

1 Introduction

The LS1028ARDB board provides two mikroBUS™ connectors, mikroBUS1 and mikroBUS2. These mikroBUS connectors support different types of click boards that can be accessed through the SPI3, UART2, PWM, or I2C interface.

This document provides the detail of enabling NFC mikroBUS™ click board on the LS1028ARDB board.

1.1 Purpose

It is intended for engineers who have to download and test NFC feature on the LS1028ARDB board. The reader is assumed to have basic knowledge about U-Boot/Linux and NXP LS1028ARDB background.

The document covers:

- Procedure to set up mikroBUS click board on LS1028ARDB
- Procedure to test NFC feature

1.2 Related documentation

The following is a list of additional documents that you can refer to for more information on the LS1028ARDB board.

- [QorIQ LS1028A Reference Design Board Getting Started Guide](#)
- [QorIQ LS1028A Reference Design Board Reference Manual](#)
- [Layerscape LS1028A BSP User Guide](#)

1.3 Scope

The scope of this document is to provide instructions on how to enable mikroBUS click board on the LS1028ARDB board.

2 Set up mikroBUS click board on LS1028ARDB

The prerequisites to set up the click board are as follows:

- NFC tag
- LS1028ARDB board
- Click board

The following figure shows the NFC tag and mikroBUS click board.

Contents

1 Introduction.....	1
2 Set up mikroBUS click board on LS1028ARDB.....	1
3 How to test NFC feature.....	4
A Revision History.....	6

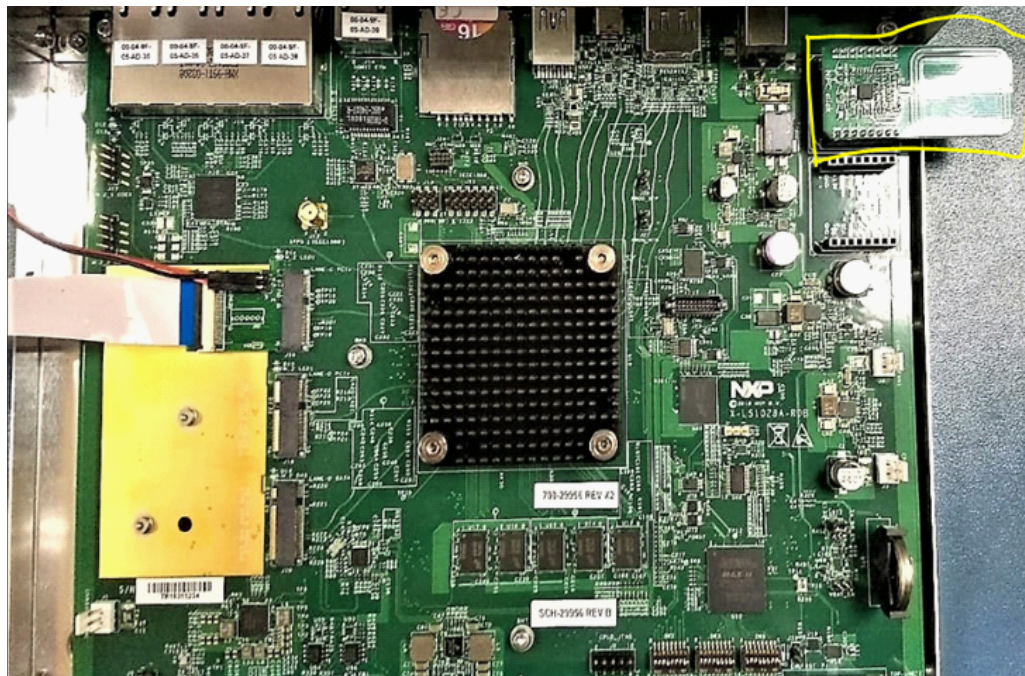




Figure 1. NFC tag and mikroBUS click board

The following steps describe how to set up the mikroBUS click board on the LS1028ARDB board.

1. Plug the click board into the LS1028ARDB board as shown below:



NOTE

The pins of mikroBUS click board should be plugged into mikroBUS click header with care, otherwise, the click board can be damaged.

2. Linux compilation: Get the NFC card reader driver code from open source and build it in the kernel as module.

- a.
- [Click here](#)
- and clone Linux driver repository as follows:

```
$ cd drivers/misc
$ git clone https://github.com/NXPnFCLinux/nxp-pn5xx.git
```

- b. Download the extra patch from open source (
- [open source link](#)
-). Apply this patch as follows:

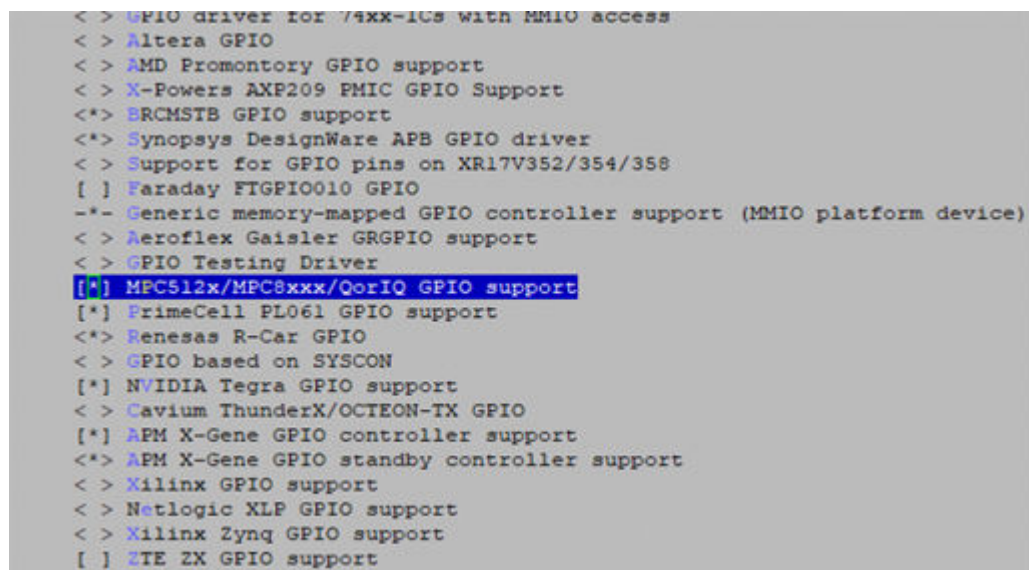
```
cd <linux_kernel_source_code_dir>
git am 0001-Enable-nfc-for-1028.patch
```

- c. Set up environment as follows:

```
$ export CROSS_COMPILE="aarch64-linux-gnu-";export ARCH="arm64"
```

- d. Enable CONFIG_GPIO_MPC8XXX option through menuconfig as follows:

```
./scripts/kconfig/merge_config.sh arch/arm64/configs/defconfig arch/arm64/configs/
lsdk.config
$make menuconfig
```



```
< > GPIO driver for 74xx-ICs with MMIO access
< > Altera GPIO
< > AMD Promontory GPIO support
< > X-Powers AXP209 PMIC GPIO Support
[*] BRCMSTB GPIO support
[*] Synopsys DesignWare APB GPIO driver
< > Support for GPIO pins on XR17V352/354/358
[ ] Paraday FTGPIO010 GPIO
-* Generic memory-mapped GPIO controller support (MMIO platform device)
< > Aeroflex Gaisler GRGPIO support
< > GPIO Testing Driver
[*] MPC512x/MPC8xxx/QorIQ GPIO support
[*] PrimeCell PL061 GPIO support
[*] Renesas R-Car GPIO
< > GPIO based on SYSCON
[*] NVIDIA Tegra GPIO support
< > Cavium ThunderX/OCTEON-TX GPIO
[*] APM X-Gene GPIO controller support
[*] APM X-Gene GPIO standby controller support
< > Xilinx GPIO support
< > Netlogic XLP GPIO support
< > Xilinx Zynq GPIO support
[ ] ZTE ZX GPIO support
```

```
$make -j4
```

Now, you can get PN7120 controller driver module **pn5xx_i2c.ko** from the `drivers/misc/nxp-pn5xx` directory.

- e. Generate
- `kernel-ls1028a-rdb.itb`
- as follows:

```
$mkimage -f kernel-ls1028a-rdb.its kernel-ls1028a-rdb.itb
```

3. RCW: Create RCW binary using RCW available at the following link: [RCW for LS1028ARDB](#).4. U-Boot: Find source code for U-Boot from the following link: [U-Boot source code](#).

Compile U-Boot with the following command:

```
$ make dictclean; make ls1028ardb_defconfig;make -j4
```

5. NFC lib: Port the Linux **libnfc-nci** stack as well as **nfcDemoApp** on the LS1028ARDB board. Both of these can be obtained from the following link: https://github.com/NXPnfcLinux/linux_libnfc-nci.

The following steps define how to build the demo and stack:

- a. Get the source code:

```
$ git clone https://github.com/NXPnfcLinux/linux_libnfc-nci.git
```

- b. Modify Makefile.am in top folder as follows:

```
--- a/Makefile.am
+++ b/Makefile.am

-nfcDemoApp_LDFLAGS = -pthread -ldl -lrt -lnfc_nci_linux
+nfcDemoApp_LDFLAGS = -pthread -ldl -lrt -lnfc_nci_linux -lstdc++
```

- c. Set up environment as follows:

```
$export ARCH=arm64
$export CROSS_COMPILE=/opt/gcc-linaro-aarch64-linux-gnu-4.8-2013.12_linux/bin/aarch64-
linux-gnu-
$export CROSS_PATH=/opt/gcc-linaro-aarch64-linux-gnu-4.8-2013.12_linux/bin
$ export PATH=${CROSS_PATH}:$PATH
$ export CC=${CROSS_COMPILE}gcc
$ export CXX=${CROSS_COMPILE}g++
$ export LD=${CROSS_COMPILE}ld
$ export AR=${CROSS_COMPILE}ar
$ export RANLIB=${CROSS_COMPILE}ranlib
$ export ac_cv_func_malloc_0_nonnull=yes
```

- d. Generate the configuration script by executing the bootstrap bash script:

```
$/bootstrap
```

- e. Call the newly created configure script to enable the generation of the Makefile recipe file:

```
$/configure --build=x86_64-unknown-linux-gnu --host=aarch64-linux-gnu --target=aarch64-
linux-gnu
$ make
$ make install
```

- f. Get demo application, configure file and stack library in below path:

```
./conf/libnfc-nci.conf
./conf/libnfc-nxp-init.conf
.libs/nfcDemoApp
.libs/libnfc_nci_linux*
libstdc++.so.6 /*It can be obtained from the tool chain path, such as "/opt/gcc-linaro-
aarch64-linux-gnu-4.8-2013.12_linux/aarch64-linux-gnu/lib/libstdc", you can get the
toolchain path by command as "aarch64-linux-gnu-gcc -print-file-name=libstdc++.so.6" */
```

3 How to test NFC feature

The following steps describe how to test NFC feature on the LS1028ARDB board.

1. Power on the board.

2. Stop at U-Boot prompt and execute the following command:

```
=> i2c mw 0x77 0x0 0x40; i2c mw 0x66 0x55 7
```

Command notes:

- `i2c mw 0x77 0x0 0x40` command enables I2C1 channel 6 of PCA9848 I2C multiplexer
 - `i2c mw 0x66 0x55 0x7` command sets RST (as output, asserted high) and INT (active-high input) signal of CPLD
3. Boot to Linux prompt.
 4. Execute the following command to place and configure the NFC Lib file in the correct path.

```
$ cp libnfc-nci.conf libnfc-nxp-init.conf /etc/
$ cp libnfc_nci* /usr/lib/
$ cp libstdc++.so.6 /usr/lib/
$ export LD_LIBRARY_PATH=/
```

5. InsmoD PN7120 controller driver as follows:

```
$ insmod ./pn5xx_i2c.ko

root@localhost # insmod ./pn5xx_i2c.ko
pn54x_dev_init
pn54x_probe
pn544 7-0028: FIRM GPIO <OPTIONAL> error getting from OF node
pn544 7-0028: CLKREQ GPIO <OPTIONAL> error getting from OF node
pn544 7-0028: 7-0028 supply nxp,pn54x-pvdd not found, using dummy regulator
pn544 7-0028: 7-0028 supply nxp,pn54x-vbat not found, using dummy regulator
pn544 7-0028: 7-0028 supply nxp,pn54x-pmuvecc not found, using dummy regulator
pn544 7-0028: 7-0028 supply nxp,pn54x-sevdd not found, using dummy regulator
pn54x_probe: request irq_gpio 481
pn54x_probe : requesting IRQ 123
```

6. Run nfcDemo App

```
$. /nfcDemoApp poll
```

7. Bring the NFC card near to the mikroBUS click board. Upon detection of the card, the tag is read and the following print is displayed.

```
root@localhost:~# nfcDemoApp poll
#####
## NFC demo ##
#####
## Poll mode activated ##
#####
... press enter to quit ...
Waiting for a Tag/Device...
NFC Tag Found
Type : 'Type A - Mifare U1'
NFCID1 : '04 4D 56 D2 9C 39 81 '
Record Found :
NDEF Content Max size : '868 bytes'
NDEF Actual Content size : '146 bytes'
ReadOnly : 'FALSE'
TNF Media
```

```

Type : text/vcard
Data : BEGIN:VCARD
VERSION:2.1
N:NXP;Sudhanshu;;;
FN:Sudhanshu NXP
TEL;X-Mobile:+91 88005 88188
TEL;X-Mobile:+918800588188
END:VCARD
146 bytes of NDEF data received :
D2 0A 85 74 65 78 74 2F 76 63 61 72 64 42 45 47 49 4E 3A 56 43 41 52 44 0D 0A 56
45 52 53 49
4F 4E 3A 32 2E 31 0D 0A 4E 3A 4E 58 50 3B 53 75 64 68 61 6E 73 68 75 3B 3B 3B 0D
0A 46 4E
3A 53 75 64 68 61 6E 73 68 75 20 4E 58 50 0D 0A 54 45 4C 3B 58 2D 4D 6F 62 69 6C
65 3A 2B
39 31 20 38 38 30 30 35 20 38 38 31 38 38 0D 0A 54 45 4C 3B 58 2D 4D 6F 62 69 6C
65 3A 2B
39 31 38 38 30 30 35 38 38 31 38 38 0D 0A 45 4E 44 3A 56 43 41 52 44 0D 0A
NFC Tag Lost
Waiting for a Tag/Device...

```

NOTE

The detection of spurious interrupts is a known issue, which will be resolved in the next release.

A Revision History

The table below summarizes the revisions to this document.

Table 1. Revision history

Revision	Date	Topic cross-reference	Change description
Rev. 0	06/2019	-	Initial public release

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.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, Freescale, the Freescale logo, and QorIQ are trademarks of NXP B.V. All other product or service names are the property of their respective owners. Arm and Cortex are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets.

© NXP B.V. 2019.

All rights reserved.

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 06/2019

Document identifier: AN12354