1/X2 pin	Lino signal name	SoC ball name	SoC alternate function	SoC alternate function name	I/O	Description/Remarks
S98	I2S_1_D_OUT	GPIO_IO19	ALT3	lcdif.D[15]	O	Pixel data bit 15
S100	I2S_1_D_IN	GPIO_IO20	ALT3	lcdif.D[16]	O	Pixel data bit 16
P1	PWM_1	GPIO_IO21	ALT3	lcdif.D[17]	O	Pixel data bit 17
P36	GPIO_4	GPIO_IO22	ALT3	lcdif.D[18]	O	Pixel data bit 18
P5	PWM_3	GPIO_IO23	ALT3	lcdif.D[19]	O	Pixel data bit 19
P38	GPIO_5	GPIO_IO24	ALT3	lcdif.D[20]	O	Pixel data bit 20
S28	CAN_2_TX	GPIO_IO25	ALT3	lcdif.D[21]	O	Pixel data bit 21
S96	I2S_1_SYNC	GPIO_IO26	ALT3	lcdif.D[22]	O	Pixel data bit 22
S30	CAN_2_RX	GPIO_IO27	ALT3	lcdif.D[23]	O	Pixel data bit 23 most significant bit

5.9 PWM – Pulse-Width Modulation

The Lino iMX91 system-on-module provides **three independent pulse-width modulation (PWM) channels** implemented using the **Timer/PWM Modules (TPM)** integrated in the NXP i.MX 91 system-on-chip. These PWM channels are intended for generating accurate and programmable timing waveforms for control and signal modulation applications. Each PWM module is based on a counter/compare architecture, allowing independent configuration of the PWM period and duty cycle on a per-channel basis.

Naming Convention
To prevent confusion with other acronyms, this datasheet uses the abbreviation TPM exclusively to denote the Trusted Platform Module. References to timer or PWM signals using TPM follow only the SoC manufacturer's original pin names or alternate functions. All other instances of TPM pertain strictly to the Trusted Platform Module.

Each PWM output can be individually routed to external pins through the SoC's I/O multiplexing system and is driven by a **selectable clock source** with a programmable prescaler. The PWM channels support edge-aligned PWM operation, as well as interrupt generation and DMA requests, and can be configured to continue operating in low-power and debug modes, subject to clock configuration.

Key Features and Supported Capabilities

- **PWM controllers:** 3 *always compatible* independent PWM channels
- **Resolution:** up to **16-bit** counter resolution duty-cycle control
- **Timing and frequency:** programmable PWM period and duty cycle[3]
- **Operating modes:**
  - Configurable output polarity
  - Continuous PWM generation
  - Edge-aligned PWM operation
- **System integration:** selectable clock sources and prescaling, interrupt generation and DMA requests

5.9.1 PWM Interfaces

Table 29 lists the PWM interface signals exposed on the Lino iMX91 X1 board-to-board connector, including their pin assignments, signal direction, and functional descriptions.

[3] Output frequency configurable depending on the selected clock source and prescaler.

Table 29: PWM interface

X1 pin	Lino signal name	SoC ball name	Soc alternate function	SoC alternate function name	I/O	Description/Remarks
P1	PWM_1	GPIO_IO21	ALT6	tpm4.CH1	O	General Purpose PWM
P3	PWM_2	GPIO_IO18	ALT6	tpm5.CH2	O	General Purpose PWM
P5	PWM_3	GPIO_IO23	ALT4	tpm6.CH1	O	General Purpose PWM

5.10 SPI – Serial Peripheral Interface

The Lino iMX91 system-on-module provides high-speed serial peripheral connectivity through the **SPI (Serial Peripheral Interface)** controllers integrated in the NXP i.MX 91 system-on-chip. The SPI subsystem enables **synchronous, full-duplex communication**. The i.MX 91 integrates **LPSPI (Low-Power SPI) controllers** that support flexible bus configurations, including multiple chip-select signals, configurable clock polarity and phase, and master or slave operation.

Each SPI interface consists of:

- a serial clock (SCK),
- master-out/slave-in (`MOSI`),
- master-in/slave-out (`MISO`), and
- one or more chip-select (CS) signals.

The controllers are designed to support both simple point-to-point connections and shared SPI buses with multiple slave devices.

Key Features and Supported Capabilities

- **Interface type:** LPSPI (Low-Power SPI) controllers
- **Operating modes:** full-duplex synchronous communication, master and slave operation
- **Clocking:** programmable SPI clock frequency, up to several tens of MHz
- **Data formats:** programmable data frame sizes (typically 8 to 32 bits), configurable clock polarity (CPOL) and phase (CPHA)
- **Chip-select support:** multiple hardware-controlled chip-select outputs per SPI controller
- **System integration:** interrupt and DMA-driven data transfers

5.10.1 SPI Interface

Tables 30 and 31 list the SPI interfaces signals exposed on the board-to-board connectors of the Lino iMX91, including pin assignments, signal direction, and functional descriptions.

Convention for Signal Direction
The signal directions (I/O) listed in the table are defined from the perspective of the **i.MX 91 SoC operating as the SPI master**.

Table 30: SPI *Always Compatible* interface

X1 pin	Lino signal name	SoC ball name	SoC alternate function	SoC alternate function name	I/O	Description/Remarks
P58	SPI_1_CS	GPIO_IO12	ALT4	spi8.PCS0	O	Peripheral select
P54	SPI_1_MISO	GPIO_IO13	ALT4	spi8.SIN	I	Controller input, peripheral output
P56	SPI_1_MOSI	GPIO_IO14	ALT4	spi8.SOUT	O	Controller output, peripheral input
P52	SPI_1_CLK	GPIO_IO15	ALT4	spi8.SCK	O	Serial clock

Table 31: SPI *Reserved* interface

X2 pin	Lino signal name	SoC ball name	SoC alternate function	SoC alternate function name	I/O	Description
S18	SPI_2_CS	SAI1_TXFS	ALT2	spi1.PCS0	O	Peripheral select Boot mode 2 with 10kΩ pull-down
S24	SPI_2_MISO	SAI1_TXC	ALT2	spi1.SIN	I	Controller input, peripheral output
S22	SPI_2_MOSI	SAI1_RXD0	ALT2	spi1.SOUT	O	Controller output, peripheral input
S20	SPI_2_CLK	SAI1_TXD0	ALT2	spi1.SCK	O	Serial clock Boot mode 3 with 10kΩ pull-down

5.11 Storage Interfaces

The Lino iMX91 system-on-module provides non-volatile and removable storage support through the **uSDHC (Ultra Secured Digital Host Controller) modules** integrated in the NXP i.MX 91 system-on-chip. These controllers enable high-performance, standards-compliant connectivity to eMMC, SD, and SDIO devices, supporting both boot and mass-storage applications.

For on-module non-volatile storage, the Lino iMX91 uses the `uSDHC1` instance configured for **eMMC operation with an 8-bit data bus**, including dedicated clock (`CLK`), command (`CMD`), data (`DAT[7:0]`), and data strobe (`DS`) signals. The data strobe enables **HS400 mode**, allowing very high throughput and making the eMMC interface suitable as the primary boot device and root filesystem storage.

In addition, the Lino iMX91 exposes a second uSDHC-based interface for **SD and SDIO cards**, supporting removable media and peripheral expansion. This interface operates with a **1-bit or 4-bit data bus** and includes clock, command, data, card-detect, and power-enable signals. Voltage selection for the SD interface is performed by the SoC through control of the PMIC, enabling **operation at 1.8 V or 3.3 V** in accordance with SD specifications.

The uSDHC modules of the i.MX 91 provide the following key features:

- **Standards compliance and compatibility** with SD Host Controller v2.0/3.0, eMMC v4.2–v5.1, SD/SDXC v3.0, and SDIO v2.0/3.0 specifications
- **Flexible bus configurations**, supporting 1/4-bit SD and SDIO modes and 1/4/8-bit eMMC modes
- **High-performance data transfer**, up to **400 Mbps for SD/SDIO** and up to **3200 Mbps for eMMC** using SDR and DDR modes
- **Advanced transfer and control features**, including multi-block operations, command queuing, Auto CMD12, SDIO interrupts, and pause/resume support
- **Integrated DMA and voltage selection support**, including ADMA for efficient data movement and configurable I/O voltage operation

Table 32: uSDHC interfaces overview

SDIO interface (SoC)	Max bus width	Description/Remarks
USDHC1	8-bit	Connected to the internal eMMC boot device not available at the module board-to-board connector
USDHC2	4-bit	Always Compatible SD interface
USDHC3	4-bit	Used for non-removable peripherals e.g., AI accelerators, wireless modules

The SD interface on the module operates in the *Always Compatible* class and supports both I/O voltage levels. Care must be taken to ensure correct I/O voltage selection when interfacing with SD memory cards. While standard SD operation uses 3.3 V signaling, UHS-I operation may switch the interface to 1.8 V as part of the card initialization sequence.

External pull-up resistors on the carrier board are not required for the SDIO signals (`CMD`, `DATA[3:0]`, and `CLK`), as the necessary pull-ups are either integrated within the SoC or provided on the module PCB.

5.11.1 eMMC Interface (uSDHC1)

The Lino iMX91 system-on-module integrates the SKYHIGH `S40FC016C3B1I00300` **eMMC (embedded MultiMediaCard)** device as the primary non-volatile storage solution. This eMMC device provides reliable, high-performance flash storage suitable for bootloader, operating system, and application data. The device is connected to the NXP i.MX 91 SoC through the native `uSDHC1` interface and operates on an **8-bit data bus**, enabling high-throughput data transfers and efficient system boot.

The eMMC interface makes use of dedicated clock (`CLK`), command (`CMD`), data (`DAT[7:0]`), and data strobe (`DS`) signals. The presence of the data strobe signal allows support for advanced high-speed modes, such as **HS400**, in accordance with the eMMC specification supported by the i.MX 91.

Table 33 lists the i.MX 91 SoC balls used for the eMMC interface on the Lino iMX91 SoM, detailing the mapping between SoC pins and the corresponding eMMC signals as implemented in the hardware design.

Table 33: eMMC interface

Lino signal name	SoC ball name	I/O	Description/Remarks
Control Signals			
eMMC_CLK	SD1_CLK	O	Clock output
eMMC_CMD	SD1_CMD	I/O	SDIO command line
eMMC_STROBE	SD1_STROBE	I	Data strobe signal
Data Signals			
eMMC_DATA0	SD1_DATA0	I/O	Data line 0
eMMC_DATA1	SD1_DATA1	I/O	Data line 1
eMMC_DATA2	SD1_DATA2	I/O	Data line 2
eMMC_DATA3	SD1_DATA3	I/O	Data line 3
eMMC_DATA4	SD1_DATA4	I/O	Data line 4
eMMC_DATA5	SD1_DATA5	I/O	Data line 5
eMMC_DATA6	SD1_DATA6	I/O	Data line 6
eMMC_DATA7	SD1_DATA7	I/O	Data line 7

X1 and X2 connectors: the eMMC interface is **not available** at the module board-to-board connectors.

5.11.2 SD Card Interface (uSDHC2)

On the Lino iMX91, the **SD card interface is implemented using uSDHC2**. The interface supports **1-bit and 4-bit SD bus** configurations and includes the required clock (CLK), command (CMD), and data (DAT[3:0]) signals, along with card-detect and power-control signals. This interface is intended for removable storage devices.

The uSDHC controller conforms to the **SD Host Controller Standard Specification v2.0 and v3.0** and is compatible with the **SD Memory Card Specification v3.0**, including

- **Standard Capacity (SDSC)**,
- **High Capacity (SDHC)**, and
- **Extended Capacity (SDXC)** cards.

It also supports **UHS-I modes**, enabling higher bus speeds when operating **at 1.8 V I/O signaling**, while maintaining **backward compatibility with 3.3 V signaling** for standard SD operation. The controller supports dynamic I/O voltage switching during card initialization, as defined by the SD standard.

Data transfers are supported in both **single-block and multi-block modes**, with configurable block sizes and integrated **DMA/ADMA support** to minimize CPU load and improve throughput. The necessary pull-up resistors and voltage-level management are provided on the module, eliminating the need for external pull-ups on the carrier board. Overall, the SD card interface on the Lino iMX91 offers a flexible, standards-compliant solution for removable storage and SDIO-based expansion.

Table 34 lists the i.MX 91 SoC balls used for the SD card interface on the Lino iMX91 SoM, detailing the mapping between SoC pins and the corresponding SD card signals as implemented in the hardware design.

Table 34: SD Card interface

X1 pin	Lino signal name	SoC ball name	I/O	Description/Remarks
Control Signals				
P68	SD_1_CLK	SD2_CLK	O	SD card clock
P66	SD_1_CMD	SD2_CMD	I/O	SD command line
P78	SD_1_CD#	SD2_CD_B	I	SD card detect (active low) 10kΩ pull-up to 1.8 V or 3.3 V depending on operating voltage
P72	SD_1_PWR_EN	SD2_RESET_B	O	SD card power enable/reset signal
Data Signals				
P74	SD_1_D0	SD2_DATA0	I/O	SD data line 0
P76	SD_1_D1	SD2_DATA1	I/O	SD data line 1
P60	SD_1_D2	SD2_DATA2	I/O	SD data line 2
P64	SD_1_D3	SD2_DATA3	I/O	SD data line 3

5.11.3 SDIO Interface (uSDHC3)

On the Lino iMX91, an additional SDIO interface is implemented using the uSDHC3 controller. This interface supports **1-bit and 4-bit SDIO bus configurations** and provides the required clock (CLK), command (CMD), and data (DAT[3:0]) signals. Unlike the SD card interface implemented with uSDHC2, this interface does not include card-detect or power-switching signals and is therefore **intended for non-removable SDIO peripherals**.

The uSDHC3 controller conforms to the **SD Host Controller Standard Specification v2.0 and v3.0** and

is compatible with the **SDIO Card Specification**, enabling connection of SDIO-based devices such as AI accelerators, wireless modules, or other peripherals. The interface operates at a **fixed I/O voltage**, as defined by the module hardware configuration, and does not support dynamic I/O voltage switching.

Data transfers on the uSDHC3 interface support **single-block and multi-block modes**, with configurable block sizes and **DMA/ADMA support** to reduce CPU load and improve data throughput.

Table 35 list the i.MX 91 SoC balls used for the uSDHC3 interface on the Lino iMX91 SoM, detailing the mapping between SoC signals and the corresponding SDIO interface connections as implemented in the hardware design.

Table 35: SDIO interface

X2 pin	Lino Signal Name	SoC Ball Name	I/O	Description/Remarks
Control Signals				
S8	SD_2_CMD	SD3_CMD	I/O	SD command line
S10	SD_2_CLK	SD3_CLK	O	SD card clock
Data Signals				
S12	SD_2_D0	SD3_DATA0	I/O	SD data line 0
S16	SD_2_D1	SD3_DATA1	I/O	SD data line 1
S2	SD_2_D2	SD3_DATA2	I/O	SD data line 2
S4	SD_2_D3	SD3_DATA3	I/O	SD data line 3

5.12 Tamper Detection

The Lino iMX91 system-on-module provides support for **tamper detection** through the tamper pins integrated into the **NXP EdgeLock® Security Subsystem** of the i.MX 91 system-on-chip. These pins are part of the SoC's secure subsystem and are designed to detect physical intrusion or unauthorized access attempts, making them suitable for security-sensitive and anti-tamper applications.

The tamper pins are connected to the i.MX 91's **secure real-time clock (RTC) and security logic**, allowing them to operate independently of the main application processor. They can be configured to monitor external signals for predefined tamper events, such as level changes or transitions, and can trigger security responses even when the system is in low-power states, provided the secure power domain is maintained.

Tamper events can be configured to generate interrupts, status flags, or security actions, including logging of the event timestamp, system reset, or secure key invalidation, depending on the software and security configuration. The tamper pins support both active and passive tamper detection mechanisms, enabling designers to implement robust physical security monitoring tailored to their application.

Table 36 lists the tamper detection interface signals exposed on the Lino iMX91 board-to-board connector.

Table 36: Tamper detection interface

X2 pin	Lino signal name	SoC ball name	I/O	Description/Remarks
S15	TAMPER0 MSP_3	TAMPER0	I	Tamper detection pin software configurable and available also in low-power modes
S27	TAMPER1 MSP_8	TAMPER1	I	Tamper detection pin software configurable and available also in low-power modes

Recommendation when not Used
When not used, TAMPERx (where x is 0 or 1) input pins shall be placed in a defined inactive state. NXP recommends disabling unused tamper inputs in software and biasing the corresponding pins to the inactive level (typically logic low) using internal or external pull resistors to prevent spurious tamper events.

5.13 Trusted Platform Module (TPM 2.0)

The Lino iMX91 System-on-Module integrates a **Trusted Platform Module (TPM) 2.0**, ST33KTPM2IWLBZA9, providing a hardware-based root of trust for secure key storage, cryptographic operations, and platform integrity functions. The TPM enhances system security by enabling features such as secure boot measurement, device authentication, and protected storage of sensitive credentials.

The TPM communicates with the host processor via a **SPI interface**, operating as an SPI slave device. Control and data transfer are performed using the standard SPI signals (SPI_CLK, SPI_MOSI, SPI_MISO, and SPI_CS#). A dedicated interrupt output (PIRQ#) allows the TPM to asynchronously signal events to the host processor. The TPM reset input (RST#) is connected to the system reset domain to ensure deterministic initialization during power-up and reset events.

An external pull-up resistor (10 kΩ to 1.8 V) is provided on the PIRQ# signal to guarantee a defined logic level when the TPM interrupt output is deasserted or in a high-impedance state, improving signal robustness during reset and low-power conditions.

Key characteristics of the ST33KTPM2IWLBZA9 include:

- **TPM specification:** compliant with **Trusted Computing Group (TCG) TPM 2.0**
- **Interface:** SPI (slave mode)
- **Security functions:** secure key storage, cryptographic acceleration, platform integrity measurement
- **Interrupt support:** dedicated PIRQ# output with on-module pull-up
- **Reset control:** hardware reset input synchronized with system reset
- **Supply voltage: 1.8 V**

The integrated TPM provides a standardized, hardware-enforced security foundation suitable for applications that require strong device identity and data protection.

5.14 UART – Universal Asynchronous Rx/Tx

5.14.1 Interface Overview

The Lino iMX91 system-on-module provides serial communication capabilities through the **UART (Universal Asynchronous Receiver/Transmitter)** controllers integrated in the NXP i.MX 91 system-on-chip. The UART interfaces are designed for reliable, low-latency, point-to-point communication and are commonly used for system debugging, console access, firmware updates, and communication with external serial peripherals.

The i.MX 91 UART controllers support **full-duplex asynchronous communication** and can be configured for a wide range of baud rates and data formats. Each UART interface provides transmit and receive data signals and may optionally support hardware flow control using RTS and CTS signals, depending on pin multiplexing and configuration. The UART subsystem integrates with the SoC's interrupt and DMA infrastructure to enable efficient data transfer with minimal CPU overhead.

Key Features and Supported Capabilities

- **Interface standard:** asynchronous serial UART
- **Data formats:** configurable word lengths (typically 7 or 8 bits), optional parity (none, even, odd), one or two stop bits
- **Baud rates:** supports standard and high-speed baud rates, up to 5 Mbps maximum, depending on configuration (oversampling configuration, peripheral clock frequency, signal integrity and board design)
- **Flow control:** optional hardware flow control (RTS/CTS), software flow control support
- **System integration:** interrupt and DMA-driven operation

5.14.2 UART Interfaces

Table 37 lists the UART interfaces exposed on the Lino iMX91 board-to-board connectors, including their pin assignments, signal direction, and functional descriptions.

Table 37: UART interfaces

X1/X2 pin	Lino signal name	SoC ball name	SoC alternate function	SoC alternate function name	I/O	Description/Remarks
UART_1						
P9	UART_1_RXD	GPIO_IO05	ALT5	uart6.RX	I	Receive Data of general-purpose UART_1
P11	UART_1_TXD	GPIO_IO04	ALT5	uart6.TX	O	Transmit Data of general-purpose UART_1
P8	UART_1_CTS	GPIO_IO06	ALT5	uart6.CTS_B	O	Clear to Send of general-purpose UART_1
P6	UART_1_RTS	GPIO_IO07	ALT5	uart6.RTS_B	I	Request to Send of general-purpose UART_1
UART_2						
P13	UART_2_RXD	GPIO_IO09	ALT5	uart7.RX	I	Receive Data of general-purpose UART_2
P15	UART_2_TXD	GPIO_IO08	ALT5	uart7.TX	O	Transmit Data of general-purpose UART_2
S7	UART_2_CTS	GPIO_IO10	ALT5	uart7.CTS_B	I	Clear to Send of general-purpose UART_2
S5	UART_2_RTS	GPIO_IO11	ALT5	uart7.RTS_B	O	Request to Send of general-purpose UART_2
UART_3						
P17	UART_3_RXD	UART1_RXD	ALT0	uart1.RX	I	Receive data of Cortex®-A55 debug UART_3[1]
P19	UART_3_TXD	UART1_TXD	ALT0	uart1.TX	O	Transmit data of Cortex®-A55 debug UART_3[1] Boot mode 0 with 10kΩ pull-down and controlled by CTRL_RECOVERY_MICO#
UART_4						
P23	UART_4_RXD	UART2_RXD	ALT0	uart2.RX	I	Receive data of Cortex®-A55 debug UART_4[2]
P25	UART_4_TXD	UART2_TXD	ALT0	uart2.TX	I	Transmit data of Cortex®-A55 debug UART_4[2] Boot mode 1 with 10kΩ pull-down

[1] Debug console for ROM bootloader, U-Boot, and operating system.
[2] Application-focused UART interface.

5.14.3 Recommended Use Cases

The Lino iMX91 provides multiple UART interfaces that can be assigned to different processor cores and software roles, enabling flexible serial communication, system debugging, and application-specific use cases. The following sections describe the **recommended UART assignments** and typical usage scenarios for each interface, based on the i.MX 91 architecture. Note that the UART peripherals are shared SoC resources and are not statically bound to either the Cortex®-A55 application cores or the Cortex®-M33 real-time core.

UART_1 – Real-Time Core Debug UART (Cortex®-M33)

The UART_1 is a general-purpose interface and is connected to the `uart6` interface of the SoC and is commonly used as a debug and console interface for the **Cortex®-M33 real-time core**. It allows independent visibility into real-time firmware execution without interfering with the Linux console running on the Cortex®-A55.

Typical usage:

- General-purpose interface, recommended to debug the Cortex®-M33
- Used for RTOS or bare-metal debug output
- Enables parallel debugging of Cortex®-A55 or Cortex®-M33 cores
- Not used by the ROM bootloader by default

UART_2 – Auxiliary / Flexible UART Interface

The UART_2 is a general-purpose interface is connected to the `uart7` interface of the SoC and is provided as an additional flexible UART interface that can be assigned to either the **Cortex®-A55 or Cortex®-M33**, depending on the application requirements. It is suitable for auxiliary serial communication or application-specific functions.

Typical usage:

- General-purpose interface, recommended to debug either Cortex®-A55 or Cortex®-M33
- Not reserved for boot or system debug
- Suitable for additional peripherals or custom protocols
- Fully software-configurable

UART_3 – Primary System Debug Interface (Cortex®-A55)

The UART_3 is connected to the `uart1` interface of the SoC and is intended as the primary debug and console interface for the i.MX 91 **Cortex®-A55 application core**. It is used throughout the **complete boot chain**, from the ROM bootloader to the operating system, making it the main interface for early bring-up and system diagnostics.

Typical usage:

- Default debug console for ROM bootloader, U-Boot, and operating system
- Associated primarily with the Cortex®-A55
- Used for boot messages, system logs, and recovery
- Enabled early during system startup

UART_4 – General-Purpose Application UART (Cortex®-A55)

The UART_4 is connected to the `uart2` interface of the SoC and provides a **general-purpose serial communication** interface intended for application-level use on the Cortex®-A55. It is not reserved for boot or debug by default and is typically used to interface with external serial devices.

Typical usage:

- Application-focused UART interface for the Cortex®-A55
- Suitable for modems, controllers, or serial bridges
- Can be used as a secondary debug or logging port

- Fully configurable by the operating system

5.15 USB – Universal Serial Bus

The Lino iMX91 system-on-module provides **USB (Universal Serial Bus)** connectivity based on the native USB controllers and PHYs integrated in the NXP i.MX 91 system-on-chip, enabling high-speed peripheral and host operation for a wide range of applications. The i.MX 91 integrates **two USB 2.0 High-Speed (HS) controllers**, each supporting data rates of **up to 480 Mbps** and compliant with the **Universal Serial Bus Revision 2.0 specification**, including applicable ECNs, errata, and the On-The-Go (OTG) and Embedded Host supplements.

Each USB interface supports **dual-role operation (Host or Device)** and uses standard USB signaling, including differential data pairs (DP/DN), VBUS detection, and ID-based role selection. The USBx_VBUS (where x is the interface designator) pins are used for VBUS sensing only and **must not** be directly connected to the 5 V USB VBUS rail. Instead, each USBx_VBUS pin is isolated from the 5 V domain by an **external 30 kΩ, 1% precision resistor**, allowing safe voltage detection while meeting the electrical requirements of the USB PHY. The USB_ID signals are **biased to 1.8 V** to support OTG role detection.

The USB 2.0 PHY parameters meet the electrical compliance requirements defined in the USB 2.0 specification. Signal integrity is ensured through PHY calibration using the USBx_TXRTUNE pins (where x is the interface designator) which, together with an external **200 Ω, 1% precision resistor to ground**, calibrate the USB DP/DN **45 Ω source impedance**.

Table 38: USB interfaces overview

Lino USB port	Speed capabilities (SoC)	Role capabilities (SoC)	Role according to Lino standard	Recovery mode
USB_1	USB 2.0	Host and client	OTG (host and client)	Supported[1]
USB_2	USB 2.0	Host and client		Not supported[2]

[1] On the i.MX 91, the **USB_1 interface** is used by the ROM bootloader for **recovery (Serial Download Mode)**.
[2] USB2 is not **used by the ROM for recovery mode** and is intended for application-level USB functionality once the system has booted.

5.15.1 USB Interfaces

Tables 39 and 40 list the interface pins for USB interfaces 1 and 2, respectively, including the corresponding Lino standard functions, i.MX 91 ball names, I/O direction, and usage descriptions.

Table 39: USB interface 1

X1 pin	Lino standard function	i.MX 91 ball name	I/O	Description/Remarks
Exposed on Connector				
P35	USB_1_D_P	USB1_D_P	I/O	Positive differential USB Signal
P33	USB_1_D_N	USB1_D_N	I/O	Negative differential USB Signal
P31	USB_1_VBUS	USB1_VBUS	I	Use this pin to detect if VBUS is present this pin is a 5V input
Not Exposed on Connector				
–	USB_A_ID	USB1_ID	I	Use this pin to detect the ID pin if you use the port in OTG mode 10kΩ pull-up to 1.8V
–	–	USB1_TXRTUNE	I	The USB transmit impedance calibration pin connected to GND

Table 40: USB interface 2

X2 Pin	Lino standard function	i.MX 91 ball name	I/O	Description/Remarks
Exposed on Connector				
S59	USB_2_D_P	USB2_D_P	I/O	Positive differential USB Signal
S57	USB_2_D_N	USB2_D_N	I/O	Negative differential USB Signal
S49	USB_2_VBUS	USB2_VBUS	I	Use this pin to detect if VBUS is present his pin is a 5V input
Not Exposed on Connector				
–	USB_B_ID	USB2_ID	I	Use this pin to detect the ID pin if you use the port in OTG mode 10kΩ pull-up to 1.8V
–	–	USB2_TXRTUNE	I	The USB transmit impedance calibration pin connected to GND

6 eFUSE Configuration

The NXP i.MX 91 system-on-chip integrates a **One-Time Programmable (OCOTP) fuse controller** located in the Always-On (AON) domain. The OCOTP block provides non-volatile configuration storage for device identity, boot configuration, security settings, trimming data, and other permanent system parameters.

Fuse values are permanently programmed into silicon and are shadowed into memory-mapped registers during system reset. The read-only shadow registers (e.g., `OCOTP_FUSE_DATAx`) reflect the programmed fuse state and are used by the BootROM and system software during initialization.

The OCOTP controller supports:

- Permanent one-time programming of fuse words
- Shadow register access for software readback
- Boot configuration via BOOT_MODE and BOOT_CFGx fields
- Security configuration for secure boot and access control
- Lock mechanisms to prevent further modification of specific fuse regions

Fuse programming is irreversible. Once programmed, a fuse bit cannot be cleared.

6.1 Shadow Registers

Upon reset, the OCOTP controller loads fuse contents into shadow registers. These registers are accessible via the memory-mapped OCOTP interface and are documented in the i.MX 91 Reference Manual as `OCOTP_FUSE_DATA0` through `OCOTP_FUSE_DATAn`.

The reset value of each shadow register depends on the programmed fuse state. Software must treat these registers as read-only representations of the non-volatile fuse contents.

Shadow Register Reload
Shadow registers are reloaded only after a system reset. After programming fuses, a reset is required before the updated values become visible through the OCOTP shadow registers.

6.2 Boot Configuration Fuses

Boot source and boot behavior are determined by dedicated eFUSE fields in the `BOOT_MODE` and `BOOT_CFG` groups.

These fuses control:

- Primary boot device selection
- FlexSPI configuration overrides
- NAND page size configuration
- DLL and timing configuration
- Recovery and override behavior

Improper configuration
Improper programming of boot configuration fuses may render the device unbootable.

6.3 Programming Procedure

Fuse programming must follow the procedure defined in the i.MX 91 Reference Manual (OCOTP chapter). The typical high-level flow is:

1. Ensure the system is powered within specification and not in low-power mode.
2. Unlock the OCOTP controller (if required).
3. Program the target fuse word via the OCOTP programming interface.
4. Wait for programming completion and verify status.
5. Perform a system reset to reload shadow registers.
6. Read back the corresponding shadow register to verify correct programming.

Programming operations are subject to strict timing and voltage requirements as defined in the reference manual. The required programming voltage and current must be maintained during fuse burning to avoid incomplete programming.

6.4 Security and Access Control

Access to OCOTP registers may be restricted depending on system configuration. In secure configurations, fuse programming may only be permitted in:

- Secure mode (TrustZone-enabled systems)
- Supervisor mode
- Early boot context (e.g., manufacturing image or bootloader stage)

Once security-related fuses (e.g., secure boot enable, root-of-trust keys, or lock bits) are programmed, reverting to a non-secure configuration is not possible.

6.5 Locking Mechanisms

Certain fuse fields include lock bits that permanently disable further writes to defined fuse regions. These lock bits prevent accidental or malicious modification of critical configuration values.

Programming lock bits should only be performed after full validation of the target configuration.

6.6 Manufacturing Considerations

Fuse programming is typically performed:

- During manufacturing provisioning
- In a secure production environment
- Using validated programming utilities (e.g., U-Boot fuse commands or NXP provisioning tools)

It is strongly recommended to:

- Validate fuse values on development hardware before mass production
- Program only the minimum required fuse bits
- Avoid programming reserved fields
- Maintain a documented fuse map for each product variant

Irreversible Operation
Programming eFUSE bits is permanent and cannot be undone. Incorrect fuse configuration, especially boot source or security fuses, may permanently disable the device.
Always validate fuse settings on evaluation hardware before production programming.

7 Test Points

The Lino iMX91 System-on-Module includes a set of **test points (TPs)** intended to support debugging, signal observation, and system validation during development, manufacturing, and bring-up. These test points provide access to selected internal signals without interfering with normal module operation, enabling efficient measurement and troubleshooting.

Each test point is identified by a unique reference designator (for example, TP01, TP02) and is connected to specific signals such as power supply rails, PMIC control signals, boot and reset signals, communication interfaces, and selected clock or status lines. Where applicable, the presence of on-module pull-up or pull-down resistors associated with these signals is documented to ensure correct interpretation of measured levels during testing.

Table 41 lists the test points available on the Lino iMX91 System-on-Module, providing access to selected debug, power, and control signals for system validation and troubleshooting.

Table 41: Test points

Test Point	Lino iMX91 signal	I/O	Description/Remarks
JTAG			
TP1	JTAG_TDI	I	Test data in: serial data input to the JTAG Test Access Port (TAP) Exposed on X2 connector, pin S59
TP3	JTAG_TDO	O	Test data out: serial data output from the JTAG TAP Exposed on X2 connector, pin S61
TP9	JTAG_TCK	I	Test clock: provides the clock signal that synchronizes all JTAG operations Exposed on X2 connector, pin S67
TP12	JTAG_TMS	I	Test mode select: controls the TAP state machine transitions when sampled on the rising edge of TCK Exposed on X2 connector, pin S65
PMIC			
TP6	V1.8	–[1]	Intermediate 1.8 V system rail generated by BUCK3. This rail supplies multiple internal and I/O-related domains, and is further distributed and filtered to downstream consumers within the module.
TP13	V1.1_DDR	–[1]	Access to the DDR core supply rail (V1.1_DDR) generated by BUCK1 of the PMIC. This rail supplies the LPDDR4 memory subsystem of the i.MX 91, specifically powering the DDR core logic and memory arrays. Tightly regulated to meet LPDDR4 electrical requirements and is critical for reliable memory operation
TP14	VDD_SOC	–[1]	Main digital core supply generated by BUCK2 of the PMIC. This rail powers the primary logic domains of the i.MX 91, including the Cortex®-A cores, interconnect, and internal logic. Voltage is dynamically controlled according to performance and power states.
TP15	V1.8	–[1]	Same 1.8 V rail as TP6, located after additional routing and decoupling. Provided as an auxiliary measurement point to verify rail stability closer to the load.
TP16	V3.3	–[1]	3.3 V I/O and peripheral supply generated by BUCK4. This rail powers external-facing interfaces and peripherals requiring 3.3 V, such as selected I/O banks and on-module components.
TP17	V3.3_1.8_SD	–[1]	PMIC LDO1 output supplying the SD interface voltage rail. This rail provides either 3.3 V or 1.8 V, selectable via the PMIC SD voltage selection mechanism, and powers the SD/SDIO I/O domain of the i.MX 91.
TP18	V0.8_ANA	–[1]	PMIC LDO2 output supplying the 0.8 V analog rail. This rail powers sensitive analog domains of the i.MX 91, such as PLLs, analog PHY blocks, and internal bias circuitry requiring low-noise supply.
TP19	V1.8_BBSM	–[1]	Backup Battery Supply Monitor (BBSM) 1.8 V rail. This test point provides access to the 1.8 V always-on supply that powers the SNVS/BBSM domain of the i.MX 91, including tamper detection, and low-power retention logic.
TP20	GND	–[1]	Ground reference test point.

Continued on next page

Table 41: Test points (Continued)

Test Point	Lino iMX91 signal	I/O	Description/Remarks
Control Signals			
TP21	CTRL_RESET_MOCI# PMIC_RESET_CPU	O	Provides access to the PMIC_RESET_CPU signal, which is used to assert a hardware reset to the i.MX 91. This reset line is driven by the PMIC and connected to the SoC reset input CTRL_RESET_MOCI#. Allows the PMIC to force the processor into a reset state during power-up, power-down, fault conditions, or brown-out events
TP22	PMIC_ON_REQ	O	Provides access to the PMIC_ON_REQ signal, which is driven by the i.MX 91 to request the PMIC to transition from standby or off state into the active power-up sequence. This signal is asserted by the SoC once the always-on domain is powered and the system intends to enter normal operation.
TP23	PMIC_STBY_REQ	O	Provides access to the PMIC_STBY_REQ signal, which is driven by the i.MX 91 to request the PMIC to transition to standby or off state from the active power-up sequence.
Not connected			
TP2			Not connected test points.
TP4			
TP5			
TP7			
TP8			
TP10			
TP11			

[1] These test points do not have a defined I/O direction, as they are intended for voltage measurement only. These points provide access to various power rails and signals for monitoring purposes and should not be driven by external sources.

8 Low Power Modes

TBA
More information about low power modes will be available on the next releases of the datasheet.

9 Recovery Mode

The **Recovery Mode (USB Serial Loader)** can be used to download new software to the Lino iMX91 even when the bootloader is no longer capable of booting the module. In the normal development process, this mode is not needed. When the module is in recovery mode, the USB_1 interface is used to connect it to a host computer. You will find additional information at our Developer Center:

https://developer.toradex.com/hardware/hardware-resources/recovery-mode/imx-ti-recovery-mode

9.1 Boot Mode Selection and Pin Mapping

The i.MX 91 enters USB Serial Loader mode when the BOOT_MODE[1:0] (UART3_TXD, board-to-board connector pin P19, and UART4_TXD, board-to-board connector pin P25) pins are sampled at reset and configured to select Serial Loader. The boot mode pins are latched during power-on reset or warm reset. When BOOT_MODE[1:0] is set to 01, the ROM bootloader bypasses all external boot devices and enters USB Serial Downloader mode.

Pin Mapping

- BOOT_MODE0 is connected to UART_3_TXD
- BOOT_MODE1 is connected to UART_4_TXD

In order to enter recovery mode, the dedicated recovery pin needs to be pulled down with $\leq$1kΩ during the initial power-on (cold boot) of the module. The CTRL_RECOVERY_MICO# function on the board-to-board connector pin P80 is standardized in the Lino module specifications. It is highly recommended to add at least a test point on the carrier board to the pin P80 to be able to enter recovery mode. There is no need for a pull-up resistor on the carrier board.

External Storage Devices
When this mode is selected, the ROM bootloader does not attempt to boot from external storage devices and instead initializes the USB interface for firmware download.

Recovery Procedure

1. Power off the system.
2. Configure the carrier board to set BOOT_MODE[1:0] = 01.
3. Connect the USB OTG/device port to a host PC.
4. Apply power or release reset (CTRL_RECOVERY_MICO#, on pin P80, is active low).
5. The module is set as a USB device controlled by the ROM bootloader.
6. Download and execute the desired firmware image (refer to the Developer Website).
7. Power off the system and restore BOOT_MODE pins to the normal boot configuration.
8. Power on the system to resume standard boot operation.

10 Known Issues

TBA
More information about known issues will be available on the next releases of the datasheet.

11 Technical Specification

11.1 Electrical Characteristics

This section describes the **electrical characteristics and power-related specifications** of the Lino iMX91 System-on-Module. It covers the integrated power management architecture, absolute maximum ratings, recommended operating conditions, and power sequencing requirements necessary to ensure reliable operation, compliance with device limits, and long-term system stability. The information provided is intended to support system design, power budgeting, and validation of carrier boards and end applications using the module.

11.1.1 PMIC – Power Management IC

The Lino iMX91 system-on-module integrates a dedicated **Power Management Integrated Circuit (PMIC)**, `MPF9453AVMA1HN`, to provide all required power rails for the NXP i.MX 91 system-on-chip and on-module peripherals. The PMIC is specifically designed to support i.MX 91 power requirements and implements a complete, sequenced, and monitored power solution optimized for low power consumption, reliability, and system safety.

The `MPF9453AVMA1HN` supplies and manages multiple voltage domains required by the i.MX 91, including core, logic, memory, analog, and I/O rails, in accordance with the power-up, power-down, and timing requirements defined in the i.MX 91 datasheet and reference manual. It supports dynamic voltage control to enable efficient operation across different performance and low-power modes, including standby and suspend states.

The PMIC integrates multiple **buck converters**, **low-dropout regulators (LDOs)**, and **power switches**, along with voltage monitoring, reset generation, and fault detection mechanisms. Tight coupling between the PMIC and the i.MX 91 allows coordinated control of power states through an I^2C interface, enabling software-managed power sequencing, brown-out detection, and thermal or overcurrent protection handling.

The PMIC is programmable using the interface `PMIC_I2C` on address `0x32`.

11.1.2 Absolute Maximum Ratings

Table 42: Absolute maximum ratings

Signal	Description	Minimum	Maximum	Unit
V_IN	Main input supply to PMIC (carrier board input)	-0.3	6.0	V
NVCC_BBSM_1P8	Battery-backed secure domain supply	-0.3	2.15	V
GPIO supply voltage NVCC_AON,NVCC_GPIO,NVCC_WAKEUP	SoC I/O pins with 3.3 V logic level	-0.3	3.8	V
ADC_IN	ADC analog input (referenced to VDD_ANA_1P8)	-0.3	2.1	V
USB VBUS input detect USB1_VBUS, USB2_VBUS	USB VBUS sense input (via external resistor divider)	-0.3	3.95	V

Notes

- Table 42 displays stress limits only; functional operation outside recommended conditions is not guaranteed.
- `NVCC_BBSM_1P8` powers the secure always-on domain and must never exceed 2.1 V.
- ADC inputs must not exceed the analog supply (`VDD_ANA_1P8`) + 0.3 V.
- `USB_VBUS` is sense-only on the i.MX 91 and must be isolated from the 5 V rail using the required external resistor network.

11.1.3 Recommended Operating Conditions

Table 43: Recommended operating conditions

Symbol	Description	Minimum	Typical	Maximum	Unit
V_IN	Main input supply to PMIC	4.75	5.0	5.25	V
NVCC_BBSM_1P8 VCC_BACKUP	Battery-backed secure domain supply (RTC / BBSM)	1.1	1.8	1.98	V

11.1.4 Power Consumption

TBA
More information about power consumption will be available on the Toradex Developer website Toradex Developer website.

11.1.5 Power Sequence

The power-up and power-down behavior of the i.MX 91 is defined by a strict sequencing of supply rails and control signals to ensure correct initialization, stable operation, and reliable shutdown of the device. This sequence is managed in coordination with the PMIC and the SoC's Battery-Backed State Machine (BBSM), governing transitions between the secure always-on state, power-up, run, and power-down modes.

Figure 2 illustrates the complete power sequencing diagram, including the required order of voltage rails, debounce and step timings, power-good indications, and reset behavior.

Figure 2: Lino iMX91 power sequence

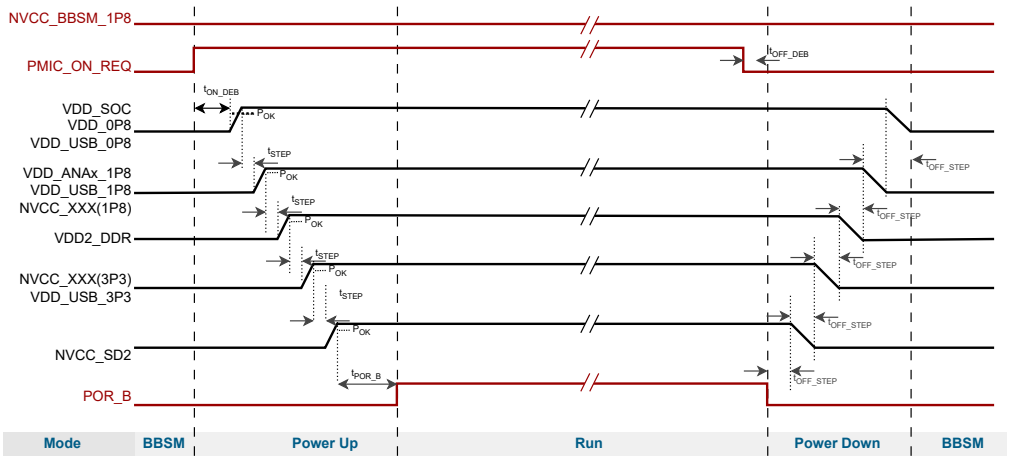


Figure 2 illustrates the complete power sequencing diagram, including the required order of voltage, as shown in Figure 2.

Table 44: Power sequence timing signals

Term	Description/Remarks
t_{ON_DEB}	Debounce delay applied after PMIC_ON_REQ[1] is asserted and before the PMIC begins enabling the first main power rail (VDD_SOC) - Ensures PMIC_ON_REQ is stable and not caused by noise or glitches - Prevents unintended power-up events - Defined by the PMIC configuration
t_{OFF_DEB}	Debounce delay applied before initiating power-down after a power-off condition is detected (for example, PMIC_ON_REQ deassertion or a shutdown request) - Filters transient shutdown events - Ensures a controlled entry into the power-down (PWRDN) state - Defined by the PMIC configuration
t_{STEP}	Delay between enabling successive power rails during power-up or disabling them during power-down - Applied between each power rail transition - Ensures correct voltage ramp sequencing - Allows each rail sufficient time to stabilize before the next step
P_{OK}	Indicates that a given power rail has reached its valid operating voltage range and is stable - Asserted by the PMIC once voltage and timing requirements are met - Used to allow progression to the next power sequencing step - May be required before releasing system reset signals
t_{POR_B}	Minimum time during which the POR_B signal remains asserted after all required power rails are valid and POK signals are asserted - Ensures internal logic and clocks are fully stabilized - Guarantees a deterministic reset release for the SoC - POR_B must remain asserted throughout the entire power-up sequence

[1] 1M Ω pull-down to GND.

Power-On-Request
POR_B must be asserted whenever VDD_SOC is powered down but NVCC_BBSM_1P8 is powered up when the processor is in BBSM mode.

Power-Up Sequence

The i.MX 91 power-up sequence shall follow the order below to ensure correct initialization of the secure/always-on domain and proper reset behavior:

1. Enable NVCC_BBSM_1P8 (BBSM/secure domain I/O supply)
2. PMIC_ON_REQ assertion by SoC (occurs after NVCC_BBSM_1P8 is valid)
3. Enable VDD_SOC digital core supplies
4. Enable all VDD_0.8V supplies (analog, PHY, PLL 0.8 V rails)
5. Enable all remaining 1.8 V supplies, including:
   - VDD_ANAx_1P8, VDD_LVDS_1P8, VDD_MIPI_1P8, VDD_USB_1P8, and analog/PHY/PLL rails
   - NVCC_XXX (1.8 V) I/O supplies
6. Enable DDR I/O supplies (DDR interface power rails)
7. Enable 3.3 V supplies *(this step may be performed simultaneously with Step 5 or Step 6 if required)*
   - All NVCC_XXX (3.3 V) I/O supplies
   - VDD_USB_3P3
8. Release POR_B only after all required rails are within specification
   - POR_B must remain asserted throughout Steps 1 to 7

Power-Down Sequence

The i.MX 91 power-down requirements are:

- Turn off NVCC_BBSM_1P8 last.
- Turn off VDD_SOC after all other non-BBSM rails, or at the same time as other non-BBSM rails.

- No specific sequencing is required for the remaining non-BBSM power rails during power-down.

11.1.6 Watchdog Signal

The Lino iMX91 System-on-Module implements a **hardware watchdog supervision mechanism** using the `PMIC_WDOG_B` signal, which is driven by the i.MX 91 SoC and monitored by the PMIC. From the SoC perspective, `PMIC_WDOG_B` is an **active-low watchdog output**, periodically toggled or asserted by software to indicate correct system operation. From the PMIC perspective, the signal is an **input** used to supervise the SoC.

If the PMIC detects that `PMIC_WDOG_B` is not asserted or toggled within the configured timeout window, it interprets this as a watchdog fault and initiates a corrective action, typically forcing a system reset by asserting the appropriate reset signals to the SoC. This mechanism enables recovery from software lockups or abnormal operating conditions.

The `PMIC_WDOG_B` signal may be optionally pulled up to the 1.8 V domain via an on-module resistor to ensure a defined logic level during reset or high-impedance states. Watchdog timing, enablement, and response behavior are programmable via the PMIC control interface, allowing the supervision mechanism to be tailored to application requirements.

11.2 Mechanical Characteristics

11.2.1 SoM Outline Dimensions

- 30mm x 30mm x 6mm

TBA
Mechanical drawings will be available on the next releases of the datasheet.

11.2.2 Mating Connector Information

To interface with the board-to-board connectors used on the Lino iMX91 system-on-module, the carrier board must be equipped with the corresponding mating connectors compatible with the Amphenol `10164228-1001A1RLF` sockets mounted on the module. To mate with the board-to-board sockets used on the Lino iMX91 system-on-module, the carrier board shall be equipped with compatible Amphenol BergStak® 0.40 mm pitch receptacle connectors. These mating connectors provide the mechanical and electrical interface required to stack the module onto the carrier board while maintaining reliable signal integrity and power delivery.

Table 45 shows the characteristics of the recommended mating connectors.

Table 45: Mating connectors

Part number (PN)	Connector family	Pitch	Positions	Type	Stacking height
10164227-1001A1RLF	BergStak®	0.40 mm	100	Receptacle	1.5 mm
10164227-1004A1RLF	BergStak®	0.40 mm	100	Receptacle	4.0 mm

These variants allow carrier board designers to select the appropriate stacking height based on mechanical clearance, thermal requirements, and enclosure constraints. Carrier board designs must follow the connector manufacturer's recommended land patterns, mechanical keep-out zones, and routing guidelines to ensure proper mating, long-term reliability, and compliance with the electrical requirements of the Lino iMX91 platform.

11.2.3 Thermal Specifications

TBA
Thermal specifications will be available on the next releases of the datasheet.

12 Product Compliance

Up-to-date information about product compliance, such as RoHS, CE, UL 94, Conflict Minerals, REACH, and others, can be found on our website[4].

[4] https://www.toradex.com/support/product-compliance

13 Device and Documentation Support

13.1 Trademarks

DISCLAIMER