AN3344

SAMA5D27 WLSOM1 Auto-Test

Scope

This application note describes how to use the unitary Auto-Test software.

This software provides the customer with information about the presence and functionality of the devices embedded in the SAMA5D27-WLSOM1.

It can be used by the customer if an embedded device malfunction is suspected.

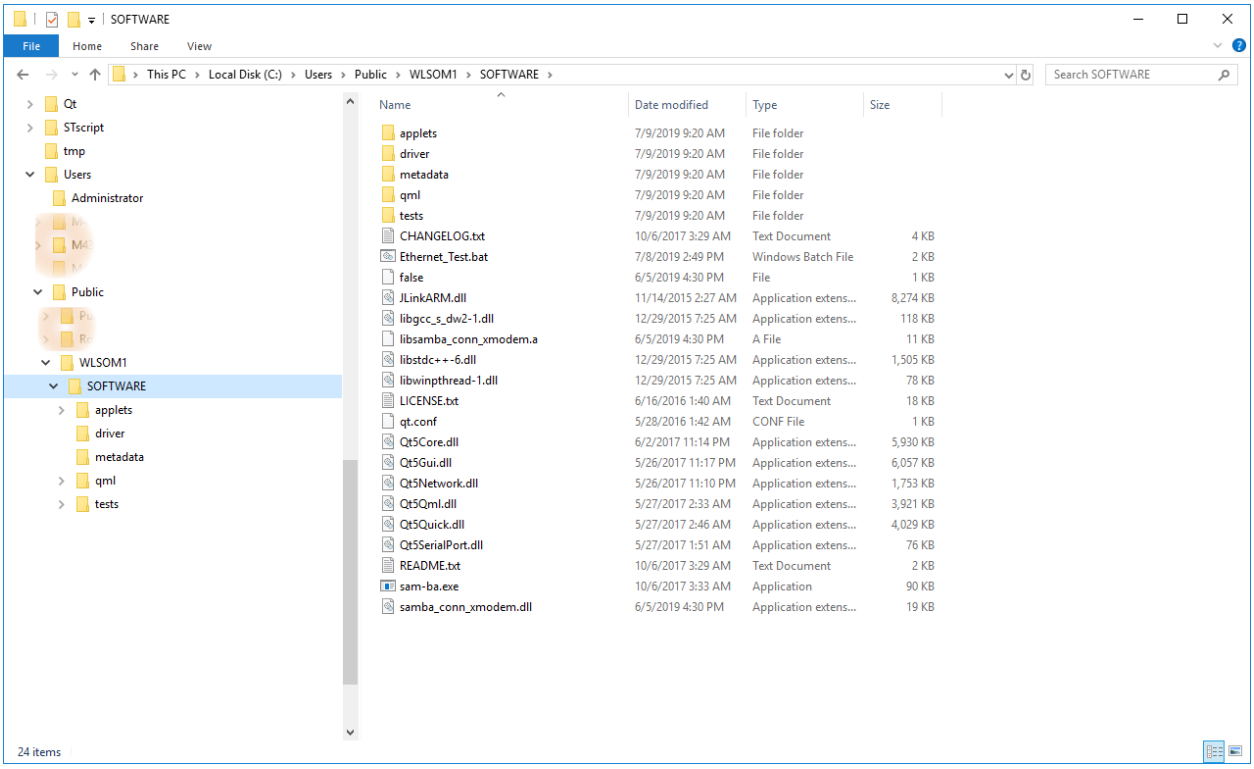
Table of Contents

1. Where to Store the Auto-Test Package

The Auto-Test software package comprises several folders and one executable file. It is important to respect the folder/file hierarchy to ensure correct functionality of the SAM-BA® tool.(1)

First, select a folder in the local workspace and copy the zip file into it, then unzip the Auto-Test software package. This folder must be named "%*DIR_PATH%*" for all SAM-BA commands that are generated.

The folder is shown below, with, as an example, the pathname %DIR_PATH% = c:\Users\Public\WLSOM1\SOFTWARE.



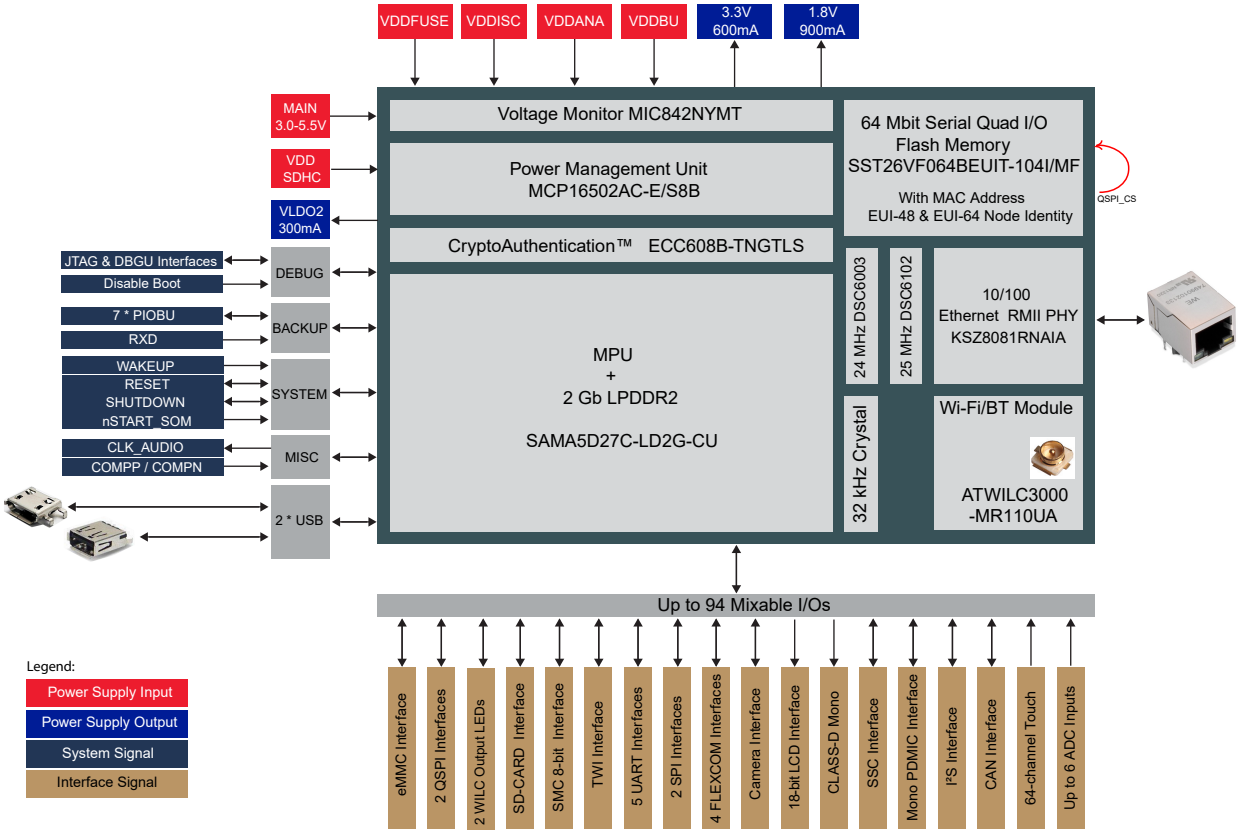
Note:

1. SAM-BA Version 3.2.1, delivered with this package, is a programming tool to launch applets for Microchip MPUs. All APIs/command line options are subject to change in any new version of SAM-BA. Only SAM-BA Version 3.2.1 should be used with the Auto-Test software; no other version is applicable.
   No other software is required. All relevant software is available in the package delivery.

2. Block Diagram

The block diagram of the SAMA5D27-WLSOM1 module shown below lists all the devices available on the module.

Test procedures are described in the following sections and describe how to verify interconnects between devices.



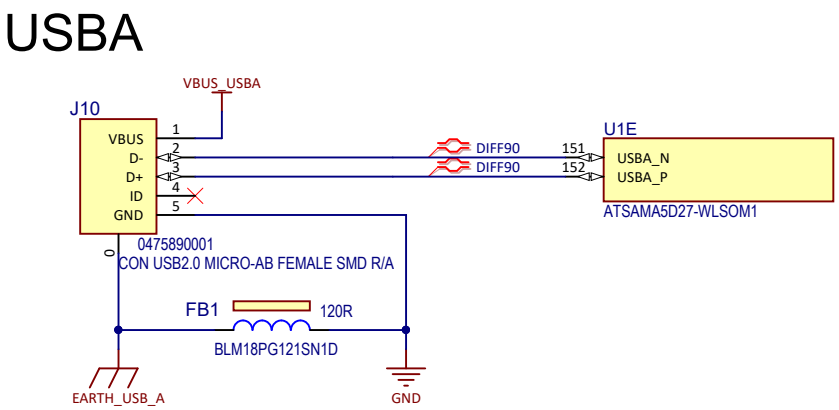
3. How to Configure the Communication Interfaces

3.1 Where to Connect the USB Interface

3.1.1 Hardware Connection

The SAMA5D27-WLSOM1 module has a USB device interface available on pins 151 and 152 (USBA_N and USBA_P, respectively). The figure below shows how the USB interface is connected.

Figure 3-1. USB Connection Schematic



This interface is required for SAM-BA to communicate with the SAMA5D27 to launch Auto-Test applets when the system enters SAM-BA Monitor. For more details about boot strategy, refer to the section “Standard Boot Strategies” in the SAMA5D2 Series data sheet.

3.1.2 Embedded Software Prerequisites

The Auto-Test software package is a standalone tool. In order to use the Auto-Test software package, it is necessary

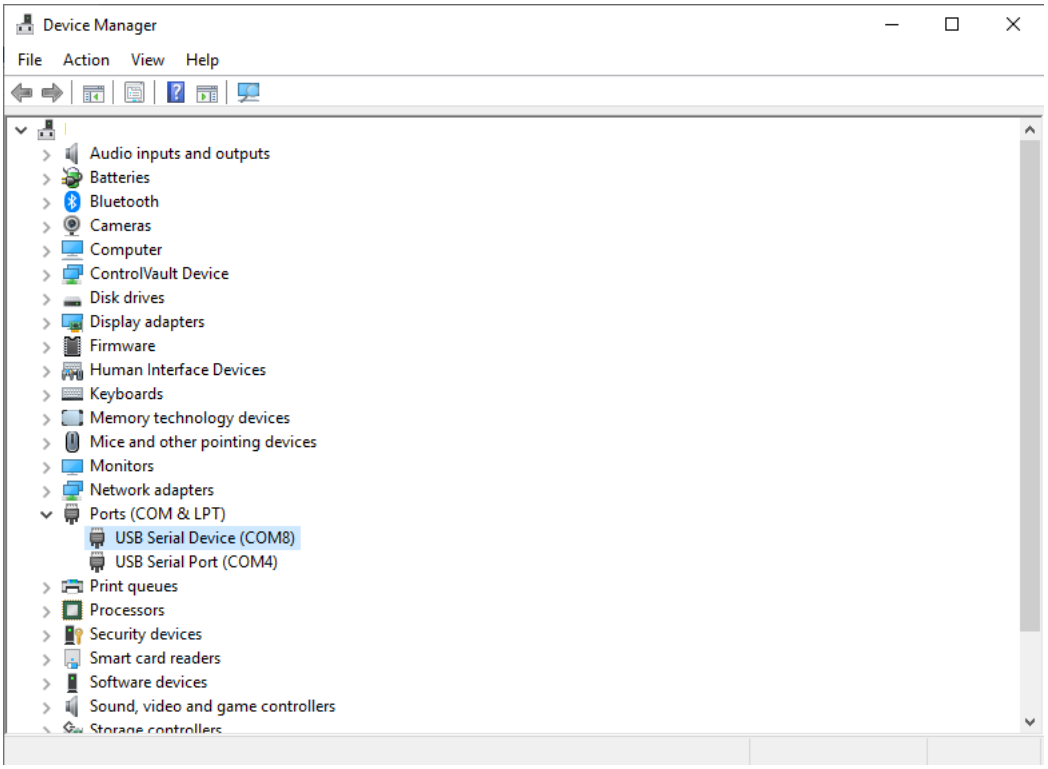
1. to check that there is no valid boot program stored in any external memory,
2. to check that no firmware is installed in the application,
3. to configure SAM-BA Monitor to allow execution of applets.

3.1.3 Updating the Windows Driver

If the USB device is not recognized by the operating system, the driver must be updated. To do so, follow the steps below:

1. Go to “Device Manager” in your control panel.
2. Select the properties of the “Unknown Device” that appears when the USB device is connected.
3. Update the driver by selecting the file “atm6124_cdc.inf” available in the folder %DIR_PATH% \SOFTWARE\driver\.

The figure below shows how the interface is seen by the operating system.



3.2 Where to Connect the Debug UART Console

The Debug UART console can be connected through seven different I/O groups.

By default (factory configuration), the UART selected is "*UART 1 - IOSET 1*". There are six other configurations available and selectable, depending on the system and board design of the customer application.

3.3 Factory Configuration

By default, the Debug UART console selected is "*UART 1 - IOSET 1*".

The following PIOs are used :

- PD2: URXD1 (UART Receive Data) – Must be connected to the Terminal TXD node.
- PD3: UTXD1 (UART Transmit Data) – Must be connected to the Terminal RXD node.

3.4 How to Customize the Debug UART Console Configuration

It may be necessary to reassign the Debug UART console if the design implementation differs from the default.

The PIO pairs that can be used as the Debug UART console on the SAMA5D2 are listed below:

- PB26, PB27: Corresponds to "*UART 0 - IOSET 1*" (Evaluation Kit default configuration)
- PD2, PD3: Corresponds to "*UART 1 - IOSET 1*" (Factory configuration)
- PD4, PD5: Corresponds to "*UART 2 - IOSET 1*"
- PC12, PC13: Corresponds to "*UART 3 - IOSET 1*"
- PC31, PD1: Corresponds to "*UART 3 - IOSET 2*"
- PB11, PB12: Corresponds to "*UART 3 - IOSET 3*"
- PB3, PB4: Corresponds to "*UART 4 - IOSET 1*"

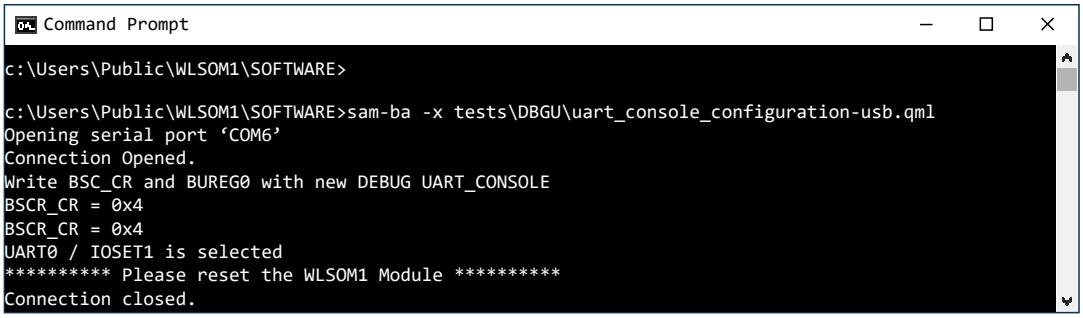
To select the appropriate Debug UART console, follow the steps below:

1. In a text editor, open the file "*uart_console_configuration-usb.qml*" located in the folder : "*%DIR_PATH%\SOFTWARE\tests\DBGU*".
2. Remove star-dash comments (/* … */) on the line to be modified. The highlighted text below shows an example for *UART2-IOSET1* selection.

```
import QtQuick 2.3
import SAMBA 3.2
import SAMBA.Connection.Serial 3.2
import "../SAMA5D2_IOSET.js" as Pioloset

Item {
  SerialConnection {
    id: connection
      onConnectionOpened: {
        var result
        var result_uart
        var result_ioset
        ....
        ....
        // ----------------------------
        // UART 2 - IOSET 1 Selected
        // Remove /*  */ if this configuration is required
        // ----------------------------

        /*Pioloset.uart2_ioset1_write(this)*/

        // ----------------------------
        // UART 3 - IOSET 1 Selected
        ...
        ...
        }
      }
    Component.onDestruction: connection.close()
}
```

```
import QtQuick 2.3
import SAMBA 3.2
import SAMBA.Connection.Serial 3.2
import "../SAMA5D2_IOSET.js" as Pioloset

Item {
  SerialConnection {
    id: connection
      onConnectionOpened: {
        var result
        var result_uart
        var result_ioset
        ....
        ....
        // ----------------------------
        // UART 2 - IOSET 1 Selected
        // Remove /*  */ if this configuration is required
        // ----------------------------

        Pioloset.uart2_ioset1_write(this)

        // ----------------------------
        // UART 3 - IOSET 1 Selected
        ...
        ...
        }
      }
    Component.onDestruction: connection.close()
}
```

3. Save and close the file. No compilation is required.
4. In the Command Prompt window, execute the following command :
   – "*sam-ba -x tests\DBGU\uart_console_configuration-usb.qml*" from the folder "*%DIR_PATH%\SOFTWARE*" .
5. The following instructions appear in the Command Prompt window (the example shows the "*UART0-IOSET1*" configuration):



For the selection to take effect, the WLSOM1 must be reset and VDDBU voltage must be present. Reset can be done in two different ways:

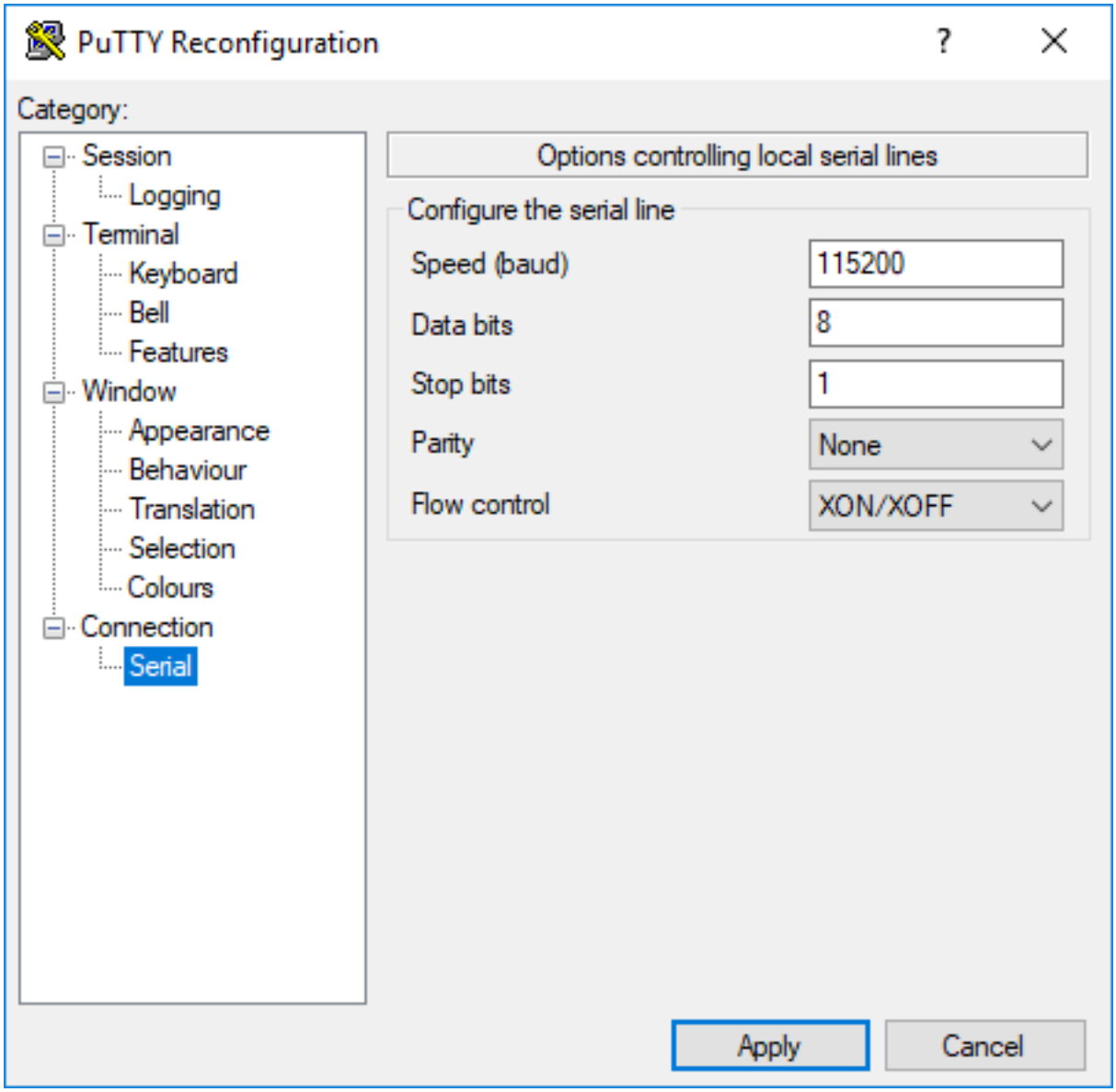
- By executing a hardware reset by shorting the NRST pin to ground.
- By executing the following command from "*%DIR_PATH%\SOFTWARE*" folder: "*sam-ba -x tests\SWRST\software_reset-usb.qml*".

3.5 Configuring the UART Terminal Interface (PC Side)

A Terminal application is used to dialog with SAM-BA Monitor embedded in the SAMA5D2 and observe data transfers from the WLSOM1 through the Debug UART interface.

A USB serial converter cable can be used to monitor data received from the WLSOM1. An example cable reference is "*TTL-232Rxxx*" from FTDI.

To configure the communication protocol, launch a Terminal Interface (such as PuTTY) and configure "*USB Serial Port (COMx)*" as follows:



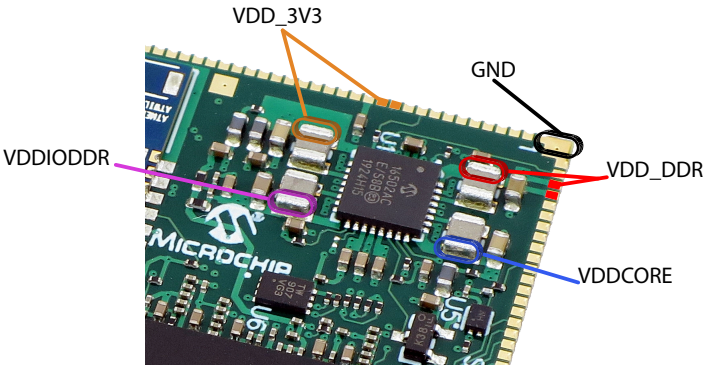
4. Failure Analysis

Failure analysis is described in the procedure below. Details on these steps are provided in the sections that follow.

1. Perform a visual inspection of the SAMA5D27-WLSOM1 module and verify that the soldering of the pads is correct.
2. Connect the module to the debug interfaces USB and UART (see How to Configure the Communication Interfaces).
3. Power the supply of the WLSOM1 module.(1)
4. Connect a PC terminal console as described above and perform a Power-On action.
   - "RomBOOT" appears on the Debug UART console.
   - The USB Device is mounted as "USB Serial Device".(2)
5. Measure the voltage on the WLSOM1 test points(3). The results should be:
   - VDD_3V3=3.3V ± 2%
   - VDDIODDR=1.2V ± 2%
   - VDDCORE=1.25V ± 2%
   - VDD_DDR=1.8V ± 2%
6. Verify the version of the SAMA5D27 microprocessor (MPU).
7. Verify 24 MHz clock input.
8. Verify the LPDDR2 memory.
9. Verify the QSPI memory.
10. Verify the ATECC608 memory.
11. Verify the WILC3000 interconnect.
12. Verify the Ethernet interconnect.
13. Verify the 32 kHz crystal.
14. Verify the PMIC version.
15. Verify any PIO.
16. Run on Linux and perform other tests.

Notes:

1. It is important that the system does not boot on an external memory before powering up the module. For more details, see WLSOM1 Embedded Software Prerequisites.
2. If the device is not recognized by the operating system, the system driver must be updated. See Windows Driver.
3. To perform PMIC voltage measurements, check the WLSOM1 output voltage measurement points illustrated in the figure below.



5. Test Procedures

5.1 Prerequisites

Before running the scripts detailed in the following sections, the Debug UART console must be modified for data output.

If the Debug UART console configuration was customized as described in How to Customize the Debug UART Console Configuration, the file "*SAMA5D27WLSOM1PROD.qml*" available in the folder "*%DIR_PATH%\SOFTWARE\qml\Tests*" must also be updated.

To do so, follow the steps below:

1. Recall the UART/IOSET that was selected when the script "*uart_console_configuration-usb.qml*" was run (see How to Customize the Debug UART Console Configuration).
2. Open the file "*SAMA5D27WLSOM1PROD.qml*".
3. Modify the lines "*instance:*" and "*ioset:*" with the indexes of the UART/IOSET from Step 1.
4. Save and close the file. No compilation is required.

An example with UART3/IOSET2 selected is given below:

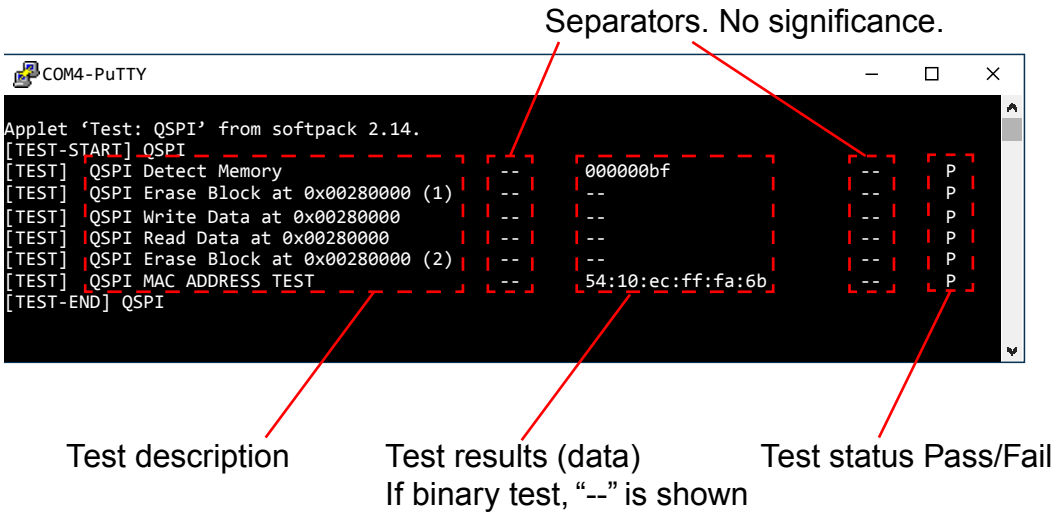
```
import QtQuick 2.3
import SAMBA 3.2
import SAMBA.Device.SAMA5D2 3.2
SAMA5D2 {
    name: "sama5d27-wlsom1-prod"
    description: "SAMA5D27-WLSOM1_PROD_BOARD"
    applets: [
                TestEcc608Applet {
                .....
                entryAddr: 0x220000
                }
    ]
    config {
        serial {
            instance: 1 /* UART1 */
            ioset: 1
        }
    }
}
```

```
import QtQuick 2.3
import SAMBA 3.2
import SAMBA.Device.SAMA5D2 3.2
SAMA5D2 {
    name: "sama5d27-wlsom1-prod"
    description: "SAMA5D27-WLSOM1_PROD_BOARD"
    applets: [
                TestEcc608Applet {
                .....
                entryAddr: 0x220000
                }
    ]
    config {
        serial {
            instance: 3 /* UART3 */
            ioset: 2
        }
    }
}
```

5.2 How to Interpret Results

Test results are returned through the UART interface. As illustrated in the screen shot below, the results are composed of:

- A test description
- A separator
- Test results: Data or separator when binary test is executed
- A separator
- A test status Pass/Fail (P/F)

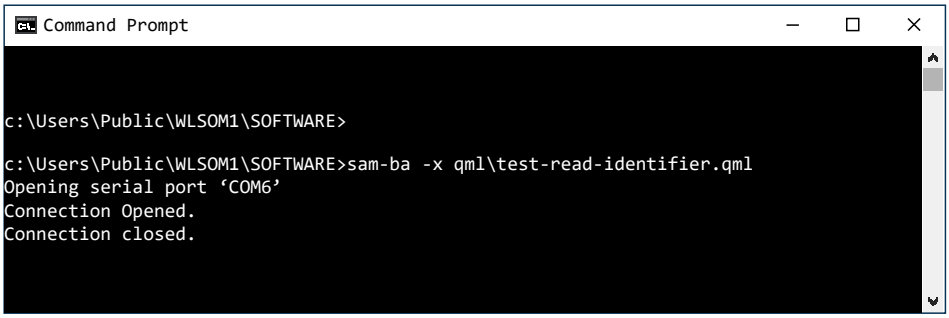


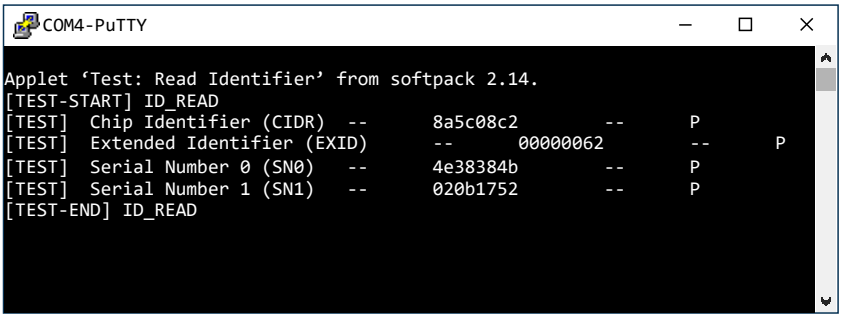
5.3 Checking the SAMA5D27 Microprocessor

The WLSOM1 module features a SAMA5D27 microprocessor (MPU).

To check the version of the SAMA5D27 being used, follow the steps below:

1. From the folder "*%DIR_PATH%\SOFTWARE*", launch the following command:
   – sam-ba -x qml\test-read-identifer.qml
2. The two code fragments below appear in the Command Prompt window and in the UART Debug interface.





The tests are passed when the Terminal Window displays:

- Chip Identifier: 8a5c08c2 or 8a5c08c4
  Note: Two possible chip identifiers can be displayed depending on the chip set version.
- Extended Identifier: 00000062
- SN0 and SN1 Serial Numbers: as 32-bit words
- Each [TEST] line: "P" status

5.4 Testing the 24 MHz MEM Clock Generator

The presence of the 24 MHz clock can be tested by redirecting it to an external GPIO. There are four possibilities:

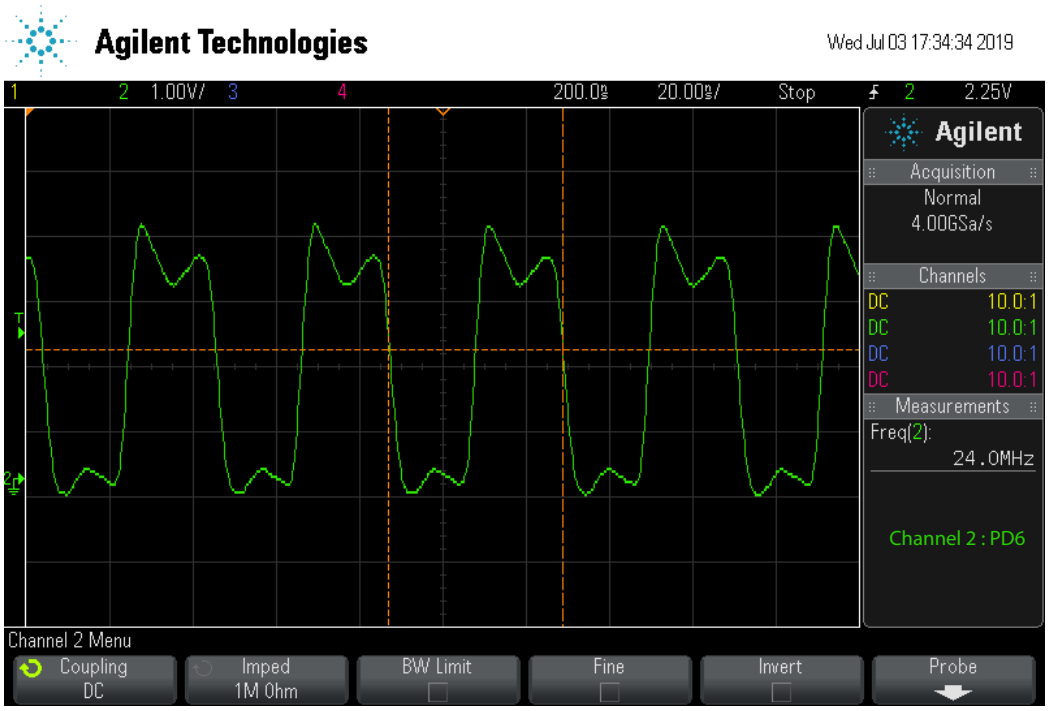
- Redirect the 24 MHz clock to PD6 (Pin 106):
  - Launch "*sam-ba -x tests\24MHz\pck1_ioset1_pd6.qml*"
- Redirect the 24 MHz clock to PC27 (Pin 35):
  - Launch "*sam-ba -x tests\24MHz\pck1_ioset2_pc27.qml*"
- Redirect the 24 MHz clock to PC28 (Pin 39):
  - Launch "*sam-ba -x tests\24MHz\pck2_ioset1_pc28.qml*"
- Redirect the 24 MHz clock to PD11 (Pin 118):
  - Launch "*sam-ba -x tests\24MHz\pck2_ioset2_pd11.qml*"

Then probe the selected GPIO to determine if the 24 MHz clock is present.

After testing, a device reset is required in order to return the GPIO to its default state. There are two ways to do this:

1. Perform an external hardware reset by shorting the NRST signal to ground.
2. From the folder "*%DIR_PATH%\SOFTWARE*", launch "*sam-ba -x tests\24MHz\pck_ioset_reset-usb.qml*"

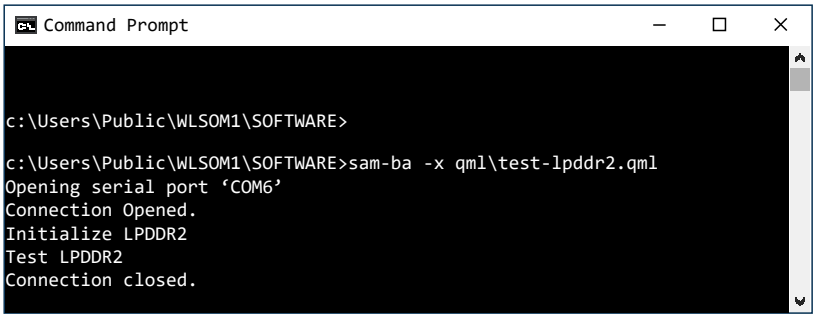
An example of the 24 MHz output signal measured on PD6 is shown below:

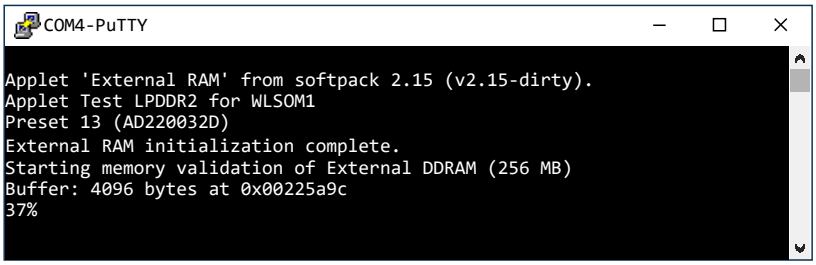


5.5 Testing the LPDDR2

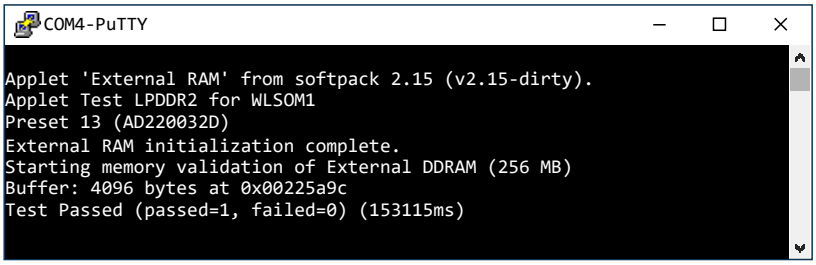
The module embeds a System-in-Package chip which contains a SAMA5D27 MPU and the LPDDR2 memory AD220032D. The following script checks the LPDDR2 memory connection to the MPU.

1. From the folder "*%DIR_PATH%\SOFTWARE*", launch the following command:
   - sam-ba -x qml\test-lpddr2.qml
2. The two code fragments below appear in the Command Prompt window and in the UART Debug interface.





3. The LPDDR2 memory integrity is good if test attempt results are completed and if the write process attempt results are "Test Passed", as shown below:

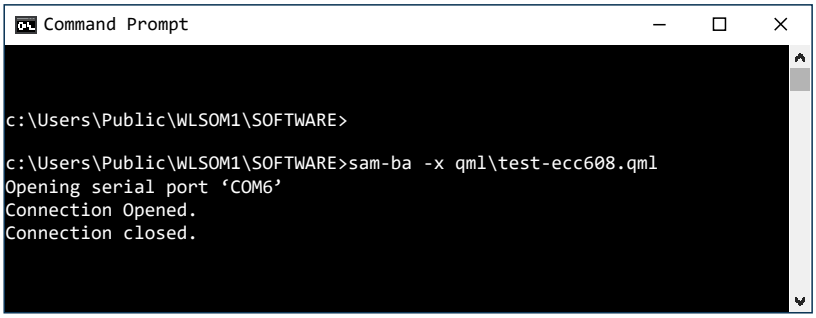


4. If the test is stuck, the only way to return to a valid state is to perform a hardware reset by tying the NRST signal to ground.

5.6 Testing the Secure Element

A secure element, ATECC608B-TNGTLS, is connected to the MPU through an I²C interface. The following script checks the presence of the secure element.

1. From the folder "*%DIR_PATH%\SOFTWARE*", launch the following command:
   – sam-ba -x qml\test-ecc608.qml
2. The two code fragments below appear in the Command Prompt window and in the UART Debug interface.



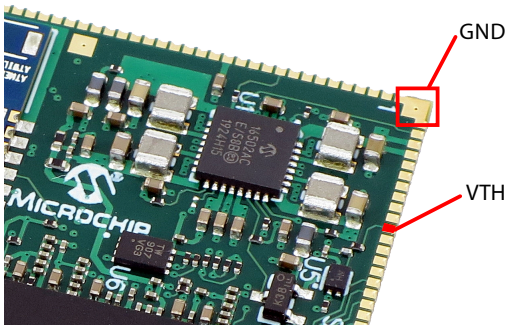
```
COM4-PuTTY
Applet 'Test: ECC608' from softpack 2.14 (v2.14-95-g7c30ef6f-dirty).
[TEST-START] ECC608
[TEST] ECC608 Wake up address     --     00000060     --     P
[TEST] ECC608 Check Data Lock     --     00000001     --     P
[TEST] ECC608 Check Config Lock   --     00000001     --     P
[TEST] ECC608 Read Serial Number  --     --           --     P
SN: 0x01 0x23 0x8E 0x9C 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
[TEST-END] ECC608
```

3. When the four tests are passed, indicated by a 'P' as shown in the screen shot above, the connection between the two devices is good.
4. If the test is stuck, the only way to return to a valid state is to perform a hardware reset by tying the NRST signal to ground.

5.7 Testing the Voltage Monitor

The module embeds the precision voltage comparator MIC842NYMT for power failure detection. The script checks whether power failure detection is working correctly.

The VTH input pin is available on pin 178 of the SAMA5D27-WLSOM1 module. The image below shows the location.



To test the detection, launch the script and apply a voltage lower than 1.24V on the pin VTH.

1. From the folder "*%DIR_PATH%\SOFTWARE*", launch the following command:
   – sam-ba -x tests\VOLT_MON\test_volt_mon-usb.qml
2. The code fragment below appears in the Command Prompt window.

```
Command Prompt


c:\Users\Customer\Desktop\WLSOM1\SOFTWARE>

c:\Users\Customer\Desktop\WLSOM1\SOFTWARE>sam-ba -x tests\VOLT_MON\test_volt
Opening serial port 'COM6'
Connection Opened.
PD31 is set in input
PD31 is read at 1
.
.
```

3. The polling timing of the PD31 GPIO can be customized by changing some parameters in the qml file. By default, the polling time is 60 seconds. After this time, the script stops its process and closes the connection with the target.
4. By increasing the parameter "*number_of_io_toggling*", this timing can be increased. If the variable *number_of_io_toggling* is equal to '1000', it corresponds to an approximate polling window of one minute. Make the change if necessary and save the file. No compilation is required.
   An example with "*number_of_io_toggling*" set to two minutes is given below:

```
Item {
        SerialConnection {
                id: connection
                onConnectionOpened: {

                        var result
                        var delay_time = 50

                        // 1000 io_toggling time is approximately equal to 1 minute
                        var number_of_io_toggling = 1000  // Here set at ~= 1 minute

                        // Set VTH input and measure it to 1 (PD31)
                        var input_pio_array = [3]
                        var input_pin_array = [31]
```

```
Item {
        SerialConnection {
                id: connection
                onConnectionOpened: {

                        var result
                        var delay_time = 50

                        // 1000 io_toggling time is approximately equal to 1 minute
                        var number_of_io_toggling = 2000  // Here set at ~= 2 minutes

                        // Set VTH input and measure it to 1 (PD31)
                        var input_pio_array = [3]
                        var input_pin_array = [31]
```

5.8 Testing the QSPI Memory

A Quad SPI memory, SST26VF064BEUIT-104I/MF, is connected to the MPU through a QSPI interface. The following script checks the presence of the QSPI memory.

1. From the folder "*%DIR_PATH%\SOFTWARE*", launch the following command:
   – sam-ba -x qml\test-qspi.qml
2. The two code fragments below appear in the Command Prompt window and in the UART Debug interface:

```
Command Prompt

c:\Users\Public\WLSOM1\SOFTWARE>

c:\Users\Public\WLSOM1\SOFTWARE>sam-ba -x qml\test-qspi.qml
Opening serial port 'COM6'
Connection Opened.
Connection closed.
```

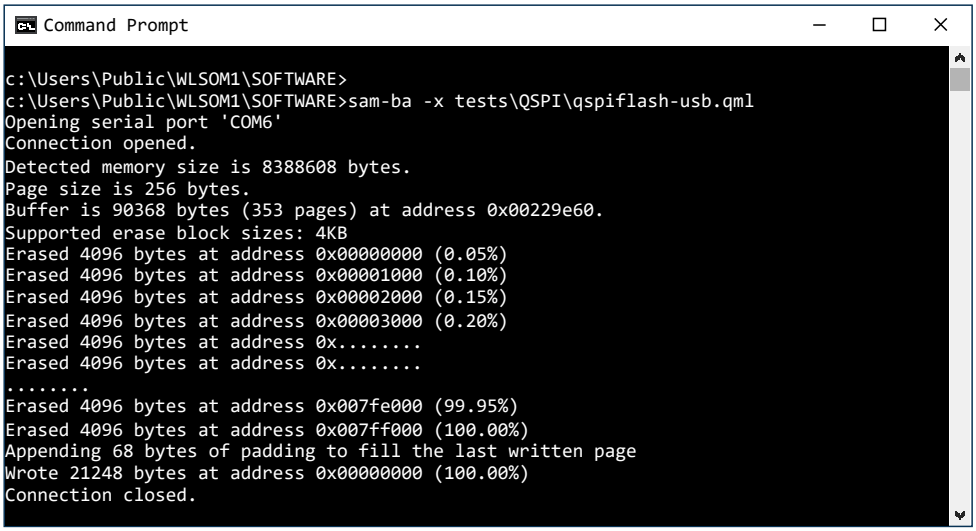
```
COM4-PuTTY

Applet 'Test: QSPI' from softpack 2.14.
[TEST-START] QSPI
[TEST]  QSPI Detect Memory      --      000000bf        --      P
[TEST]  QSPI Erase Block at 0x00280000 (1)      --      --      --      P
[TEST]  QSPI Write Data at 0x00280000   --      --      --      P
[TEST]  QSPI Read Data at 0x00280000    --      --      --      P
[TEST]  QSPI Erase Block at 0x00280000 (2)      --      --      --      P
[TEST]  QSPI MAC ADDRESS TEST   --      54:10:ec:ff:fa:6b       --      P
[TEST-END] QSPI
```

3. When the six tests are passed, indicated by 'P' as shown in the screen shot above, the connection between the two devices is good.

Another script is available that checks the QSPI memory integrity. To perform this test, the QSPI memory must be empty or must contain only elements that are not important for the customer application.

Important: This test erases all data currently contained in the QSPI memory.

1. From the folder "*%DIR_PATH%\SOFTWARE*", launch the following command
   – sam-ba -x tests\QSPI\qspiflash-usb.qml
2. The following code fragment appears in the Command Prompt window:



3. QSPI integrity is good if test attempt results are 100% and if the write process attempt results are 100%.

5.9 Testing the Ethernet PHY

5.9.1 Configure the Ethernet Script

Before running the Ethernet script, the Debug UART console must be modified for data output.

If the Debug UART console configuration was customized as described in How to Customize the Debug UART Console Configuration, the file "*Ethernet_Test.bat*" available in the folder "*%DIR_PATH%\SOFTWARE*" must also be updated.

To do so, follow the steps below:

1. Recall the UART/IOSET that was selected when the script "*uart_console_configuration-usb.qml*" was run.
2. Open the file "*Ethernet_Test.bat*" with a text editor.
3. Add the comment "rem " at the beginning of the line with UART1-IOSET1 configuration. This selection is the factory default configuration and must be commented out to enable the case. See the result highlighted in the example below.
4. Remove the comment "rem " on the line with the UART/IOSET from Step 1.
5. Save and close the file. No compilation is required.

An example with UART3/IOSET2 selected is given below:

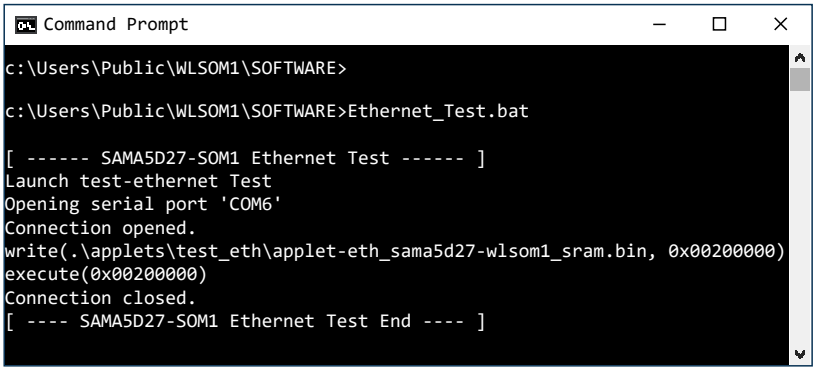
```
@echo off
echo.

rem To select if UART0-IOSET1 is used (Evaluation Kit Default Configuration)
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart0-1.bin

rem To select if UART1-IOSET1 is used (Factory Default Configuration)
set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart1-1.bin

rem To select if UART2-IOSET1 is used
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart2-1.bin

rem To select if UART3-IOSET1 is used
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart3-1.bin

rem To select if UART3-IOSET2 is used
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart3-2.bin

rem To select if UART3-IOSET3 is used
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart3-3.bin
```

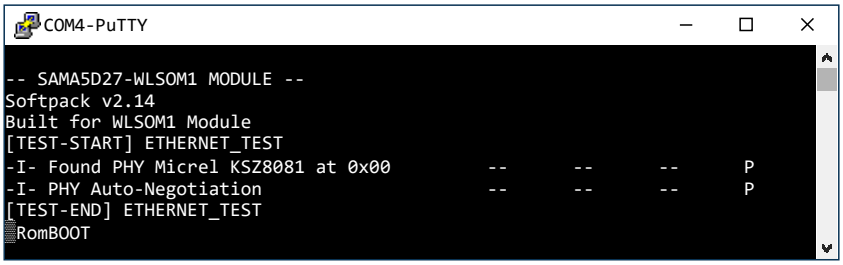
```
@echo off
echo.

rem To select if UART0-IOSET1 is used (Evaluation Kit Default Configuration)
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart0-1.bin

rem To select if UART1-IOSET1 is used (Factory Default Configuration)
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart1-1.bin

rem To select if UART2-IOSET1 is used
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart2-1.bin

rem To select if UART3-IOSET1 is used
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart3-1.bin

rem To select if UART3-IOSET2 is used
set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart3-2.bin

rem To select if UART3-IOSET3 is used
rem set TEST_APP=.\applets\test_eth\applet-eth_sama5d27-wlsom1_sram_uart3-3.bin
```

5.9.2 Run the Ethernet Script

A 10/100 Ethernet PHY, KSZ8081RNAIA, is connected to the MPU through the RMII interface. The following script checks the presence of the Ethernet PHY.

1. Connect a cable to the RJ45 connector. This cable must be connected to a network.
2. From "*%DIR_PATH%\SOFTWARE*" folder, launch the following command:
   – Ethernet_Test.bat
3. The two code fragments below appear in the Command Prompt window and in the UART Debug interface.





When the two tests are passed, indicated by a ‘P’ as shown in the screen shot above, the connection between the two devices is good.

5.10 Testing the 32 kHz Crystal

The presence of the 32 kHz crystal is tested by redirecting it to an external GPIO. There are four possibilities:

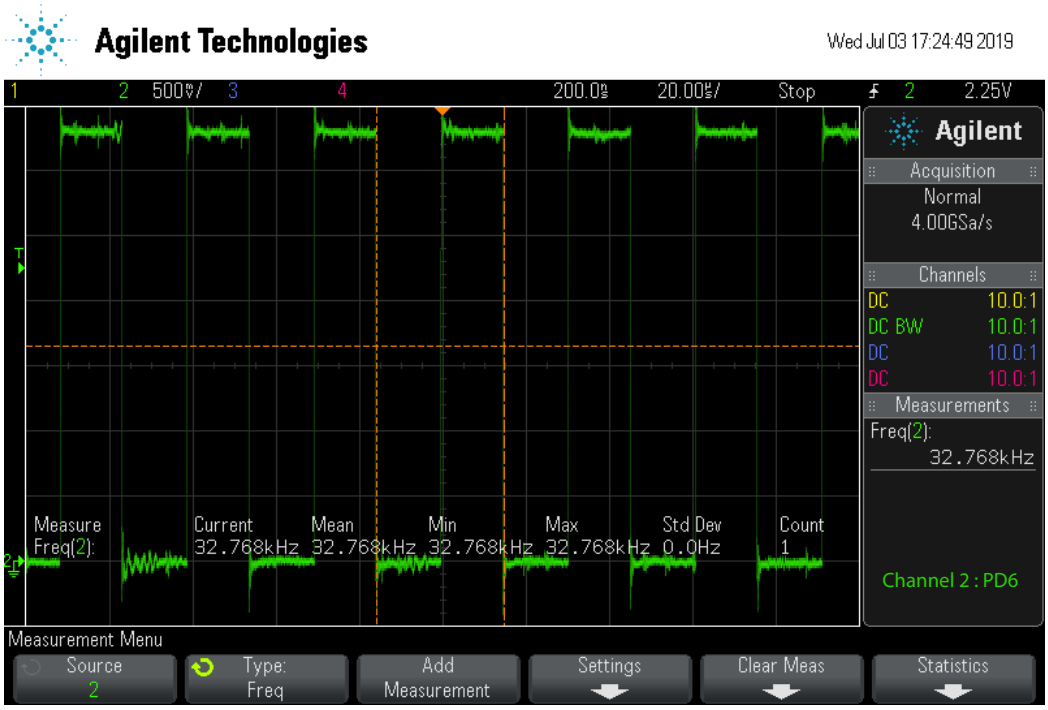
- Redirect the 32 kHz clock to PD6 (Pin 106):
  – Launch "*sam-ba -x tests\32KHZ\pck1_ioset1_pd6.qml*"
- Redirect the 32 kHz clock to PC27 (Pin 35):
  – Launch "*sam-ba -x tests\32KHZ\pck1_ioset2_pc27.qml*"
- Redirect the 32 kHz clock to PC28 (Pin 39):
  – Launch "*sam-ba -x tests\32KHZ\pck2_ioset1_pc28.qml*"
- Redirect the 32 kHz clock to PD11 (Pin 118):
  – Launch "*sam-ba -x tests\32KHZ\pck2_ioset2_pd11.qml*"

Then probe the selected GPIO to determine if the 32 kHz clock is present.

After testing, a device reset is required in order to return the GPIO to its default state. There are two ways to do this:

1. Perform an external hardware reset. (Momentarily tie the NRST signal to GND.)
2. Launch "*sam-ba -x tests\32KHZ\pck_ioset_reset-usb.qml*".

An example of 32 kHz output signal measured on PD6 is shown below:

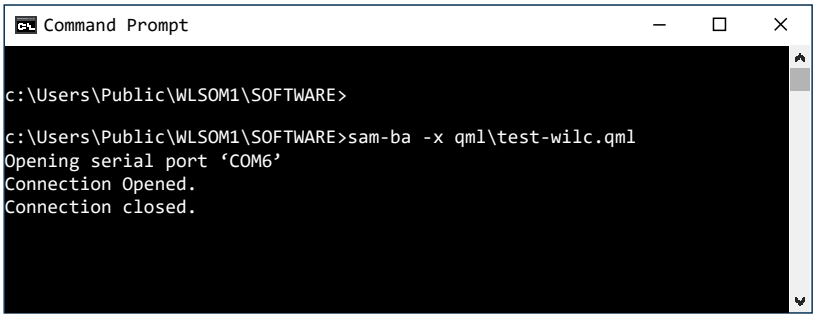


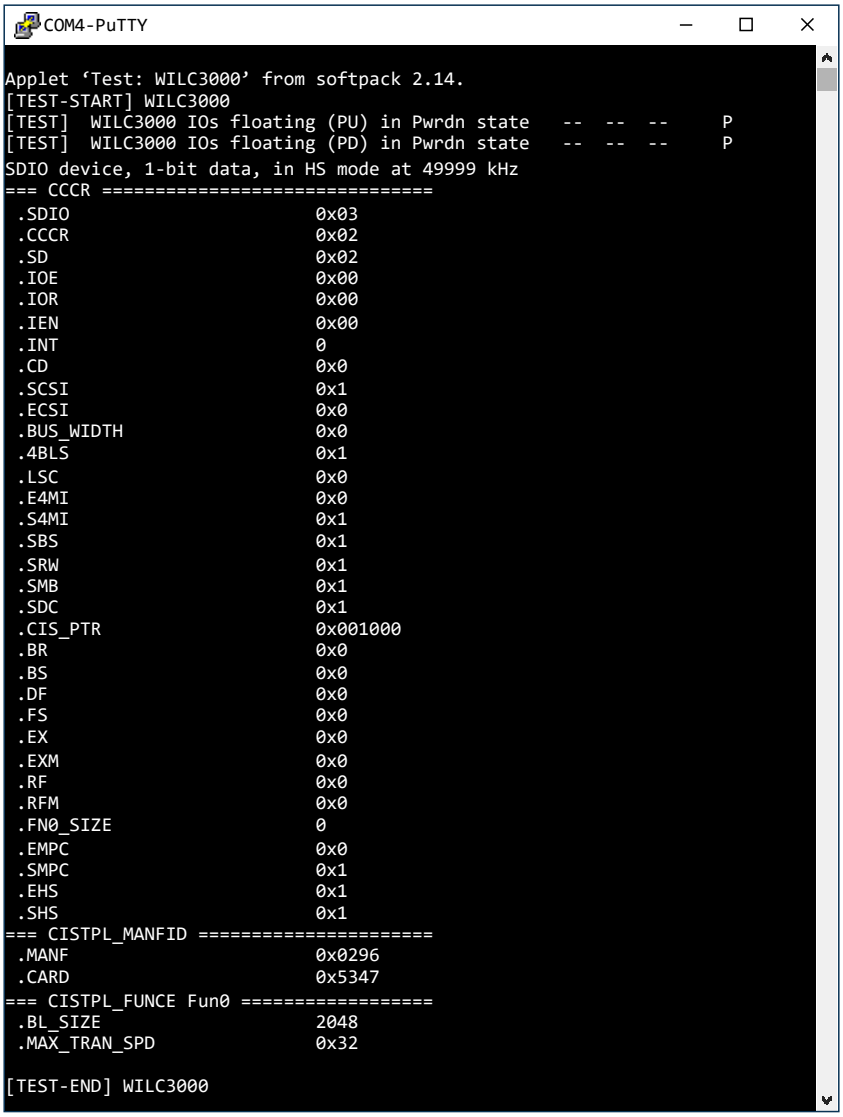
5.11 Testing the Wi-Fi®/Bluetooth® Module

A wireless module, ATWILC3000-MR110UA, is connected to the MPU through two serial interfaces. The following script checks the presence of the wireless module.

Note: The test does not perform a radio transmission test, just the correct connection of the radio module.

1. From the folder "*%DIR_PATH%\SOFTWARE*", launch the following command:
   - sam-ba -x qml\test-wilc.qml
2. The two code fragments below appear in the Command Prompt window and in the UART Debug interface:



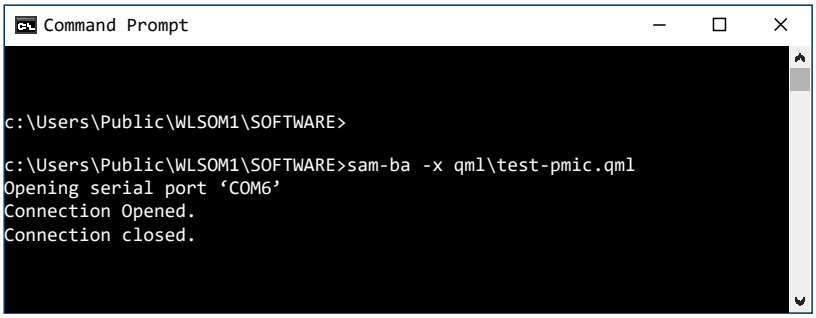


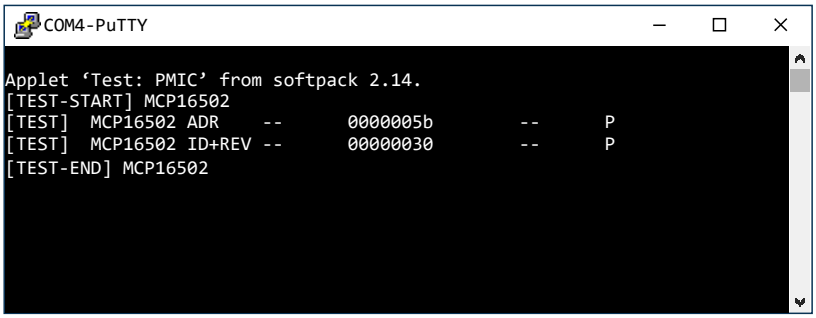
3. When the four tests are passed, indicated by a 'P' as shown in the screen shot above, the connection between the two devices is good.

5.12 Testing the Power Management IC

The power management unit is connected to the SAMA5D2 MPU through an I²C interface. The following script checks the presence of the power management unit.

1. From the folder "*%DIR_PATH%\SOFTWARE*", launch the following command:
   – sam-ba -x qml\test-pmic.qml
2. The two code fragments below appear in the Command Prompt window and in the UART Debug interface:





3. When the two tests "*MCP1502 ADR*" and "*MCP16502 ID+REV*" are passed, indicated by a 'P' as shown in the screen shot above, the connection between the two devices is correct.

5.13 Testing SAMA5D2 PIOs

The following sections describe how to check the interconnect between the SAMA5D2 MPU embedded in the WLSOM1 module and the rest of the application board. A script forces GPIO lines to output and to toggle from low to high level and confirms that the connection is good.

Note: Before launching the script, it is important to check that the I/O toggling does not affect the reliability and/or operability of the customer application by creating conflicts or signal contention. For example, if a GPIO is tied to ground by an external device and the test is launched, this GPIO will be set as output and forced to VDD. As a result, a short circuit is created between VDD and GND and the pad will be damaged.

5.13.1 Configure the File GPIO-Toggling-usb.qml

To carry out the GPIO toggling test, a specific qml file (GPIO-Toggling-usb.qml) is required and is available in the following folder:

- "*%DIR_PATH%\SOFTWARE\tests\GPIO*"

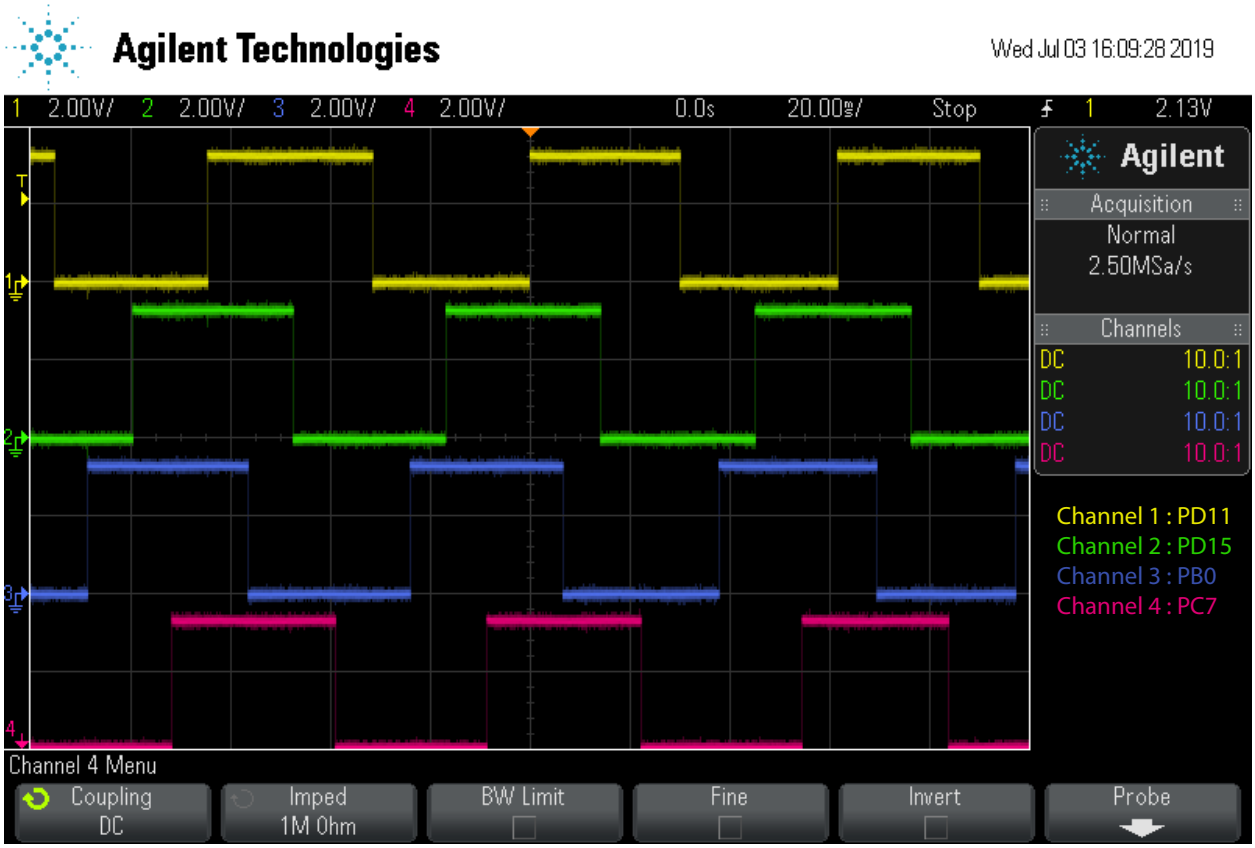
To change the GPIO list, open the file "*GPIO-Toggling-usb.qml*" in a text editor and modify the GPIO list. To modify the script, follow the steps below:

1. Modify the tables "*output_pio_array*" and "*output_pin_array*". For details, refer to the Note below.
2. Modify the variable "*number_of_toggling_cycle*" depending on the application.
3. Modify the variable "*number_of_io_toggling*". It must be equal to the number of I/Os listed in the two tables above.
4. Modify the variable "*delay_time*" depending on the application. The variable "*delay_time*" is expressed in ms.
5. Save and close the file. No compilation is required.

An example is given below with toggling function for the PIOs PD11, PD15, PB0 and PC7:

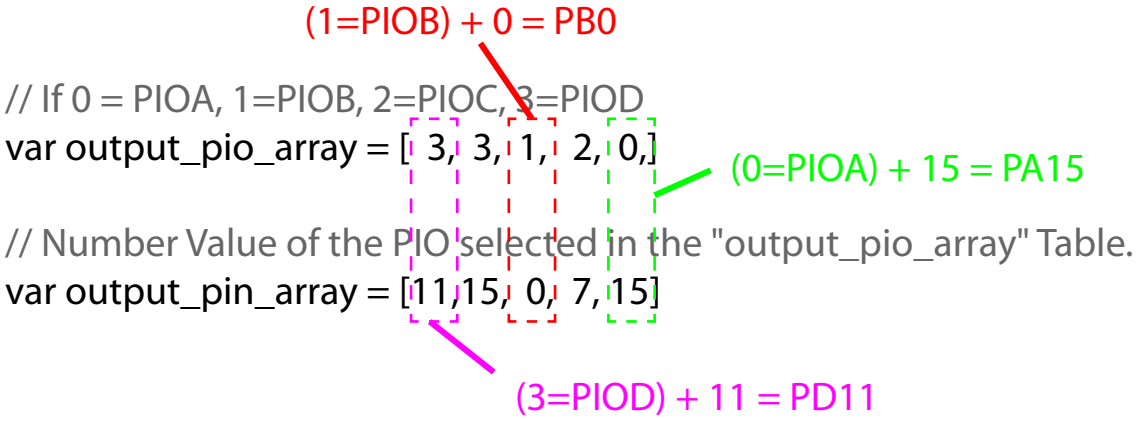
```
import QtQuick 2.3
import SAMBA 3.2
import SAMBA.Connection.Serial 3.2
import "../SAMA5D2_PIO.js" as Pio

/*
....
*/

Item {
        SerialConnection {
                id: connection
                onConnectionOpened: {

                        // If 0 = PIOA, 1=PIOB, 2=PIOC, 3=PIOD
                        var output_pio_array = [  3,  3, 1, 2]

                        // Number Value of the PIO selected in the "output_pio_array" Table.
                        var output_pin_array = [11,15, 0, 7]

                        var number_of_toggling_cycle = 10
                        var number_of_io_toggling = 4           // Number of toggling I/Os
                        var delay_time = 5                      // Expressed in ms.

                        print("PIOs are set in output")
                        for (var i=0; i<number_of_io_toggling;i++) {
                                Pio.setup_output(this,output_pio_array[i],output_pin_array[i])
                        }

                        for (var j=0; j<number_of_toggling_cycle;j++) {
                                for (var i=0; i<number_of_io_toggling;i++) {
                                        Pio.output_clear(this,output_pio_array[i],output_pin_array[i])
                                        Utils.msleep(delay_time)
                                }
                                for (var i=0; i<number_of_io_toggling;i++) {
                                        Pio.output_set(this,output_pio_array[i],output_pin_array[i])
                                        Utils.msleep(delay_time)
                                }
                        }

                        print("PIOs are cleared")
                        for (var i=0; i<number_of_io_toggling;i++) {
                                Pio.output_clear(this,output_pio_array[i],output_pin_array[i])
                                Utils.msleep(delay_time)
                        }
                }
                onConnectionFailed: print("Connection failed: " + message)
        }
        Component.onDestruction: connection.close()
}
```

Applying this setup, the GPIO toggling is as follows:



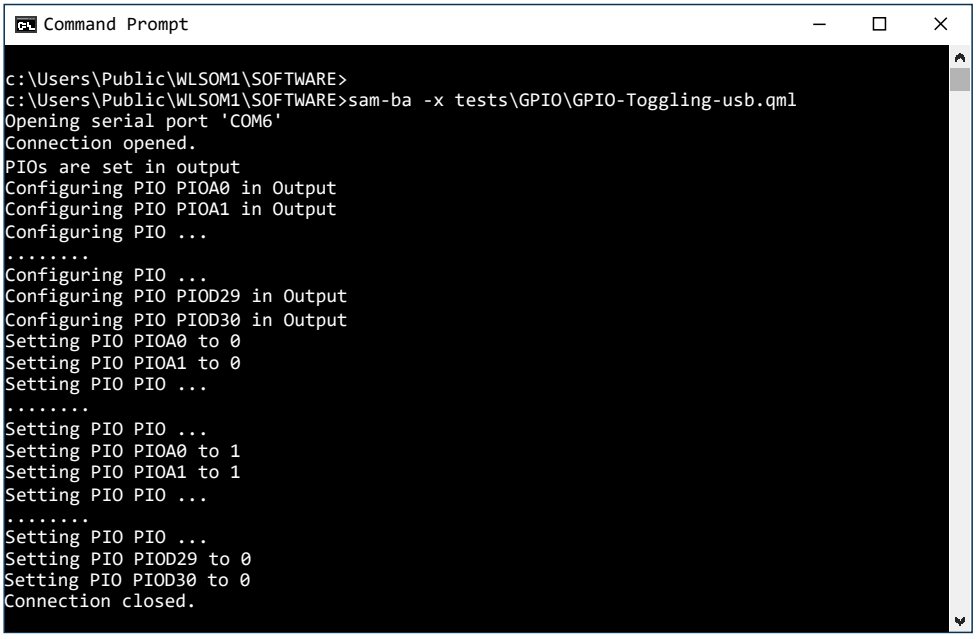
Notes:
The two tables "*output_pio_array*" and "*output_pin_array*" can be modified if the following rules are respected:

- There is no limitation of PIOs. Any PIO can be added in the two tables. The number of variables available in "*output_pio_array*" must be equal to the number of variables in "*output_pin_array*".
- Each PIO is decomposed as follows:
  - I/O Bank Name (e.g. PIOAx, PIOBx, PIOCx, PIODx)
    - I/O bank PIOAx is converted to "0" (for script interpretation)
    - I/O bank PIOBx is converted to "1"
    - I/O bank PIOCx is converted to "2"
    - I/O bank PIODx is converted to "3"
  - I/O line number (from 0 to 31)
- The figure below illustrates how columns must be arranged to allow script interpretation during execution.



5.13.2 Run and Analyze the Script

1. From the folder "*%DIR_PATH%/SOFTWARE/*", launch the following command:
   – sam-ba -x tests\GPIO\GPIO-Toggling-usb.qml
2. The following text appears in the Command Prompt window:



3. During the test activity, it is possible to monitor the selected GPIO switching from 0V to VDD_3V3 voltage. If this is observed, then the interconnect between the MPU and customer board is good.

6. Revision History

6.1 Rev. B - 03/2021

Throughout: Updated all secure element references to ATECC608B (was ATECC608A).
Added Chip Identifier in 5.3 Checking the SAMA5D27 Microprocessor.

6.2 Rev. A - 12/2019

First issue

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 devices:

- 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 and under normal conditions.
- There are dishonest and possibly illegal methods being used in attempts to breach the code protection features of the Microchip devices. We believe that these methods require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Attempts to breach these code protection features, most likely, cannot be accomplished without violating Microchip's intellectual property rights.
- Microchip is willing to work with any customer who is concerned about the integrity of its code.
- 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. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-7752-5

Quality Management System

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

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: www.microchip.com/support Web Address: www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4485-5910 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-72400 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820