AVR64EA48

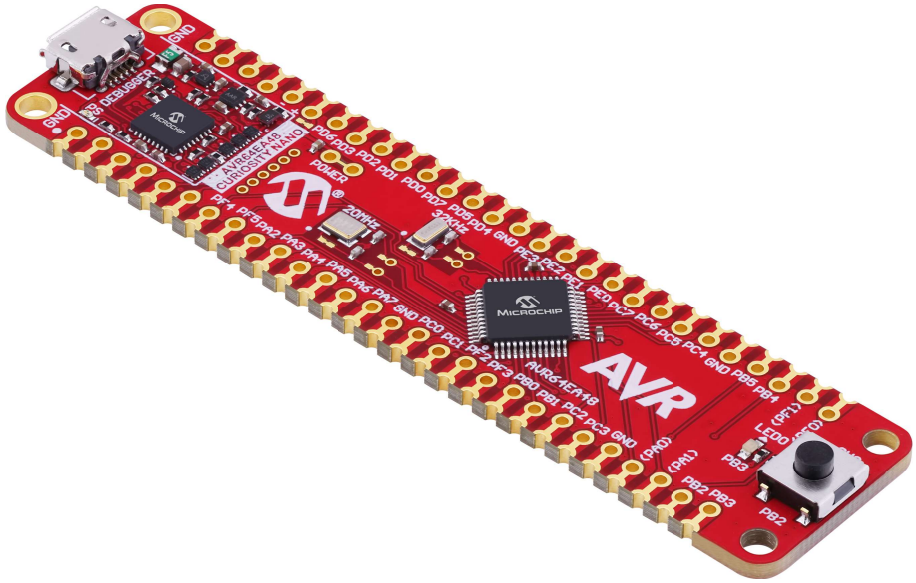
AVR64EA48 Curiosity Nano Hardware User Guide

Preface

The AVR64EA48 Curiosity Nano evaluation kit (EV66E56A) is a hardware platform to evaluate the AVR® EA Family microcontrollers. This board has the AVR64EA48 microcontroller (MCU) mounted.

Supported by both MPLAB® X IDE and Microchip Studio, the board provides easy access to the features of the AVR64EA48 to explore how to integrate the device into a custom design.

The Curiosity Nano series of evaluation boards include an on-board debugger. No external tools are necessary to program and debug the AVR64EA48.



- **MPLAB® X IDE and Microchip Studio** - Software to discover, configure, develop, program, and debug Microchip microcontrollers.
- **Code examples on MPLAB Discover** - Get started with code examples.
- **AVR64EA48 website** - Find documentation, data sheets, sample, and purchase microcontrollers.
- **AVR64EA48 Curiosity Nano website** - Kit information, latest user guide and design documentation.

Table of Contents

Preface....1

1. Introduction....3
   - 1.1. Features....3
   - 1.2. Board Overview....3

2. Getting Started....4
   - 2.1. Curiosity Nano Quick Start MPLAB® Xpress....4
   - 2.2. Quick Start....4
   - 2.3. Design Documentation and Relevant Links....5

3. Curiosity Nano....7
   - 3.1. On-Board Debugger Overview....7
   - 3.2. Curiosity Nano Standard Pinout....14
   - 3.3. Power Supply....14
   - 3.4. Low-Power Measurement....18
   - 3.5. Programming External Microcontrollers....19
   - 3.6. Connecting External Debuggers....22

4. Hardware Description....25
   - 4.1. Connectors....25
   - 4.2. Peripherals....26

5. Hardware Revision History and Known Issues....30
   - 5.1. Identifying Product ID and Revision....30
   - 5.2. Revision 3....30

6. Document Revision History....31

7. Appendix....32
   - 7.1. Schematic....33
   - 7.2. Assembly Drawing....35
   - 7.3. Curiosity Nano Base for Click boards™....36
   - 7.4. Disconnecting the On-Board Debugger....37
   - 7.5. Getting Started with IAR™....38

Microchip Information....41
- The Microchip Website....41
- Product Change Notification Service....41
- Customer Support....41
- Microchip Devices Code Protection Feature....41
- Legal Notice....41
- Trademarks....42
- Quality Management System....43
- Worldwide Sales and Service....44

1. Introduction

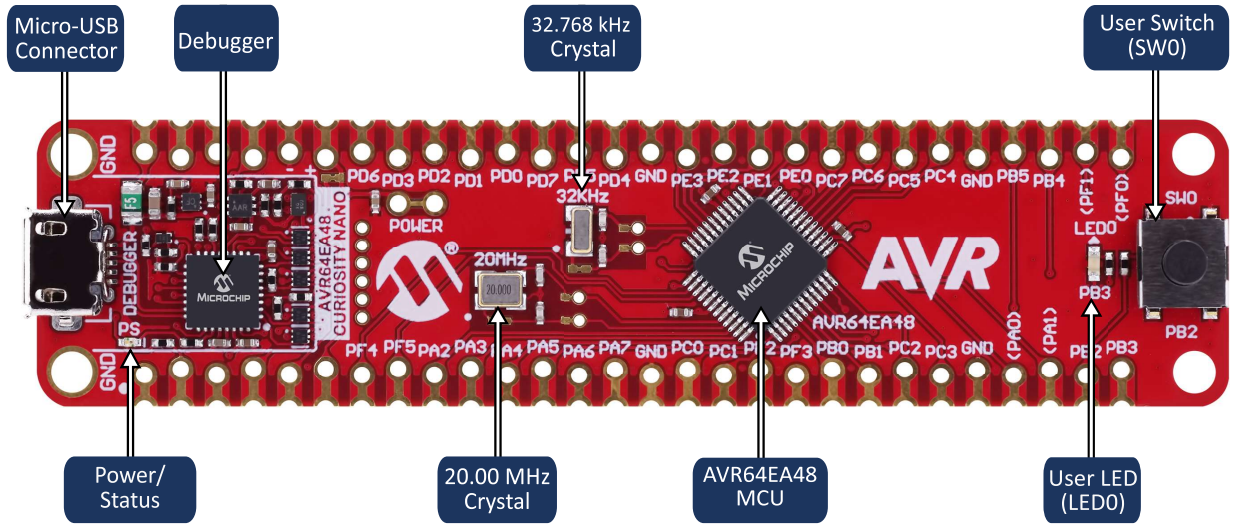
1.1 Features

- AVR64EA48 Microcontroller
- One Yellow User LED
- One Mechanical User Switch
- One 32.768 kHz Crystal
- One 20 MHz Crystal
- On-Board Debugger:
  - Board identification in Microchip MPLAB® X IDE and Microchip Studio
  - One green power and status LED
  - Programming and debugging
  - Virtual serial port (CDC)
  - Two debug GPIO channels (DGI GPIO)
- USB Powered
- Adjustable Target Voltage:
  - MIC5353 LDO regulator controlled by the on-board debugger
  - 1.8–5.1V output voltage (limited by USB input voltage)
  - 500 mA maximum output current (limited by ambient temperature and output voltage)

1.2 Board Overview

The Microchip AVR64EA48 Curiosity Nano evaluation kit is a hardware platform to evaluate the AVR64EA48 microcontroller.

Figure 1-1. AVR64EA48 Curiosity Nano Board Overview



2. Getting Started

2.1 Curiosity Nano Quick Start MPLAB® Xpress

MPLAB Xpress Cloud-Based IDE is part of the MPLAB Cloud tools ecosystem, leveraging the intuitive MPLAB Discover for finding projects and code examples and the MPLAB Code Configurator graphical configuration tool to provide an all-in cloud experience.

Steps to start exploring the Curiosity Nano platform with MPLAB Xpress:

1. Go to www.mplab-xpresside.microchip.com/ to open MPLAB Xpress.
2. Create a new standalone project for AVR64EA48.
3. Use the MPLAB Xpress Code Configurator, or write your code.
4. Compile and download the application HEX file.
5. Connect a USB cable (Standard-A to Micro-B or Micro-AB) between the PC and the debug USB port on the board.
6. Copy the application HEX file into the *CURIOSITY* mass storage drive to program the application into the AVR64EA48.

   To use the advanced debug features of the Curiosity Nano kit, package the MPLAB Xpress project for MPLAB X IDE, and follow the quick start guide in the next section.

2.2 Quick Start

Steps to start exploring the AVR64EA48 Curiosity Nano board:

1. Download Microchip MPLAB® X IDE and MPLAB® XC C Compiler, or download Microchip Studio.
2. Launch MPLAB® X IDE or Microchip Studio.
3. Optional: Use MPLAB® Code Configurator to generate drivers and examples.
4. Write\Develop the application code.
5. Connect a USB cable (Standard-A to Micro-B or Micro-AB) between the PC and the debug USB port on the board.
6. Program your application onto the device.

   The AVR64EA48 device on the AVR64EA48 Curiosity Nano board is programmed and debugged by the on-board debugger. Therefore, no external programmer or debugger tool is required.

Info:

- MPLAB® X IDE supports XC C compilers and the AVR® and Arm® Toolchains (GCC Compilers)
- Microchip's XC8 C Compiler supports all 8-bit PIC® and AVR® microcontrollers
- Included in the Microchip Studio download are the AVR® and Arm® Toolchains (GCC Compilers)

2.2.1 Driver Installation

When the board connects to the computer for the first time, the operating system will perform a driver software installation. The driver file supports both 32- and 64-bit versions of Microsoft® Windows®. The drivers for the board are included with both MPLAB® X IDE and Microchip Studio.

2.2.2 Kit Window

When the board is connected to a computer and powered on, the green status LED will be lit, and both MPLAB® X IDE and Microchip Studio will auto-detect which boards are connected. The Kit Window in MPLAB® X IDE and Microchip Studio will present relevant information like data sheets and board documentation.

Tip: If closed, reopen the Kit Window in MPLAB® X IDE through the menu bar **Window** > **Kit Window**.

2.2.3 MPLAB® X IDE Device Family Packs

Microchip MPLAB® X IDE requires specific information to support devices and tools. This information is contained in versioned packs. For the AVR64EA48 Curiosity Nano board, MPLAB® X version 6.0 with device family pack “AVR-Ex_DFP” version 2.1.48 and tool pack “nEDBG_TP” version 1.10.488 or newer is required. For more information on packs and how to upgrade them, refer to the MPLAB® X IDE User’s guide - Work with Device Packs.

Tip: The latest device family packs are available through **Tools** > **Packs** in MPLAB® X IDE or online at Microchip MPLAB® X Packs Repository.

2.2.4 Microchip Studio Device Family Pack

Microchip Studio requires specific information to support devices. This information is contained in versioned packs. For the AVR64EA48 Curiosity Nano board, the device family pack “AVR-Ex_DFP” version 2.2.56 or newer is required. For more information on packs and how to upgrade them, refer to the Microchip Studio User Guide - Device Pack Manager.

Tip: The latest device family packs are available through **Tools** > **Device Pack Manager** in Microchip Studio or online at Microchip Studio Packs Repository.

2.3 Design Documentation and Relevant Links

The following list contains links to the most relevant documents and software for the AVR64EA48 Curiosity Nano board:

- **MPLAB® X IDE** - MPLAB X IDE is a software program that runs on a PC (Windows®, Mac OS®, Linux®) to develop applications for Microchip microcontrollers and digital signal controllers. It is named an Integrated Development Environment (IDE) because it provides a single integrated “environment” to develop code for embedded microcontrollers.
- **Microchip Studio** - Free IDE for the development of C/C++ and assembler code for microcontrollers.
- **IAR Embedded Workbench® for AVR®** - This is a commercial C/C++ compiler available for AVR microcontrollers. There is a 30-day evaluation version and a 4 KB code-size-limited kick-start version available from their website.
- **MPLAB® XC Compilers** - MPLAB® XC8 C Compiler is available as a free, unrestricted-use download. Microchips MPLAB® XC8 C Compiler is a comprehensive solution for the project’s software development on Windows®, macOS® or Linux®. MPLAB® XC8 supports all 8-bit PIC® and AVR® microcontrollers (MCUs).
- **MPLAB® Xpress Cloud-Based IDE** - MPLAB Xpress Cloud-Based IDE is an online development environment containing the most popular features of our award-winning MPLAB X IDE. This simplified and distilled application is a faithful reproduction of our desktop-based program, allowing users an easy transition between the two environments.

- **MPLAB® Code Configurator** - MPLAB Code Configurator (MCC) is a free software plug-in that provides a graphical interface to configure peripherals and functions specific to your application.
- **Microchip Sample Store** - Microchip sample store where one can order samples of devices.
- **MPLAB Data Visualizer** - MPLAB Data Visualizer is a program used for processing and visualizing data. The Data Visualizer can receive data from various sources, such as serial ports and the on-board debugger's Data Gateway Interface, as found on Curiosity Nano and Xplained Pro boards.
- **Atmel Data Visualizer** - Studio Data Visualizer is a program used for processing and visualizing data. The Data Visualizer can receive data from various sources such as serial ports, the on-board debugger's Data Gateway Interface found on Curiosity Nano and Xplained Pro boards, and power data from the Power Debugger.
- **MPLAB Discover** - MPLAB Discover is a tool to help you find Microchip example projects and collateral for Microchip devices.
- **AVR64EA48 Curiosity Nano website** - Kit information, latest user guide and design documentation.
- **AVR64EA48 Curiosity Nano on Microchip Direct** - Purchase this kit on Microchip Direct.

3. Curiosity Nano

Curiosity Nano is an evaluation platform of small boards with low pin count microcontroller (MCU) boards with on-board debuggers and access to most microcontrollers I/Os. The Curiosity Nano platform offers easy integration with MPLAB® X IDE and Microchip Studio. All boards are identified in the IDE. When connected, a Kit Window appears with links to key documentation, including relevant user guides, application notes, data sheets, and example code. Everything is easy to find. The on-board debugger features a virtual serial port (CDC) for serial communication to a host PC and a Data Gateway Interface (DGI) with debug GPIO pin(s).

3.1 On-Board Debugger Overview

AVR64EA48 Curiosity Nano contains an on-board debugger for programming and debugging. The on-board debugger is a composite USB device consisting of several interfaces:

- A debugger that can program and debug the AVR64EA48 in both MPLAB® X IDE and Microchip Studio
- A mass storage device that allows drag-and-drop programming of the AVR64EA48
- A virtual serial port (CDC) that is connected to a Universal Asynchronous Receiver/Transmitter (UART) on the AVR64EA48 and provides an easy way to communicate with the target application through terminal software
- A Data Gateway Interface (DGI) for code instrumentation with logic analyzer channels (debug GPIO) to visualize program flow

The on-board debugger controls a Power and Status LED (marked PS) on the AVR64EA48 Curiosity Nano board. The table below shows how the different operation modes control the LED.

Table 3-1. On-Board Debugger LED Control

Operation Mode	Power and Status LED
Boot Loader mode	The LED blinks slowly during power-up
Power-up	The LED is ON
Normal operation	The LED is ON
Programming	Activity indicator: The LED blinks slowly during programming/debugging
Drag-and-drop programming	**Success:** The LED blinks slowly for 2 sec. **Failure:** The LED blinks rapidly for 2 sec.
Fault	The LED blinks rapidly if a power fault is detected
Sleep/Off	The LED is OFF. The on-board debugger is either in a sleep mode or powered down. This can occur if the board is externally powered.

Info: Slow blinking is approximately 1 Hz, and rapid blinking is about 5 Hz.

3.1.1 Debugger

The on-board debugger on the AVR64EA48 Curiosity Nano board appears as a Human Interface Device (HID) on the host computer's USB subsystem. The debugger supports full-featured programming and debugging of the AVR64EA48 by using both MPLAB X IDE and Microchip Studio and some third-party IDEs.

Remember: Keep the debugger's firmware up-to-date. Firmware upgrades automatically when using MPLAB X IDE or Microchip Studio.

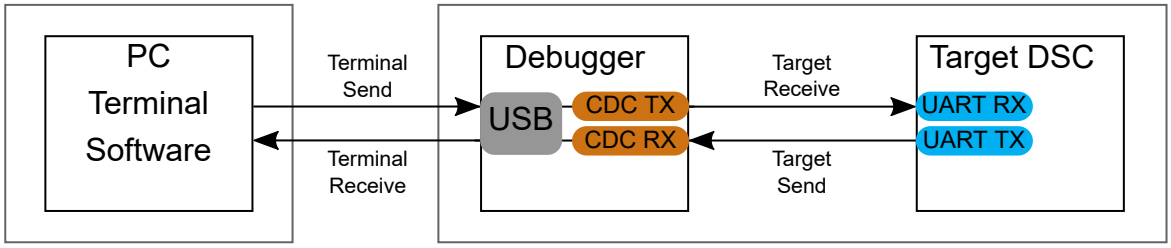
3.1.2 Virtual Serial Port (CDC)

The virtual serial port (CDC) is a general purpose serial bridge between a host PC and a target device.

3.1.2.1 Overview

The on-board debugger implements a composite USB device with a standard Communications Device Class (CDC) interface, which appears on the host as a virtual serial port. Use the CDC to stream arbitrary data between the host computer and the target in both directions: All characters sent through the virtual serial port on the host computer will be transmitted as UART on the debugger's CDC TX pin. The UART characters captured on the debugger's CDC RX pin will be returned to the host computer through the virtual serial port.

Figure 3-1. CDC Connection



Info: The debugger's CDC TX pin is connected to a UART RX pin on the target for receiving characters from the host computer, as shown in the figure above. Similarly, the debugger's CDC RX pin is connected to a UART TX pin on the target for transmitting characters to the host computer.

3.1.2.2 Operating System Support

On Windows® machines, the CDC will enumerate as *Curiosity Virtual COM Port* and appear in the Ports section of the Windows Device Manager. The COM port number can also be found there.

Info: On older Windows systems, the CDC requires a USB driver. Both MPLAB X IDE and Microchip Studio installationsinclude this driver.

On Linux® machines, the CDC will enumerate and appear as `/dev/ttyACM#`.

Info: tty* devices belong to the "dialout" group in Linux, so it may be necessary to become a member of that group to have permission to access the CDC.

On Mac® machines, the CDC will enumerate and appear as `/dev/tty.usbmodem#`. Depending on the terminal program used, it will appear in the available list of modems as `usbmodem#`.

Info: For all operating systems, use a terminal emulator that supports DTR signaling. See 3.1.2.4. Signaling.

3.1.2.3 Limitations

Not all UART features are implemented in the on-board debugger CDC. The constraints are outlined here:

- **Baud rate**: Must be in the range of 1200 bps to 500 kbps. Any baud rate outside this range will be set to the closest limit without warning. Baud rate can be changed on-the-fly.
- **Character format**: Only 8-bit characters are supported.
- **Parity**: Can be odd, even, or none.
- **Hardware flow control**: Not supported.
- **Stop bits**: One or two bits are supported.

3.1.2.4 Signaling

During USB enumeration, the host OS will start both the communication and data pipes of the CDC interface. At this point, it is possible to set and read back the baud rate and other UART parameters of the CDC, but data sending and receiving will not be enabled.

The terminal must assert the DTR signal when it connects to the host. As this is a virtual control signal implemented on the USB interface, it is not physically present on the board. Asserting the DTR signal from the host will indicate to the on-board debugger that a CDC session is active. The debugger will enable its level shifters (if available) and start the CDC data send and receive mechanisms.

Deasserting DTR in debugger firmware version 1.20 or earlier has the following behavior:

- Debugger UART receiver is disabled, and no further data will be transferred to the host computer
- Debugger UART transmitter will continue to send queued data ready for transfer, but no new data is accepted from the host computer
- Level shifters (if available) are not disabled, and the debugger CDC TX line remains driven

Deasserting DTR in debugger firmware version 1.21 or later has the following behavior:

- Debugger UART receiver is disabled, and no further data will be transferred to the host computer
- Debugger UART transmitter will continue to send queued data ready for transfer, but no new data is accepted from the host computer
- Once the ongoing transmission is complete, level shifters (if available) are disabled, and the debugger CDC TX line will become high-impedance

Remember: Set up the terminal emulator to assert the DTR signal. Without the signal, the on-board debugger will not send or receive data through its UART.

Tip: The on-board debugger's CDC TX pin will not be driven until the CDC interface is enabled by the host computer. Also, there are no external pull-up resistors on the CDC lines connecting the debugger and the target, which means that the lines are floating during power-up. The target device may enable the internal pull-up resistor on the pin connected to the debugger's CDC TX pin to avoid glitches resulting in unpredictable behavior like framing errors.

3.1.2.5 Advanced Use

CDC Override Mode
In ordinary operation, the on-board debugger is a true UART bridge between the host and the device. However, in certain use cases, the on-board debugger can override the basic Operating mode and use the CDC TX and RX pins for other purposes.

Dropping a text file into the on-board debugger's mass storage drive can be used to send characters out of the debugger's CDC TX pin. The filename and extension are trivial, but the text file will start with the characters:

```
CMD:SEND_UART=
```

Debugger firmware version 1.20 or earlier has the following limitations:

- The maximum message length is 50 characters – all remaining data in the frame are ignored
- The default baud rate used in this mode is 9600 bps, but if the CDC is already active or configured, the previously used baud rate still applies

Debugger firmware version 1.21 and later has the following limitations/features:

- The maximum message length will vary depending on the MSC/SCSI layer timeouts on the host computer and/or operating system. A single SCSI frame of 512 bytes (498 characters of payload) is ensured, and files up to 4 KB will work on most systems. The transfer will complete on the first NULL character encountered in the file.
- The baud rate used is always 9600 bps for the default command:

  ```
  CMD:SEND_UART=
  ```

- Do not use the CDC Override mode simultaneously with data transfer over the CDC/terminal. If a CDC terminal session is active when receiving a file via the CDC Override mode, it will be suspended for the duration of the operation and resumed once complete.
- Additional commands are supported with explicit baud rates:

  ```
  CMD:SEND_9600=
  ```

  ```
  CMD:SEND_115200=
  ```

  ```
  CMD:SEND_460800=
  ```

USB-Level Framing Considerations
Sending data from the host to the CDC can be done byte-wise or in blocks, chunked into 64-byte USB frames. Each frame will be queued for transfer to the debugger's CDC TX pin. Sending a small amount of data per frame can be inefficient, particularly at low baud rates, as the on-board debugger buffers frames but not bytes. A maximum of four 64-byte frames can be active at any time. The on-board debugger will throttle the incoming frames accordingly. Sending full 64-byte frames containing data is the most efficient method.

When receiving data on the debugger's CDC RX pin, the on-board debugger will queue up the incoming bytes into 64-byte frames, which are sent to the USB queue for transmission to the host when they are full. Incomplete frames are also pushed to the USB queue at approximately 100 ms intervals, triggered by USB start-of-frame tokens. Up to eight 64-byte frames can be active at any time.

An overrun will occur if the host (or the software running on it) fails to receive data fast enough. When this happens, the last-filled buffer frame recycles instead of being sent to the USB queue, and a complete data frame will be lost. To prevent this occurrence, the user will ensure that the CDC data pipe is continuously read, or the incoming data rate will be reduced.

Sending Break Characters
The host can send a UART break character to the device using the CDC, which can be useable for resetting a receiver state-machine or signalling an exception condition from the host to the application running on the device.

A break character is defined as a sequence of at least 11 zero bits transmitted from the host to the device.

Not all UART receivers have support for detecting a break, but a correctly-formed break character usually triggers a framing error on the receiver.

Sending a break character using the debugger's CDC has the following limitations:

- Sending a break must NOT be done simultaneously as using the CDC Override mode (drag-and-drop). Both these functions are temporary states, so they must be used independently.
- Sending a break will cause data currently being sent to be lost. Be sure to wait a sufficient amount of time to allow all characters in the transmission buffer to be sent (see above section) before sending the break, which is also in line with expected break character usage: For example, reset a receiver state-machine after a timeout occurs waiting for data to be returned to the host.
- The CDC specification allows for debugger-timed breaks of up to 65534ms in duration to be requested. For simplicity, the debugger will limit the break duration to maximum 11 bit-durations at its minimum supported baud rate.
- The CDC specification allows for indefinite host-timed breaks. It is the responsibility of the terminal application/ user to release the break state in this case.

Note: Sending break characters is available in debugger firmware version 1.24 and later.

3.1.3 Mass Storage Device

The on-board debugger includes a simple Mass Storage Device implementation, which is accessible for read/write operations via the host operating system to which it is connected.

It provides:

- Read access to basic text and HTML files for detailed kit information and support
- Write access for programming Intel® HEX formatted files into the target device's memory
- Write access for simple text files for utility purposes

3.1.3.1 Mass Storage Device Implementation

The on-board debugger implements a highly optimized variant of the FAT12 file system with several limitations, partly due to the nature of FAT12 itself and optimizations made to fulfill its purpose for its embedded application.

The Curiosity Nano USB device is USB Chapter 9-compliant as a mass storage device but does not, in any way, fulfill the expectations of a general purpose mass storage device. This behavior is intentional.

When using the Windows operating system, the on-board debugger enumerates as a Curiosity Nano USB Device found in the disk drives section of the device manager. The CURIOSITY drive appears in the file manager and claims the following available drive letter in the system.

The CURIOSITY drive contains approximately one MB of free space and does not reflect the target device's Flash size. When programming an Intel HEX file, the binary data are encoded in ASCII with metadata providing a large overhead, so 1 MB is a trivially chosen value for disk size.

It is not possible to format the CURIOSITY drive. The filename may appear in the disk directory listing when programming a file to the target, which is merely the operating system's view of the directory that, in reality, has not been updated. It is not possible to read out the file contents. Removing and replugging the board will return the file system to its original state, but the target will still contain the previously programmed application.

Copy a text file starting with "`CMD:ERASE`" onto the disk to erase the target device.

By default, the CURIOSITY drive contains several read-only files for generating icons as well as reporting status and linking to further information:

- `AUTORUN.ICO` – icon file for the Microchip logo
- `AUTORUN.INF` – system file required for Windows Explorer to show the icon file
- `KIT-INFO.HTM` – redirect to the development board website
- `KIT-INFO.TXT` – a text file containing details about the board's debugger firmware version, board name, USB serial number, device, and drag-and-drop support
- `STATUS.TXT` – a text file containing the programming status of the board

Info: The on-board debugger dynamically updates `STATUS.TXT`. The contents may be cached by the OS and, therefore, may not reflect the correct status.

3.1.3.2 Fuse Bytes

Fuse Bytes (AVR® MCU Targets)
When doing drag-and-drop programming, the debugger masks out any fuse bits attempting to disable Unified Program and Debug Interface (UPDI), meaning that the UPDI pin cannot be used in its Reset or GPIO modes; selecting one of the alternative functions on the UPDI pin will render the device inaccessible without using an external debugger capable of high-voltage UPDI activation.

3.1.3.3 Limitations of Drag-and-Drop Programming

Lock Bits
Lock bits included in the hex file will be ignored when using drag-and-drop programming. To program lock bits, use MPLAB® X IDE or Microchip Studio.

Enabling CRC Check in Fuses
It is not advisable to enable the CRC check fuses in the target device when using drag-and-drop programming because a subsequent chip erase (which does not affect fuse bits) will cause a CRC mismatch, and the application will fail to boot. To recover a target from this state, a chip erase must be done using MPLAB® X IDE or Microchip Studio, which automatically will clear the CRC fuses after erasing.

3.1.3.4 Special Commands

Several utility commands are supported by copying text files to the mass storage disk. The filename or extension is irrelevant – the command handler reacts to content only.

Table 3-2. Special File Commands

Command Content	Description
`CMD:ERASE`	Executes a chip erase of the target
`CMD:SEND_UART=`	Sends a string of characters to the CDC UART. See "CDC Override Mode."

`CMD:SEND_9600=`
`CMD:SEND_115200=`
`CMD:SEND_460800=`
	Sends a string of characters to the CDC UART at the specified baud rate. Note that only the baud rates explicitly specified here are supported. See "CDC Override Mode." (Debugger firmware v1.25.6 or newer.)

`CMD:RESET`	Resets the target device by entering Programming mode and exiting Programming mode immediately afterward. The exact timing can vary according to the programming interface of the target device. (Debugger firmware v1.25.6 or newer.)
`CMD:POWERTOGGLE`	Powers down the target and restores it after a 100 ms delay. If external power is provided, this has no effect. (Debugger firmware v1.25.6 or newer.)
`CMD:0V`	Powers down the target device by disabling the target supply regulator. If external power is provided, this has no effect. (Debugger firmware v1.25.6 or newer.)
`CMD:1V8`	Sets the target voltage to 1.8V. If external power is provided, this has no effect. (Debugger firmware v1.25.6 or newer.)
`CMD:3V3`	Sets the target voltage to 3.3V. If external power is provided, this has no effect. (Debugger firmware v1.25.6 or newer.)

Info: The content sent to the mass storage emulated disk triggers the commands listed here and provides no feedback in the case of either success or failure.

3.1.4 Data Gateway Interface (DGI)

Data Gateway Interface (DGI) is a USB interface transporting raw and timestamped data between on-board debuggers and host computer-based visualization tools. MPLAB Data Visualizer is used on the host computer to display any debug GPIO data. It is available as a plug-in for MPLAB X IDE or a stand-alone application that can be used in parallel with MPLAB X IDE or Microchip Studio.

Although DGI encompasses several physical data interfaces, the AVR64EA48 Curiosity Nano implementation includes logic analyzer channels:

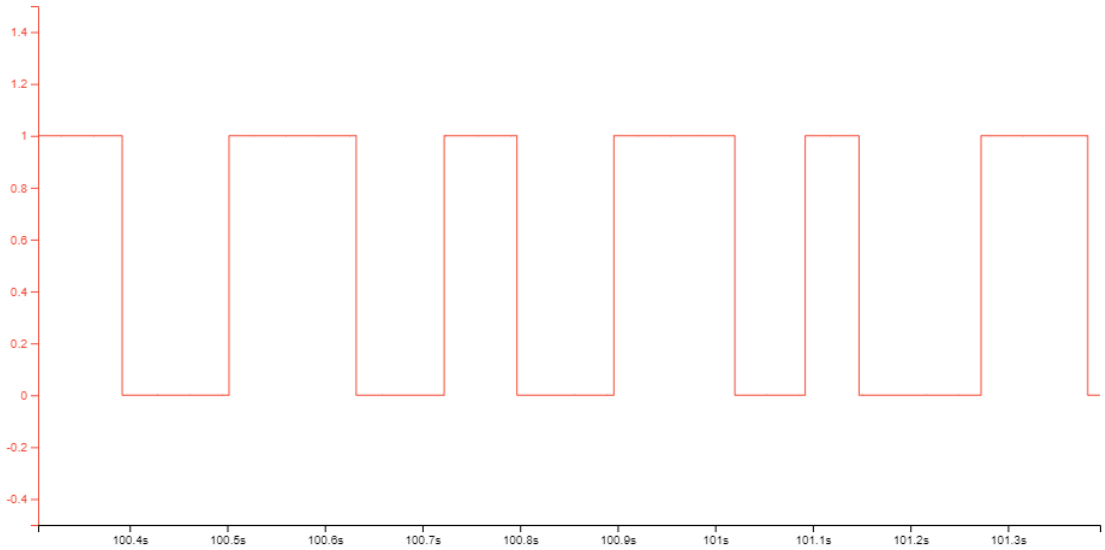
- Two debug GPIO channels (also known as DGI GPIO)

3.1.4.1 Debug GPIO

Debug GPIO channels are timestamped digital signal lines connecting the target application to a host computer visualization application. They are typically used to plot low-frequency events on a time axis, such as when given Application state transitions occur.

The figure below shows the monitoring of the Digital state of a mechanical switch connected to a debug GPIO in MPLAB Data Visualizer.

Figure 3-2. Monitoring Debug GPIO with MPLAB Data Visualizer



Debug GPIO channels are timestamped, so the resolution of DGI GPIO events is determined by the resolution of the DGI Timestamp module.

Important: Although signal bursts of higher frequency can be captured, the frequency range of signals for which debug GPIO can be used is up to about 2 kHz. Attempting to capture signals above this frequency will result in data saturation and overflow, which may cause the DGI session to be aborted.

3.1.4.2 Timestamping

When captured by the debugger, DGI sources are timestamped. The timestamp counter implemented in the Curiosity Nano debugger increments at a 2 MHz frequency, providing a timestamp resolution of a half microsecond.

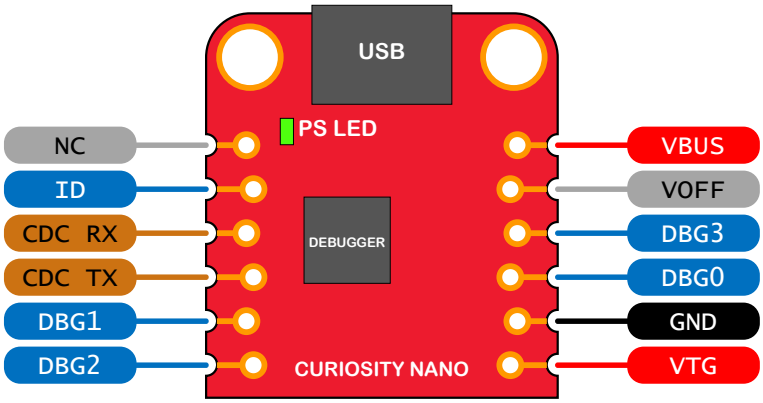
3.2 Curiosity Nano Standard Pinout

The 12-edge connections closest to the USB connector on Curiosity Nano boards have a standardized pinout. The program/debug pins have different functions depending on the target programming interface, as shown in the table and figure below.

Table 3-3. Curiosity Nano Standard Pinout

Debugger Signal	Target MCU	Description
ID	—	ID line for extensions
CDC TX	UART RX	USB CDC TX line
CDC RX	UART TX	USB CDC RX line
DBG0	UPDI	Debug data line
DBG1	GPIO1	Debug GPIO1
DBG2	GPIO0	Debug GPIO0
DBG3	RESET	Reset line
NC	—	No connect
V_{BUS}	—	V_{BUS} voltage for external use
$\overline{VOFF}$	—	$\overline{Voltage\ Off}$ input. Disables the target regulator and target voltage when pulled low.
VTG	—	Target voltage
GND	—	Common ground

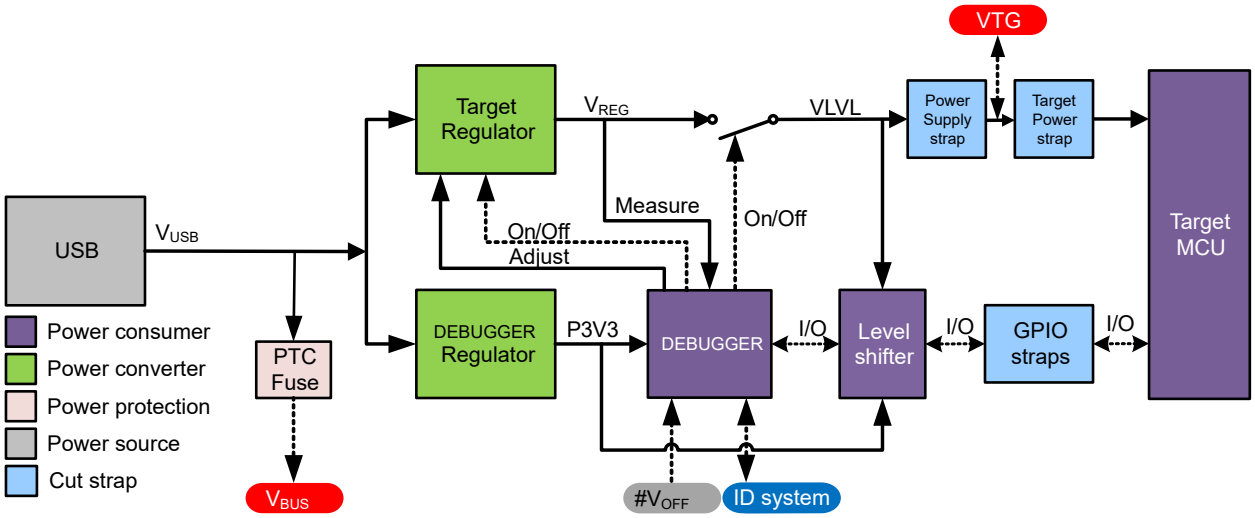
Figure 3-3. Curiosity Nano Standard Pinout



3.3 Power Supply

The USB port powers the board. It contains two LDO regulators, one to generate 3.3V for the on-board debugger and an adjustable LDO regulator for the target AVR64EA48 microcontroller and its peripherals. The voltage from a USB connector can vary between 4.4V and 5.25V (according to the USB specification) and will limit the maximum voltage supplied to the target. The figure below shows the entire power supply system on AVR64EA48 Curiosity Nano.

Figure 3-4. Power Supply Block Diagram



3.3.1 Target Regulator

The target voltage regulator is a MIC5353 variable output LDO. The on-board debugger can adjust the voltage output supplied to the board target section by manipulating the MIC5353's feedback voltage. The hardware implementation is limited to an approximate voltage range from 1.7V to 5.1V. Additional output voltage limits are configured in the debugger firmware to ensure that the output voltage never exceeds the hardware limits of the AVR64EA48 microcontroller. The voltage limits configured in the on-board debugger on AVR64EA48 Curiosity Nano are 1.8–5.5V.

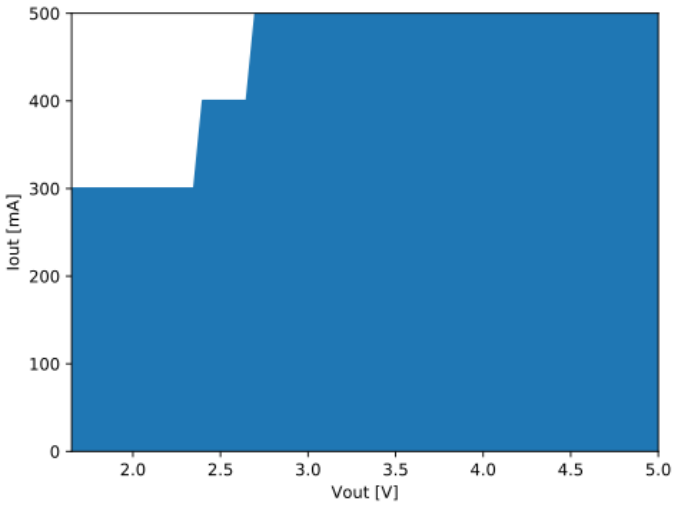
Info: The factory default target voltage is 3.3V. It can be changed through MPLAB® X IDE project properties and in the Microchip Studio device programming dialog. Any change to the target voltage is persistent, even after a power toggle. The resolution is less than 5 mV but may be limited to 10 mV by the adjustment program.

Info: The voltage settings setup in MPLAB® X IDE is not applied immediately to the board. The new voltage setting is applied to the board when accessing the debugger, like pushing the Refresh Debug Tool Status button in the project dashboard tab or programming/reading program memory.

Info: There is an easy option to adjust the target voltage with a drag-and-drop command text file to the board, which supports a set of mutual target voltages. See section 3.1.3.4. Special Commands for further details.

MIC5353 supports a maximum current load of 500 mA. It is an LDO regulator in a small package placed on a small printed circuit board (PCB), and the thermal shutdown condition can be reached at lower loads than 500 mA. The maximum current load depends on the input voltage, the selected output voltage, and the ambient temperature. The figure below shows the safe operating area for the regulator, with an input voltage of 5.1V and an ambient temperature of 23°C.

Figure 3-5. Target Regulator Safe Operation Area



The voltage output of the target regulator is continuously monitored (measured) by the on-board debugger. An error condition will be flagged, and the target voltage regulator will be switched off, detecting and handling any short-circuit conditions if it is more than 100 mV over/under the set device voltage. It will also detect and handle if an external voltage, which causes V_{CC}_TARGET to move outside the voltage setting monitoring window of ±100 mV, is suddenly applied to the VTG pin without setting the V_{OFF} pin low.

Info: The on-board debugger has a monitoring window of V_{CC}_TARGET±100 mV, and the status LED will blink rapidly if the external voltage is under this limit. The on-board debugger status LED will continue to shine if the external voltage surpasses this limit. When removing the external voltage, the status LED will start blinking rapidly until the on-board debugger detects the new situation and turns the target voltage regulator back on.

3.3.2 External Supply

Instead of the on-board target regulator, an external voltage can power the AVR64EA48 Curiosity Nano. When shorting the Voltage Off (VOFF) pin to the ground (GND) pin, the on-board debugger firmware disables the target regulator, and it is safe to apply an external voltage to the VTG pin.

It is also safe to apply an external voltage to the VTG pin when no USB cable is plugged into the DEBUG connector on the board.

The VOFF pin can be tied low/let go at any time, which will be detected by a pin-change interrupt to the on-board debugger, which controls the target voltage regulator accordingly.

WARNING Applying an external voltage to the VTG pin without shorting VOFF to GND may cause permanent damage to the board.

WARNING Do not apply any voltage to the VOFF pin. Let the pin float to enable the power supply.

WARNING The absolute maximum external voltage is 5.5V for the on-board level shifters, and the standard operating condition of the AVR64EA48 is 1.8–5.5V. Applying a higher voltage may cause permanent damage to the board.

Info: The on-board debugger status LED will blink rapidly and shut the on-board regulator off when applying an external voltage without pulling the VOFF pin low - and an external supply pulls the voltage lower than the monitoring window's lower limit (target voltage setting – 100 mV). The status LED starts blinking rapidly until the on-board debugger detects the new situation and switches the target voltage regulator back on if suddenly removing an external voltage when the VOFF pin is not pulled low.

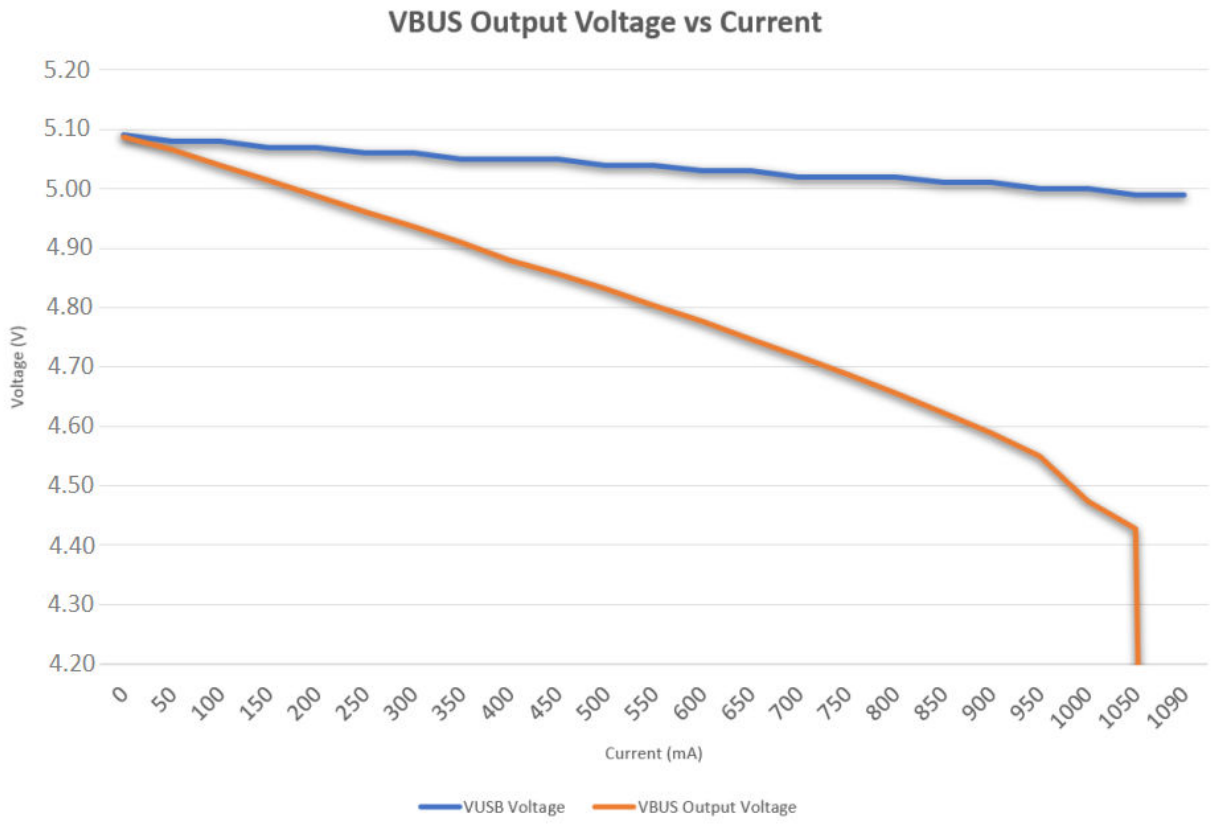
Programming, debugging, and data streaming are still possible with an external power supply. The USB cable will power the debugger and signal level shifters. Both regulators, the debugger, and the level shifters are powered down when the USB cable is removed.

Info: In addition to the power consumed by the AVR64EA48 and its peripherals, approximately 100 µA will be drawn from any external power source to power the on-board level shifters and voltage monitor circuitry when a USB cable is plugged into the DEBUG connector on the board. When a USB cable is unplugged, some current is used to supply the level shifter's voltage pins, which have a worst-case current consumption of approximately 5 µA. Typical values may be as low as 100 nA.

3.3.3 VBUS Output Pin

AVR64EA48 Curiosity Nano has a VBUS output pin that can be used to power external components that need a 5V supply. The VBUS output pin has a PTC fuse to protect the USB against short circuits. A side effect of the PTC fuse is a voltage drop on the VBUS output with higher current loads. The chart below shows the voltage versus the current load of the VBUS output.

Figure 3-6. VBUS Output Voltage vs. Current



3.3.4 Power Supply Exceptions

This section sums up most exceptions that can occur with the power supply.

Target Voltage Shuts Down
Not reaching the target voltage setting can happen if the target section draws too much current at a given voltage and cause the thermal shutdown safety feature of the MIC5353 regulator to kick in. To avoid this, reduce the current load of the target section.

Target Voltage Setting is Not Reached
The USB input voltage (specified to be 4.4V-5.25V) limits the maximum output voltage of the MIC5353 regulator at a given voltage setting and current consumption. If a higher output voltage is needed, use a USB power source with a higher input voltage or use an external voltage supply on the VTG pin.

Target Voltage is Different From Setting
An externally applied voltage to the VTG pin without setting the VOFF pin low can cause this. If the target voltage differs more than 100 mV over/under the voltage setting, the on-board debugger will detect it, and the internal voltage regulator will shut down. To fix this issue, remove the applied voltage from the VTG pin, and the on-board debugger will enable the on-board voltage regulator when the new condition is detected. Note that the PS LED will blink rapidly if the target voltage is below 100 mV of the setting but will ordinarily turn on when higher than 100 mV above it.

No, Or Very Low Target Voltage, and PS LED is Blinking Rapidly
A full or partial short circuit can cause this and is a particular case of the issue above. Remove it, and the on-board debugger will re-enable the on-board target voltage regulator.

No Target Voltage and PS LED is Lit 1
This situation occurs if the target voltage is set to 0.0V. Set the target voltage to a value within the specified voltage range for the target device to fix this.

No Target Voltage and PS LED is Lit 2
This situation can be the issue if power jumper J100 and/or J101 are cut, and the target voltage regulator is set to a value within the specified voltage range for the target device. To fix this, solder a wire/bridge between the pads for J100/J101, or add a jumper on J101 if a pin-header is mounted.

V_{BUS} Output Voltage is Low or Not Present
If the VBUS output voltage is low or missing, the reason is probably a high-current drain on V_{BUS}, and the protection fuse (PTC) will reduce the current or cut off completely. Reduce the current consumption on the VBUS pin to fix this issue.

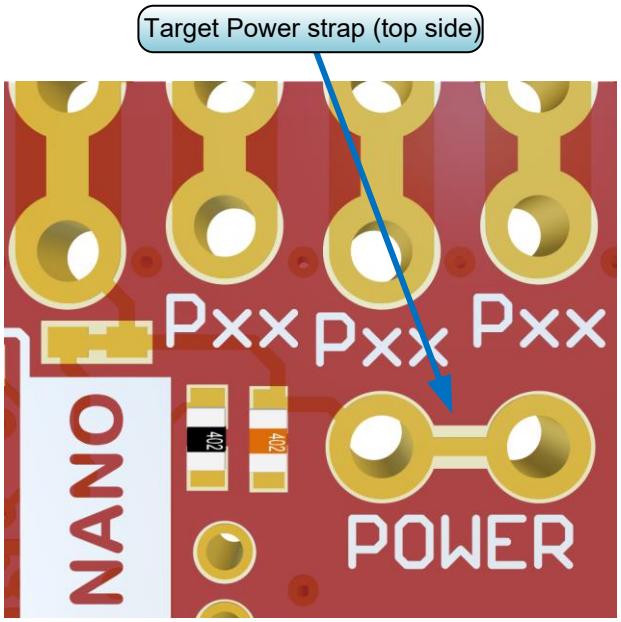
3.4 Low-Power Measurement

Power to the AVR64EA48 is connected from the on-board power supply and *VTG* pin through a 100 mil pin-header marked with “POWER” in silkscreen (J101). To measure the power consumption of the AVR64EA48 and other peripherals connected to the board, cut the *Target Power strap* and connect an ammeter over it.

To measure the lowest possible power consumption, follow these steps:

1. Cut the POWER strap with a sharp tool.
2. Solder a 1x2 100 mil pin-header in the footprint.
3. Connect an ammeter to the pin header.
4. Write firmware that:
    a. Tri-states any I/O connected to the on-board debugger.
    b. Sets the microcontroller in its lowest Power sleep mode.
5. Program the firmware into the AVR64EA48.

Figure 3-7. Target Power Strap



Tip: A 100-mil pin-header can be soldered into the *Target Power strap* (J101) footprint for a simple connection of an ammeter. Place a jumper cap on the pin-header once the ammeter is no longer needed.

Info: The on-board level shifters will draw a small amount of current even when not used. Maximum 2 µA can be drawn from each I/O pin connected to a level shifter. Therefore, the worst-case maximum for the five on-board level shifters is 10 µA. Keep any I/O pin connected to a level shifter in the tri-state to prevent leakage. All I/Os connected to the on-board debugger are listed in 4.2.3.1. On-Board Debugger Connections. The on-board level shifters can be completely disconnected to prevent leakage, as described in 7.4. Disconnecting the On-Board Debugger.

3.5 Programming External Microcontrollers

Use the on-board debugger on AVR64EA48 Curiosity Nano to program and debug microcontrollers on external hardware.

3.5.1 Supported Devices

All external AVR microcontrollers with the UPDI interface can be programmed and debugged with the on-board debugger with Microchip Studio.

External SAM microcontrollers having a Curiosity Nano Board can be programmed and debugged with the on-board debugger with Microchip Studio.

AVR64EA48 Curiosity Nano can program and debug external AVR64EA48 microcontrollers with MPLAB X IDE.

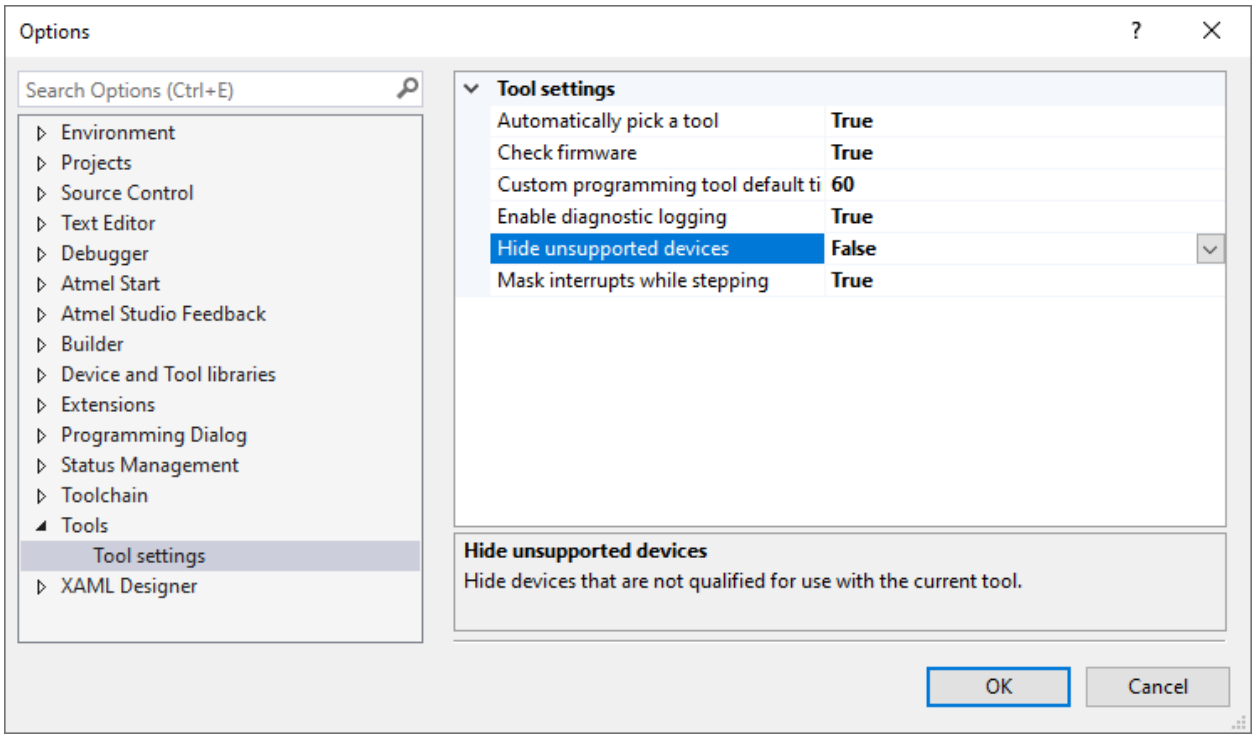
3.5.2 Software Configuration

No software configuration is required to program and debug the same device mounted on the board.

To program and debug a different microcontroller than the one mounted on the board, configure Microchip Studio to allow an independent selection of devices and programming interfaces.

1. Navigate to **Tools** > **Options** through the menu system at the application top.
2. Select the **Tools** > **Tool settings** category in the options window.
3. Set the **Hide unsupported devices** option to **False**.

Figure 3-8. Hide Unsupported Devices

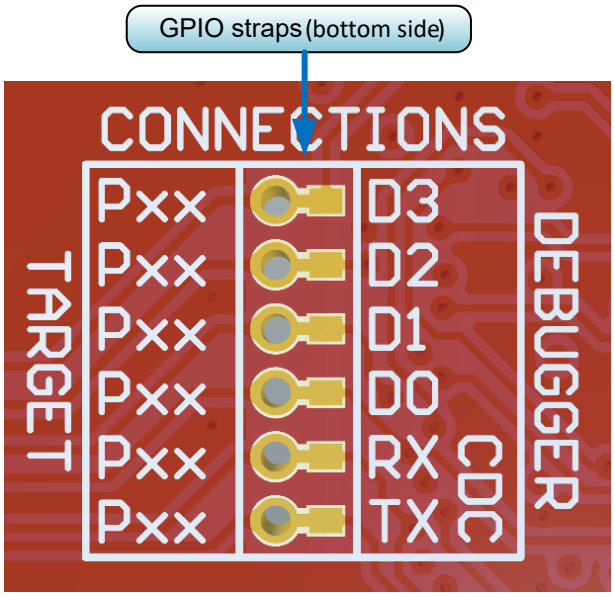


Info: Microchip Studio allows any microcontroller and interface to be selected when the **Hide unsupported devices** setting is set to **False** - also microcontrollers and interfaces not supported by the on-board debugger.

3.5.3 Hardware Modifications

The on-board debugger is connected to the AVR64EA48 by default. Remove these connections before any external microcontroller can be programmed or debugged. Cut the *GPIO straps* shown in the figure below with a sharp tool to disconnect the AVR64EA48 from the on-board debugger.

Figure 3-9. Programming and Debugging Connections to Debugger



Info: Cutting the connections to the debugger will disable programming, debugging, and data streaming from the AVR64EA48 mounted on the board.

Tip: Solder 0Ω resistors across the footprints or short circuit them with solder to reconnect the signals between the on-board debugger and the AVR64EA48.

3.5.4 Connecting to External Microcontrollers

The figure and table below show where to connect the programming and debugging signals to program and debug external microcontrollers. The on-board debugger can supply power to the external hardware or use an external voltage as a reference for its level shifters. Read more about the power supply in 3.3. Power Supply.

The on-board debugger and level shifters actively drive data and clock signals used for programming and debugging (DBG0, DBG1, and DBG2). Pull-down resistors are required on the ICSP™ data and clock signals to debug PIC® microcontrollers. All other interfaces are functional with or without pull-up or pull-down resistors.

DBG3 is an open-drain connection and requires a pull-up resistor to function.

Remember:

- Connect GND and VTG to the external microcontroller
- Tie the VOFF pin to GND if the external hardware has a power supply
- Make sure there are pull-down resistors on the ICSP data and clock signals (DBG0 and DBG1) to support the debugging of PIC microcontrollers

Figure 3-10. Curiosity Nano Standard Pinout

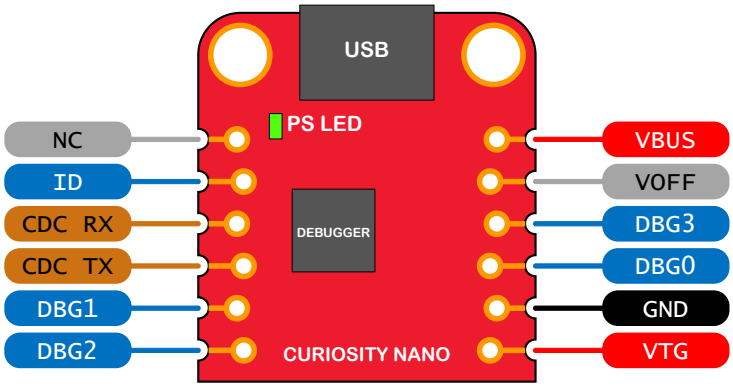


Table 3-4. Programming and Debugging Interfaces

Curiosity Nano Pin	UPDI	ICSP™	SWD
DBG0	UPDI	DATA	SWDIO
DBG1	—	CLK	SWCLK
DBG2	—	—	—
DBG3	—	$\overline{\text{MCLR}}$	$\overline{\text{RESET}}$

3.6 Connecting External Debuggers

Even though there is an on-board debugger, external debuggers can be connected directly to the AVR64EA48 Curiosity Nano to program/debug the AVR64EA48. When not actively used, the on-board debugger keeps all the pins connected to the AVR64EA48 and board edge in tri-state. Therefore, the on-board debugger will not interfere with any external debug tools.

Figure 3-11. Connecting the MPLAB® PICkit™ 4 In-Circuit Debugger/Programmer to AVR64EA48 Curiosity Nano

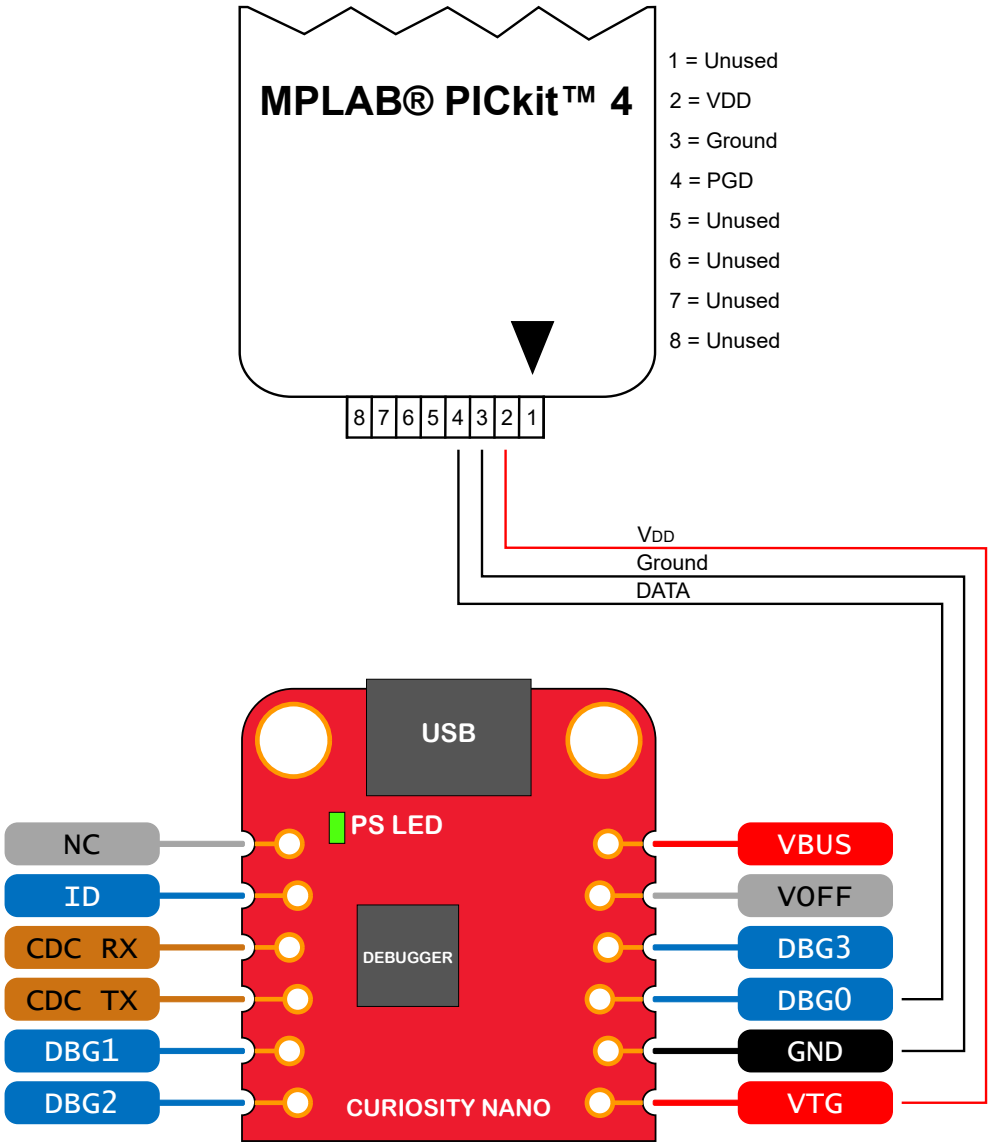
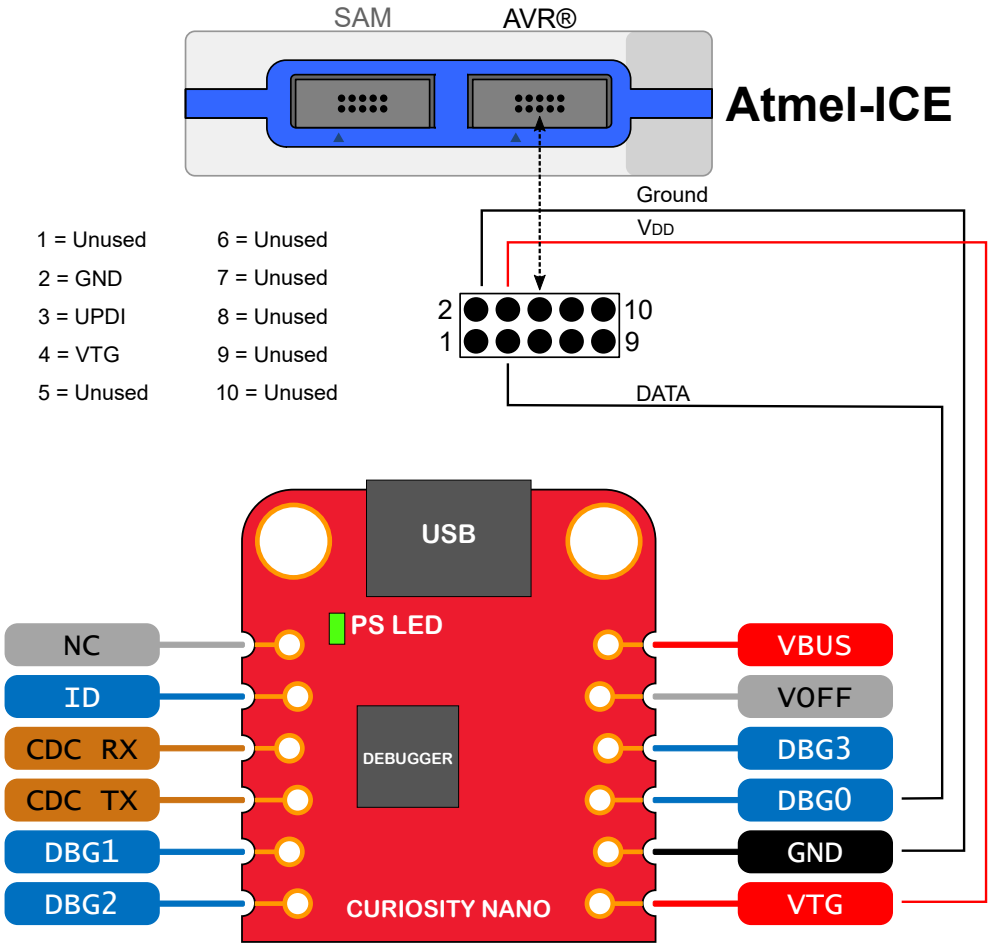


Figure 3-12. Connecting the Atmel-ICE to AVR64EA48 Curiosity Nano



⚠ CAUTION To avoid contention between the external debugger and the on-board debugger, do not start any programming/debug operation with the on-board debugger through MPLAB® X IDE or Microchip Studio or mass storage programming while the external tool is active.

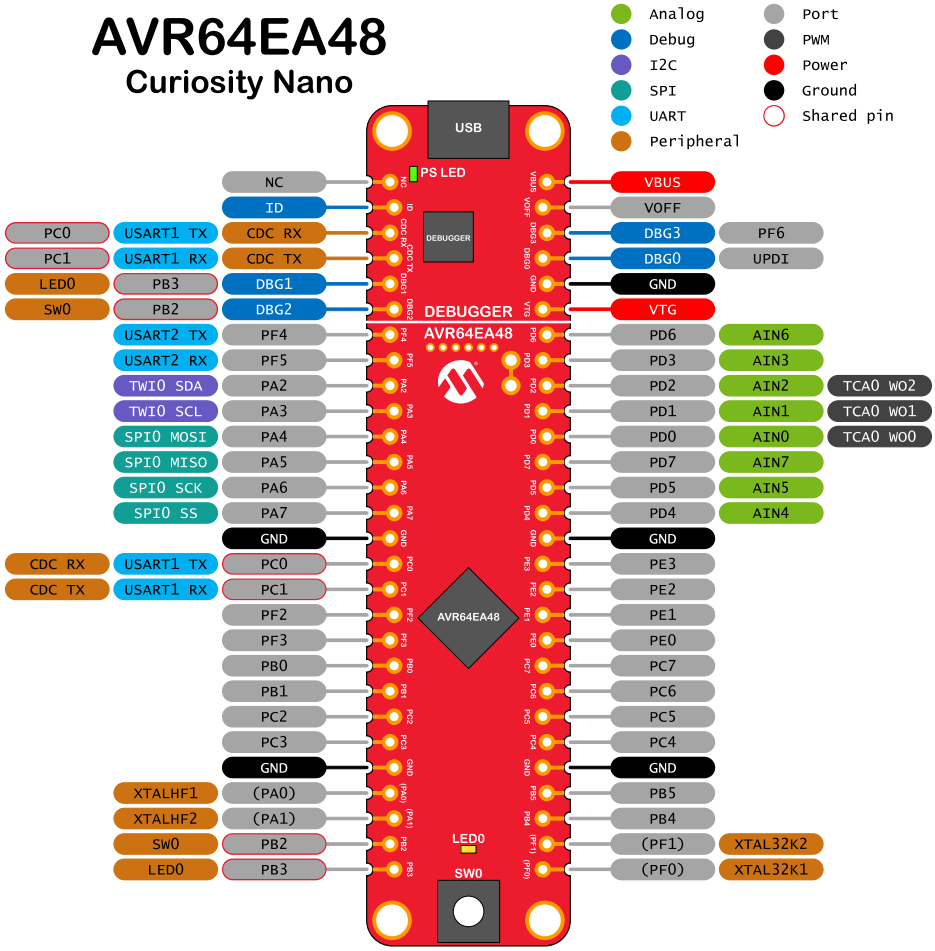
4. Hardware Description

4.1 Connectors

4.1.1 AVR64EA48 Curiosity Nano Pinout

All the AVR64EA48 I/O pins are accessible at the edge connectors on the board. The image below shows the board pinout. Refer to the I/O Multiplexing and Considerations section in the AVR64EA48 data sheet for all available functions on each pin.

Figure 4-1. AVR64EA48 Curiosity Nano Pinout

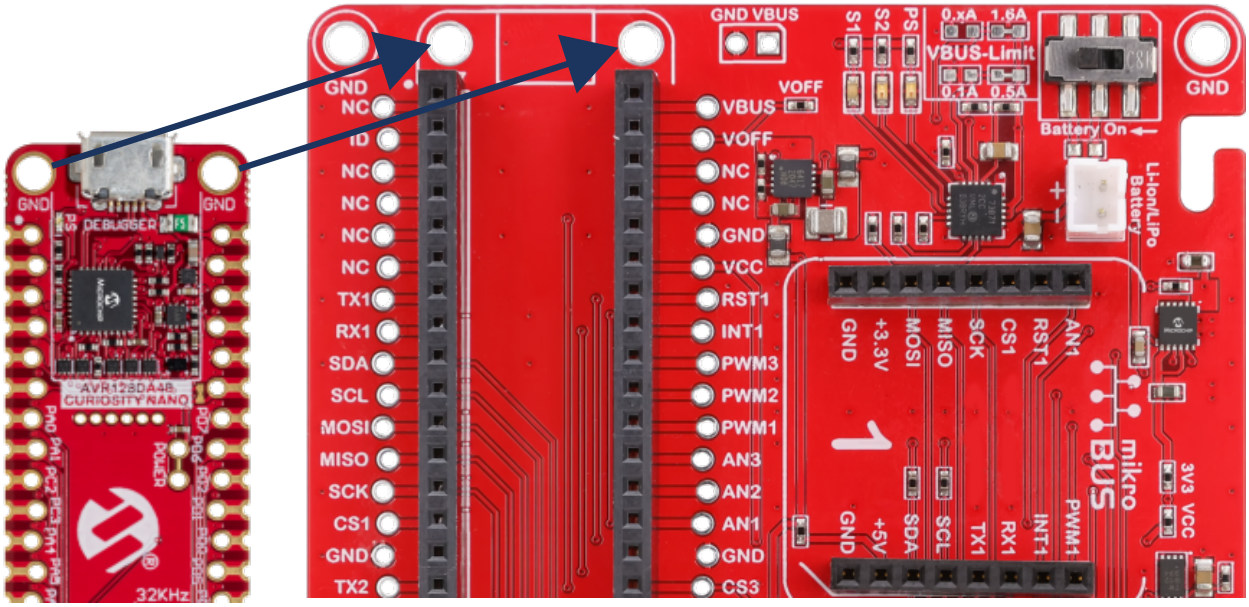


4.1.2 Using Pin-Headers

The edge connector footprint on AVR64EA48 Curiosity Nano has a staggered design where each hole is shifted 8 mil (~0.2 mm) off-center. The hole shift allows regular 100 mil pin-headers to be used without soldering on the board. The pin-headers can be used in applications like pin sockets and prototyping boards without issues once they are firmly in place.

Figure 4-2. Attaching Pin-Headers to the Curiosity Nano Board

Figure 4-3. Connecting to Curiosity Nano Base for Click boards



Tip: Start at one end of the pin-header and gradually insert the header along the length of the board. Once all the pins are in place, use a flat surface to push them in.

Tip: For applications where the pin-headers will be used permanently, it is still recommended to solder them in place.

Important: Once the pin-headers are in place, they are hard to remove by hand. Use a set of pliers and carefully remove the pin-headers to avoid damage to the pin-headers and PCB.

4.2 Peripherals

4.2.1 LED

One yellow user LED is available on the AVR64EA48 Curiosity Nano board. It can be controlled by either GPIO or PWM. Driving the connected I/O line to GND can also activate the LED.

Table 4-1. LED Connection

AVR64EA48 Pin	Function	Shared Functionality
PB3	Yellow LED0	Edge connector, On-board debugger

4.2.2 Mechanical Switch

The AVR64EA48 Curiosity Nano board has one mechanical switch - a generic user-configurable switch. Pressing it will connect the I/O pin to ground (GND).

Tip: There is no externally connected pull-up resistor on the switch. Enable the internal pull-up resistor on Pin PB2 to use it.

Table 4-2. Mechanical Switch

AVR64EA48 Pin	Description	Shared Functionality
PB2	User switch (SW0)	Edge connector, On-board debugger

4.2.3 On-Board Debugger Implementation

AVR64EA48 Curiosity Nano features an on-board debugger that can be used to program and debug the AVR64EA48 using UPDI. The on-board debugger also includes a virtual serial port (CDC) interface over UART and debug GPIO. MPLAB X IDE and Microchip Studio can be used as a front-end for the on-board debugger for programming and debugging. MPLAB Data Visualizer can be used as a front-end for the CDC and debug GPIO.

4.2.3.1 On-Board Debugger Connections

The table below shows the connections between the target and the debugger section. All the connections between the target and the debugger are tri-stated when the debugger is not using the interface. Hence, there are few contaminations of the signals, e.g., the pins can be configured to anything the user wants.

For further information on how to use the capabilities of the on-board debugger, see 3.1. On-Board Debugger Overview.

Table 4-3. On-Board Debugger Connections

AVR64EA48 Pin	Debugger Pin	Function	Shared Functionality
PC1	CDC TX	UART RX (AVR64EA48 RX line)	Edge connector
PC0	CDC RX	UART TX (AVR64EA48 TX line)	Edge connector
UPDI	DBG0	UPDI	Edge connector
PB3	DBG1	GPIO1	Edge connector
PB2	DBG2	SW0/GPIO0	Edge connector
PF6	DBG3	RESET	Edge connector

4.2.4 20 MHz Crystal

On the AVR64EA48 Curiosity Nano board is a mounted 20.00 MHz crystal of type VXM7-9061-20M0000000. The crystal is accurate to within 20 ppm. Using the External High-Frequency Crystal Oscillator (XOSCHF) module in AVR64EA48 with the mounted 20.00 MHz crystal generates a clock signal that is significantly more accurate than the Internal High-Frequency Oscillator (OSCHF) module. With the Auto-Tune feature, the 20 MHz can be used as a reference to improve the accuracy of the internal oscillator.

The 20.00 MHz crystal is connected to AVR64EA48 on PA0 and PA1, which are also routed to the edge connector through two solder points. PA0 and PA1 are disconnected from the edge connector by default to reduce the chance of an external signal causing contention with the crystal and to remove excessive capacitance on the lines.

Next to the 20.00 MHz crystal is a cut strap (J217) that can be used to measure the oscillator safety factor, which is done by cutting the strap and adding a 0402 SMD resistor across the strap. The AN2648 application note from Microchip contains detailed information about the oscillator allowance and safety factor.

Figure 4-4 shows cut straps and solder points.

Table 4-4. Crystal Connections

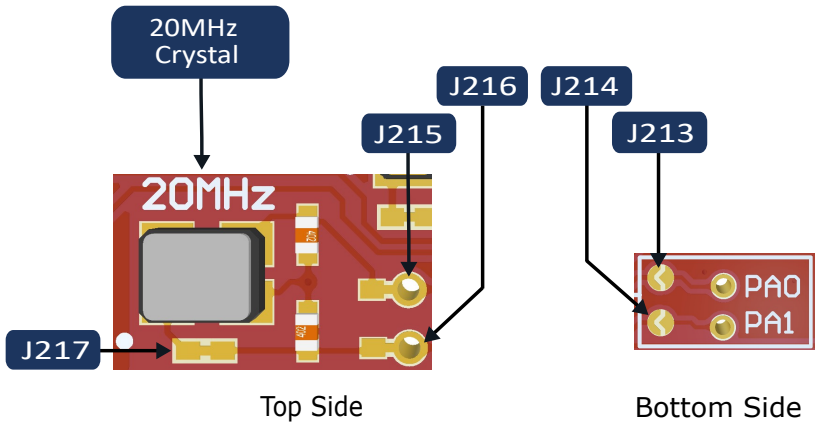
AVR64EA48 Pin	Function	Shared Functionality
PA0	XTALHF1 (Crystal input)	Edge connector
PA1	XTALHF2 (Crystal output)	Edge connector

Some hardware modifications are required to use PA0 and PA1 as GPIO.

WARNING Before doing any hardware modifications, make sure the board is disconnected from USB or external power.

- Disconnect the 20.00 MHz crystal by cutting the two straps on the top side of the board next to the crystal (J215, J216). The crystal must be disconnected when using the pin as GPIO, as this might harm the crystal.
- Connect the I/O lines to the edge connector by placing solder blobs on the circular solder points marked PA0 and PA1 on the bottom side of the board (J213, J214)

Figure 4-4. Crystal Connection and Cut Straps



4.2.5 32.768 kHz Crystal

On the AVR64EA48 Curiosity Nano board, a VMK3-9002-32K7680000TR 32.768 kHz crystal is mounted. The crystal is accurate to within 20 ppm. Using the external 32.768 kHz Crystal Oscillator (XOSC32K) module in AVR64EA48 with the mounted 32.768 kHz crystal generates a clock signal that is significantly more accurate than the internal 32.768 kHz Ultra Low-Power Oscillator (OSC32K) module can generate alone, which is an ideal reference clock for the Real-Time Counter module in AVR64EA48.

The 32.768 kHz crystal is connected to AVR64EA48 on PF0 and PF1, which are also routed to the edge connector through two solder points. PF0 and PF1 are disconnected from the edge connector by default to reduce the chance of an external signal causing contention with the crystal and to remove excessive capacitance on the lines.

Next to the 32.768 kHz crystal is a cut strap (J209), used to measure the oscillator safety factor by cutting the strap and adding a 0402 SMD resistor across. More information about the oscillator allowance and the safety factor is available in the AN2648 application note from Microchip.

Figure 4-5 shows the cut straps and solder points.

Table 4-5. 32.768 kHz Crystal Connections

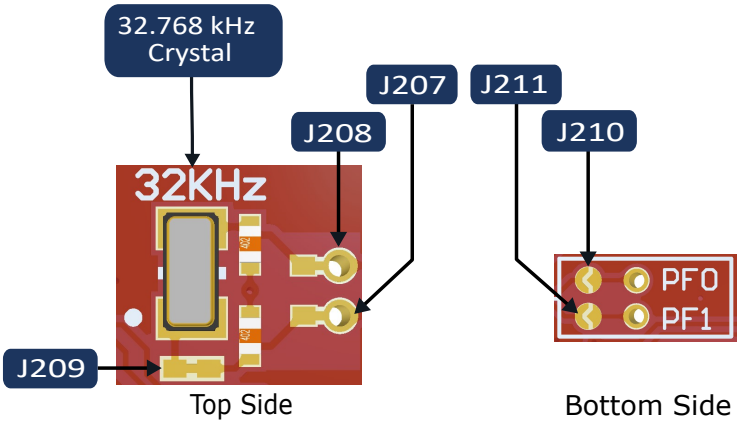
AVR64EA48 Pin	Function	Shared Functionality
PF0	XTAL32K1 (Crystal input)	Edge connector
PF1	XTAL32K2 (Crystal output)	Edge connector

Some hardware modifications are required to use PF0 and PF1 as GPIO.

WARNING Before doing any hardware modifications, make sure the board is disconnected from the USB or external power.

- Disconnect the 32.768 kHz crystal by cutting the two straps on the top side of the board next to the crystal (J208, J207). The crystal must be disconnected when using the pin as GPIO, or it might harm the crystal.
- Connect the I/O lines to the edge connector by placing solder blobs on the circular solder points marked PF0 and PF1 on the bottom side of the board (J210, J211)

Figure 4-5. Crystal Connection and Cut Straps



5. Hardware Revision History and Known Issues

This user guide provides information about the latest available revision of the board. The following sections contain information about known issues, a revision history of older revisions, and how older revisions differ from the latest revision.

5.1 Identifying Product ID and Revision

There are two ways to find the revision and product identifier of the AVR64EA48 Curiosity Nano: Either by utilizing the MPLAB X IDE or Microchip Studio Kit Window or by looking at the sticker on the bottom of the PCB.

The Kit Window will pop up when connecting AVR64EA48 Curiosity Nano to a computer with MPLAB X IDE or Microchip Studio running. The first six digits of the serial number, listed under kit information, contain the product identifier and revision.

Tip: If closed, the Kit Window can be opened in MPLAB X IDE through the menu bar **Window** > **Kit Window**.

The same information is found on the sticker on the bottom side of the PCB. Most boards will have the identifier and revision printed in plain text as A09-nnnn\rr, where “nnnn” is the identifier and “rr” is the revision.

The serial number string has the following format:

```
"nnnnrrssssssssss"
n = product identifier
r = revision
s = serial number
```

The product identifier for AVR64EA48 Curiosity Nano is A09-3428.

5.2 Revision 3

Revision 3 is the initially released board revision

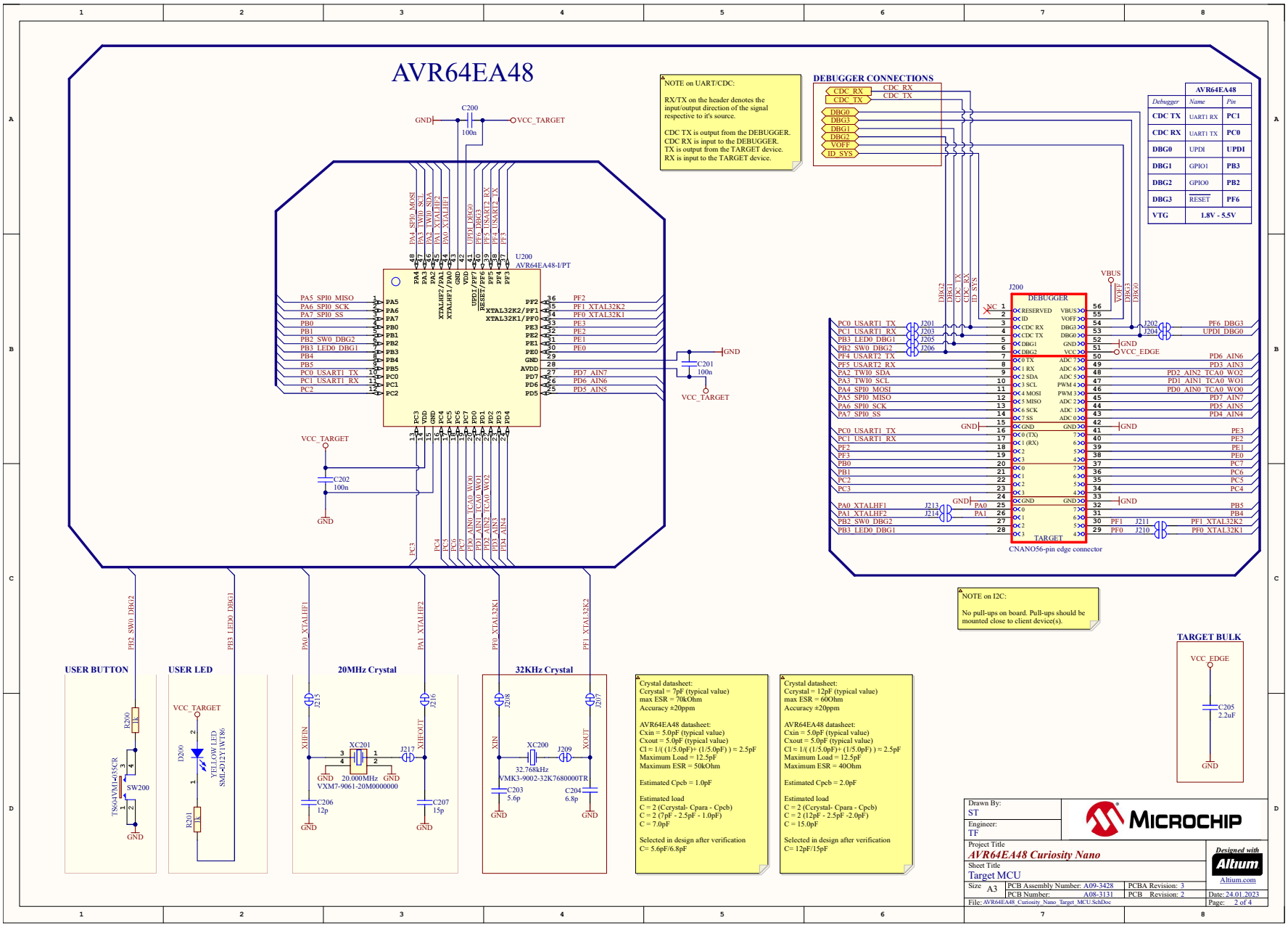
6. Document Revision History

Doc. Rev.	Date	Comments
A	02/2023	Initial document release

7. Appendix

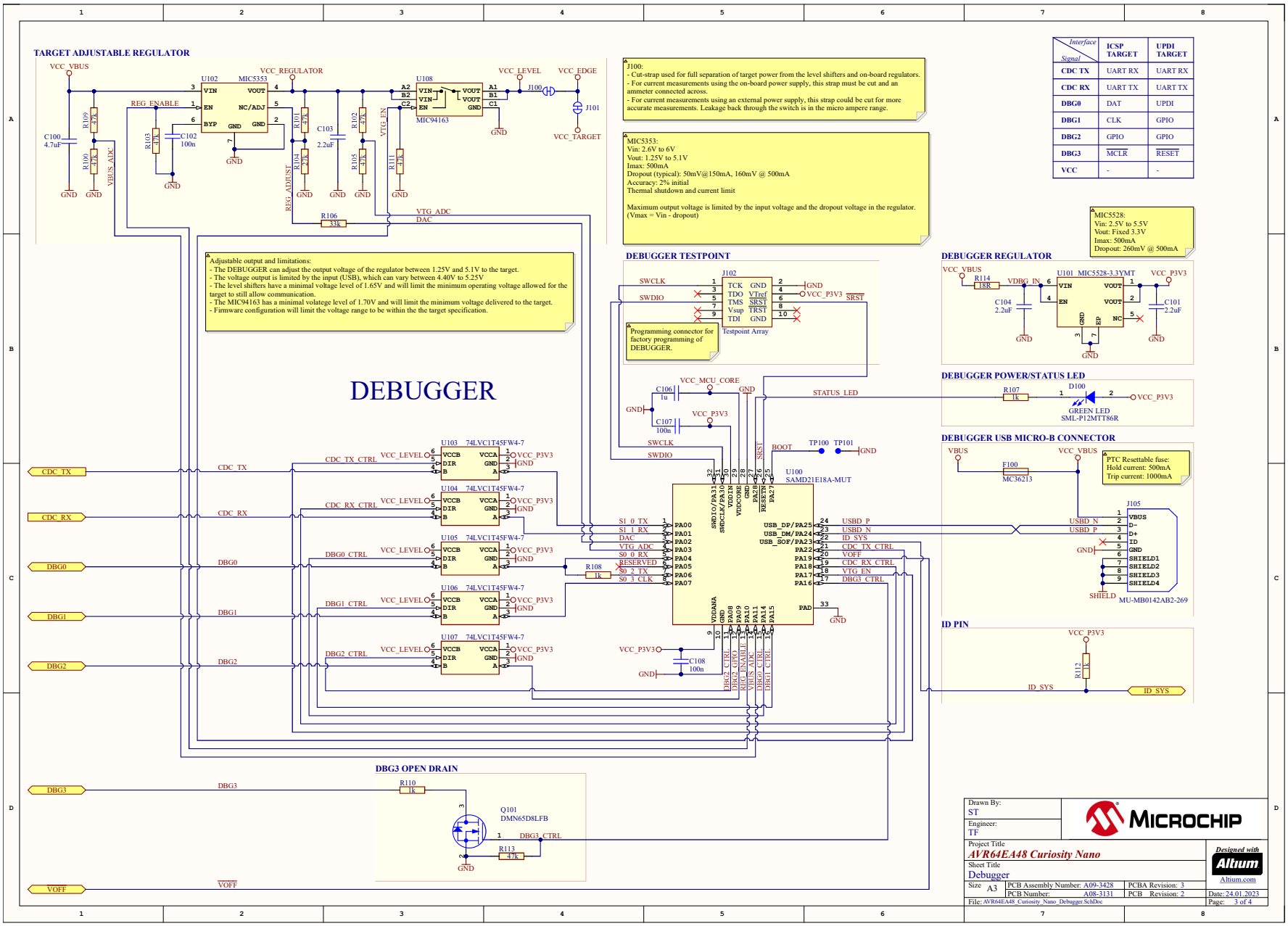
7.1 Schematic

Figure 7-1. AVR64EA48 Curiosity Nano MCU Schematic



7.1 Schematic

Figure 7-2. AVR64EA48 Curiosity Nano Debugger Schematic



7.2 Assembly Drawing

Figure 7-3. AVR64EA48 Curiosity Nano Assembly Drawing Top

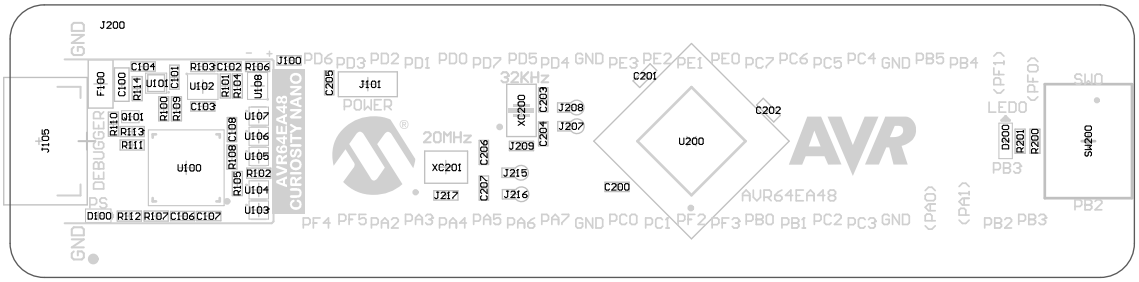
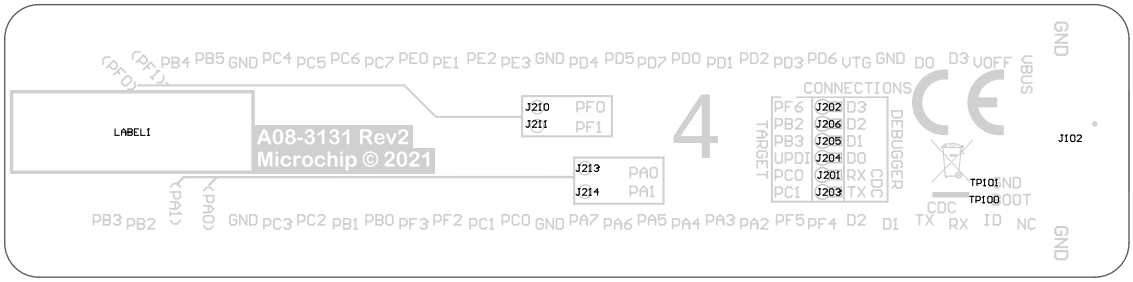
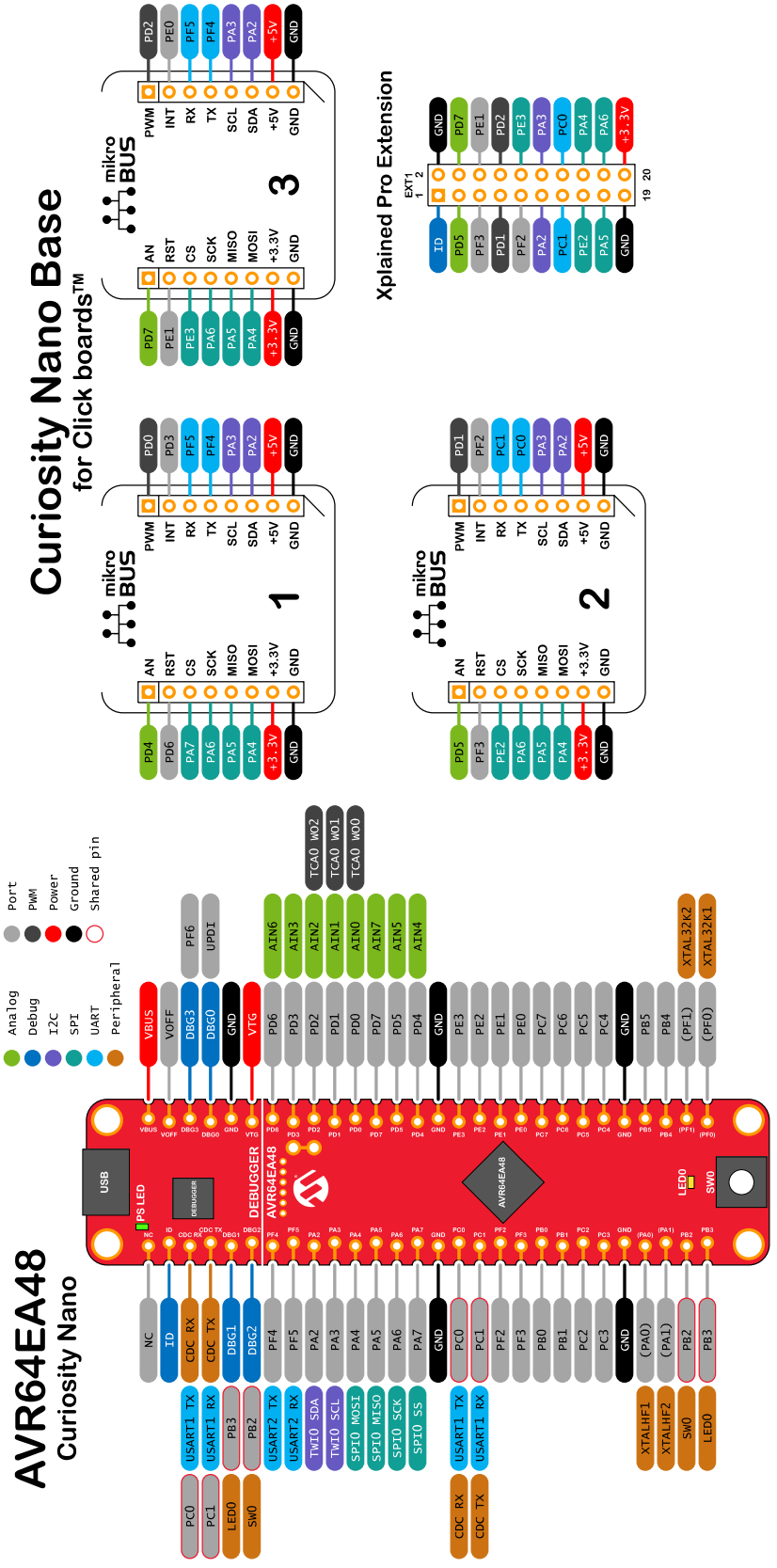


Figure 7-4. AVR64EA48 Curiosity Nano Assembly Drawing Bottom



7.3 Curiosity Nano Base for Click boards™

Figure 7-5. AVR64EA48 Curiosity Nano Pinout Mapping



7.4 Disconnecting the On-Board Debugger

The on-board debugger and level shifters can be completely disconnected from the AVR64EA48.

The block diagram below shows all connections between the debugger and the AVR64EA48. The rounded boxes represent connections to the board edge. The signal names shown are also printed in silkscreen on the bottom side of the board.

To disconnect the debugger, cut the straps shown in Figure 7-7.

Attention: Cutting the GPIO straps to the on-board debugger will disable the virtual serial port, programming, debugging, and data streaming. Cutting the power supply strap will disconnect the on-board power supply.

Tip: Reconnect any cut connection by using solder. Alternatively, mount a 0Ω 0402 resistor.

Tip: When the debugger is disconnected, an external debugger can be connected to holes, as shown in Figure 7-7. 3.6. Connecting External Debuggers describes how to connect an external debugger.

Figure 7-6. On-Board Debugger Connections Block Diagram

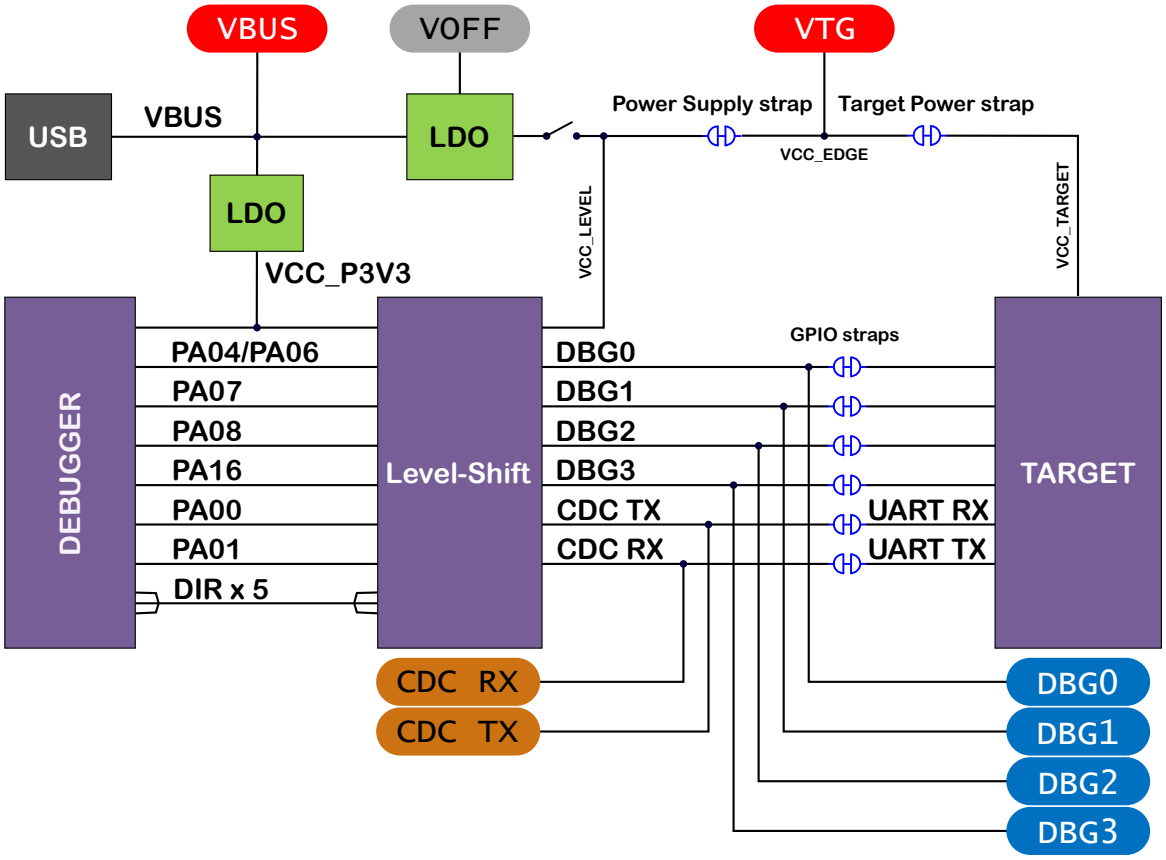
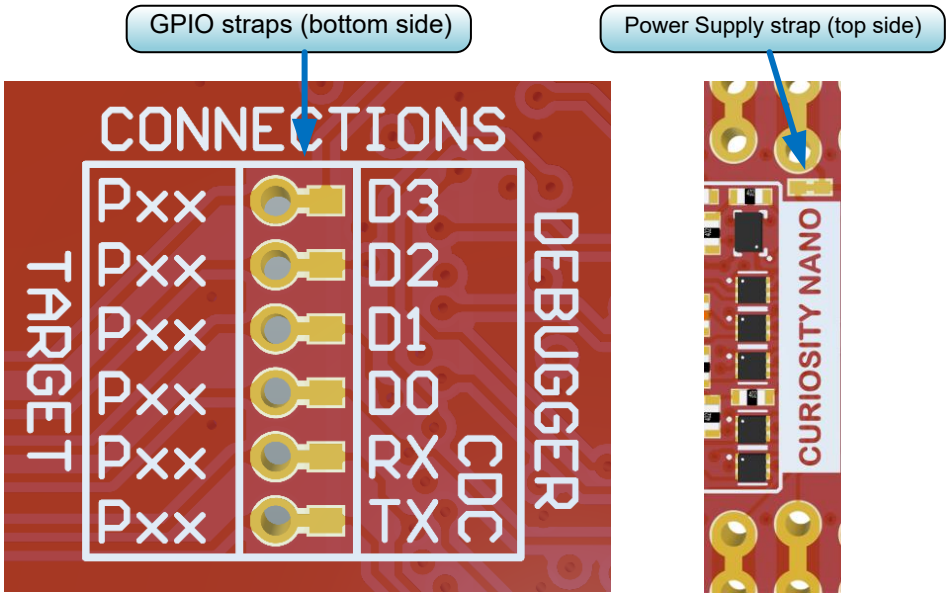


Figure 7-7. On-Board Debugger Connection Cut Straps



7.5 Getting Started with IAR™

IAR Embedded Workbench® for AVR® is a proprietary high-efficiency compiler not based on GCC. Programming and debugging of AVR64EA48 Curiosity Nano is supported in IAR™ Embedded Workbench for AVR using the Atmel-ICE interface. To get the programming and debugging to work, some initial settings must be set up in the project.

The following steps will explain how to get the project ready for programming and debugging:

1. Make sure that the project to be configured is opened. Open the **OPTIONS** dialog for the project.
2. In the category **General Options**, select the **Target** tab. Select the device for the project, or if not listed, the core of the device, as shown in Figure 7-8.
3. In the category **Debugger**, select the **Setup** tab. Select **Atmel-ICE** as the driver, as shown in Figure 7-9.
4. In the category **Debugger** > **Atmel-ICE**, select the **Atmel-ICE 1** tab. Select **UPDI** as the interface. Optionally select the **UPDI** frequency, as shown in Figure 7-10.

Info: If the selection of Debug Port (mentioned in step 4) is grayed out, the interface is preselected, and the user can skip this configuration step.

Figure 7-8. Select Target Device

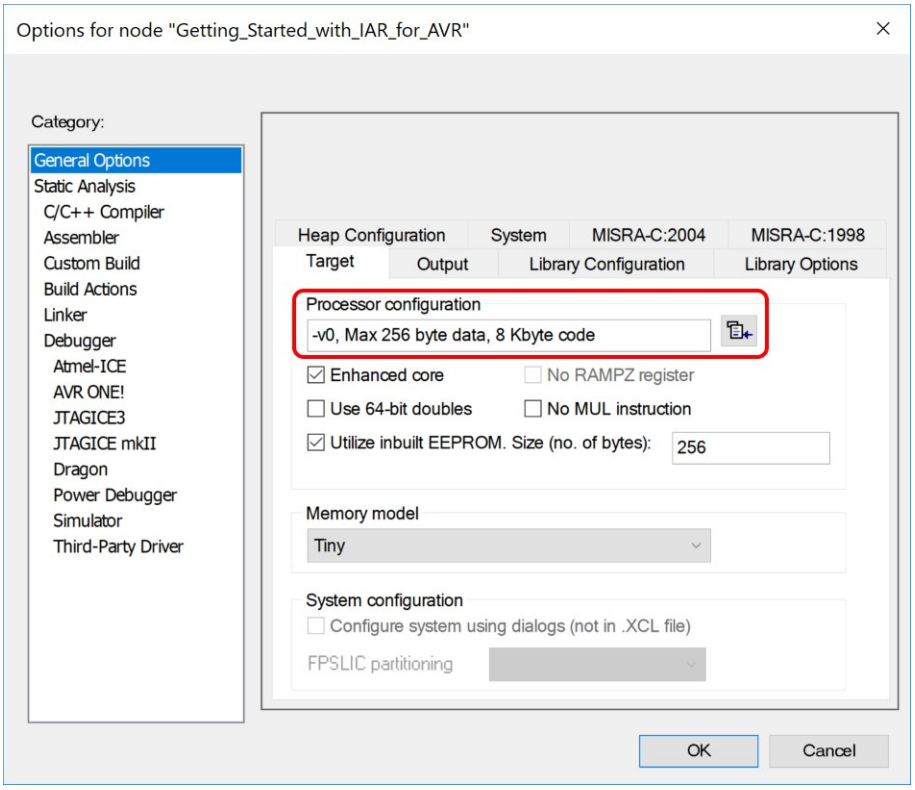


Figure 7-9. Select Debugger

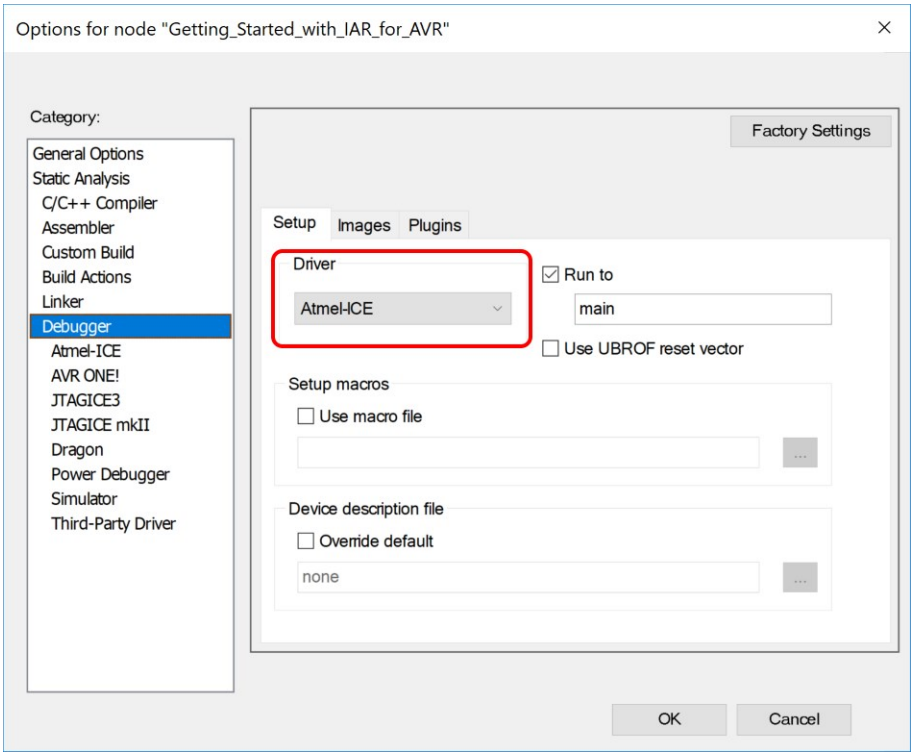
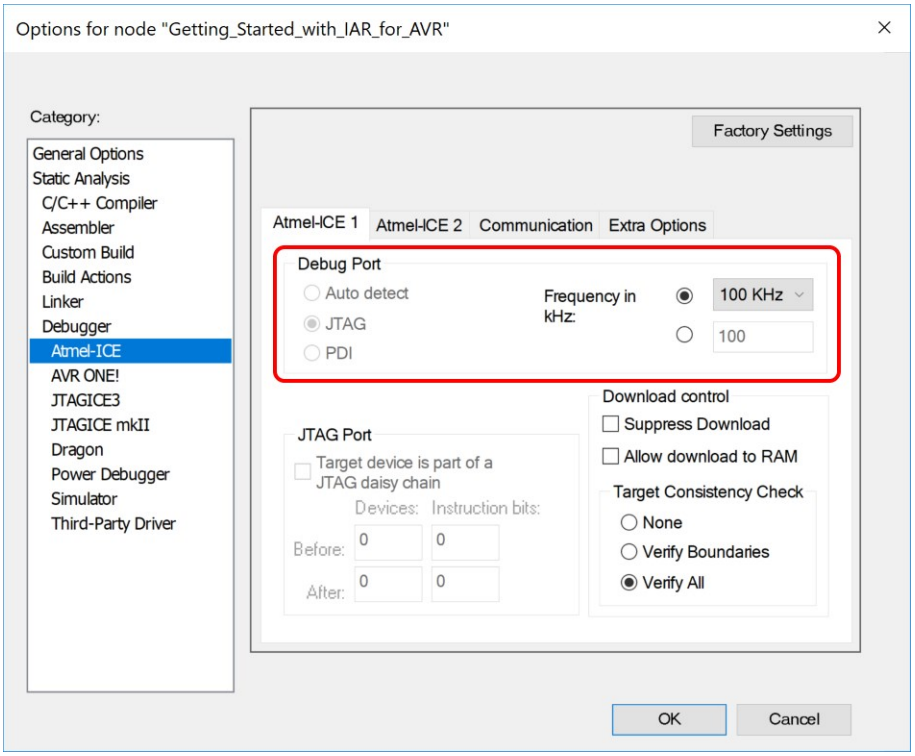


Figure 7-10. Configure Interface



Microchip Information

The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable". Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded

by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP "AS IS". MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP'S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, CryptoMemory, CryptoRF, dsPIC, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, Flashtec, Hyper Speed Control, HyperLight Load, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet- Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, TrueTime, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, Augmented Switching, BlueSky, BodyCom, Clockstudio, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, Espresso T1S, EtherGREEN, GridTime, IdealBridge, In-Circuit Serial Programming, ICSP, INICnet, Intelligent Paralleling, IntelliMOS, Inter-Chip Connectivity, JitterBlocker, Knob-on-Display, KoD, maxCrypto, maxView, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, RTAX, RTG4, SAM-ICE, Serial Quad I/O, simpleMAP, SimpliPHY, SmartBuffer, SmartHLS, SMART-I.S., storClad, SQI, SuperSwitcher, SuperSwitcher II, Switchtec, SynchroPHY, Total Endurance, Trusted Time, TSHARC, USBCheck, VariSense, VectorBlox, VeriPHY, ViewSpan, WiperLock, XpressConnect, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

ISBN: 978-1-6683-1942-0

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.