# User Manual

## DA9213, DA9214, and DA9215 Performance Boards

### UM-PM-029


## Abstract

*This document is a user guide for DA9213, DA9214, and DA9215 Performance Boards. It provides the basic information on how to configure the PCB, how to install and use the Power Commander (GUI) software.*

# Contents



# Figures



# Tables

# 1 Terms and Definitions

| | |
|---|---|
| BGA | Ball Grid Array |
| CPU | Central Processing Unit |
| DDR | Double Data Rate SDRAM (Synchronous Dynamic Random Access Memory) |
| DUT | Device Under Test |
| GPU | Graphic Processing Unit |
| GUI | Graphical User Interface |
| PCB | Printed Circuit Board |
| POL | Point Of Load |
| PSU | Power Supply Unit |
| USB | Universal Serial Bus |

# 2 References

[1] DA9213, DA9214, and DA9215, Datasheet, Dialog Semiconductor.

# 3 Introduction

DA9213, DA9214, and DA9215 are multi-phase buck converters optimized for the supply of CPUs, GPUs, and DDR memory rails in smartphones, tablets, and other handheld applications.

DA9213 is configured as a single channel 4-phase buck converter capable of delivering up to 20 A output current. DA9214 is configured as 2-channel, 2-phase buck converter capable of delivering up to 10 A output current per channel. DA9215 is configured as 2-channel PMIC with a 3-phase buck converter capable of delivering up to 15 A output current and a single-phase buck converter capable of delivering up to 5 A output current.

The output voltage is configurable in the range 0.3 to 1.57 V. The input voltage range of 2.8 to 5.5 V makes it suited for a wide variety of low voltage systems, including all Li-Ion battery supplied applications.

DA9213, DA9214, and DA9215 Performance Boards have been designed to allow measurement, evaluation, and programming of the device.

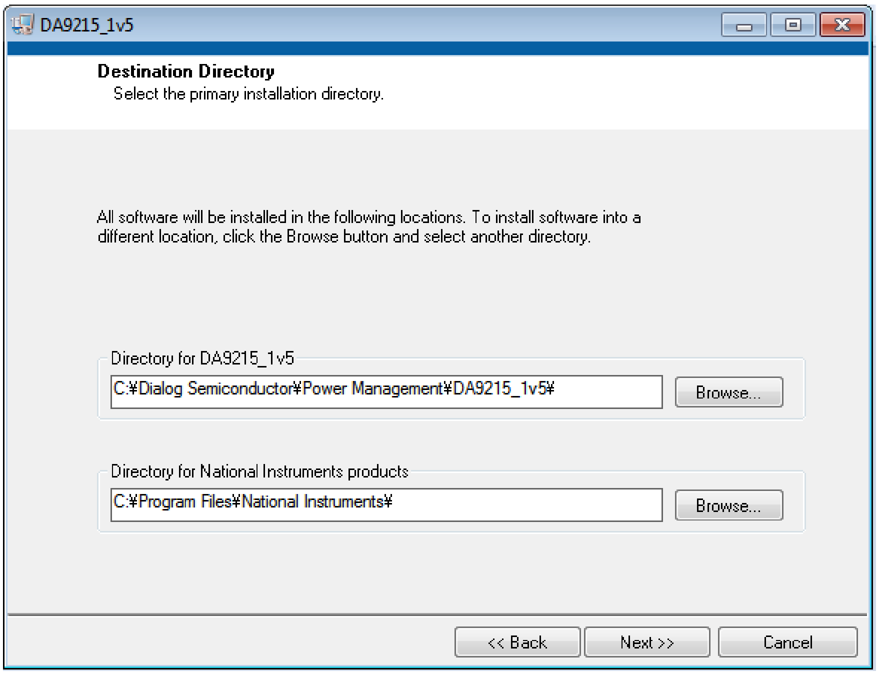
The Power Commander software DA9215_1v5 can be used for DA9213, DA9214, and/or DA9215 Performance Boards. It uses a simple graphical interface, allowing the device to be controlled via the USB port of a PC. The mini USB connection is visible on the board and as long as the cable is connected to the USB port of the PC the green LED D3 is on.

The software allows the configuration of the device by write and read operations to all control registers, and provides monitoring of device status. The software uses operating system Windows 2000/XP/Vista/Windows 7 with a USB1.1 or USB2.0 interface.

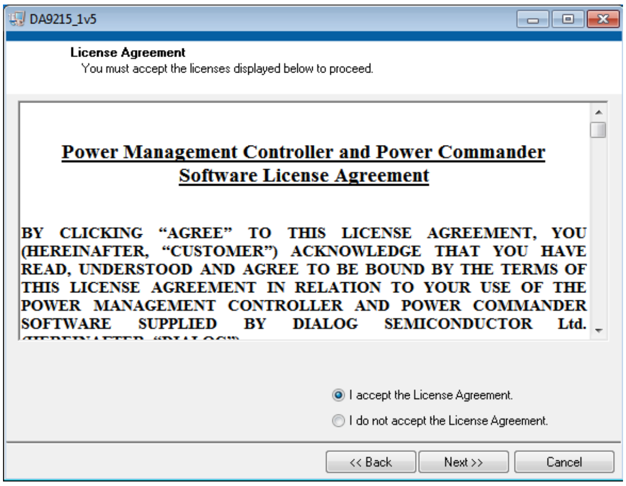
# 4 Software Installation

Plug the USB stick into a spare port on your computer and run the program **setup.exe** to start the automated script. This file can be found in the Software directory.

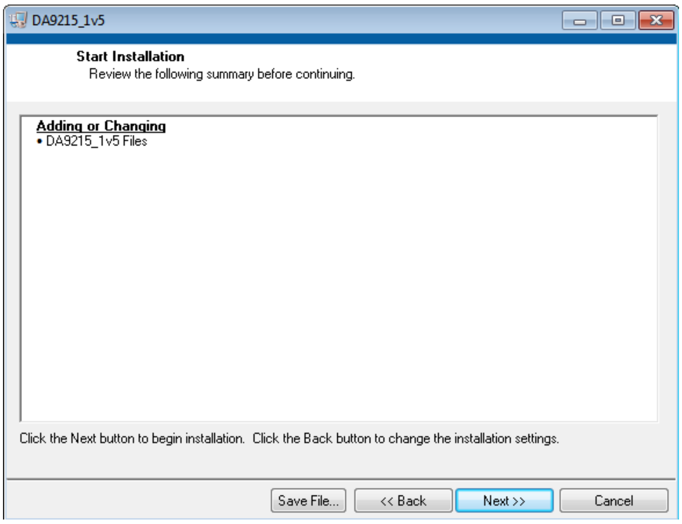
By default, the directory C:\Dialog Semiconductor\Power Management\DA9215_1v5 is used.



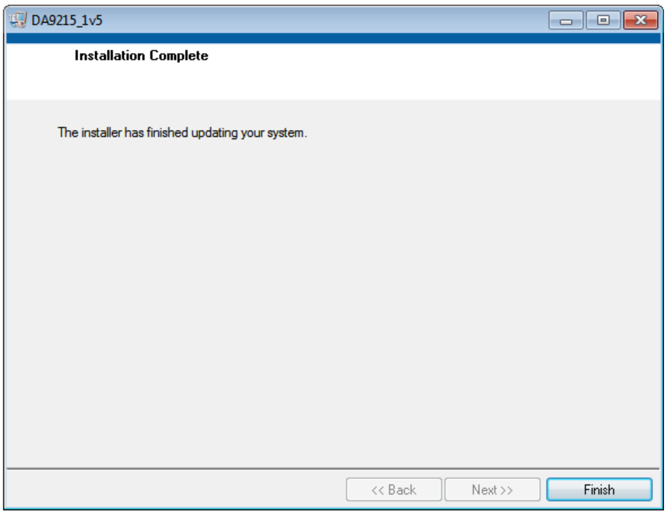

1. Click **Next** to accept the default software installation location.




2. Click the radio button to accept the software license agreements twice.



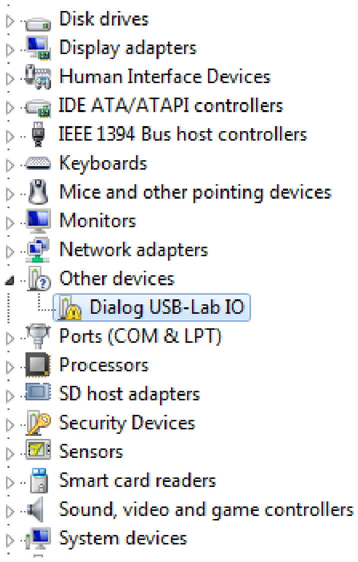

3. Click **Next** to begin the software installation.



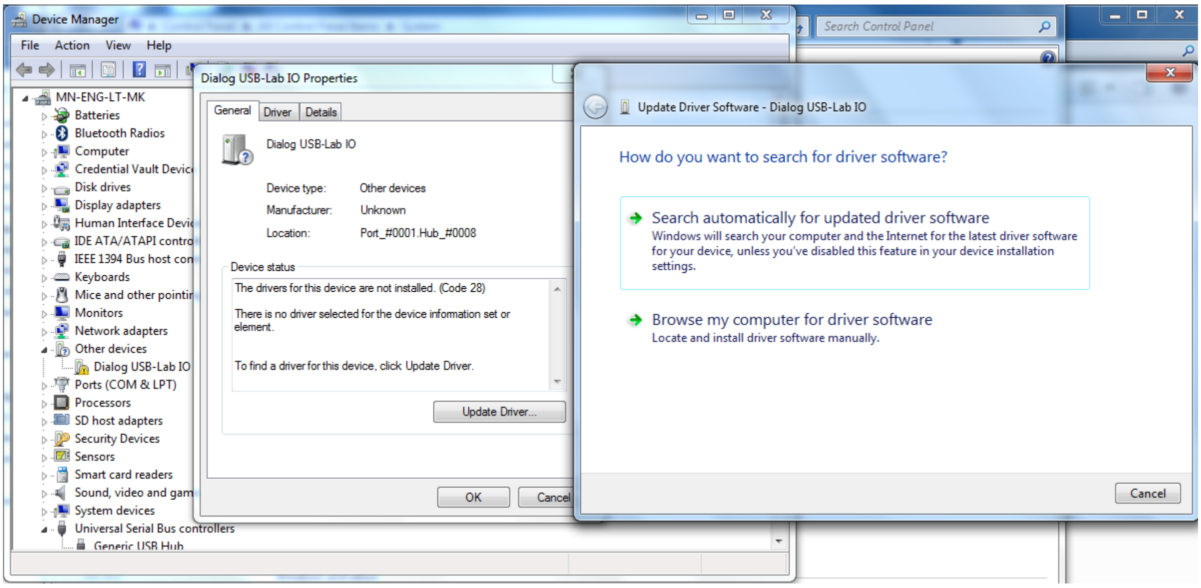

4. When the installer indicates it has finished updating, click **Finish** to complete the process.

| NOTE |
|---|
| After the installation has been completed you need to restart your computer. |

5. Plug in the USB cable so that Windows detects the USB device. A prompt for the drivers, which should be automatically located in the **Driver_PID-1011** directory of the media, appears. If this does not happen automatically, open the Device Manager:




6. Double-click on the unknown Dialog device, and update the driver as shown:




7. Browse to the **Driver_PID-1011** directory of the media. When installed correctly, Dialog USB Driver is listed on the Device Manager.

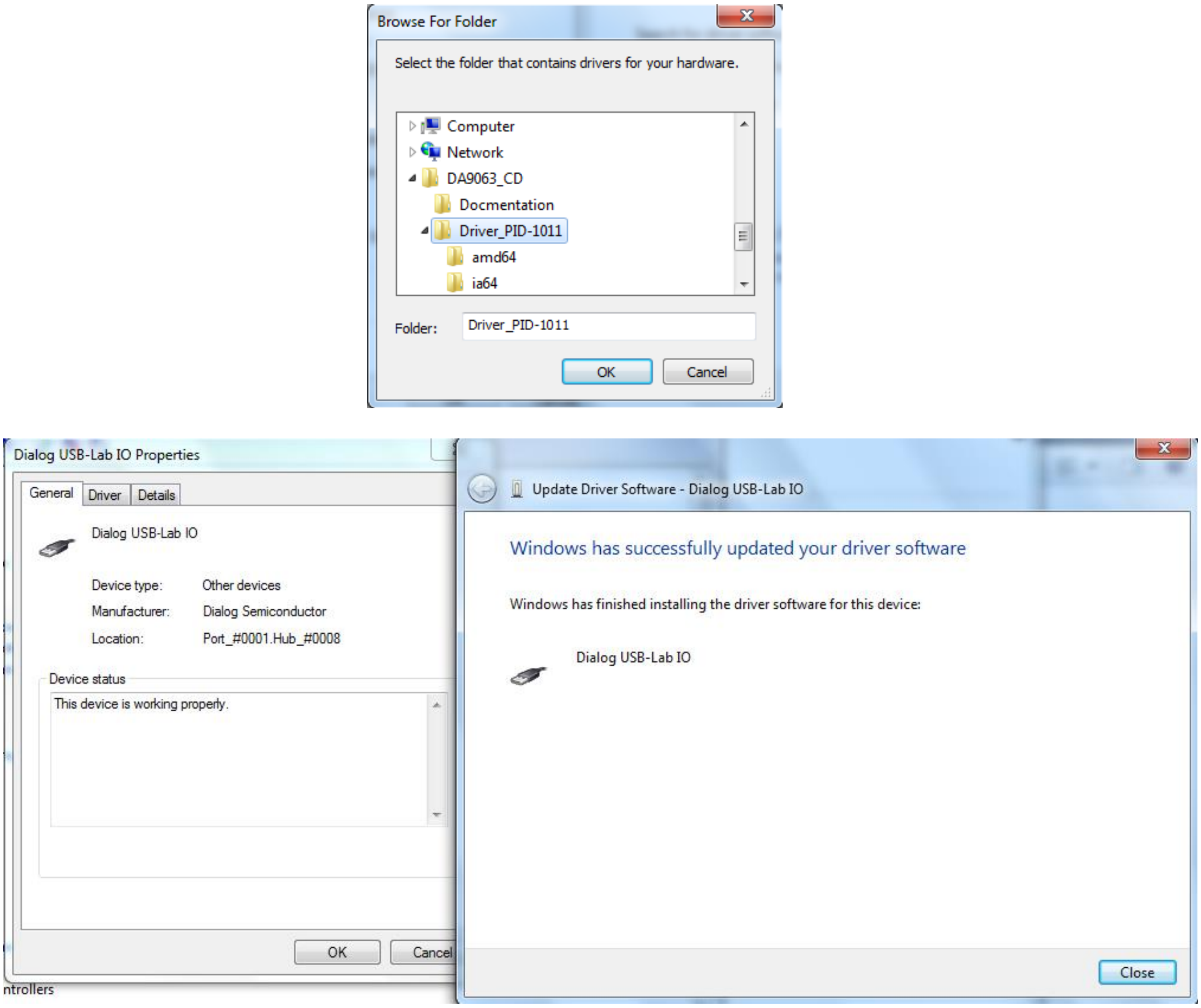



**Figure 1: Successful Installation Procedure**

To uninstall the software please use the Windows **Add/Remove Programs** function, which is in the **Start**, **Settings**, **Control Panel** menu.

# 5 Hardware

Each device has its dedicated evaluation PCB. DA9213 Performance Board is based on PCB numbered 227-02-A, DA9214 Performance Board is based on PCB numbered 227-04-D, and DA9215 Performance Board is based on PCB numbered 227-07-B.

The Performance Board includes an USB-I$^2$C bridge for communication with the device and a few external active components to reduce the requirement for external equipment. It also has some jumper links to provide access to different configurations and measurement test points (see Appendix A). Altering the jumper positions should only be done with a complete understanding of the links description and a supplemental configuration of some registers may be required.

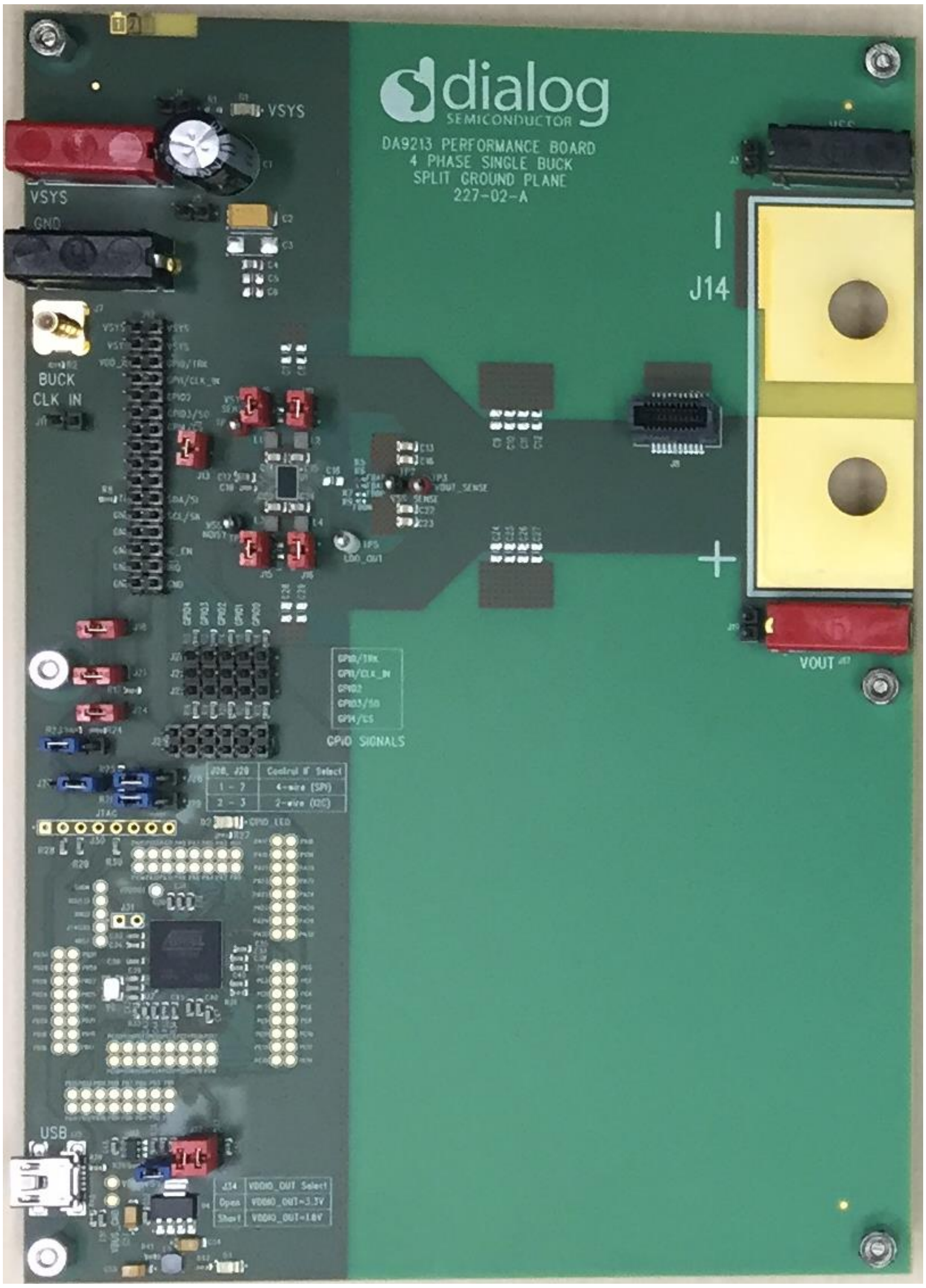



**Figure 2: DA9213 Performance Board (227-02-A)**

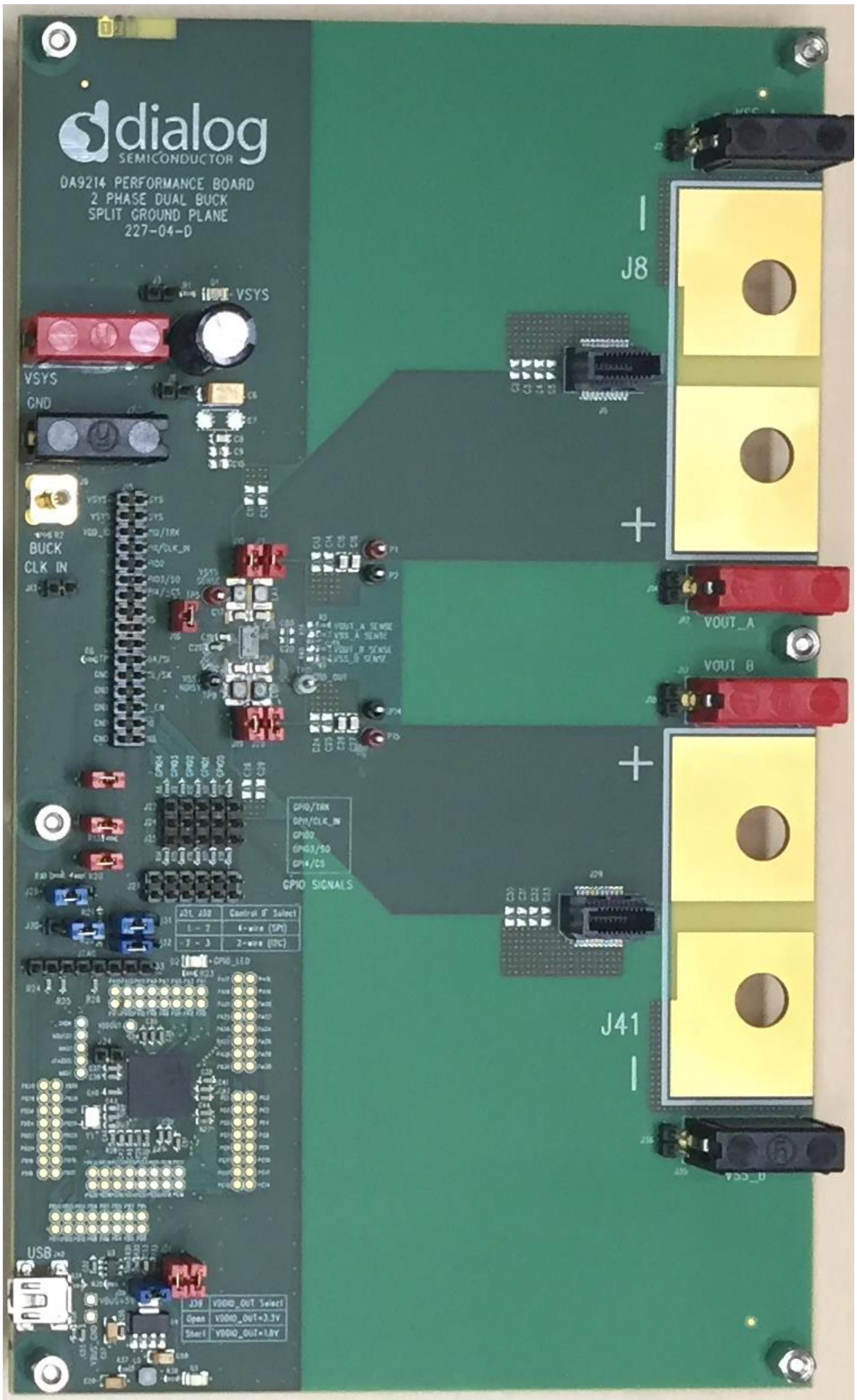



Figure 3: DA9214 Performance Board (227-04-D)

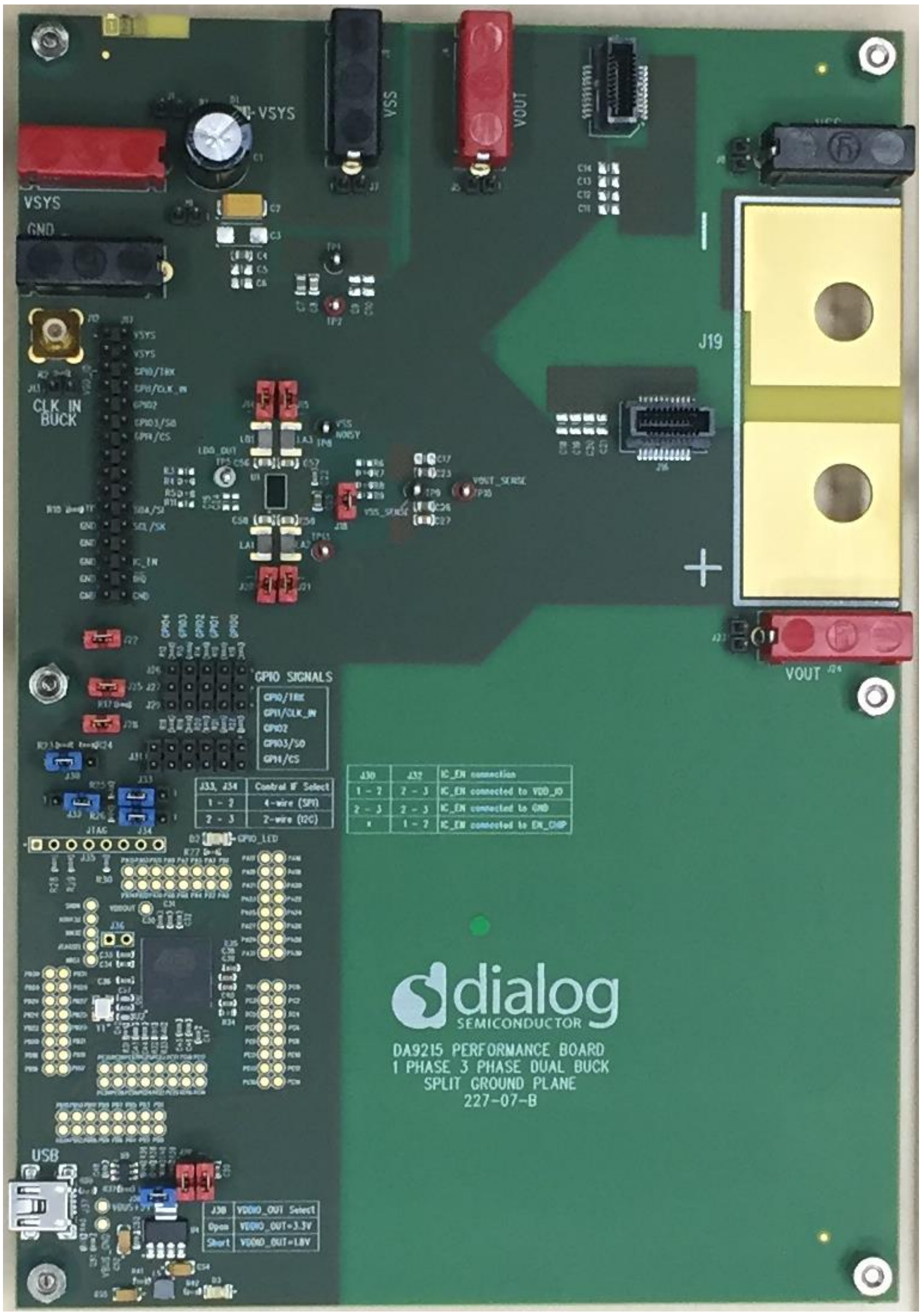



Figure 4: DA9215 Performance Board (227-07-B)

# 6 Configuration Tab

The software is launched by clicking the shortcut in the **Start** menu.

The optimum setting for the PC display size is 1024x768 pixels or above. The font size on the PC display should be normal (95dpi). It is important to note that a display size other than the recommended setting may affect the way in which the tabs appear.

A pop-up for selecting a configuration file appears on start-up. If you do not have any to load, click **Cancel**.

Figure 5 appears, with the **USB OK?** LED lit if the USB cable is correctly connected and the USB interface is functioning.

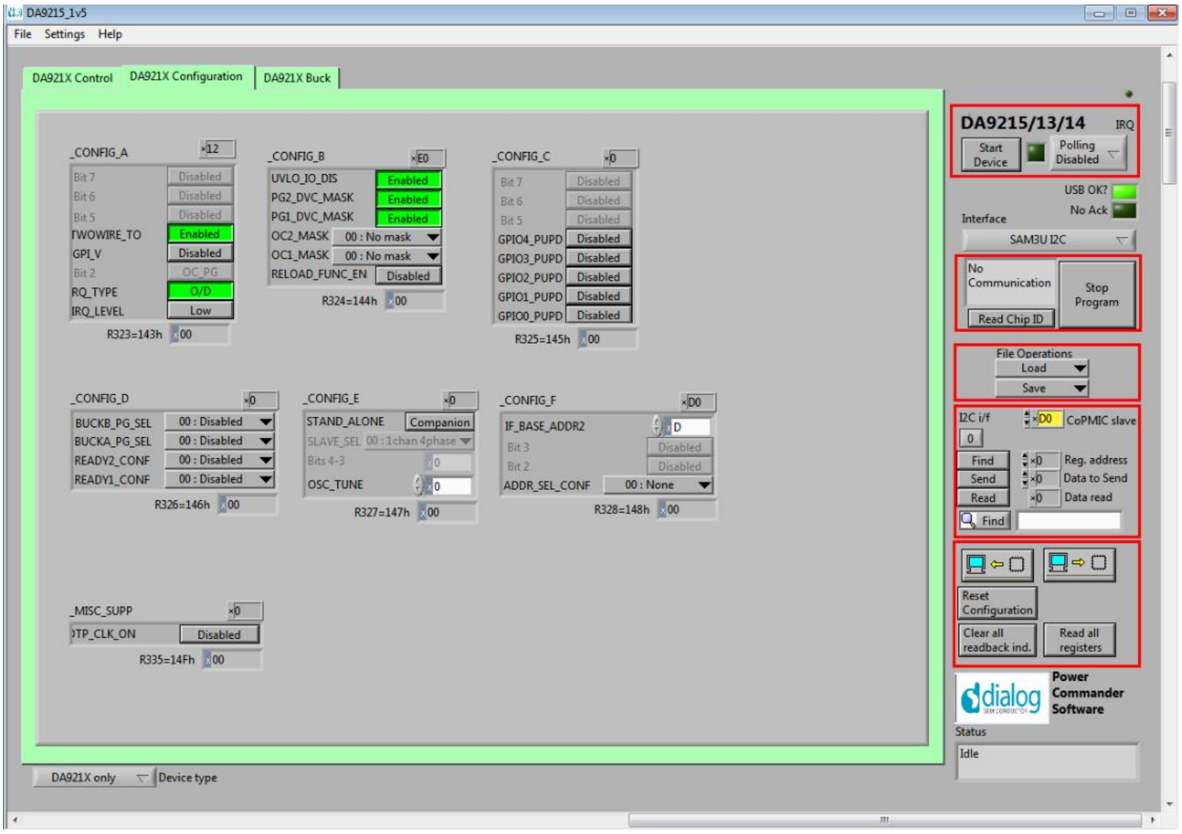



**Figure 5: Initial Interface**

Three tabs are available across the top:

1. **Control**, for the main settings of I/Os, the interface and the status/event registers.
2. **Configuration**, for advanced DA9213, DA9214, or DA9215 configuration.
3. **Buck**, which includes all the buck settings (output voltages, operating modes)

As the software has been developed to evaluate DA921X and DA9063 (main PMIC) separately or together on the same PCB, the specific DUT can be configured in the drop-down menu at the bottom left of the screen. By default this is **DA921X only** as showwn in Figure 6.

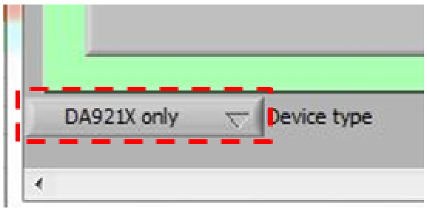
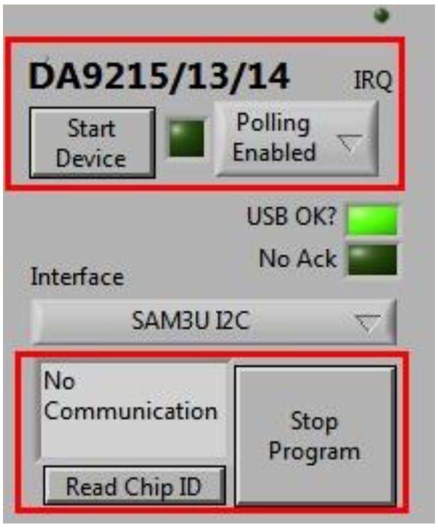



Figure 6: DUT Selection

## 6.1 Interface Indicators




***Start Device***

When pressed it automatically configures the register R327 CONFIG_E for standalone operation.

***Polling Enabled***

If disabled, the main read-backs from the device are suppressed. This is used to force the communication over the bus to be silent.

If this is set to automatic, the program only polls the device while the application is the topmost window. If obscured by another program or window, polling is disabled.

***LED***

If device is active this is green, otherwise it is red.

***Interface***

Select between USB I²C control, offline mode, and SPI. Switching to offline, then back to USB re-initialises the USB interface.

***USB OK?***

The light indicates that the USB is fine and communicating.

***Read Chip ID***

These fields indicate the device version and trim status when the device is active. When inactive, the version and trim status are not correct.

### Stop Program

This terminates the program but leaves it inactive on the screen.

## 6.2 File Load/Save


### Load

Load previously saved text files.

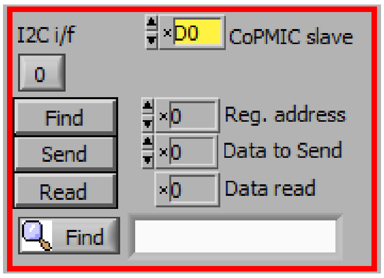
### Save

Save current panel state to a text file. Selecting the **Register Dump** option saves the current register values to the text file, see Appendix A.

**Note 1** There is a difference between **Save** and **Register Dump**. **Save** dumps the contents of all panel controls to the file (a save state operation), while **Register Dump** reads the device contents (including status registers) into the file.

## 6.3 Register Access

This is probably the most important section of the **Configuration Panel**.




### CoPMIC slave

Set the slave address of the device. This affects all I²C communications. See also the register INTERFACE in the *DA9213/4/5 Datasheet* [1].

### Find

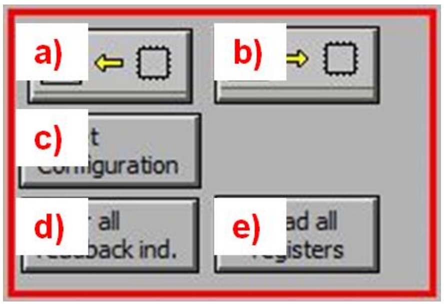
Finds a control matching a full or partial register name, a control bit name, a register number (for example, R208 or D0h). Pressing **Find** repetitively steps through all matching items.

### Send

Sends single byte data to I²C device using **Slave Address**, **Register Address**, and **Data to Send**.

### Read

Reads single byte data from I²C device using **Slave Address** and **Register Address**.




***a) Synchronize Panel from Device***

Reads all the register contents of the device and updates the panel to match.

***b) Synchronize Device from Panel***

Writes all the device registers to match the panel (refresh operation).

***c) Reset Configuration***

Resets the registers to values specified in configuration file.

***d) Clear all read-back indication***

Sets all read-back indicators to 0.

***e) Read all read-back***

Reads all registers, comparing with the panel controls.

## 6.4 Control Tabs

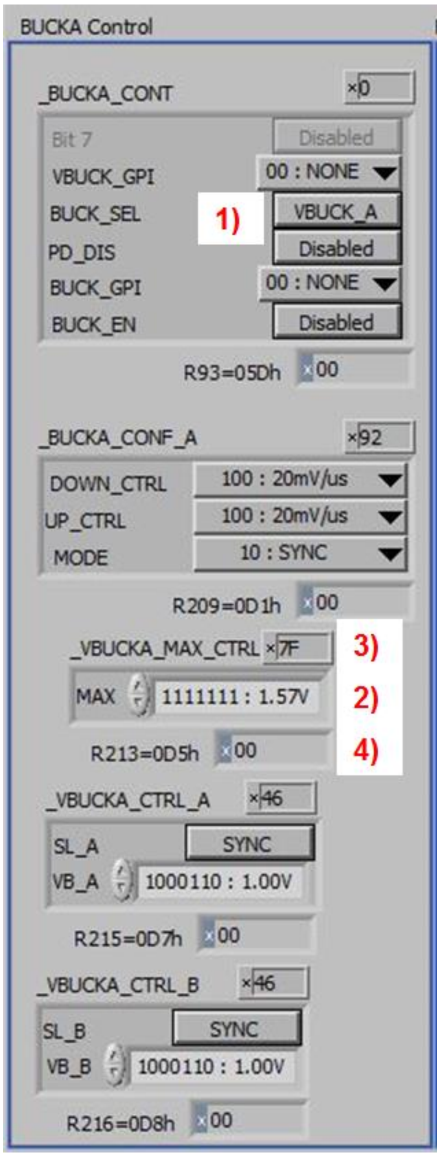
The tabs **DA921X Control**, **DA921X Configuration**, and **DA921X Buck** all have the same format.

Each register cluster comprises a control with a mixture of:

1. Boolean toggle buttons.
2. Multi-value ring controls or slide controls.
3. A hex indicator showing the total equivalent value.
4. A read-back indicator showing the current contents of the register. The read-back indicator is labeled with the register number in both decimal and hex.

Read-back indicators can be switched individually by clicking on the **x** to decimal, octal, hex, or binary, or they may all be changed at once between hex and binary by using the **Settings**, **Binary Indicators** menu item.




Changing a register control immediately sends the value to the selected register, and reads the value back again, comparing the result with the hex indicator. Note that all bits of the registers are sent at once. Therefore, this does not allow changing multiple bits simultaneously.

If the read-back indicator is red, it indicates that the current value does not match the panel.

# 7 Quick Start Tutorial Guide

This section provides instructions for starting up the evaluation board and the software to get the device up and running. It also tackles the first steps of making modifications on the device.

DA9214 Performance Board (227-04-D) is used for this example; however, the same method applies for both DA9213 and DA9215 Performance Boards.

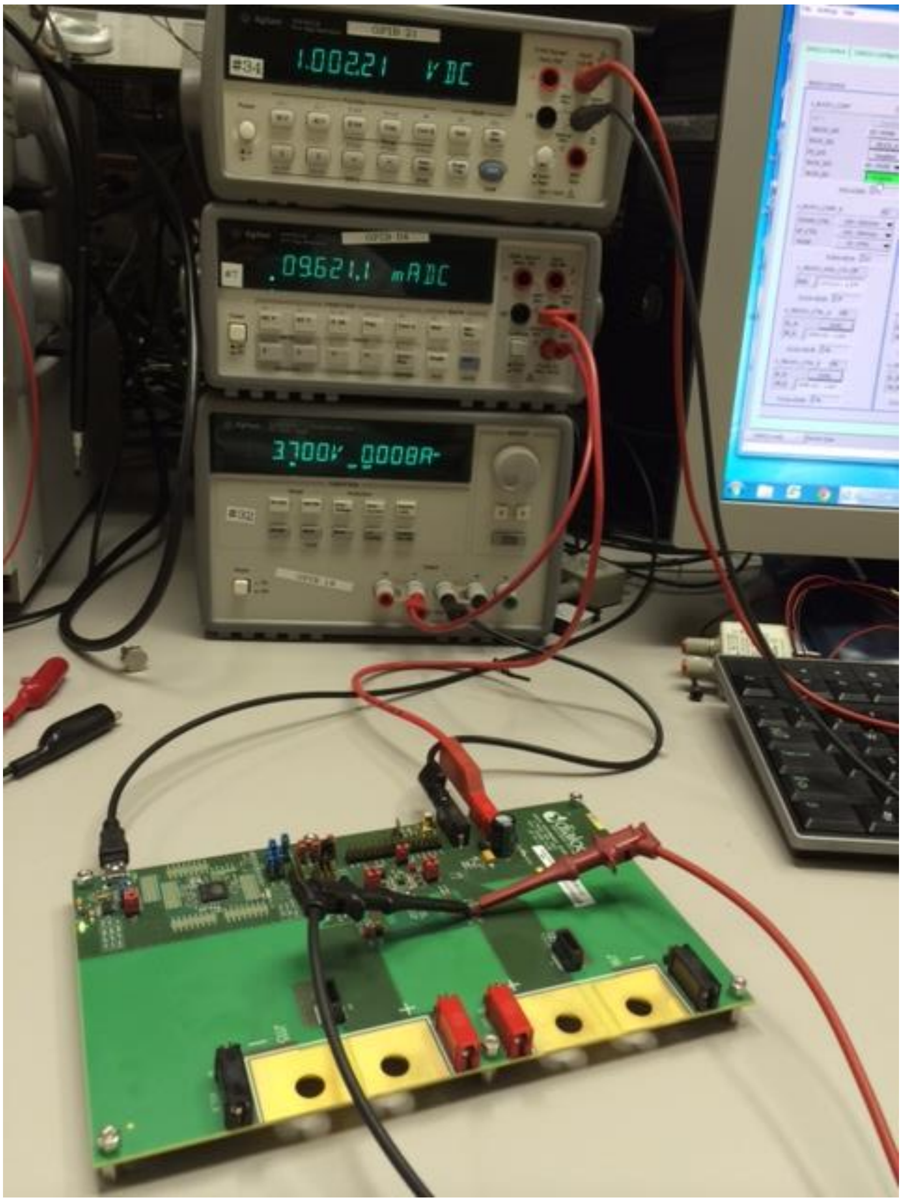



**Figure 7: Measurement Set-Up**

Supply DA9214 from the VSYS and GND connectors as shown in Figure 7. A voltage of 3.7 V (typical Li-Ion battery voltage) is applied and the input current is measured via a multi-meter. The output voltage can be directly measured via a second multi-meter.

Start the GUI software installed on your computer. The **USB OK?** LED on the right-hand side of the screen should be off if the USB cable is not plugged. Once the USB cable is connected the LED displays a steady green. The Interface selector should show **SAM3U I2C**. If not, change it to **off** and then to **SAM3U I2C**.

At the top right of the configuration tab the interface indicator should display the same details as shown in Figure 8.

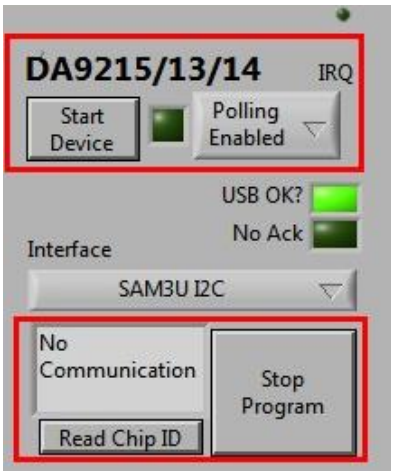



**Figure 8: Top Right Interface Indicator**

Select the **Start Device** button to configure the software for standalone operation correctly. This enables control of the DA9214 in standalone mode. In this mode, DA9214 acknowledges all I²C commands, which does not always happen if it is in companion mode.

For details please refer to the STAND_ALONE bit in the Datasheet [1].

Synchronize the panel from the device, by pressing the icon (indicated by the red arrow in Figure 9), the target voltages and all DA9214 parameters are updated to the content of the IC registers.

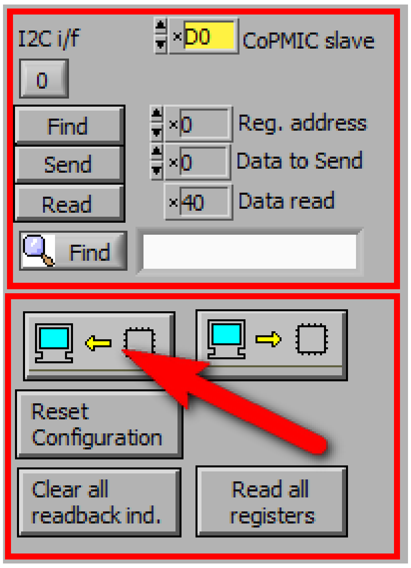



**Figure 9: Bottom Right Interface Indicator**

Enable the BUCKA converter by pressing on the BUCK_EN button in BUCKA Control, which is equivalent to writing 0x01 into the related bit, see Figure 10. An increase in the current drawn from the external supply will occur.

| NOTE |
| --- |
| The quiescent current shown on the multi-meter may still be higher than expected. This is due to the missing configuration and connection of floating I/O ports and will be reduced as soon as they are correctly set. |

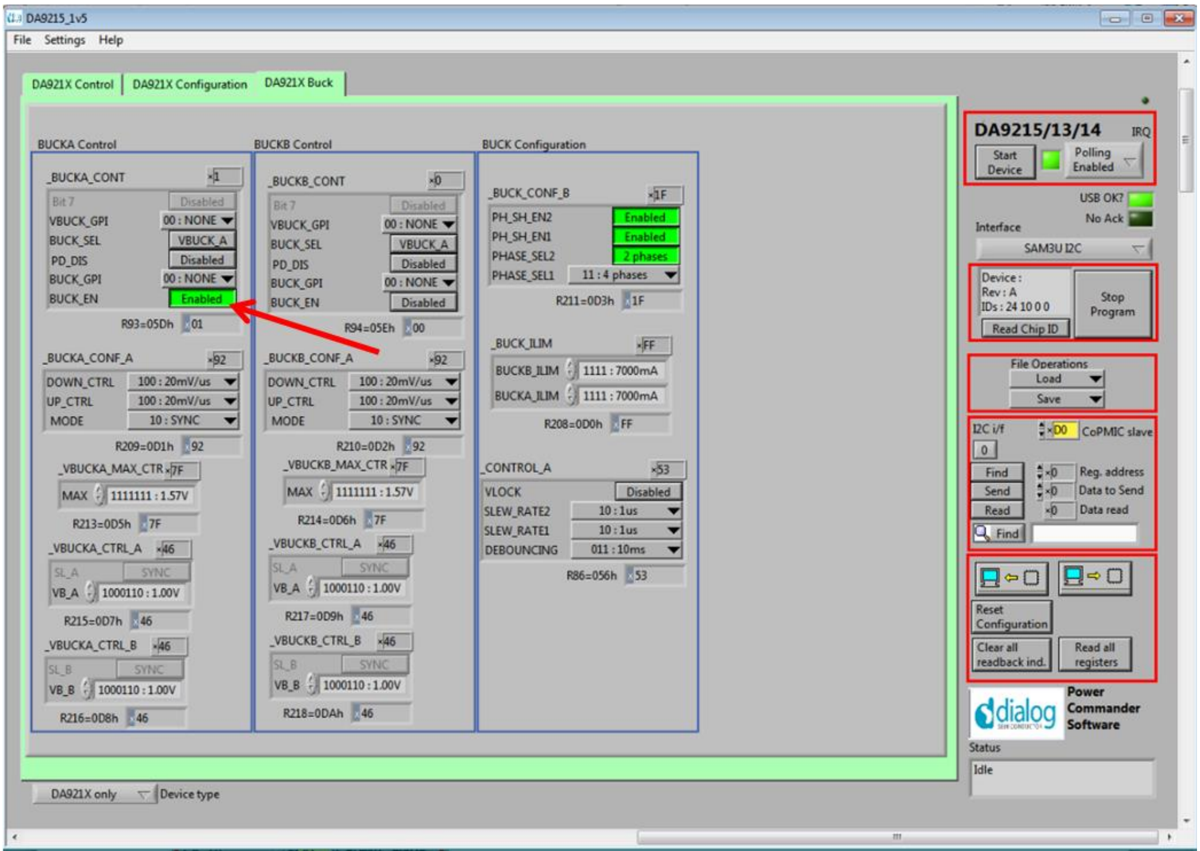



**Figure 10: Enabling BUCKA**

The same operation can also be done via the I²C interface panel by selecting the register address, the data to send and pressing the **Send** button. A read-back can be performed by clicking the **Read** button, see Figure 11.

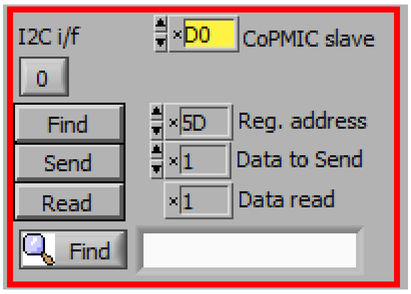



**Figure 11: Middle Right Interface Indicator**

After enabling the BUCKA converter it is possible to monitor the output voltage on the PCB via an external multi-meter connected to the VOUT_A pin, as shown in Figure 7. The read-back should provide a value close to 1.0 V. If the part is not trimmed the value can slightly differ from the target.

It is also possible to directly monitor some relevant signals on the PCB, for example, the inductor current of each phase.

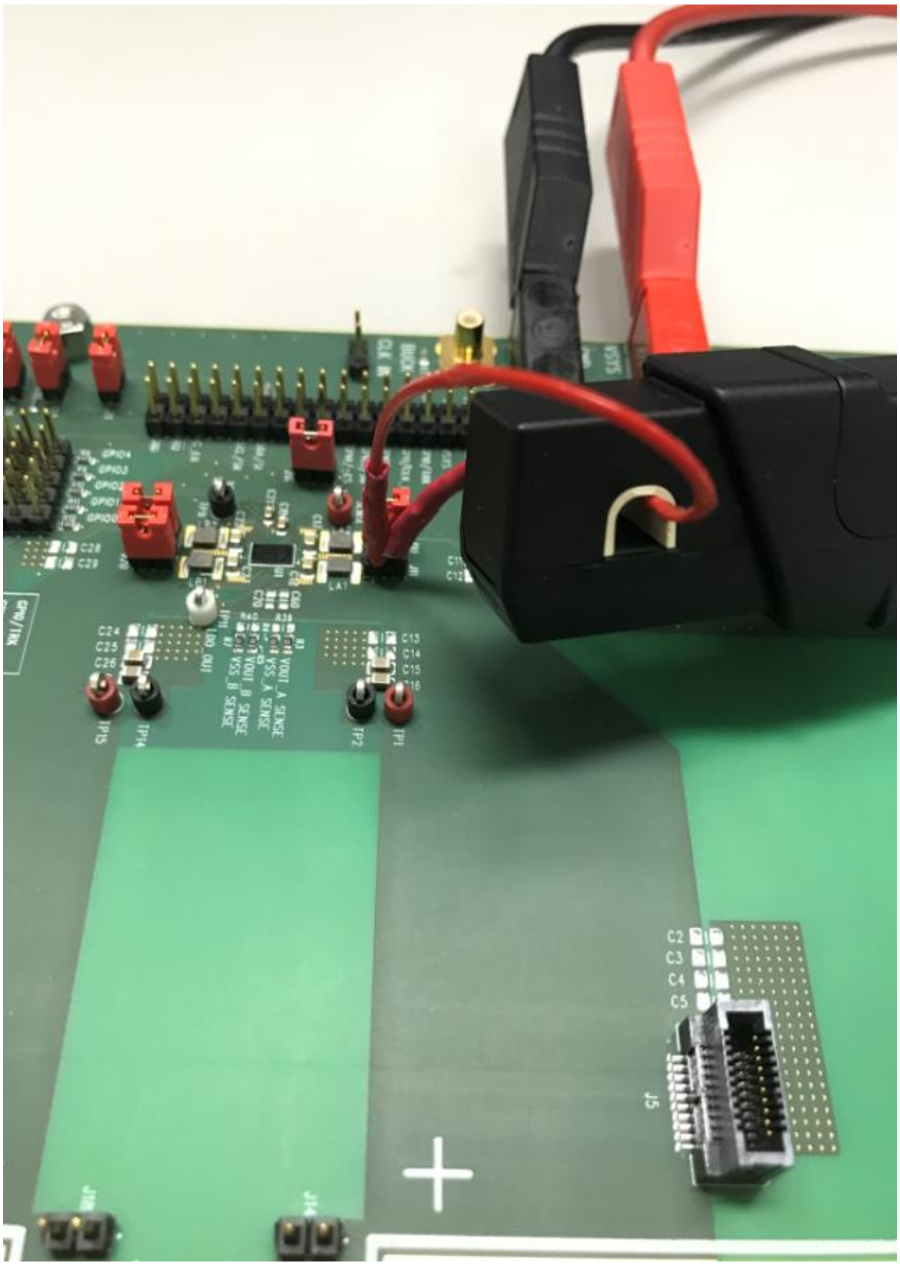


**Figure 12: Monitoring the Inductor Current of BUCKA**

When BUCKA is enabled, only phase-1 (LX_A1) is switching, whilst phase-2 (LX_A2) does not. This is happens when PH_SH_EN1 is asserted. Since no load is applied to the output of the buck converter, DA9214 automatically optimizes the number of active phases and turns off phase-2 to save power.

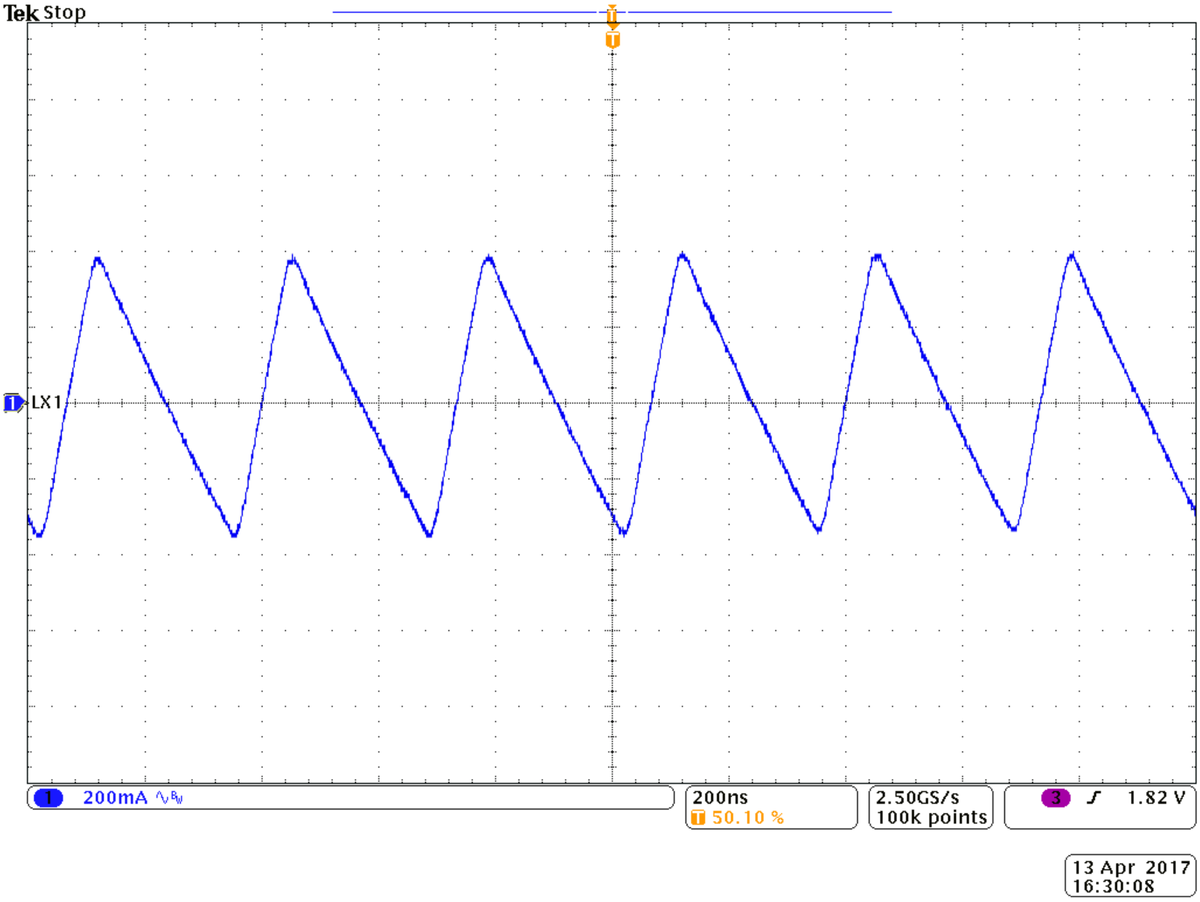
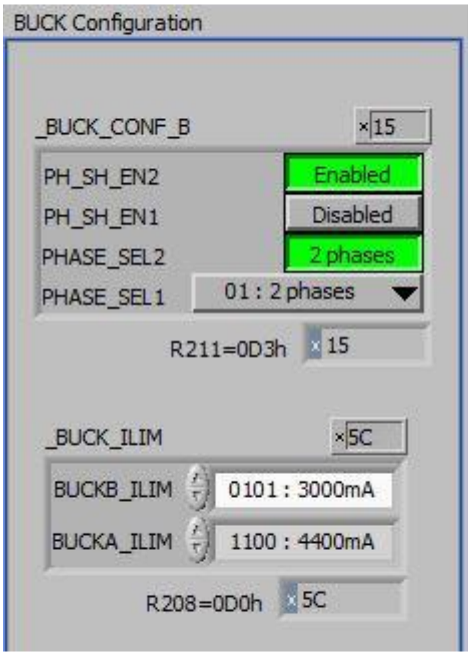



**Figure 13: Inductor Current of Phase-1, Phase-Shedding Enabled**

To disable the phase shedding set the PH_SH_EN1 bit to zero, thereby forcing both phases to be enabled.

The inductor current of phase-1 and phase-2 when BUCKA operates with two active phases is shown in Figure 14.



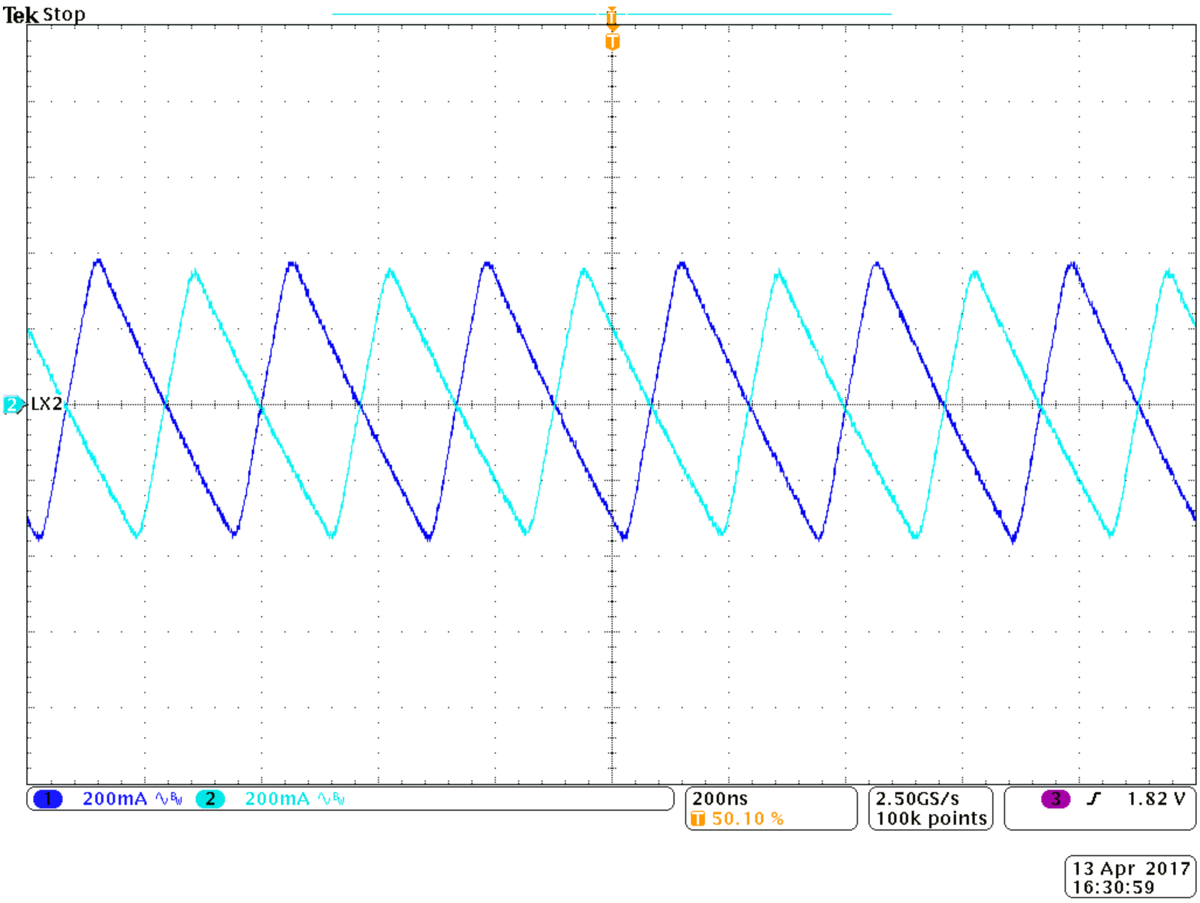




**Figure 14: Inductor Current of Phase-1 and Phase-2, Phase-Shedding Disabled**

There is an easy way to modify the output voltage of BUCKA. This is by using the VBUCKA_CTRL_A and VBUCKA_CTRL_B registers, see the red box in Figure 15.

In the **BUCK_SEL** field choose **VBUCK_A** (see the red arrow in Figure 15), now changing the value in **VBUCKA_CTRL_A** changes the output voltage. This initiates a DVC transition towards the new selected voltage.

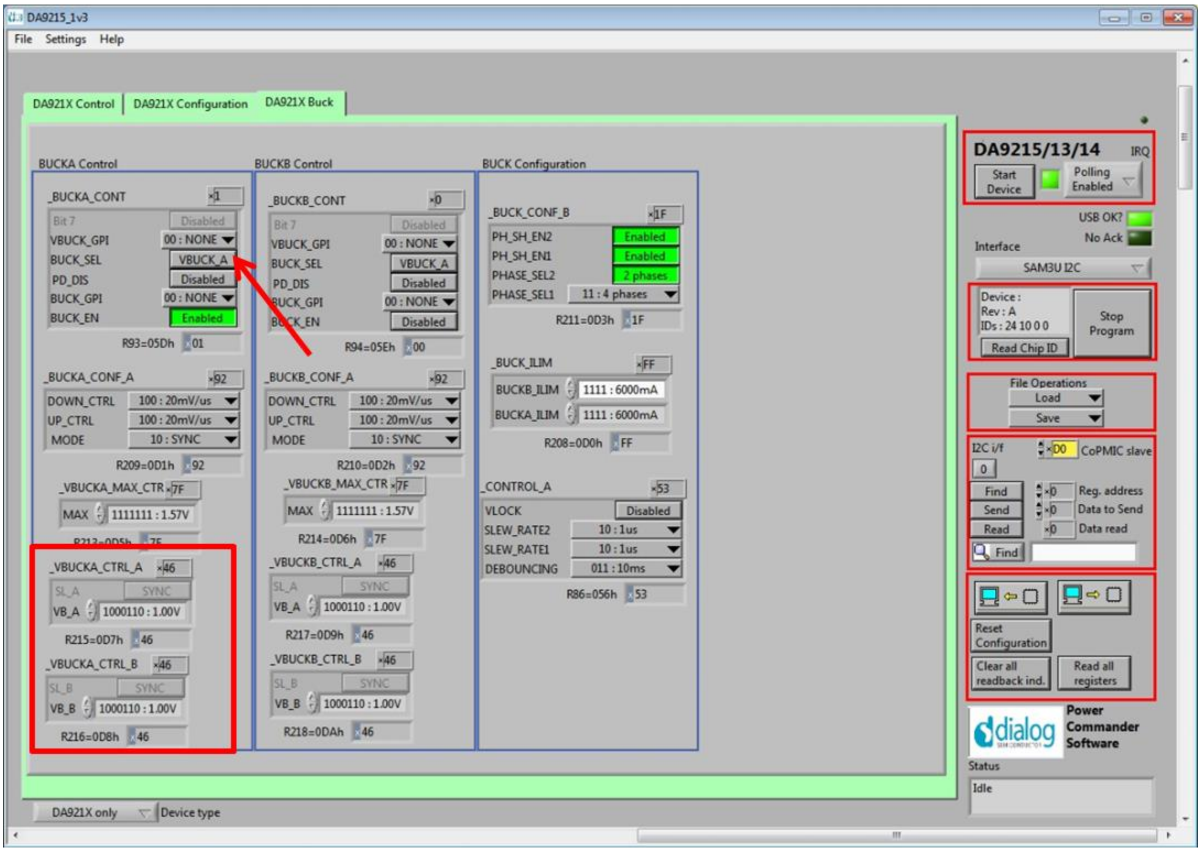


**Figure 15: Output Voltage Control**

By toggling the value on **BUCK_SEL** it is also possible to change the output voltage between the value contained in **VBUCKA_CTRL_A** and the value contained in **VBUCKA_CTRL_B**.

This example configures 1.2 V in **VBUCKA_CTRL_A** and 0.8 V in **VBUCKA_CTRL_B**. After asserting the **BUCK_SEL** bit the DVC transition in the output voltage is as shown in Figure 16.

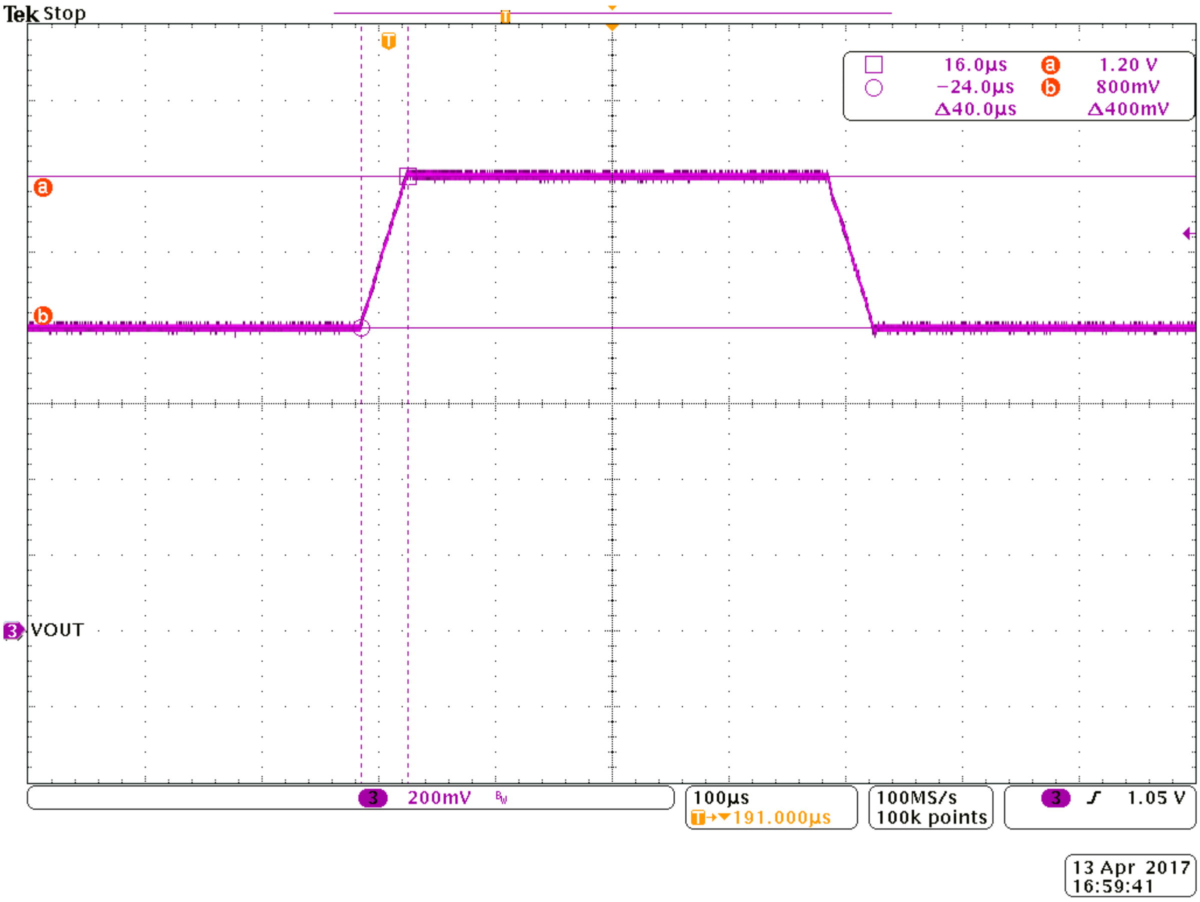



**Figure 16: DVC Transition**

The voltage is decreased by 400 mV in approximately 40 µs, therefore, the slope of the DVC transition can be seen as 10 mV/µs.

The value can be set on the **SLEW_RATE1** field of **CONTROL_A** register. This field is located on the **Buck** tab of the GUI, as shown by the red arrow in Figure 17.

Do not confuse the slew rate setting of DVC voltage changes with the UP_CTRL, DOWN_CTRL of each buck, as they do only refer to the power up and power down slope.

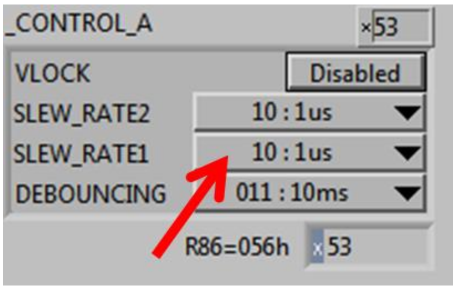



**Figure 17: Buck Tab CONTROL_A Register Field**

There is another means of controlling the output voltage of the buck converter by using one of the GPIO ports available on the device.

This example uses GPI4 in the **VBUCK_GPI** field of **BUCKA_CONT** register. GPIO4 is already selected as the input port active high in the related registers as shown in Figure 18.

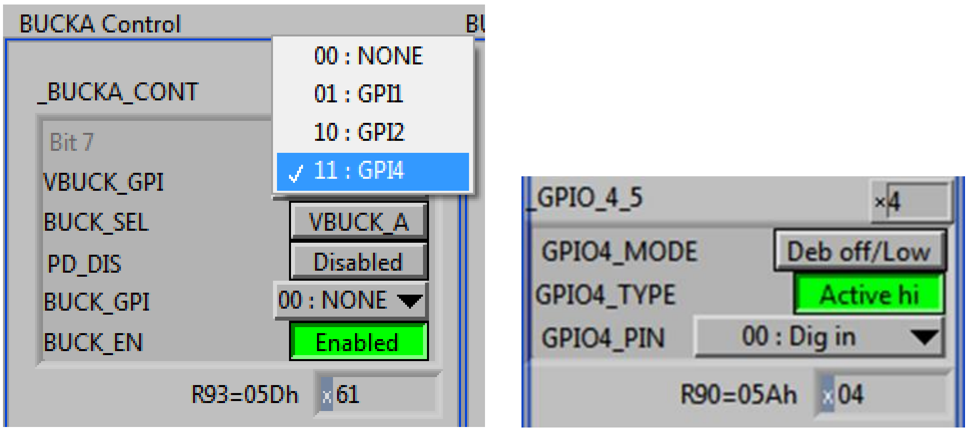



**Figure 18: GPIO Port Voltage Control**

Simply change the connection of red jumper (highlighted by the red circle in Figure 19) which is reflected into a change of the output voltage of BUCKA between VBUCKA_CTRL_A and VBUCKA_CTRL_B. So the BUCK_SEL bit functionality has been replaced by the GPI4 input port.

The same procedure can be applied to the **BUCK_GPI** field, which selects a GPIO to implement the buck enable/disable functionality. By changing the connection of the selected GPIO the buck converter can be seen to enable and disable.

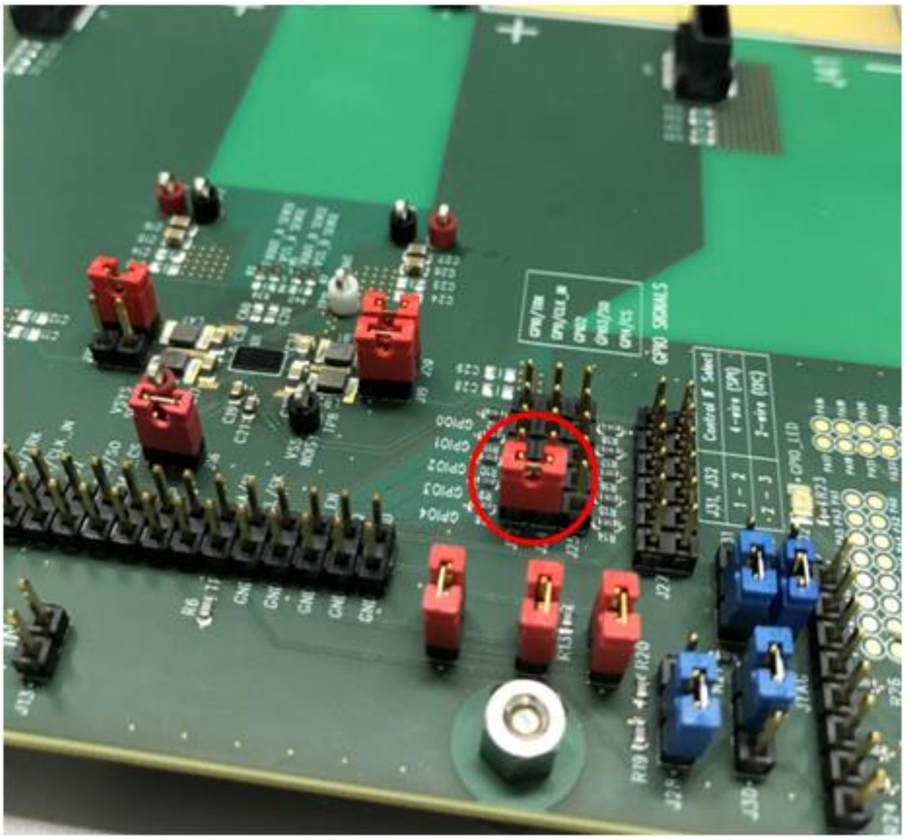


**Figure 19: Port Control of the Buck Output Voltage**

To turn off the whole IC and not only the buck converter use the IC_EN port (see also jumper link J28). By toggling the position of the jumper to the alternative position, the whole IC goes off and the current measured from the supply is close to zero.

DA9214 is a very powerful and flexible device, so it is possible that issues will arise through continued and personalized usage. For questions and clarification please refer to your local Dialog support team.

# 8 Troubleshooting

This section is an aid to resolving some of the more common issues.

## 8.1 Software Issues

The USB device should automatically install without difficulty. Make sure that the installation finds and uses the driver contained on the supplied media.

If the program is started before the USB Interface board is plugged in, the program will default to the offline mode. This can be useful to explore the software in a desk environment without the hardware attached. If the board is subsequently attached, move the Interface control to **SAM3U I2C** or **SAM3U SPI**.

The software can have unpredictable effects when used in conjunction with a USB hub. It is recommended that a direct connection is made to the USB interface board.

The software is optimized for a display screen size of 1024 by 768 pixels or greater, with fonts set to Normal (96dpi).

There have been reported issues of unpredictable display effects when large fonts (120dpi) are used. This can be changed from the Window **Properties**, **Settings** tab; choose **Advanced**, then select **Normal size** from the drop-down menu.

If communications are apparently lost, first click the **Start Device** button. This attempts to make the device go active. Also switching the **Interface mode** to **Offline**, then back to **USB** can reinitialize the USB interface. Alternatively, unplug the USB then reconnect so that the software detects it and reinitializes.

## 8.2 Hardware Issues

Most hardware problems can be traced to incorrect jumper positions.

Check carefully all jumper positions. Use the jumper table details and the board schematic as a guide to the jumper functions and locations, see Appendix A.

Take particular care to ensure the power supply configuration is correct.

It is usually preferable that the USB is deactivated if the device supply is missing. This will mimic the target system operation in which the host processor has no power if the PMIC has no supply. It also avoids unintended current flow between a USB port output and an unpowered GPIO input.

# Appendix A - Links Description

**Table 1: DA9213 Performance Board (227-02-A) Links Description**

| Link | Position 1 (Pin 1-2) | Position 2 (Pin 2-3) | Function |
|---|---|---|---|
| J1 | To VSYS | | Enables diode D1 when VSYS supply is connected |
| J3 | BUCKA local GND | | BUCKA VSS local headers |
| J5 | To GND | | Short VSYS to GND |
| J9 | To VOUT | | Connects L1 to the output of the BUCKA converter |
| J10 | To VOUT | | Connects L2 to the output of the BUCKA converter |
| J11 | To clock input | | Connects GPIO1 to plug for external clock input signal |
| J12 | Signal headers | | I/Os and main signals |
| J13 | to VSYS | | Connects VSYS pin to the VSYS rail |
| J15 | To VOUT | | Connects L3 to the output of the BUCKA converter |
| J16 | to VOUT | | Connects L4 to the output of the BUCKA converter |
| J18 | to VDDIO | | Connections for GPIOs pull up resistors |
| J19 | BUCKA local VOUT | | BUCKA VOUT local headers |
| J20 | | | Connects GPIOs to 100 kΩ pull up resistors |
| J21 | to VDDIO | | connections for nIRQ (via 100 kΩ pull up resistors) |
| J22 | | | Connection for GPIOs |
| J23 | | | Connects GPIOs to GND (via 100 kΩ pull down resistors) |
| J24 | to VDDIO | | Connection for $I^2C$ interface pull up (via 2.2 kΩ resistors) |
| J25 | | | Connects GPIOs to ATSAM3U |
| J26 | to VDDIO<br>(via 100kΩ resistor) | to GND<br>(via 1 kΩ resistor) | Connection for IC_EN |
| J27 | to ATSAM3U port | to J29 | Connection for IC_EN |
| J28 | to SPI_SCLK | to SCL_0 | Selection of SPI or $I^2C$ clock |
| J29 | to SPI_MOSI | to SDA_0 | Selection of SPI or $I^2C$ data |
| J32 | to VDDIO_OUT | | Connects VDDIO to the VDDIO_OUT rail generated by external LDO |
| J33 | to VDDIO_OUT | | Connection of VDDIO_USB used by ATSAM3U |
| J34 | low voltage VDDIO | | Selection of VDDIO_OUT voltage:<br>open => 3.3 V<br>short => 1.8 V |

**Table 2: DA9214 Performance Board (227-04-D) Links Description**

| Link | Position 1 (Pin 1-2) | Position 2 (Pin 2-3) | Function |
|---|---|---|---|
| J2 | BUCKA local GND | | BUCKA VSS local headers |
| J3 | To VSYS | | Enables diode D1 when VSYS supply is connected |
| J6 | To GND | | Short VSYS to GND |
| J10 | To VOUT | | Connects LA2 to the output of the BUCKA converter |
| J11 | To VOUT | | Connects LA1 to the output of the BUCKA converter |
| J13 | To clock input | | Connects GPIO1 to plug for external clock input signal |
| J14 | BUCKA local VOUT | | BUCKA VOUT local headers |
| J15 | Signal headers | | I/Os and main signals |
| J16 | to VSYS | | Connects VSYS pin to the VSYS rail |
| J18 | BUCKB local VOUT | | BUCKB VOUT local headers |
| J19 | To VOUT | | Connects LB2 to the output of the BUCKB converter |
| J20 | to VOUT | | Connects LB1 to the output of the BUCKB converter |
| J21 | to VDDIO | | Connections for GPIOs pull up resistors |
| J22 | | | Connects GPIOs to 100 kΩ pull up resistors |
| J23 | to VDDIO | | connections for nIRQ (via 100 kΩ pull up resistors) |
| J24 | | | Connection for GPIOs |
| J25 | | | Connects GPIOs to GND (via 100 kΩ pull down resistors) |
| J26 | to VDDIO | | Connection for $I^2C$ interface pull up (via 2.2 kΩ resistors) |
| J27 | | | Connects GPIOs to ATSAM3U |
| J28 | to VDDIO<br>(via 100 kΩ resistor) | to GND<br>(via 1 kΩ resistor) | Connection for IC_EN |
| J30 | to ATSAM3U port | to J29 | Connection for IC_EN |
| J31 | to SPI_SCLK | to SCL_0 | Selection of SPI or $I^2C$ clock |
| J32 | to SPI_MOSI | to SDA_0 | Selection of SPI or $I^2C$ data |
| J36 | BUCKB local GND | | BUCKB VSS local headers |
| J37 | to VDDIO_OUT | | Connects VDDIO to the VDDIO_OUT rail generated by external LDO |
| J38 | to VDDIO_OUT | | Connection of VDDIO_USB used by ATSAM3U |
| J39 | low voltage VDDIO | | Selection of VDDIO_OUT voltage:<br>open => 3.3 V<br>short => 1.8 V |

**Table 3: DA9215 Performance Board (227-07-B) Links Description**

| Link | Position 1 (Pin 1-2) | Position 2 (Pin 2-3) | Function |
|---|---|---|---|
| J1 | To VSYS | | Enables diode D1 when VSYS supply is connected |
| J5 | BUCKB local VOUT | | BUCKB VOUT local headers |
| J7 | BUCKB local GND | | BUCKB VSS local headers |
| J8 | BUCKA local GND | | BUCKA VSS local headers |
| J9 | To GND | | Short VSYS to GND |
| J13 | To clock input | | Connects GPIO1 to plug for external clock input signal |
| J14 | to VOUT | | Connects LB1 to the output of the BUCKB converter |
| J15 | To VOUT | | Connects LA3 to the output of the BUCKA converter |
| J17 | Signal headers | | I/Os and main signals |
| J18 | to VSYS | | Connects VSYS pin to the VSYS rail |
| J20 | To VOUT | | Connects LA1 to the output of the BUCKA converter |
| J21 | To VOUT | | Connects LA2 to the output of the BUCKA converter |
| J22 | to VDDIO | | Connections for GPIOs pull up resistors |
| J23 | BUCKA local VOUT | | BUCKA VOUT local headers |
| J25 | to VDDIO | | connections for nIRQ (via 100 kΩ pull up resistors) |
| J26 | | | Connects GPIOs to 100 kΩ pull up resistors |
| J27 | | | Connection for GPIOs |
| J28 | to VDDIO | | Connection for $I^2C$ interface pull up (via 2.2 kΩ resistors) |
| J29 | | | Connects GPIOs to GND (via 100 kΩ pull down resistors) |
| J30 | to VDDIO<br>(via 100 kΩ resistor) | to GND<br>(via 1 kΩ resistor) | Connection for IC_EN |
| J31 | | | Connects GPIOs to ATSAM3U |
| J32 | to ATSAM3U port | to J29 | Connection for IC_EN |
| J33 | to SPI_SCLK | to SCL_0 | Selection of SPI or $I^2C$ clock |
| J34 | to SPI_MOSI | to SDA_0 | Selection of SPI or $I^2C$ data |
| J39 | to VDDIO_OUT | | Connects VDDIO to the VDDIO_OUT rail generated by external LDO |
| | | | |
| J40 | to VDDIO_OUT | | Connection of VDDIO_USB used by ATSAM3U |
| J38 | low voltage VDDIO | | Selection of VDDIO_OUT voltage:<br>open => 3.3 V<br>short => 1.8 V |

# Appendix B - Performance Board Features

DA9214 Performance Board (227-04-D) is used in this section as an example. DA9213 and DA9215 Performance Boards (227-02-A and 227-07-B) have the same features.

## B.1 USB-I²C Bridge

The USB-I²C bridge is mainly used for two purposes:

1. As a source of I²C and SPI control signals.
2. To provide the discrete signals for the GPIOs.

The USB-I²C bridge is powered via the USB cable via a fixed 3.3 V regulator.

Jumpers J31 and J32 on 227-04-D are used to select between I²C and SPI communication to DA9214. This has to be configured with the IF_TYPE field of INTERFACE2 register.

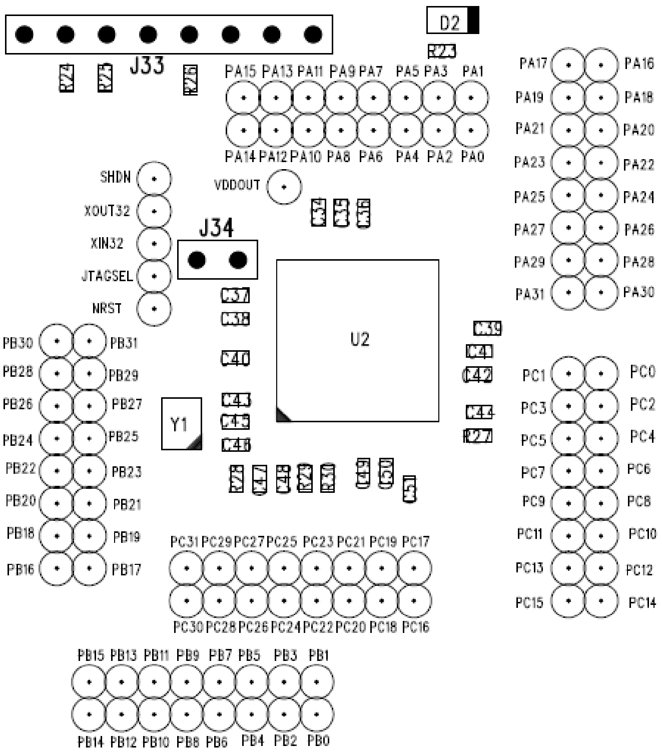



**Figure 20: ATSAM3U µ-Controller for USB-I²C bridge**

## B.2 I/O and Supply Regulators

The external LDO regulators on the DA9214 Performance Board are used to generate power supply for the ATSAM3U and the I/O voltage needed by the device.

A voltage of 3.3 V or 1.8 V can be selected as supply for the I/O 9 (see jumper link J39).

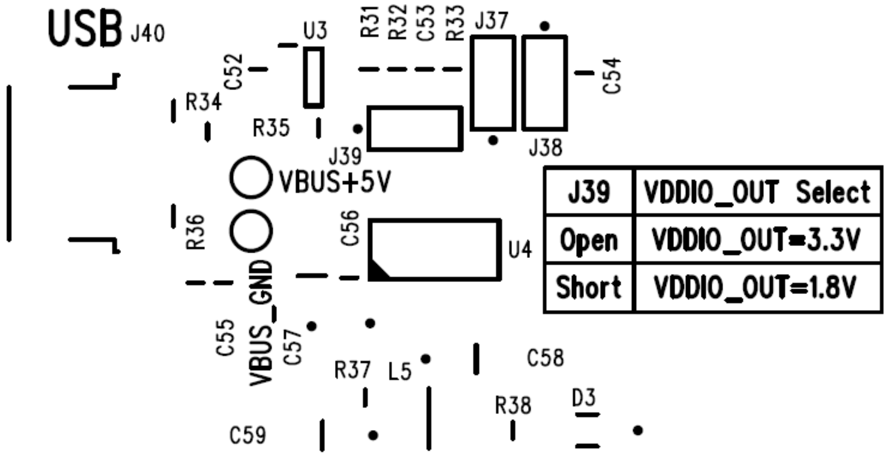



**Figure 21: Mini USB Input and Internal Rails Generation**

## B.3 Control and I/O Signals

All of the control and I/O signals from DA9214 appear on the headers pins shown above. These are the most useful monitor points for debug purposes.

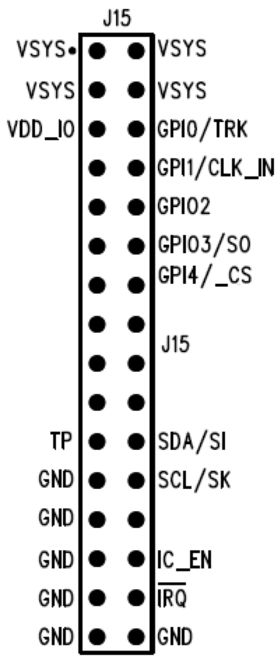



**Figure 22: Control and I/O Signals Headers**

If required, a suitable connector can bring these signals to the system board for integrated development. If used in this way, the other links which also control these pins should be removed to avoid logic or voltage clashes.

The arrangement of the jumper links to the GPIO pins is shown in Figure 23. The jumpers may be inserted on J27 for USB control and monitor. The jumpers can also be placed on the headers J22, J24, J25 as follows:

- Between J22 and J24, connecting GPIOs to VDD_IO via 100 kΩ pull up resistor
- Between J24 and J25, connecting GPIOs to ground via 100 kΩ pull down resistor
- Left open for external connections

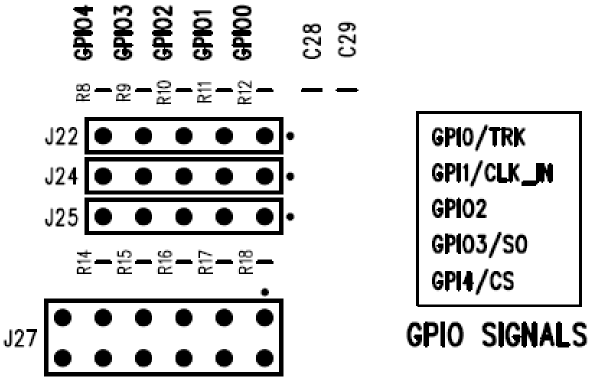



Figure 23: GPIOs Control Jumpers

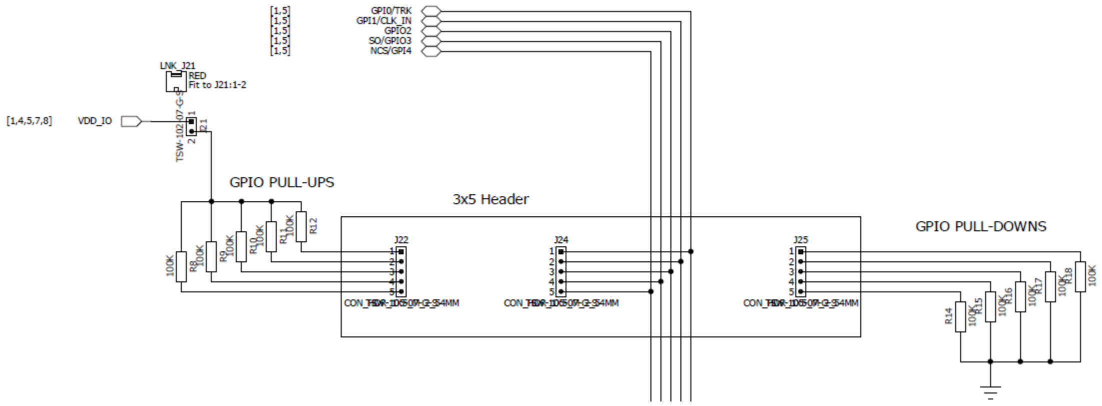



Figure 24: GPIOs Control Schematic

# Appendix C - Scripting

The software includes the ability to save and load a text file containing hex codes representing the register addresses and data. This allows a basic scripting functionality on DA9213, DA9214, and DA9215 for advanced users.

If you wish to use register names directly (easier to read), instead of register addresses, from the **Settings** menu select **Reg names in file**. When selected the slave address is replaced by the word **WRITE2**. This is generally preferable and more readable. The register names are stored in an external file. Hex codes for slave address and register address are still accepted on reading in the file.

**Note 2** The suffix 2 is used to identify coPMIC devices (such as DA9213, DA9214, or DA9215) and distinguish them from the main PMIC devices (such as DA9063). Although they can share the I$^2$C address if STAND_ALONE is set to zero (see Datasheet for details on STAND_ALONE bit operation), the GUI offers the possibility to assign different I$^2$C addresses to main and coPMIC devices. Thus **WRITE** automatically uses the main PMIC I$^2$C address and **WRITE2** uses the coPMIC I$^2$C address assigned, see Section 6.3.

## C.1 Text File Format

- Numbers apart from time delays are expressed in Hex, separated by tabs. The use of 0x in front of the hex value is optional.
- The first parameter is the device slave address in 8 bit format.
- Alternatively the first parameter can be a token:
  - **WRITE2** writes to the device at the currently selected slave address (I$^2$C mode only)
  - **READ2** reads from the device a the values of a number of registers
  - **PORT** sets the selected digital control line to the specified value (1 or 0)
  - **PORTREAD** reads the value of the specified digital control line
  - **PORTDIR** sets the direction of the digital control line to an input if the value is 0, or an output if set to 1.
  - **DELAY** or **WAIT** implements a time delay specified up to 65535 milliseconds. The delay time is specified in decimal, or hex if preceded by 0x.
  - **ITERATE** causes the whole script to be repeat the specified number of times.
- If you wish to use the GPIO-related commands, you need to jumper the respective jumpers of J27, to allow the ATSAM3U µC device to really operate on the specific port. The levels are not forced by DA92112 but instead this is done externally by the ATSAM3U.
- The second parameter is the register address as a name or hex value.
- The third parameter is the data.
- Comments (lines beginning with //) are permitted in the file.
- Inline comments (//comment) are permitted.
- The data will be processed in the order written, and written directly to the specified device. The screen controls will be updated once command in the file have finished.
- The use of the slave address in the file allows any device attached to the I$^2$C bus to be controlled.
- The token PORT2 allows control over the GPIOs which are configured as inputs. The second parameter is the port name. The third parameter is 0 or 1.
- For read operations, the result of the read is passed to the history log window

## C.2 Examples

```
//Test DA9214
//17.03.2017

//Read registers in 2 ways
//See the equivalent results
//Read from register 0xD1 n_BUCK1_CONF_A
//for the next 10 registers
READ2 n_BUCK1_CONF_A 10
READ2 0xD1 10

//Change value of register n_BUCK1_CONF_A
WRITE2 n_BUCK1_CONF_A FF
//Read back the new configured value
READ2 n_BUCK1_CONF_A 1

//Configure the GPIO2 as output port
//Note the "_N" identifying the slave
PORTDIR GPIO2_N 1
//Set the level of GPIO2 low
PORT GPIO2_N 0
//Read back the voltage at GPIO2
PORTREAD GPIO2_N
//Set the level of GPIO2 high
PORT GPIO2_N 1
//Read back the voltage at GPIO2
PORTREAD GPIO2_N
```

The results are displayed in the history log, which can be viewed by selecting **Setting**, **History Log**.

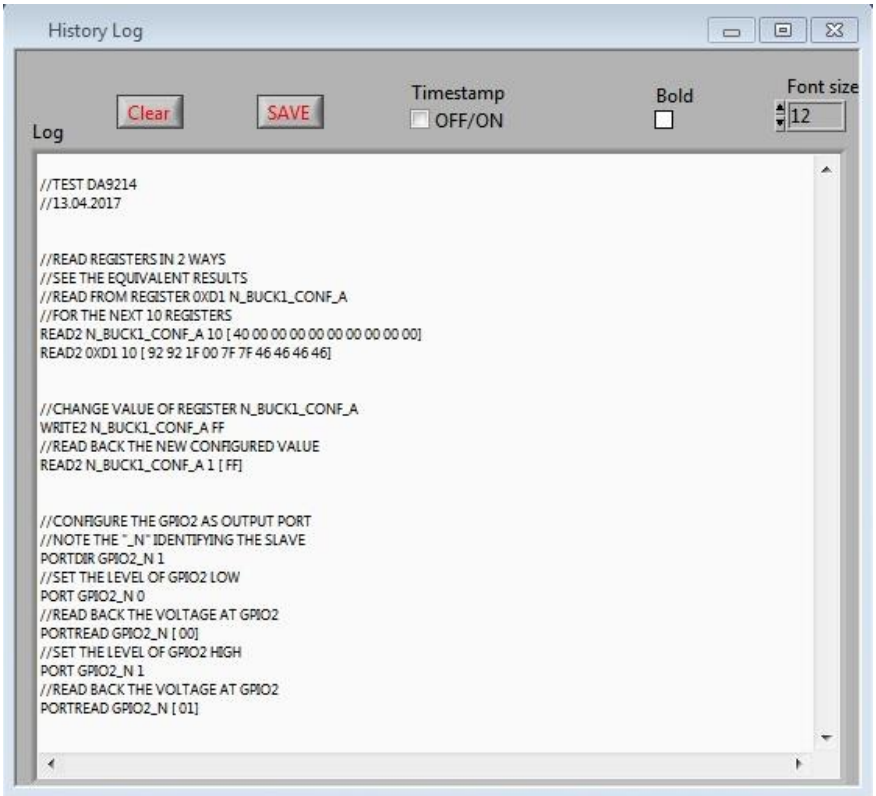



**Figure 25: History Log**

# Revision History

| Revision | Date | Description |
|---|---|---|
| 1.0 | 26-May-2017 | Initial version. |
| 2.0 | 25-Feb-2022 | File was rebranded with new logo, copyright and disclaimer |

## Status Definitions

| Status | Definition |
|---|---|
| DRAFT | The content of this document is under review and subject to formal approval, which may result in modifications or additions. |
| APPROVED<br>or unmarked | The content of this document has been approved for publication. |

## RoHS Compliance

Dialog Semiconductor's suppliers certify that its products are in compliance with the requirements of Directive 2011/65/EU of the European Parliament on the restriction of the use of certain hazardous substances in electrical and electronic equipment. RoHS certificates from our suppliers are available on request.

### Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com


### Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.