Application Report
SLOA210B–November 2014–Revised July 2017

Build Instructions for neard (Linux NFC) for AM335x + TRF7970A

Josh Wyatt, Erick Macias, and Tim Simerly — *Texas Instruments*
Mark Greer — *Animal Creek Technologies*

ABSTRACT

This document describes the process required to correctly build and test Linux NFC (neard) on AM335x (Cortex-A8) based BeagleBone + BeagleBone RF Cape + TRF7970ATB connected platform, using the Texas Instruments Sitara Software Development Kit (SDK) v3.03 and associated TI SDK file system in conjunction with the neard NFC stack.

Contents

List of Figures

1 Scope

This document describes how to update the Linux NFC Subsystem and TRF7970A driver, how to build the neard daemon, how to install those pieces onto a target file system, and how to test NFC functions using TI's Software Development Kit (SDK) v3.03.

AM335x SDK Product Download: http://www.ti.com/tool/processor-sdk-am335x

Processor SDK Linux Collateral:
http://processors.wiki.ti.com/index.php/Processor_SDK_Linux_Software_Developer's_Guide

Building the kernel, associated modules, and neard is done on a host machine (Linux PC) and the resulting output is copied to the respective directories on the target file system (SD Card).

NOTE: This build was performed and tested on a Linux PC with Ubuntu 16.04. Any older version of Ubuntu or other type of system is not supported.

2 Hardware

Figure 1 through Figure 3 show the hardware that is used with the provided software.

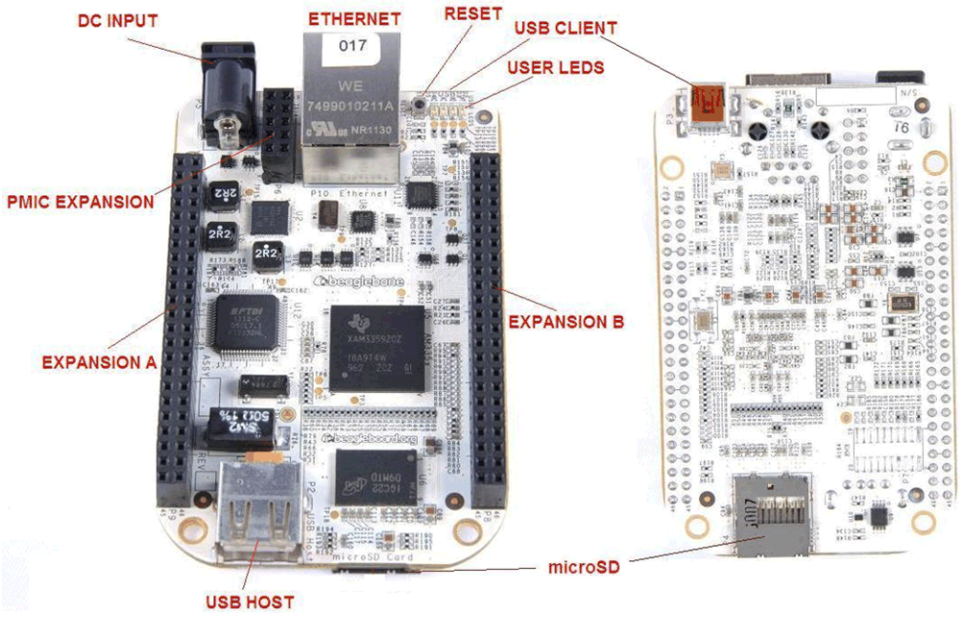

Figure 1. BeagleBone White Development Board

Sitara is a trademark of Texas Instruments.
ARM is a registered trademark of ARM Ltd.
All other trademarks are the property of their respective owners.

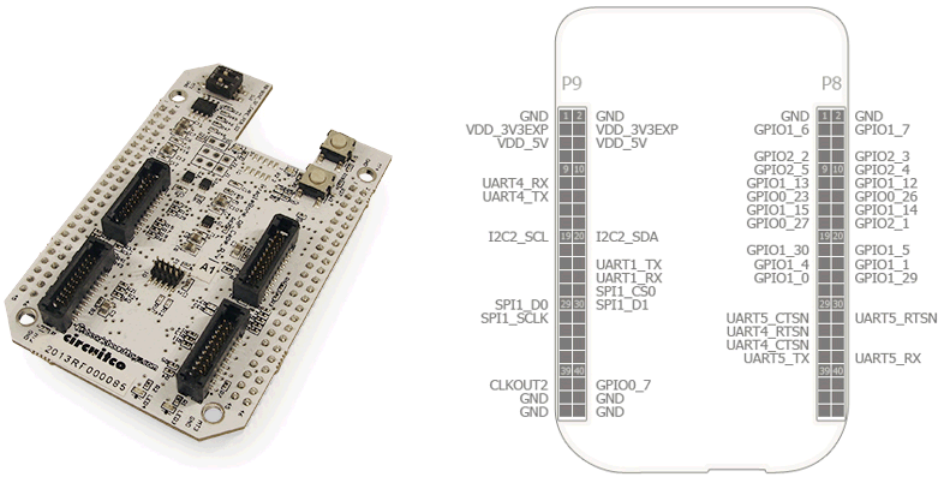

Figure 2. BeagleBone RF Cape Board and Pinout

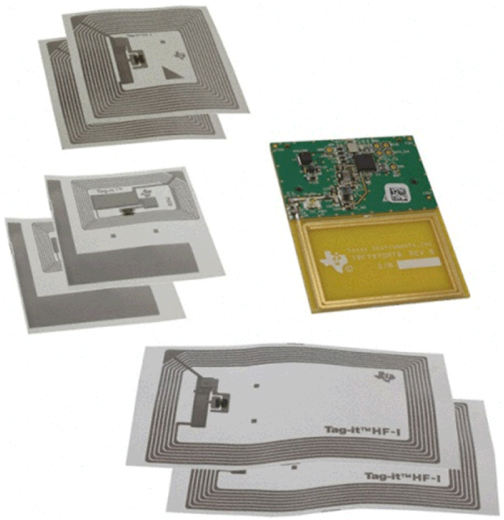

Figure 3. TRF7970ATB NFC/RFID Transceiver Module (With NFC/RFID Inlays)

Figure 4 shows the boards stacked together to create the complete hardware platform.

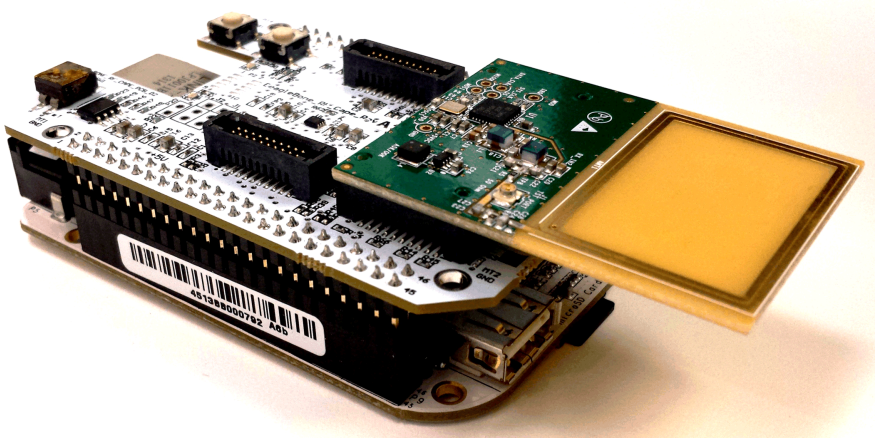

Figure 4. Connected Hardware Stack-Up

3 Installation of SDK for AM335x

To install the AM335x SDK onto a Linux PC, start with downloading the AM335X Linux SDK Essentials binary from TI: http://software-dl.ti.com/processor-sdk-linux/esd/AM335X/latest/index_FDS.html. Make sure to select the option for **PROCESSOR-SDK-LINUX-AM335x**

Follow the instructions in the Processor SDK Linux Getting Started Guide to install the SDK.

The recommended install location is the default location. For example, */home/user/ti-processor-sdk-linux-am335x-evm-03.03.00.04*

NOTE: This application report assumes that the SDK's root file system is installed on an SD Card.

After installation of the SDK is completed, the Linux Kernel Users Guide provides information on the steps required to build the Linux kernel for AM335x processors. It is recommended to go through these steps to ensure that the SDK install executed correctly on the Linux PC before trying to build the kernel for NFC.

After completing the installation of the Linux SDK and validating the kernel build, the Linux PC is now setup to build the Linux kernel with NFC.

4 Setting up an SD Card

This guide covers the build steps for the BeagleBone + BeagleBone RF Cape + TRF7970ATB connected platform. In order to run the Linux NFC stack on the BeagleBone, an SD card must be used. This section covers how to format an SD card for this purpose.

The steps required to build an SD card using the target file system provided with the SDK can be found at this reference: http://processors.wiki.ti.com/index.php/Processor_SDK_Linux_create_SD_card_script

Take the steps for "SD card using default images" which will install the pre-built SDK image that is required to run the Linux NFC stack.

5 Accessing neard Git Repositories

Before taking any further steps, acquire the latest neard support files from git repositories. It is recommended to execute these steps first to avoid any possible pathing issues due to subsequent steps.

1. Move to the main directory of the SDK
   (a) cd **<path to SDK>**
2. Clone the following git repository, which will download the appropriate neard source files:
   (a) git clone --branch sdk-3.3.0.4/updates https://github.com/animalcreek/neard.git
3. Move to the directory of the SDK where the source for the Linux kernel is installed
   (a) cd **<path to SDK>**/board-support/linux-4.4.41+gitAUTOINC+f9f6f0db2d-gf9f6f0db2d
4. Update the kernel source with the new source files that will allow it to build with NFC neard support by performing the following 3 git commands:
   (a) git remote add linux-trf7970a https://github.com/animalcreek/linux-trf7970a.git
   (b) git fetch linux-trf7970a sdk-3.3.0.4/updates
   (c) git checkout -b sdk-3.3.0.4/updates linux-trf7970a/sdk-3.3.0.4/updates

6 Building the Linux Kernel With NFC

Before starting the build steps, ensure the PATH environmental variable contains the location of the Arago cross-compile tools for the AM335x SDK that is being used:

export PATH=**/<absolute path to SDK>**/linux-devkit/sysroots/x86_64-arago-linux/usr/bin:$PATH

Perform the following steps to create a new kernel image with supporting modules:

NOTE: Some of the following steps may require *root* access on the Linux machine.

1. Verify that you are in the directory of the SDK where the source for the Linux kernel is installed which is **<path to SDK>**/board-support/linux-4.4.41+gitAUTOINC+f9f6f0db2d-gf9f6f0db2d
2. Build the Linux kernel with the new NFC files obtained from git by following these commands:
   (a) make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- distclean
   (b) make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- tisdk_am335x-bone-trf7970a_defconfig
   (c) make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- zImage
   (d) make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- modules
   (e) make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- am335x-bone-trf7970a.dtb
   (f) make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- am335x-boneblack-trf7970a.dtb

7 Steps for Moving Linux Build Files to the SD Card

With the Linux kernel now built, the next step is to move the newly generated files to the correct locations on the SD card.

1. Insert the SD Card into your Linux PC
2. Copy the newly built modules onto the SD Card:
   (a) sudo make ARCH=arm INSTALL_MOD_PATH=/media/**<userid>**/rootfs modules_install
3. Remove existing links to zImage and dtb files:
   (a) sudo rm /media/**<userid>**/rootfs/boot/zImage
   (b) sudo rm /media/**<userid>**/rootfs/boot/am335x-bone.dtb
   (c) sudo rm /media/**<userid>**/rootfs/boot/am335x-boneblack.dtb
4. Copy the kernel and .dtb files to the SD card.
   (a) sudo cp arch/arm/boot/zImage /media/**<userid>**/rootfs/boot/zImage-trf7970a
   (b) sudo cp arch/arm/boot/dts/am335x-bone-trf7970a.dtb /media/ **<userid>** /rootfs/boot/devicetree-zImage-am335x-bone-trf7970a.dtb
   (c) sudo cp arch/arm/boot/dts/am335x-boneblack-trf7970a.dtb /media/**<userid>**/rootfs/boot/devicetree-zImage-am335x-boneblack-trf7970a.dtb
5. Link the new files so they are used when booting:
   (a) sudo ln -s zImage-trf7970a /media/**<userid**>/rootfs/boot/zImage
   (b) sudo ln -s devicetree-zImage-am335x-bone-trf7970a.dtb /media/**<userid>**/rootfs/boot/am335x-bone.dtb
   (c) sudo ln -s devicetree-zImage-am335x-boneblack-trf7970a.dtb /media/**<userid>**/rootfs/boot/am335x-boneblack.dtb

8 Building the neard Daemon

To build the neard daemon, follow the steps below:

1. Setup the environment prior to building the neard daemon:
   (a) cd **<path to SDK>**
   (b) source linux-devkit/environment-setup
2. Move to the directory where the neard source files were added from git earlier:
   (a) cd neard
3. Build the neard daemon:
   (a) ./bootstrap-configure (this sets up the make files)
   (b) make CFLAGS+=-Wno-cast-align (this builds neard)

NOTE: Make sure that libtool is installed (that is, sudo apt-get install libtool).

4. Copy the newly build neard files to the target file system:
   (a) sudo cp src/org.neard.conf /media/**<userid>**/rootfs/etc/dbus-1/system.d/
   (b) sudo mkdir -p /media/**<userid>**/rootfs/usr/local/bin
   (c) sudo cp src/neard /media/**<userid>**/rootfs/usr/local/bin
   (d) sudo cp -a test /media/**<userid>**/rootfs/home/root
5. Unmount the boot and rootfs partitions from the SD card:
   (a) sudo umount /media/**<userid>**/boot
   (b) sudo umount /media/**<userid>**/rootfs

9 Testing NFC/RFID Tag Functionality Using neard

To interact (read or write) with NFC tag platforms from the target file system, boot from the SD card and place or present an NFC tag platform to the antenna on the TRF7970ATB board.

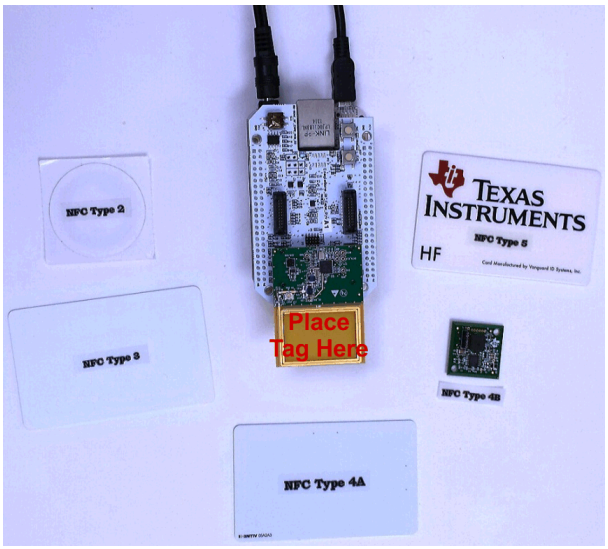

Figure 5. Supported NFC Forum Tag Platforms with Target Hardware

Login as 'root' and start the neard daemon:

1. neard

From the home directory on the target, the following commands are to be done in the order listed. Each time a tag is presented, step 2 must precede the list or dump commands. Otherwise, the transmitter will not be on to power the passive tag.

1. test/test-adapter powered nfc0 on
2. test/test-adapter poll nfc0 on Initiator
3. test/test-tag list
4. test/test-tag dump

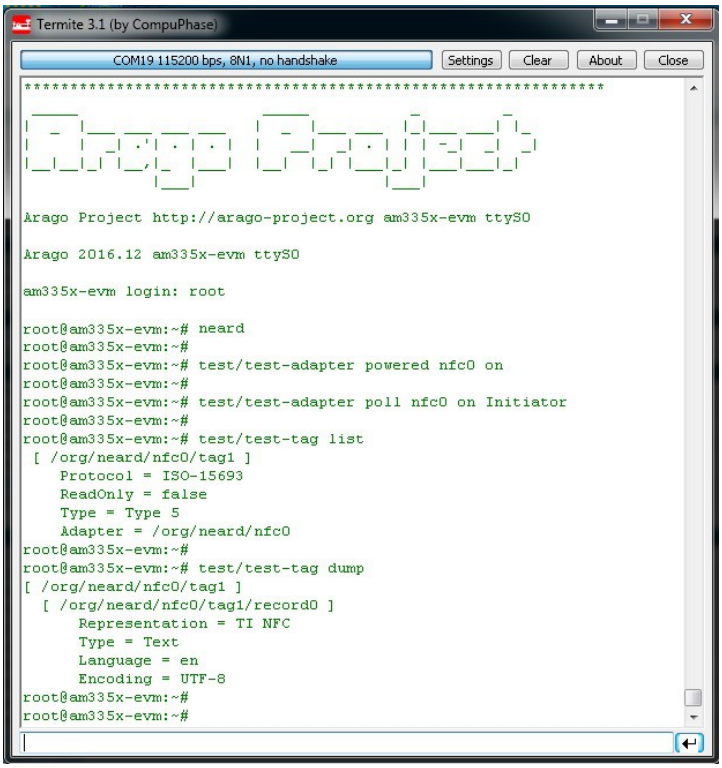

Figure 6. Configuring TRF7970A, Entering Polling Loop for NFC Tags, and Detecting an NFC Tag

NOTE: Make sure that you leave the NFC card over the NFC antenna on the TRF7970ATB during these steps. Otherwise you will only get a command prompt back after typing each of the "test/test-tag" attempts.

9.1 Testing NFC/RFID Tag Reading Using neard

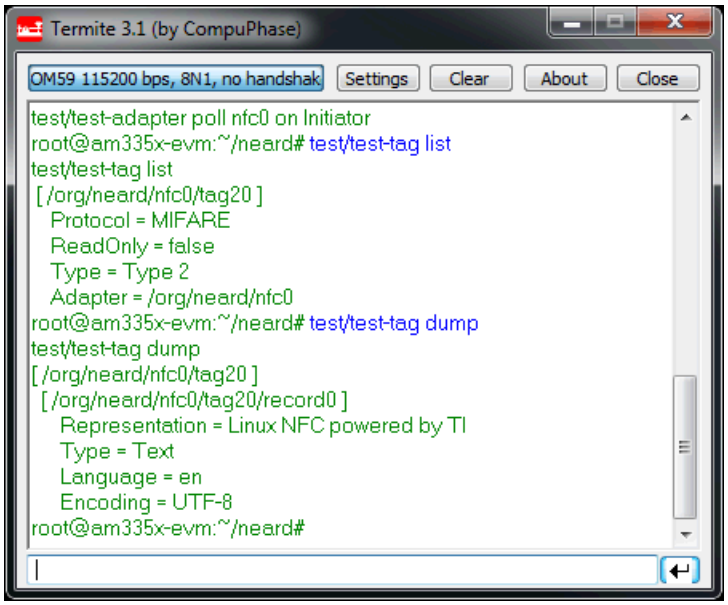

Figure 7. Reading NFC Type 2 Tag Platform (List and Dump)

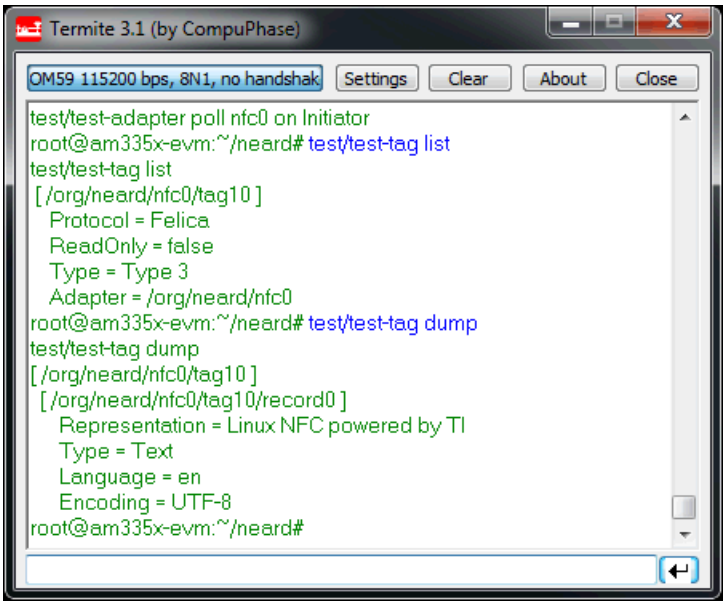

Figure 8. Reading NFC Type 3 Tag Platform (List and Dump)

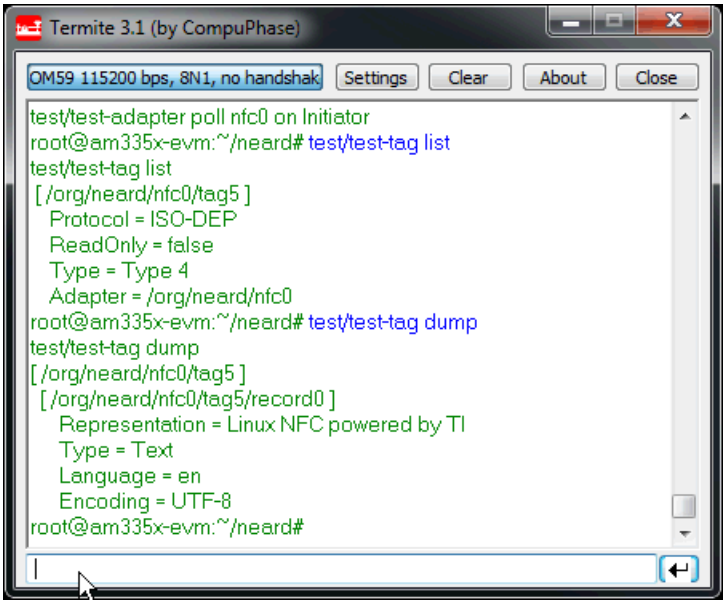

Figure 9. Reading NFC Type 4A Tag Platform (List and Dump)

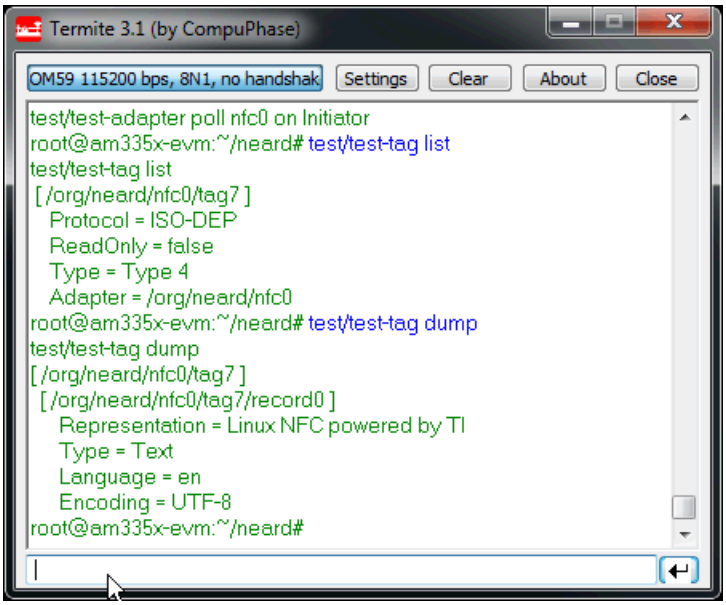

Figure 10. Reading NFC Type 4B Tag Platform (List and Dump)

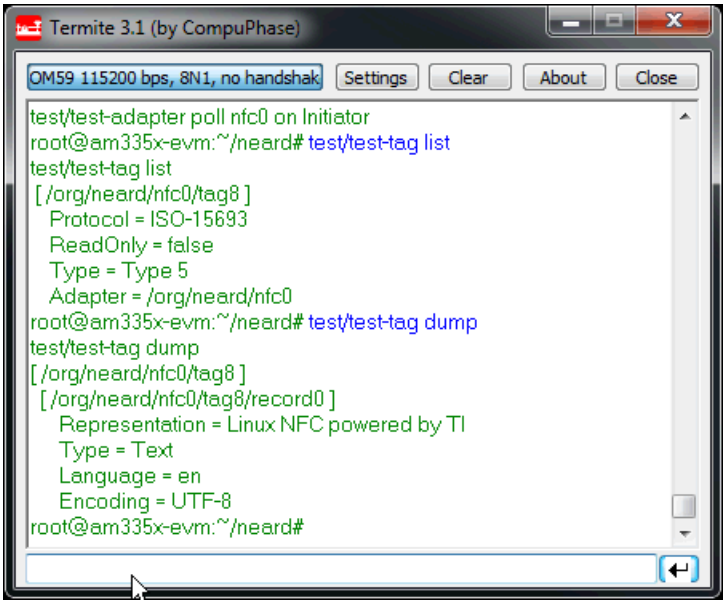

Figure 11. Reading NFC Type 5 Tag Platform (List and Dump)

9.2 Testing NFC/RFID Tag Writing Using neard

When writing tags, the format shown in Figure 12 must be followed. This menu is always available for reference by typing "test/test-tag"

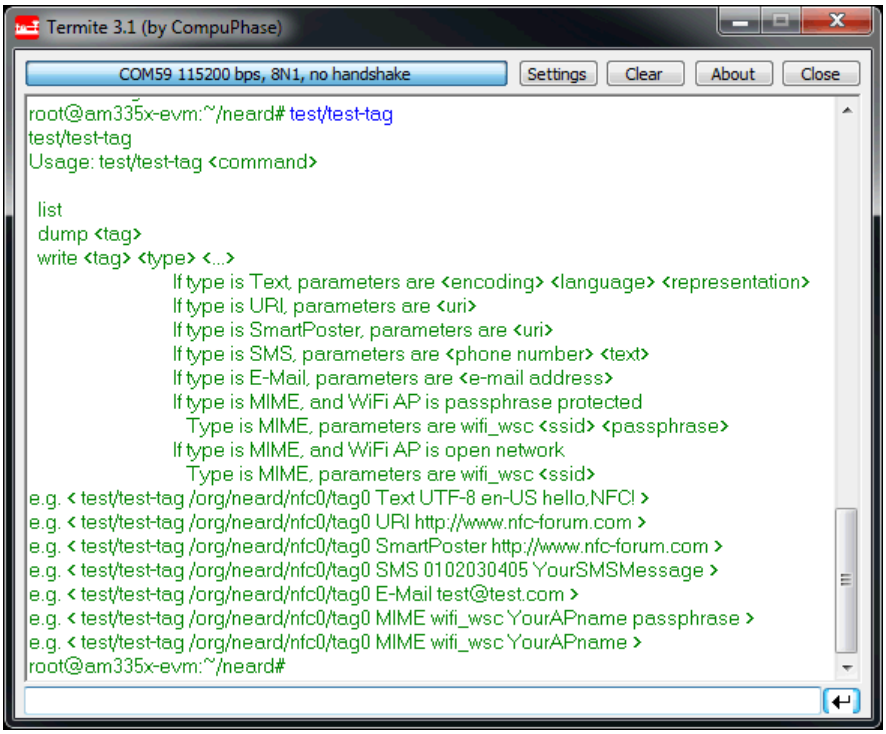

Figure 12. test/test-tag Help Screen

Figure 13 shows process of writing a URI RTD. Note that polling loop was started and a tag dump was done to get the tag number (in this case tag21) which is needed by the tag write command.

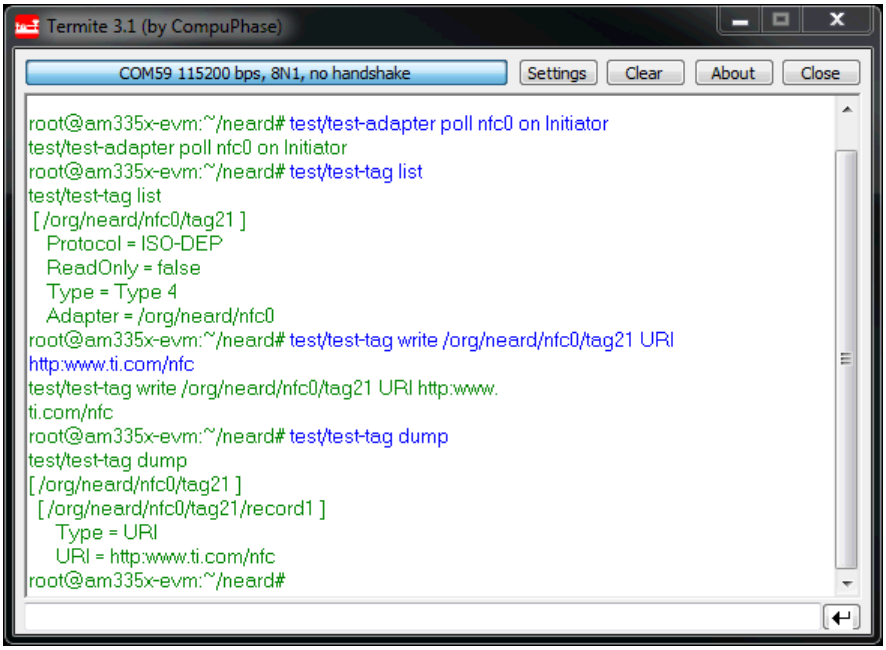

Figure 13. Writing a URI RTD to Tag and Reading Back

10 Porting Notes

The TRF7970A driver has been tested on an AM335x based BeagleBone White board + an RF Cape + a TRF7970ATB. Here are some notes to help you get the TRF7970A working on a different set of hardware.

The processor side communicates with the TRF7970A using SPI with Slave Select, so to be able to communicate with the device; your hardware must have an SPI interface and a working SPI driver. Be sure that the SPI and TRF7970A CONFIG_ options are enabled before building the kernel.

The TRF7970A driver uses Device Tree (DT) properties to get the information it needs to function. There is some documentation in Documentation/devicetree/bindings/net/nfc/trf7970a.txt (in the kernel source) to help you set it up.

This snippet example is used for the BeagleBone platform:

```
&spi1 {
        status = "okay";
        trf7970a@0 {
              compatible = "ti,trf7970a";
              reg = <0>;
              pinctrl-names = "default";
              pinctrl-0 = <&trf7970a_default>;
              spi-max-frequency = <2000000>;
              interrupt-parent = <&gpio0>;
              interrupts = <31 0>;
              ti,enable-gpios = <&gpio2 3 GPIO_ACTIVE_LOW>,
                                <&gpio2 4 GPIO_ACTIVE_LOW>;
              vin-supply = <&trf7970atb_reg>;
              autosuspend-delay = <30000>;
              irq-status-read-quirk;
              en2-rf-quirk;
              status = "okay";
      };
};
```

The TRF7970A requires the specification of both irq-status-read-quirk and en2-rf-quirk. The TRF7970A driver expects your platform interrupt and regulator information to be set up correctly. If your platform has a pin multiplexer (pinmux), then it must also be set up correctly.

11 References

1. AM3358 Sitara™ ARM® Cortex-A8 Microprocessor
2. Processor SDK for AM335x Sitara™ Processors
3. *TRF7970A Multiprotocol Fully Integrated 13.56-MHz RFID and NFC Transceiver IC*
4. TRF7970ATB Evaluation Module
5. Linux NFC Project
6. NFC Forum

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from June 9, 2017 to July 11, 2017	Page
• Removed "t5t-rmb-extra-byte-quirk" from the code example and description in Section 10, *Porting Notes*	12

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265