User Manual

DA14531 Frequently Asked Questions

UM-B-149

Abstract

This document addresses the Frequently Asked Questions for the DA14531

Contents

Abstract **1**

1 Terms and Definitions **4**

2 Software **5**

2.1 Can I Remap SWD Pins to GPIOs? 5

2.2 Can I Use the Old Pro-DK (DA14580DEVKT-P) to Program a DA14531 Daughter Board via 1-wire UART? 5

2.3 How to Enable and Test the Debug UART Mechanism in the DA145xx Pro Development Kit? 6

2.4 How Many Sleep Modes are Available in DA14531 and How to Configure These? 8

2.5 How to Make the DA14531 Advertising with Advertising Scan Indication (ADV_SCAN_IND) Packets? 9

2.6 Can I Use Any of the DA14531 Timers in Sleep Mode? 9

2.7 What Development Path Should I Follow to Migrate my FW from DA14585 to DA14531? 9

2.8 What is the difference between Clockless wake-up and Clocked wake-up controller? 9

2.9 Can the HW Reset be Mapped to Any Other GPIO? 9

2.10 How can I put the system into hibernation (clockless or shipping) mode? 10

2.11 How can I the QDIDs of the DA14531? 10

2.12 How can I Modify Advertising Data 10

2.13 Are the DSPS and AT CodeLess Available for the DA14531 and TINY Module? 10

3 Hardware **11**

3.1 Can I Supply a Load (Such as a Sensor, LED, Flash Memory, etc.) from the Output of the DC-DC when Boost Mode is Active? 11

3.2 Is it Possible to Create a Two-Layer Board ? 11

3.3 When Does Bypass Mode Make Sense? 11

3.4 Can I NOT Use External 32 kHz Crystal as a Low Power Clock? 11

3.5 Can I Program the External FLASH with 2-wire UART? 11

3.6 How Can I Burn the SPI Flash with the DA145xx Development Kit? 11

3.7 Which Retention-RAM Blocks can be Retained in DA14531? 12

3.8 Does Programming the DA14531 from 2-wire UART Disable the Reset Functionality on P0_0? 12

3.9 Can the DA14531 in the WLCSP Package be Configured to Boot from External SPI Flash and Still have the UART and RESET Functionality? 12

3.10 What are the Expected Power Consumption Values for DA14531? 13

3.11 What is the OTP configuration script? 13

3.12 How Can the SPI Operate at 32 MHz (Fast Boot)? 13

3.13 Can an External Processor Control the DA14531? 13

3.14 Is the DA14531 FCC Certified? 13

3.15 How I can Find Schematics Symbols and PCB Footprints in Any Format? 13

3.16 Can I Use a DA14531 Development Kit (USB or PRO) to Debug my Custom Hardware? 14

3.17 Does Dialog Offer a DA14531 Module? 14

3.18 What is the Expected Power Consumption for the DA14531 Module? 14

4 RF & Radio **15**

4.1 What is the Intermediate Frequency (IF) of the Radio? 15

4.2 What does the Local Oscillator (LO) Frequency Does the Radio Usage? 15
4.3 What Communication Range can be Achieved with the DA14531? 15
4.4 Is a Pi Filter on the RFIOp Port Required? 15
4.5 Is 2 Mb/sec Data Throughput Supported? 15
4.6 Can I Change the Tx Power Level Using the Production Test Firmware? 16
4.7 Is Any Calibration of the RF Parameters Required in Production? 18

Revision History 19

Figures

Figure 1: DA1458x Pro DK Wiring 5
Figure 2: SmartSnippets Toolbox Board Setup 6
Figure 3: Define CFG_PRINTF Macro 6
Figure 4: UART2 Configuration Settings 7
Figure 5: Include arch_console.h Header 7
Figure 6: The DA145xx Wiring 8
Figure 7: Result in the Serial Monitor 8
Figure 8: DA14531 Retention-RAM 12
Figure 9: Tx Power Command 16
Figure 10: Tx Power Response 16
Figure 11: Configuration of DA145xx Pro DK for Single-Wire UART Applied on P05 17
Figure 12: Configuration of user_periph_setup.h for Single-Wire UART Applied on P05 17

1 Terms and Definitions

BLE	Bluetooth® Low Energy
COM	Communication Port
EEPROM	Electrically Erasable Programmable Memory
GPIO	General Purpose Input/Output
HCI	Host Controller Interface
HW	Hardware
IDE	Integrated Development Environment
IF	Intermediate Frequency
LED	Light Emitting Diode
LO	Local Oscillator
OTP	One Time Programmable
RF	Radio frequency
RFI	Op Radio frequency Input/Output
SDK	Software Development Kit
SoC	System on Chip
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
SWD	Serial Wire Debug
SW	Software
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

2 Software

This section describes the software-related Frequently Asked Questions.

2.1 Can I Remap SWD Pins to GPIOs?

Yes, it is possible to remap the SWD to different GPIOs.

The SWD signal mapping is defined by SYS_CTRL_REG[DEBUGGER_ENABLE] bitfield. Set the this bitfield to:

- **0x0** to disable the debugger
- **0x1** to configure the SW_CLK and SW_DIO to P0_2 and P0_5 accordingly (default option for WLCSP17 package)
- **0x2** to configure the SW_CLK and SW_DIO to P0_2 and P0_1 accordingly
- **0x3** to configure the SW_CLK and SW_DIO to P0_2 and P0_10 accordingly (default option for FCGQFN24 package). This option is not available in WLCSP17 package

Example:

To configure the SW_CLK and SW_DIO to P0_2 and P0_1, the SYS_CTRL_REG[DEBUGGER_ENABLE] bitfield should be programmed with 0x2, as follows :

```
SetBits16(SYS_CTRL_REG, DEBUGGER_ENABLE, 2);
```

NOTE
The current settings are applicable as soon as user firmware is running. If the device resets or reboots, then the booter will revert the SWD pins back to the default settings. The procedure is also described in Note 1 of the DA14531 datasheet. Link: https://www.dialog-semiconductor.com/sites/default/files/2021-03/DA14531_datasheet_3v3_0.pdf

2.2 Can I Use the Old Pro-DK (DA14580DEVKT-P) to Program a DA14531 Daughter Board via 1-wire UART?

Yes, you can use it to download firmware via 1-wire UART. When the DA14531 daughter board is attached to the old Pro-DK, then the **P0_5** is exposed on header **J7**.**Pin6 (P2_5)** .

Jumper wires should be used from J5 to J7 plus a 100 Ohm resistor, as shown in Figure 1.

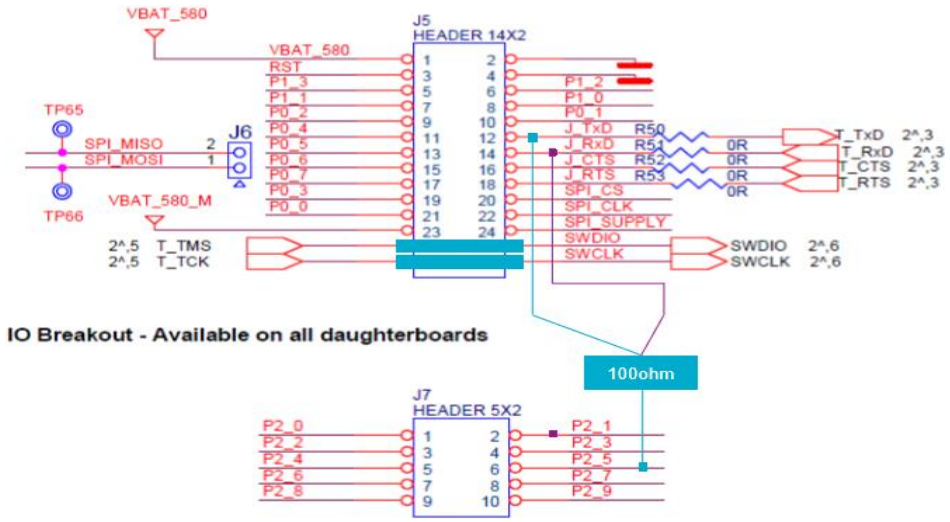

Figure 1: DA1458x Pro DK Wiring

Example:

How to program the DA14531 with the Production Test Firmware via 1-wire UART using the old Pro-DK.

Steps:

1. Use the Keil IDE to open and compile the Production Test project from the SDK6.0.14. The Keil project is located under the projectstarget_appsprod_testprod_testKeil_5 SDK path. The already exiting binary for the Production Test can be also used, which is located under binariesda14531prod_test.
2. After compilation, create the prod_test_531.hex in the Keil_5out_DA14531Objects folder.
3. Open SmartSnippets Toolbox V5.0.16. See the UM-B-083 for more information.
4. In Board Setup select **single wire UART** communication (see Figure 2).

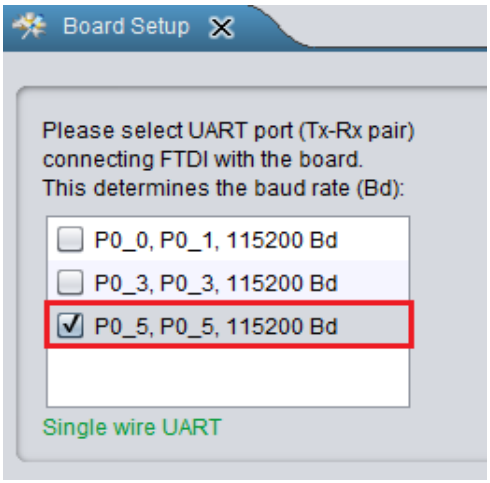

Figure 2: SmartSnippets Toolbox Board Setup

5. Open the RF Master.
6. Download the prod_test_531.hex.
7. Reset the DA1458x Pro DK. You can now use the RF Master.

2.3 How to Enable and Test the Debug UART Mechanism in the DA145xx Pro Development Kit?

The *ble_app_barebone* example of the SDK6.0.14 is used for this demonstration. The project is located under the *projects\target_apps\ble_examples* SDK path. Follow the steps to enable the debug UART mechanism. The DA145xx Pro Development Kit is required as well.

Steps:

1. Define the **CFG_PRINTF** macro in da1458x_config_basic.h (see Figure 3).

```
/*********************************************************************************************************************/
/* UART Console Print. If CFG_PRINTF is defined, serial interface logging mechanism will be enabled.                 */
/* If CFG_PRINTF_UART2 is defined, then serial interface logging mechanism is implented using UART2, else UART1 */
/* will be used.                                                                                                     */
/*********************************************************************************************************************/
#define CFG_PRINTF
#ifdef CFG_PRINTF
    #define CFG_PRINTF_UART2
#endif
```

Figure 3: Define CFG_PRINTF Macro

2. UART2 settings are defined in user_periph_setup.h, which can be changed according to application requirements. P06 is used to enable UART2 Tx functionality (see Figure 4).

```
/****************************************************************************************************/
/* UART2 configuration to use with arch_console print messages                                      */
/****************************************************************************************************/
// Define UART2 Tx Pad
#if defined (__DA14531__)
    #define UART2_TX_PORT           GPIO_PORT_0
    #define UART2_TX_PIN            GPIO_PIN_6
#else
    #define UART2_TX_PORT           GPIO_PORT_0
    #define UART2_TX_PIN            GPIO_PIN_4
#endif

// Define UART2 Settings
#define UART2_BAUDRATE              UART_BAUDRATE_115200
#define UART2_DATABITS              UART_DATABITS_8
#define UART2_PARITY                UART_PARITY_NONE
#define UART2_STOPBITS              UART_STOPBITS_1
#define UART2_AFCE                  UART_AFCE_DIS
#define UART2_FIFO                  UART_FIFO_EN
#define UART2_TX_FIFO_LEVEL         UART_TX_FIFO_LEVEL_0
#define UART2_RX_FIFO_LEVEL         UART_RX_FIFO_LEVEL_0
```

Figure 4: UART2 Configuration Settings

3. Include the arch_console.h header file in user_barebone.c (see Figure 5).

```
/*
 * INCLUDE FILES
 ****************************************************************************************************
 */

#include "rwip_config.h"             // SW configuration
#include "gap.h"
#include "app_easy_timer.h"
#include "user_barebone.h"
#include "co_bt.h"
#include "arch_console.h"
```

Figure 5: Include arch_console.h Header

4. Add the following code in user_app_adv_start()

```
char ch = 0xFF;
arch_printf("Device starts advertising") ;
arch_printf("\n\r") ;
arch_printf("Print 1 byte") ;
arch_puts(&ch);
arch_printf("\n\r");
```

5. Install the wiring from **J2.27** to **J1.17** as shown in Figure 6.

Figure 6: The DA145xx Wiring

6. For Teraterm Configurations, make sure that the correct COM port is selected. The COM port is likely to be different in your setup. Select the lowest number of the two options.
7. Run the example and the result will show in the Serial Monitor (see Figure 7).

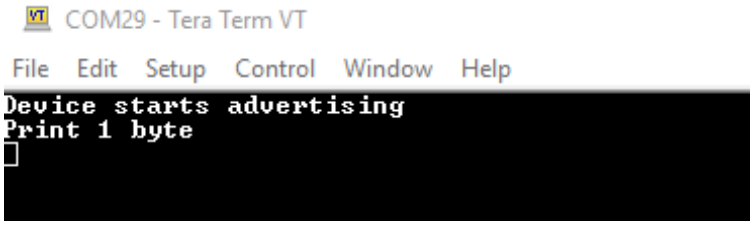

Figure 7: Result in the Serial Monitor

2.4 How Many Sleep Modes are Available in DA14531 and How to Configure These?

1. DA14531 SoC supports three (3) sleep modes, namely:
2. Extended sleep mode (with or without OTP copy).
3. Deep sleep mode (also known as shipping clocked mode).
4. Hibernation mode (also known as shipping clock-less mode)

See the Sleep Mode Tutorial for an overview on how to configure the DA14531 SoC in any of the three available sleep modes.

In addition to the tutorial, please check out the DA14531 Bluetooth® Low Energy Eddystone beacon with Hibernation Or Deep Sleep Mode and DA14531 Configuring Hibernation and State-aware hibernation mode SW Examples.

2.5 How to Make the DA14531 Advertising with Advertising Scan Indication (ADV_SCAN_IND) Packets?

The **ADV_SCAN_IND PDU** is used from the peripheral to send a scan response when operating in non-connectable mode. The device should be configured in a non-connectable advertising mode and also the scan response data should be defined. For more information, see the Simple Beacon Example and Bluetooth® Low Energy Advertising Tutorial.

2.6 Can I Use Any of the DA14531 Timers in Sleep Mode?

Yes. Timer 1 can be kept active in sleep mode as the clock source is selectable between **System Clock (sys_clk)** and **Low Power Clock (lp_clk).** For more information, see the DA14531 Timer1 Software Example.

NOTE
When the DA14531 is configured to any of the available sleep modes, the System Clock is disabled, so the device will run with the Low Power Clock.

2.7 What Development Path Should I Follow to Migrate my FW from DA14585 to DA14531?

Our latest SDK6.0.14 is common for DA14531 and DA14585/586. If an existing application is based on DA14585/586 and SDK6.0.10, the application can be easily ported to DA14531 and SDK6.0.14. Most of the functions will be changed automatically. Just some configurations must be checked before finishing the migration. The SDK Porting Guide document describes the changes and the steps needed to port an application that has been developed under DA14585/DA14586 SDK 6.0.10 release, to the latest DA14585/586/531 SDK 6.0.14 release.

2.8 What is the difference between Clockless wake-up and Clocked wake-up controller?

The difference is that **Clockless wake-up controller** is only used when the system is in hibernation mode and the **Clocked wake-up controller** is used when the system is either in extended or deep sleep mode. In hibernation mode all clocks are stopped, but in extended or deep sleep mode the wakeup controller resides in the **PD_SLP** power domain and operates on the **LP_CLK**.

See **Sections 17 and 18** in the DA14531 datasheet for more information on Clockless and Clocked wake-up controllers.

The Sleep Mode tutorial demonstrates the Clockless wake-up. The Bluetooth® Low Energy Notify Button Wakeup SW Example demonstrates the Clocked wake-up.

2.9 Can the HW Reset be Mapped to Any Other GPIO?

No, the HW reset is dedicated to **P0_0** and can be only disabled.

To disable it, the WR_CTRL_REG[DISABLE_HWR] bitfield should be set to **1**.

Additionally, the GPIO driver of the SDK provides a function to disable the hardware reset functionality on P0_0 : GPIO_Disable_HW_Reset()

The driver is located in 6.0.14.1114sdkplatformdrivergpio SDK path.

However, the **Power-On-Reset (POR)** can be mapped to any of the available GPIOs.

The POR_PIN_REG[POR_PIN_POLARITY] and POR_PIN_REG[POR_PIN_SELECT] bifields should be programmed to set the polarity and select the POR GPIO accordingly.

The reset time can be set by programming the POR_PIN_REG[POR_TIMER_REG].

POR_TIME: Time = POR_TIME x 4096 x RC32K clock period

The user must keep the reset pin to high or low (based on the set POLARITY) for the duration programmed for POR_TIME.

2.10 How can I put the system into hibernation (clockless or shipping) mode?

The entire process to achieve the clockless mode is explained in the Configuring to Hibernation Mode SW Example.
The Sleep Mode tutorial also demonstrates how to configure the DA14531 into hibernation mode.

2.11 How can I the QDIDs of the DA14531?

Use the following links to find the QDIDs of DA14531 for Controller, Host and Profile Subsystems:

1. QDID Controller Subsystem
2. QDID Host Subsystem
3. QDID Profile Subsystem

2.12 How can I Modify Advertising Data

According to Bluetooth® Low Energy specifications, the adverting data (and the scan response data) should follow a specific format. Refer to **Section 3.1. Changing the Advertising Data** in the Bluetooth® Low Energy Advertising Tutorial to see the proper step to change the advertising string.

2.13 Are the DSPS and AT CodeLess Available for the DA14531 and TINY Module?

Yes, DSPS and CodeLess available for the DA14531 and TINY module.

Dialog Serial Port Service (DSPS)

The DSPS emulates a serial cable communication. It provides a simple substitute for RS-232 connections, including the familiar software flow control logic via Bluetooth® Low Energy. The SPS software distribution includes the application and profile source codes and supports GAP Central/Peripheral roles.

Click on the DSPS support page for more info!

CodeLess™ AT Commands

The CodeLess allows you to quickly get started with wireless IoT applications with a set of AT Commands. The CodeLess AT commands platform allows control over a local UART connected device as well as a remote device via Bluetooth® Low Energy. You can create simple demos / applications / proof of concepts without any code development or build you own application on top!

Click on the CodeLess support page for more info!

3 Hardware

This section describes the hardware-related Frequently Asked Questions.

3.1 Can I Supply a Load (Such as a Sensor, LED, Flash Memory, etc.) from the Output of the DC-DC when Boost Mode is Active?

Yes. **Table 12** in the DA14531 datasheet specifies the maximum load current that the DC-DC can supply at various output voltages. In boost mode the DC-DC output is used to power the clamps, POR circuit, read/write the OTP and drive the GPIO's. If the OTP is not being read/written and the GPIO's are not being driven, then at least 95% of the output current capability is available to supply an external load. Note that during boot, no more than 50 µA should be drawn from the VBAT_HIGH rail (see **Table 4** in the **DA14531 datasheet**). This may mean that the load will have to be connected to a GPIO rather than directly to the output of the DC-DC.

3.2 Is it Possible to Create a Two-Layer Board ?

Yes. No micro-vias are needed for both packages. The application note DA14531 Hardware Guidelines gives examples with 4-layer PCB, but this can be reduced to two layers. The middle layers are only used to interconnect the top and bottom layer.

3.3 When Does Bypass Mode Make Sense?

Bypass mode can be used when the lowest power consumption is not important. Bypass mode prevents use of the DC-DC Inductor. This saves $0.02 and reduces the board size!

See the following two(2) hardware design examples for DA14531 QFN/Bypass and DA14531 WLCSP/Bypass configurations.

In the software side, the **CFG_POWER_MODE_BYPASS** macro should be defined in the da1458x_config_basic.h file.

3.4 Can I NOT Use External 32 kHz Crystal as a Low Power Clock?

The 32 kHz crystal is “truly” optional. Internal RCX can be used as **Low Power Clock (LPC)** instead of 32 kHz crystal.

In SDK6.0.14, the Low Power Clock can be selected by setting the **CFG_LP_CLK** macro in da1458x_config_advanced.h to one of the following values:

- LP_CLK_XTAL32 //External XTAL32 oscillator
- LP_CLK_RCX20 //Internal RCX20 clock
- LP_CLK_FROM_OTP //Use the selection in the corresponding field of OTP Header

3.5 Can I Program the External FLASH with 2-wire UART?

Yes. **Figure 32** in document DA14531 Getting Started shows the Jumper configuration to program the external SPI FLASH through 2-wire UART. In addition, application note DA14531 Booting Options describes different boot options with serial interfaces such as I2C, UART, SPI and how to program a firmware into Flash, EEPROM and OTP.

3.6 How Can I Burn the SPI Flash with the DA145xx Development Kit?

Use the **Flash Programmer** from SmartSnippets toolbox. There are three options available:

1. Use JTAG. **Figure 30** in document DA14531 Getting Started shows the Jumper configuration to program the SPI FLASH through JTAG. Use default Pin configurations.
2. Use 1-Wire UART either via P05 or P03.
3. Use 2-Wire UART. **Figure 31** in document DA14531 Getting Started shows the Jumper configuration to program the SPI FLASH with 2-wire UART.

You can also refer to the ezFlashCLI tool to flash DA14531. The ezFlashCLItool relies on the Segger J-Link™ library to control the Smartbond SWD interface.

In case of theDA14531 TINY Module, the Dialog Smartbond™ Flash Programmer can be used as well. It is available for Windows OS, Linux OS and Mac OS. Please see the User Manual.

3.7 Which Retention-RAM Blocks can be Retained in DA14531?

DA14531 has **three (3)** RAM cells that can be retained individually. If the **CFG_CUSTOM_SCATTER_FILE** macro is undefined in da1458x_config_advanced.h file, then the system will calculate which blocks to retain based on the default SDK scatter file and the size of the current image.

- **RAM3 block is always retained,** since it contains data that are used from ROM and needs to be retained.
- **RAM1 holds the IVT** so it needs to be retained, thus if an image is small enough to fit into RAM1 it is possible to power off RAM2 cell.

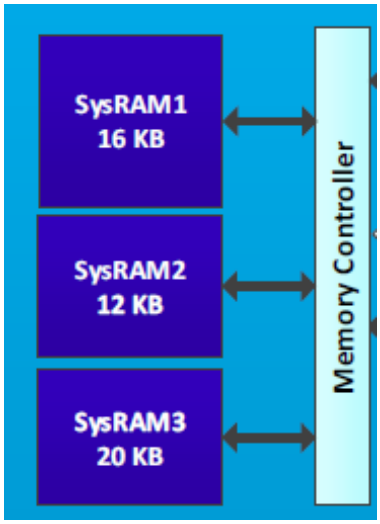

Figure 8: DA14531 Retention-RAM

3.8 Does Programming the DA14531 from 2-wire UART Disable the Reset Functionality on P0_0?

No, the RESET function is assigned by default to pin **P0_0.**

The reset functionality can be disabled via the HWR_CTRL_REG by application as soon as user firmware is running. The default function of the **P0_0** is the HW reset: when the booter is running the reset functionality is disabled from the booter when the sequential scanning is running and **P0_0** is used as an interface pin.

As soon as the user software is downloaded, the booter will re-enable the reset function.

3.9 Can the DA14531 in the WLCSP Package be Configured to Boot from External SPI Flash and Still have the UART and RESET Functionality?

It is better to use JLINK to program such a condition. The RESET function can be based on power-on-reset or change to another pin. The CSP package has less pins than the WLCSP package so, it is hard to support all functions together. In addition, application note DA14531 Booting Options describes the different boot options that use serial interfaces such as I2C, UART, SPI and how to program a firmware into Flash, EEPROM and OTP.

3.10 What are the Expected Power Consumption Values for DA14531?

See section 3.3 DC Characteristics in the DA14531 datasheet.

3.11 What is the OTP configuration script?

The OTP configuration script is used to program registers with values that are defined during production, testing, to store user trim values for the application software and to define the UART time-out timer during boot. The OTP configuration script is executed by the Booter to prepare and initialize the system before the CPU starts to run the application code. See **Section 4.4.2** Configuration Script in the DA14531 datasheet.

3.12 How Can the SPI Operate at 32 MHz (Fast Boot)?

The SPI Flash operates at 2 MHz by default. However, a specific configuration script command can be used to overwrite the default 2 MHz clock speed of the SPI boot path and set the clock speed to 32 MHz.

3.13 Can an External Processor Control the DA14531?

Yes, it is possible. There are **two (2)** possibilities where an external processor can control the DA14531.

1. Over GTL, which is a Dialog proprietary protocol.

See **Section 1.4.2**. External Processor in UM-B-119: DA14585-DA14531 SW Platform Reference. The SDK includes two projects for this kind of applications namely prox_reporter_ext and prox_monitor_ext. Both are located under the projects\target_apps\ble_examples SDK path.

See also the UM-B-143 Dialog External Processor Interface user manual that provides all necessary information to create GTL based applications. It includes detailed description of the Generic Transport Layer protocol (GTL), the messages exchanged as well as sequence flow diagrams for protocol operations.

Please also refer to the Booting the DA14531 with Codeless Through a STM32 and STM32 SUOTA via DA14531.

2. Over HCI, which is a standard Bluetooth® Low Energy SIG protocol.

The SDK includes a project for HCI applications and can be found in projects\target_apps\hci SDK path.

3.14 Is the DA14531 FCC Certified?

Yes, the DA14531 SoC is certified. See the following links:

1. DA14531 EN 300 328 certification test report (WLCSP)
2. DA14531 EN 300 328 certification test report (QFN)

3.15 How I can Find Schematics Symbols and PCB Footprints in Any Format?

Schematic symbols and PCB footprints for DA14531 are available on the SnapEDA website:

https://www.snapeda.com/search/?q=DA14531&search-type=parts

After sign up, you can download these in just about any format (Altium, Orcad, Eagle, etc.) for free! Both the QFN and CSP variants are supported.

3.16 Can I Use a DA14531 Development Kit (USB or PRO) to Debug my Custom Hardware?

Yes, see **Section 18** in the Getting Started with SDK6 (HTML) .

More details about the USB and PRO DKs can be found in the Development Kit-USB and Development Kit-Pro user manuals.

Please also check the Quick Started Guide with the USB Development Kit and Getting Started with the Pro Development Kit (HTML) documents.

3.17 Does Dialog Offer a DA14531 Module?

Yes, Dialog offers the module **DA14531 SmartBond TINY™ Module** (see the product page).

The DA14531 SmartBond TINY™ Module, based on the world's smallest and lowest power Bluetooth® 5.1 System-on-Chip, brings the DA14531 SoC advantages to an integrated module. It requires a power supply and a printed circuit board to build a Bluetooth® application.

The module is targeting broad market use and will be certified across regions providing significant savings in development cost and time-to-market.

It comes with an integrated antenna and easy to use software making Bluetooth® Low Energy development easier than ever before.

This awesome combination takes mobile connectivity to applications previously out of reach, enabling of the next billion IoT devices, with SmartBond TINY™ at their core.

Please check the TINY module datasheet and the Getting Started guide.

3.18 What is the Expected Power Consumption for the DA14531 Module?

See **Table 4: DC Characteristics** in the TINY module datasheet.

4 RF & Radio

This section describes the RF & Radio-related Frequently Asked Questions.

4.1 What is the Intermediate Frequency (IF) of the Radio?

The **Intermediate Frequency (IF)** of the radio is 1 MHz. See application note Bluetooth® Direct Test Mode that describes the required RF parameters for certification documents and how to setup the RF testing modes for the DA14531 Bluetooth® Low Energy SoC.

4.2 What does the Local Oscillator (LO) Frequency Does the Radio Usage?

The Local Oscillator (LO) frequency is: (**2 x Carrier Frequency**) + 1 MHz.
See application note Bluetooth® Direct Test Mode that describes the required RF parameters for certification documents and how to setup the RF testing modes for the DA14531 Bluetooth® Low Energy SoC.

4.3 What Communication Range can be Achieved with the DA14531?

The **Bluetooth® SIG** has published a range estimator on their website that can be used to calculate the effective, reliable range between Bluetooth® devices: https://www.bluetooth.com/bluetooth-technology/range/

4.4 Is a Pi Filter on the RFIOp Port Required?

Yes, to meet the product specification for ETSI, FCC and ARIB, a Pi Filter is needed to remove the out of band emission. Also, the extra Pi Filter for the RF circuit is needed to avoid Spurious Emissions Reduction. For more information on the antenna filters, see application note Designing Printed Antennas for Bluetooth® Smart.
The DXF files are available too.

4.5 Is 2 Mb/sec Data Throughput Supported?

No. DA14531 is fully compliant with Bluetooth® 5.1 standard, where 2 Mb/s PHY support is optional. So, DA14531 supports up to 1 Mb/s.

4.6 Can I Change the Tx Power Level Using the Production Test Firmware?

The **Production Test** firmware that comes along with the SDK6.0.14 supports a configurable TX level. The project is located under the projects*target_apps**prod_test SDK* path. The TX level can be changed by sending an appropriate command via UART (HCI_SET_TX_POWER_CMD_OPCODE). In addition, the commands can be sent either with two-wire UART over P00/P01 or single-wire UART over P03 or P05.

The available commands and responses are shown in Figure 9 and Figure 10:

Byte Description	Value
HCI Command Packet	0x01
Command Opcode LSB	0x1B
Command Opcode MSB	0xFE
Parameter Length	0x01
data	value / Tx Power (dBm) 0x0C / 2.5 0x0B / 1.5 0x0A / 1 0x09 / 0 0x08 / -1 0x07 / -2 0x06 / -3.5 0x05 / -5 0x04 / -7 0x03 / -10 0x02 / -13.5 0x01 / -19.5

Figure 9: Tx Power Command

Byte Description	Value
HCI Event Packet	0x04
Event Code	0x0E
Parameter Length	0x04
Num_HCI_Command_Packets	0x05
Command_Opcode LSB	0x1B
Command_Opcode MSB	0xFE
Status	0x00

Figure 10: Tx Power Response

For this demonstration, single-wire UART is used over **P05** to set the tx power at -18 dBm.

1. Prepare the DA145xx Pro Development kit for single-wire UART applied on P05 (see Figure 11). Install the **three (3)** Jumpers in Header J1 in the positions indicated by the blue rectangles in Figure 11. See also Figure 27 in Getting Started with the Pro Development Kit (HTML) document.

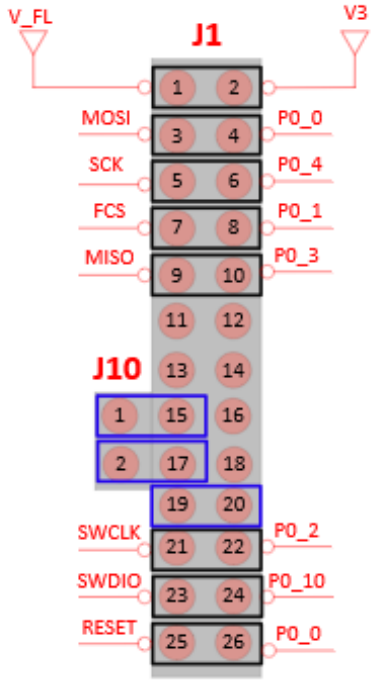

Figure 11: Configuration of DA145xx Pro DK for Single-Wire UART Applied on P05

2. Open the **prod_test** Keil project and navigate to user_periph_setup.h. Configure the GPIOs for UART_TX / UART_RX for single-wire UART as shown in Figure 12.

```
/*****************************************************************************/
/* CONFIG_UART_GPIOS                                                         */
/*    -defined      Uart Port/Pins are defined by external tool              */
/*    -undefined    Uart Port/Pins are defined in the current project        */
/*****************************************************************************/
#undef CONFIG_UART_GPIOS

/*****************************************************************************/
/* UART pin configuration                                                    */
/* Supported Port/Pin Combinations:                                          */
/* Tx: P00, Rx: P01                                                          */
/* Tx: P02, Rx: P03                                                          */
/* Tx/Rx: P03 (1-Wire UART)                                                  */
/* Tx: P04, Rx: P05                                                          */
/* Tx/Rx: P05 (1-Wire UART)                                                  */
/* Tx: P06, Rx: P07                                                          */
/*****************************************************************************/

#if defined(__DA14531__)
    #define UART1_TX_GPIO_PORT  GPIO_PORT_0
    #define UART1_TX_GPIO_PIN   GPIO_PIN_5
    #define UART1_RX_GPIO_PORT  GPIO_PORT_0
    #define UART1_RX_GPIO_PIN   GPIO_PIN_5
#else
    #define UART1_TX_GPIO_PORT  GPIO_PORT_0
    #define UART1_TX_GPIO_PIN   GPIO_PIN_4
    #define UART1_RX_GPIO_PORT  GPIO_PORT_0
    #define UART1_RX_GPIO_PIN   GPIO_PIN_5
#endif
```

Figure 12: Configuration of user_periph_setup.h for Single-Wire UART Applied on P05

3. Build and run the prod_test.
4. To set the tx power at -18 dBm, enter the following characters **01 1B FE 01 01** over UART.

0x01 -> every command starts with a 0x01

0x1B 0xFE -> the op code of the sleep command 0xFE1B

0x01 -> parameter length

0x01 -> set the tx power at -18 dBm

4.7 Is Any Calibration of the RF Parameters Required in Production?

No, this is not required. Dialog Bluetooth® Low Energy devices are calibrated when the IC is manufactured. The accuracy of the 32MHz clock is essential for the quality of the RF. So, based on the application and the selected 32MHz xtal this might need trimming at application level. The Dialog Production Line Tool (PLT) supports XTAL trimming for 16 devices in parallel.

Revision History

Revision	Date	Description
1.2	03-Feb-2022	Updated logo, disclaimer, copyright.
1.1	24-Mar-2021	Added new FAQs and updated the links to support website.
1.0	18-Jan-2020	Initial version.

DA14531 Frequently Asked Questions

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 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.