Radar Baseboard MCU7 Plus

XENSIV™ 60 GHz radar system platform

Board version V3.1

About this document

Scope and purpose

This application note describes the "Radar Baseboard MCU7 Plus" (DEMO RADAR MCU7 PLUS), part of Infineon's 60 GHz radar system platform. It also introduces the concept of the platform, which supports multiple sensors. At the heart of the board is the Microchip ATSAMS70Q21, a 32-bit Arm® Cortex®-M7 microcontroller unit (MCU). The microcontroller enables interfacing up to two sensors. It also provides a Hi-Speed USB 2.0 interface to a host computer for visualization or fast data processing. In addition, the board is compatible with the Arduino MKR standard, which facilitates access to existing mass-market daughter boards for mass data storage or wireless communication Arduino boards.

Intended audience

The intended audience for this document are design engineers, technicians, and developers of electronic systems, working with Infineon's XENSIV™ 60 GHz radar sensors.

Related documents

Additional information can be found in the documentation provided with the Radar Development Kit tool in the Infineon Developer Center (IDC), or from www.infineon.com/60GHz.

Table of contents

About this document **1**
Table of contents **2**
1 Introduction **3**
1.1 60 GHz radar system platform 3
1.2 Board overview and key features 3
2 Sensor interface **5**
2.1 Overview 5
2.2 Supply 6
2.3 Level shifters 8
2.4 Sensor connectors 9
2.5 Current measurement of radar sensor shields 11
3 Arduino MKR interface **12**
4 Firmware **13**
4.1 Overview 13
4.2 USB 14
4.3 EEPROM 14
4.4 RGB LED 14
4.5 Firmware development and debugging 15
4.6 Flashing via bootloader 16
5 Microcontroller pin map **17**
5.1 Sensor interface 1 17
5.2 Sensor interface 2 18
5.3 Arduino MKR interface 20
5.4 Other pins 21
References **22**
Revision history **23**
Disclaimer **24**

1 Introduction

1.1 60 GHz radar system platform

The 60 GHz radar system platform is the evaluation platform for Infineon's 60 GHz radar sensors. It consists of the "Radar Baseboard MCU7 Plus" as the microcontroller board and a 60 GHz radar sensor shield, e.g., a BGT60UTR11AIP shield or a BGT60UTR13DAIP shield.

Figure 1 illustrates the "Radar Baseboard MCU7 Plus". There are white circular markings adjacent to the shield harnesses. The 60 GHz radar sensor shields also have similar markings. These must be aligned in order to plug the sensor shield in the correct orientation.

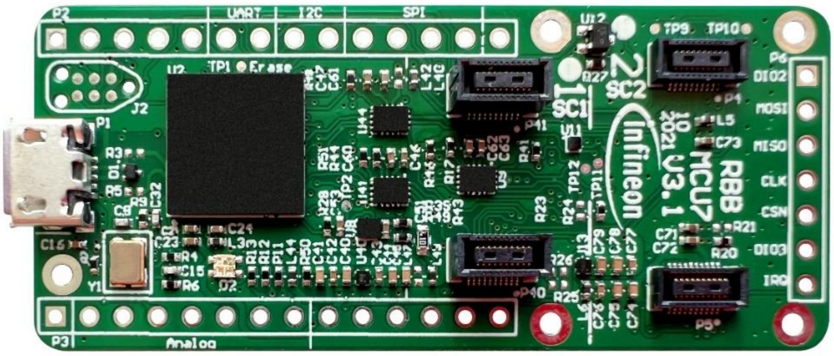

Figure 1 The 60 GHz radar system platform with a sensor shield unplugged

1.2 Board overview and key features

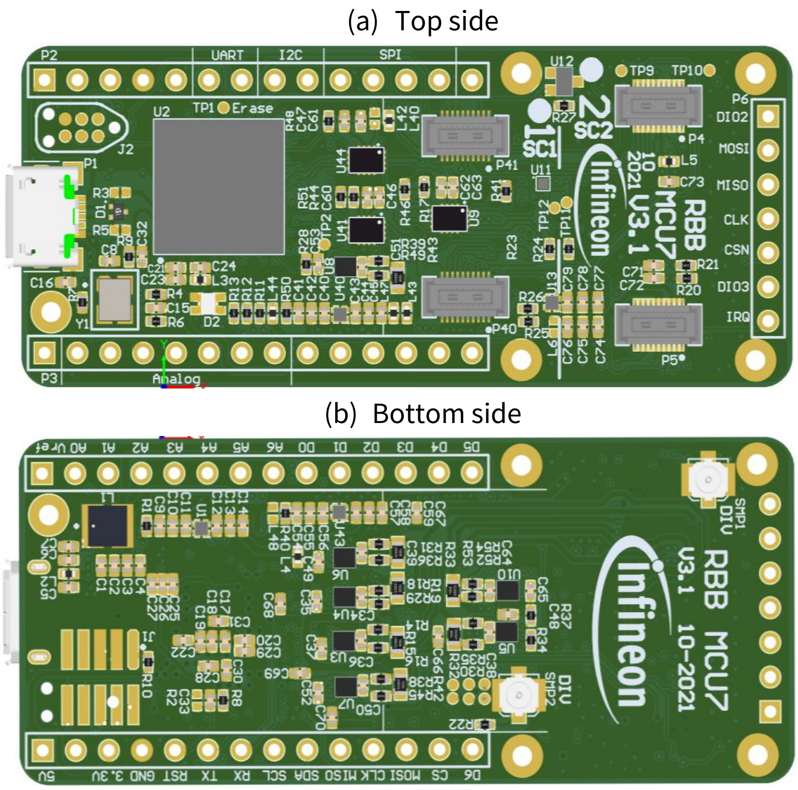

Figure 2 The "Radar Baseboard MCU7 Plus"

The "Radar Baseboard MCU7 Plus", illustrated in Figure 2, is a 59.4 mm x 25.4 mm printed circuit board (PCB). Its main purpose is to provide a generic sensor interface for Infineon's 60 GHz radar sensors. The central MCU can perform radar data processing or forward the sensor data to a USB interface or an Arduino MKR interface. The board's block diagram is depicted in Figure 3 and its key features are as follows:

- **Arm® Cortex®-M7 MCU** – The central device on the "Radar Baseboard MCU7 Plus" is an ATSAMS70Q21, a 32-bit Arm® Cortex®-M7. It runs at a clock frequency of 300 MHz and contains a 384 kB SRAM. The microcontroller is supplied with a 3.3 V supply, which is decoupled from the other 3.3 V supplies for the sensors to minimize cross-talk between the digital signals and the sensor supply.
- **USB interface** – Since radar data acquisition can generate significant amounts of raw data, a Hi-Speed-USB interface with data rates of up to 480 Mbit/s is used on the "Radar Baseboard MCU7 Plus". This provides sufficient bandwidth to forward the radar data to a host PC for visualization or some other powerful computer for radar data processing.
- **Sensor interface** – The "Radar Baseboard MCU7 Plus" features two radar sensor interfaces. Each sensor daughter shield contains an EEPROM connected via an I^2C interface. This EEPROM can be used to store a descriptor indicating the type of the baseboard and sensor. This can be used by the firmware to communicate properly with the board. It is even possible to detect a falsely inserted board (rotated by 180 degrees). To enable communication with a radar sensor, the board provides an SPI connection, two analog-to-digital converter (ADC) channels, a digital-to-analog converter (DAC) channel and 10 GPIO pins.
- **Arduino MKR connector** – As radar data processing may require very different application-specific hardware, the "Radar Baseboard MCU7 Plus" supports the Arduino MKR interface. This gives access to a wide assortment of inexpensive add-on hardware for rapid prototyping, like SD cards for raw data storage, wireless communication and so on.
- **RGB LED** – The board has an RGB LED to indicate its status. At start-up of the board, the LED will blink. If a radar sensor is detected, the LED will blink green. A detailed description of the LED status codes can be found in section 4.4.

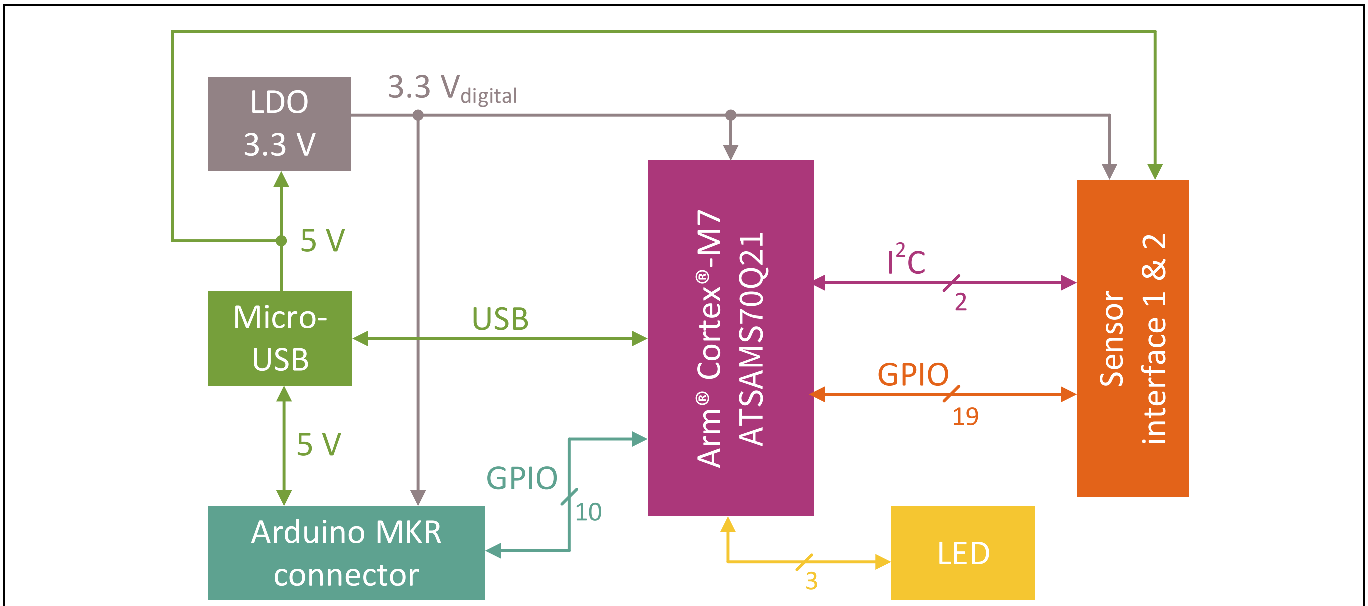

Figure 3 **Block diagram of the "Radar Baseboard MCU7 Plus"**

2 Sensor interface

This section covers the sensor interface that was introduced in the previous section.

2.1 Overview

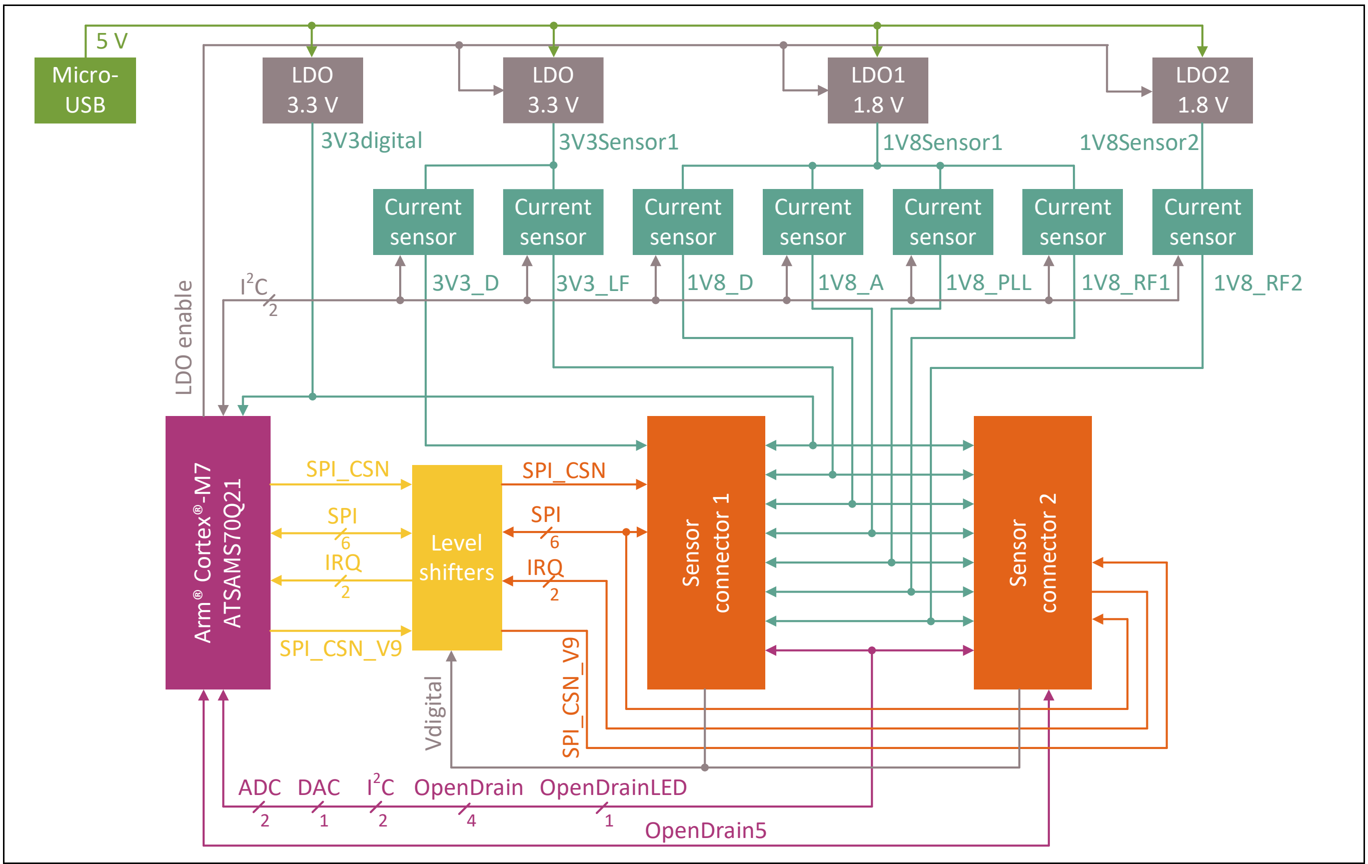

Figure 4 **Block diagram of sensor interface of the "Radar Baseboard MCU7 Plus"**

The block diagram of a single sensor interface of the "Radar Baseboard MCU7 Plus" can be seen in Figure 4. When the board boots up or when a sensor shields is plugged into the sensor connectors, the sensor supply is deactivated. During start-up, the firmware attempts to read out the BGT register from the shield for a Chip-ID. If the shield is a BGT from the Avian family, it will reply with the correct code. If no Avian Chip-ID is detected, the firmware will check if it's a BGT60LTR11AIP which has a code written in the registry, based on the silicon version.

The main communication interface with the sensor shield is a serial peripheral interface (SPI). Given the discrepancy between the MCU supply voltage of 3.3 V and the sensor supply voltage of 1.8 V, level shifters are employed to translate between the different voltage levels. The supply voltage of the level shifters is 3.3 V on one side, and on the other side the power can be supplied either directly from the 1.8 V supply or from the sensor connector. If the supply is provided by the sensor connector, the daughter shield can provide its correct level shifter supply voltage from the supplied voltages.

Besides the five SPI signals, five more digital signals are translated:

- Three shield selection signals to the sensor shields
- Two interrupt signals from the sensor shields to the MCU to indicate that some action by the MCU is required.

In addition, the translated signals the connector also provides:

- Two ADC channels
- One DAC channel
- Six open-drain output signals. One of these is intended to control an LED on the sensor shield. The MCU will activate the LED by pulling the signal to a low level, and deactivate the LED by putting the signal into High-Z mode.

2.2 Supply

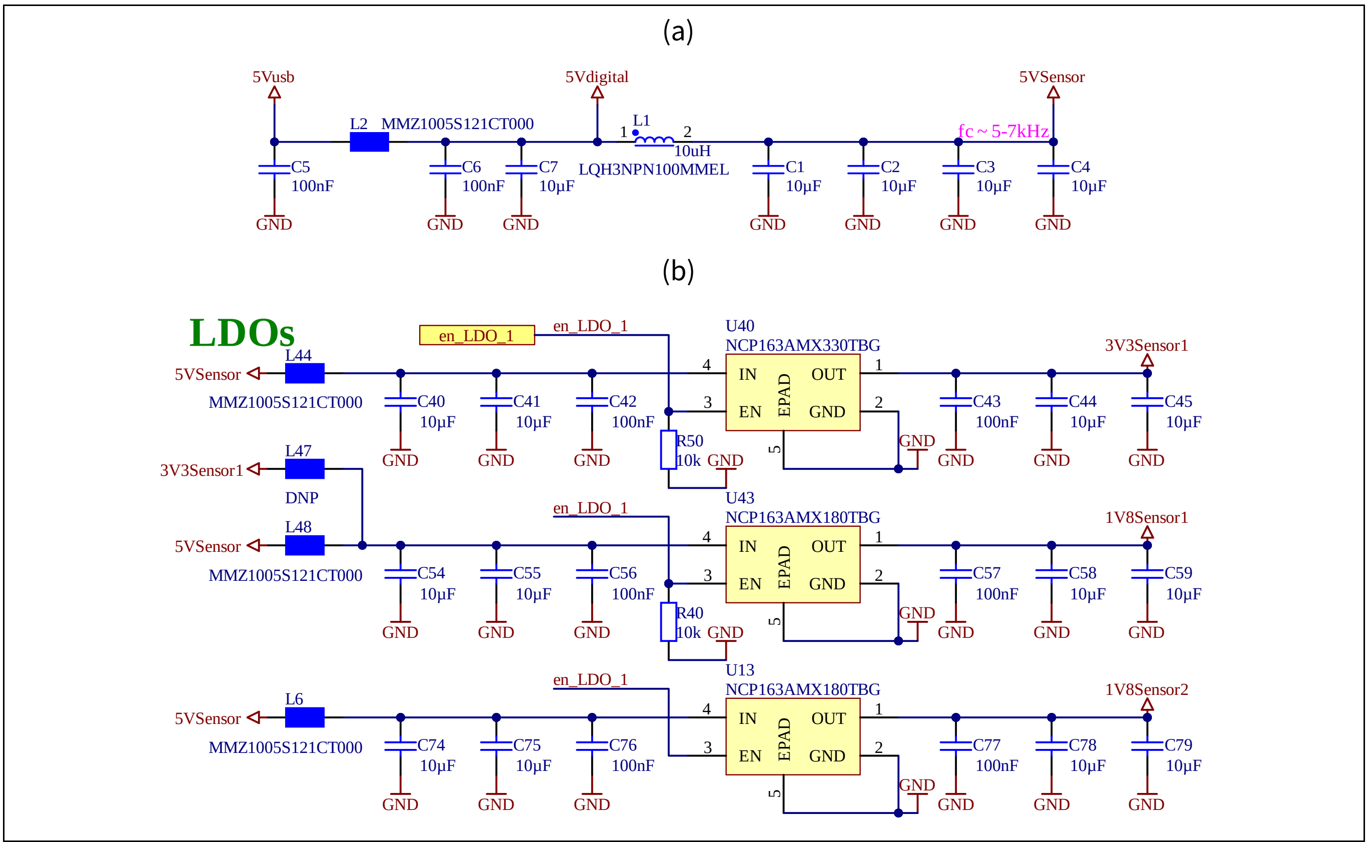

Figure 5 **Sensor supply circuit schematics**

Because radar sensors are very sensitive to voltage fluctuations on the supply, a low-noise power supply is vital. This subsection discusses the supply circuit on the "Radar Baseboard MCU7 Plus". Figure 5a depicts the initial stage of the system supply's circuit. In order to block high-frequency (HF) noise entering the circuit via the USB supply, a low-pass filter has been integrated. This filter consists of the ferrite bead L2 and the capacitors C6 and C7. The filtered voltage is used to supply the MCU (through a 3.3 V LDO). The MCU operation may cause additional unwanted voltage fluctuations on the supply. Therefore, another low-pass filter is placed in the supply. The filter consists of the coil L1 and the capacitors C1, C2, C3 and C4, which results in a cut-off frequency below 8 kHz. This yields a stable supply voltage with very low noise inside a frequency range from about 10 kHz up to the MHz regime.

Figure 5b shows the additional supply circuitry that is required for each sensor. A ferrite bead is placed at the input side of the supply to block any HF noise from upstream (such as might be caused by the MCU). This is fed into an LDO to generate the required voltage of 3.3 V and 1.8 V for the sensor supply. The LDOs used for power conversion are ON Semiconductor's NCP163 low-noise LDOs with an output noise of 6.5 µVRMS. They provide a power supply rejection ratio above 90dB. The remaining noise in the system is mainly caused by the LDOs and tends to be strongest at frequencies around 100 kHz where the LDOs stop working efficiently.

If ghost targets appear in the radar signal, it may be possible to improve the performance by optimizing the power supply. On the "Radar Baseboard MCU7 Plus", the supply of the 1.8 V LDOs can be changed from 5 V to 3.3 V by swapping L47 with L48. This will affect the noise performance on the 1.8 V supply "1V8Sensor1". This configuration suppresses external low frequency better than a single LDO.

Each LDO generates noise at its maximum suppression frequency, which is typically around 100 kHz. This frequency is in the frequency range of the radar IF signals. Hence, noise in that this frequency range can cause ghost targets in radar data processing. In the configuration with two LDOs in series (with L47), the noise caused by the first LDO stage will be amplified by the second stage. To minimize the noise around 100 kHz, a single LDO stage can be used in the configuration with L48. To summarize, for supplies with strong low-frequency noise components, it is better to use L48, while for low-noise supplies L47 is preferable.

2.3 Level shifters

This subsection explains the level shifter circuit on the “Radar Baseboard MCU7 Plus”.

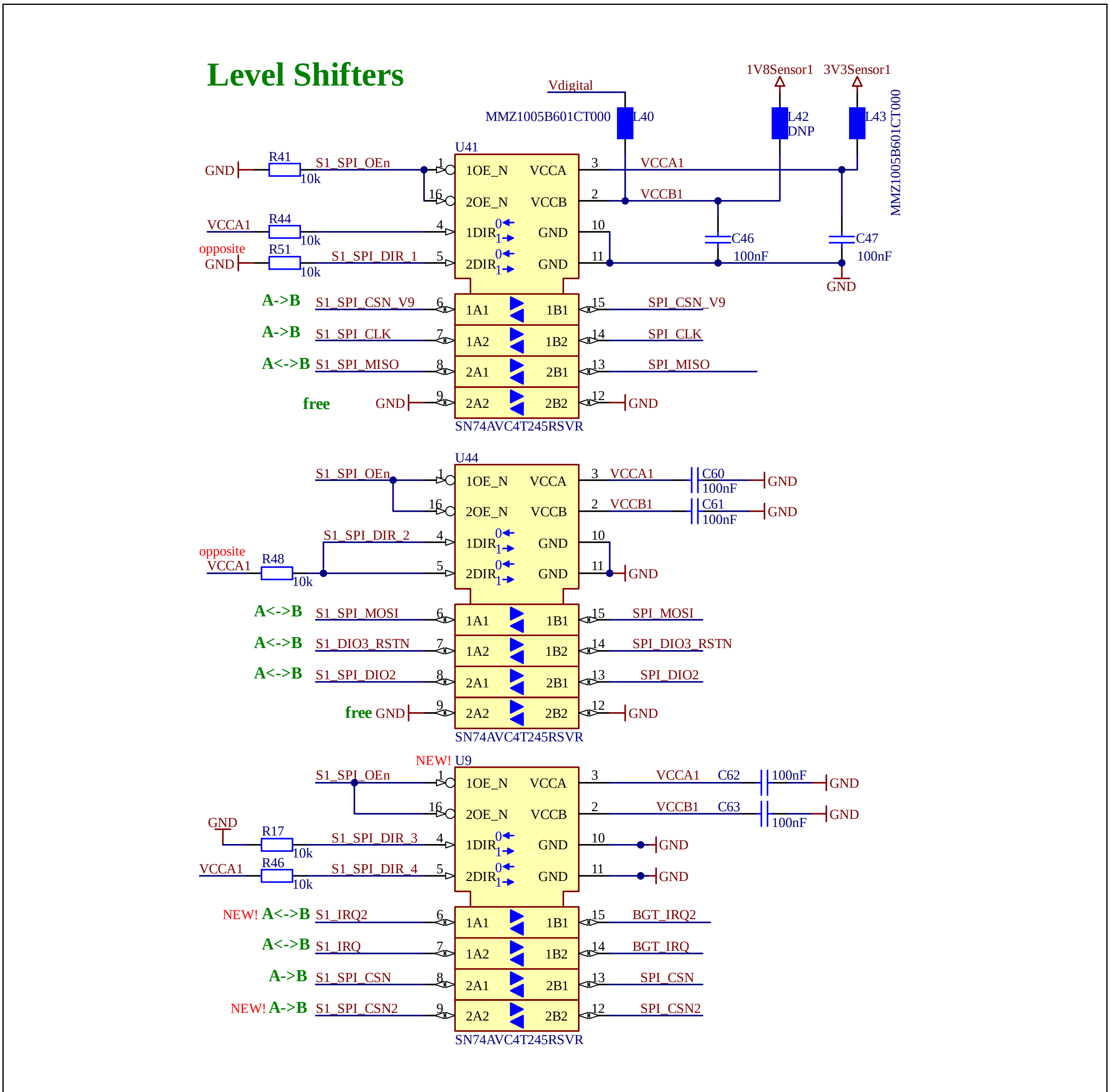

Figure 6 Level shifter circuit of sensor

Figure 6 illustrates the level shifter circuit for sensors. The level shifters can either be powered by one of the sensor supplies or with a voltage from the sensor interface. With this voltage from the sensor interface, the daughter card has the option to choose its preferred supply voltage. To minimize interference from digital signals on the supply voltage, the level shifters are decoupled from the sensor supplies. This is done by using ferrite beads with high suppression in the MHz regime.

A pull-up and pull-down resistors are connected to the enable and direction pins. This provides a pre-set so that SPI level translation is enabled with the direction configured so that the MCU is the master, and the sensor is the slave. If a different configuration is required, the MCU can directly drive those pins. See Table 5 and Table 6 for details.

2.4 Sensor connectors

As illustrated in Figure 7, the board features two sensor shield interfaces, labeled as “SC1” and “SC2” respectively. Next to the label, a white filled circle is drawn as a marker. A complementary marker is to be found on the sensor shield – see Figure 1. For correct mounting of a sensor, the markers of the “Radar Baseboard MCU7 Plus” and the sensor shield have to be aligned.

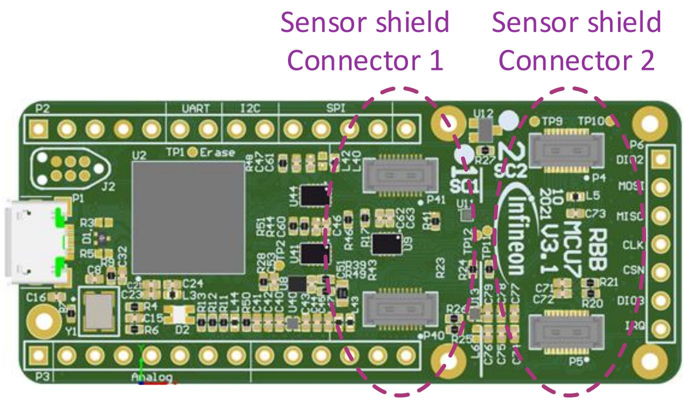

Figure 7 Sensor interfaces on the “Radar Baseboard MCU7 Plus”

Hirose’s DF40C-20DS-0.4V connectors are used for both the sensor interfaces on the “Radar Baseboard MCU7 Plus”. On the sensor shield, matching DF40C-20DP-0.4V connectors are used. There is a risk of the connectors wearing out when regularly plugged into and unplugged from the sensor. To prevent this, do not lift the sensor on the short side out of the connector, as illustrated in Figure 8a. Instead, simply pull on the long side of the sensor, thereby tilting the short side, as shown in Figure 8b. This will significantly increase the lifetime of the connectors.

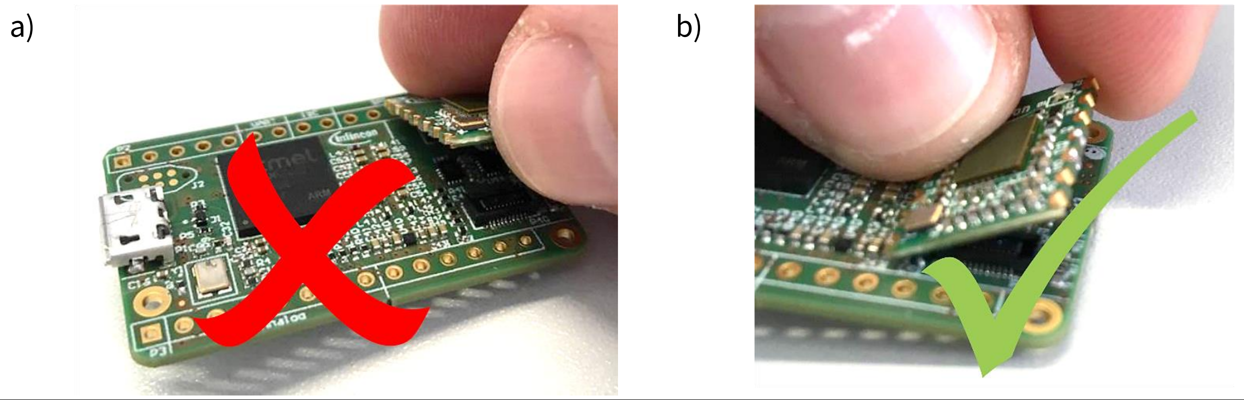

Figure 8 How to unplug the sensor-to-sensor connectors

Figure 9 depicts the schematics of the sensor shield connectors. The two connectors of the sensor shield interface 1 are separated by 12 mm, while the connectors’ separation of the sensor shield interface 2 is 16.29 mm.

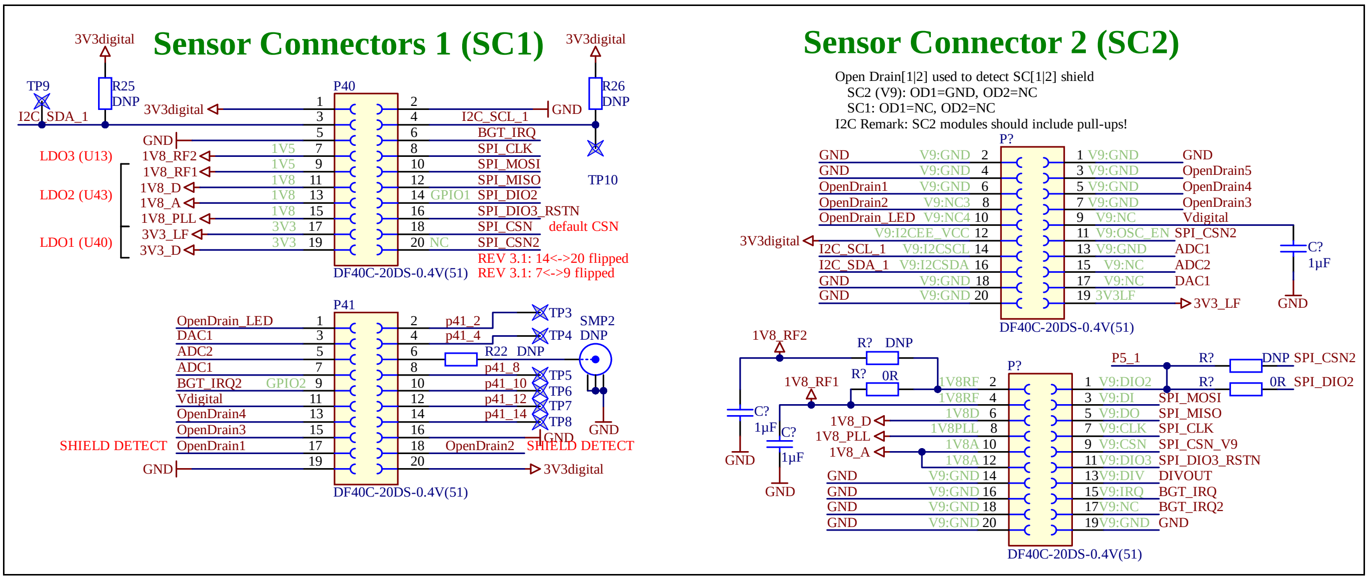

Figure 9 Schematics of the sensor connectors

If a sensor is plugged in incorrectly (rotated by 180 degrees), GND and 3V3digital will still be matched between the MCU board and sensor shield. Therefore, the sensor's I²C signal lines for the EEPROM will still be pulled up and the MCU will notice that OpenDrain1 and OpenDrain2 are pulled up instead of its I²C signals. This is how the MCU can detect that the sensor is plugged in correctly (I²C signals pulled up) or incorrectly (OpenDrain1 and OpenDrain2 pulled up).

The main sensor interface is full duplex SPI. Besides the ten level-shifted signals that were explained in Section 2.3, one line (Vdigital) provides a possible supply source for the level shifters, which allows the daughter shields to set their digital voltage level. Additionally, to these signals, there are three analog lines connected to the MCU, two ADC channels and a DAC channel, which can be used on the sensor shield. Furthermore, there are five digital signals connected, which can only be used in open-drain mode because they are not level shifted.

2.5 Current measurement of radar sensor shields

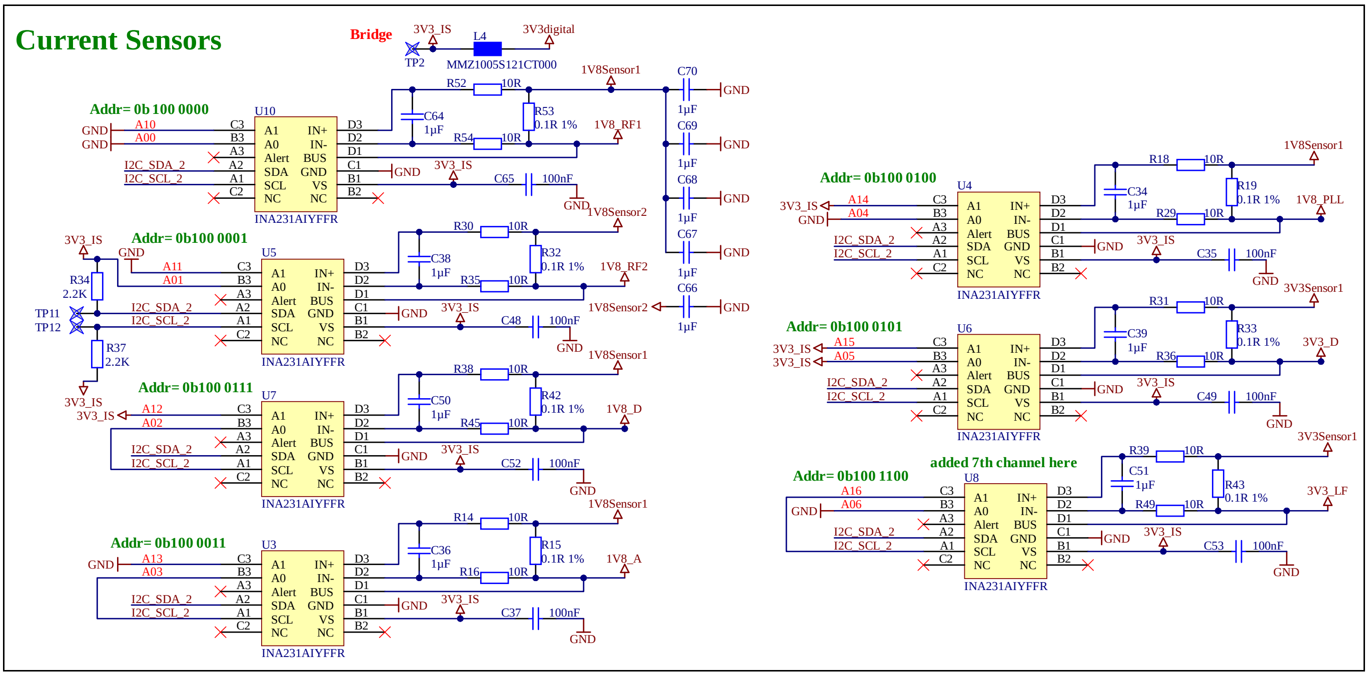

Figure 10 Schematic of the current sensors

There are seven current sensors to dynamically monitor the currents flowing into the daughter shields for each power supply domain, respectively. The detail schematic of the current sensors are shown in Figure 10. The 1.8 V power trace “1V8_RF2” comes from the dedicated LDO “1V8Sensor2”. All the current sensors are communicating with MCU through I^2C bus.

3 Arduino MKR interface

The Arduino MKR interface is compatible with Arduino MKR family of boards that can be used for rapid prototyping. This new board family contains (among others) the following boards, which can be used as daughter cards:

- MKR Zero – a board with an SD card slot which enables raw data storage on an SD card
- MKR1000 – a board with a Wi-Fi module for wireless data transfer
- MKR Vidor 4000 – a board with an FPGA for parallel data processing and SDRAM for a RAM extension
- MKR2UNO adapter – an adapter board that gives access to the standard Arduino environment with a huge number of daughter cards
- and many more; a quick Google search for “Arduino MKR boards overview” can provide a good summary.

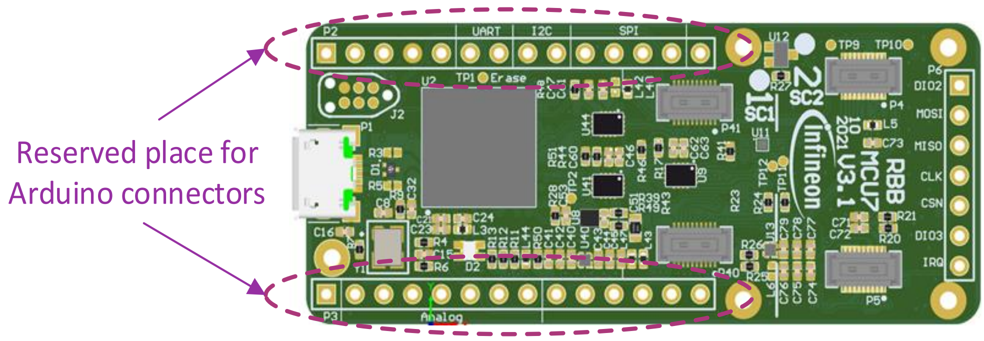

Figure 11 Arduino MKR connectors on the “Radar Baseboard MCU7 Plus”

Besides a 5 V supply, a 3.3 V supply and a voltage reference, the Arduino MKR interface contains both an analog and a digital interface. The analog interface features seven analog channels. The digital interface is composed of a UART interface, an SPI interface and seven GPIO pins – see Table 6 for details. Figure 11 shows the Arduino MKR connectors on the “Radar Baseboard MCU7 Plus”. If a sensor is mounted on the same side as the Arduino MKR connectors, the connectors might influence the radar antennas. For this reason, it is recommended to only solder Arduino MKR headers on the bottom side of the radar baseboard.

4 Firmware

4.1 Overview

The "Radar Baseboard MCU7 Plus" comes with a default firmware that is intended to serve as a bridge between a host (typically a PC) and the sensor shields, which are mounted on the sensor connectors. For this, the firmware implements logic to:

- communicate with the host via USB
- read and write sensor registers
- read sensor data via SPI
- check if a sensor shield is plugged into one of the connectors
- read and write the EEPROM on the sensor shield (for example, to read or write shield/application specific values)
- control some auxiliary peripherals such as status LEDs on the baseboard and sensor shield.

In its current state, the firmware does not implement any signal processing. It can receive messages from the host telling it how to configure the sensor, and it can forward the acquired data from the sensor to the host. However, if signal processing is required, this functionality can be added as the MCU is powerful enough. The firmware is written in a way to minimize the effort of porting it to other microcontrollers. Therefore, the firmware is structured into the following layers.

- **Hardware abstraction layer (HAL)** – This is the lowest layer and implements all the MCU-specific logic, such as how to interface with SPI, GPIOs, timers and other HW blocks. When porting to a different MCU, this layer should be the only one affected. It also implements a USB CDC serial interface that is used by the communication layer to connect to a host device.
- **Driver layer** – This layer sits on top of the hardware layer. It provides functions to parameterize and read data from the sensor. It creates appropriate SPI packets to apply those parameters to the sensor and calls the SPI functions from the HAL to initiate the data transfer. Furthermore, it decodes SPI data packets from the sensor received by the HAL. Thereby, the sensor measurements are extracted and passed on to the layer above. The driver layer is independent of the MCU. However, it is specific to the sensor and needs to be adapted for every supported sensor.
- **Communication layer** – This layer sits on top of the driver layer and the hardware layer. It receives request or command messages from the host via the USB interface. These messages are decoded, and the contained data is extracted. This is done by calling the corresponding functions of the driver layer. When those functions return with a result, a response message with the result is forwarded to the host. Messages to the host are also generated on receipt of sensor data from the driver layer. This layer is hardware independent, too. It uses the HAL USB CDC interface for data transmission to the host.
- **Actual firmware** – This layer sits on top of all other layers. The firmware calls the initialization functions of all lower layers at start-up for configuration of things like SPI speed and GPIO directions. Once the start-up phase has completed, an endless loop is entered, inside which the communication layer is called to fetch and transmit messages to and from the host.

When integrating a sensor in a custom project without the "Radar Baseboard MCU7 Plus" serving as an intermediary, the following steps will need to be taken in the firmware:

- The HAL needs to be ported to the MCU or application processor used in the project.
- The driver layer can be taken as is without any modifications.
- The communication layer will not be needed; instead, the algorithm can be put at this level.

- The Radar Baseboard MCU7 Plus's firmware code will not be used; however, the initialization code should be moved into the MCU of the project.

When porting the whole firmware to a different MCU, it should be sufficient to port the HAL to that MCU. When adding an algorithm to the firmware, this should be put above the driver layer. Depending on the requirements, the algorithm can replace the communication layer or be operated in parallel to it.

4.2 USB

The firmware implements a communication device class (CDC) device on the USB interface. That means the USB interface works like a serial port, implementing a bidirectional interface transmitting a stream of bytes in both directions.

On top of this serial data stream, a message-based protocol is implemented by putting start and end markers around each message. The first byte of the message specifies the target module of the firmware for which the message is intended. Bytes 2 and 3 of each message set the length of the message. That way, the protocol decoder knows the length of the message. In conjunction with the start and end markers, it is able to split the received stream into its constituent packets. The firmware source code includes a part of the communication library that can be integrated into a host application and provides functions to create and decode the firmware messages.

4.3 EEPROM

The EEPROM provides storage which may be used to contain shield/application specific data. For example

- calibration data for the antenna, or
- settings based on specific use cases.

4.4 RGB LED

The status of the board is indicated using the RGB LED. After startup of the board and while the board is idle, the board status is indicated as shown in Table 1. While active, the board status is indicated as shown in Table 2.

Table 1 Board status after startup and while idle as indicated by the RGB LED

Board status	Blinking pattern			
	Duration 1 and color 1	Duration 2 and color 2	Duration 3 and color 3	Duration 4 and color 4
No error detected	500 ms green	500 ms off		
No RF shield detected	500 ms green	500 ms red		
Wrong orientation of RF shield	500 ms red	500 ms off		
Unknown RF shield (shield may be broken or not connected properly)	500 ms yellow	500 ms red		
Internal error during initialization/detection (this may indicate a HW or FW problem)	250 ms yellow	250 ms off	500 ms red	500 ms off

Table 2 System status while active as indicated by the RGB LED

System status	Blinking pattern			
	Duration 1 and color 1	Duration 2 and color 2	Duration 3 and color 3	Duration 4 and color 4
Board is idle	500 ms green	500 ms off		
Reading data from SPI	Red			
Sending data to host	Blue			

4.5 Firmware development and debugging

The firmware is delivered as a project for the Atmel Studio 7 IDE. Hence, compiling, flashing and debugging work out-of-the-box by simply pressing the corresponding buttons in the IDE.

If the user requires a debugger connector, the board features two options, as illustrated in Figure 12. On the top side of the “Radar Baseboard MCU7 Plus”, there is the layout for a Tag Connect debugger. If the user does not want to buy a Tag Connect debug cable, they can solder a normal 10-pin Arm debugging connector to the pads of component J1 on the bottom side of the “Radar Baseboard MCU7 Plus”.

When using a debugger connector, the firmware can be programmed directly from Atmel Studio 7 using an Atmel-ICE programmer or compatible tools [1].

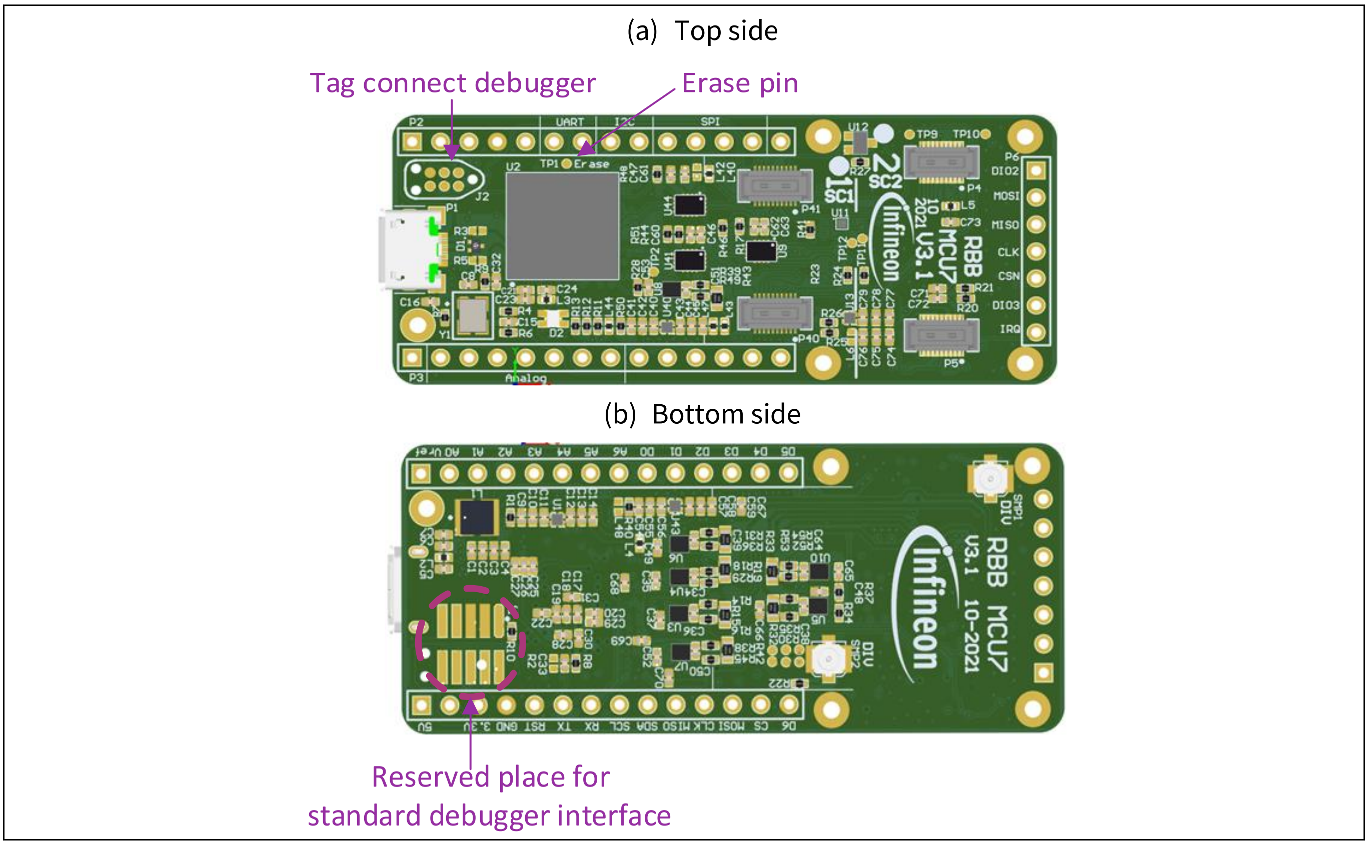

Figure 12 Debug connectors on the “Radar Baseboard MCU7 Plus”

4.6 Flashing via bootloader

The microcontroller contains a bootloader. This enables firmware updates via the USB interface if a debugger is not available. Since the device normally boots directly into the firmware without going into the bootloader must be manually activated at first. This can be done by performing the following sequence:

- Disconnect the device from USB.
- Hold the MCU “ERASE” pin to high by connecting TP1 to 3.3 V (see Figure 12).
- Connect the device to USB again while holding TP1 high.
- Release the “ERASE” pin (TP1).
- The device is now in bootloader mode.

Once the device is in bootloader mode, the firmware can be programmed using the SAM-BA GUI [2].

5 Microcontroller pin map

5.1 Sensor interface 1

The RF shield will force back the Vdigital towards the Radar Baseboard MCU7 Plus. Depending on the MMIC IO signal levels it can have voltage levels, from 1.5V up-to 3.3 V. The RF shield must take care of the Vdigital signal levels.

Table 3 Microcontroller pin map of sensor interface 1

Signal name(s)	Signal group	MCU pin	Description	Supply domain
I2C_SCL_1	I^2C	PD28	Clock pin of I^2C interface for the EEPROM	3.3 V
I2C_SDA_1	I^2C	PD27	Data pin of I^2C interface for the EEPROM	3.3 V
S1_SPI_CSN SPI_CSN	SPI	PA11	Chip select pin of SPI sensor interface	Vdigital
S1_SPI_CSN2 SPI_CSN_2	SPI	PA9	2nd sensor on RF shield support	Vdigital
S1_SPI_MISO SPI_MISO	SPI	PA12	Master Input Slave Output pin of SPI sensor interface	Vdigital
S1_SPI_MOSI SPI_MOSI	SPI	PA13	Master Output Slave Input pin of SPI sensor interface	Vdigital
S1_SPI_CLK SPI_CLK	SPI	PA14	Clock pin of SPI sensor interface	Vdigital
S1_SPI_DIO2 SPI_DIO2	SPI	PA17	QSPI Slave Input/Output pin of SPI sensor interface	Vdigital
S1_DIO3_RSTN SPI_DIO3_RSTN	SPI	PD31	Hardware reset + QSPI Slave Input/Output pin of SPI sensor interface	Vdigital
S1_IRQ BGT_IRQ	GPIO	PC6	Level shifted GPIO, used for interrupt request of sensor	Vdigital
S1_IRQ2 BGT_IRQ2	GPIO	PA25	Level shifted GPIO, used for interrupt request of 2nd sensor	Vdigital
ADC1	Analog	PE5	ADC channel 1, analog front end 0 of the MCU	3.3 V
ADC2	Analog	PC15	ADC channel 2, analog front end 1 of the MCU	3.3 V
DAC1	Analog	PB13	DAC channel, DAC0 of MCU	3.3 V
OpenDrain1	Open drain	PD16	Not level shifted GPIO, either 3.3 V push-pull or open drain, shield detection	3.3 V
OpenDrain2	Open drain	PC9	Not level shifted GPIO, either 3.3 V push-pull or open drain, shield detection	3.3 V
OpenDrain3	Open drain	PC28	Not level shifted GPIO, either 3.3 V push-pull or open drain	3.3 V
OpenDrain4	Open drain	PA27	Not level shifted GPIO, either 3.3 V push-pull or open drain	3.3 V

Signal name(s)	Signal group	MCU pin	Description	Supply domain
OpenDrain_LED	Open drain	PC1	Not level shifted GPIO, either 3.3 V push-pull or open drain	3.3 V
S1_SPI_OEn	Level shifter	PD24	Disable pin of the level shifter for SPI communication – with external pull-down resistor (high: disable, low: enable)	3.3 V
S1_SPI_DIR1	Level shifter	PD11	Direction pin of the level shifter for SPI communication (MISO) – with external pull-down resistor (high: MCU → sensor, low: sensor → MCU)	3.3 V
S1_SPI_DIR2	Level shifter	PD18	Direction pin of the level shifter for SPI communication (MOSI, DIO2,3) – with external pull-up resistor (high: MCU → sensor, low: sensor → MCU)	3.3 V
S1_SPI_DIR3	Level shifter	PD17	Direction pin of the level shifter for SPI communication (BGT_IRQ 1,2) – with external pull-down resistor (high: MCU → sensor, low: sensor → MCU)	3.3 V
S1_SPI_DIR4	Level shifter	PA10	Direction pin of the level shifter for SPI communication (SPI_CSN 1,2) – with external pull-up resistor (high: MCU → sensor, low: sensor → MCU)	3.3 V

5.2 Sensor interface 2

Table 4 Microcontroller pin map of sensor interface 2

Signal name(s)	Signal group	MCU pin	Description	Supply domain
I2C_SCL_1	I^2C	PD28	Clock pin of I^2C interface for the EEPROM	3.3 V
I2C_SDA_1	I^2C	PD27	Data pin of I^2C interface for the EEPROM	3.3 V
S1_SPI_CSN_V9 SPI_CSN_V9	SPI	PA7	Chip select pin of SPI sensor interface	Vdigital
S2_SPI_CSN2 SPI_CSN_2	SPI	PA9	2nd sensor on RF shield support	Vdigital
S2_SPI_MISO SPI_MISO	SPI	PA12	Master Input Slave Output pin of SPI sensor interface	Vdigital
S2_SPI_MOSI SPI_MOSI	SPI	PA13	Master Output Slave Input pin of SPI sensor interface	Vdigital
S2_SPI_CLK SPI_CLK	SPI	PA14	Clock pin of SPI sensor interface	Vdigital
S1_SPI_DIO2 SPI_DIO2	SPI	PA17	QSPI Slave Input/Output pin of SPI sensor interface	Vdigital
S1_DIO3_RSTN SPI_DIO3_RSTN	SPI	PD31	Hardware reset + QSPI Slave Input/Output pin of SPI sensor interface	Vdigital
S1_IRQ	GPIO	PC6	Level shifted GPIO, used for interrupt request of sensor	Vdigital

Signal name(s)	Signal group	MCU pin	Description	Supply domain
BGT_IRQ				
S1_IRQ2 BGT_IRQ2	GPIO	PA25	Level shifted GPIO, used for interrupt request of 2nd sensor	Vdigital
ADC1	Analog	PE5	ADC channel 1, analog front end 0 of the MCU	3.3 V
ADC2	Analog	PC15	ADC channel 2, analog front end 1 of the MCU	3.3 V
DAC1	Analog	PB13	DAC channel, DAC0 of MCU	3.3 V
OpenDrain1	Open drain	PD16	Not level shifted GPIO, either 3.3 V push-pull or open drain, shield detection	3.3 V
OpenDrain2	Open drain	PC9	Not level shifted GPIO, either 3.3 V push-pull or open drain, shield detection	3.3 V
OpenDrain3	Open drain	PC28	Not level shifted GPIO, either 3.3 V push-pull or open drain	3.3 V
OpenDrain4	Open drain	PA27	Not level shifted GPIO, either 3.3 V push-pull or open drain	3.3 V
OpenDrain5	Open drain	PD14	Not level shifted GPIO, either 3.3 V push-pull or open drain	3.3 V
OpenDrain_LED	Open drain	PC1	Not level shifted GPIO, either 3.3 V push-pull or open drain	3.3 V
S1_SPI_OEn	Level shifter	PD24	Disable pin of the level shifter for SPI communication – with external pull-down resistor (high: disable, low: enable)	3.3 V
S1_SPI_DIR1	Level shifter	PD11	Direction pin of the level shifter for SPI communication (MISO) – with external pull-down resistor (high: MCU → sensor, low: sensor → MCU)	3.3 V
S1_SPI_DIR2	Level shifter	PD18	Direction pin of the level shifter for SPI communication (MOSI, DIO2,3) – with external pull-up resistor (high: MCU → sensor, low: sensor → MCU)	3.3 V
S1_SPI_DIR3	Level shifter	PD17	Direction pin of the level shifter for SPI communication (BGT_IRQ1,2) – with external pull-down resistor (high: MCU → sensor, low: sensor → MCU)	3.3 V
S1_SPI_DIR4	Level shifter	PA10	Direction pin of the level shifter for SPI communication (SPI_CSN1,2) – with external pull-up resistor (high: MCU → sensor, low: sensor → MCU)	3.3 V

Table 5 P6 Jumper pin connections to sensor interface 2

Connector	Signal name	MCU pin	Description	Supply domain
P6.1	DIO2	PD28	QSPI Slave Input/Output pin of SPI sensor interface	Vdigital
P6.2	MOSI	PA13	Master Output Slave Input pin of SPI sensor interface	Vdigital
P6.3	MISO	PA12	Master Input Slave Output pin of SPI sensor interface	Vdigital

Connector	Signal name	MCU pin	Description	Supply domain
P6.4	CLK	PA14	Clock pin of SPI sensor interface	Vdigital
P6.5	CSN	PA9	2nd sensor on RF shield support	Vdigital
P6.6	DIO3	PD31	HW reset + QSPI Slave Input/Output pin of SPI sensor interface	Vdigital
P6	IRQ	PA25	Level shifted GPIO, used for interrupt request of the 1st sensor	Vdigital

Note: *In case if P6 shall be used with customer module, the sensor connector 2 (SC2) requires some signals be set to the right level.*
1. Enable the level shifters on the MCU board: OpenDrain5 is set to GND.
2. Enable Shield2 detection: OpenDrain1 = OpenDrain2 = GND
This can be achieved by, e.g., placing a "dummy RF shield on SC2" - which does not have the MMIC soldered + customer Module on P6.
After USB-Plug-in, the customer module shall be detected well.

5.3 Arduino MKR interface

Table 6 Microcontroller pin map of the Arduino MKR interface

Connector	Signal name	MCU pin	Description	Supply domain
P2.1	5Vusb	–	External 5 V supply	5V
P2.2		–		
P2.3	3V3digital	VDDIO	External 3.3 V supply	3.3 V
P2.4	GND	–	Ground line	
P2.5	N_RST	NRST	Inverted hardware reset pin of the MCU – with external pull-up resistor	3.3 V
P2.6	UART_TX	PD26	GPIO, MCU's UART Tx-pin	3.3 V
P2.7	UART_RX	PD25	GPIO, MCU's UART Rx-pin	3.3 V
P2.8	I2C_SCL_2	PA4	GPIO, clock pin of I^2C interface – shared with sensor interface 2 for the EEPROM	3.3 V
P2.9	I2C_SDA_2	PA3	GPIO, data pin of I^2C interface – shared with sensor interface 2 for the EEPROM	3.3 V
P2.10	SPI0_MISO	PD20	GPIO, Master Input Slave Output pin of SPI sensor interface	3.3 V
P2.11	SPI0_CLK	PD22	GPIO, clock pin of SPI sensor interface	3.3 V
P2.12	SPI0_MOSI	PD21	GPIO, Master Output Slave Input pin of SPI sensor interface	3.3 V
P2.13	SPI0_CSN	PD12	GPIO, chip select pin of SPI sensor interface	3.3 V
P2.14	PWM6	PA5	GPIO, PWM channel 1 – low side 3	3.3 V
P3.1	Vref	VREF	Reference voltage for analog front ends of the MCU	
P3.2	A0	PC0	GPIO, ADC channel 0, analog front end 1 of the MCU	3.3 V

Connector	Signal name	MCU pin	Description	Supply domain
P3.3	A1	PC12	GPIO, ADC channel 1, analog front end 1 of the MCU	3.3 V
P3.4	A2	PE0	GPIO, ADC channel 2, analog front end 1 of the MCU	3.3 V
P3.5	A3	PA19	GPIO, ADC channel 3, analog front end 0 of the MCU	3.3 V
P3.6	A4	PB3	GPIO, ADC channel 4, analog front end 0 of the MCU	3.3 V
P3.7	A5	PE4	GPIO, ADC channel 5, analog front end 0 of the MCU	3.3 V
P3.8	A6	PA21	GPIO, ADC channel 6, analog front end 0 of the MCU	3.3 V
P3.9	PWM0	PC22	GPIO, PWM channel 0 – low side 3	3.3 V
P3.10	PWM1	PC21	GPIO, PWM channel 0 – high side 3	3.3 V
P3.11	PWM2	PC20	GPIO, PWM channel 0 – low side 2	3.3 V
P3.12	PWM3	PC19	GPIO, PWM channel 0 – high side 2	3.3 V
P3.13	PWM4	PC18	GPIO, PWM channel 0 – low side 1	3.3 V
P3.14	PWM5	PA23	GPIO, PWM channel 0 – high side 0	3.3 V

5.4 Other pins

Table 7 Microcontroller pin map of other pins

Signal group	Signal name	MCU pin	Description	Supply domain
RGB LED	PWMC1_L0_red	PD3	Red diode of the RGB LED, PWM channel 1 – low side 0	3.3 V
RGB LED	PWMC1_L1_green	PD5	Red diode of the RGB LED, PWM channel 1 – low side 0	3.3 V
RGB LED	PWMC1_L2_blue	PD7	Red diode of the RGB LED, PWM channel 1 – low side 0	3.3 V
LDO	en_LDO_1	PC30	Enable of LDOs for sensor interface	3.3 V

References

[1] Microchip. (2018, 05). Atmel Studio 7 User Guide. https://www.microchip.com/content/dam/mchp/documents/atmel-start/Atmel-Studio-7-User-Guide.pdf

[2] Microchip. (2020, 12 20). SAM-BA Boot Assistant. https://www.microchip.com/en-us/development-tool/SAM-BA-In-system-Programmer

[3] Infineon Technologies AG. AN599: Radar Baseboard MCU7

Revision history

Document revision	Date	Description of changes
1.00	2023-06-06	Initial version

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2023-06-06

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

AN_2212_PL32_2301_155750