



How to port the NFC Reader Library to K64F

Jordi Jofre (Speaker)

Angela Gemio (Host)

Webinar instructions

Audio settings:

- You are in “listen only” mode due to possible background noise
- Set **mic & speakers** option (headset + external mic advised)

For questions:

- Write them in section “Questions” of your GoToWebinar panel
- The host will receive and compile them for the Q&A time

Materials will be made available to you

The session will be recorded for on-demand viewing

Please answer the **evaluation** questions after session.

Scheduled sessions for 11/10/2017

10:00 am - 10:30 am CET

05:00 pm - 05:30 pm CET



HOW TO PORT THE NFC READER LIBRARY TO K64F

WEBINAR SERIES: NFC SOFTWARE INTEGRATION

JORDI JOFRE
NFC READERS
NFC EVERYWHERE
11/10/2017



PUBLIC



SECURE CONNECTIONS
FOR A SMARTER WORLD

Agenda

NFC software integration webinar series

Session I, [14th September](#)

How to integrate NFC frontends in Linux.

Session II, [28th September](#)

How to integrate NFC controllers in Linux.

Session III, [11th October](#)

How to port the NFC Reader Library to K64F.



Agenda

NFC software integration webinar series

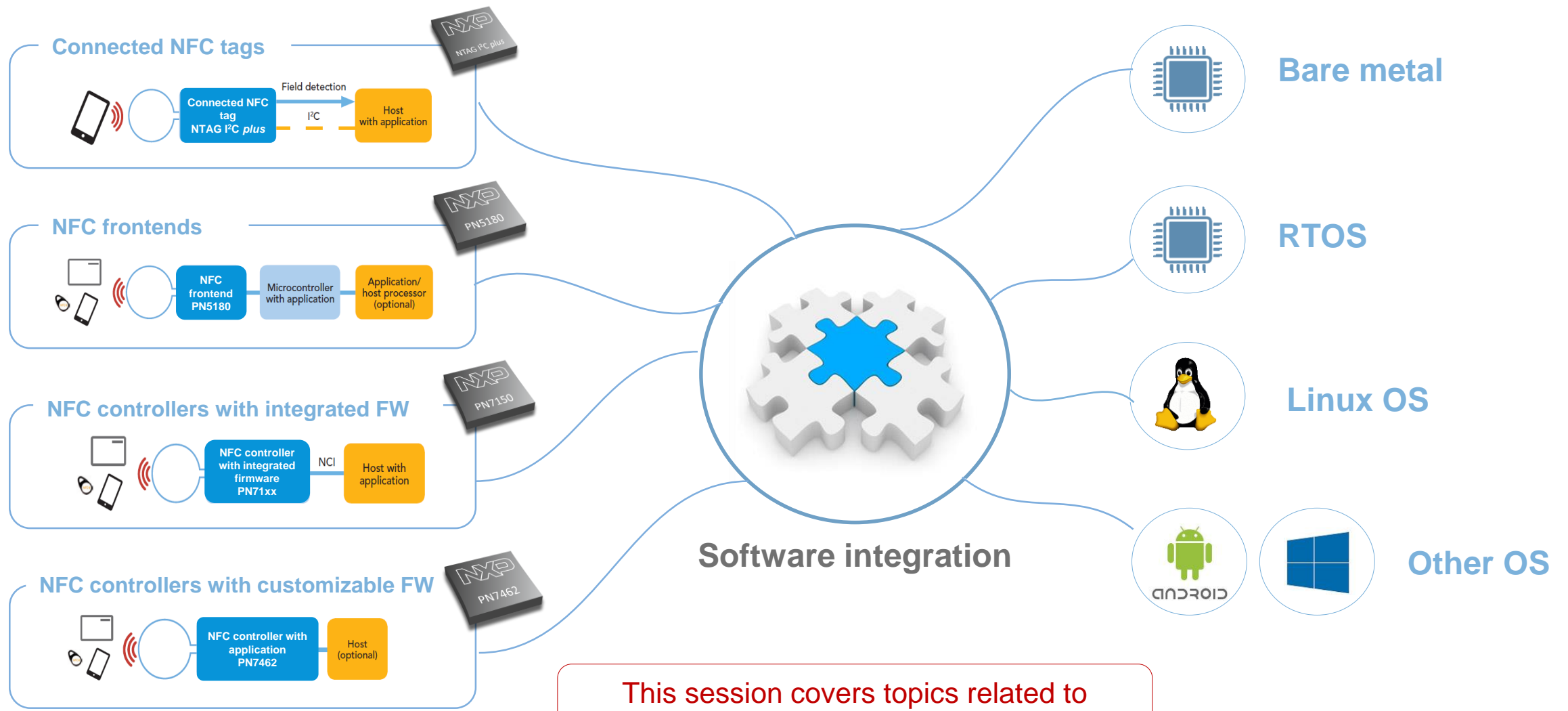
Session III, [11th October](#)

How to port the NFC Reader Library to K64F

- ▶ NFC Reader Library positioning and architecture
- ▶ **Hands-on:** NFC Reader Library porting to K64F.
- ▶ General considerations to port NFC Reader Library to your target MCU.
- ▶ Q&A

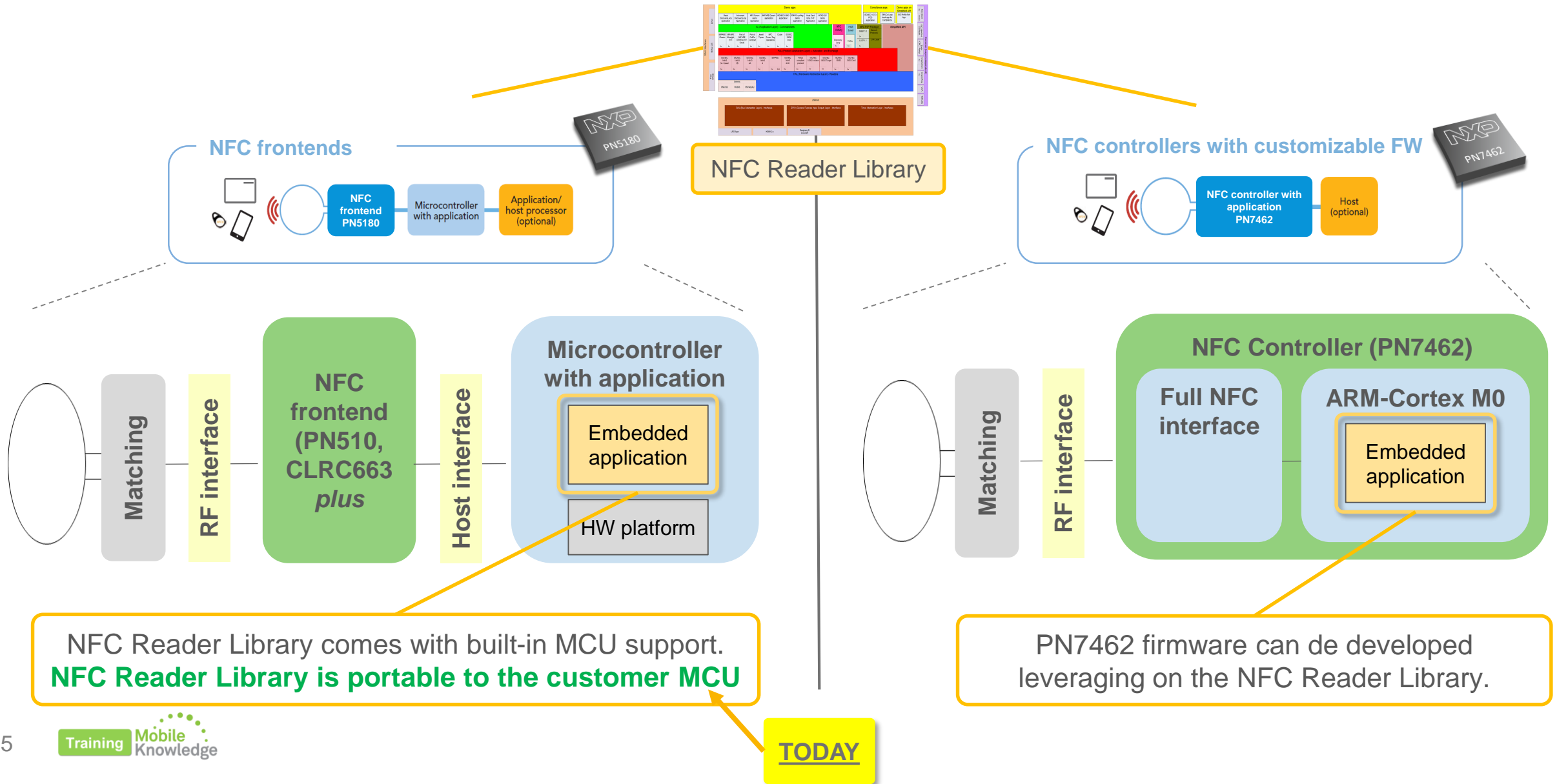


NXP software support for NFC integration in any platform



This session covers topics related to NFC reader software integration in bare metal and RTOS systems.

NFC Reader Library integration for RTOS and non-RTOS systems

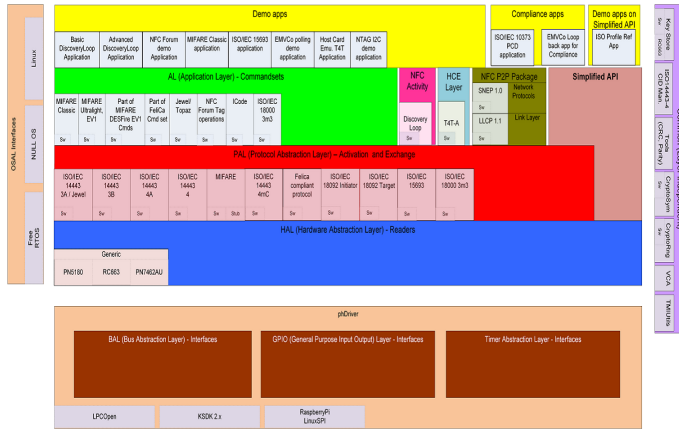


NFC Reader Library contents and architecture



NFC Reader Library release contents

NFC Reader Library API



Software examples



Documentation



NFC Reader Library API:

- Freely downloadable.
- Full implementation of all NFC protocols
- NDA version with full support for MIFARE DESFire EV2 and MIFARE Plus EV1
- SW package for MCUXpresso

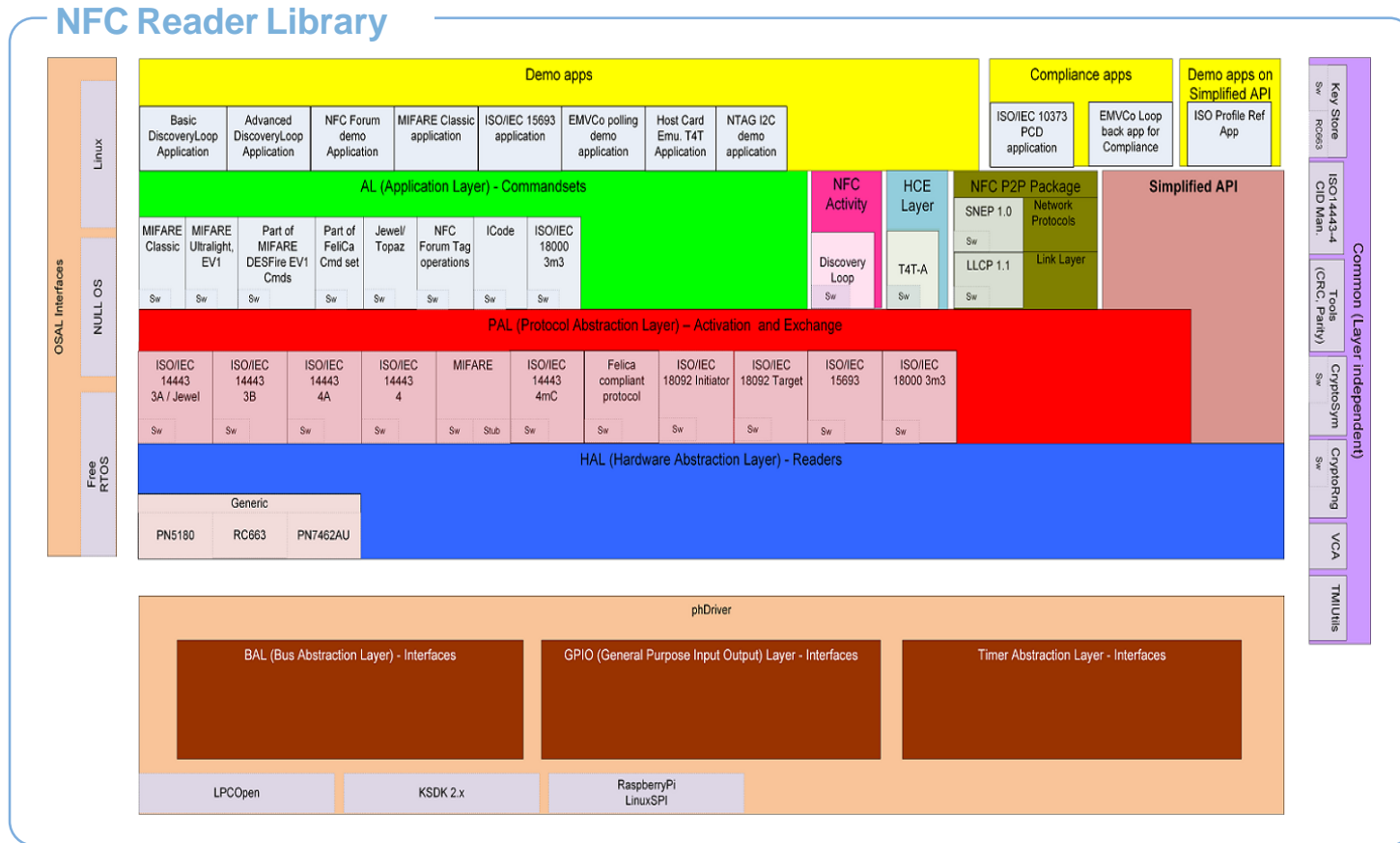
Software examples:

- BasicDiscoveryLoop
- AdvancedDiscoveryLoop
- NFCForum
- MIFARE Classic
- ISO15693
- EMVCo Loopback
-

Documentation:

- API documentation
- Generated from source file annotations
- Provided as HTML document

NFC Reader Library support for multiple products and platforms



Supported products:*

- CLRC663 *plus*
- PN5180
- PN7462AU

Supported dev boards:*

- CLEV6630B
- PNEV5180B
- PNEV7462B

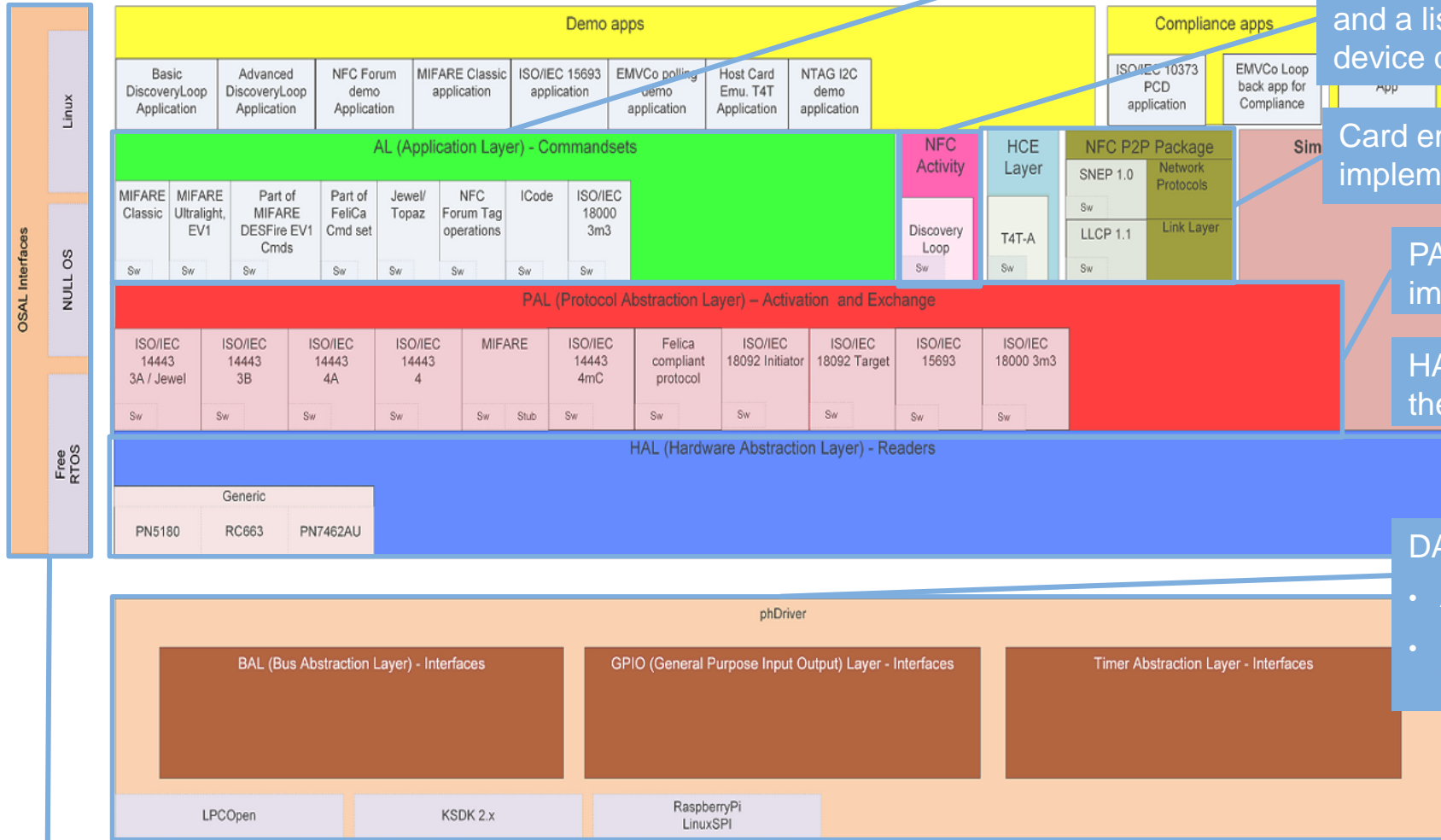
Supported platforms:*

- LPC1769, LPC11U68
- FRDM-K82F
- Raspberry Pi Model 3
- ... and portable to other MCUs and platforms.

Info and more information: www.nxp.com/pages/:NFC-READER-LIBRARY

* NFC Reader Library v5.02.00

NFC Reader Library architecture



AL contains application-specific implementations for various contactless cards (card command sets)

Discovery loop component implements a poll mode* and a listen mode** for contactless card and NFC device detection

Card emulation (HCE) and P2P NFC modes implementation

PAL components contain hardware-independent implementations of contactless protocols

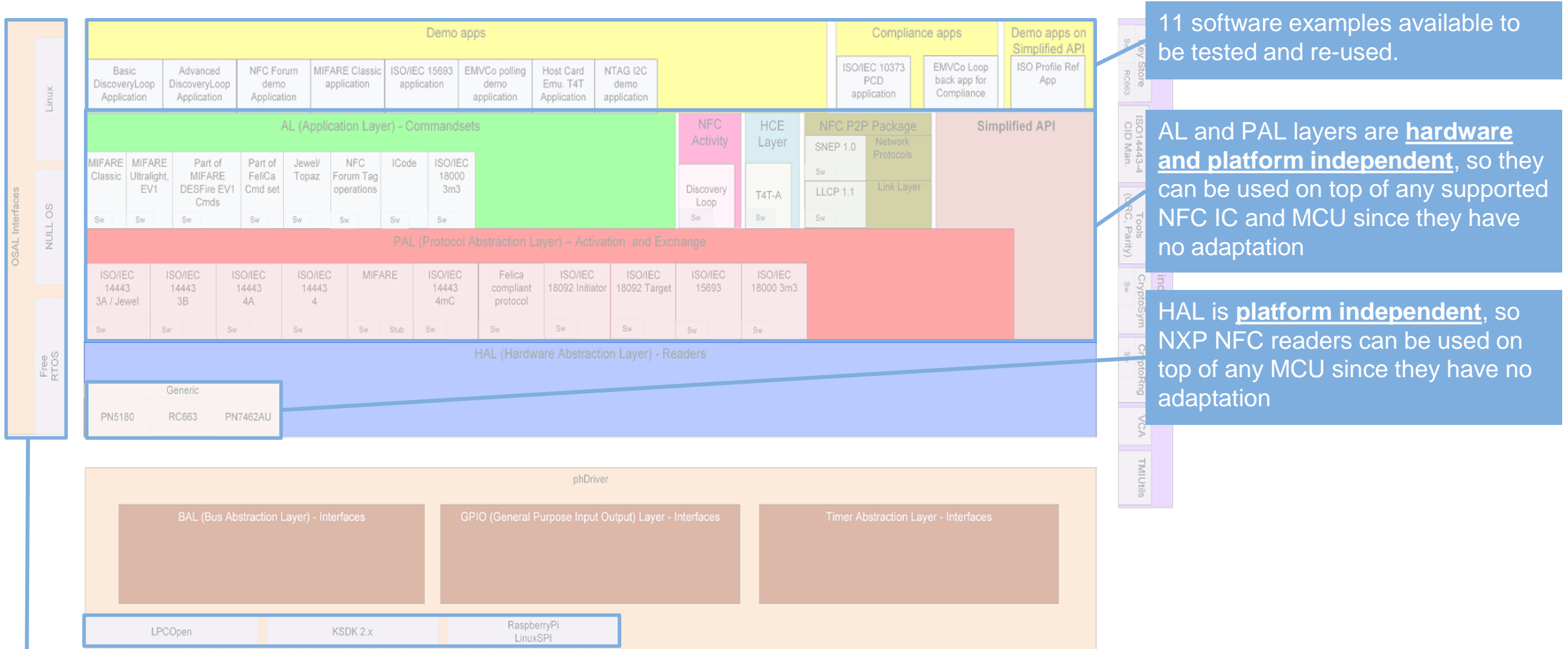
HAL components abstract the functionality of the NFC reader IC to a generic interface

DAL provides:

- API to abstract MCU GPIO functionalities
- BAL abstraction layer for the interface between host MCU and the NFC reader IC

OSAL which abstracts the OS simplifying development in multiple SW platforms

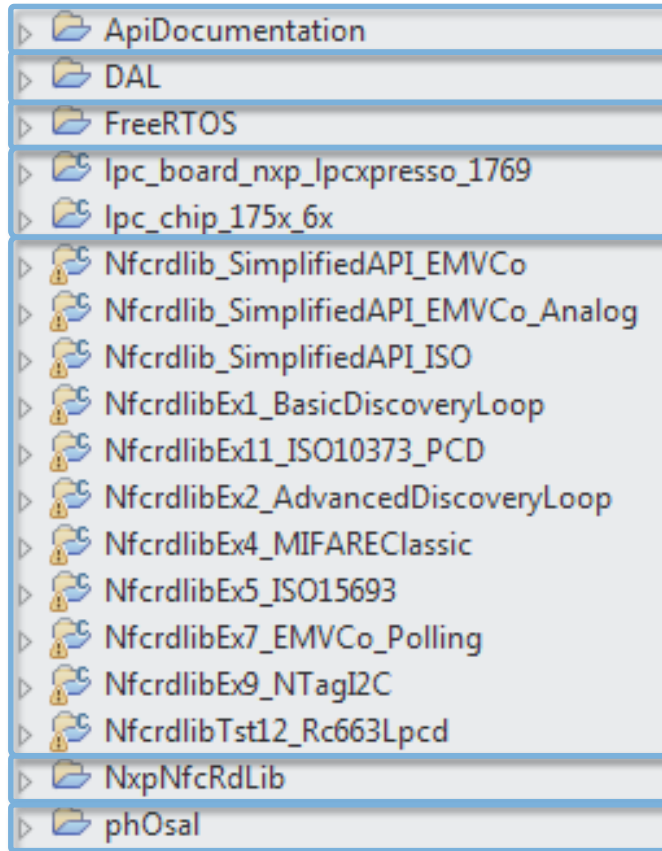
NFC Reader Library architecture (II)



The software examples can be imported and run the supported platforms without any adaptation.
 Support for other platforms requires adaptations in OSAL and DAL.



NFC Reader Library structure



API documentation
Driver abstraction layer
FreeRTOS support
Platform support

NFC Reader Library
examples

NFC Reader Library source code
OS abstraction layer



NFC Reader Library porting to K64F

Main steps



Main steps for NFC Reader Library porting to FRDM-K64F

NFC Reader Library package for FRDM-K82F

Without FRDM-K64F support

Prepare the HW for software development.

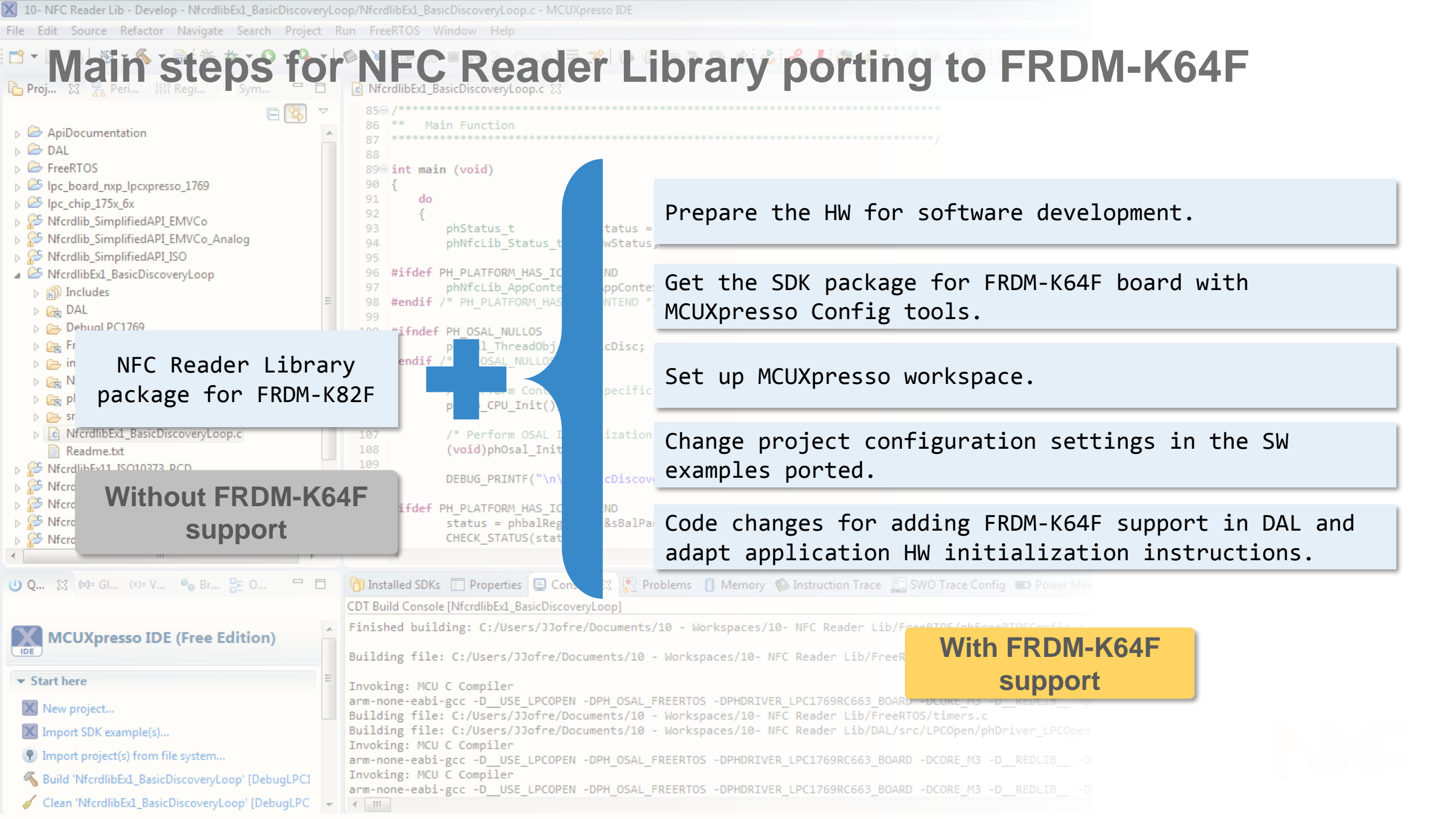
Get the SDK package for FRDM-K64F board with MCUXpresso Config tools.

Set up MCUXpresso workspace.

Change project configuration settings in the SW examples ported.

Code changes for adding FRDM-K64F support in DAL and adapt application HW initialization instructions.

With FRDM-K64F support



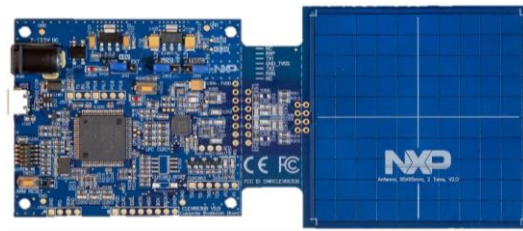
NFC Reader Library porting to K64F Setup

HW design:

We select CLRC663 *plus* and Kinetis K64F ICs for designing an NFC reader



Prototype with NXP reference dev boards



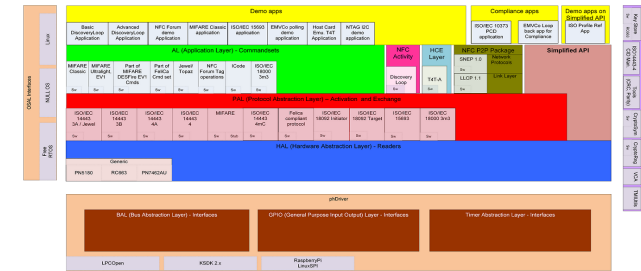
CLEV6630B



FRDM-K64F

SW development:

NFC Reader Library supports CLRC663 *plus* but no driver is available for K64F



Port NFC Reader Library to K64F host MCU

Key message:

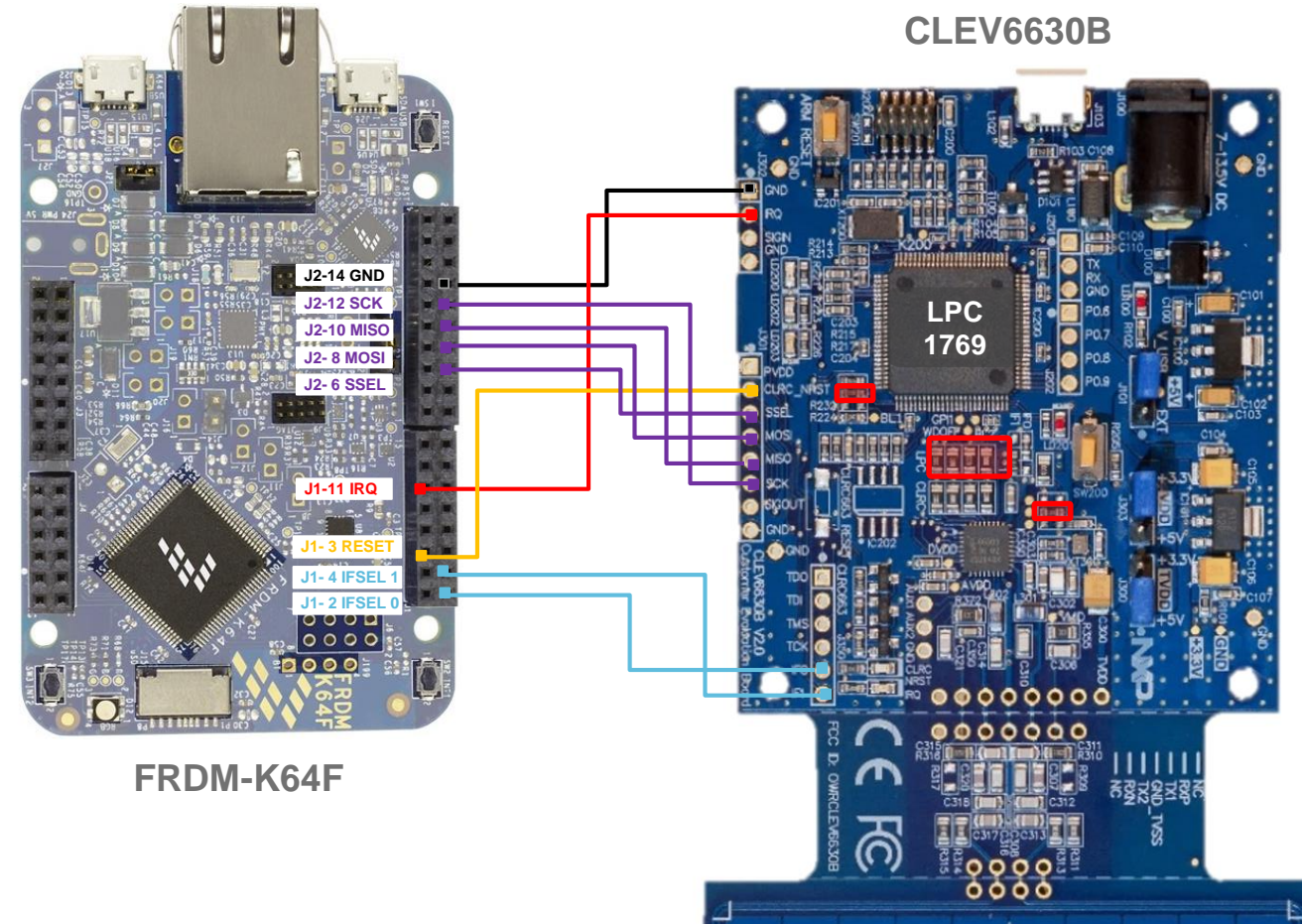
NFC Reader Library SW stack can be easily adapted to support your target MCU!


NFC Reader Library porting to K64F
Setting up the hardware



CLEV6630B board wiring with FRDM-K64F board

Pin function	FRDM-K64F	CLEV6630B
MOSI	J2-8	MOSI
MISO	J2-10	MISO
SCK	J2-12	SCK
SSEL	J2-6	SSEL
RESET	J1-3	CLRC_NRST
IRQ	J1-11	IRQ
IFSEL0	J1-2	IF0
IFSEL1	J1-4	IF1
GND	J2-14	GND



 Resistors that need to be removed to decouple the LPC1769 MCU from CLEV6630 and connect K64F MCU. Described in [AN11908](#) doc.

Key message:

CLEV6630B boards enables easy connection to your target MCU!

NFC Reader Library porting to K64F
Setting up the dev. environment



Get latest NFC Reader Library release

The screenshot shows the 'Overview' tab selected. The page title is 'NFC Reader Library - Software support for NFC Frontend solutions'. The 'Overview' section contains a paragraph: 'Feature complete software support library for NFC Frontend ICs. Designed to give developers a faster and simpler way to deliver NFC-enabled products. This multi-layer library, written in C, makes it easy to create NFC based applications. Special features, including interrupt-based event handling, Free RTOS support and MISRA-C compliancy, are provided along with the NFC Reader Library. The software is designed in a way to be easily portable to many different microcontrollers.' The 'Features' section lists several bullet points: 'All NXP® NFC frontend ICs', 'Full feature set according to NFC Forum', 'Synchronous API', 'Modular, multi-layer design', 'Interrupt-based event handling', 'Supports RTOS and non RTOS based architectures', 'Supports multiple development environments', 'Certification test applications for EMVCo Forum, ISO/IEC 10373-6 PICC/PCD', and 'Same software stack for PN512, CLRC663.'

The screenshot shows the 'Downloads' tab selected. The page title is 'NFC Reader Library - Software support for NFC Frontend solutions'. The 'Product Downloads' section features a download icon and the text 'NFC Reader Library - Software support for NFC Frontend solutions'. Below this, there are four bullet points: 'Latest Version', 'Previous Version', 'Updates & Patches', and 'Existing and New licenses'. A 'Download' button with an external link icon is visible on the right side of the page.

The screenshot shows the 'Product Information' page. The page title is 'NFC Reader Library'. There is a 'Register' button and a note: 'To register a New Product please click on the button'. Below this, there are tabs for 'Current' and 'Previous'. A table lists the available versions:

Version	Description	Download Log
05.02.01	NFC Reader Library for PN7462	Download Log
05.02.00	NFC Reader Library for CLRC663	Download Log
05.02.00	NFC Reader Library for Kinetis K82F	Download Log
05.02.00	NFC Reader Library for PN512	Download Log
05.02.00	NFC Reader Library for PN5180	Download Log
4.04.05	NFC Reader Library for PN512	Download Log

Step 1:

Browse to www.nxp.com/pages/:NFC-READER-LIBRARY

Step 2:

Go to **Downloads** tab and click **download** button

Step 3:

Download **NFC Reader Library for Kinetis K82F** package

NFC Reader Library porting is done starting from the Kinetis K82F package

Generate a downloadable SDK package for FRDM-K64F board

The screenshot shows the MCUXpresso Config Tools web interface. At the top, there's a navigation bar with 'NXP MCUXpresso OVERVIEW TOOLS MANAGE' and a user profile 'nxp65964'. Below the navigation bar, the main heading is 'MCUXpresso Config Tools' with a sub-heading: 'MCUXpresso Config Tools provides a set of system configuration tools that help users of all levels with a Kinetis or LPC-based MCU solution. Let it be your guide from first evaluation to production development.' A 'CFG' logo is also present.

The main content area features a 'Select or create a configuration' dropdown menu with 'FRDM-K64F' selected. Below this, there are several tool icons and descriptions:

- Config Settings:** Specify optional middleware and environment settings for your configuration.
- SDK Builder:** Generate a downloadable SDK archive for use with desktop MCUXpresso Tools. (This tool is highlighted with a blue box and a blue arrow pointing to the 'Edit Configuration' dialog in the lower screenshot.)
- Project Cloner:** Download an existing standalone SDK example project.
- Pins Tool:** Assign signals to pins, set electrical properties, and generate initialization code.
- Clocks Tool:** Setup the system clocks and generate initialization code.

Below the tools, there are sections for 'What's new' (dated Mar 31) and 'Additional Links'.

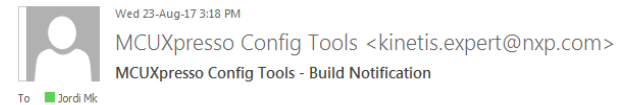
The lower screenshot shows the 'SDK Builder' tool interface. It has a 'Current Configuration' dropdown set to 'FRDM-K64F'. A 'Request Build' button is visible. An 'Edit Configuration' dialog is open, showing a search bar, a 'Select a Device, Board, or Kit' dropdown menu with 'Boards' expanded to show 'Kinetics' and 'FRDM-K64F' selected, and a 'Name your configuration' field with 'FRDM-K64F' entered. 'Cancel' and 'Save changes' buttons are at the bottom of the dialog.

Step 1: Go to MCUXpresso Config tools.

- Browse to <https://mcuxpresso.nxp.com>
- Select **SDK Builder**

Step 2: Customize SDK package for your MCU.

- Select **FRDM-K64F** in the **Current Configuration** drop down menu.
- Select **Request Build**.
- Download your custom SDK package.



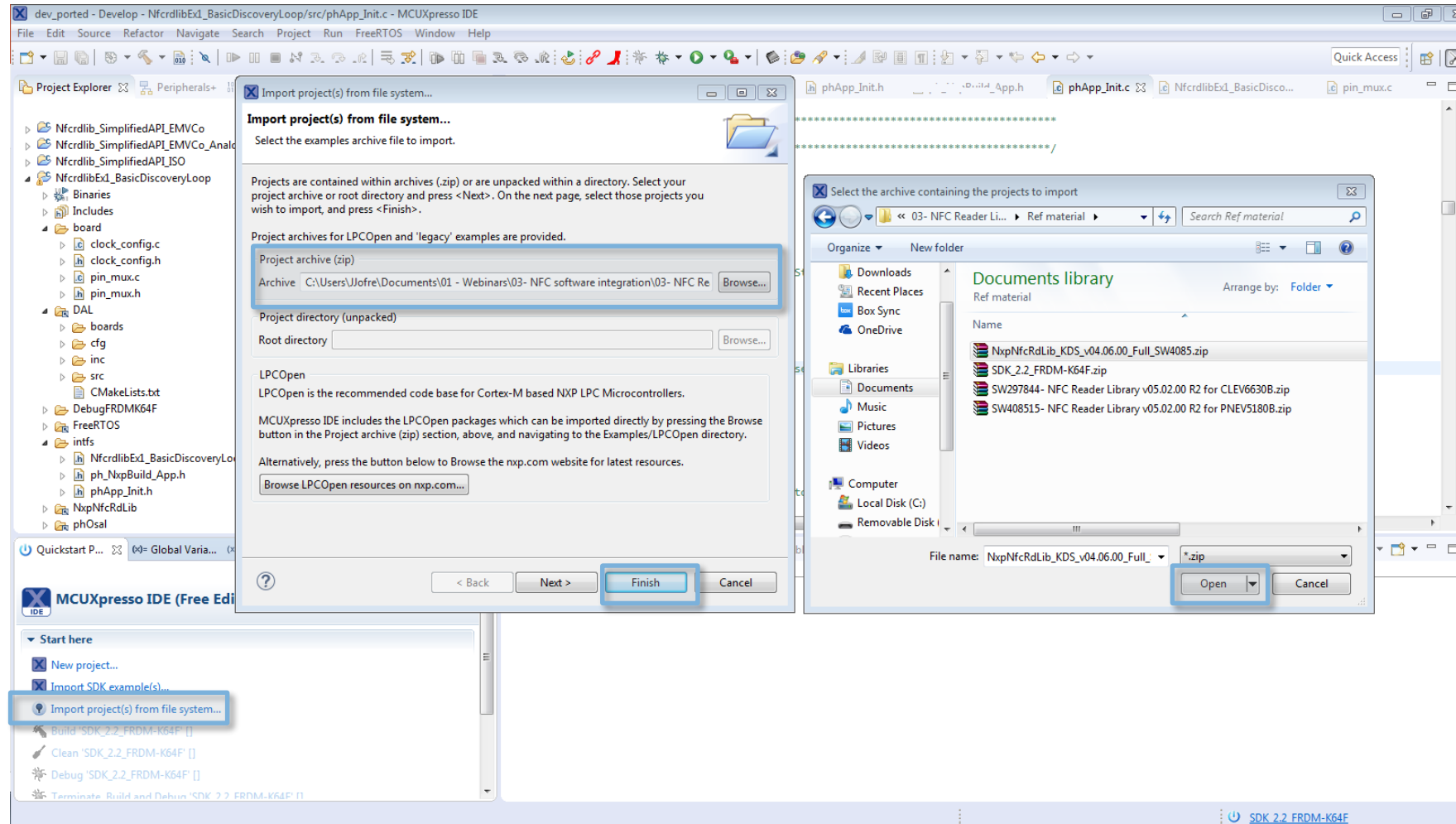
NXPN MCUXpresso Config Tools User,

Your recent build request (SDK_2.2_FRDM-K64F) has completed and is now available for download from within your MCUXpresso Config Tools SDK Build Archive (<http://mcuxpresso.nxp.com>).

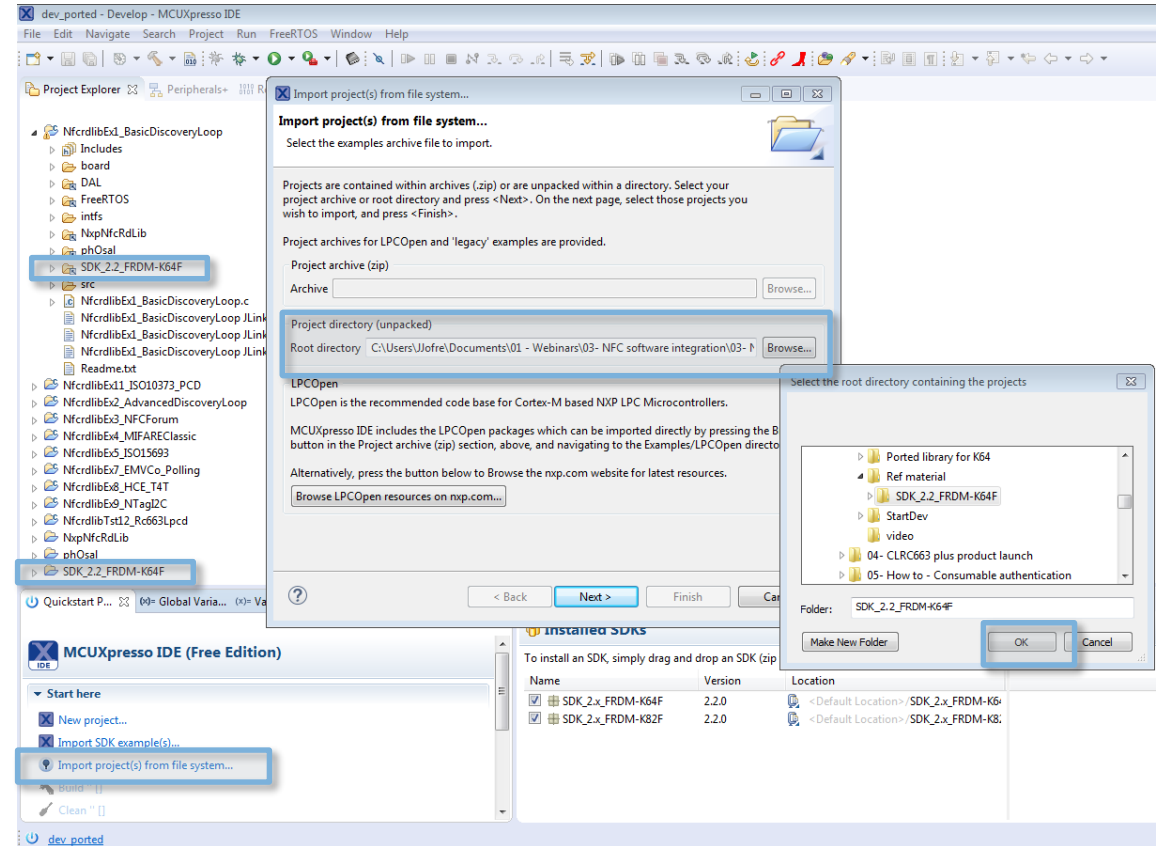
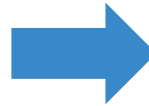
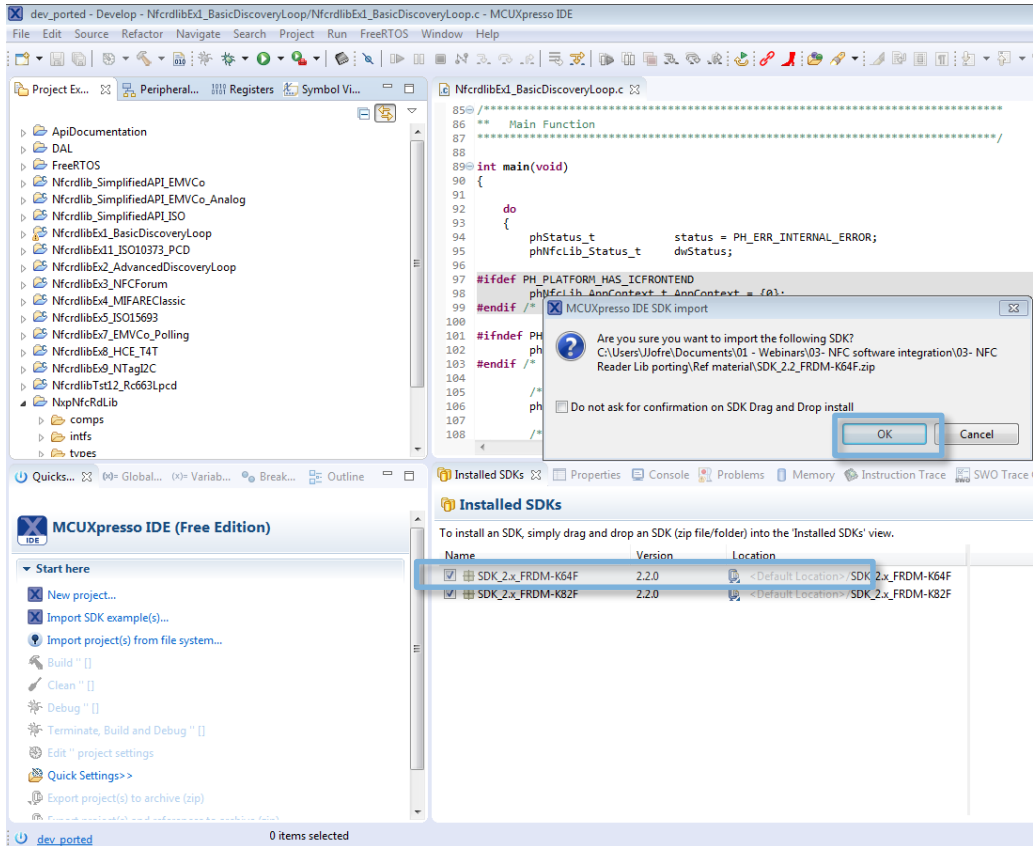
Download the package from the following link: <https://mcuxpresso.nxp.com/en/license?hash=3967a155a1806482798aa3590af4f481&uvid=55853>



Import NFC Reader Library into MCUXpresso workspace



Install and link FRDM-K64F SDK into MCUXpresso workspace



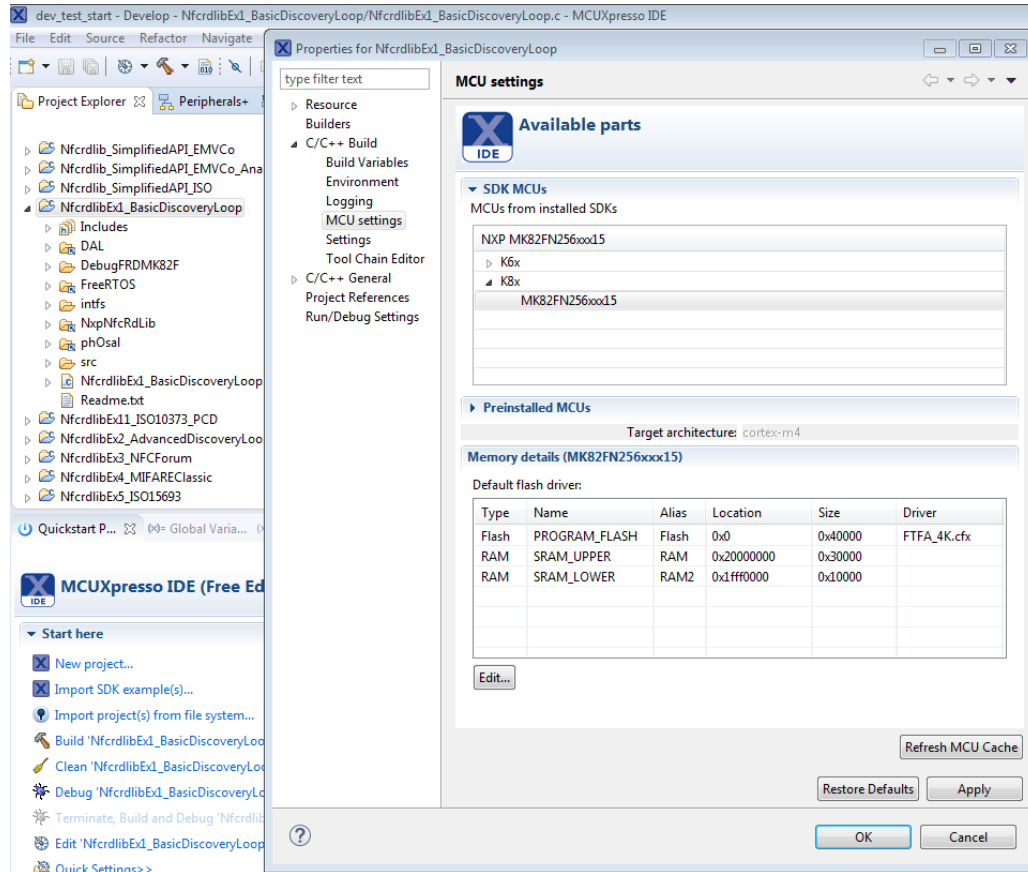
Drag and drop the FRDM-K64F SDK into *Installed SDKs* tab of MCUXpresso IDE

Import FRDM-K64F SDK into workspace and link it with the software examples.

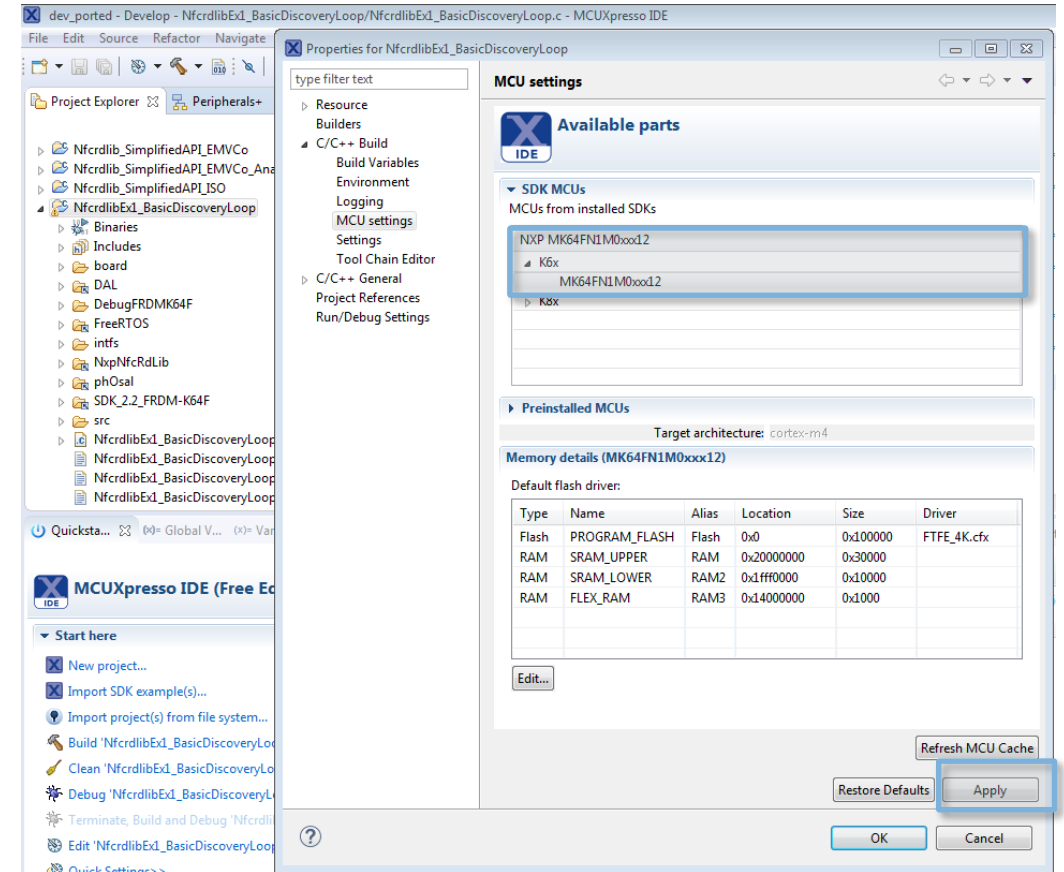
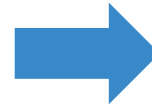
NFC Reader Library porting to K64F Configuration changes



Select FRDM-K64F SDK in the project MCU settings



NFC Reader Library NfcrdlbEx1 for FRDM-K82F



NFC Reader Library NfcrdlbEx1 ported to FRDM-K64F

In *Project properties* → *MCU settings*:

Change SDK MCU to FRDM-K64F SDK

Define FRDM-K64F SDK preprocessor symbols in the project

Properties for NfcrdlbEx1_BasicDiscoveryLoop

Configuration: UdebugFRDMK82F [Active]

Settings

Tool Settings

Build steps

Build Artifact

Binary Parsers

Error Parsers

MCU C Compiler

Dialect

Preprocessor

Optimization

Debugging

Warnings

Miscellaneous

Architecture

MCU Assembler

General

Architecture & Headers

MCU Linker

General

Libraries

Miscellaneous

Shared Library Settings

Architecture

Managed Linker Script

Multicore

Do not search system directories (-nostdinc)

Preprocess only (-E)

Defined symbols (-D)

__REDLIB__
CR_INTEGER_PRINTF
SDK_DEBUGCONSOLE=0
__MCUXPRESSO__
_USE_CMSIS
DEBUG
SDK_OS_BAREMETAL
FSL_RTOS_BM
CPU_MK82FN256VLL15_cm4
CPU_MK82FN256VLL15
PH_OSAL_FREERTOS
PHDRIVER_FRDM_K82FPN5180_BOARD
NXPBUILD_CUSTOMER_HEADER_INCLUDED

Undefined symbols (-U)

DAL → BoardSelection.h

```
#ifdef PHDRIVER_FRDM_K82FPN5180_BOARD
#include <Board_FRDM_K82FPn5180.h>
#endif
```

NFC Reader Library NfcrdlbEx1 for FRDM-K82F

Properties for NfcrdlbEx1_BasicDiscoveryLoop

Configuration: DebugFRDMK64F [Active]

Settings

Tool Settings

Build steps

Build Artifact

Binary Parsers

Error Parsers

MCU C Compiler

Dialect

Preprocessor

Optimization

Debugging

Warnings

Miscellaneous

Architecture

MCU Assembler

General

Architecture & Headers

MCU Linker

General

Libraries

Miscellaneous

Shared Library Settings

Architecture

Managed Linker Script

Multicore

Do not search system directories (-nostdinc)

Preprocess only (-E)

Defined symbols (-D)

__REDLIB__
CR_INTEGER_PRINTF
SDK_DEBUGCONSOLE=0
__MCUXPRESSO__
_USE_CMSIS
DEBUG
SDK_OS_BAREMETAL
FSL_RTOS_BM
CPU_MK64FN1M0VLL12_cm4
CPU_MK64FN1M0VLL12
PH_OSAL_FREERTOS
PHDRIVER_FRDM_K64FRC663_BOARD
NXPBUILD_CUSTOMER_HEADER_INCLUDED

Undefined symbols (-U)

DAL → BoardSelection.h

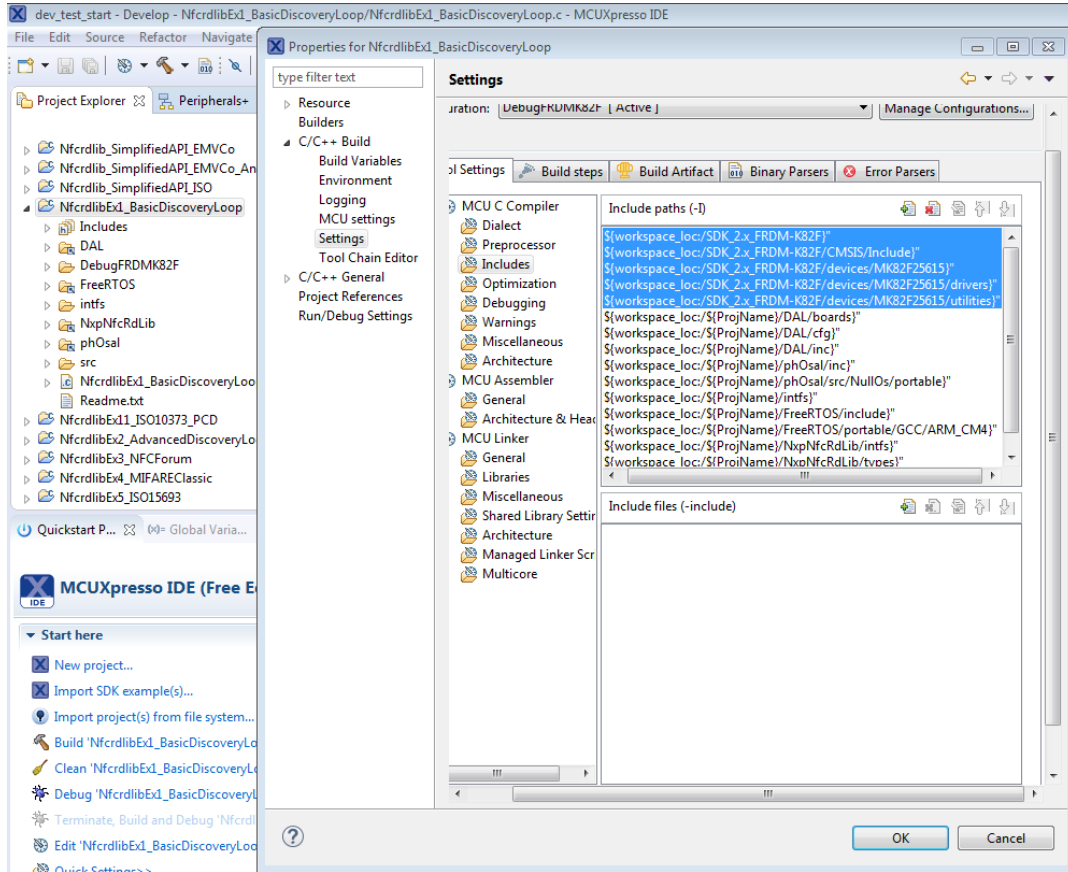
```
#ifdef PHDRIVER_FRDM_K64FRC663_BOARD
#include <Board_FRDM_K64FRC663.h>
#endif
```

NFC Reader Library NfcrdlbEx1 ported to FRDM-K64F

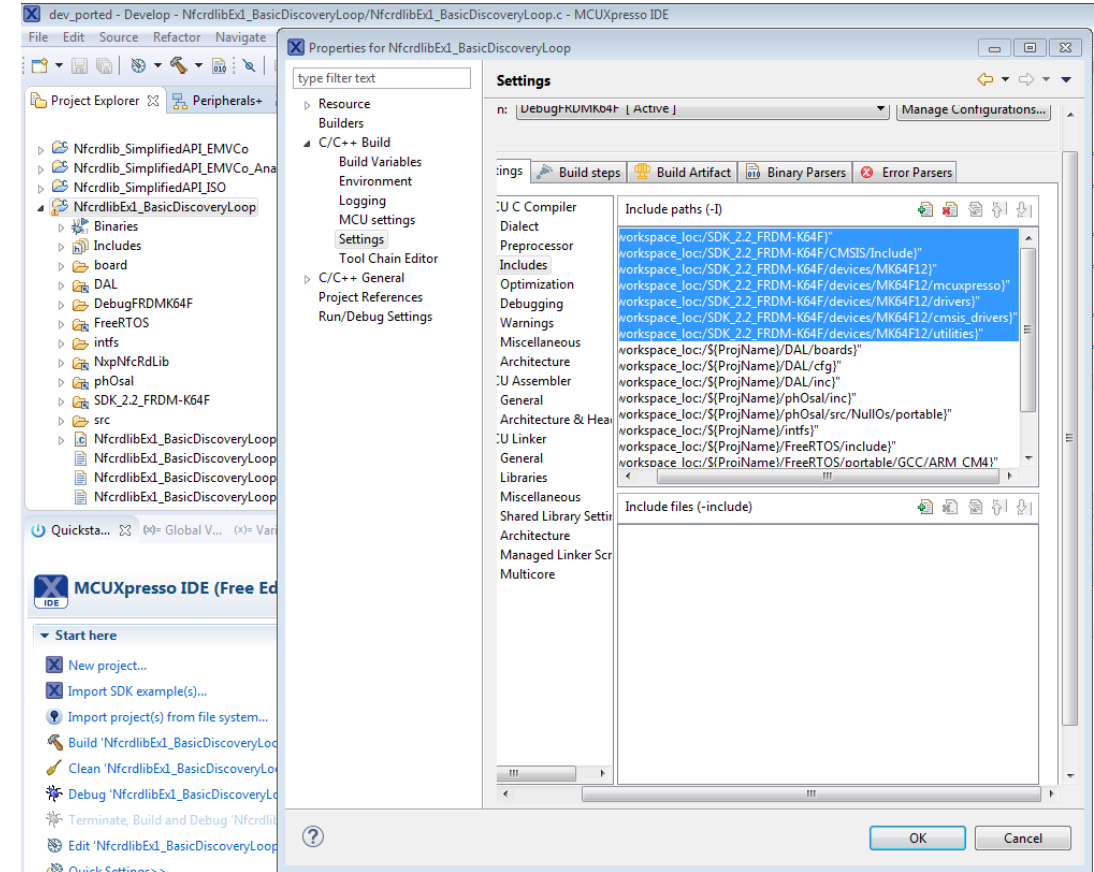
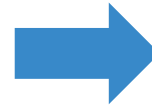
In *Project properties* → *Settings* → *Preprocessor*:

Change preprocessor defined symbols

Add include paths for FRDM-K64F SDK files



NFC Reader Library NfcrdlbEx1 for FRDM-K82F

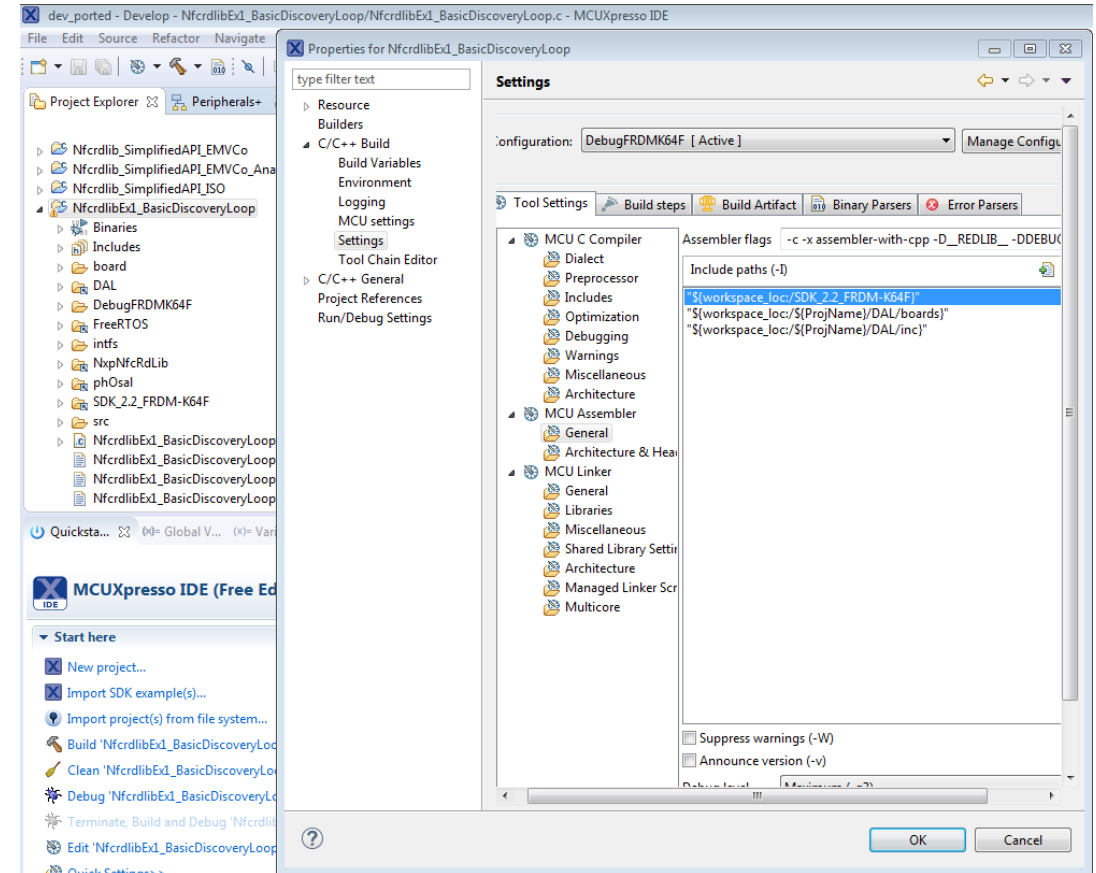
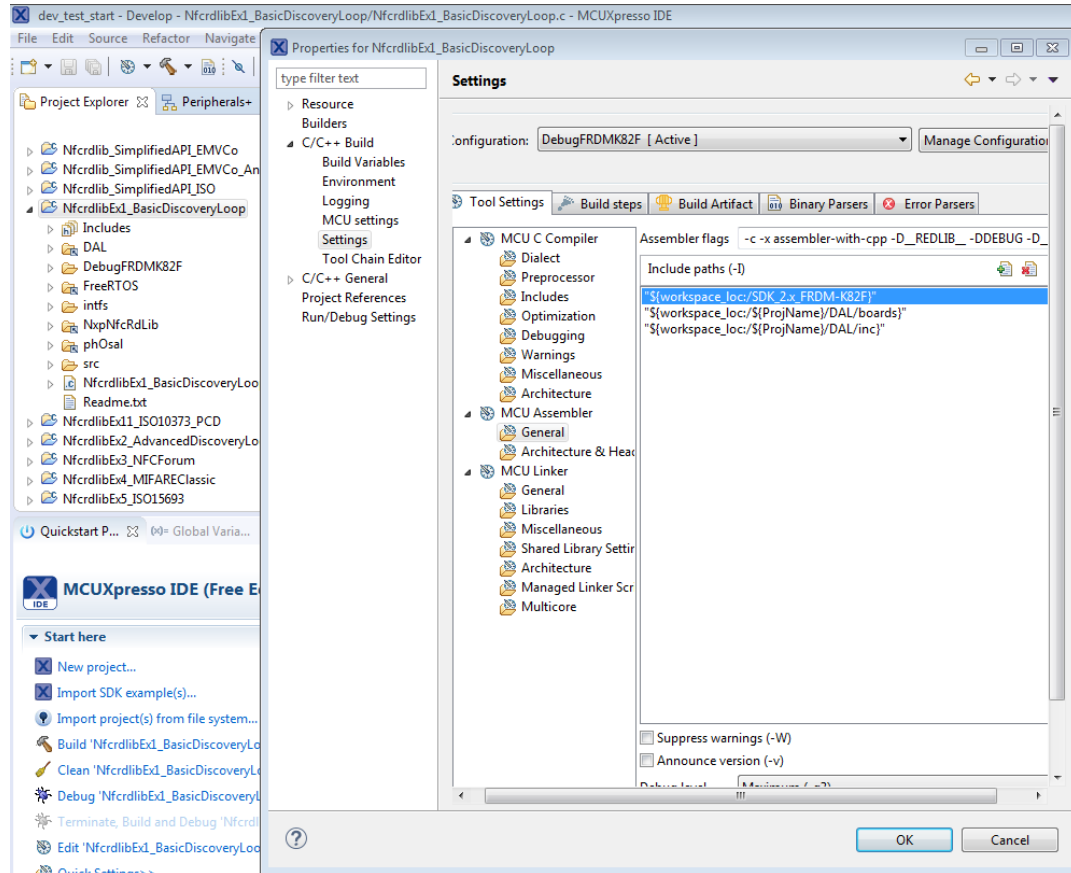


NFC Reader Library NfcrdlbEx1 ported to FRDM-K64F

In *Project properties* → *Settings* → *Includes*:

Add include paths for FRDM-K64F SDK files

Add include path for FRDM-K64F MCU assembler



NFC Reader Library NfcrdlbEx1 for FRDM-K82F

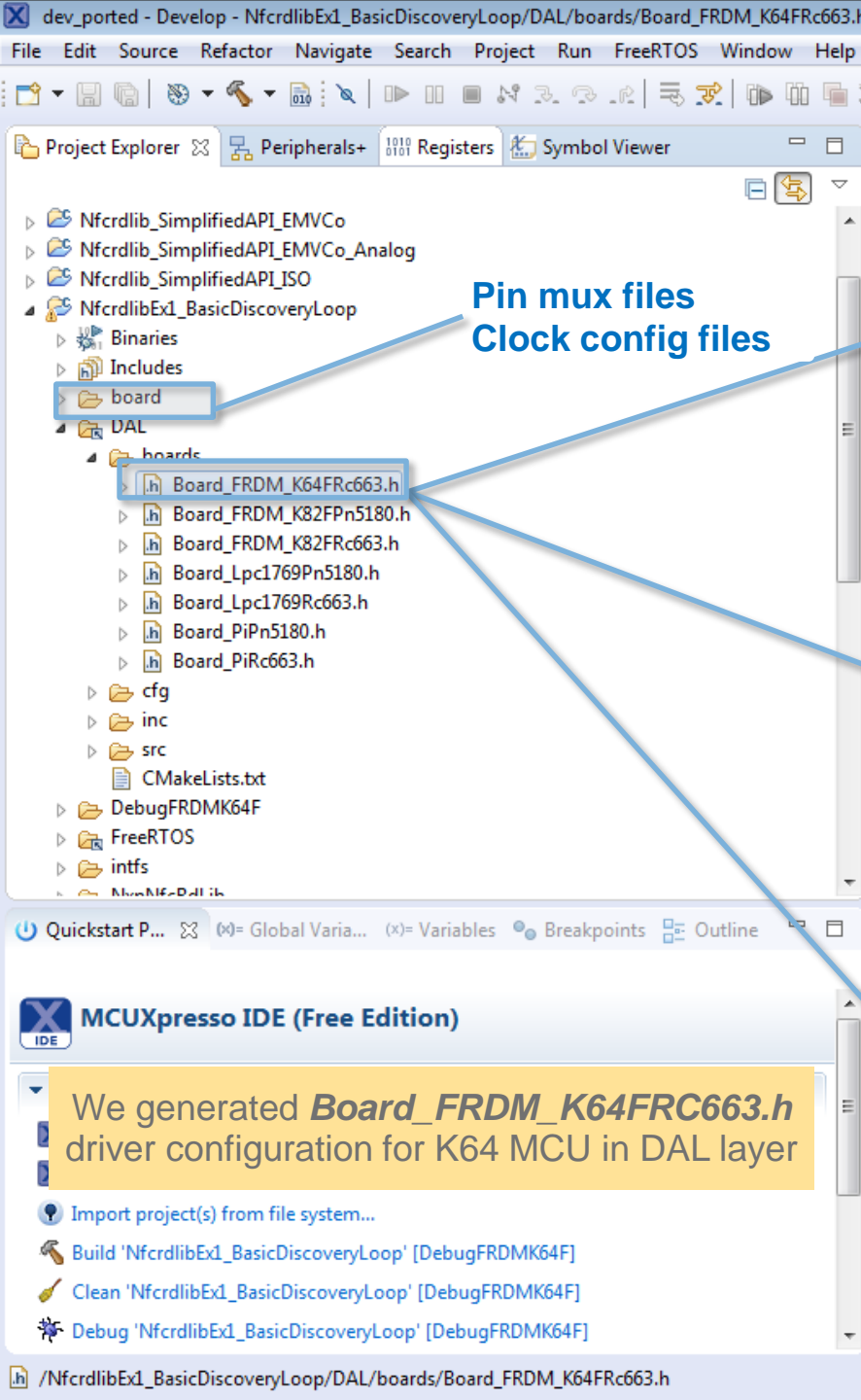
NFC Reader Library NfcrdlbEx1 ported to FRDM-K64F

In **Project properties** → **Settings** → **MCU assembler**:

Add include path for MCU assembler

NFC Reader Library porting to K64F Code changes





Pin mux files
Clock config files

DAL driver adaptation for FRDM-K64F

Board Pin/Gpio configurations

```
#define PHDRIVER_PIN_RESET ((GPIO_PORT_B << 8) | 19) /**< Reset pin, Pin13, GPIOA, PORTA */
#define PHDRIVER_PIN_IRQ ((GPIO_PORT_C << 8) | 0) /**< IRQ pin, Pin7, GPIOC, PORTC */
```

SPI configuration

```
#define ENABLE_PORT_SSP_1 kCLOCK_PortD//SCK
#define PORT_SSP_1 PORTD
#define FIRST_PINNUM_SSP 1

#define ENABLE_PORT_SSP_2 kCLOCK_PortD//MOSI
#define PORT_SSP_2 PORTD
#define SECOND_PINNUM_SSP 3

#define ENABLE_PORT_SSP_3 kCLOCK_PortD//MISO
#define PORT_SSP_3 PORTD
#define THIRD_PINNUM_SSP 2

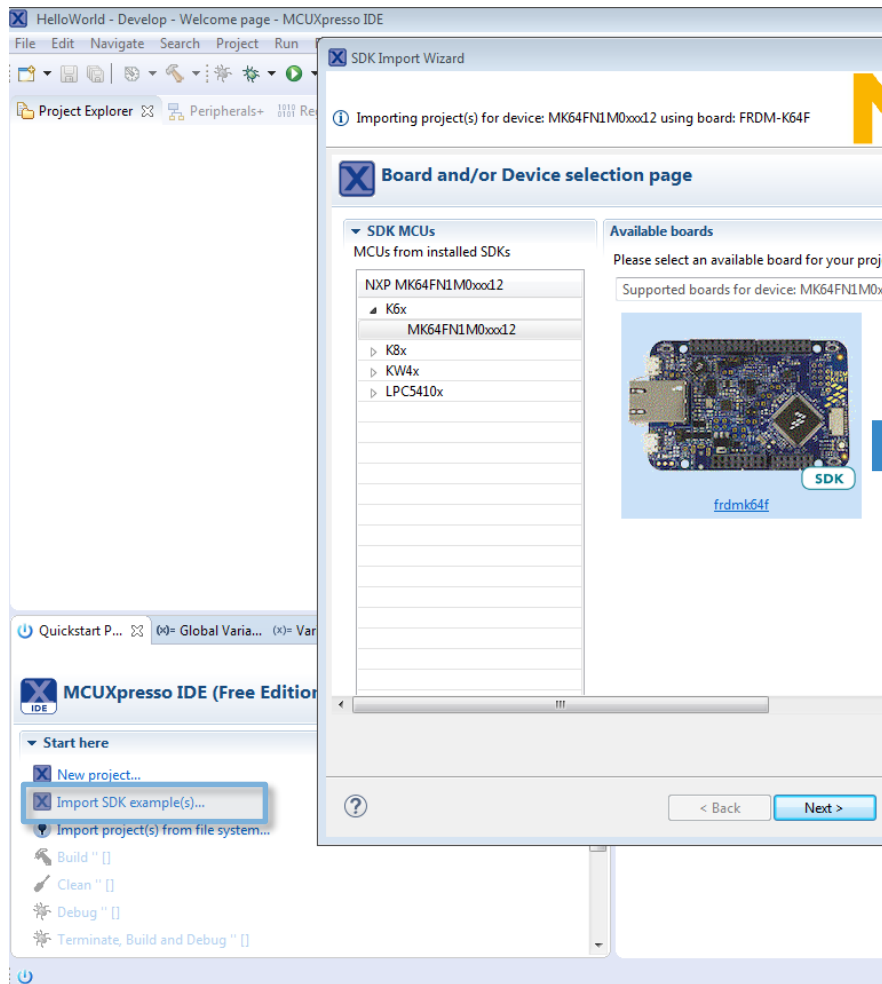
#define PHDRIVER_PIN_SSEL ((GPIO_PORT_D << 8) | 0) /**< SSEL pin, Pin14, GPIOA, PORTA */
```

Timer configuration

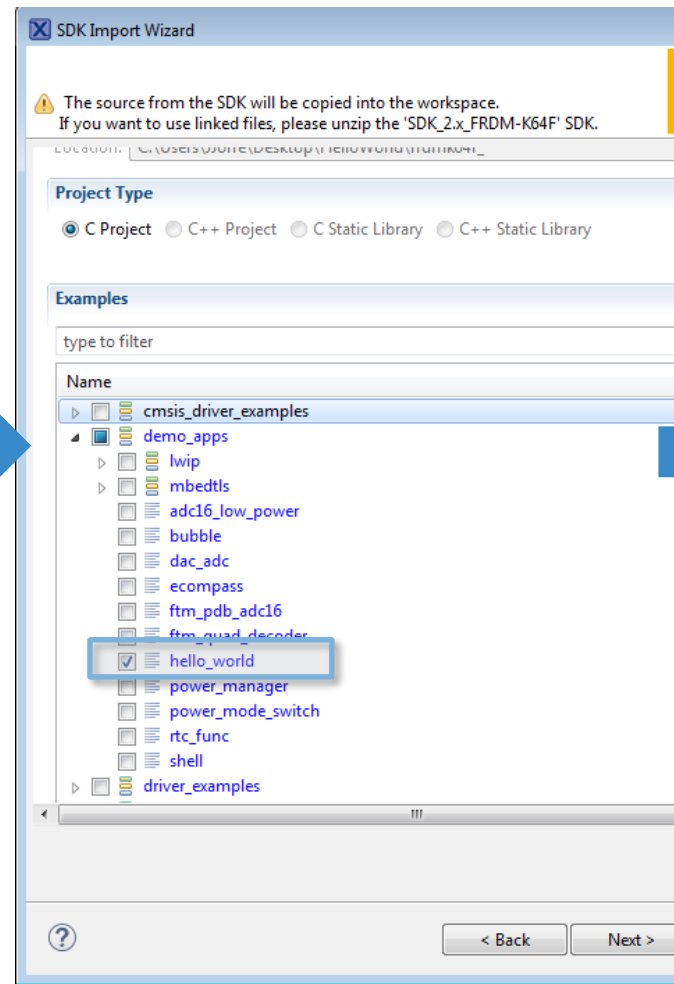
```
#define PH_DRIVER_KSDK_PIT_TIMER PIT
#define PH_DRIVER_KSDK_PIT_CLK kCLOCK_BusClk
#define PH_DRIVER_KSDK_TIMER_CHANNEL kPIT_Chnl_0 /**< PIT channel number 0 */
#define PH_DRIVER_KSDK_TIMER_NVIC PIT0_IRQn
```



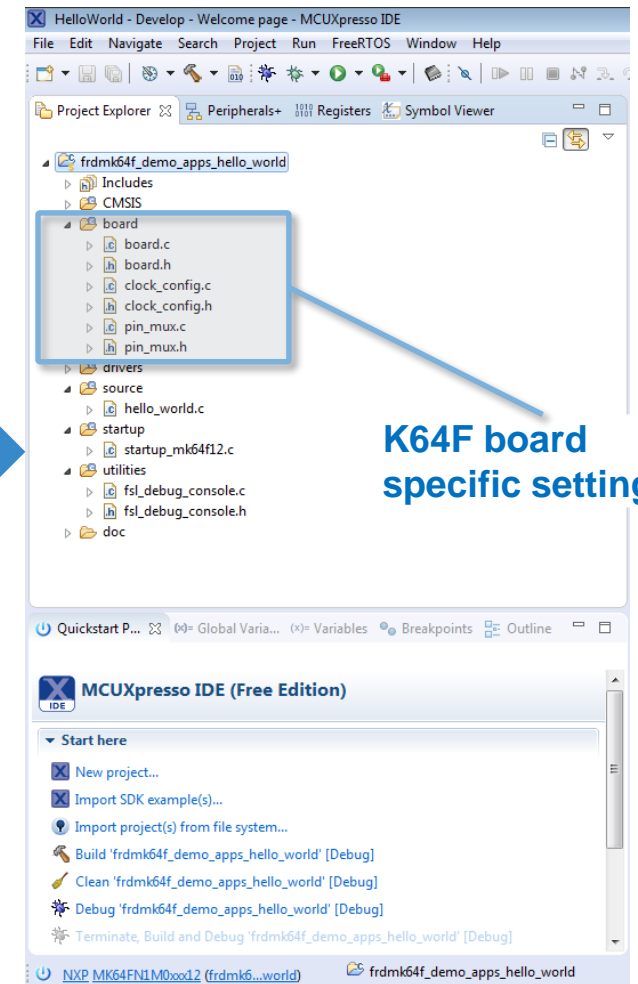
Use SDK examples to get K64F board specific configuration



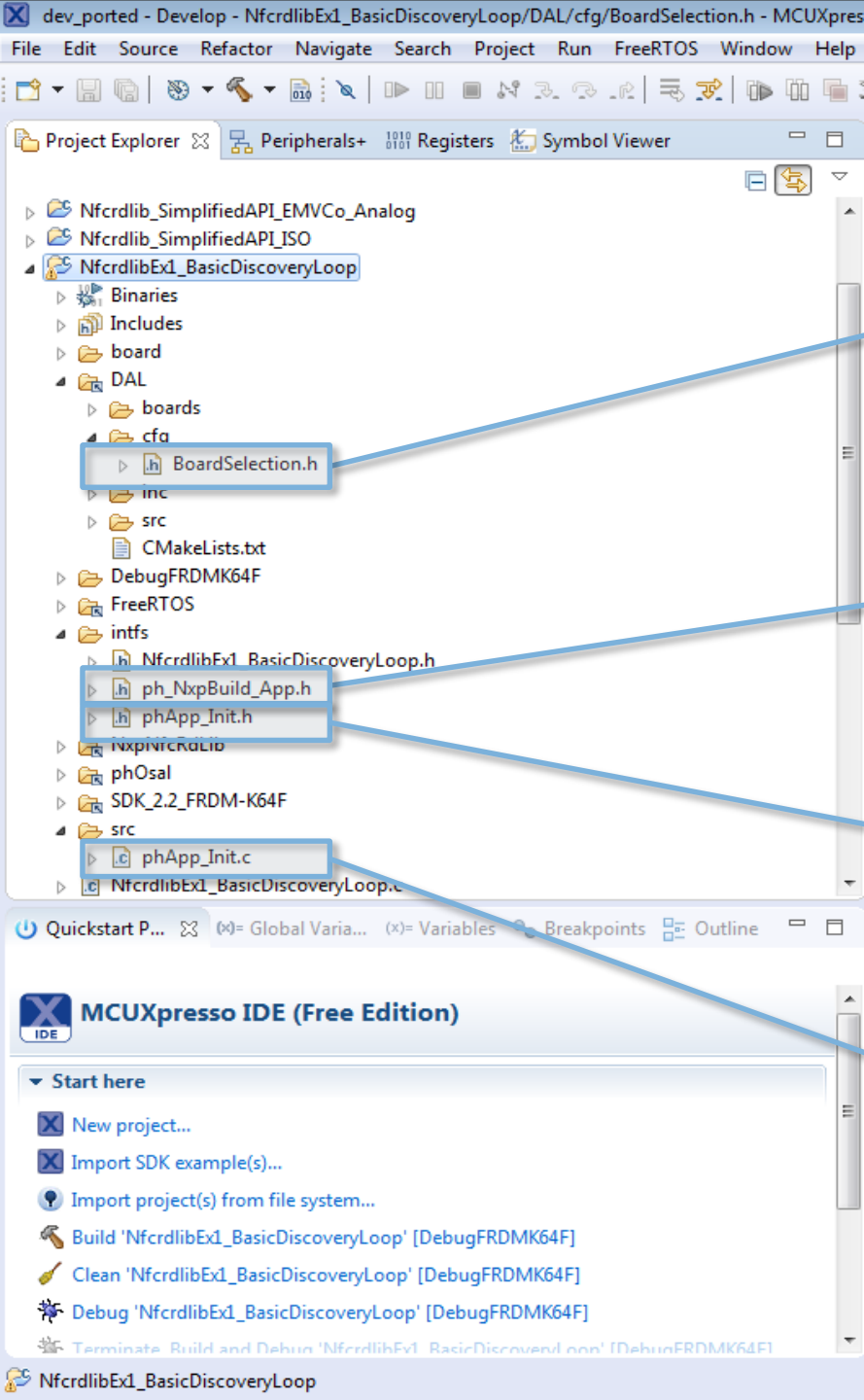
1. Import K64F SDK examples



2. Select HelloWorld example



3. Use board folder source code as reference



Added FRDM_K64F macro definitions

BoardSelection.h

```
#ifndef PHDRIVER_FRDM_K64FRC663_BOARD
#include <Board_FRDM_K64Frc663.h>
#endif
```

phNxpBuild_App.h

```
#if defined(PHDRIVER_LPC1769RC663_BOARD) \
|| defined(PHDRIVER_FRDM_K82FRC663_BOARD) \
|| defined(PHDRIVER_FRDM_K64FRC663_BOARD)
#define NXPBUILD__PHHAL_HW_RC663
```

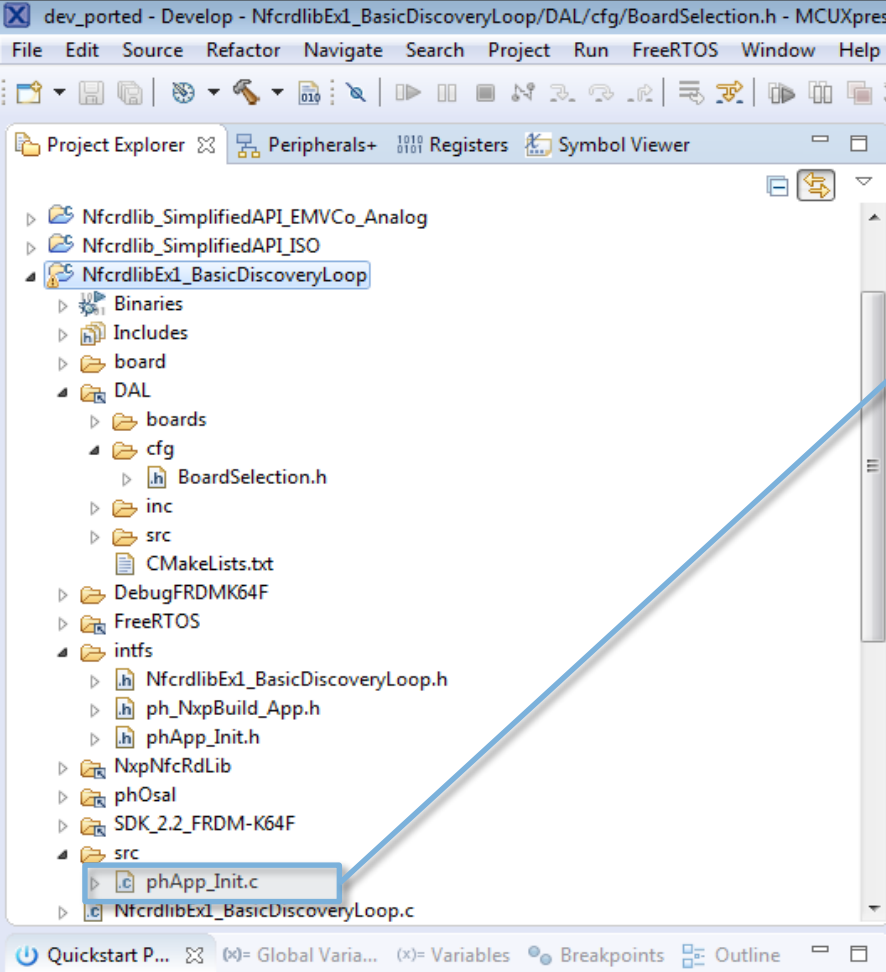
phApp_Init.h

```
#if defined(PHDRIVER_FRDM_K64FRC663_BOARD)
#define PHDRIVER_KINETIS_K64
```

phApp_Init.c

```
#ifndef PHDRIVER_KINETIS_K64
static void phApp_K64_Init(void);
#endif /* PHDRIVER_KINETIS */
```





Add FRDM-K64F CPU initialization code

phApp_Init.c

```
void phApp_CPU_Init(void){  
#if defined PHDRIVER_KINETIS  
    phApp_K64_Init();  
    ...  
}
```

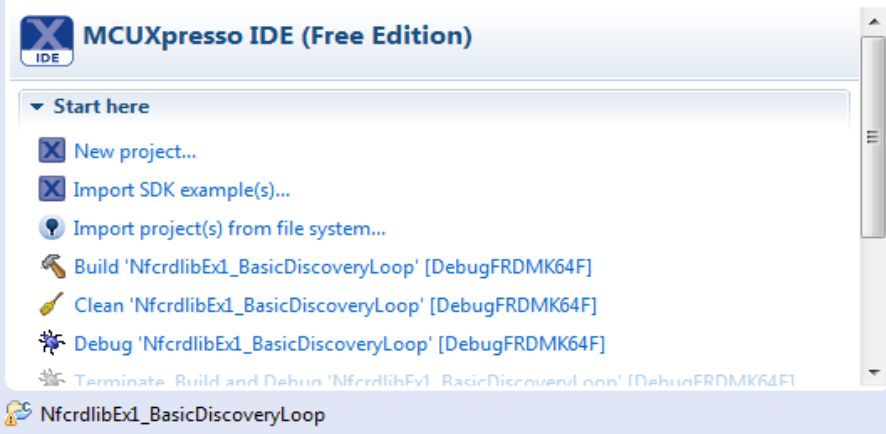


phApp_K64_Init() function

```
BOARD_BootClockRUN(); /* Code for BOARD_Clock Run configuration*/  
  
SystemCoreClockUpdate();  
  
PIT_GetDefaultConfig(&pitConfig);  
  
PIT_Init(PIT, &pitConfig); /* Init pit module */  
  
BOARD_InitPins(); /* Initialize UART pins below used to Print */
```

phApp_Init.c includes code specific to HW used (FRDM-K64F)

Clock and pin_mux config files were taken from FRDM-K64F SDK example



Project Explorer

- Peripherals+
- Registers
- Symbol Viewer
- Nfcrdlib_SimplifiedAPI_EMVCo
- Nfcrdlib_SimplifiedAPI_EMVCo_Analog
- Nfcrdlib_SimplifiedAPI_ISO
- NfcrdlibEx1_BasicDiscoveryLoop
 - Binaries
 - Includes
 - board
 - DAL
 - DebugFRDMK64F
 - FreeRTOS
 - intfs
 - NxpNfcrdLib
 - phOsal
 - SDK_2_2_FRDM-K64F
 - src
 - NfcrdlibEx1_BasicDiscoveryLoop.c
 - NfcrdlibEx1_BasicDiscoveryLoop JLink DebugFRDMK64F.launch
 - NfcrdlibEx1_BasicDiscoveryLoop JLink default.launch
 - NfcrdlibEx1_BasicDiscoveryLoop JLink test.launch

```

85 /*****
86 **  Main Function
87 *****/
88
89 int main(void)
90 {
91
92     do
93     {
94         phStatus_t      status = PH_ERR_INTERNAL_ERROR;
95         phNfcLib_Status_t  dwStatus;
96
97 #ifdef PH_PLATFORM_HAS_ICFRONTEND
98     phNfcLib_AppContext_t AppContext = {0};
99 #endif /* PH_PLATFORM_HAS_ICFRONTEND */
100
101 #ifndef PH_OSAL_NULLLOS
102     phOsal_ThreadObj_t BasicDisc;
103 #endif /* PH_OSAL_NULLLOS */
104
105     /* Perform Controller specific initialization. */
106     phApp_CPU_Init();
107
108     /* Perform OSAL specific initialization. */

```

No consoles to display at this time.

MCUXpresso IDE (Free Edition)

Start here

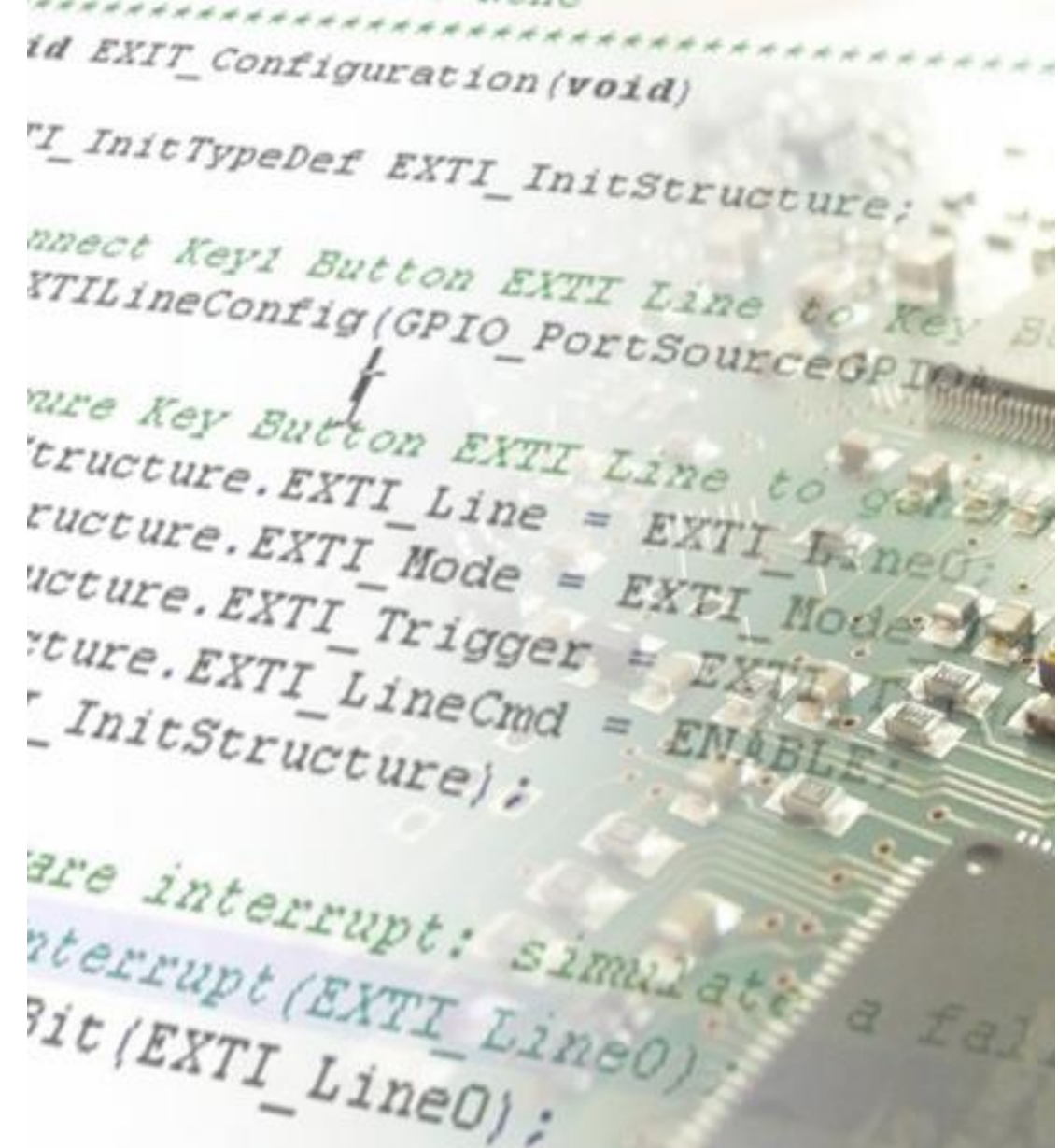
- New project...
- Import SDK example(s)...
- Import project(s) from file system...
- Build 'NfcrdlibEx1_BasicDiscoveryLoop' [DebugFRDMK64F]
- Clean 'NfcrdlibEx1_BasicDiscoveryLoop' [DebugFRDMK64F]
- Debug 'NfcrdlibEx1_BasicDiscoveryLoop' [DebugFRDMK64F]
- Terminate, Build and Debug 'NfcrdlibEx1_BasicDiscoveryLoop' [DebugFRDMK64F]
- Edit 'NfcrdlibEx1_BasicDiscoveryLoop' project settings
- Quick Settings>>
- Export project(s) to archive (zip)
- Export project(s) and references to archive (zip)

Wrap up & Q&A



General considerations to port NFC Reader Library to your target MCU

- Adapt the MCU drivers → DAL Layer
 - Timers
 - Interrupts
 - GPIOs pin configuration
 - Host interface configuration
- Adapt the OS → OSAL layer
 - Porting the FreeRTOS
 - Porting to a different OS
- Adapt the sample projects
 - Project configuration settings, includes & macros
 - Main application HW init code (e.g. Clocks)



```
*****
id EXIT_Configuration(void)

r_t_InitTypeDef EXTI_InitStructure;

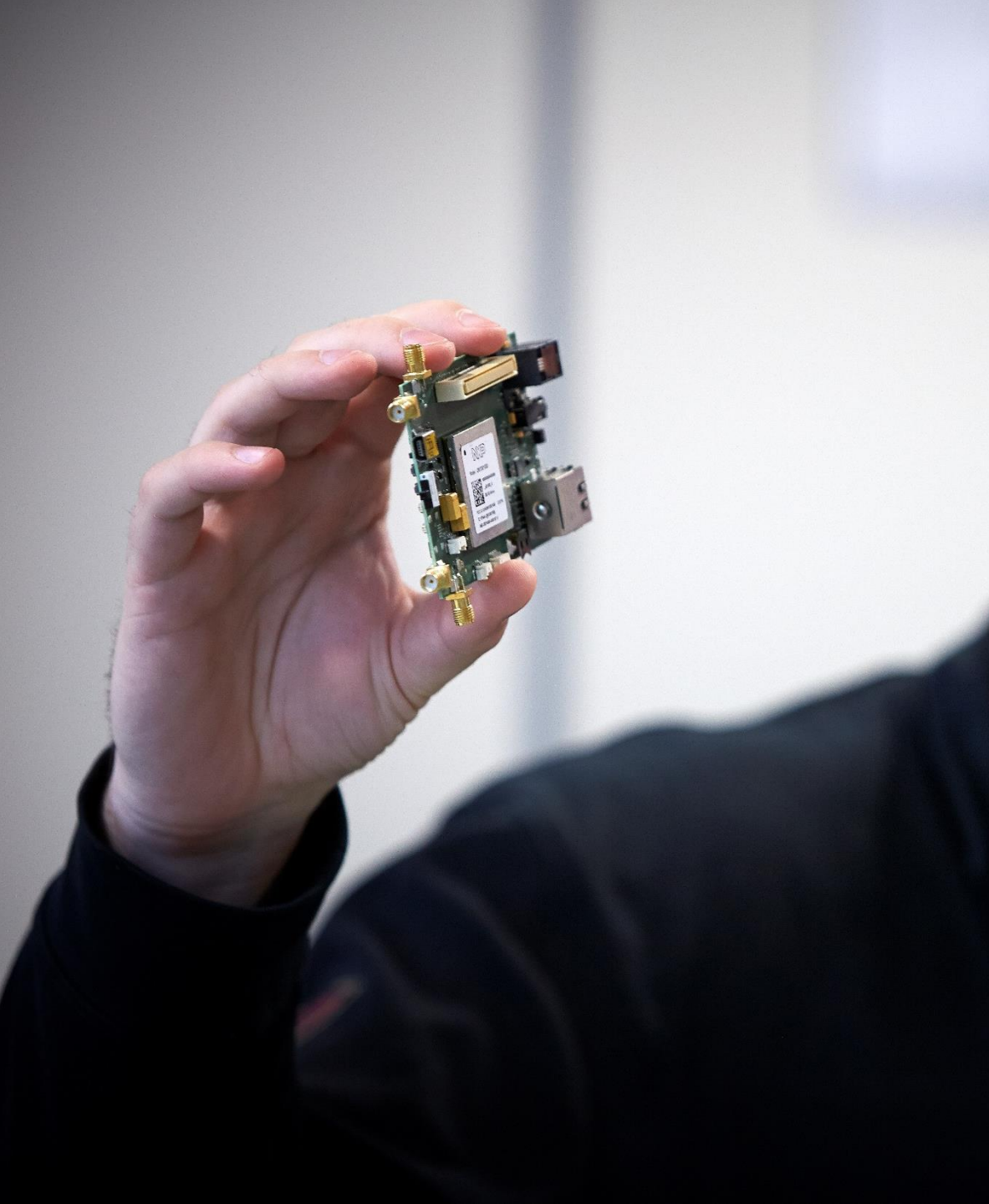
connect Key1 Button EXTI Line to GPIO

EXTILineConfig(GPIO_PortSourceGPIOA,

Configure Key Button EXTI Line to generate

structure.EXTI_Line = EXTI_Line0;
structure.EXTI_Mode = EXTI_Mode_Interrupt;
structure.EXTI_Trigger = EXTI_Trigger_Falling;
structure.EXTI_LineCmd = ENABLE;
r_InitStructure);

are interrupt: simulate a fall
interrupt(EXTI_Line0);
bit(EXTI_Line0);
```

Reference links & info

- NFC Reader Library
www.nxp.com/pages/:NFC-READER-LIBRARY
- CLRC663 *plus*
www.nxp.com/products/:CLRC66303HN
- CLRC663 *plus* development kit
www.nxp.com/demoboard/OM26630
- FRDM-K64F board
www.nxp.com/demoboard/FRDM-K64F

Software development in Android and iOS

Embedded software for MCUs

JCOP, Java Card operating Systems

Hardware design and development

Digital, analog, sensor acquisition, power management

Wireless communications WiFi, ZigBee, Bluetooth, BLE

Contactless antenna RF design, evaluation and testing

MIFARE applications

End-to-end systems, readers and card-related designs

EMVco applications

Readers, cards, design for test compliancy (including PCI)

Secure Element management

GlobalPlatform compliant backend solutions

Secure services provisioning OTA, TSM services



We help companies leverage the mobile and contactless revolution



MobileKnowledge

Roc Boronat 117, P3M3
08018 Barcelona
(Spain)

Get in touch with us

www.themobileknowledge.com

mk@themobileknowledge.com





How to port the NFC Reader Library to K64F

Jordi Jofre (Speaker)

Angela Gemio (Host)

Thank you for your kind attention!

Please remember to fill out our **evaluation survey** (pop-up)

Check your email for **material download** and on-demand **video** addresses

Please check NXP and MobileKnowledge websites for **upcoming webinars** and **training sessions**

<http://www.nxp.com/support/classroom-training-events:CLASSROOM-TRAINING-EVENTS>

www.themobileknowledge.com/content/knowledge-catalog-0

