

WCT1011A/WCT1013A Automotive MP-A9 V4.0 Wireless Charging Application

1. Key Features

The Automotive MP-A9_Rev1.0 (MP-A9_Rev1_SCH-29323_B2, MP-A9_Rev1_LAY-29323_B2) wireless charging TX demo is used to transfer power wirelessly to a charged device. A charged device can be any electronic device equipped with a dedicated Qi wireless charging receiver.

The main parameters of the wireless charging transmitter (WCT) are as follows:

- The input voltage ranges from 9 V DC to 16 V DC (automotive bat range).
- The input voltage can drop down to 6 V DC level during the start-stop function.
- The nominal delivered power to the receiver is 15 W (at the output of the receiver) and can be compatible with 5W receiver.
- Designed to meet the Qi 1.2.4 specification.
- Operation frequency: 125 kHz for Qi devices.

Contents

1. Key Features	1
2. Hardware Setup	2
3. Application Operation	5
4. Hardware Description	6
5. Application Monitoring and Control Through FreeMASTER	14
6. Application Monitoring Through Console	19
7. Programming New Software and Calibration	21
8. Software Description	41
9. System Bring Up	47
10. Revision History	51

2. Hardware Setup

2.1. Pack content

1. WCT Automotive MP-A9 (WCT-15WTXAUTO) demo board
2. Power supply connector
3. Power supply 12V/3A



Figure 1. Hardware pack contents

2.2. Board description

The WCT board is connected to the system by the main power connector. It comprises the automotive battery connection (red wire = +12 V line, black wire = GND line), the CAN connection (yellow wires), and the IGNITION (blue wire).

The connectors on the bottom edge of the board provides JTAG connection for programming and debugging, 2xSCI for the FreeMASTER tool connection for the debug option (only one SCI is available for digital buck-boost platform), and console connection, and the temperature connector and backup touch sense connector are placed on the edge of the board. The thermal resistor circuits can be used to develop the temperature sensing and protection.

The circuitry on the board is covered by the metal shield to lower the EMI and provides a fixed position for the coils. [Figure 2](#) shows the device.

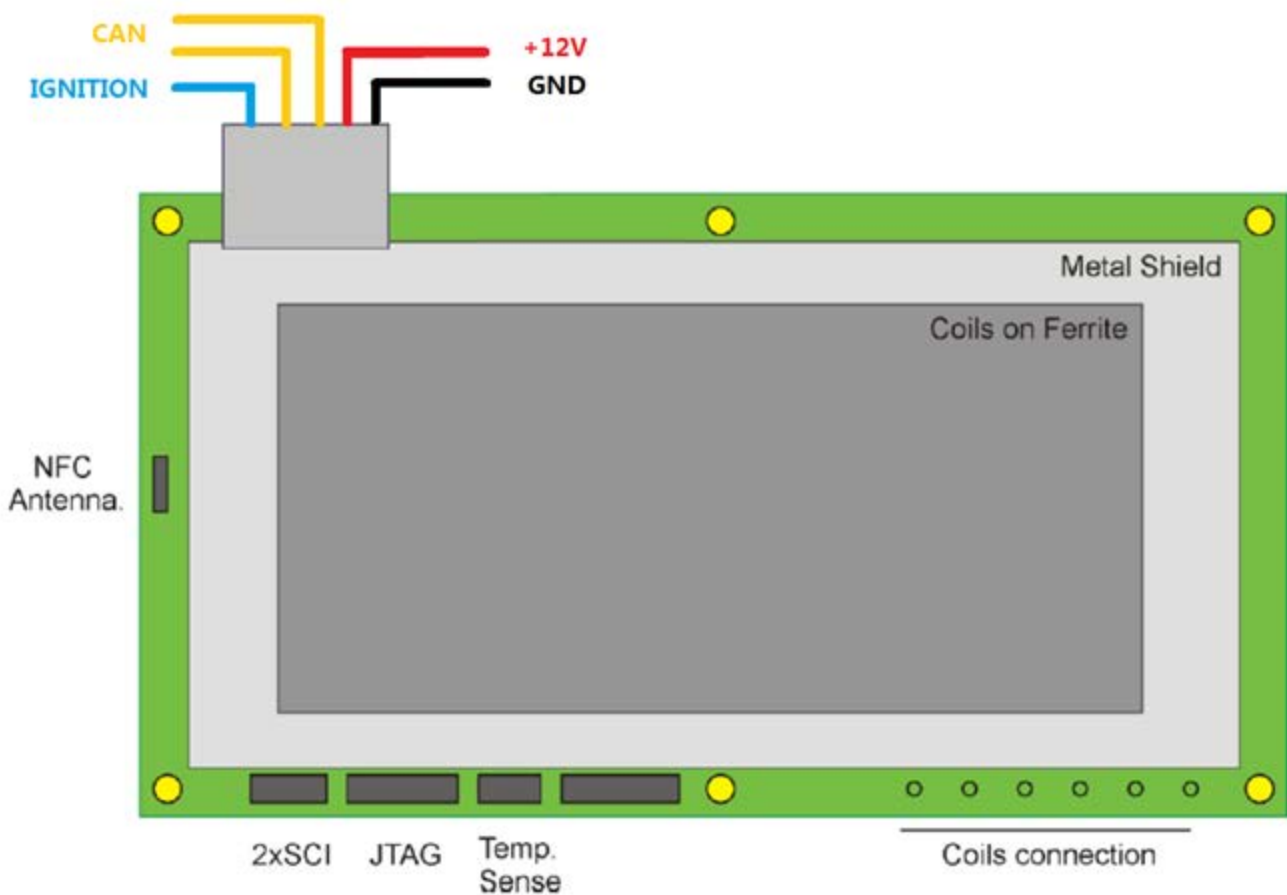


Figure 2. Device

2.3. Powering on a board

To power on a board, perform the following steps:

1. Plug power supply 12 V to the socket.

2. Plug the power supply connector into the board.
3. Connect power supply 12 V and power supply connector.

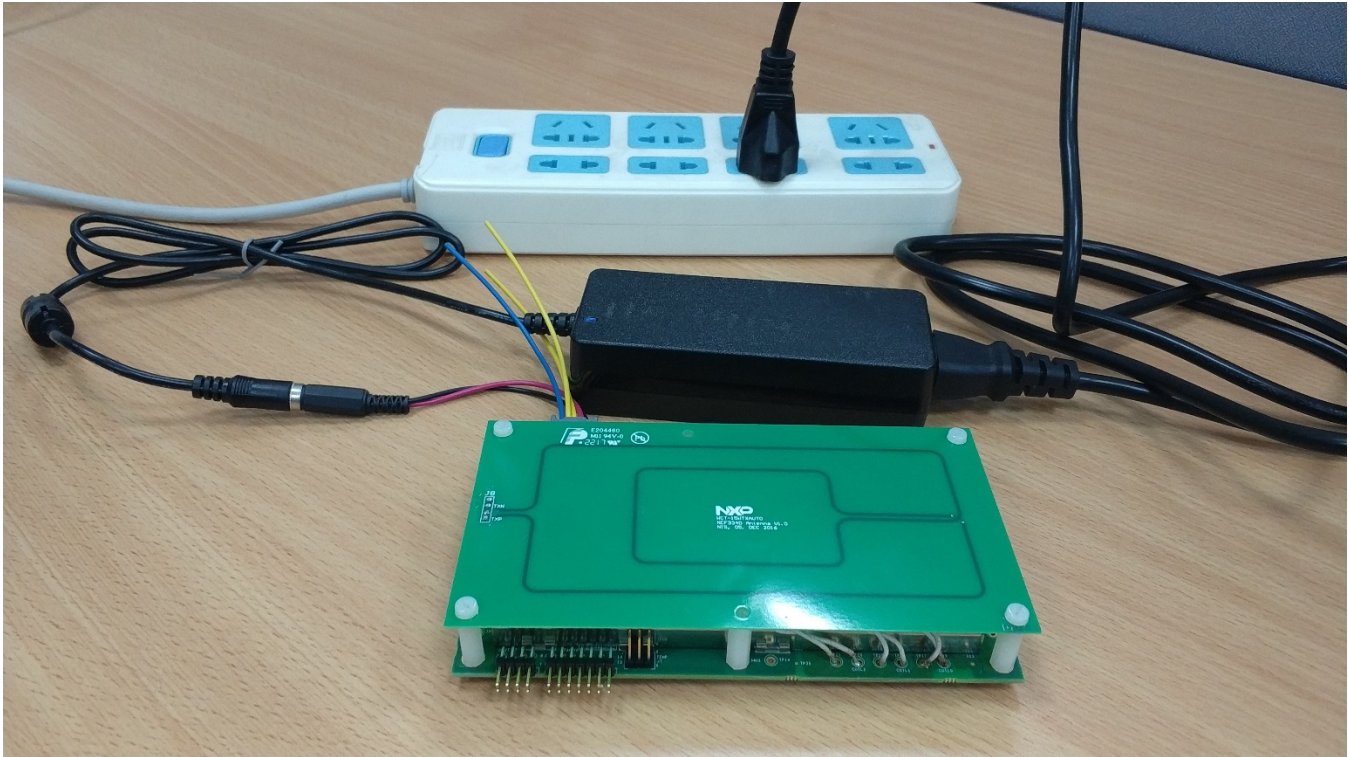


Figure 3. Power supply components

2.4. Hardware setup for FreeMASTER and Console communication

To set up the hardware for FreeMASTER and Console communication, perform the following steps:

1. Find two UART-to-USB adaptor boards, and successfully install this UART-to-USB device driver on the computer. The virtual serial port on the computer should work well.
2. Plug the USB-UART converting board to SCI connector J2 according to the SCH signal pin position. The two channels of UART are for different purposes: FreeMASTER and Console.

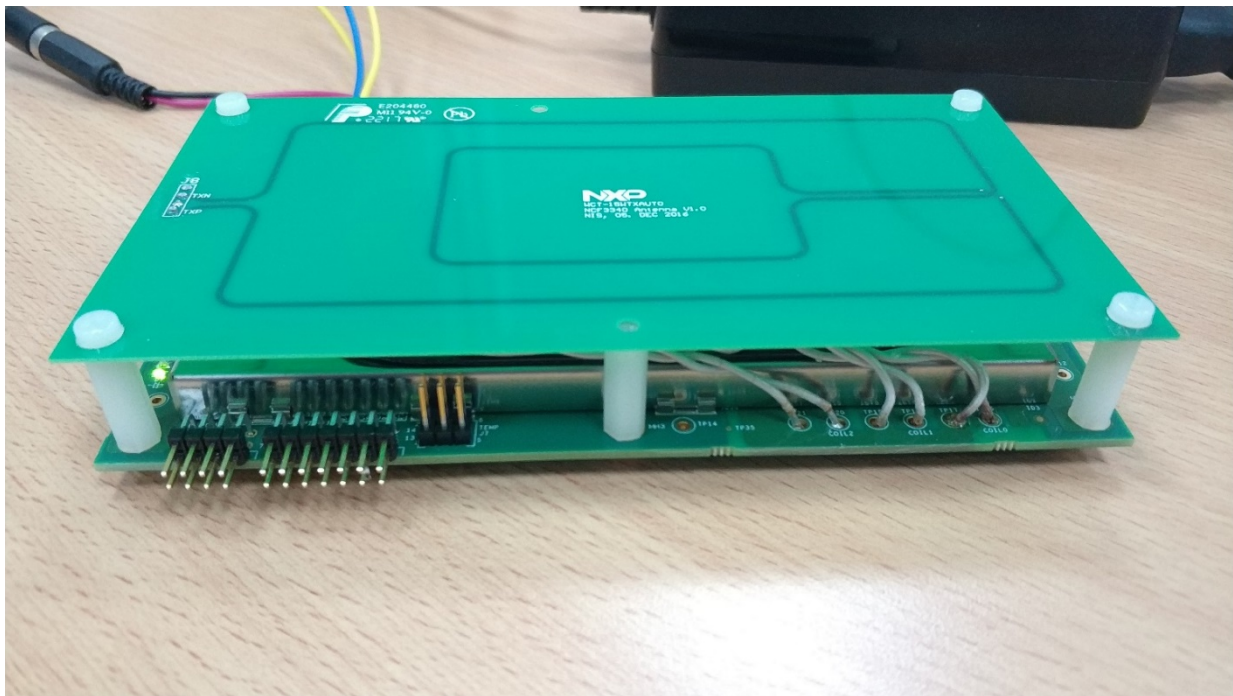
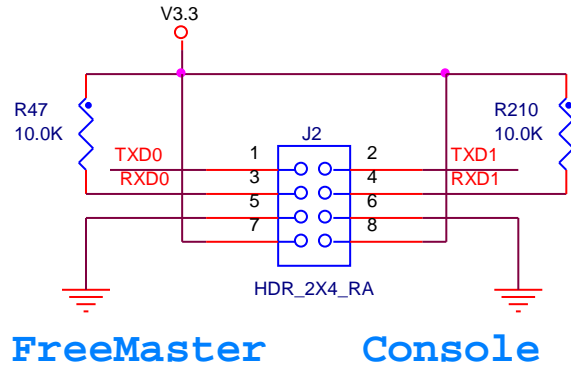


Figure 4. SCIs and JTAG connectors

3. Application Operation

Connect the demo to the supply voltage +12 V DC. The WCT starts to send periodically the power ping to check whether the compatible device wireless charging receiver (WCR) is placed on the charging surface.

When the Qi-compliant device is placed on the top of the TX coils area, the WCT starts the charging process. If there is no correct Qi answer from the WCR side, the TX does not start the Qi charging process.

If the WCR answers properly, the power transfer starts. The actual level of the transferred power is controlled by the WCT in accordance with WCR requirements. The receiver sends messages to the WCT through ASK on the coil resonance power signal, and the transmitter sends the information to the receiver by FSK per the Qi specification. The power transfer is terminated if the receiver is removed from the WCT magnetic field.

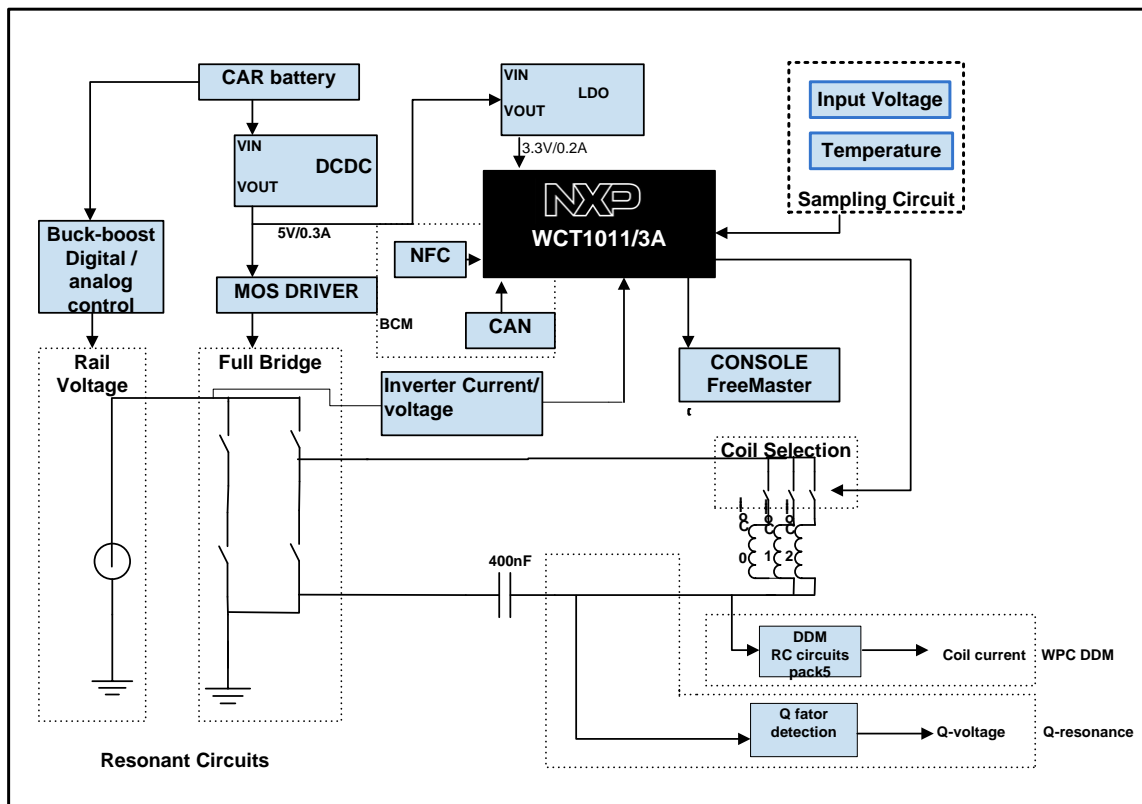
The system supports all Qi WCR devices: Qi_Ver-1.0 compliance and Qi_Ver-1.1 compliance, and the Qi EPP 15 W receiver. The system supports all the FOD features for different receivers. For the low-power 5 W receiver, the power loss FOD is supported. For the EPP 15 W receiver, both Q-value method and power loss FOD method are supported.

4. Hardware Description

Figure 5 shows the block diagram of the automotive wireless charger MP-A9.

Go to the NXP website to obtain the latest Hardware Design files.

The whole design consists of several blocks, which are described in the following sections.



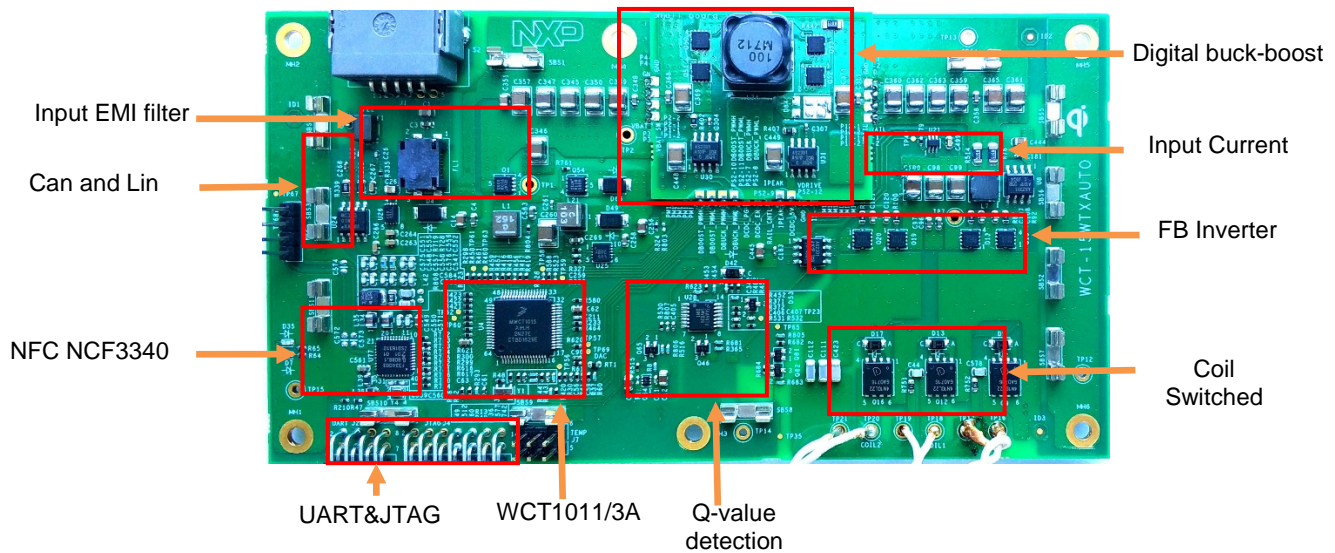


Figure 5. Block diagram of the automotive wireless charger MP-A9

4.1. Input EMI filter

The input connector J1 provides the whole connection to the car wiring. It connects the battery voltage to the WCT and CAN communication interface.

The input filter consists of the Common Mode Filter FL1 and the filter capacitors C1, C3, C4, C14, and L1.

The main battery voltage switch is equipped with MOSFET Q1. This stage is controlled by the main controller WCT1011A/WCT1013A and the IGNITION signal. The hardware overvoltage protection (more than 20 V DC) is also implemented by D1 and Q2 to this switch.

4.2. System voltage DCDC and LDO

The 12 V Car Battery input is connected to a buck converter U25 (MPQ4558). Its output is 5 V and supplies LDO U26 (MPQ8904), MOSFET Driver, and CAN Transceiver. The 3.3 V output of LDO is mainly for WCT1011A/WCT1013A and other 3.3 V powered components.

Generally, the DCDC works at the light-load conditions. It is preferable to select a high efficiency in the light-load condition.

4.3. Rail voltage generated by digital buck-boost or analog buck-boost chip

The Qi specification for the MP-A9 topology requires the DC voltage control to control the power transferred to the receiver. The buck-boost converter is selected to obtain the regulated DC voltage in the range from 1 V DC to 24 V DC for the full-bridge inverter power supply. The buck-boost can be digitally controlled by the WCT chip or the individual analog buck-boost converter and the WCT chip just controls the output voltage feedback.

Digital buck-boost module includes the drivers, the full-bridge converter, and the output voltage feedback. The DCDC converter's control loop is implemented by the firmware, and the control parameters can be optimized with different main circuit parameters, such as the inductor and output capacitor.

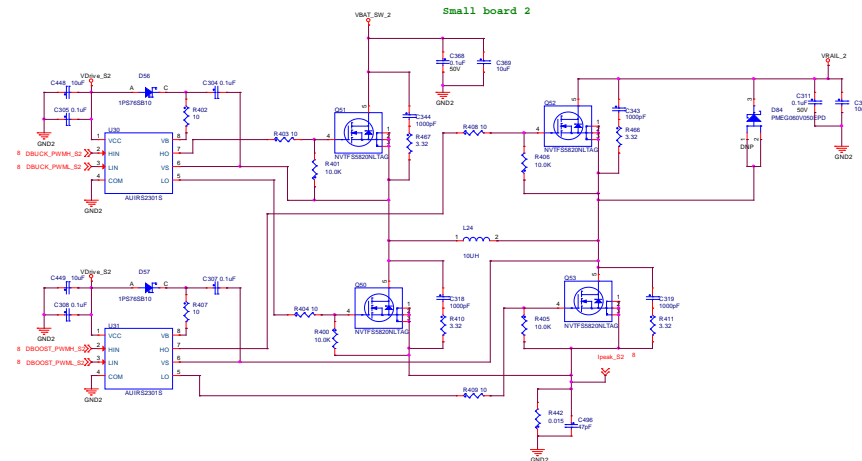


Figure 6. Digital buck-boost main circuits

For the analog buck-boost module, the LTC3789 is selected to generate the rail voltage. The WCT chip controls the rail voltage by one analog signal. This analog signal affects the analog buck-boost converter feedback, and then the system can get the rail voltage as the system expects.

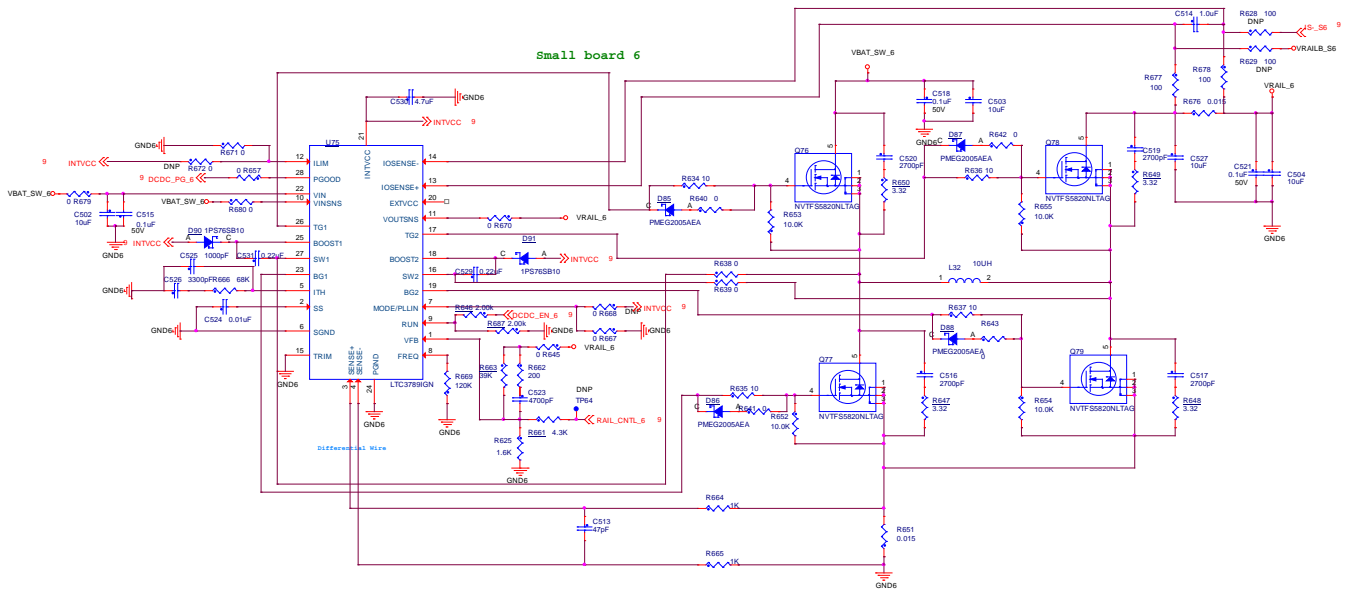


Figure 7. Analog buck-boost main circuits

The digital buck-boost is recommended for use on this wireless charging solution due to its lower cost, simpler hardware circuits, and easier to be controlled.

4.4. Full-bridge and resonant circuits

The full-bridge power stage consists of two MOSFET Drivers, U8 and U9, as well as four power MOSFETs, Q13, Q15, Q19, and Q20. The MOSFET Drivers are powered by the stable voltage level 5 V DC that decreases the power losses in the drivers and MOSFETs. The full-bridge power stage converts the variable DC voltage VRAIL to the square wave 50 % duty-cycle voltage with 125 kHz frequency. The range of the used frequency (120 kHz to 130 kHz) is defined in the Qi specification for the MP-A9 topology.

The resonant circuits consist of C111, C112, C423, C580, and coils, all of which are fixed values defined in the Qi specification for the MP-A9 topology. The snubber RC pairs connected in parallel to power MOSFETs are used to lower the high frequency EMI products. The Vrail discharge circuit Q46, R376, is switched ON while the system is terminated.

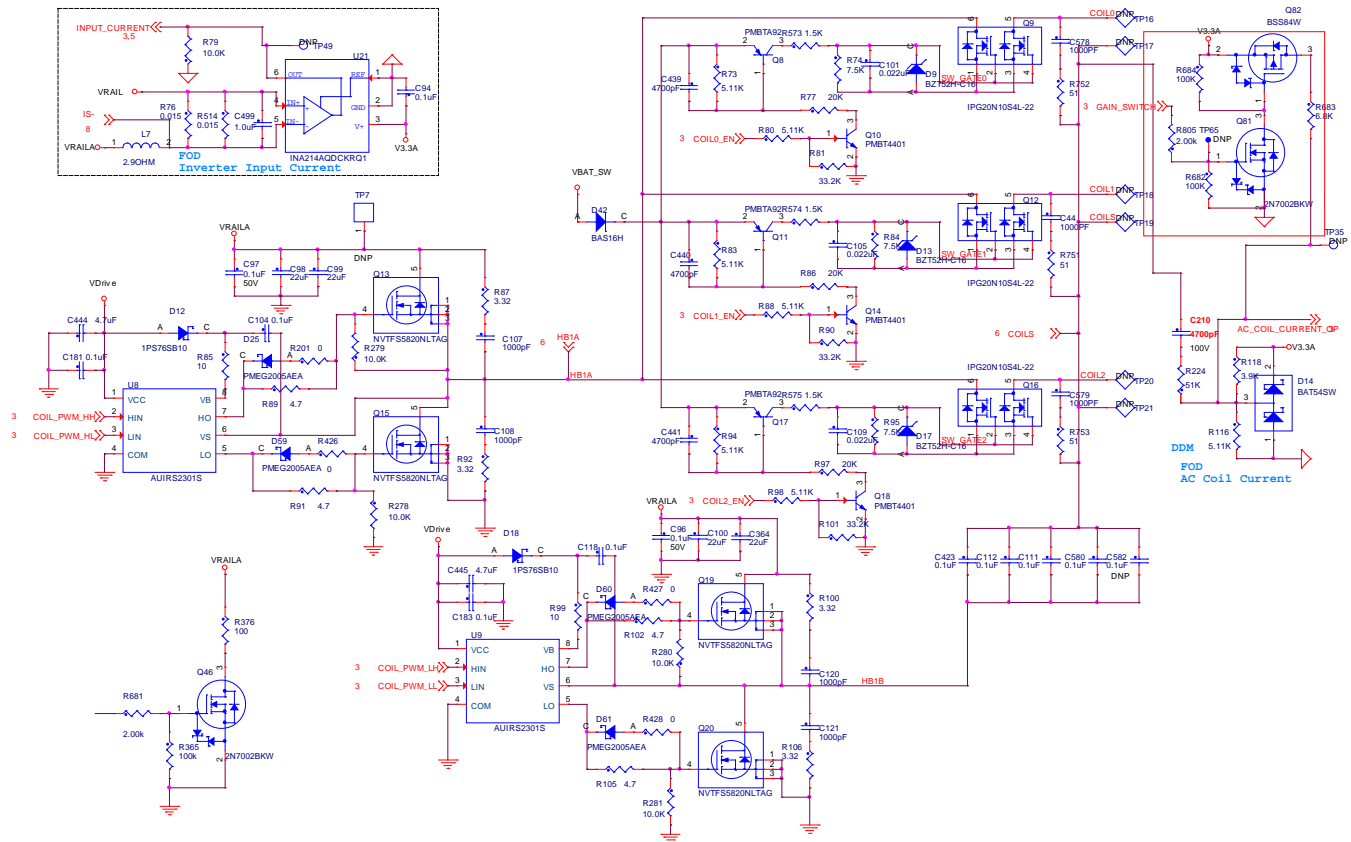


Figure 8. Full bridge circuits and coil selection circuits

4.5. Communication

There is bi-way communication between the medium power transceiver and receiver. Communication from RX to TX: The receiver measures the received power and sends back to transmitter the information about the received power level. This message is amplitude modulated (AM) on the coil current and sensed by TX.

The RC circuits (C210, R116, R118, R224), known as DDM, sample the signals from the coil, compress the signal amplitude, and feed to ADC B-channel of WCT1011A/WCT1013A. The information about the current amplitude and modulated data are processed by the embedded software routine.

Communication from TX to RX: TX shall negotiate with RX in the negotiation phase if requested by RX. TX uses FSK Modulation to communicate with RX, and the communication frequency is about 512 times operating frequency.

4.6. FOD based on power loss

The power loss P_{LOSS} , which is defined as the difference between the Transmitted Power P_{PT} and the Received Power P_{PR} , i.e. $P_{LOSS} = P_{PT} - P_{PR}$, provides the power absorption in Foreign Objects, as shown in Figure 9.

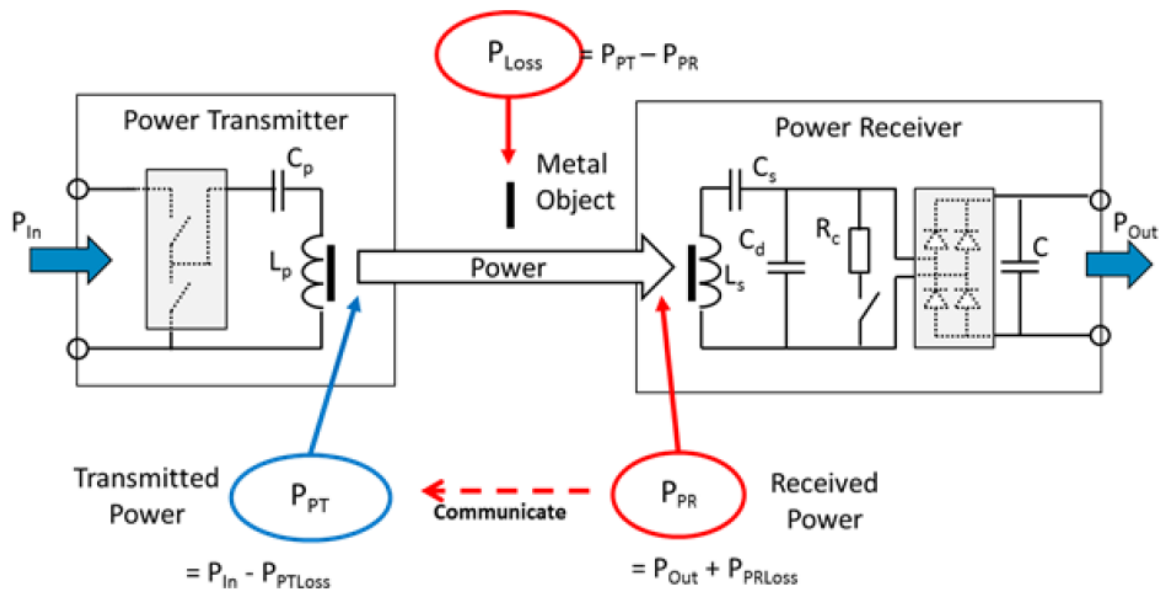


Figure 9. Power loss illustrated

When the FO is implemented in the power transfer, the power loss increases accordingly, and the FO can be detected based on the power loss method.

Power loss FOD method is divided into two types: FOD for baseline power profile (TX and RX can transfer no more than 5 W of power) and extensions power profile (TX and RX can transfer power above 5 W).

4.6.1. Power loss FOD baseline

The equation for power loss FOD baseline is $P_{LOSS} = P_{PT} - P_{PR}$.

The Transmitted Power P_{PT} represents the amount of power that leaves the TX due to the magnetic field of the TX, and $P_{PT} = P_{in} - P_{PTLoss}$, where P_{in} represents the input power of the TX and P_{PTLoss} is the

power dissipated inside the TX. P_{in} could be measured by sampling input voltage and input current, and P_{PTloss} could be estimated through the coil current.

The Received Power P_{PR} represents the amount of power that is dissipated within the RX due to the magnetic field of the TX, and $P_{PR} = P_{Out} + P_{PRloss}$. The power P_{Out} is provided at the RX's output and P_{PRloss} is the power lost inside the RX.

When NXP WCT-15WTXAUTO charges the baseline profile RX, the power loss baseline is applied. The TX continuously monitors P_{LOSS} , and if it exceeds the threshold several times, the TX terminates power transfer.

4.6.2. Power loss FOD extensions

Typically, a RX estimates the power loss inside itself to determine its Received Power. Similarly, the TX estimates the power loss inside itself to determine its Transmitted Power. A systematic bias in these estimates results in a difference between the Transmitted Power and the Received Power, even if there is no Foreign Object present on the Interface Surface. To increase the effectiveness of the power loss method, the TX can remove the bias in the calculated power loss by calibration. For this purpose, the TX and Power RX execute the calibration phase before the power transfer phase starts. The TX needs to verify that there is no FO present on its interface surface before the calibration phase and FOD based on Q factor could work.

As the bias in the estimates can be dependent on the power level, the TX and RX determine their Transmitted Power and Received Power at two load conditions — a “light” load and a “connected” load. The “light” load is close to the minimum expected output power, and the “connected” load is close to the maximum expected output power. Based on the two load conditions, the Power Transmitter can calibrate its Transmitted Power using linear interpolation. Alternatively, the Power Transmitter can calibrate the reported Received Power.

Take calibrated Transmitted Power as an example:

$$P_{PT}^{cal} = a * P_{PT} + b$$

$$a = \frac{P_{PR}^{(connected)} - P_{PR}^{(light)}}{P_{PT}^{(connected)} - P_{PT}^{(light)}}$$

$$b = \frac{P_{PT}^{(connected)} * P_{PR}^{(light)} - P_{PR}^{(connected)} * P_{PT}^{(light)}}{P_{PT}^{(connected)} - P_{PT}^{(light)}}$$

Therefore, the TX uses the calibrated Transmitted Power to determine the power loss as follows:

$$P_{LOSS} = P_{PT}^{cal} - P_{PR}$$

When NXP WCT-15WTXAUTO charges an RX baseline, only the power loss FOD baseline works. If a RX extension is placed on NXP WCT-15WTXAUTO, the Q factor would be measured at first to detect if there is a FO presents. If yes, the TX would stop charging; otherwise, the TX can proceed to calibration phase and power transfer phase, and power loss FOD extension works to detect if a FO is inserted during power transfer phase.

For details of FOD, see the *WCT1011A /WCT1013A Automotive MP-A9 Run-Time Debug User's Guide* (WCT101XARTDUG).

4.7. FOD based on Q factor change

A change in the environment of the TX coil typically causes its inductance to decrease or its equivalent series resistance to increase. Both effects lead to a decrease of the TX coil's Q factor. The RX sends a packet including the reference Q factor for TX to compare and determine if FO exists, as shown in [Figure 10](#).

The reference Q factor is defined as the Q factor of Test Power Transmitter #MP1's Primary Coil at an operating frequency of 100 kHz with RX positioned on the interface surface and no FO nearby. Due to the differences between its design and that of Test Power Transmitter #MP1, the difference between the frequency it uses to determine its Q factor and 100 kHz. The TX needs to convert the Q factor it measured to that of Test Power Transmitter #MP1. NXP provides the conversion method and needs to get the parameters on board at first. The TX would do auto-calibration and get parameters at the first time powering up after flashed new image, and then these parameters are written to flash. Therefore, it is necessary to ensure that there is no object on the TX surface at the first time powering up after flashed new image.

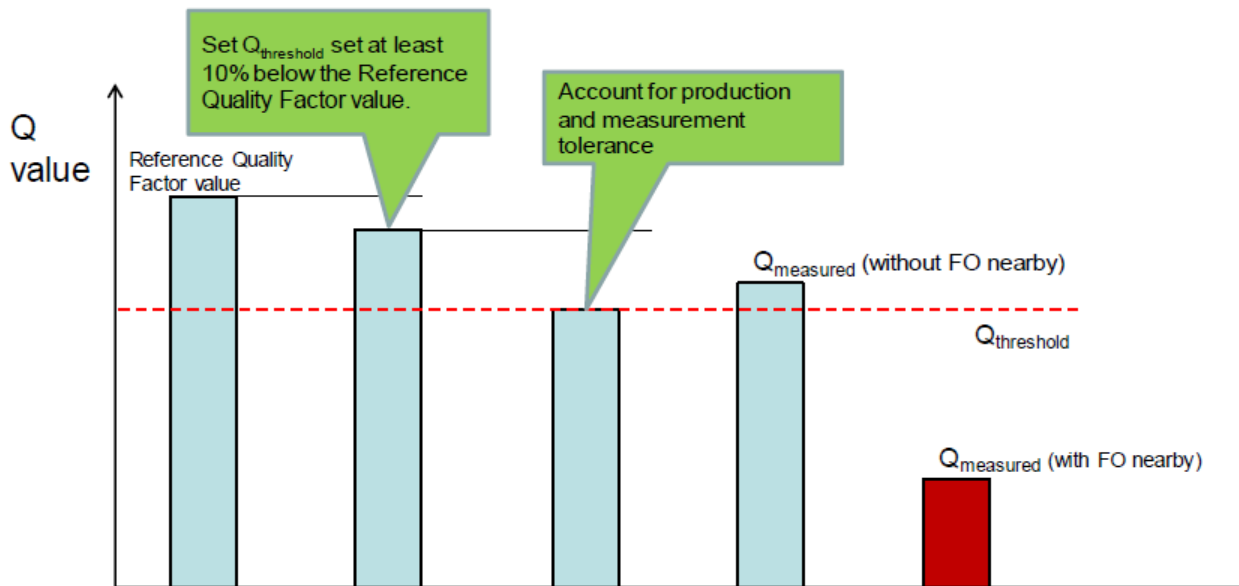


Figure 10. Quality factor threshold example

4.7.1. Free Resonance Q factor

The free resonance Q factor detection is to detect the decay rate of the resonance signal, as shown in [Figure 11](#). With the system's high Q, driving just a few pulses near resonant frequency are sufficient to serve as impulses and start the system ringing. Collect ADC data of tank voltage (or coil current), and then get the decay rate of the signal.

$$Q = \pi / (-\ln(\text{Rate}))$$

Rate is the value of decay rate of resonance signal.

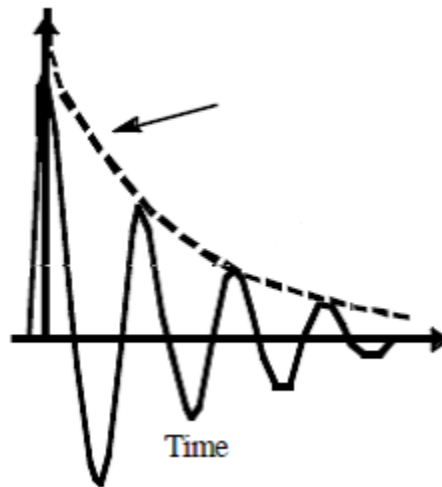


Figure 11. Resonance signal

The circuit for free resonance Q measurement is as shown in the following figure, which samples the signal on resonance capacitors.

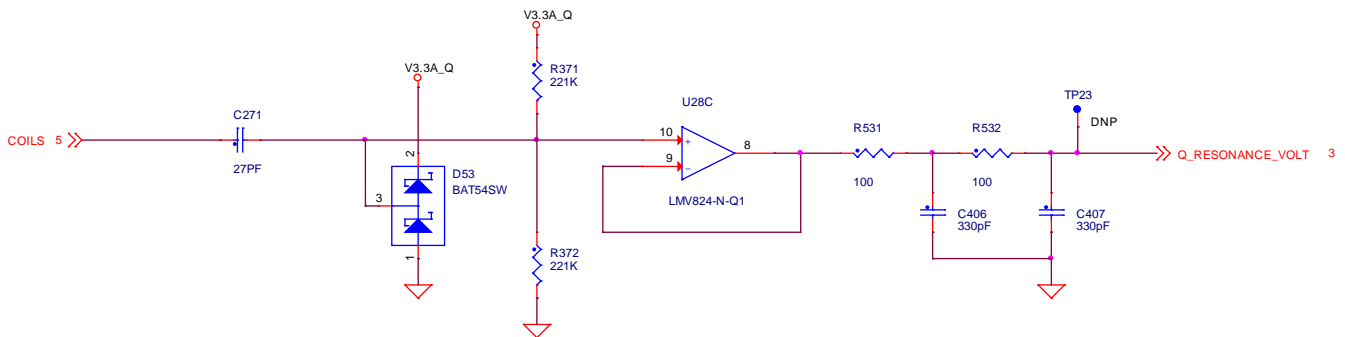


Figure 12. Free resonance Q measurement circuit

4.8. Coil selection

The Qi specification defines the MP-A9 as the more-than-one coil topology with one coil energized at a time to realize the free position charging.

The coil selection topology connects only one coil to the full-bridge inverter at a time. The coil is equipped with the dual N-MOSFETs, Q9, Q12, or Q16, controlled by the WCT1011A/WCT1013A controller through the control interface based on the low power bipolar transistors.

4.9. Analog sensing

Some ports of the ADC A-channel of WCT1011A/WCT1013A are used for sensing analog signals, such as temperature, full-bridge input current, input voltage, and rail voltage.

5. Application Monitoring and Control Through FreeMASTER

FreeMASTER is a user-friendly real-time debug monitor and data visualization tool for application development and information management. Supporting nonintrusive variable monitoring on a running system, FreeMASTER allows the data from multiple variables to be viewed in an evolving oscilloscope-like display or in a common text format. The application can also be monitored and operated from the web-page-like control panel.

5.1. Software setup

To set up the software, perform the following steps:

1. Install FreeMASTER V2.0.2 or later version from the NXP website: nxp.com/freemaster
2. Plug the USB-UART converting board to SCI connector J2, and connect the FreeMASTER MicroUSB port to your computer.
3. Open the Device Manager, and check the number of the COM port.

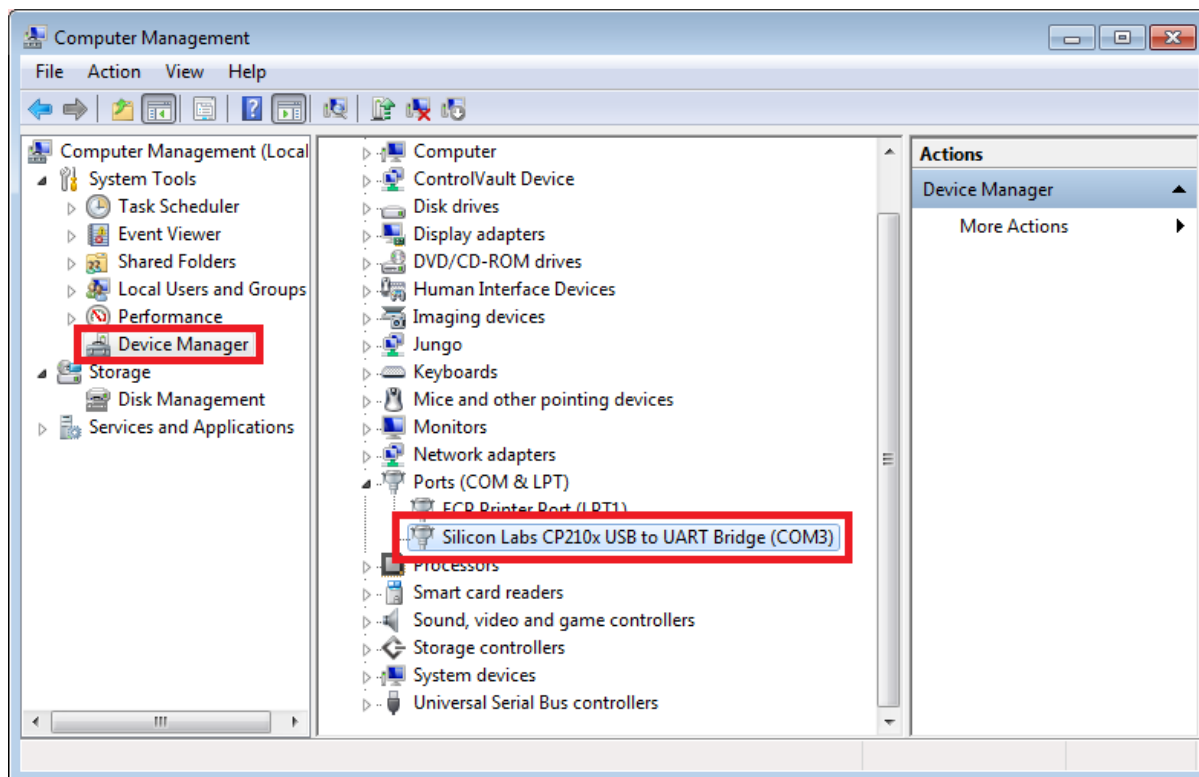


Figure 13. Device manager

4. Unpack the embedded source code to your local disk.
5. Start the FreeMASTER application by opening:
 - MWCT1013A
 <unpacked_files_location>/15W_MP/example/wct1013a/ wct1013A.pmp
 - MWCT1011A
 <unpacked_files_location>/15W_MP/example/wct1011a/ wct1011A.pmp
6. Choose **Project** → **Options**.

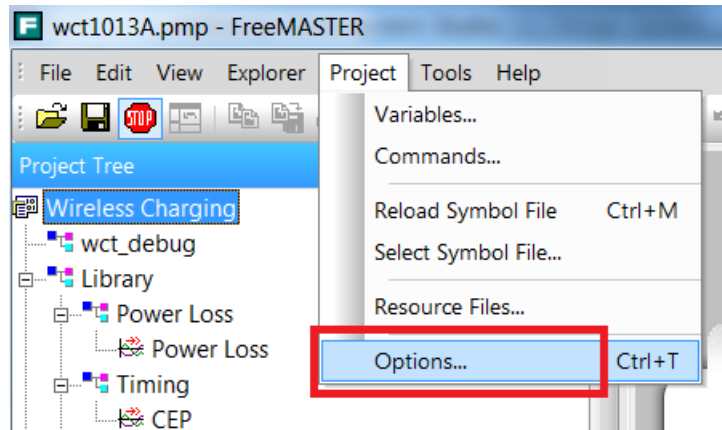


Figure 14. Choosing Options

7. Ensure that the correct virtual **Port** (according to Step 3) and **Speed** are selected.

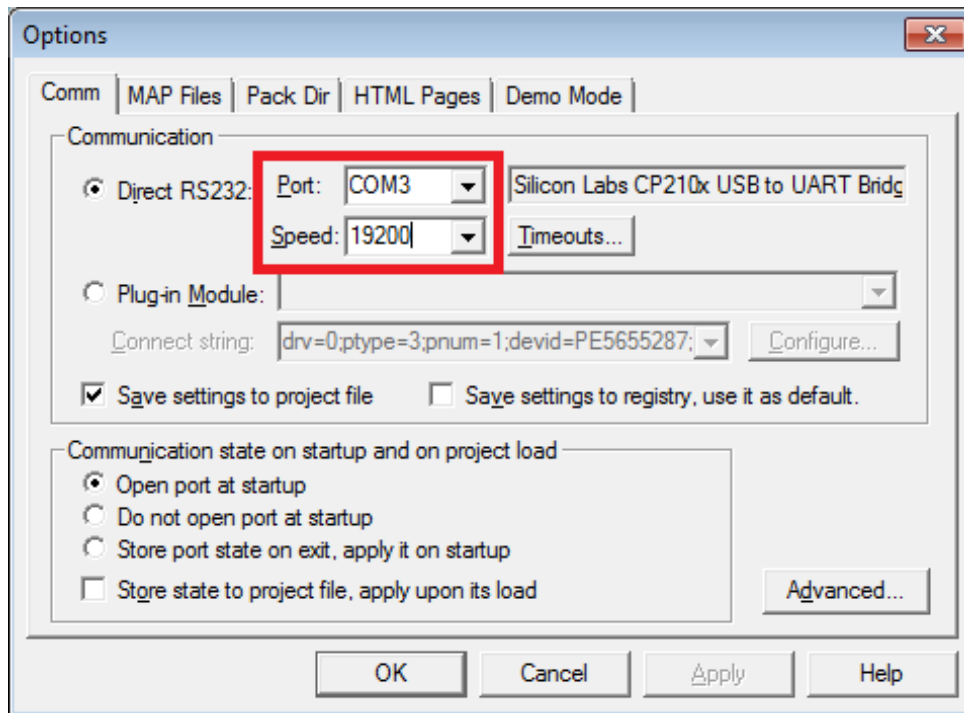


Figure 15. Setting Port and Speed

8. Ensure that the MAP file is correct. The default directories are as follows:

- MWCT1013A

<unpacked_files_location>/15W_MP/build/demo/wct1013Ademo/demo_ldm_debug/wct1013A
demo_debug.elf

- MWCT1011A

<unpacked_files_location>/15W_MP/build/demo/wct1011Ademo/demo_sdm_debug/wct1011A
demo_debug.elf

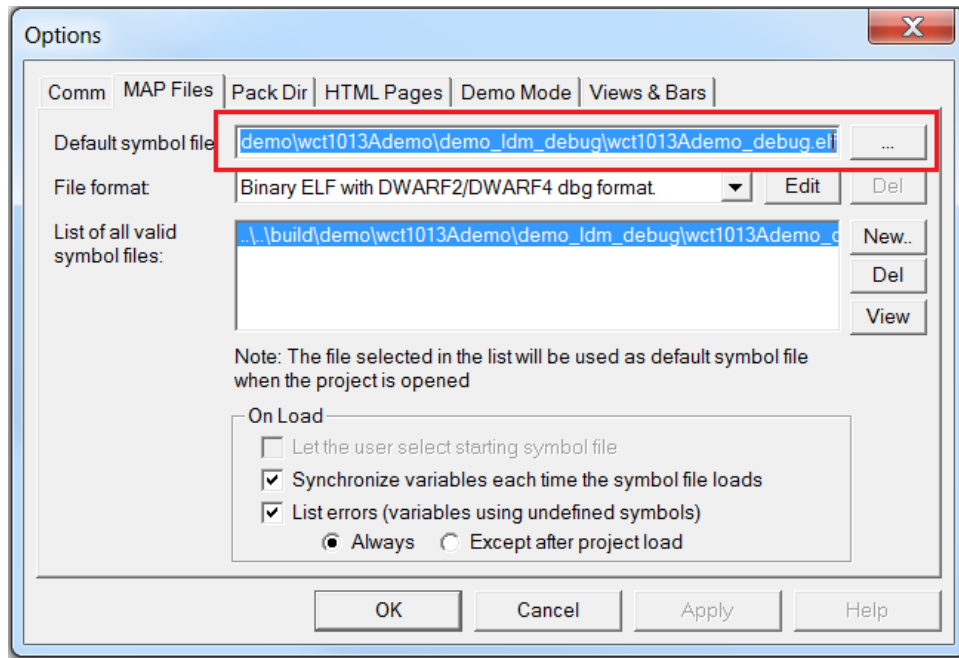


Figure 16. Setting the MAP file

9. Connect FreeMASTER.

Power on MP-A9, and then start the communication by clicking the STOP button on the FreeMASTER GUI.

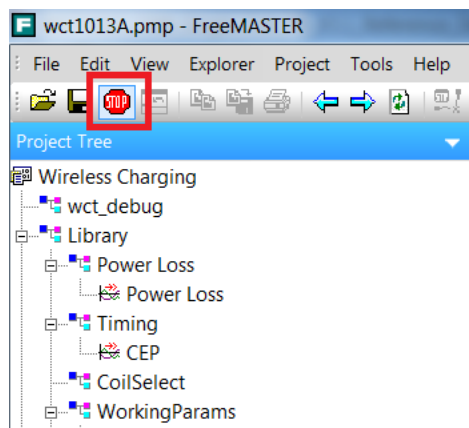


Figure 17. Stop button

5.2. Real-time application variables monitoring

FreeMASTER enables monitoring and updating all the application global variables. In this application, several key variables are displayed in the scope windows. These variables are divided into different blocks as shown in the following figure.

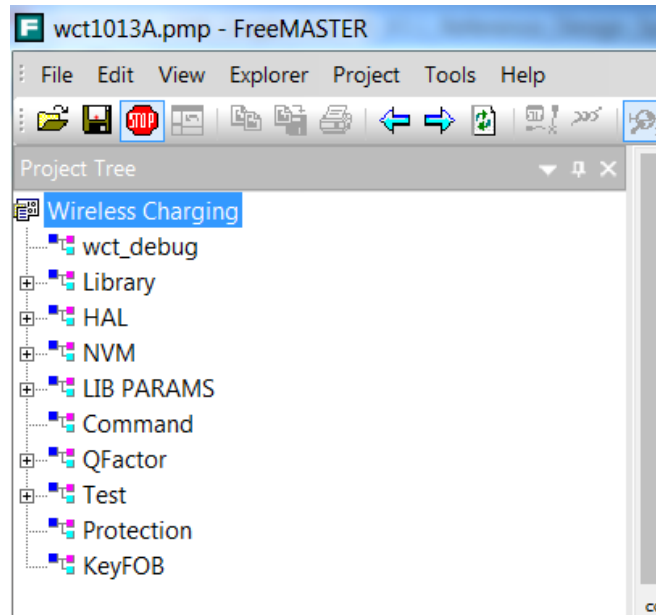


Figure 18. Real-time application variables

- wct_debug

This block shows the variables used for the GUI command.

- Library

This block contains power loss variables, timing variables, coil selection variables, working parameters, system status, DDM variables, and RX information.

- HAL

This block contains ADC raw data and DDM buffer values.

- NVM

This block lists all NVM parameters. Q factor sub-block shows the Q factor calibration constants. RRQD sub-block shows the quick removal calibration constants. FOD sub-block shows the FOD characterization calibration constants. Normalization sub-block shows the FOD normalization constants. Analog sub-block shows the rail voltage calibration constants.

- LIB PARAMS

This block lists all parameters used for the WCT library.

- Command

The command variable is used to stop WCT, start WCT, and do auto-calibration.

- QFactor

This block contains the variables for Q factor detection.

- Test

This block contains some variables for debugging.

- Protection

This block contains protection variables, such as input voltage protection, input current protection, and temperature protection.

- KeyFOB

This block contains the variables used for KeyFob.

NOTE

Besides the variables above, all the global variables can be added to FreeMASTER. The procedure to generate and add variables to watch window is described in the FreeMASTER user manual.

5.3. Application parameters modification

The application parameters (NVM parameters) can be easily viewed and changed on the control panel. The control panel contains the web page elements (buttons, check boxes, and text fields) that enable a user-friendly way to visualize and change the application control parameters.

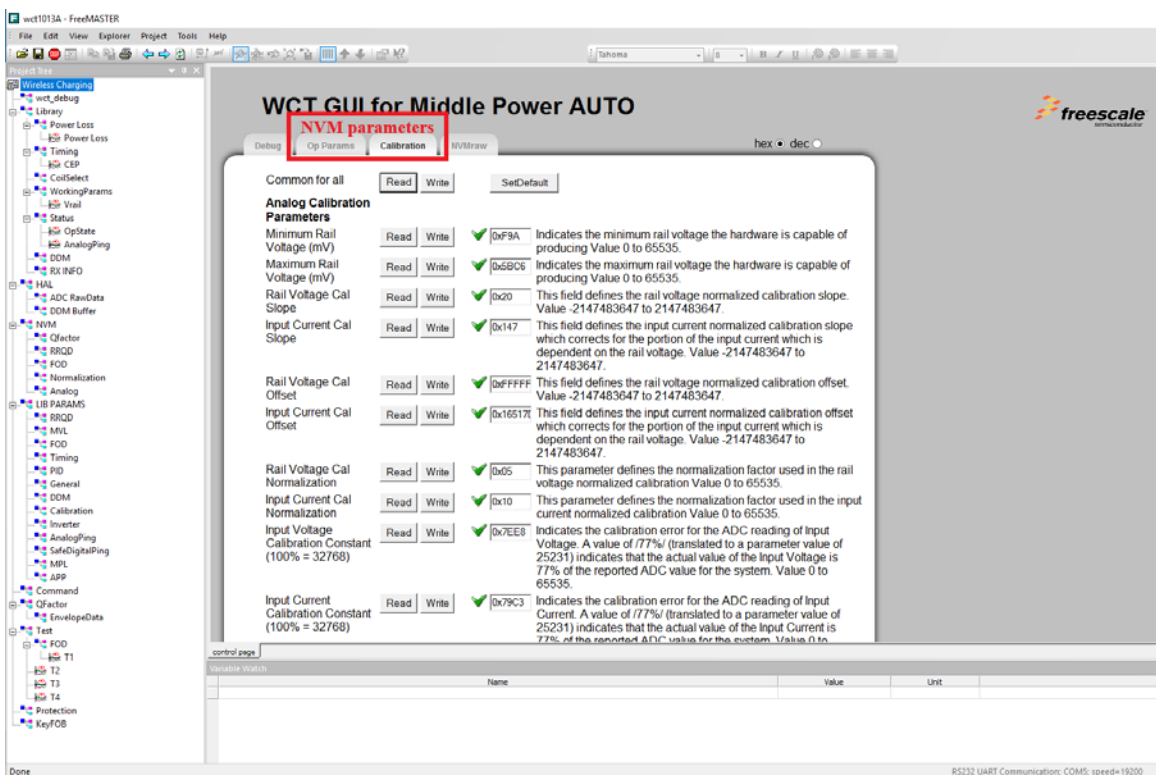


Figure 19. Application variables

The application variables are divided into three tabs:

- *Op Params* – enables access to the variables related to the operation control

- **Calibration** – group of parameters for calibration of the input current, input voltage, and foreign objects detector

The meaning of each parameter is described next to the text field.

NOTE

The parameters of Calibration can be changed at run-time. But the parameters of Op Params cannot take effect immediately. To modify the parameters in the Op Params page: enter the debug mode, modify the parameters, and exit the debug mode. The parameters can then take effect.

6. Application Monitoring Through Console

The application sends some information and error states through SCI to the console. The information is sent when the board is turned on, when the device is charging, or in case of some error state.

For digital buck-boost platform, only one SCI port (SCI0) is available. SCI0 is used for FreeMASTER by default. Users select an alternative method to enable the debug console.

1. Disable FreeMASTER and configure SCI0 as the debug console.

- a) `#define DEBUG_CONSOLE_SUPPORTED (TRUE)`
`#define FREEMASTER_SUPPORTED (FALSE)`

The macros are defined in example->wct101xa->configure-> appcfg.h.

- b) `#define QSCI_CONSOLE_INDEX 0`
`#define QSCI_FREEMASTER_INDEX 1`

The macros are defined in example->wct101xa->driver->qsci.h

- c) `gWCT_Params.tDebugConfig.bGeneralDbg = 1;`

This variable is in wct_LibParams.c.

2. Change FreeMASTER communication interface to JTAG and configure SCI0 as the debug console.

- a) `#define DEBUG_CONSOLE_SUPPORTED (TRUE)`
`#define FREEMASTER_SUPPORTED (TRUE)`

The macros are defined in example->wct101xa->configure-> appcfg.h.

- b) `#define QSCI_CONSOLE_INDEX 0`
`#define QSCI_FREEMASTER_INDEX 1`

The macros are defined in example->wct101xa->driver->qsci.h

- c) `#define FMSTR_USE_SCI 0 /* To select SCI communication interface */`
`#define FMSTR_USE_JTAG 1 /* 56F8xxx: use JTAG interface */`

The macros are defined in example->wct101xa->configure-> freemaster_cfg.h.

- d) `gWCT_Params.tDebugConfig.bGeneralDbg = 1;`

This variable is in wct_LibParams.c.

1.1 Software setup

1. Plug the USB-UART converting board to SCI connector J2, and connect the console MicroUSB port to the computer.
2. Open **Device Manager**, and check the number of the COM port.

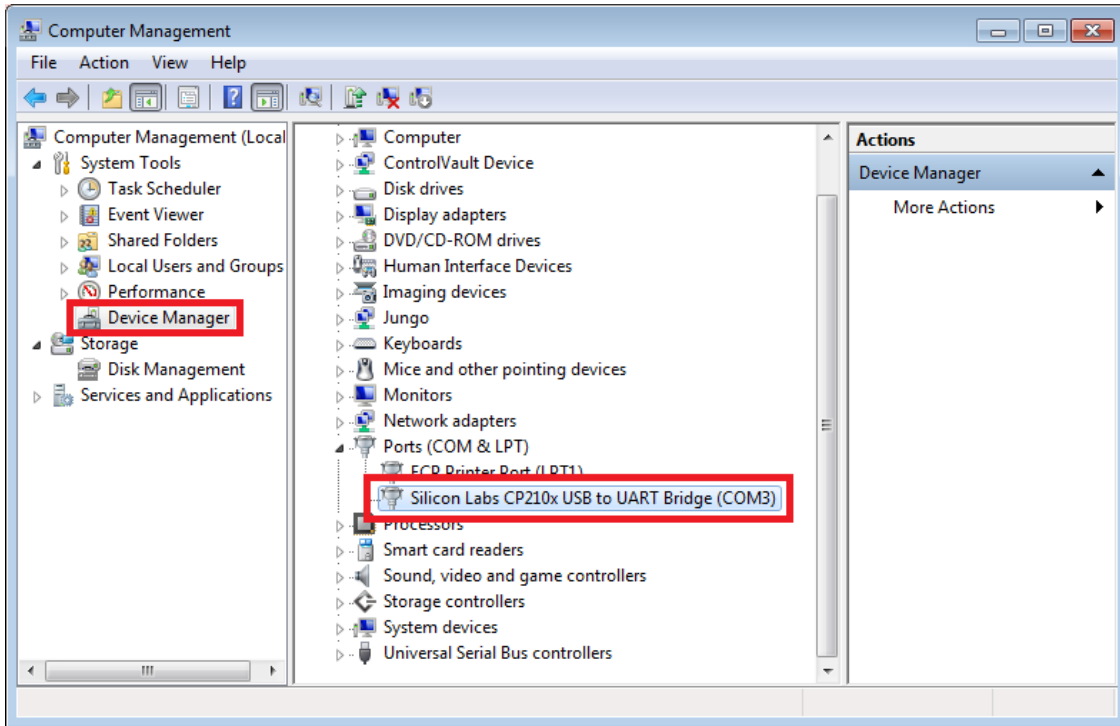


Figure 20. Device Manager

3. Run the communication program supporting console, such as HyperTerminal or RealTerm.
4. The following table shows the communication setup.

Table 1. Port configurations

Port number	Serial port from Device Manager
Baud	19200
Data Bits	8
Stop Bits	1
Parity	None
Hardware Flow Control	None
Display As	ASCII

5. Open the port or start communication, which depends on the used Terminal.

7. Programming New Software and Calibration

Users are provided with a software package, which includes a WCT1011A/WCT1013A project and a Bin file (.elf or .S). Users can flash alternative to the board. After flashing new software, board calibration must be carried out.

7.1. Install latest CodeWarrior

NOTE

The following steps demonstrate the installation of CodeWarrior for Microcontrollers v10.7 as an example.

1. Download installation files.

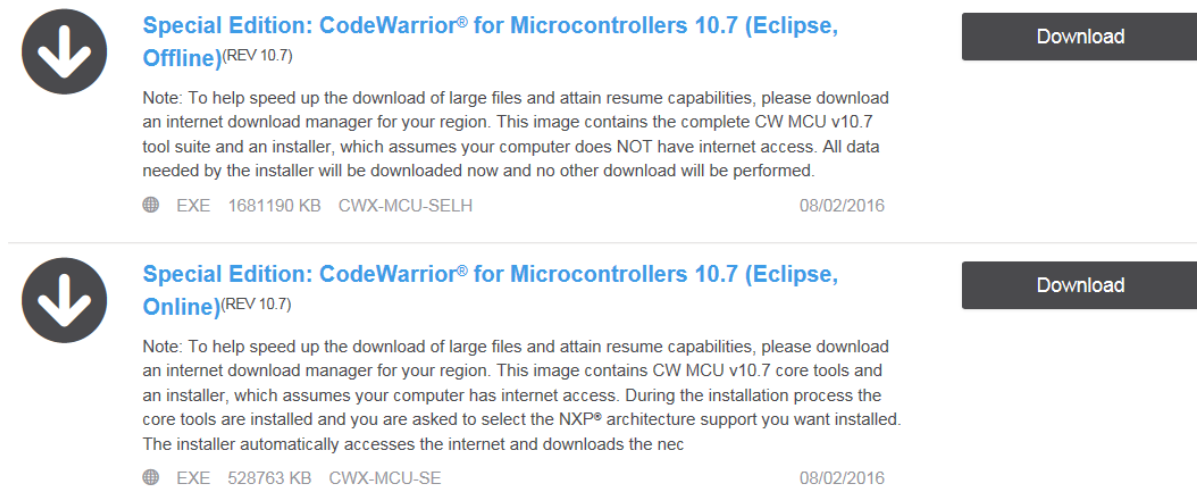
For proper installation of CodeWarrior 10.7, install both **CodeWarrior for Microcontrollers 10.7** and **CodeWarrior for MCUs v10.7 service pack**.

Access the following webpage and log in:

https://www.nxp.com/products/developer-resources/software-development-tools/codewarrior-development-tools/codewarrior-development-suites/codewarrior-development-suite-special:CW-SUITE-SPECIAL?tab=Design_Tools_Tab

Click **Download** for **CodeWarrior Special Edition (offline or online)**.

IDE - Debug, Compile and Build Tools (8)





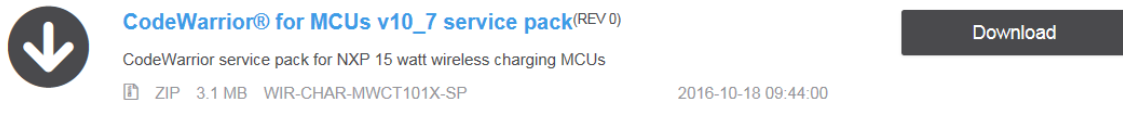
	Special Edition: CodeWarrior® for Microcontrollers 10.7 (Eclipse, Offline) ^(REV 10.7)	Download
	Note: To help speed up the download of large files and attain resume capabilities, please download an internet download manager for your region. This image contains the complete CW MCU v10.7 tool suite and an installer, which assumes your computer does NOT have internet access. All data needed by the installer will be downloaded now and no other download will be performed.	
	EXE 1681190 KB CWX-MCU-SELH 08/02/2016	
	Special Edition: CodeWarrior® for Microcontrollers 10.7 (Eclipse, Online) ^(REV 10.7)	Download
	Note: To help speed up the download of large files and attain resume capabilities, please download an internet download manager for your region. This image contains CW MCU v10.7 core tools and an installer, which assumes your computer has internet access. During the installation process the core tools are installed and you are asked to select the NXP® architecture support you want installed. The installer automatically accesses the internet and downloads the nec	
	EXE 528763 KB CWX-MCU-SE 08/02/2016	

Figure 21. Clicking Download for CodeWarrior Special Edition

Download CodeWarrior for Microcontrollers 10.7 service pack from the following link.

https://www.nxp.com/products/power-management/wireless-charging-ics/15-watt-wireless-charging-transmitter-ics-for-automotive-applications:MWCT1x1xA?tab=Design_Tools_Tab

IDE - Debug, Compile and Build Tools (2)



CodeWarrior® for MCUs v10_7 service pack^(REV 0)
CodeWarrior service pack for NXP 15 watt wireless charging MCUs
ZIP 3.1 MB WIR-CHAR-MWCT101X-SP 2016-10-18 09:44:00

Download

Figure 22. Downloading CodeWarrior for MCU v10.7 service pack

2. Double-click **CW_MCU_v10.7_b160721_SE.exe** after downloading.

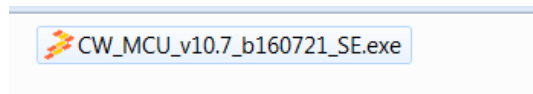


Figure 23. Setup file

3. Make sure that **DSC** is selected.

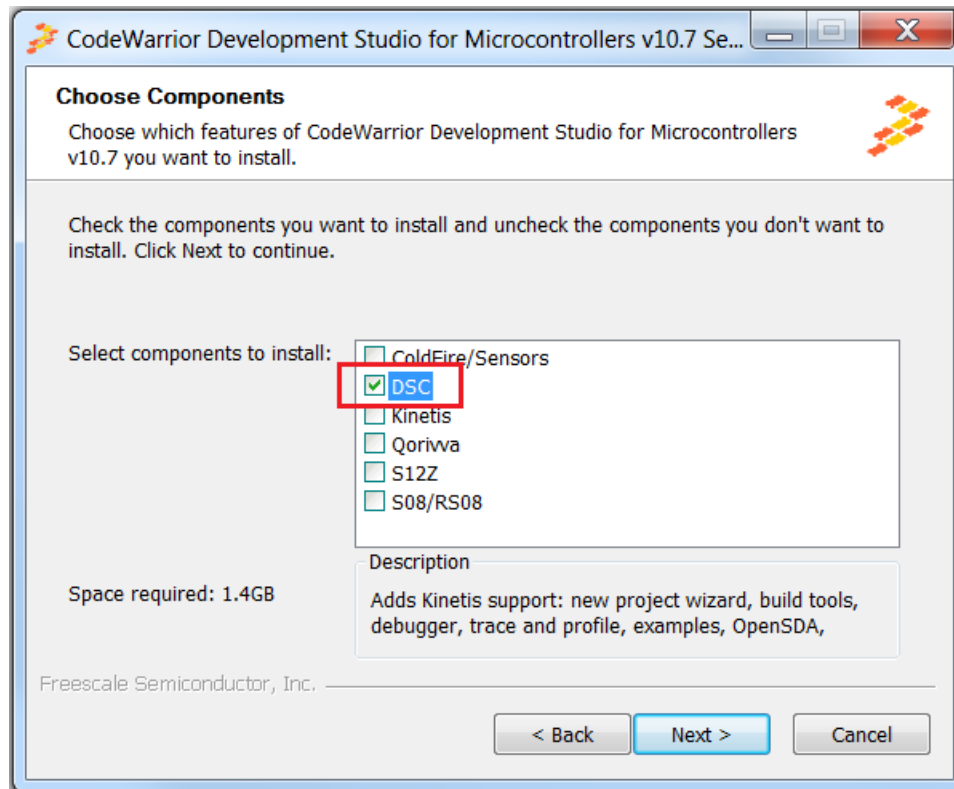


Figure 24. DSC installed

4. Launch CodeWarrior, create a folder workspace, and select it as the default workspace.

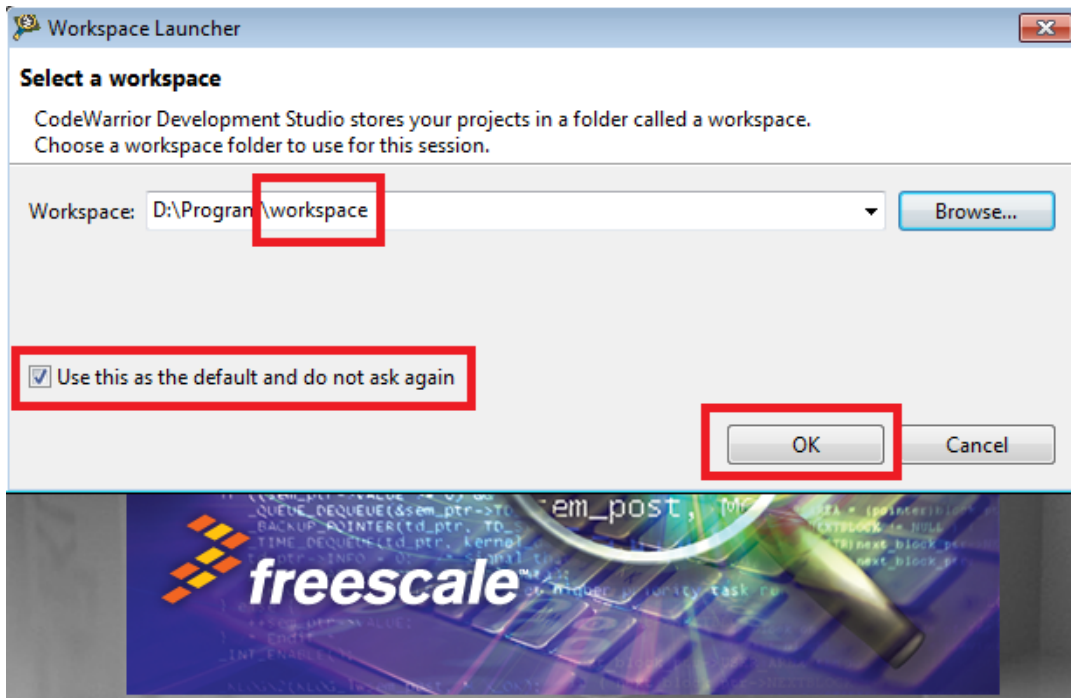


Figure 25. Workspace Launcher dialog box

5. Choose **Help** → **Install New Software**.

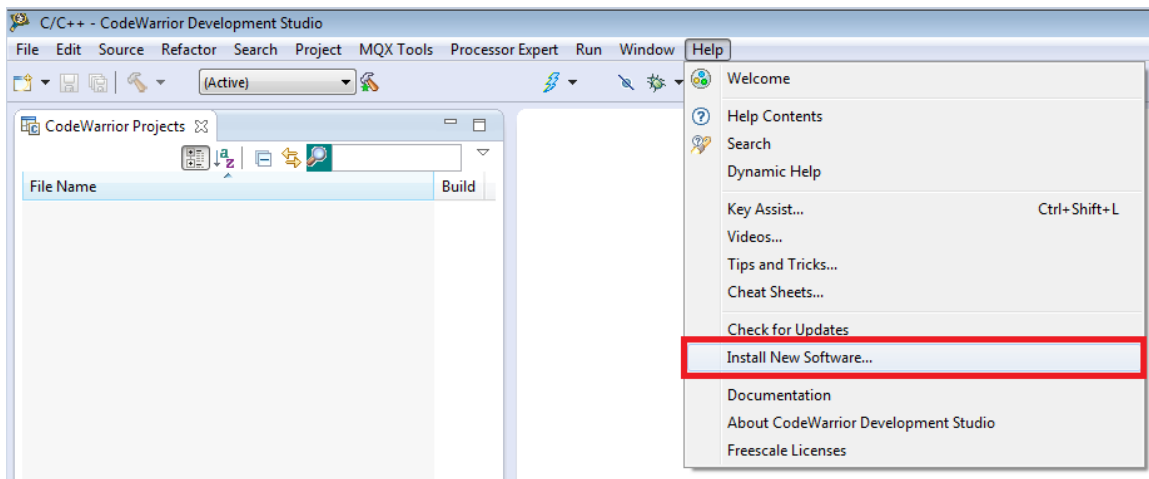


Figure 26. Install New Software

- Click **Add** and **Archive**, and then select the **mcu10_7.Wireless_Charging_MWCT101x.win.sp.v1.0.1.zip** file.

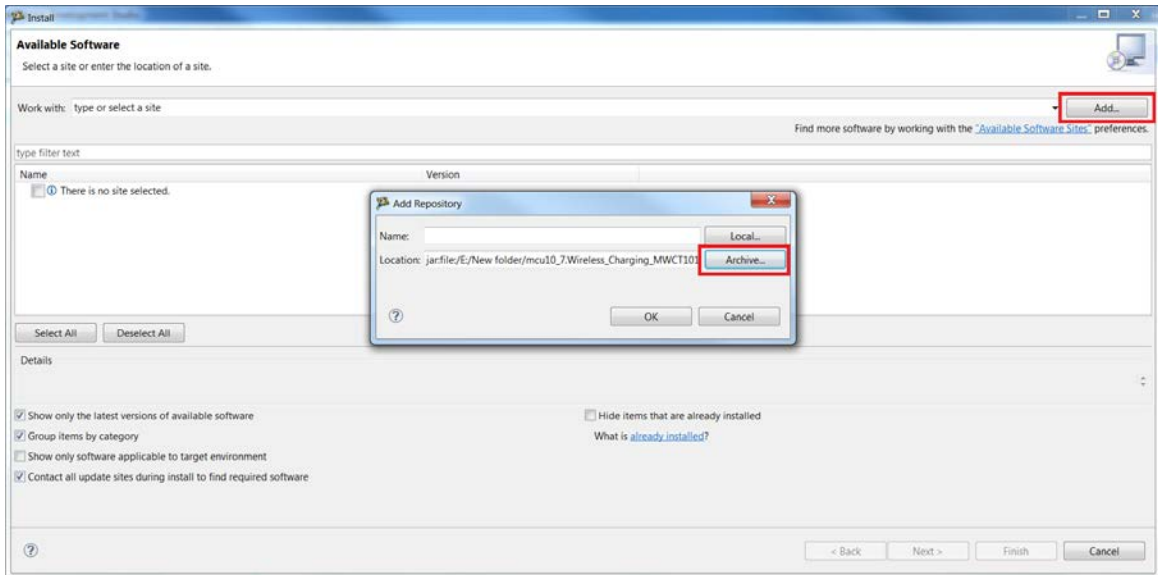


Figure 27. Selecting the update pack

- Select **MCU v10.7 DSC Service Packs**, and then click **Next**.

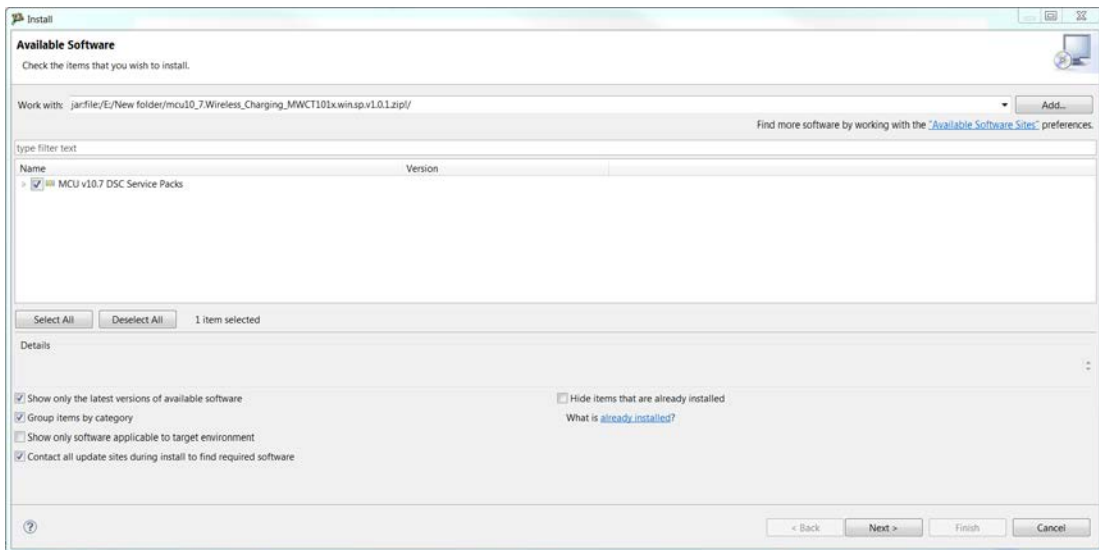


Figure 28. Selecting service packs

- Review the license terms. If you agree with the license terms, select **I accept the terms of the license agreement**, and then click **Finish**.

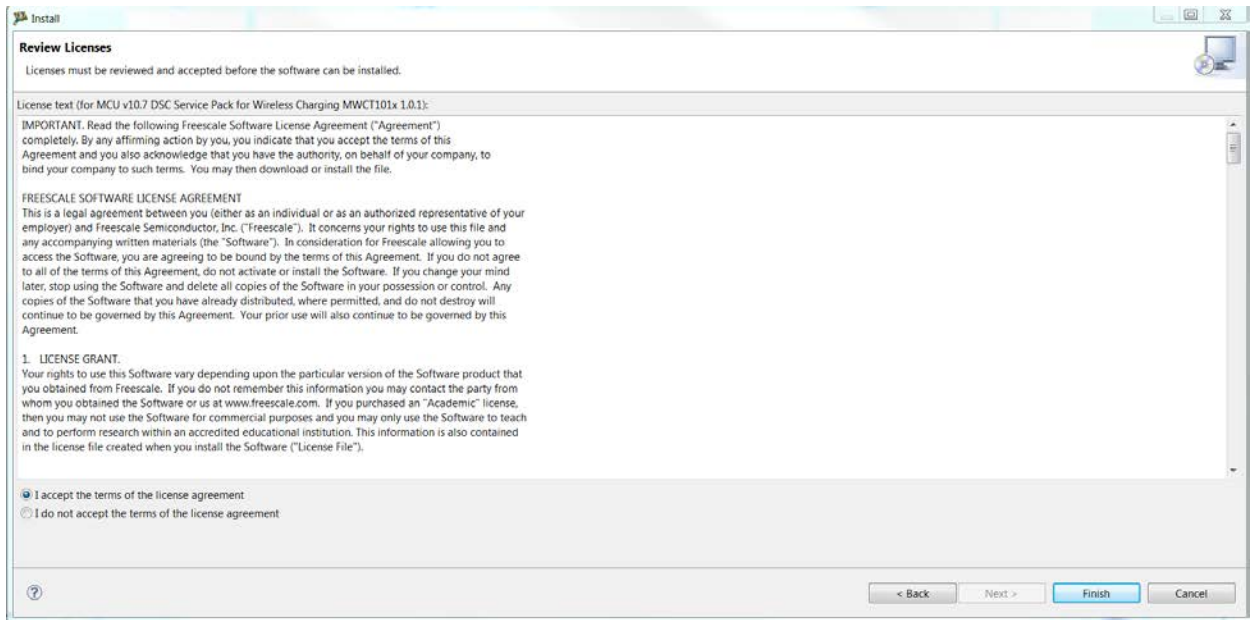


Figure 29. Installation finished

7.2. Board and programmer connection

Connect the 14-pin debug cable to J4 of the board (notice pin-1 position of cable).



Figure 30. Connecting the debug cable to the board

7.3. Program the project files

1. Import a project.
2. Right-click in the **CodeWarrior Projects** window and choose **Import** to import an existing project, as shown in the following figures. If the CodeWarrior Projects window is not displayed, open it through Window -> Show View -> CodeWarrior Projects.

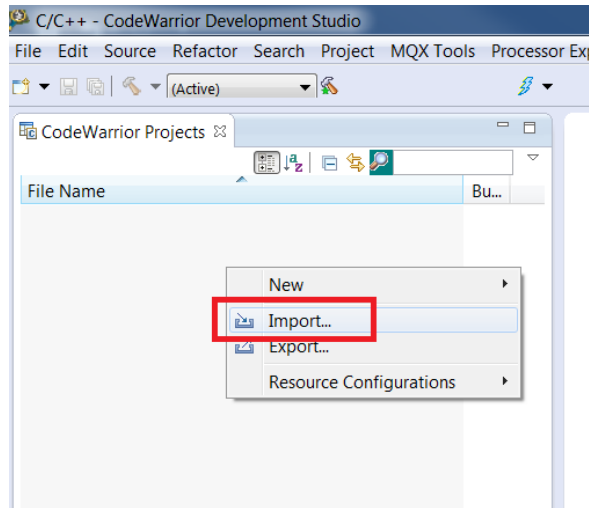


Figure 31. Importing a project (1)

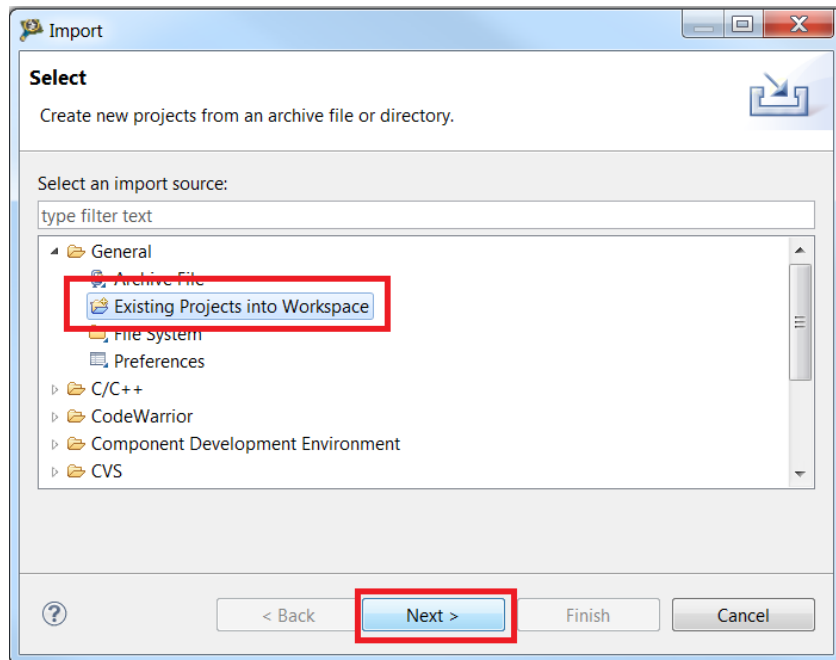


Figure 32. Importing a project (2)

3. Select the project directory, as show in this figure.

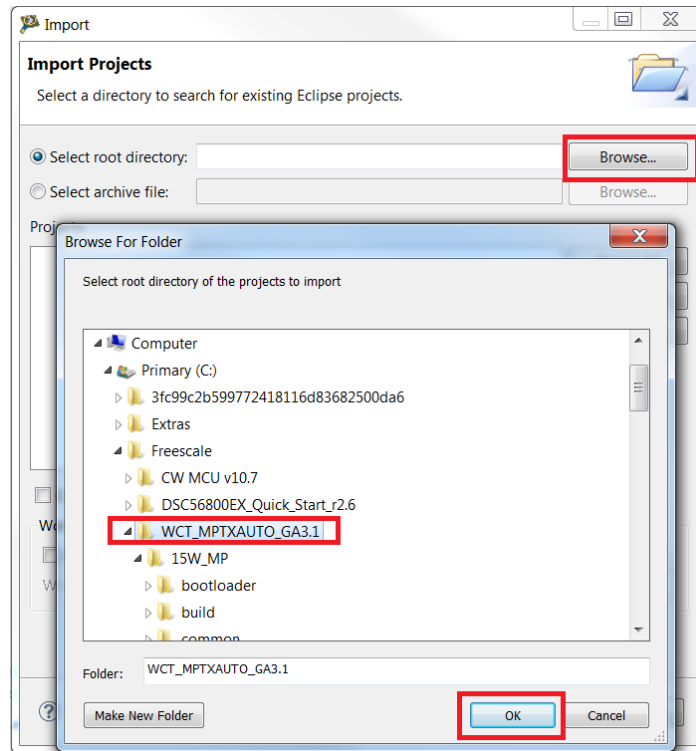


Figure 33. Importing a project (3)

4. Select WCT1013A project.

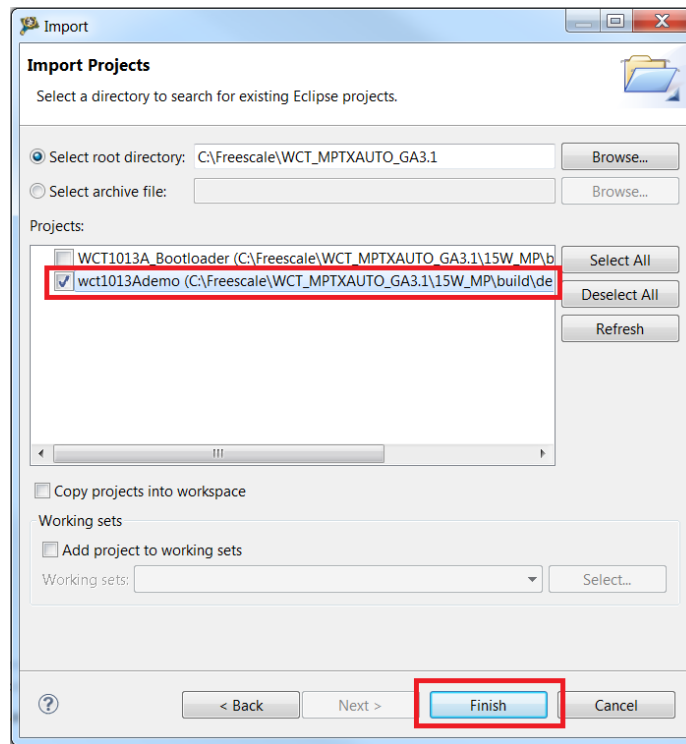


Figure 34. Importing a project (4)

5. Build a project.

Select build configurations by clicking the project name in the project window shown in the following figure. The demo_ldm_debug build contains debug information. The demo_ldm_release is same with demo_ldm_debug except debug information.

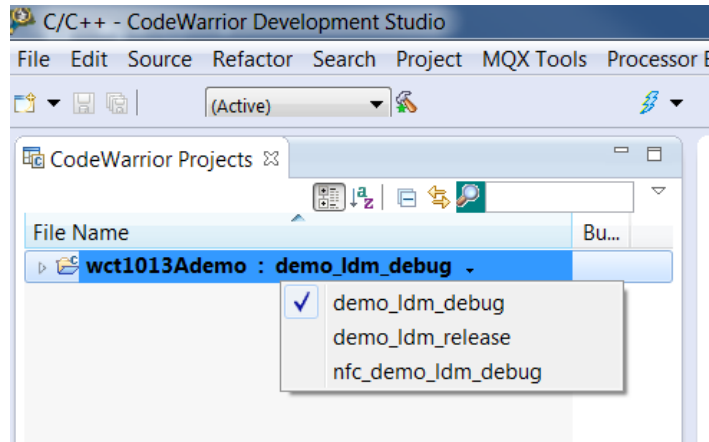


Figure 35. Building a project (1)

6. Right-click **wct1013Ademo**, and then choose **Clean Project** and **Build Project**.

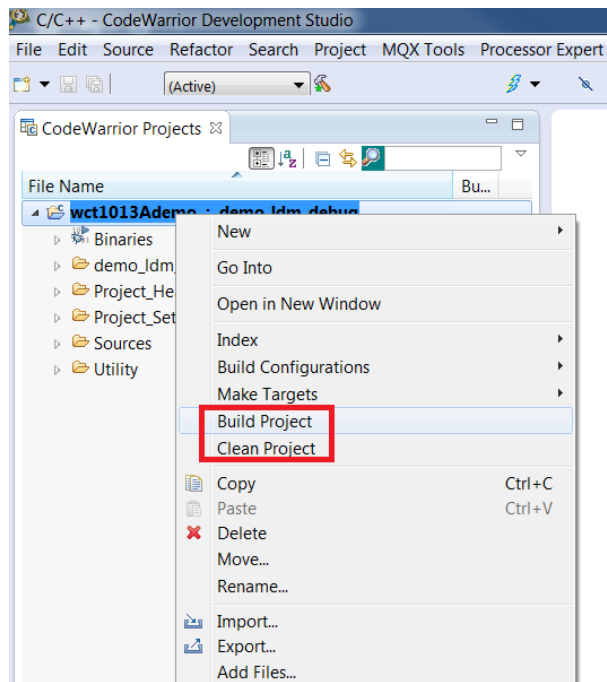


Figure 36. Clean Project and Build Project

7. Download the project.

Download the project from the **Debug** drop-down list or from **Run -> Debug**. In **Download Configurations**, select a download configuration according to your build configurations and debugger type, USB TAP, PnE Multilink, or OSJTAG.

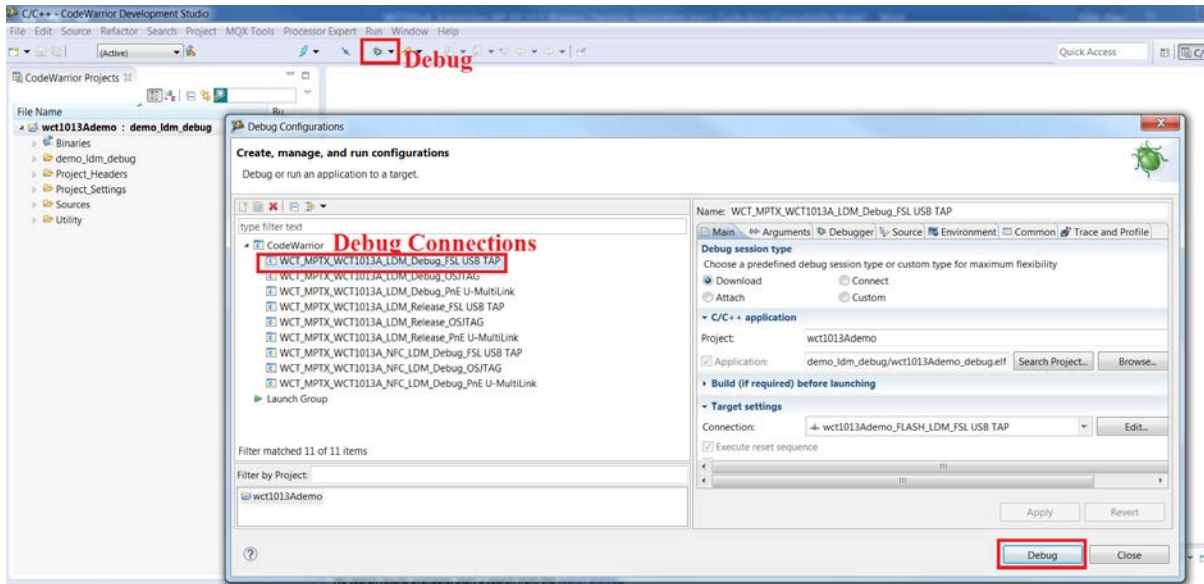


Figure 37. Downloading the project

After the project is downloaded, the MCU stops at the startup code. Click the **Run** button or press the F8 key to run MCU. Make sure that there is no object on the TX surface before making MCU run. Due to the auto-calibration for rail voltage, Q-factor, and quick removal will be done at the first-time the TX runs after flashed image.

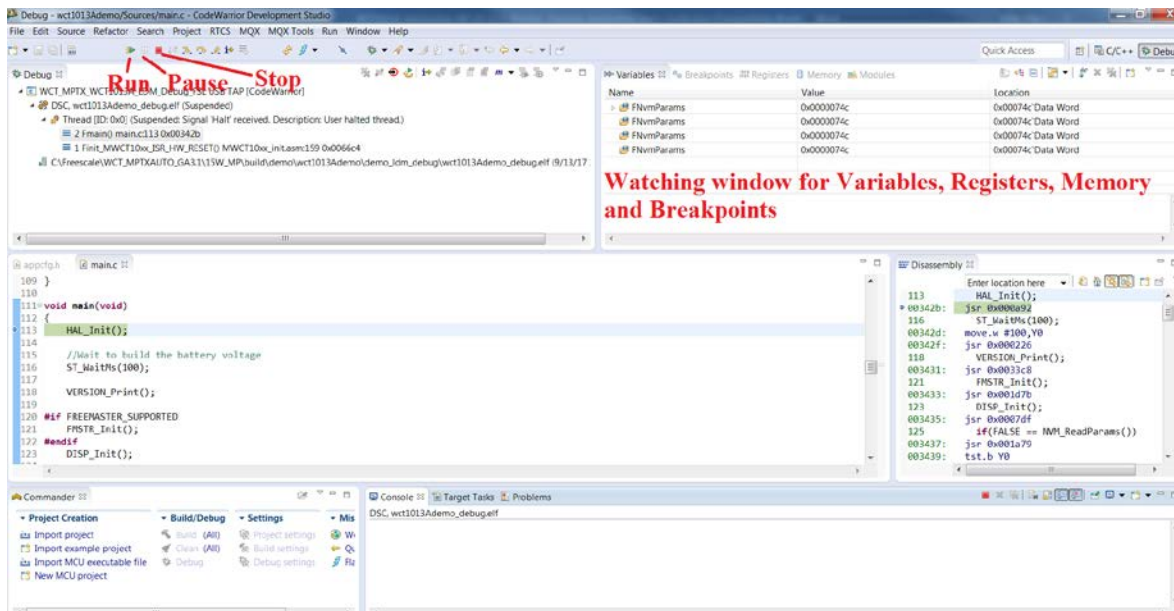


Figure 38. Project Downloaded

7.4. Program the Bin file (.elf or .S)

1. Choose **Flash Programmer** → **Flash File to Target**.

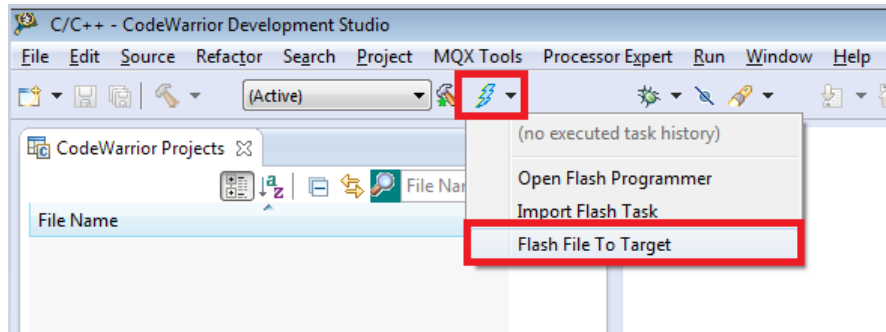


Figure 39. Choosing Flash File to Target

2. Click **New** to create a new connection.

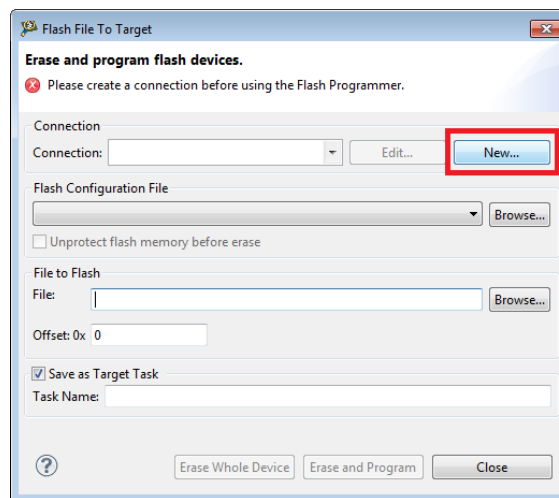


Figure 40. Creating a new connection

3. In the **Name** text box, enter a connection name (any name is OK), and click **New** to create a target.

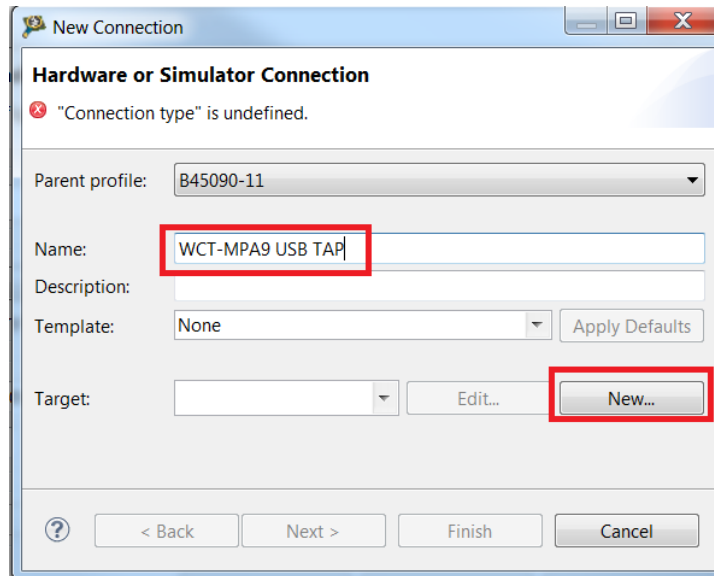


Figure 41. Entering a connection name

4. In the **Name** text box, enter a target name (any name is OK but cannot be same with the connection name), and choose **dsc.MWCT101x -> MWCT1013A** from the **Target Type** drop-down list.

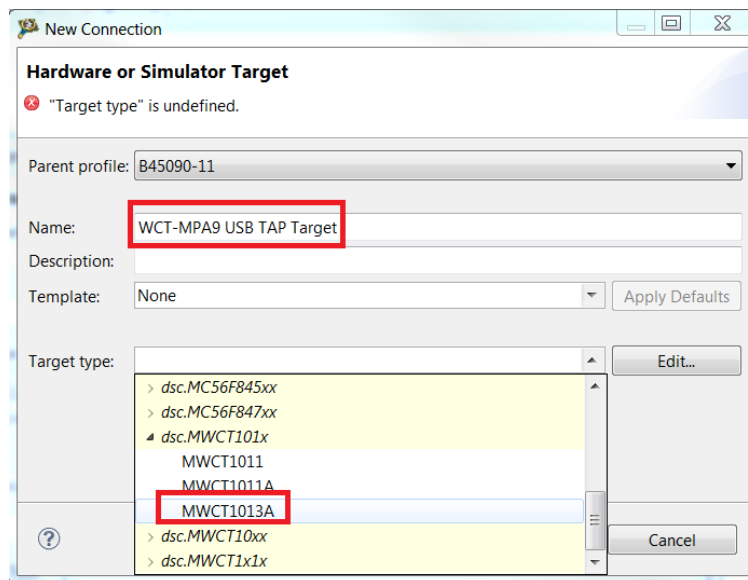


Figure 42. Choosing MWCT1013A

5. Select **Execute reset** and **Initialize target**, set the initialization target file path to the CW installation folder, and then select **MWCT1013A.tcl**.

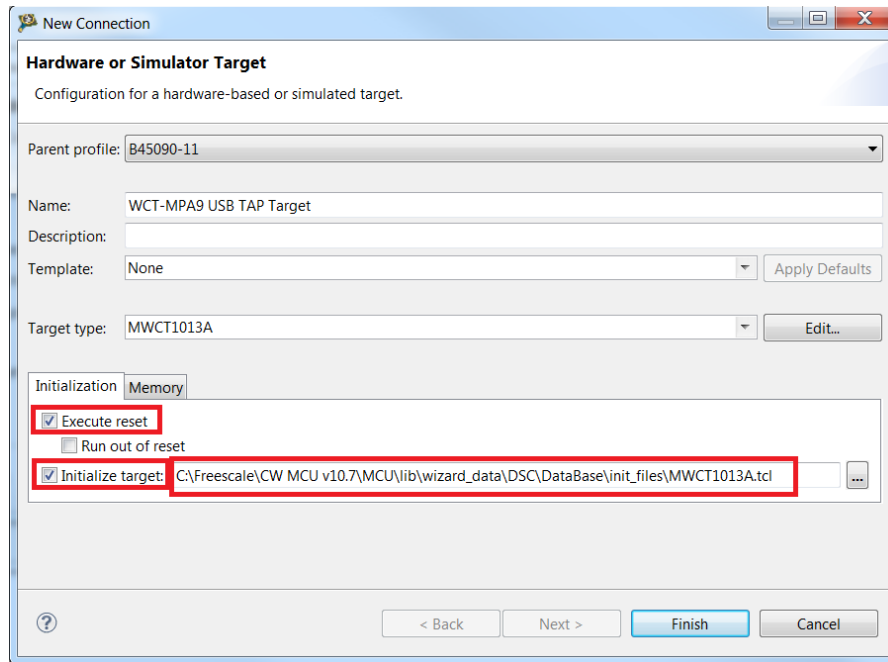


Figure 43. Executing reset and initializing target

6. Click the **Memory** tab.
7. Select **Memory configuration**, set the memory configuration file path to the CW installation folder, and then select **MWCT1013A.mem**.
8. Click **Finish**.

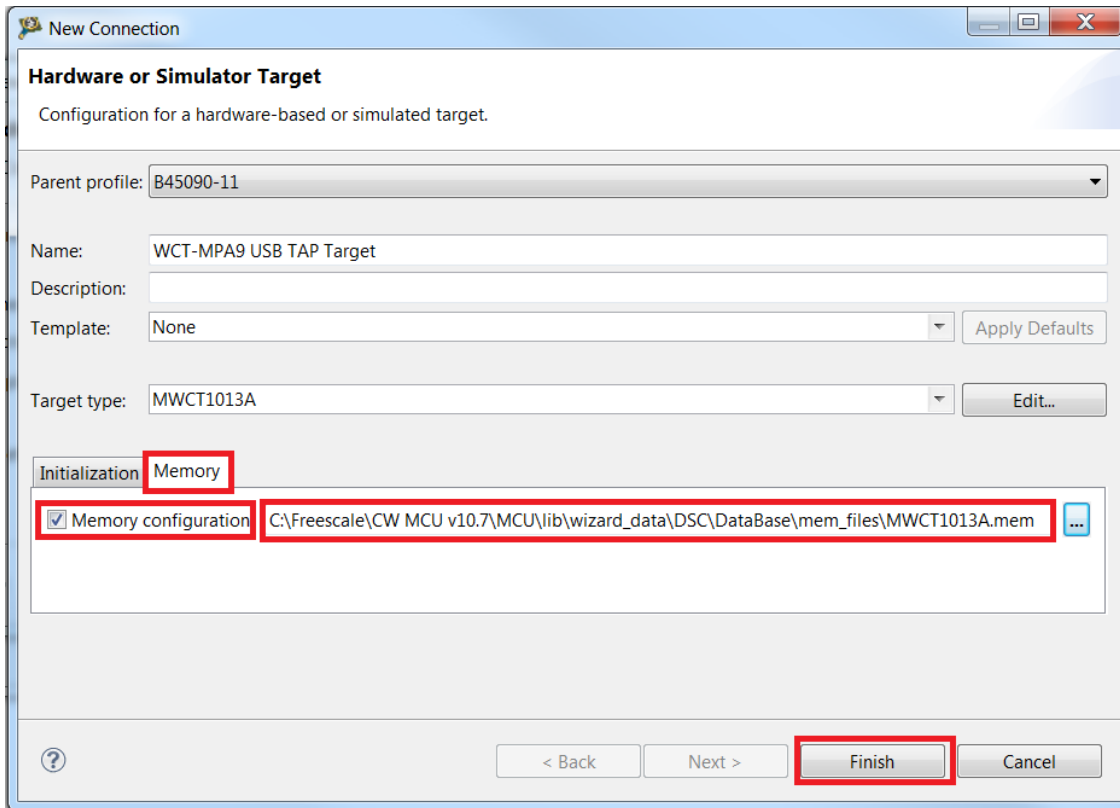


Figure 44. Memory configuration

9. Select **USB TAP** for the **Connection type**, and then click **Finish**.

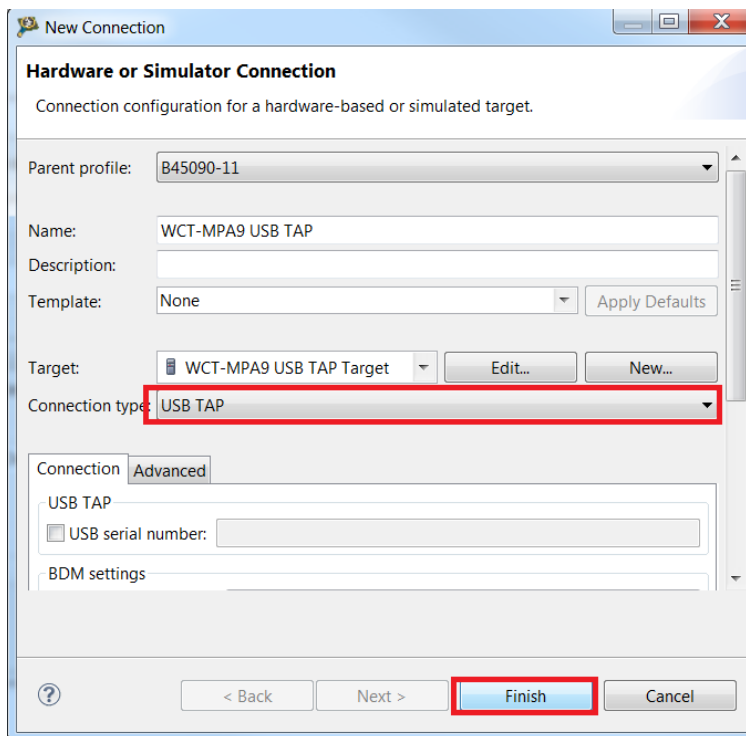


Figure 45. Setting the connection type

10. Set the Bin file path to be **File to Flash**. Select **Save the Target Task** for future programming. Power on MP-A9 and click **Erase and Program**.

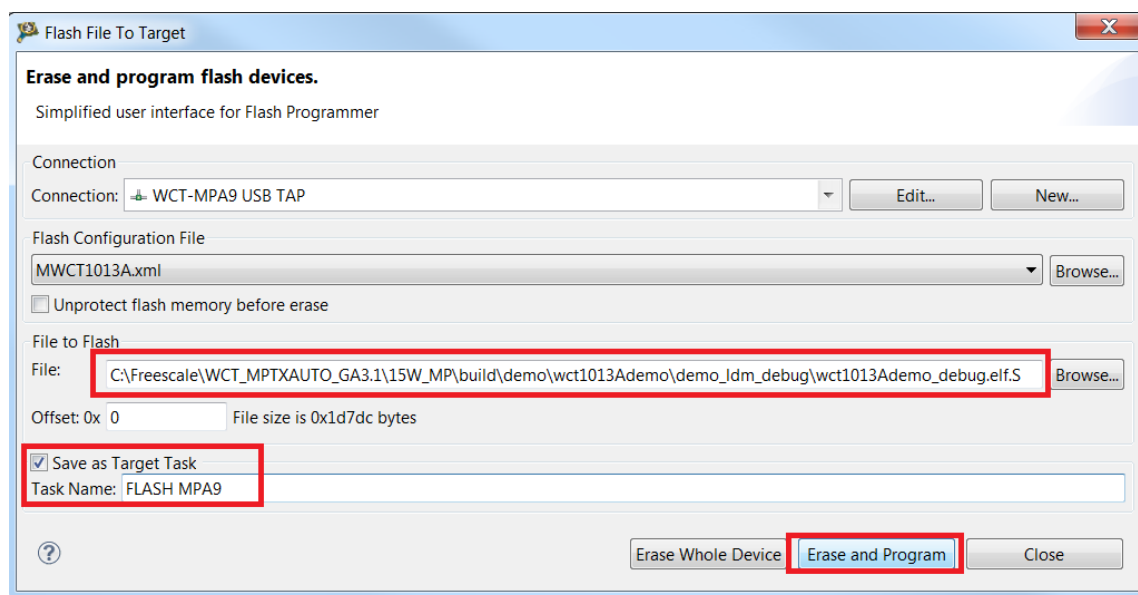


Figure 46. Erase and Program

11. Specify the task path and click OK to save the task.

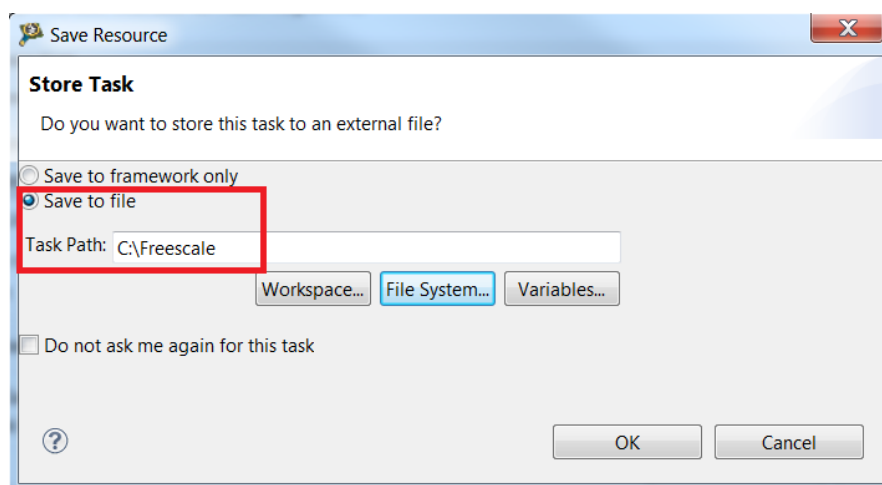


Figure 47. Selecting the task path

12. When program is finished, the **Console** window displays the following log.

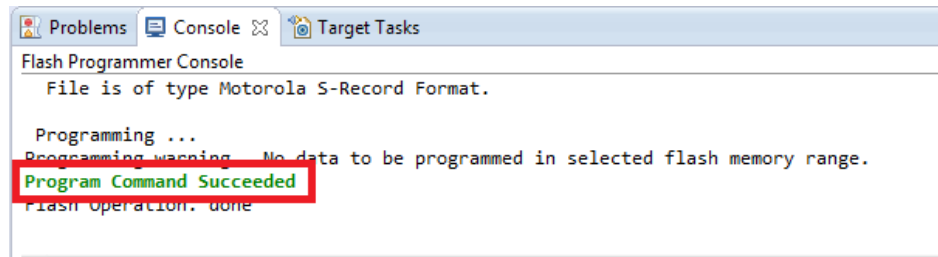


Figure 48. Programming finished log

13. For future programming, select **FLASH MPA9** and wait until the programming is finished.

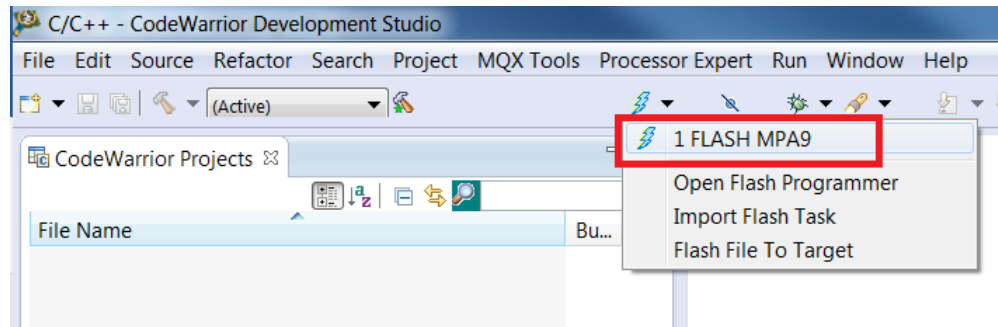


Figure 49. Future programming

7.5. Program by boot loader

Boot loader is independent of the application codes, and it can reside in flash forever after downloading an application code. When a system is reset, the boot loader starts to run. After a boot delay in seconds, the boot loader jumps to the programmed application code, even without receiving the application S-record file. When receiving the application S-record file, the boot loader programs the application code to the on-chip flash. After completion, the boot loader jumps to the application startup code.

1. The boot loader code is not flashed to the board by default. Download the boot loader code. Import the boot loader project and follow the steps described in Section 7.3 to download the project.

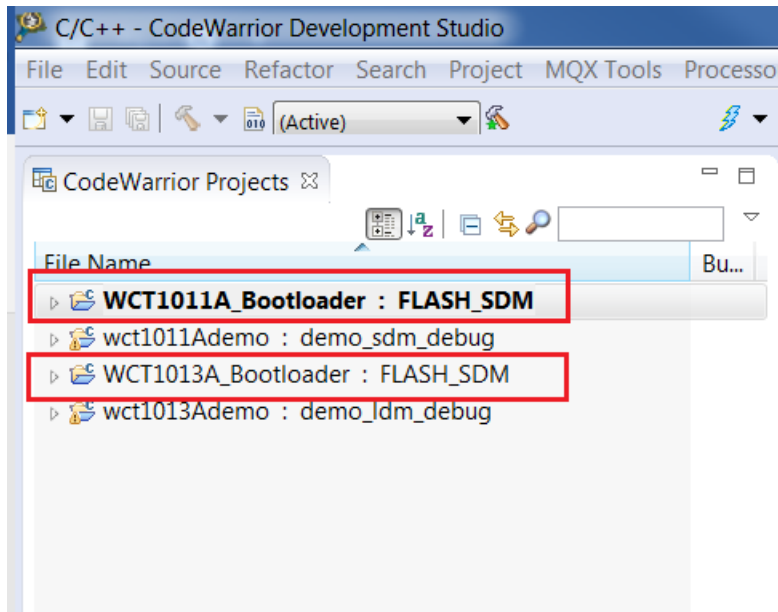


Figure 50. Boot loader project

2. Generate an application S-record file.

Enable the boot loader function. Set the following macro to TRUE in appcfg.h.

```
#define BOOTLOADER_USED    (TRUE)
```

Follow the figure below to configure the application project settings. Select the option **Sort by Address**. Set **Max S-Record Length** to a value that does not exceed **255**. Select **DOS (\\r\\n)** for **S-Record EOL Character**.

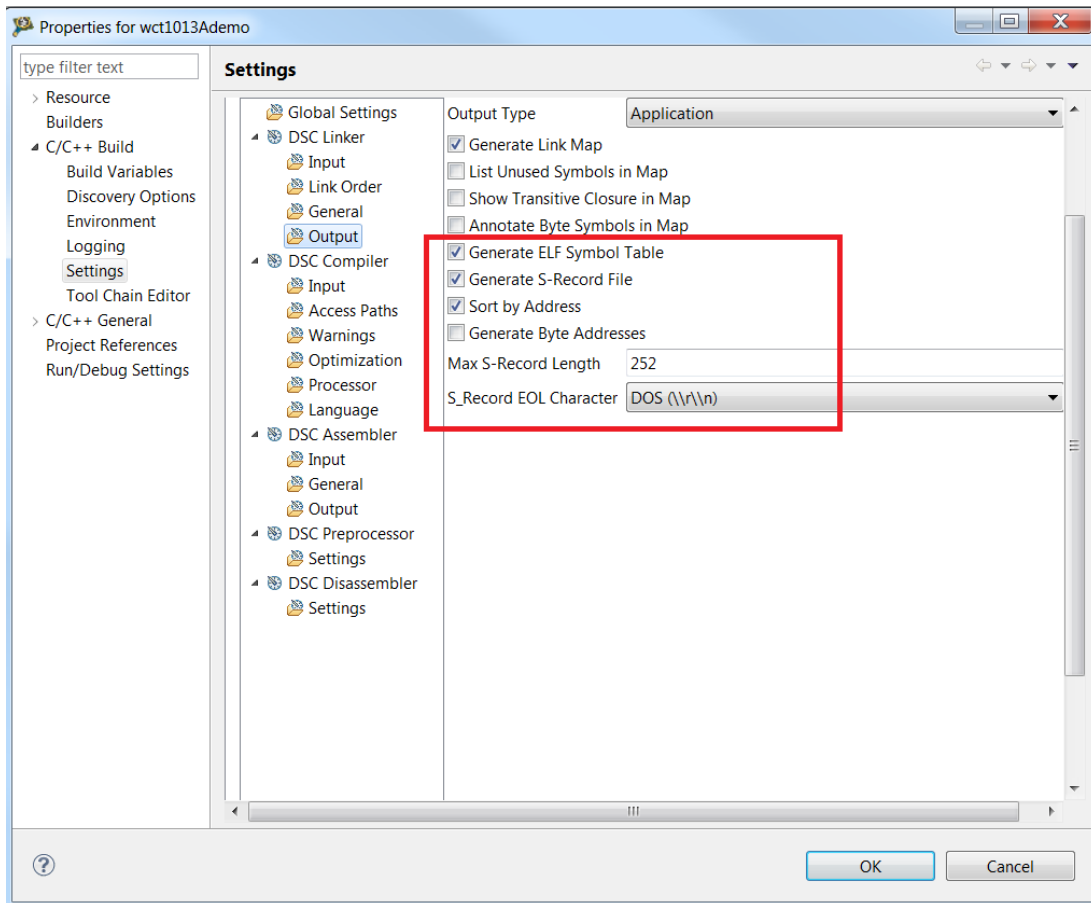


Figure 51. Configure project settings

Recompile the project after completing the configuration above. The generated S-record file is in <unpacked_files>\15W_MP\build\demo\wct1013Ademo\demo_ldm_debug for the WCT1013A chip or <unpacked_files>\15W_MP\build\demo\wct1011Ademo\demo_sdm_debug for the WCT1011A chip. The S-record file is the combined p and x S-record file without “.p” or “.x” in the extension name (.s).

3. Download the application code by the boot loader.

Most serial terminal programs can be used to send an S-Record file from a host to the WCT1011A/WCT1013A board through the boot loader. For example, Tera Term can be used in the Windows platform.

SCI0 is used for communication. Plug the USB-UART convertor to the SCI connector J2 and the computer. Open Tera Term, and then select **Serial** and **Port**. Check the COM port in the Device Manager.

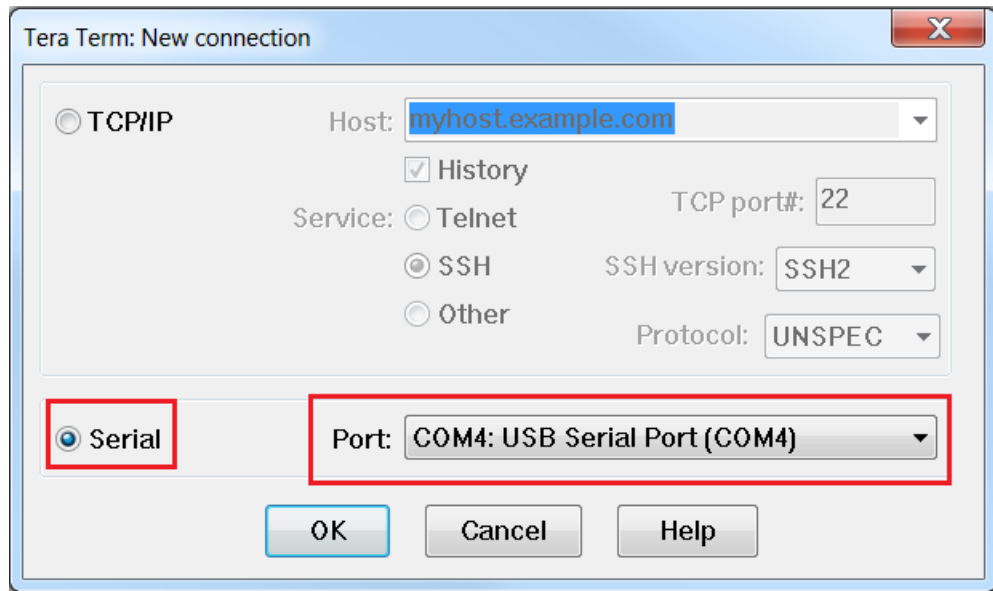


Figure 52. Tera Term connection

Choose **Setup** -> **Serial port** to configure the COM properties as shown in [Figure 54](#).

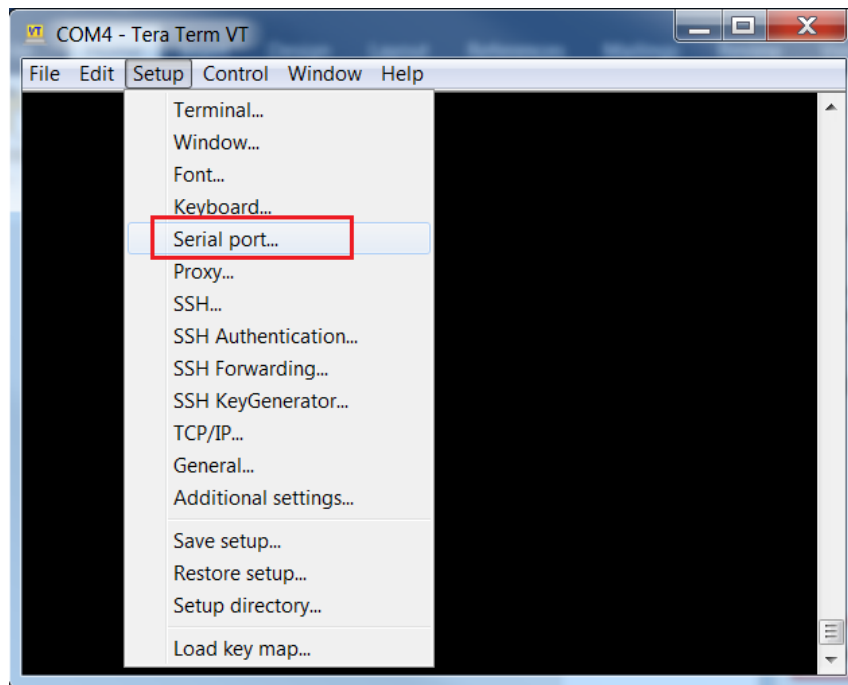


Figure 53. Serial port

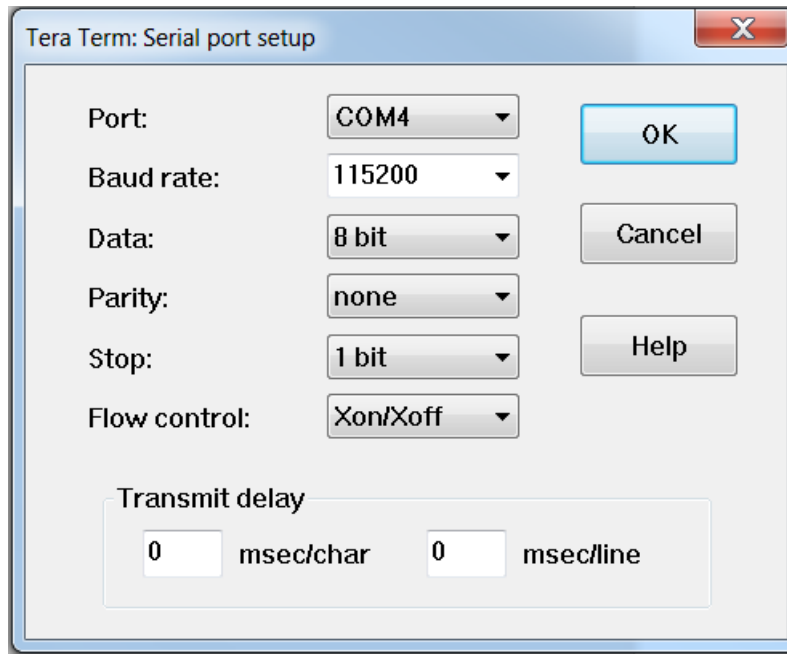


Figure 54. Serial port setup

Choose **File** -> **Send file**.

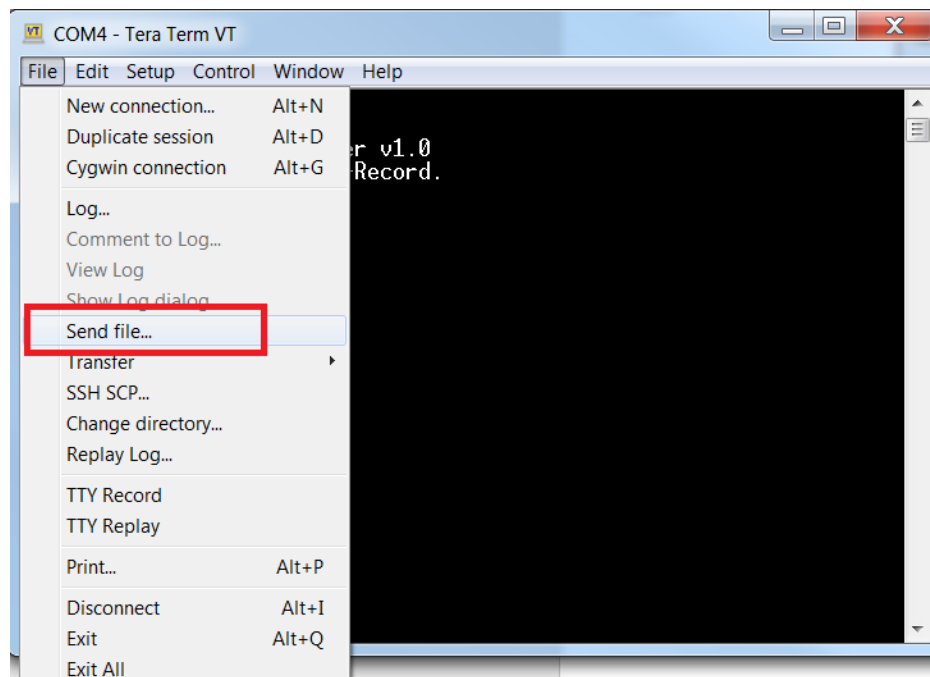


Figure 55. Figure 55 Send file

Select the application S-Record file and click **Open**.

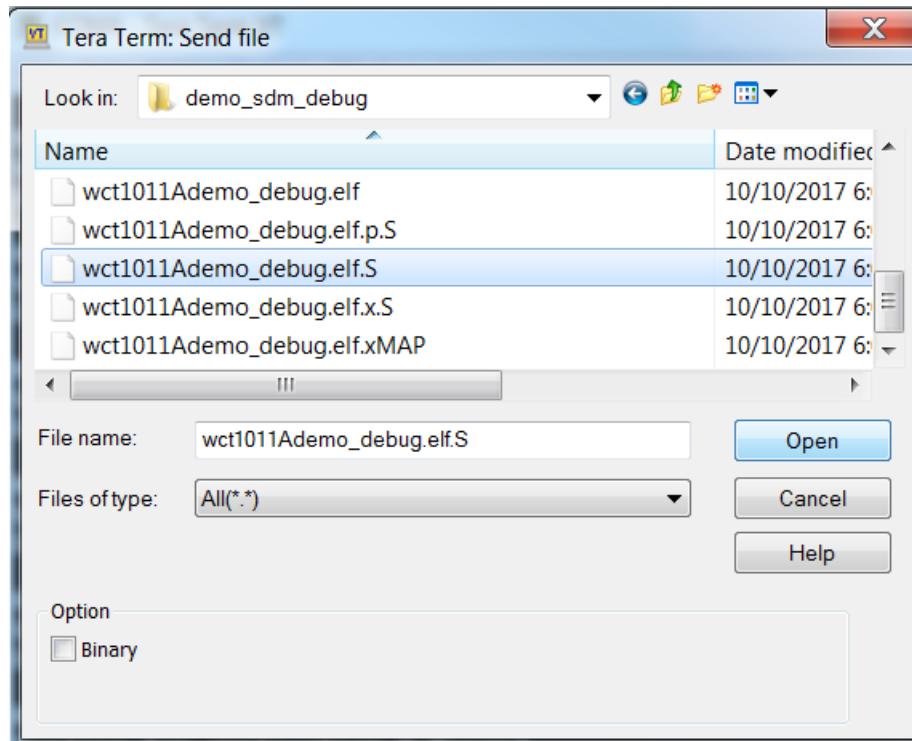


Figure 56. Send application S-Record file

The downloading progress is displayed in the **Tera Term** window. After downloading is completed, the application code starts.

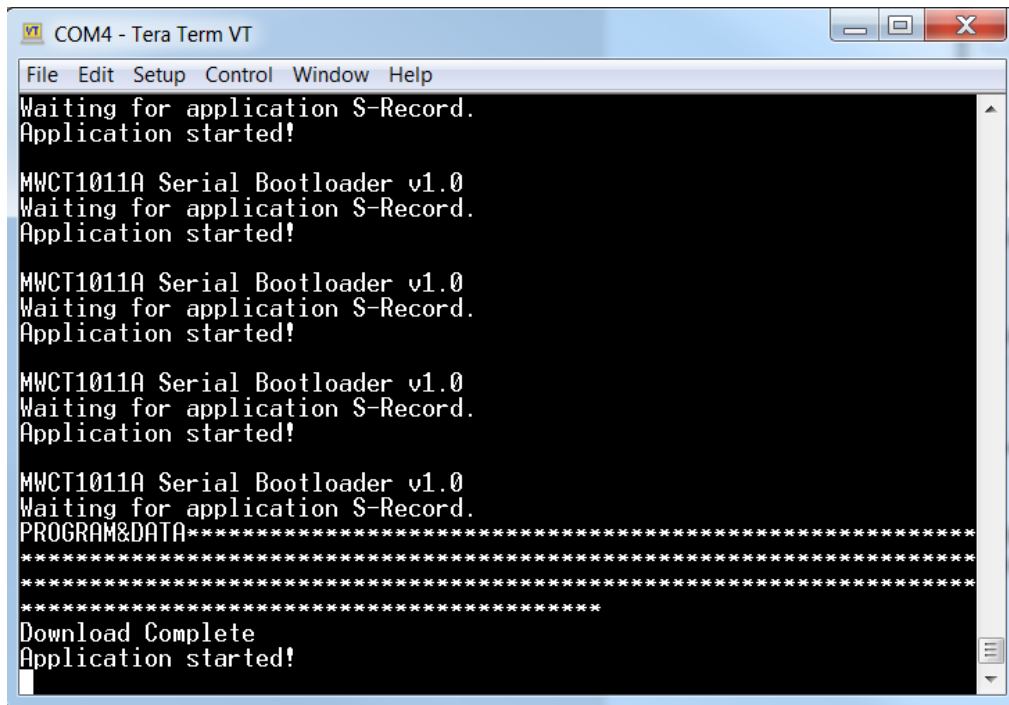


Figure 57. Download application S-Record file

7.6. Board calibration

NXP provides the FreeMASTER GUI tool for calibration and parameters tuning. For board calibration, see the *WCT1011A/WCT1013A Automotive MP-A9 V4.0 Run-Time Debugging User's Guide* (WCT101XAV40RTDUG).

8. Software Description

8.1. Software overview

8.1.1. Directory Structure

The following figure shows an example of the directory structure of the whole WCT1013A_MP-A9 distribution.

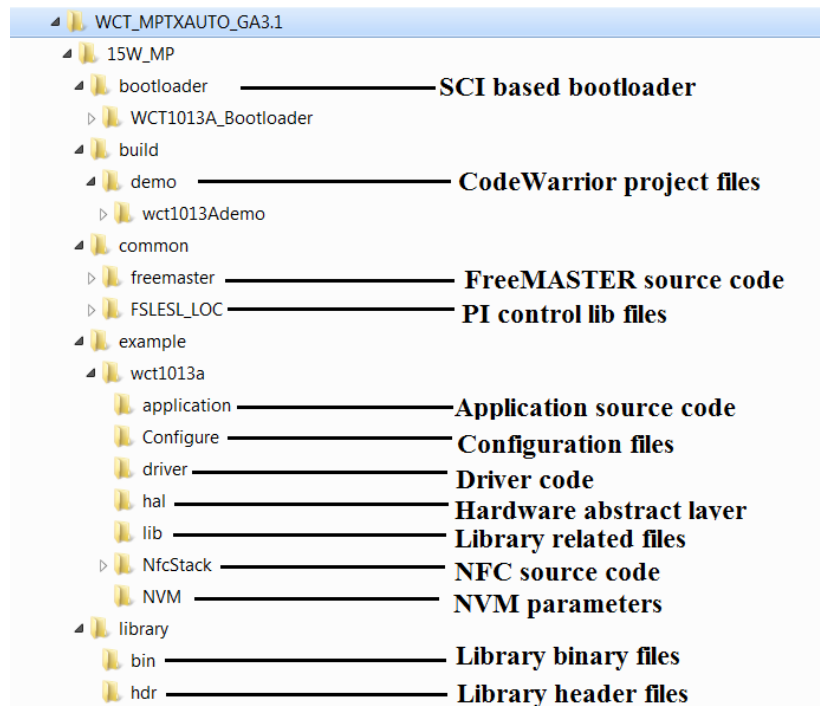


Figure 58. Directory structure of the whole WCT1013A_MP-A9 distribution

8.2. CodeWarrior Projects

There are four CodeWarrior projects in the package. The following figure shows all the four projects in CodeWarrior GUI when all of them are imported.

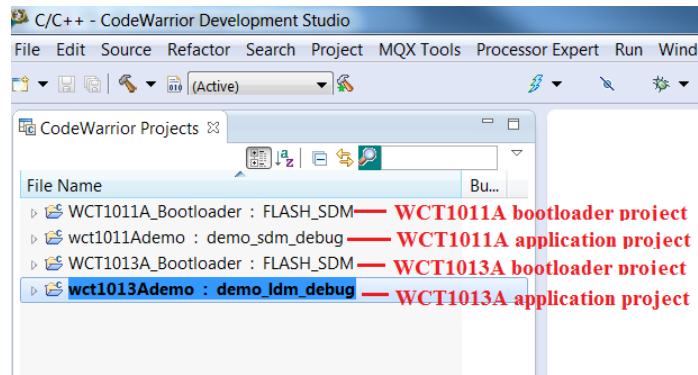


Figure 59. CW projects

Combined with different program models and different user cases, multiple build configurations are predefined in respective projects.

There are two program models provided for WCT parts.

- **Small Program Model:** The compiler generates a more efficient switch table, when the code is in the range 0x0-0xFFFF. This model is more efficient, but the code size is limited to 64 KB words.
- **Larger Program Model:** Extends DSP56800E addressing range by providing 24-bit address capability to instructions. That allows user accesses beyond the 64 KB word boundary of 16-bit addressing.

For WCT1011A, there are two build configurations:

- **demo_sdm_debug:** Small Program Model, including code for debugging.
- **demo_sdm_release:** Small Program Model, excluding debugging code to save memory size.

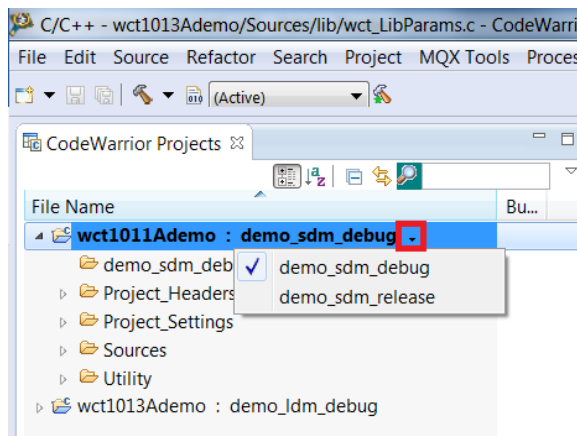


Figure 60. WCT1011A build configuration

For WCT1013A, there are three build configurations:

- **demo_ldm_debug:** Large Program Model, including code for debugging.
- **demo_ldm_release:** Large Program Model, excluding debugging code to save memory size.

- nfc_demo_ldm_debug: NFC dedicated build configuration. Large Program Model, including code for debugging.

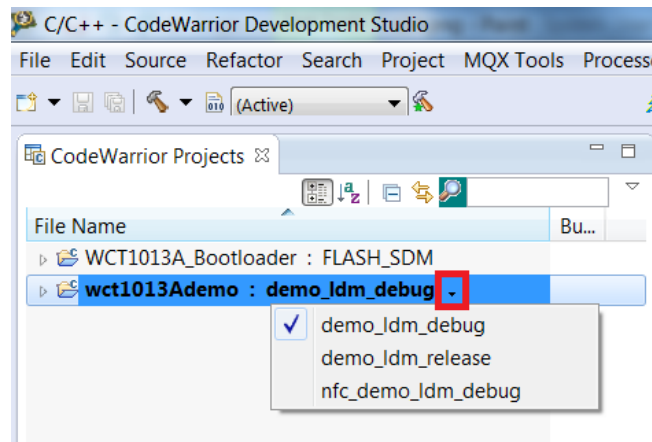


Figure 61. WCT1013A build configuration

8.3. Functional description

NXP provides full-featured wireless charging functions on the reference board. If a certain function is not needed, disable it by the definitions in the header file.

8.3.1. FreeMASTER

- FreeMASTER is supported. The following configuration is used to enable or disable it.

Table 2. FreeMASTER configurations

Configurations	Default value	Location	Description
FREEMASTER_SUPPORTED	TRUE	appcfg.h	Enables or disables the function. Set TRUE to enable it. Set FALSE to disable it.

- FreeMASTER communication interface configuration is in freemaster_cfg.h. SCI interface is enabled and JTAG interface is disabled by default. Only one interface can be set at a time.

```
#define FMSTR_USE_SCI      1    /* To select SCI communication interface */
#define FMSTR_USE_JTAG    0    /* 56F8xxx: use JTAG interface */
```

- FreeMASTER SCI port is SCI0 by default.

```
1) #define FMSTR_SCI_BASE      0xE080 /* base address of SCI_0 */
```

The macro is in freemaster_cfg.h.

```
2) #define QSCI_FREEMASTER_INDEX 0 //QSCI0
```

The macro is in qsci.h.

8.3.2. Low power mode

Low power mode is supported. In the analog ping interval, the MCU enters LPSTOP mode and DCDC module is closed. The following configuration is used to enable or disable it.

Table 3. Low power mode configurations

Configurations	Default value	Location	Description
LOW_POWER_MODE_ENABLE	FALSE	appcfg.h	Enables or disables the function. Set TRUE to enable it. Set FALSE to disable it.

8.3.3. Debug console

Debug console is supported. The following configuration is used to enable or disable it.

Table 4. Debug console configurations

Configurations	Default value	Location	Description
DEBUG_CONSOLE_SUPPORTED	FALSE	appcfg.h	Enables or disables the function. Set TRUE to enable it. Set FALSE to disable it.

For WCT1011A/WCT1013A digital buck-boost platform, only one SCI port is available. This port is default used for FreeMASTER. If debug console is used, disable FreeMASTER or change FreeMASTER communication interface to JTAG. The configurations are described in Chapter 6.

8.3.4. Bootloader

Bootloader is supported. The following configuration is used to enable or disable it.

Table 5. Bootloader configurations

Configurations	Default value	Location	Description
BOOTLOADER_USED	FALSE	appcfg.h	Enables or disables the function. Set TRUE to enable it. Set FALSE to disable it.

If bootloader is used, download the bootloader project to the board. Then change BOOTLOADER_USED to TRUE, rebuild application project, and download application .S file by bootloader.

8.3.5. DCDC control type

NXP provides two DCDC control types: digital buck-boost and analog buck-boost. The default one is digital buck-boost. The following configuration is used to switch the DCDC control type.

Table 6. DCDC control type configurations

Configurations	Default value	Location	Description
DIGITAL_BUCKBOOST	TRUE	appcfg.h	Switch DCDC control type. Set TRUE to enable digital buck-boost. Set FALSE to enable analog buck-boost.

8.3.6. Library functions

Some key functions are implemented in WPC library and can be enabled or disabled in application. The following configurations are used to enable or disable them.

Table 7. Library functions configurations

Function	Function description	Variables	Default value	Description
FOD	Foreign object detection based on power loss during power transfer state	gWCT_Params.uCtrlBit.bFODEnable	1	FOD feature enable or disable. 1: enable, 0: disable.
RX removal quick detection	RX removal quick detection based on current and voltage (~350ms) instead of communication timeout (1.5s)	gWCT_Params.uCtrlBit.bRRQDEnable	1	RX removal quick detection enable or disable. 1: enable, 0: disable.
Safe digital ping	Digital ping with lower coupling to avoid large current damage of big metal	gWCT_Params.uCtrlBit.bSafeDPEnable	1	Safe digital ping enable or disable. 1: enable, 0: disable.
Maximum voltage limit	Refer to MVL in WPC Qi specification	gWCT_Params.uCtrlBit.bMVLEnable	1	Maximum voltage limit enable or disable. 1: enable, 0: disable.
Fast charging	Enable fast charging for some types of phones, contact NXP for details	gWCT_Params.uCtrlBit.bFastChargingEnable	1	Fast charging enable or disable. 1: enable, 0: disable.
Analog Ping	Use several power pulses to detect object before digital ping	gWCT_Params.uCtrlBit.bAnalogPingDisable	0	Analog ping enable or disable. 0: enable, 1: disable.
Q factor recharge retry	Foreign object removal detection based on Q factor method during recharge retry state	gWCT_Params.uCtrlBit.bQfactorRetry	1	Using Q factor method for recharge retry state. 1: enable, 0: disable.
Maximum power limit	TX maximum output power limit	gWCT_Params.uCtrlBit.bMPLEnable	1	Maximum power limit enable or disable. 1: enable, 0: disable.
Active power protection	Refer to overvoltage protection in WPC Qi specification	gWCT_Params.uCtrlBit.bActivePowerProtectionEnable	1	Active power protection enable or disable. 1: enable, 0: disable
Low power mode for recharge retry state	Low power mode enable or disable when TX is under recharge retry state	gWCT_Params.uCtrlBit.bRechargeRetryLowPowerEnable	0	Low power mode enable or disable under recharge retry state. 1: enable, 0: disable
Power down for maximum rail voltage	Power down or keep maximum rail voltage when rail voltage exceed maximum	gWCT_Params.uCtrlBit.bPowerDownForMaxVrail	1	Power down or not when rail voltage exceed maximum value. 1: power down,

				0: keep maximum
Keyfob avoidance	Enable TX jump to other frequency instead of operation frequency to avoid electrical interference during key detection	gHAL_bDBGFobType	0	0: gHAL_bFobActive is level effectively, 1: gHAL_bFobActive is edge effectively.
		gHAL_bFobActive	0	0: Keyfob is not active, 1: Keyfob is active
		gWCT_Params.uCtrlBit.byKeyfobAvoidanceDisableCoil	0	Charging enable or disable during keyfob avoidance state. 0: enable charging, 1: disable charging

8.4. Protection mechanisms

The following table lists the protections that can be implemented.

Table 8. Protection mechanisms

Protection	Default limits	Variables	Description
Rail voltage	23000 mV	gPROT_Params.wMaxRailVol	Application implemented. If the rail voltage exceeds the limit, charging is turned off. The limit value can be changed by FreeMASTER GUI.
Battery voltage	Min: 8000 mV Max: 18000 mV Hysteresis: 1000 mV	gPROT_Params.wMinBatteryVol gPROT_Params.wMaxBatteryVol gPROT_Params.wBatteryHystVol	Application implemented. If the battery voltage exceeds the maximum, the WCT library stops. When the battery voltage is lower than the difference between maximum and hysteresis, WCT library starts. When the battery voltage is lower than the minimum, the WCT library stops. When the battery voltage is higher than the sum of the minimum and hysteresis, the WCT library starts. The minimum and maximum limit can be changed by FreeMASTER GUI. The hysteresis value can be changed in PROT_Init().
Input current	4000 mA	gPROT_Params.wMaxInputCurrent	Application implemented. If the input current exceeds the limit, charging is turned off. The limit value can be changed by FreeMASTER GUI.
Coil current	6500 mA	gPROT_Params.wMaxCoilCurrent	Application implemented. If the coil current exceeds the limit, charging is turned off. The limit value can be changed by FreeMASTER GUI.
Temperature	Max: 60°C Hysteresis: 10°C	gPROT_Params.swTemperatureThreshold gPROT_Params.wTemperatureHyst	Application implemented. If the temperature exceeds the maximum, the WCT library stops. When the temperature is lower than

			the difference between the maximum and hysteresis, the WCT library starts. The maximum limit can be changed by FreeMASTER GUI. The hysteresis value can be changed in PROT_Init().
Safe digital ping	200 mA	gPROT_Params.wSafeDigitalPingInputCurrentThreshold	Library implemented. If the input current sampled at the beginning of the digital ping exceeds the limit, digital ping stops. The limit value can be changed in PROT_Init().
FOD	BPP RX: 600 mW EPP RX: 1000 mW	gWCT_Params.wLPPowerLossThresholdInOperationMode gWCT_Params.wMPPowerLossThresholdInOperationMode	Library implemented. If the power loss exceeds the limit, charging is turned off for 5 minutes. The limit value can be changed by FreeMASTER GUI.

9. System Bring Up

9.1. Ping sequences

When low power mode is disabled and no receiver is placed on the charging surface, the ping sequence is as follows:

Digital ping appears at about every 5 seconds and the analog ping appears at about every 400 ms. There are 12 to 13 analog ping between two digital pings.

The following figures show the PWM waveforms of the ping sequence and ping patterns.

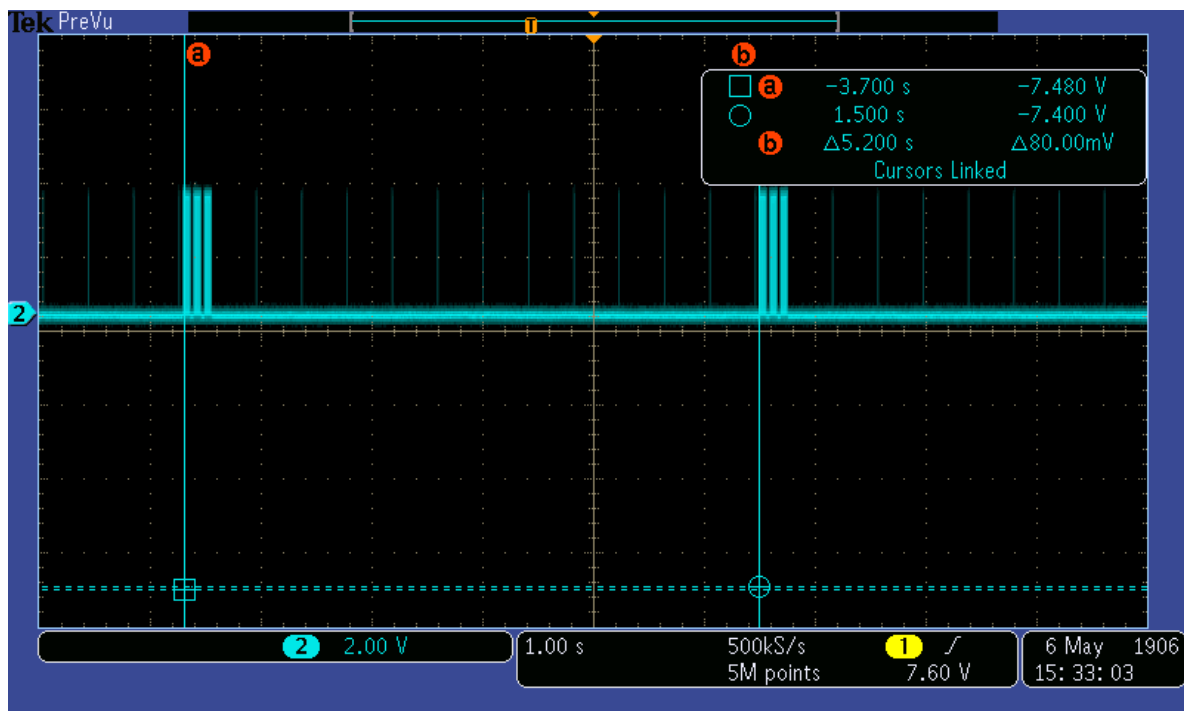


Figure 62. Digital ping interval

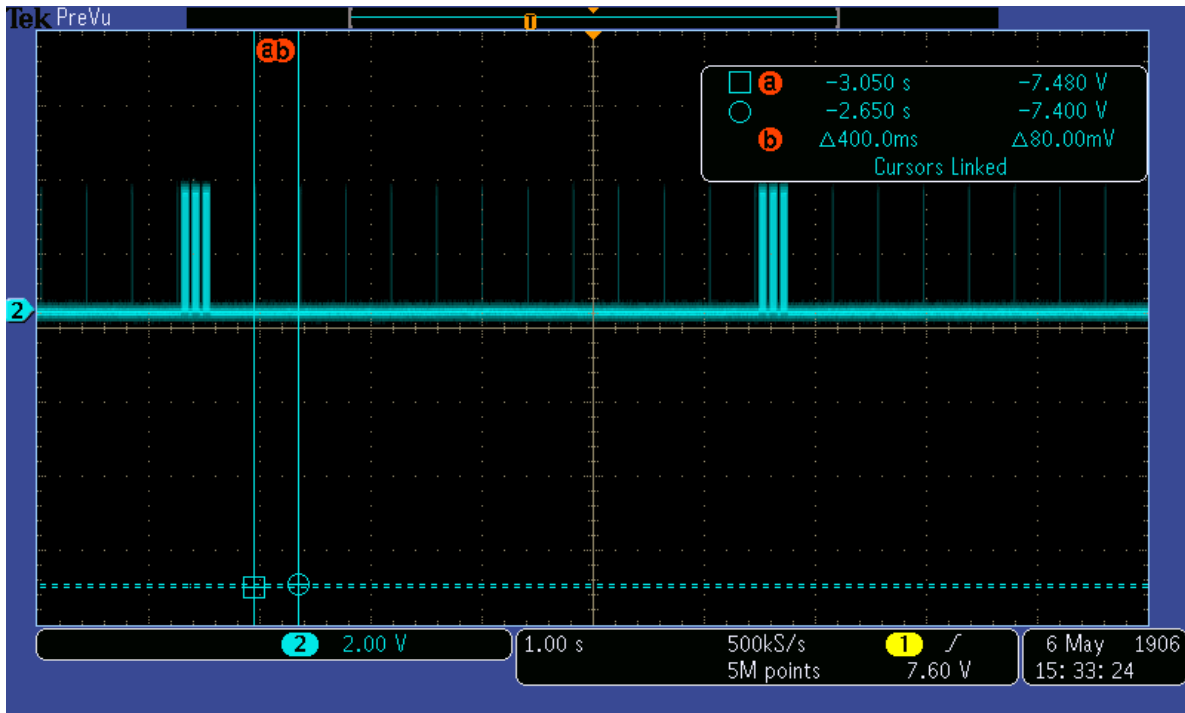


Figure 63. Analog ping interval

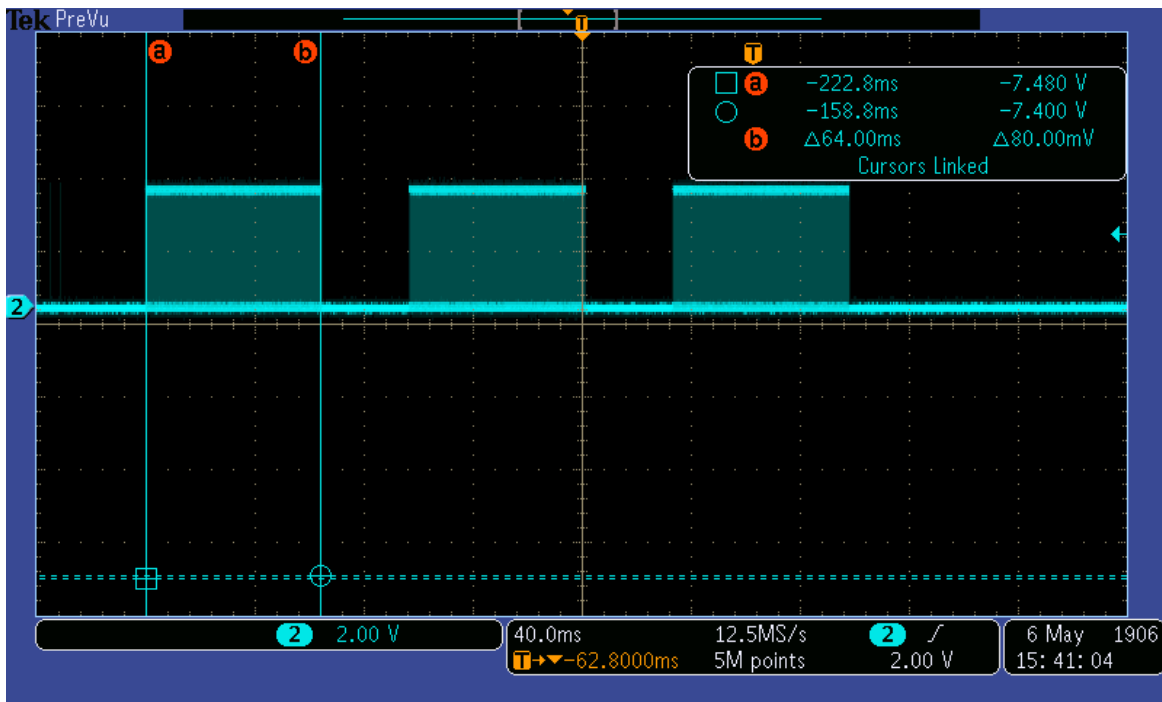


Figure 64. Digital ping pattern

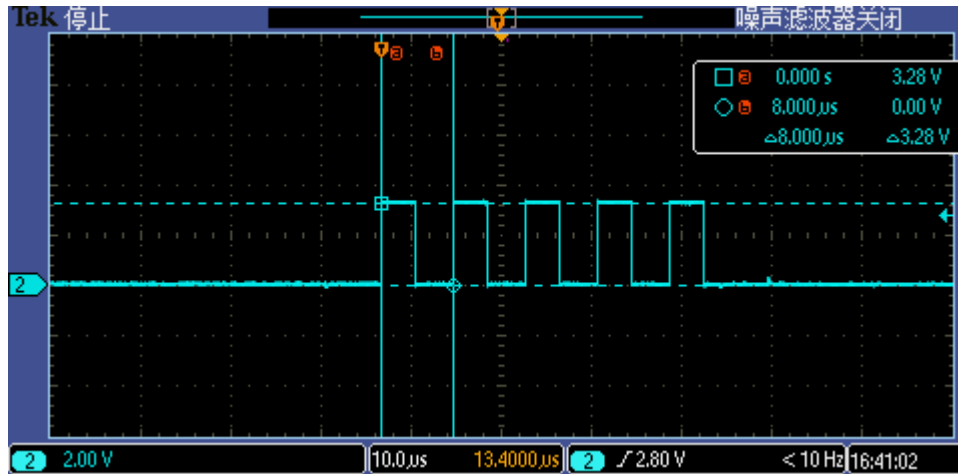


Figure 65. Analog ping pattern

9.2. LED indication

The default LED display modes for different TX working states are shown in the following table.

Table 9. LED display modes

LED No.	LED operational status					
	Standby	Charging	Charging complete	D fault	TX fault	RX fault
LED 1 (Red)	Off	Blink	Off	On	On	On
LED 2 (Green)	Blink	On	On	Off	Off	Off

The display pattern can be changed in `WCT_UpdateDevUsrIndication()`.

9.3. Debug messages

The system is able to print messages from a specified SCI port to inform users about what happened in the system. The messages help users to understand the system working procedure and debug the issues.

Message: ID, T/O

Print information when identification packet times out.

Message: EXT ID, T/O

Prints information when extended identification packet times out.

Message: CFG, HOLDOFF Invalid

Prints information when hold off time is out of range.

Message: CFG, Incorrect Count, Count1 (Count2)

Prints information when the amount of optional configuration packets received does not match the number in the configuration packet.

- *Count 1*: the number in configuration packet
- *Count 2*: TX received optional configuration packets amount

Message: CFG, Over CP

Prints information when the RX power is beyond capability.

Message: CFG, Rcvd 0xXX

Prints the packet type that should not be received in the configuration phase.

Message: CFG, T/O

Prints information when the configuration packet times out.

Message: XFER, INCOMP

Prints information when the RX version is not compatible.

Message: XFER, RCVPWR T/O

Prints information when receive power packet times out.

Message: XFER, PLoss T/O

Prints information when FOD happens.

Message: PROP, <Packet type>

Prints the proprietary packet header.

Message: XFER, Rcvd 0xXX Reset

Prints the packet type that is not received in the power transfer phase. Charging is reset.

Message: XFER, Rcvd 0xXX

Prints the packet type that is not defined in WPC specification.

10. Revision History

The following table provides the revision history.

Table 10. Revision history

Revision number	Date	Substantive changes
0	10/2017	Initial release
1	05/2018	Update according to WCT SW v4.0

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

NXP, the NXP logo, Freescale, the Freescale logo are trademarks of NXP B.V. All other product or service names are the property of their respective owners.

© 2018 NXP B.V.

Document Number: WCT101XAV40AUG
Rev. 4.0
05/2018