Debugging Support Using a J-Link Probe
WICED™ Studio 4

Associated Part Family: CYW20706A2

This document provides debugging support via a SEGGER J-Link Pro debug probe. It is intended for software developers of Bluetooth applications that use Cypress® WICED™ Studio and CYW20706A2 System on a Chip (SoC).

Contents

1 Introduction

This document provides help to those debugging Bluetooth applications for CYW20706A2 [2] developed using WICED Studio. To debug a Bluetooth software application, perform the following steps:

1. Prepare the application being debugged by making the following makefile changes:

    a. In the makefile.mk of the debug application, add the DEBUG compilation flag as shown below.

    ```
    ###################################################################
    # Application sources.
    ###################################################################
    APP_SRC = hello_sensor.c
    C_FLAGS += DEBUG
    ```

    b. Define the SWD interface.

    ```
    ###################################################################
    # SWD_CK: SWDCK_ON_UART_TXD, SWDCK_ON_P11
    # SWD_IO: SWDIO_ON_UART_RXD, SWDIO_ON_P31, SWDIO_ON_P15
    ###################################################################
    C_FLAGS += -DSWD_CLK=SWDCK_ON_UART_TXD
    C_FLAGS += -DSWD_IO=SWDIO_ON_UART_RXD
    ```

    ● If P31 is used as the SWDIO signal, the PUART interface will be unavailable since this pin will be repurposed for the SWD interface. Hence, application traces should not be routed to the PUART interface.

2. Program the target board with the application to be debugged.

    ● For initial programming, refer to the WICED Studio Quick Start Guide [1]. Pertinent programming sections in [1]:

    a. Connect the WICED Development Board.

    b. Verify Driver Installation.

    c. Build and Load a Sample Application.

    ● If a target board has been programmed but the nonvolatile storage has been or is presumed to be corrupted, then see Appendix D: Recovering a Corrupt WICED Development Board in [1].

3. Set up the debug environment (see Section 2).

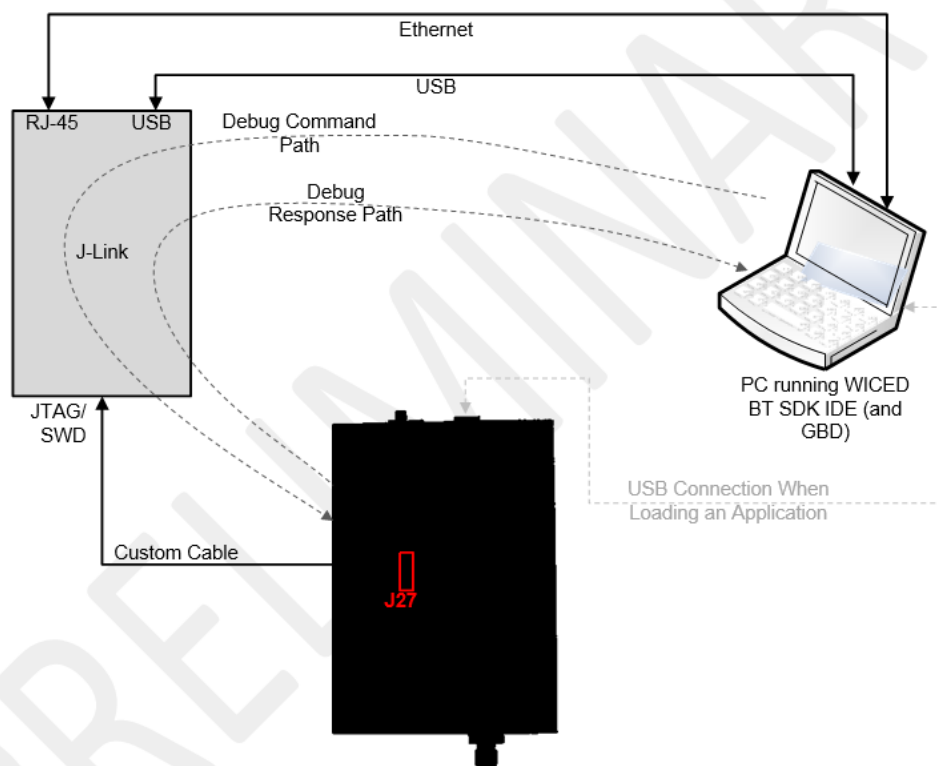4. Set up the target board for debugging (see Section 5).

5. Initiate a debugging session (see Section 7).

2 Setting up for Debugging

To prepare for a debugging session, configure the setup shown in Figure 1.

- For a list of the hardware and software items that pertain to Figure 1, see Section 3.

- For information on the custom cable that connects the J-Link JTAG/SWD port (J52) to the serial wire debug (SWD) port (J27) on the target board, which in this case is the CYW920707V2_EVAL board (see Section 4).

Figure 1. Debug Setup

3   Hardware and Software Requirements

Revision	Description of Change
Reference design board	A hardware reference design board based on a Cypress CYW2070x SoC. Throughout this document, a Cypress CYW920707V2_EVAL board is assumed to be the target board.
SEGGER J-Link Pro debug probe	A debug probe provided by SEGGER Microcontroller GmbH & Co. KG. See www.segger.com for debug probe information.
	The J-Link Pro contains a GNU Project Debugger (GDB) server that connects to GDB running remotely (typically within the Cypress WICED Studio IDE). The GDB server receives GDB commands, communicates those commands serially to the debug target via the J-Link Pro's 20-pin JTAG connector, and forwards target responses back to GDB.
Debug target to J-Link cabling	A cable connecting the debug target (J27 SWD port of the CYW920707V2_EVAL) to the J-Link 20-pin JTAG/SWD connector.
	(See Debug Target to J-Link Serial Wire Debug Connections).
PC	A computer that hosts the following software items:
	• The Cypress WICED Studio IDE
	• J-Link Pro GDB server interface GUI and driver software. On a Windows system, the Windows J-Link Software Package (`JLinkGDBServer.exe`) provides the GDB server interface GUI.
	• J-Link Commander

Revision	Description of Change
PC to J-Link Pro Ethernet cable (optional)	Although optional, this interface is required for those wanting to connect the J-Link Pro as a network node. Use of this option allows J-Link configuration via a web server on the J-Link Pro. This interface is required if the USB interface is not used.
PC to J-Link Pro USB cable (optional)	This interface is available to those not requiring network access to their J-Link Pro, and is required if the Ethernet interface is not used.

● In this document, all references to GDB are references to GDB executing within the Cypress WICED Studio IDE. Although it is possible to run GDB outside of WICED Studio, doing so will preclude using some of the automated debug capabilities in WICED Studio.

4 Debug Target to J-Link Serial Wire Debug Connections

Figure 2 shows the connections between J27 of the CYW920707V2_EVAL board and the J-Link JTAG/SWD connector as well as other required connections at each of those connectors.

Figure 2. J-Link to Debug Target Cable

To J27 on the
CYW920707V2_EVAL board

J-Link
(JTAG/SWD header)
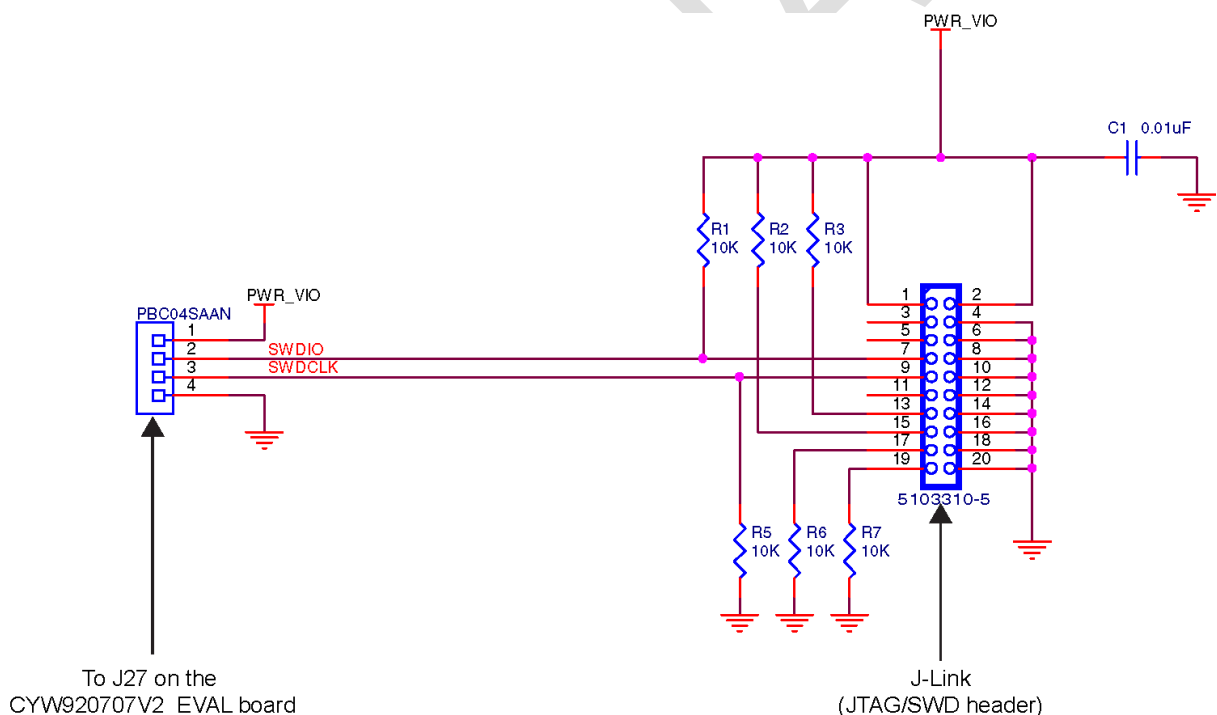
● To be able to reset the CYW920707V2_EVAL board directly from the debugger, connect pin 15 of the JTAG/SWD connector (Figure 2) to J22 pin 1 of the CYW920707V2_EVAL board.

5 Setting Up the Target Board for Debugging

If the target board has not been programmed with the target application to be debugged, then first program the target board with the target application (see [1]); otherwise, perform the following steps:

1. Ensure that the PC to target board USB connection is disconnected.

2. Attach a 3.2V power supply to the board by connecting TP1 to 3.2V and TP2 to ground.

3. Remove all jumpers on J26 to ensure the FTDI chip does not interfere with the SWD signals.

Figure 3. Target Board Setup for Debugging
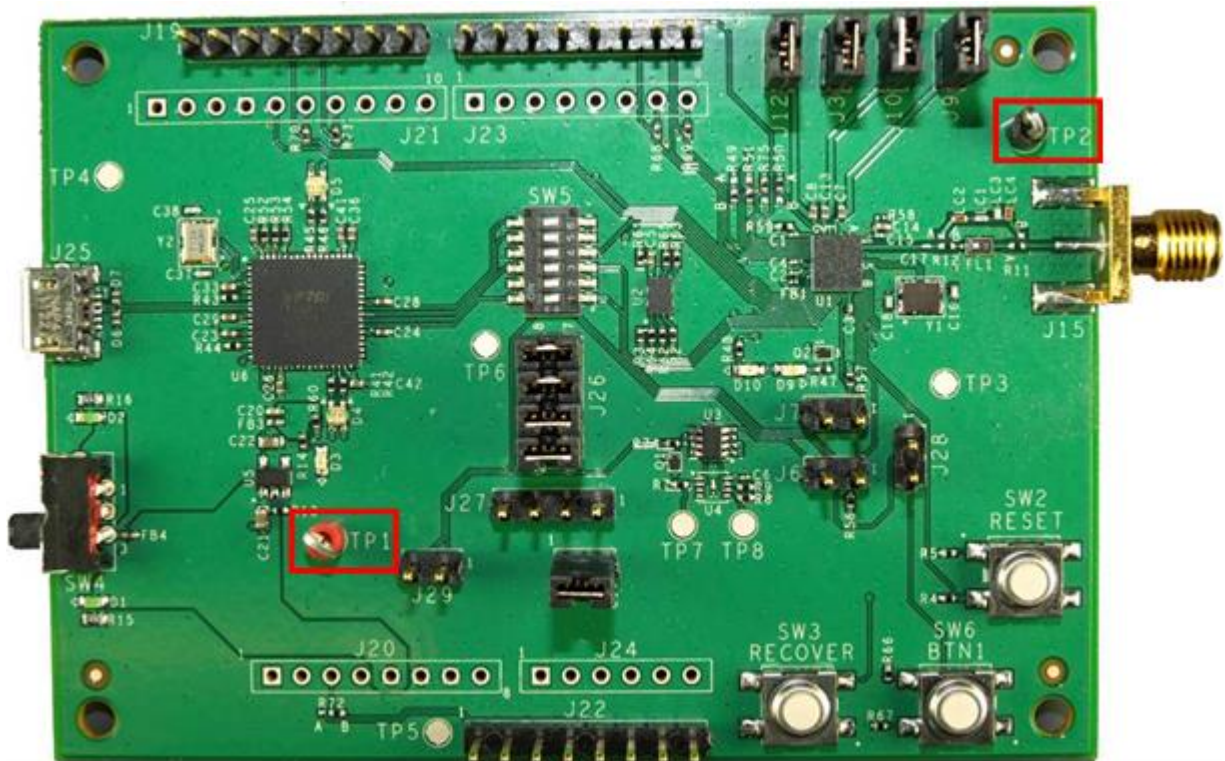
6 Debug Configuration

Before launching the Debug perspective, ensure the Debug Configuration is as follows:
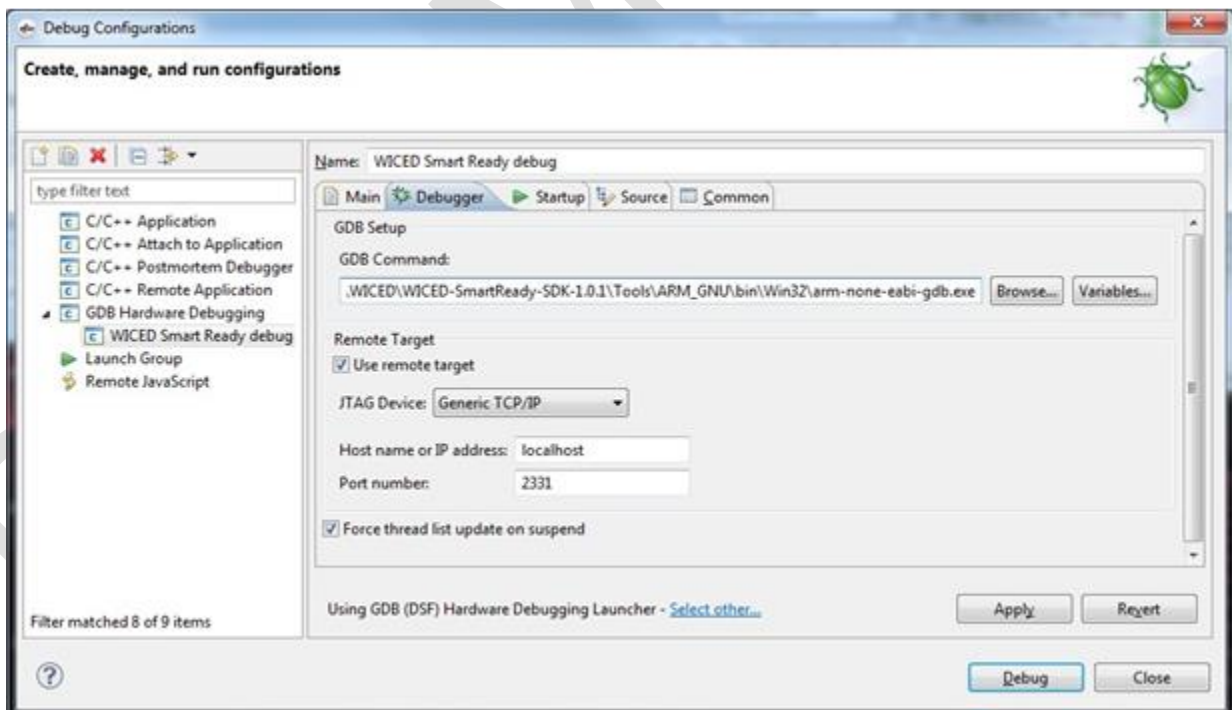
1. Open the Debug Configurations window.
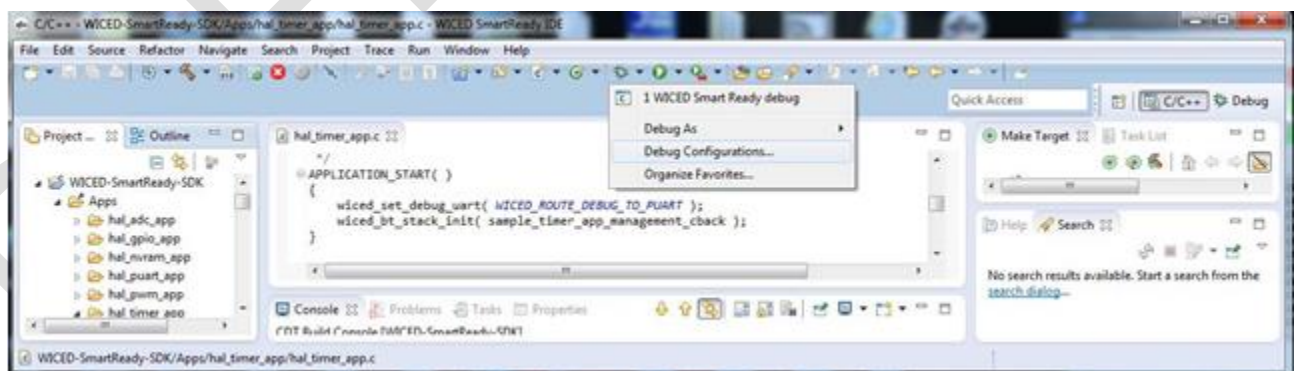
2. Select the **WICED Studio debug** selection under GDB Hardware Debugging in the left pane of the configurations window.

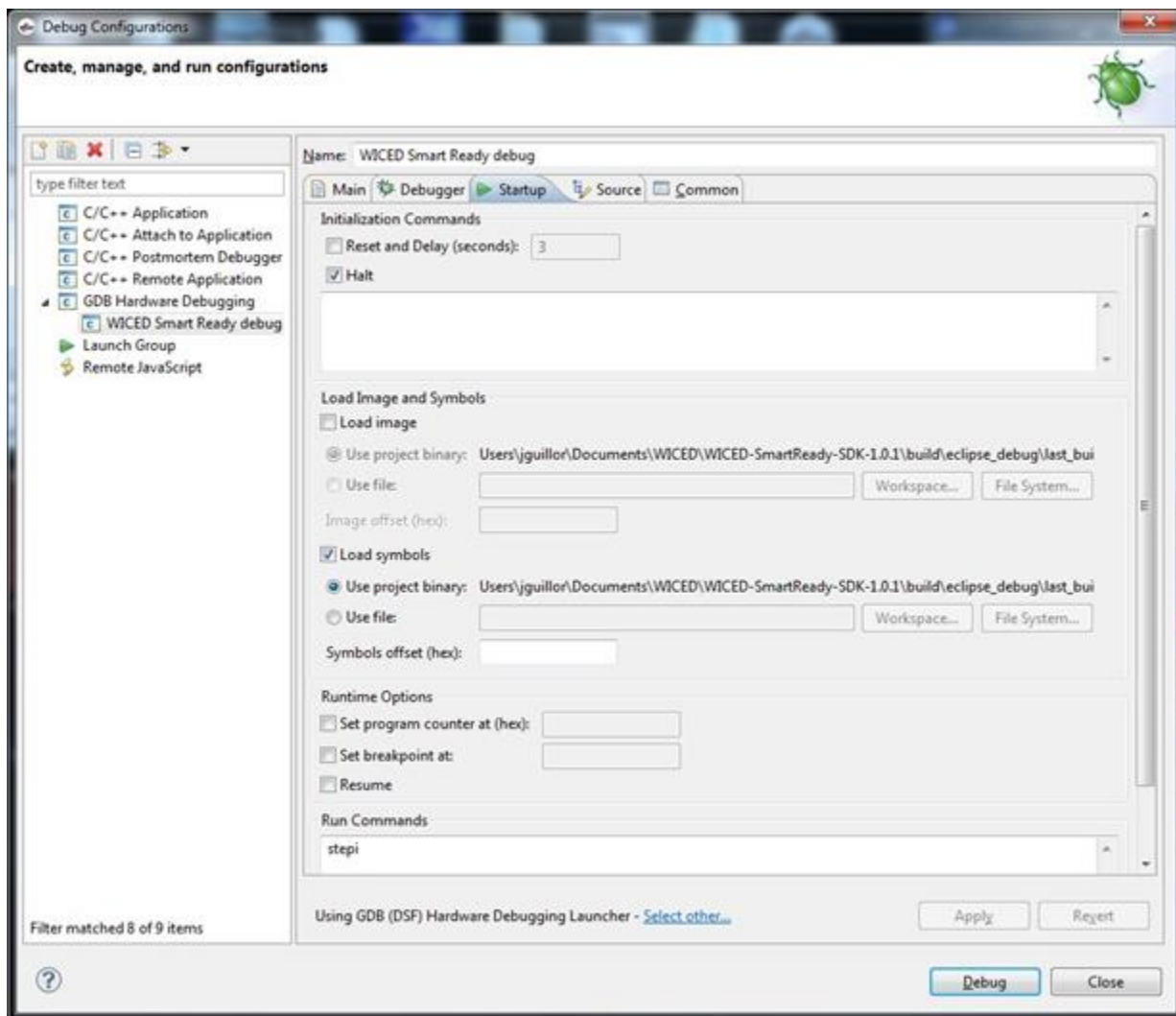3. Under the Main tab, ensure the C/C++ Application is pointing to your project's `last_build.elf` file.

4. In the Project field, make sure the appropriate project is selected.
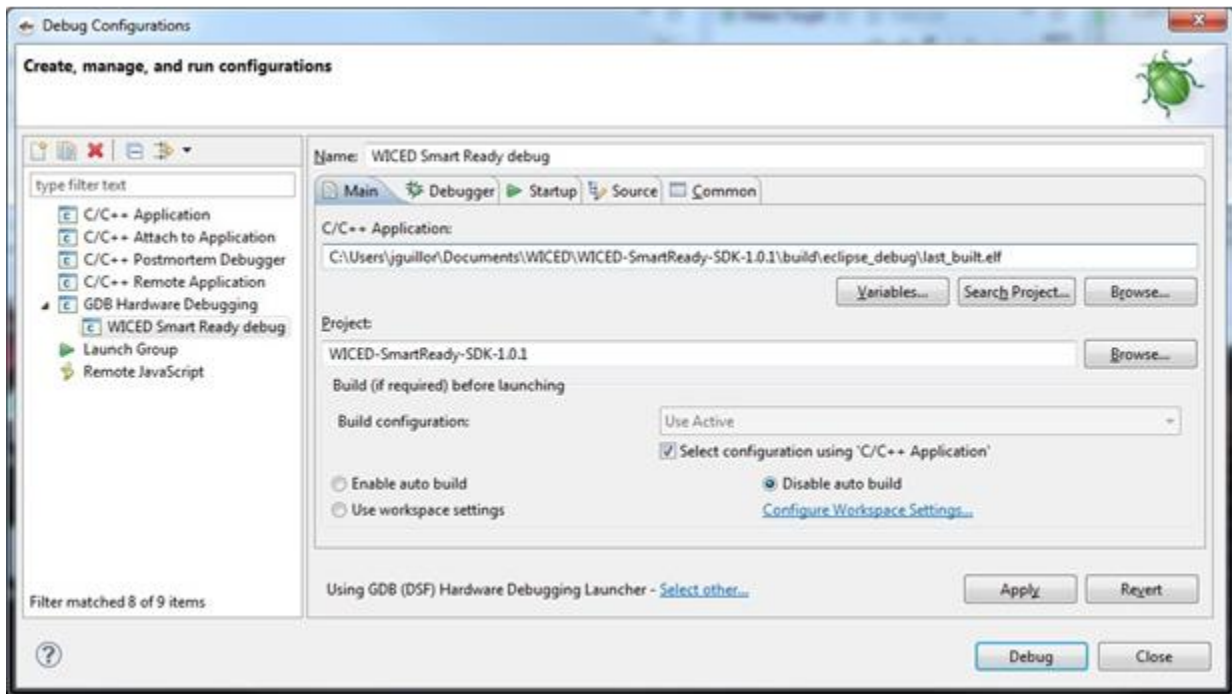
5. Under the Debugger tab, set the configuration settings as shown below.

6. Lastly, under the Startup tab, match the following configuration settings.

7. Click **Apply** and close the Debug Configurations window.

7 Running the Debugger

To debug a software application that previously has been loaded to a target device, perform the following steps:

1. Verify the debug setup (see Figure 1 on page 3).

2. Set up the target board if it has not yet been set up (see Section 2).

   - When setting up the target board for debugging, connect the USB cable between the PC and the J-Link Pro before connecting the J-Link Pro JTAG/SWD port to the target board.

3. Reset the target board by pushing and releasing the onboard reset button (SW2) or from within the debugger (see the note following Figure 2 on page 10 in order to reset via the debugger).

4. Start the GDB server, which in Windows can be done using the Windows J-Link Software Package (JLinkGDBServer.exe).

5. Set up the GDB server. In the Cypress evaluation setup, the following configurations were used (see Figure 4 on page 8 for a GBD configuration screenshot example):

- Connection to J-Link: **USB**

- Target device **Cortex-M3**

- Target interface **SWD**

- Speed **1000 kHz**

Figure 4. GDB Server Configuration

6. Click **OK** to close the GDB server configuration window. After closing the configuration window, verify that the three **J-Link** and **CPU** indicators are green.
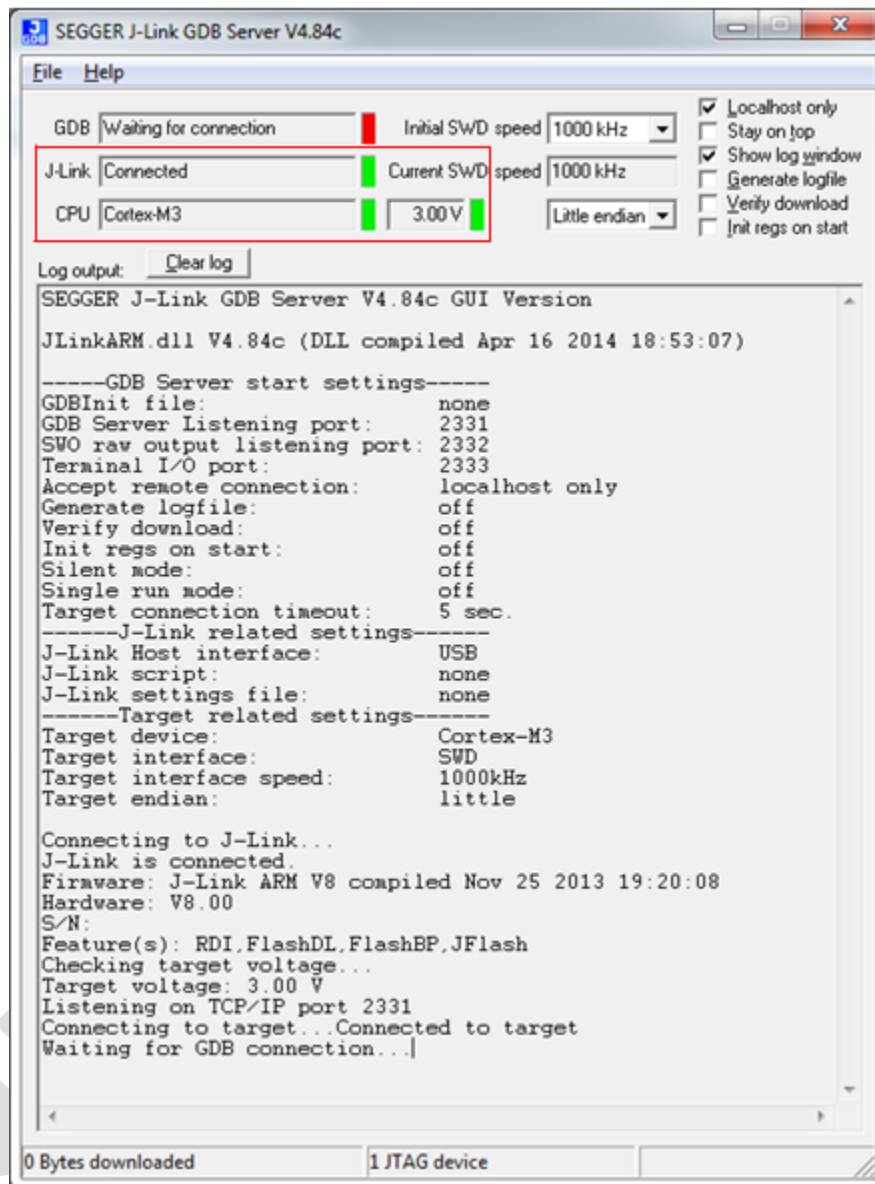
6. In the WICED Studio IDE, click the **Debug** icon and select **WICED debug**.
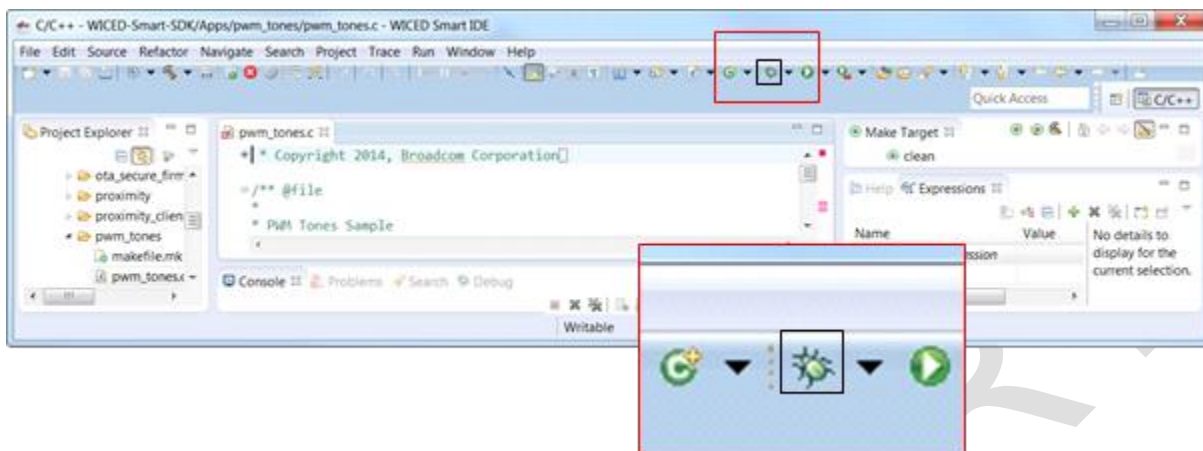
7. Open the Expressions window (Window > Show View > Other > Expressions) and type an expression.

For example:

    a. Type `spar_debug_continue`

    b. Change the default value 0x0 to a nonzero value.

    c. Click Resume (click **Run > Resume** or press **F8**)
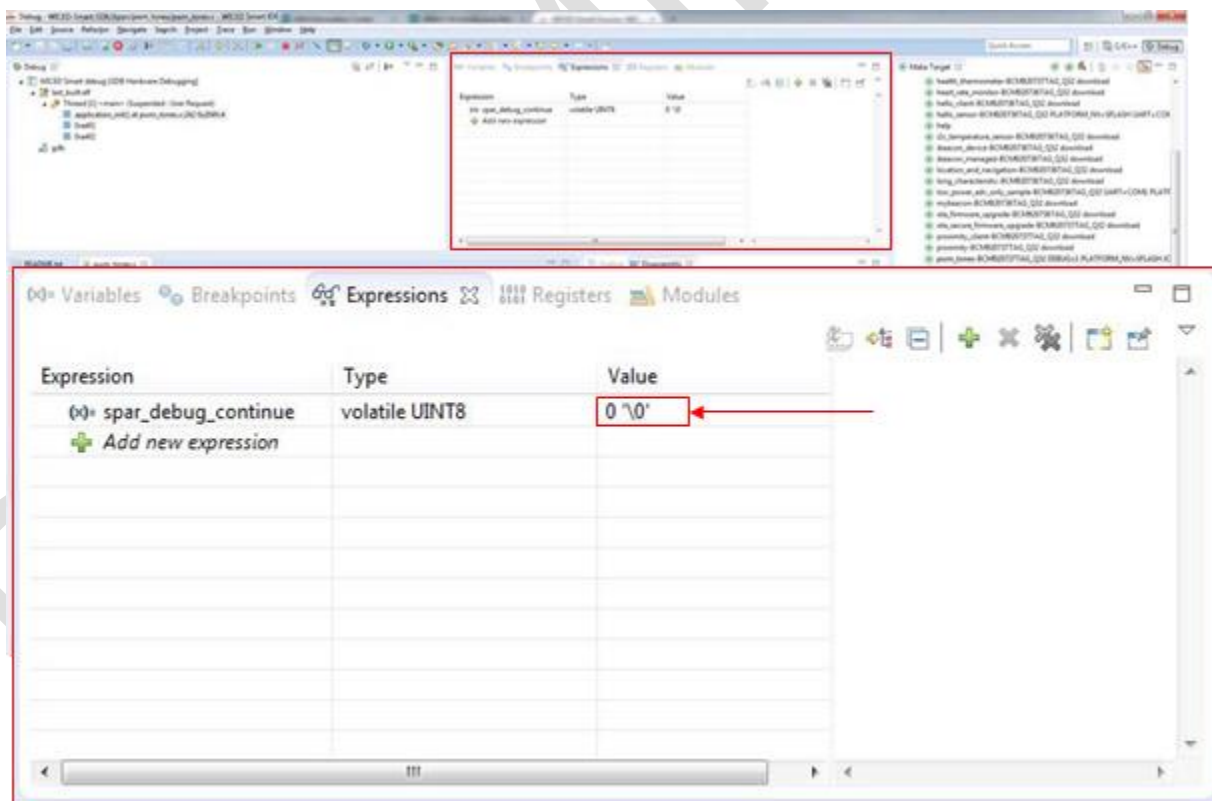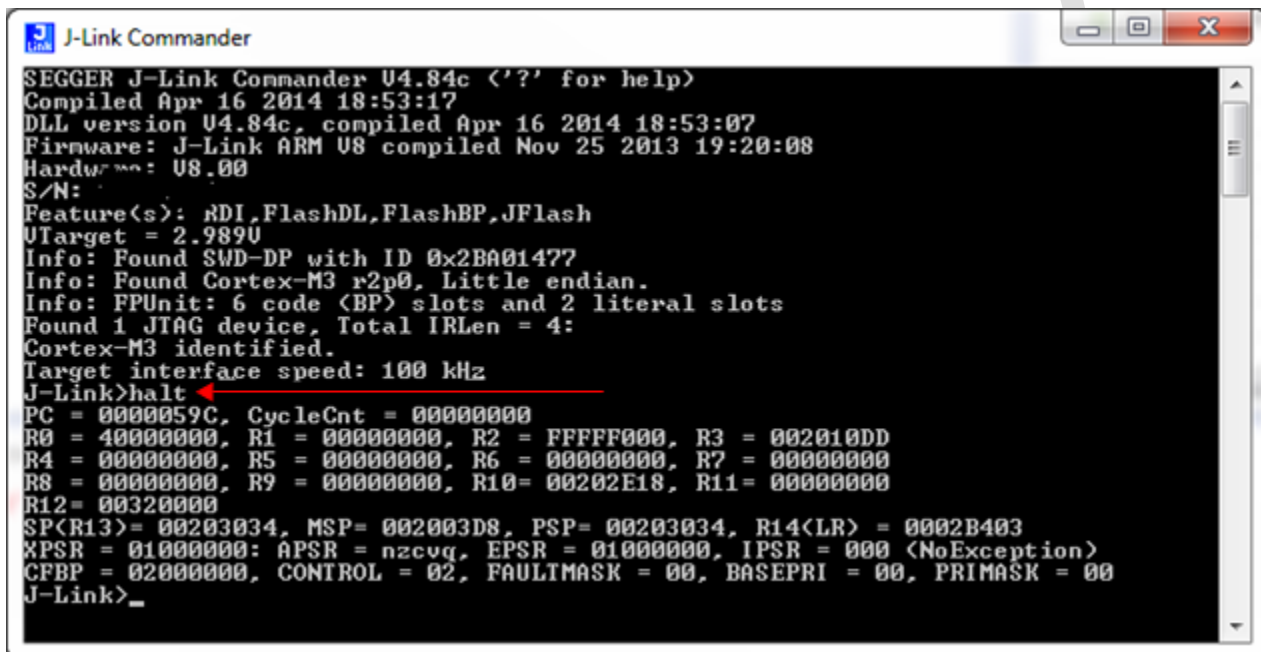
8. Set breakpoints where the CPU is to be halted by double-clicking in the left margin of the code window.

9. Step into, over, and out of instructions using the IDE buttons.

Upon resuming, the Suspend button in the IDE may not halt the CPU. If you experience this behavior, use the J-Link Commander tool to halt the processor (see Figure 5). After halting succeeds, the IDE will reveal that the CPU has been halted.

Figure 5. J-Link Commander Example for Halting the CPU

References

[1] WICED Studio Quick Start Guide for BT CYW20706 (ANCYWICED004)

[2] Debugging Support Using A J-Link Probe (AN16535)

[3] J-Link (www.segger.com)

Document History

Document Title: AN216762 - Debugging Support Using a J-Link Probe

Document Number: 002-16762

Revision	ECN	Submission Date	Description of Change
**	TBD	10/17/2016	Initial Revision - preliminary

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Cypress Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community (CDC)

Forums | Projects | Videos | Blogs | Training | Components

WICED IoT

Uniting CDC and WICED Solutions

Technical Support

cypress.com/support

PSoC is a registered trademark and WICED and PSoC Creator are trademarks of Cypress Semiconductor Corporation. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

	Cypress Semiconductor	Phone	: 408-943-2600
	198 Champion Court	Fax	: 408-943-4730
	San Jose, CA 95134-1709	Website	: www.cypress.com