MCUXPRESSO IDE, SDK AND CONFIG TOOLS TRAINING

HAND ON BASED ON LPC54608

7th Jul 2017

SECURE CONNECTIONS
FOR A SMARTER WORLD

AGENDA

- **MCUXpresso Software And Tools Overview**
- **MCUXpresso SDK**
  - Web Builder
  - File Structure
- **MCUXpresso IDE**
  - Importing/Building
  - Debugging
- **MCUXpresso Config Tool**
  - Project Cloner
  - Pins Tool
  - Clocks Tool
- **LPC54608 LCD Lab, Key API and EmWin Demo**

MCUXPRESSO SOFTWARE AND TOOLS OVERVIEW

NXP
MCUXpresso
Software and Tools
COMMON TOOLKIT FOR THOUSANDS OF KINETIS® & LPC MICROCONTROLLERS
IDE
SDK
CFG
www.nxp.com/mcuxpresso

MCUXpresso Software and Tools

for Kinetis and LPC microcontrollers

MCUXpresso IDE

Edit, compile, debug and optimize in an intuitive and powerful IDE

MCUXpresso SDK

Runtime software including peripheral drivers, middleware, RTOS, demos and more

MCUXpresso Config Tools

Online and desktop tool suite for system configuration and optimization

MCUXpresso Software and Tools

- Common toolkit across Kinetis and LPC microcontrollers
- Easy to use
- High quality
- Shared software experience and broader portfolio support
- Offers easy migration and scalability
- Supports large ARM® Cortex®-M ecosystem
- Built on the 'best of' Kinetis SDK, LPCXpresso and Kinetis Design Studio IDEs

MCUXpresso Software and Tools

- IDE
- SDK
- Config Tools

For NXP's ARM® Cortex®-M controllers

- Kinetis MCUs
- LPC Microcontrollers
- i.MX Application Processors

MCUXpresso Software & Tools — Products

Integrated Development Environment (IDE)

IDE

- Offers edit, compile, debug, and many more tools with an intuitive and powerful interface
- Brings "best of" legacy IDEs (LPCXpresso and Kinetis® Design Studio) together, including GNU tool integration and library, multicore capable debugger, as well as trace functionality
- Debug connections that support all Freedom, Tower®, and LPCXpresso development boards plus industry leading commercial debug probes

Software Development Kit (SDK)

SDK

- The software framework and reference for application development with NXP's MCUs based on ARM® Cortex®-M cores
- Includes production-grade software with integrated RTOS, integrated stacks and middleware, reference software, and more
- Highest quality with MISRA compliance on all drivers; checked with Coverity® static analysis tools
- Available in custom downloads based on user selections of MCU, evaluation board, and optional software components

System Configuration Tools

CFG

- Integrated configuration and development tools for Kinetis, LPC and i.MX products
- A suite of evaluation and configuration tools that helps guide users from first evaluation to production software development
- Includes SDK builder, power estimator, pins and clocks tools
- Available in online and desktop versions

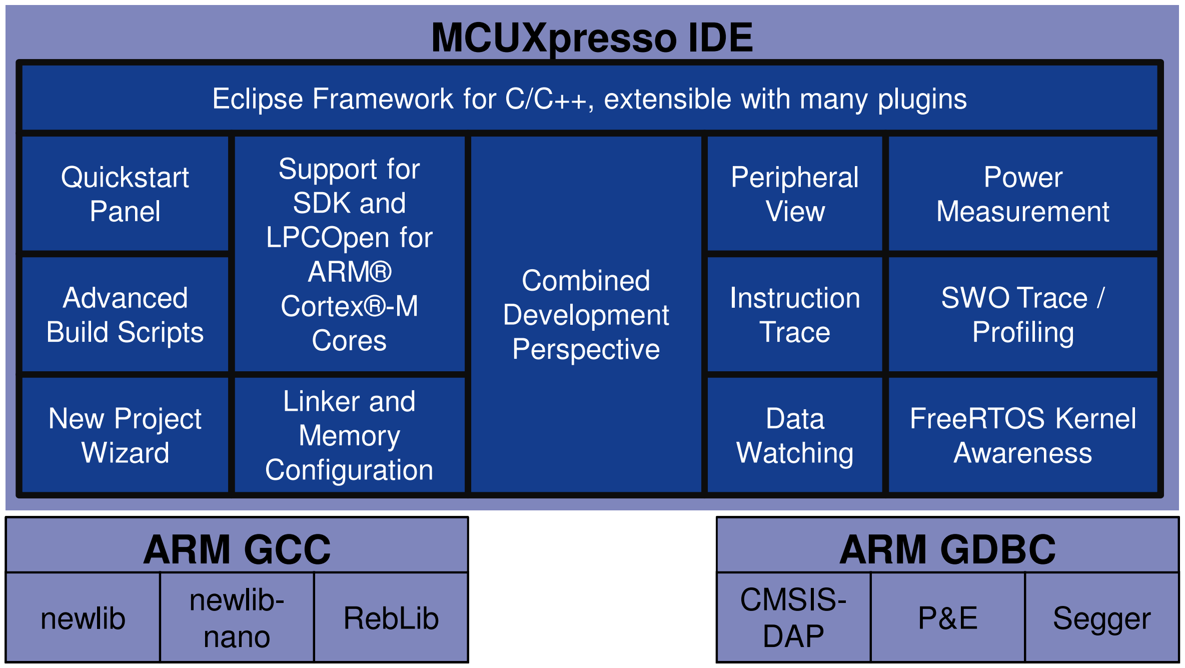
Origins of MCUXpresso Software & Tools

Kinetis and LPC SW Independent software and tools		**MCUXpresso Software and Tools** Supporting Kinetis & LPC Cortex-M MCUs
LPCXpresso IDE **& Kinetis Design Studio**	→	IDE **MCUXpresso IDE**
Kinetis SDKv2	→	SDK **MCUXpresso SDK**
Kinetis Expert	→	CFG **MCUXpresso Config Tools**

Learn more at: www.nxp.com/mcuxpresso/ide

MCUXpresso IDE

Free Eclipse and GCC-based IDE for C/C++ development on Kinetis and LPC MCUs



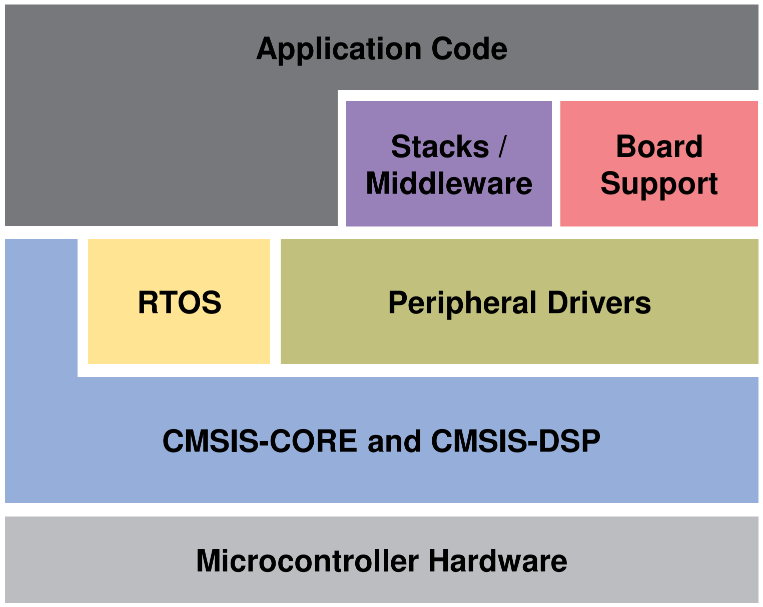
Product Features

- **Feature-rich**, **unlimited code size**, optimized for **ease-of-use**, based on **industry standard Eclipse** framework for **NXP's Kinetis** and **LPC MCUs**
- **Application development** with Eclipse and GCC-based IDE for advanced **editing**, **compiling** and **debugging**
- Supports **custom** development boards, **Freedom**, **Tower** and **LPCXpresso** boards with debug probes from **NXP**, **P&E** and **Segger**
- **Free Edition**: Full Featured, **unlimited Code Size**, no special activation needed, community based support
- **Pro Edition:** Email IDE support, **Advanced Trace Features**

Learn more at: www.nxp.com/mcuxpresso/sdk

MCUXpresso SDK

The software framework and reference for Kinetis & LPC MCU application development



Product Features

Architecture:

- CMSIS-CORE compatible
- Single driver for each peripheral
- Transactional APIs w/ optional DMA support for communication peripherals

Integrated RTOS:

- FreeRTOS v9
- RTOS-native driver wrappers

Integrated Stacks and Middleware

- USB Host, Device and OTG
- lwIP, FatFS
- Crypto acceleration plus wolfSSL & mbedTLS
- SD and eMMC card support

Reference Software:

- Peripheral driver usage examples
- Application demos
- FreeRTOS usage demos

License:

- BSD 3-clause for startup, drivers, USB stack

Toolchains:

- MCUXpresso IDE
- IAR®, ARM® Keil®, GCC w/ Cmake

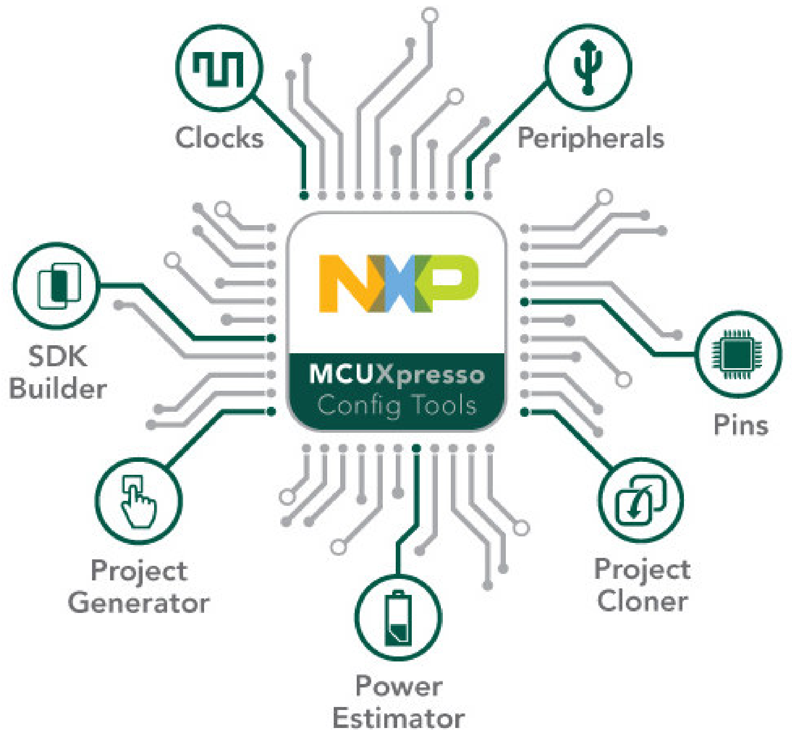
Quality

- Production-grade software
- MISRA 2004 compliance
- Checked with Coverity® static analysis tools

Learn more at: www.nxp.com/mcuxpresso/config

MCUXpresso Config Tools

Integrated configuration and development tools for LPC and Kinetis MCUs



MCUXpresso Config Tools is a suite of evaluation and configuration tools that helps guide users from first evaluation to production software development.

SDK Builder packages custom SDKs based on user selections of MCU, evaluation board, and optional software components.

Pins, Clocks, and Peripheral tools generate initialization C code for custom board support. Features validation of inputs and cross-tool conflict resolution.

Project Generator creates new SDK projects with generated Pins and Clocks source files.

Project Cloning creates a standalone SDK project based on a example application available within SDK release.

Power Estimation tool provides energy and battery-life estimates based on a user's application model.
Available as a standalone tool for select devices.

MCUXpresso Software and Tools Additional Resources

NXP
MCUXpresso
Software and Tools
COMMON TOOLKIT FOR THOUSANDS OF KINETIS® & LPC MICROCONTROLLERS
IDE
SDK
CFG
www.nxp.com/mcuxpresso

Web pages

- MCUXpresso Software and Tools – www.nxp.com/mcuxpresso
  - MCUXpresso SDK – www.nxp.com/mcuxpresso/sdk
  - MCUXpresso IDE – www.nxp.com/mcuxpresso/ide
  - MCUXpresso Config Tools – www.nxp.com/mcuxpresso/config

Supported Devices

- Supported Devices Table (Community Doc)

Communities

- MCUXpresso Software and Tools – https://community.nxp.com/community/mcuxpresso
  - MCUXpresso SDK: https://community.nxp.com/community/mcuxpresso/mcuxpresso-sdk
  - MCUXpresso IDE: https://community.nxp.com/community/mcuxpresso/mcuxpresso-ide
  - MCUXpresso Config Tools: https://community.nxp.com/community/mcuxpresso/mcuxpresso-config

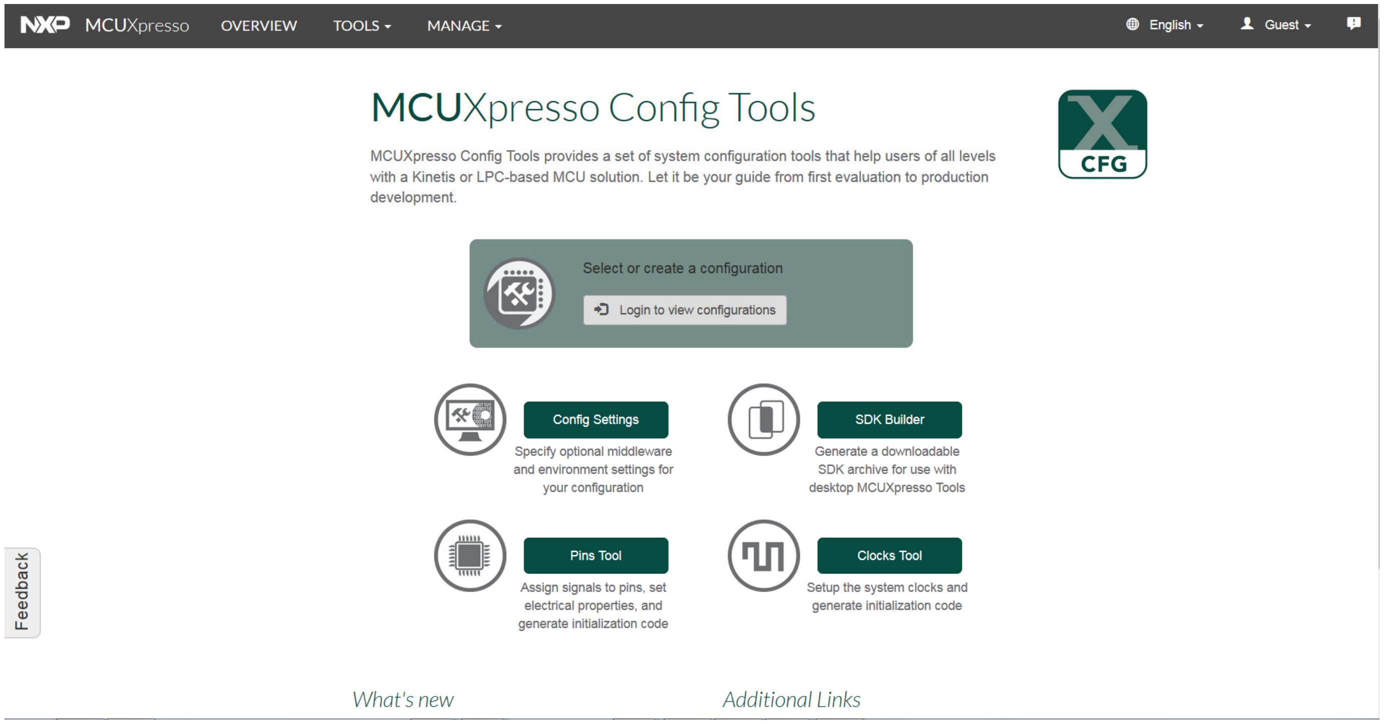
AGENDA

- **MCUXpresso Software And Tools Overview**
- **MCUXpresso SDK**
  - Web Builder
  - File Structure
- **MCUXpresso IDE**
  - Importing/Building
  - Debugging
- **MCUXpresso Config Tool**
  - Project Cloner
  - Pins Tool
  - Clocks Tool
- **LPC54608 LCD Lab, Key API and EmWin Demo**

MCUXPRESSO SDK

MCUXPRESSO SDK WEB BUILDER

MCUXpresso Homepage



MCUXpresso Config Tools

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.

Select or create a configuration

Login to view configurations

Config Settings — Specify optional middleware and environment settings for your configuration

SDK Builder — Generate a downloadable SDK archive for use with desktop MCUXpresso Tools

Pins Tool — Assign signals to pins, set electrical properties, and generate initialization code

Clocks Tool — Setup the system clocks and generate initialization code

What's new

Additional Links

https://mcuxpresso.nxp.com/en/welcome

Configuration

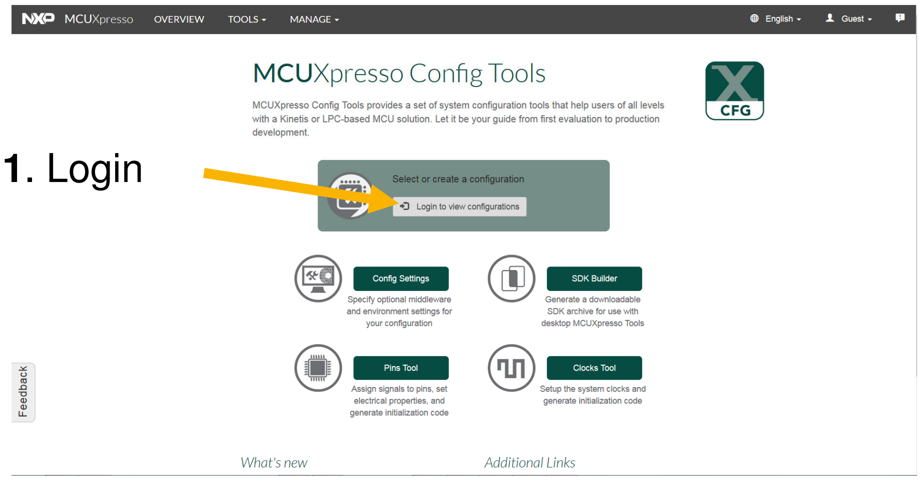
- What is a configuration?
  - A group of configured settings used across the MCUXpresso configuration tools (SDK builder, Pins, and Clocks)

- What is included in a configuration?
  - SDK builder configuration settings (e.g. Board/Processor, Toolchain, Host OS, etc.)
  - Pin assignments in the Pins Tool
  - Clock initializations in the Clocks Tool

- Configurations can be saved and shared as a .mex file

Get Started

1. Login



MCUXpresso Config Tools

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.

Specify optional middleware and environment settings for your configuration

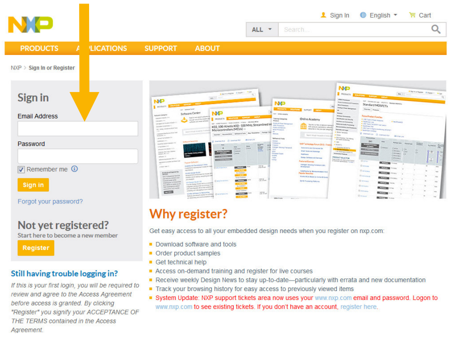
Generate a downloadable SDK archive for use with desktop MCUXpresso Tools

Assign signals to pins, set electrical properties, and generate initialization code

Setup the system clocks and generate initialization code

Routed to nxp.com

2. Enter account info



Not yet registered?
Start here to become a new member

Still having trouble logging in?

If this is your first login, you will be required to review and agree to the Access Agreement before access is granted. By clicking "Register" you signify your ACCEPTANCE OF THE TERMS contained in the Access Agreement.

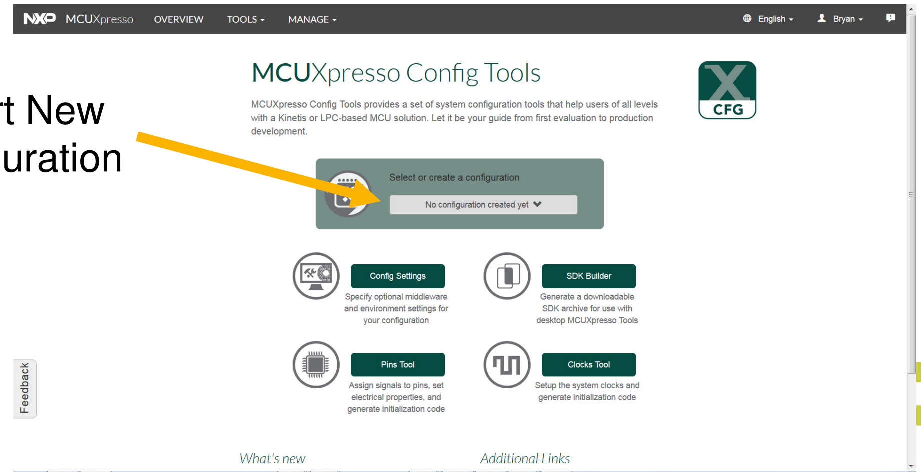
Why register?

Get easy access to all your embedded design needs when you register on nxp.com:

- Download software and tools
- Order product samples
- Get technical help
- Access on-demand training and register for live courses
- Receive weekly Design News to stay up-to-date—particularly with errata and new documentation
- Track your browsing history for easy access to previously viewed items
- System Update: NXP support tickets area now uses your www.nxp.com email and password. Logon to www.nxp.com to see existing tickets. If you don't have an account, register here.

Return to mcuxpresso.nxp.com

3. Start New Configuration



MCUXpresso Config Tools

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.

Specify optional middleware and environment settings for your configuration

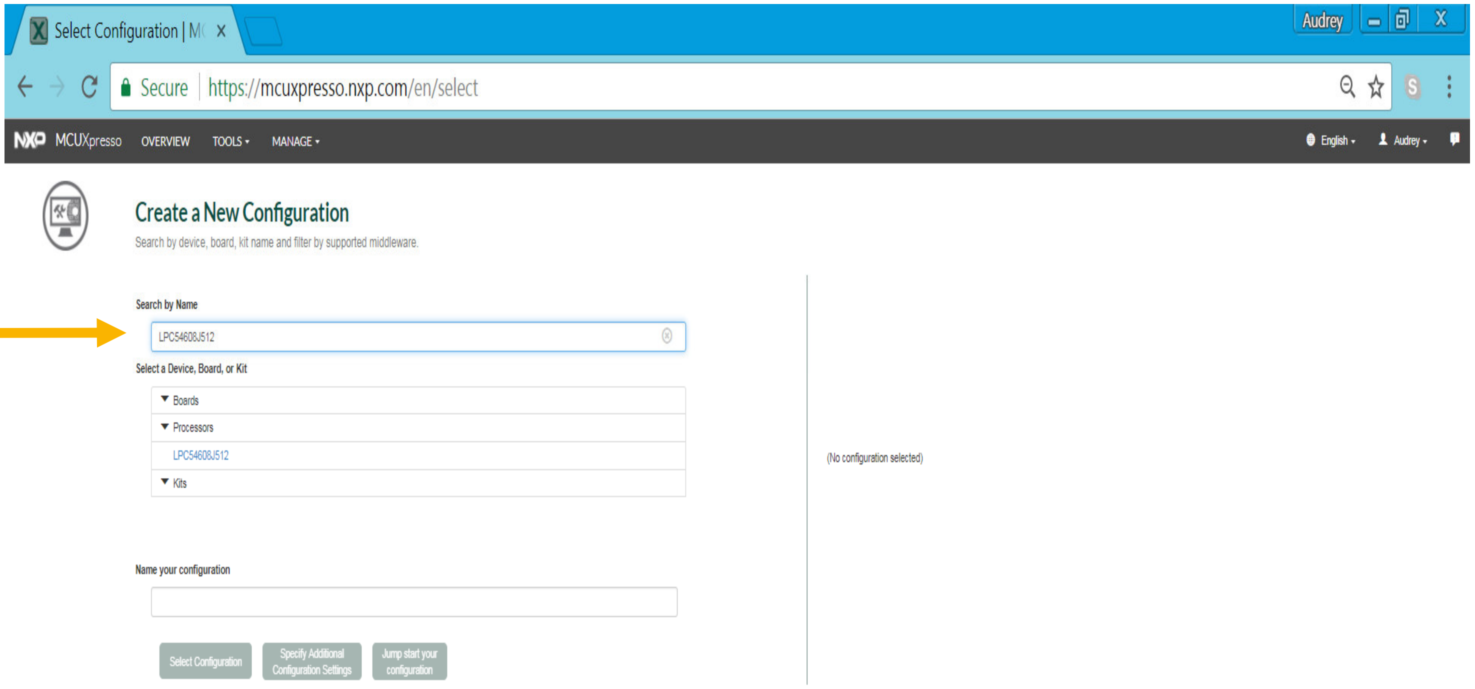
Generate a downloadable SDK archive for use with desktop MCUXpresso Tools

Assign signals to pins, set electrical properties, and generate initialization code

Setup the system clocks and generate initialization code

Create a New Configuration (1/3)

1. Type in search


Select Configuration | M
Audrey
Secure
https://mcuxpresso.nxp.com/en/select
MCUXpresso
OVERVIEW
TOOLS
MANAGE
English
Audrey

Create a New Configuration

Search by device, board, kit name and filter by supported middleware.

Search by Name

LPC54608J512

Select a Device, Board, or Kit

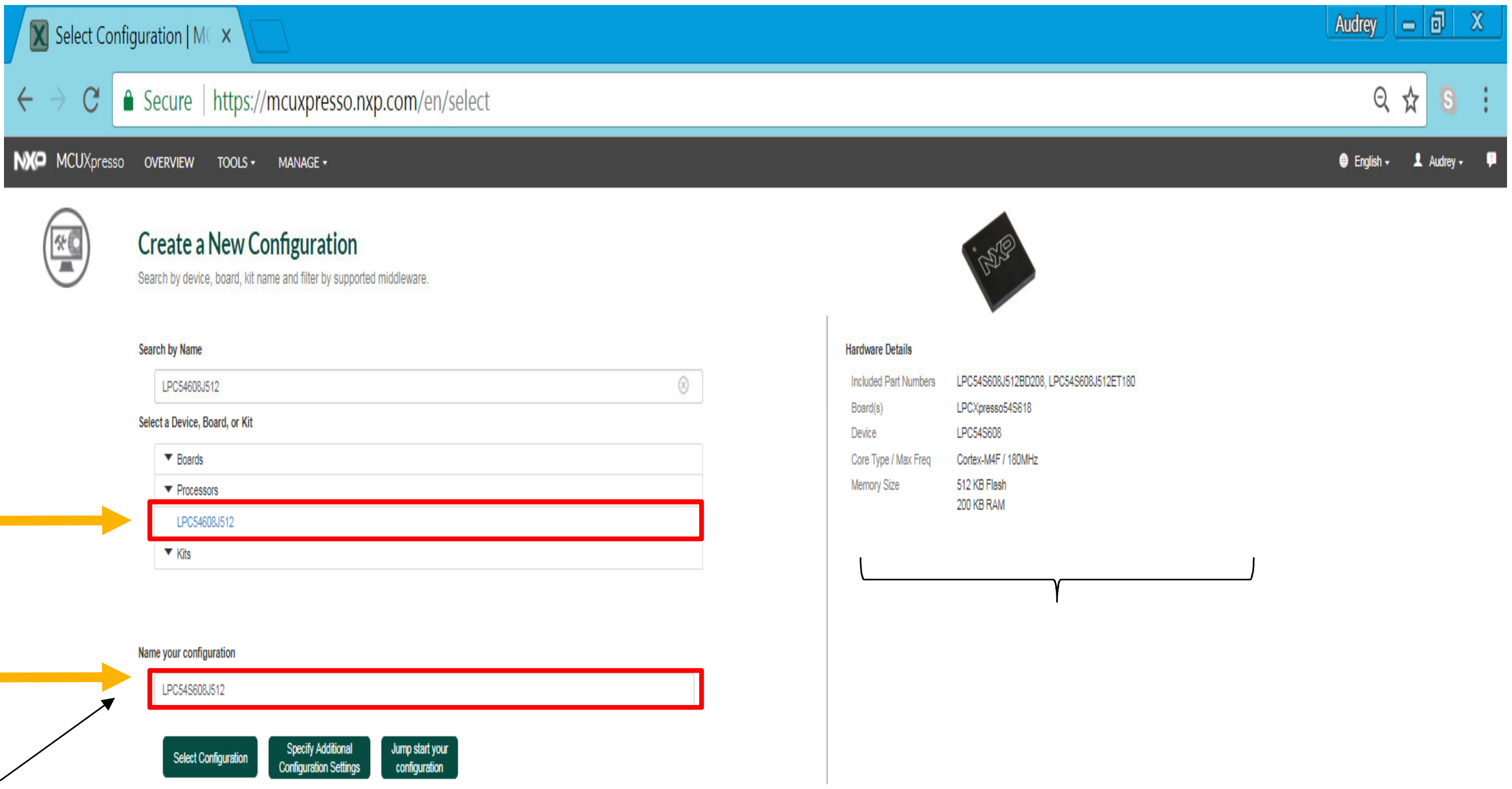
- Boards
- Processors
  - LPC54608J512
- Kits

Name your configuration

(No configuration selected)

Select Configuration
Specify Additional Configuration Settings
Jump start your configuration

Create a New Configuration (2/3)


Select Configuration | M
Audrey
Secure
https://mcuxpresso.nxp.com/en/select
MCUXpresso
OVERVIEW
TOOLS
MANAGE
English
Audrey
Select Configuration
Specify Additional Configuration Settings
Jump start your configuration

Create a New Configuration

Search by device, board, kit name and filter by supported middleware.

Search by Name

LPC54608J512

Select a Device, Board, or Kit

▼ Boards

▼ Processors

LPC54608J512

▼ Kits

Name your configuration

LPC54S608J512

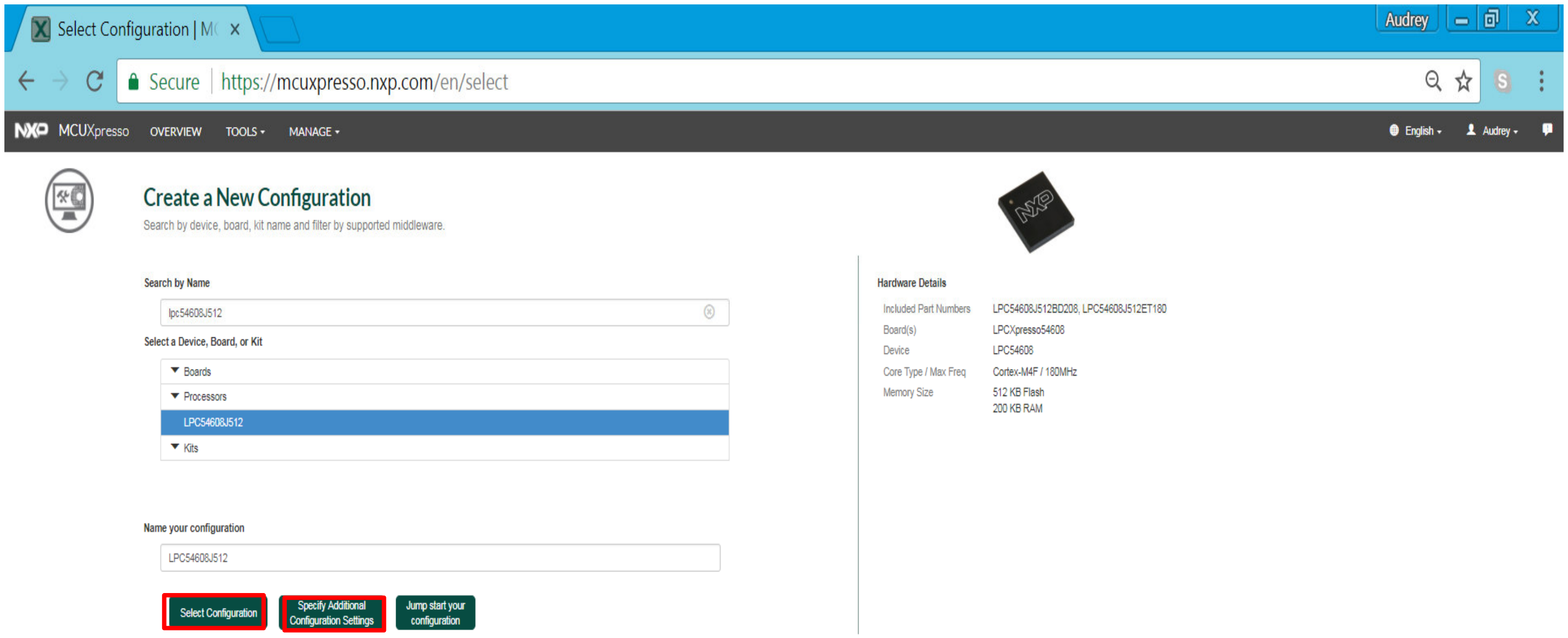
Hardware Details

Included Part Numbers	LPC54S608J512BD208, LPC54S608J512ET180
Board(s)	LPCXpresso54S618
Device	LPC54S608
Core Type / Max Freq	Cortex-M4F / 180MHz
Memory Size	512 KB Flash 200 KB RAM

2. Make selection

Configuration name automatically assigned. Name can be modified

Create a New Configuration (3/3)


Select Configuration | MC
Audrey
Secure
https://mcuxpresso.nxp.com/en/select
MCUXpresso
OVERVIEW
TOOLS
MANAGE
English
Audrey

Create a New Configuration

Search by device, board, kit name and filter by supported middleware.

Search by Name

lpc54608J512

Select a Device, Board, or Kit

- Boards
- Processors
  - LPC54608J512
- Kits

Name your configuration

LPC54608J512

Select Configuration
Specify Additional Configuration Settings
Jump start your configuration

Hardware Details

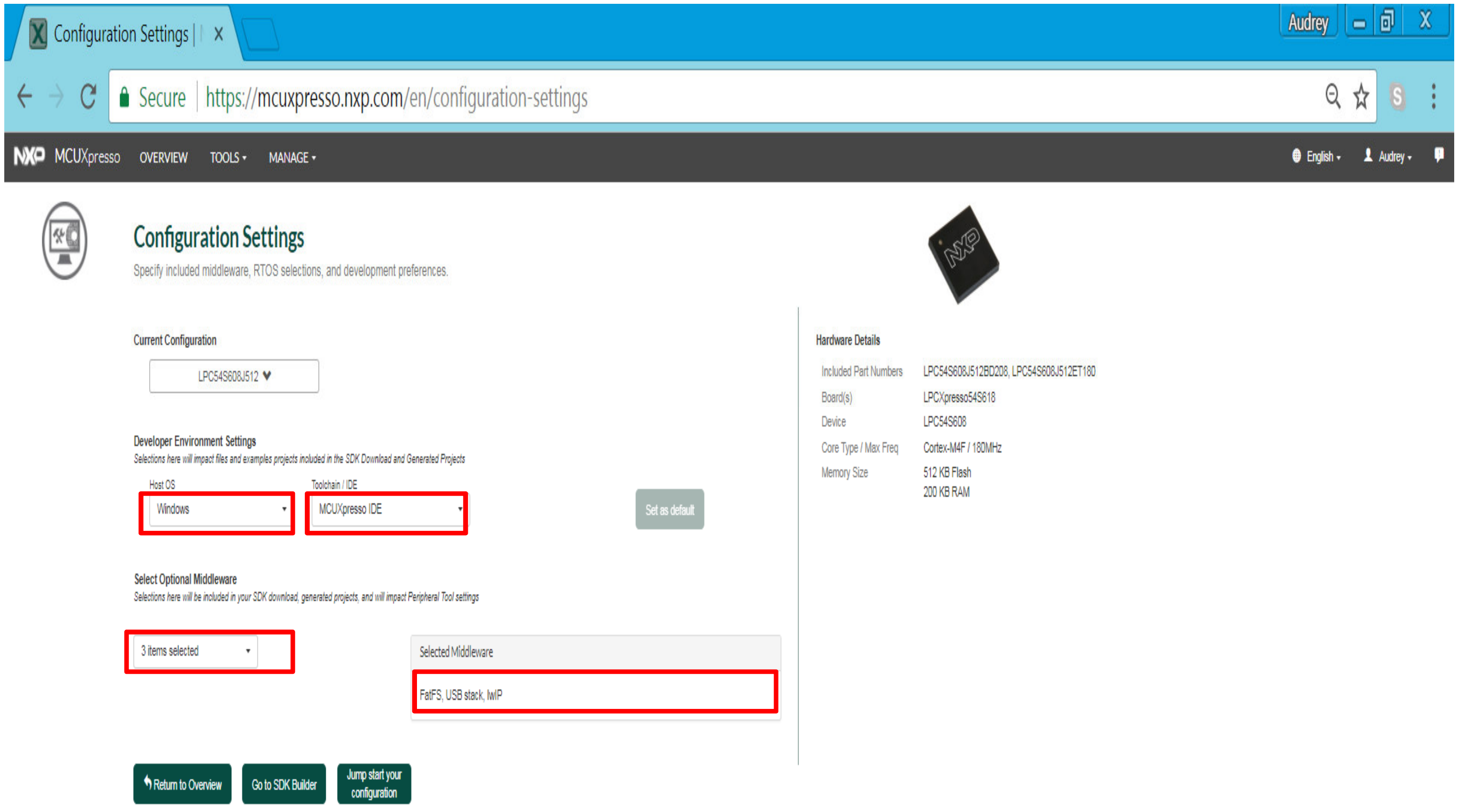
Included Part Numbers	LPC54608J512BD208, LPC54608J512ET180
Board(s)	LPCXpresso54608
Device	LPC54608
Core Type / Max Freq	Cortex-M4F / 180MHz
Memory Size	512 KB Flash 200 KB RAM

- <u>Select Configuration</u>
  - Proceed to builder with default options selected for toolchain, OS, and middleware
- <u>Specify Additional Configuration Settings</u>
  - Select toolchain, OS, and middleware other than default.

Additional Configuration Settings

- User selects:
  - Host OS
  - Toolchain/IDE
  - Middleware

- Defaults (first session):
  - Host OS -> **Windows**
  - IDE -> **MCUXpresso**
  - Middleware -> **FatFS, USB Stack***, **lwIP***


Configuration Settings | X
Audrey
Secure https://mcuxpresso.nxp.com/en/configuration-settings
MCUXpresso OVERVIEW TOOLS MANAGE
English Audrey
Set as default
Return to Overview
Go to SDK Builder
Jump start your configuration

Configuration Settings

Specify included middleware, RTOS selections, and development preferences.

Current Configuration

LPC54S608J512

Developer Environment Settings

Selections here will impact files and examples projects included in the SDK Download and Generated Projects

Host OS: Windows

Toolchain / IDE: MCUXpresso IDE

Select Optional Middleware

Selections here will be included in your SDK download, generated projects, and will impact Peripheral Tool settings

3 items selected

Selected Middleware

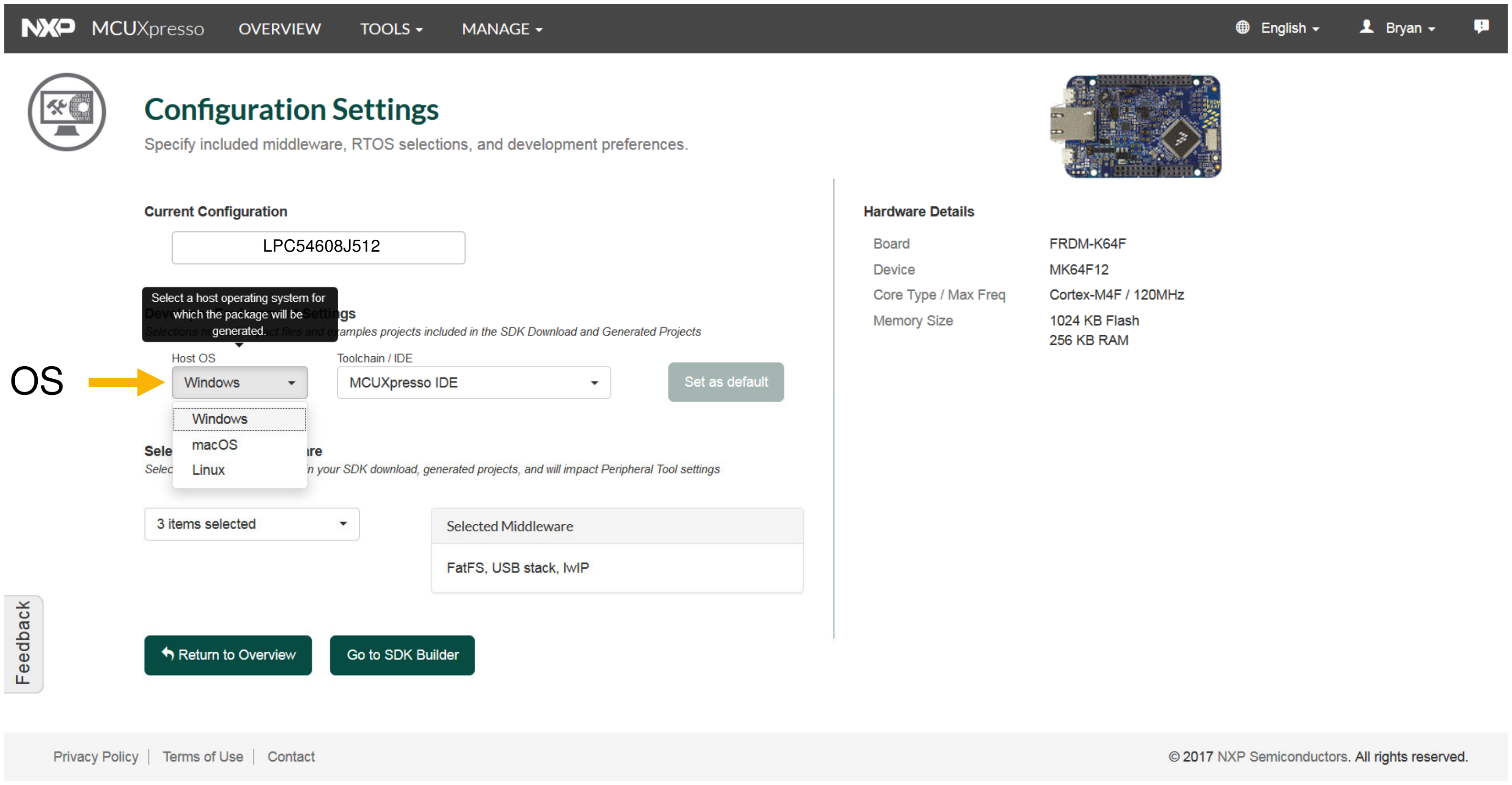
FatFS, USB stack, lwIP

Hardware Details

Included Part Numbers	LPC54S608J512BD208, LPC54S608J512ET180
Board(s)	LPCXpresso54S618
Device	LPC54S608
Core Type / Max Freq	Cortex-M4F / 180MHz
Memory Size	512 KB Flash 200 KB RAM

Additional Configuration Settings: Choose an OS

Select Host OS



Configuration Settings

Specify included middleware, RTOS selections, and development preferences.

Current Configuration

LPC54608J512

Select a host operating system for which the package will be generated.

Developer Settings

Selections for project files and examples projects included in the SDK Download and Generated Projects

Host OS: Windows (Windows / macOS / Linux)

Toolchain / IDE: MCUXpresso IDE

Set as default

Select Middleware

Select... in your SDK download, generated projects, and will impact Peripheral Tool settings

3 items selected

Selected Middleware: FatFS, USB stack, lwIP

Return to Overview | Go to SDK Builder

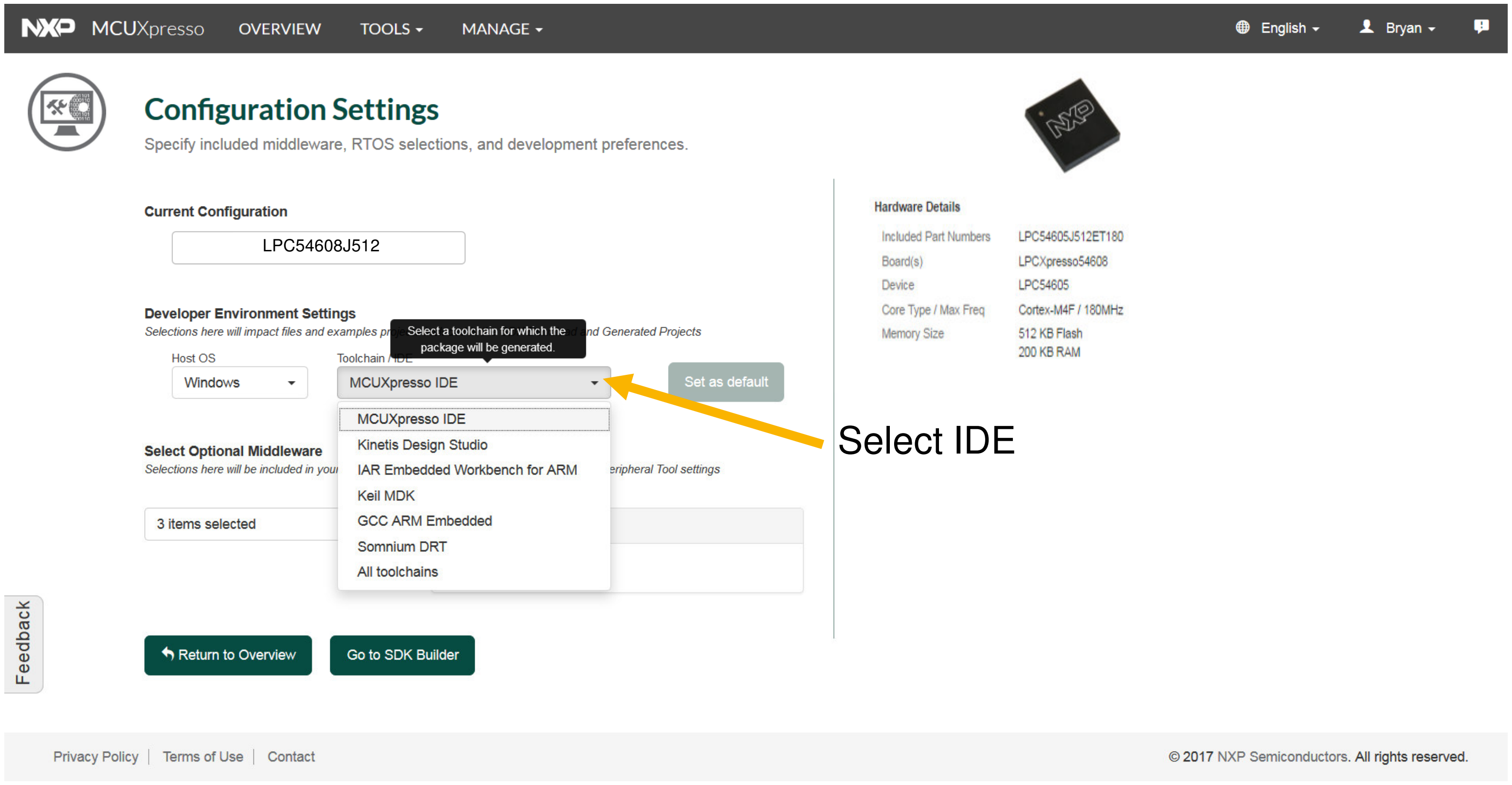
Hardware Details

Board	FRDM-K64F
Device	MK64F12
Core Type / Max Freq	Cortex-M4F / 120MHz
Memory Size	1024 KB Flash 256 KB RAM

Privacy Policy | Terms of Use | Contact

© 2017 NXP Semiconductors. All rights reserved.

Additional Configuration Settings: Choose an IDE



NXP MCUXpresso OVERVIEW TOOLS MANAGE English Bryan

Configuration Settings

Specify included middleware, RTOS selections, and development preferences.

Current Configuration

LPC54608J512

Developer Environment Settings

Selections here will impact files and examples pr... ...nd Generated Projects

Select a toolchain for which the package will be generated.

Host OS: Windows

Toolchain / IDE: MCUXpresso IDE

Set as default

- MCUXpresso IDE
- Kinetis Design Studio
- IAR Embedded Workbench for ARM
- Keil MDK
- GCC ARM Embedded
- Somnium DRT
- All toolchains

Select Optional Middleware

Selections here will be included in your ... ...eripheral Tool settings

3 items selected

Return to Overview | Go to SDK Builder

Hardware Details

Included Part Numbers	LPC54605J512ET180
Board(s)	LPCXpresso54608
Device	LPC54605
Core Type / Max Freq	Cortex-M4F / 180MHz
Memory Size	512 KB Flash 200 KB RAM

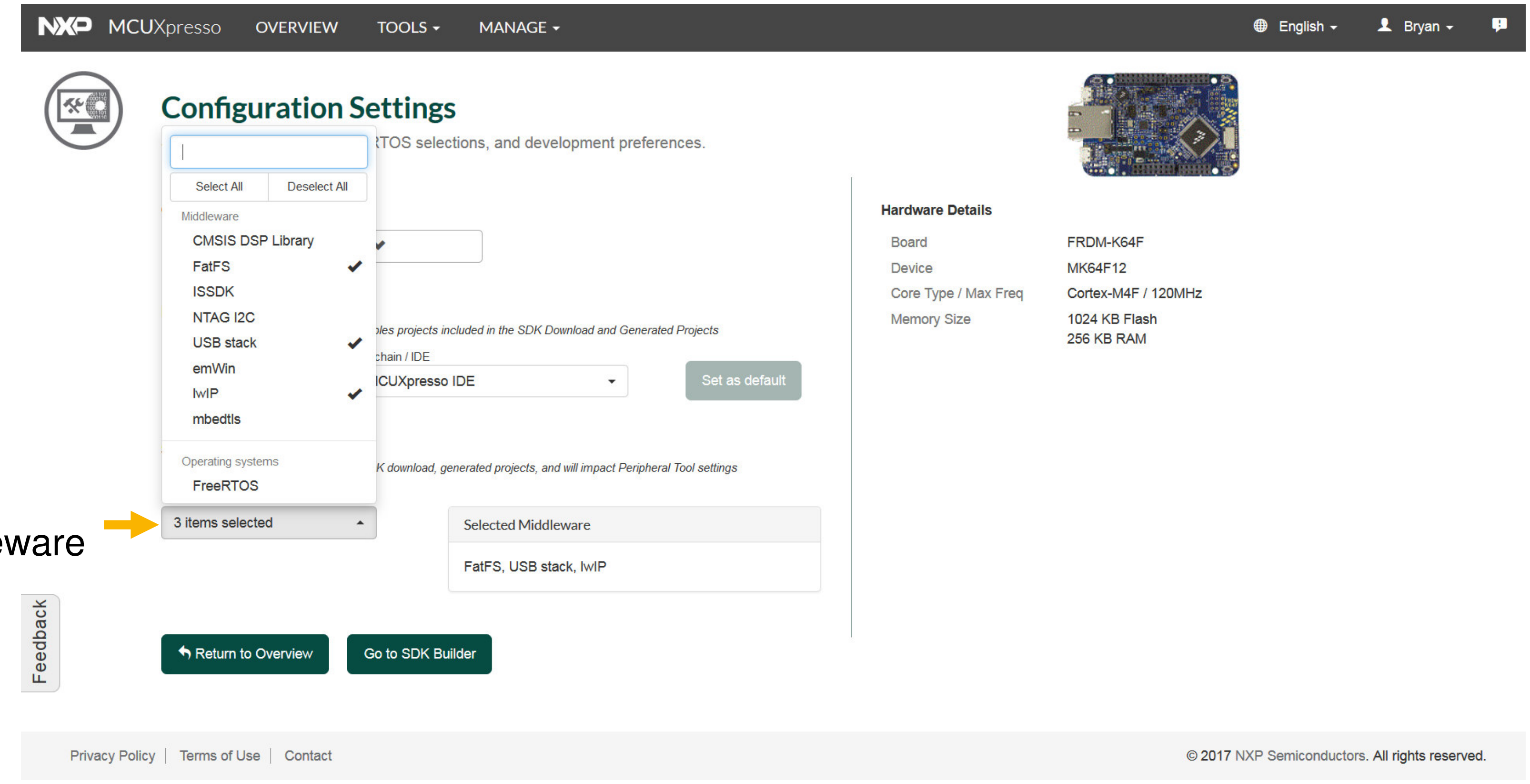
Feedback

Privacy Policy | Terms of Use | Contact

© 2017 NXP Semiconductors. All rights reserved.

Select IDE

Additional Configuration Settings: Choose Middleware/RTOS



NXP MCUXpresso OVERVIEW TOOLS MANAGE English Bryan

Configuration Settings

RTOS selections, and development preferences.

Select All | Deselect All

Middleware
- CMSIS DSP Library
- FatFS ✓
- ISSDK
- NTAG I2C
- USB stack ✓
- emWin
- lwIP ✓
- mbedtls

Operating systems
- FreeRTOS

3 items selected

oles projects included in the SDK Download and Generated Projects

chain / IDE: CUXpresso IDE — Set as default

K download, generated projects, and will impact Peripheral Tool settings

Selected Middleware
FatFS, USB stack, lwIP

Return to Overview | Go to SDK Builder

Feedback

Hardware Details

Board	FRDM-K64F
Device	MK64F12
Core Type / Max Freq	Cortex-M4F / 120MHz
Memory Size	1024 KB Flash 256 KB RAM

Privacy Policy | Terms of Use | Contact

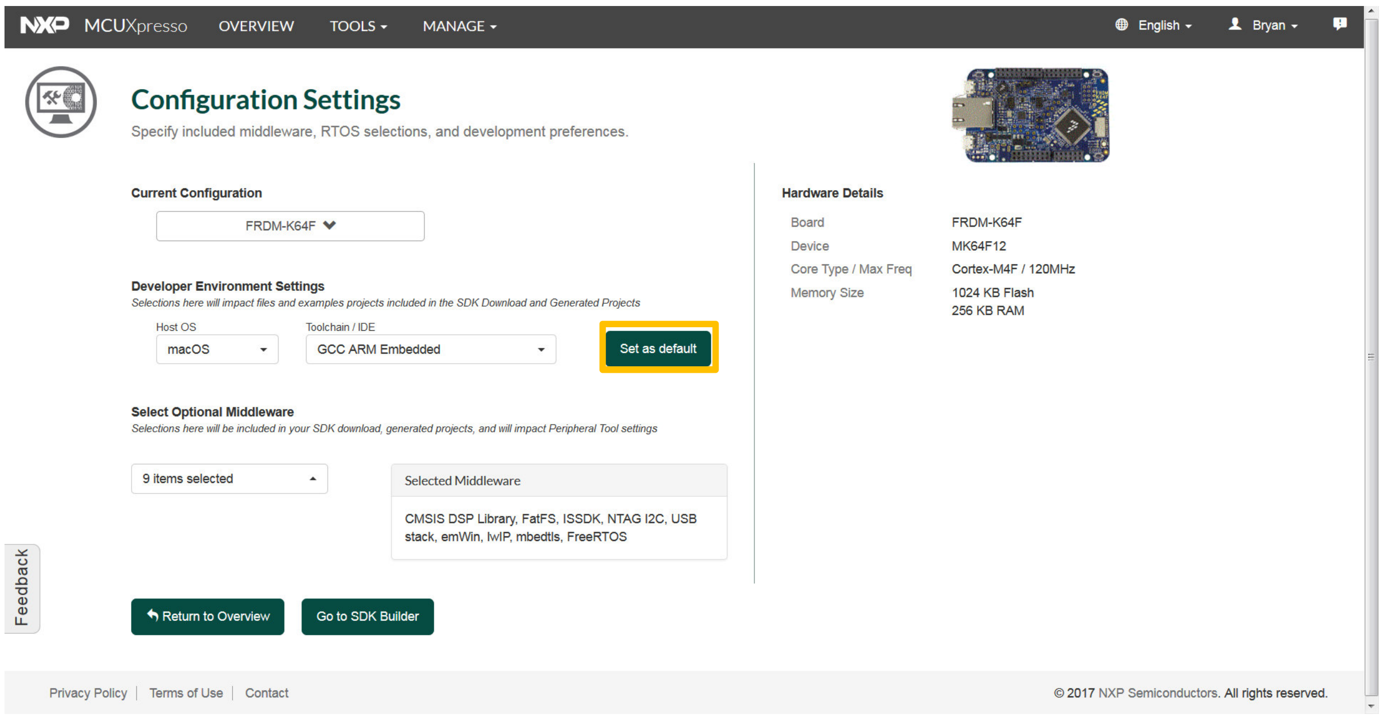
© 2017 NXP Semiconductors. All rights reserved.

Select Middleware

Change Default Build Settings

- Select “Set as Default” to save Host OS and Toolchain to preferences

- Future configurations will use these build settings as defaults



Configuration Settings

Specify included middleware, RTOS selections, and development preferences.

Current Configuration

FRDM-K64F

Developer Environment Settings

Selections here will impact files and examples projects included in the SDK Download and Generated Projects

Host OS: macOS

Toolchain / IDE: GCC ARM Embedded

Set as default

Select Optional Middleware

Selections here will be included in your SDK download, generated projects, and will impact Peripheral Tool settings

9 items selected

Selected Middleware

CMSIS DSP Library, FatFS, ISSDK, NTAG I2C, USB stack, emWin, lwIP, mbedtls, FreeRTOS

Hardware Details

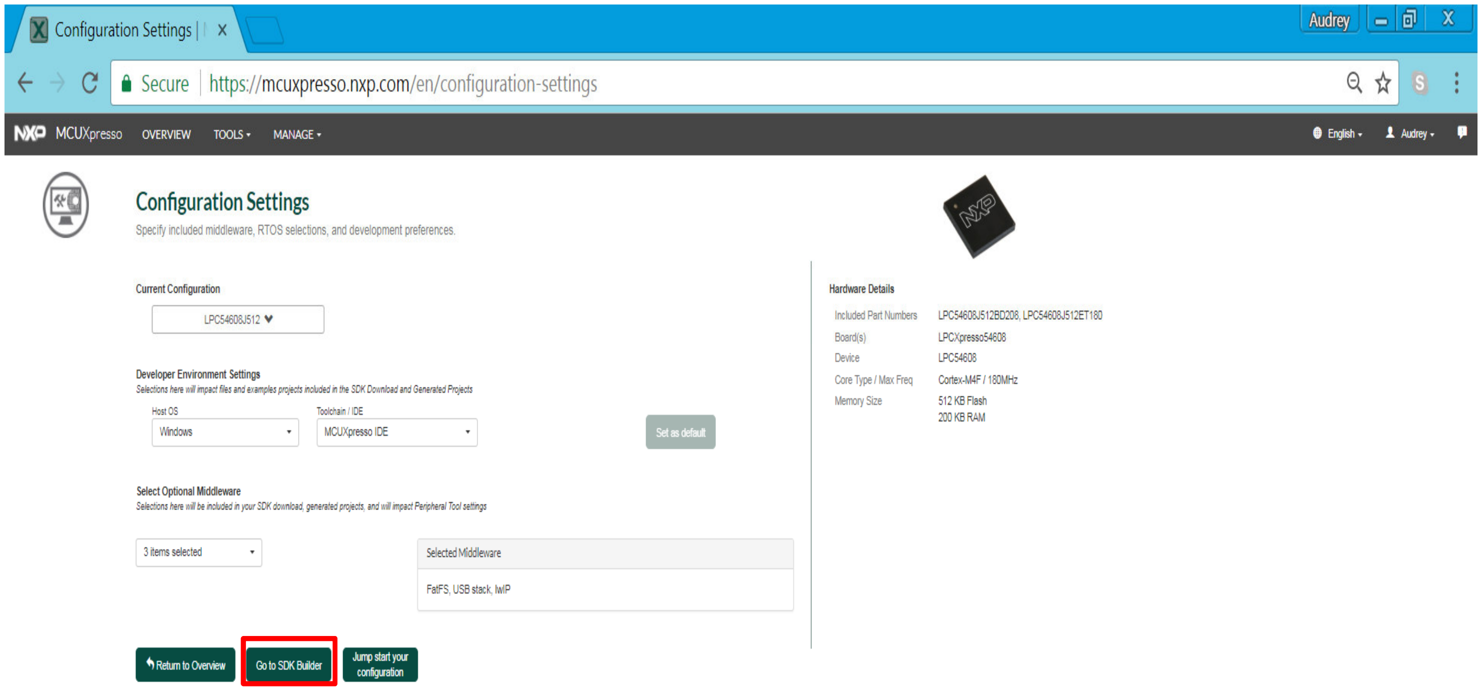
Board	FRDM-K64F
Device	MK64F12
Core Type / Max Freq	Cortex-M4F / 120MHz
Memory Size	1024 KB Flash 256 KB RAM

Return to Overview | Go to SDK Builder

Privacy Policy | Terms of Use | Contact

© 2017 NXP Semiconductors. All rights reserved.

Finish Settings



Configuration Settings

Specify included middleware, RTOS selections, and development preferences.

Current Configuration

LPC54608J512

Developer Environment Settings

Selections here will impact files and examples projects included in the SDK Download and Generated Projects

Host OS: Windows

Toolchain / IDE: MCUXpresso IDE

Select Optional Middleware

Selections here will be included in your SDK download, generated projects, and will impact Peripheral Tool settings

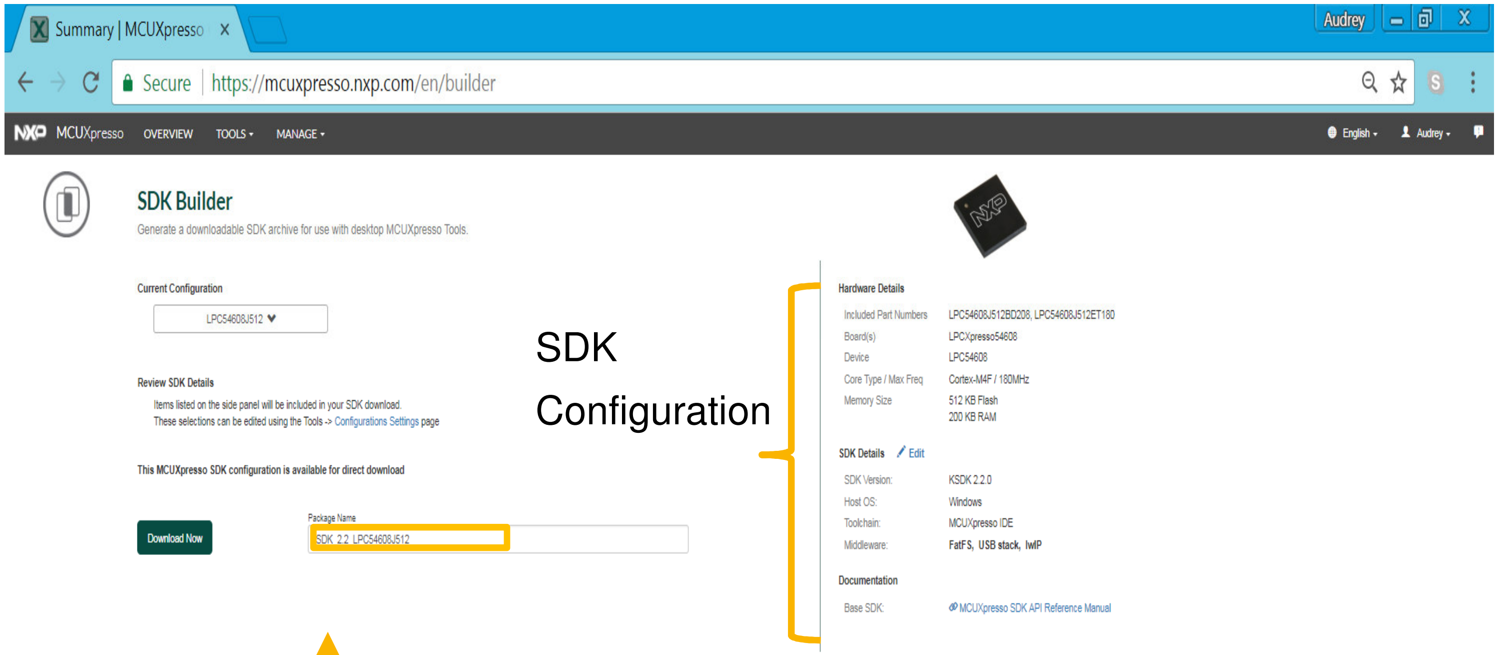
3 items selected

Selected Middleware: FatFS, USB stack, lwIP

Hardware Details

Included Part Numbers	LPC54608J512BD208, LPC54608J512ET180
Board(s)	LPCXpresso54608
Device	LPC54608
Core Type / Max Freq	Cortex-M4F / 180MHz
Memory Size	512 KB Flash 200 KB RAM

Build SDK



SDK Builder

Generate a downloadable SDK archive for use with desktop MCUXpresso Tools.

Current Configuration

LPC54608J512

Review SDK Details

Items listed on the side panel will be included in your SDK download.
These selections can be edited using the Tools -> Configurations Settings page

This MCUXpresso SDK configuration is available for direct download

Download Now

Package Name: SDK_2.2_LPC54608J512

Hardware Details

Included Part Numbers	LPC54608J512BD208, LPC54608J512ET180
Board(s)	LPCXpresso54608
Device	LPC54608
Core Type / Max Freq	Cortex-M4F / 180MHz
Memory Size	512 KB Flash 200 KB RAM

SDK Details Edit

SDK Version:	KSDK 2.2.0
Host OS:	Windows
Toolchain:	MCUXpresso IDE
Middleware:	FatFS, USB stack, lwIP

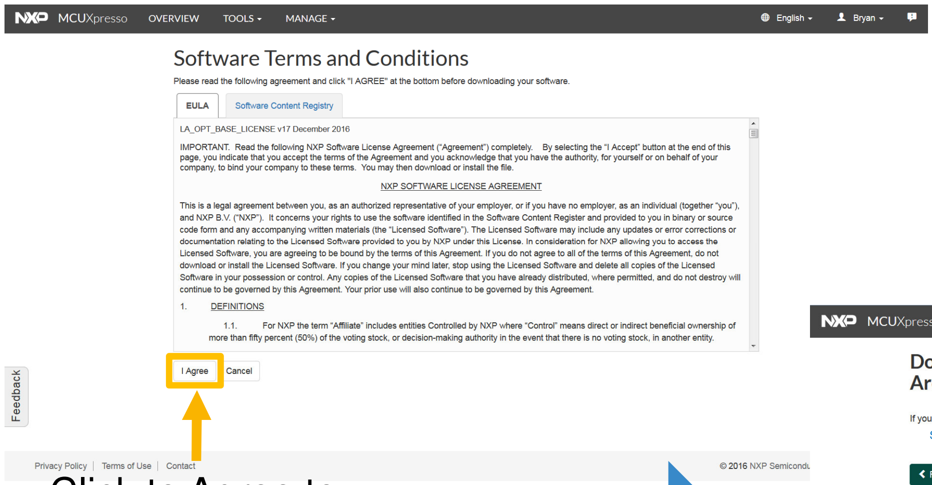
Documentation

Base SDK:	MCUXpresso SDK API Reference Manual

SDK Configuration

After reviewing configuration, click to download SDK

Download SDK



Software Terms and Conditions

Please read the following agreement and click "I AGREE" at the bottom before downloading your software.

LA_OPT_BASE_LICENSE v17 December 2016

IMPORTANT. Read the following NXP Software License Agreement ("Agreement") completely. By selecting the "I Accept" button at the end of this page, you indicate that you accept the terms of the Agreement and you acknowledge that you have the authority, for yourself or on behalf of your company, to bind your company to these terms. You may then download or install the file.

NXP SOFTWARE LICENSE AGREEMENT

This is a legal agreement between you, as an authorized representative of your employer, or if you have no employer, as an individual (together "you"), and NXP B.V. ("NXP"). It concerns your rights to use the software identified in the Software Content Register and provided to you in binary or source code form and any accompanying written materials (the "Licensed Software"). The Licensed Software may include any updates or error corrections or documentation relating to the Licensed Software provided to you by NXP under this License. In consideration for NXP allowing you to access the Licensed Software, you are agreeing to be bound by the terms of this Agreement. If you do not agree to all of the terms of this Agreement, do not download or install the Licensed Software. If you change your mind later, stop using the Licensed Software and delete all copies of the Licensed Software in your possession or control. Any copies of the Licensed Software that you have already distributed, where permitted, and do not destroy will continue to be governed by this Agreement. Your prior use will also continue to be governed by this Agreement.

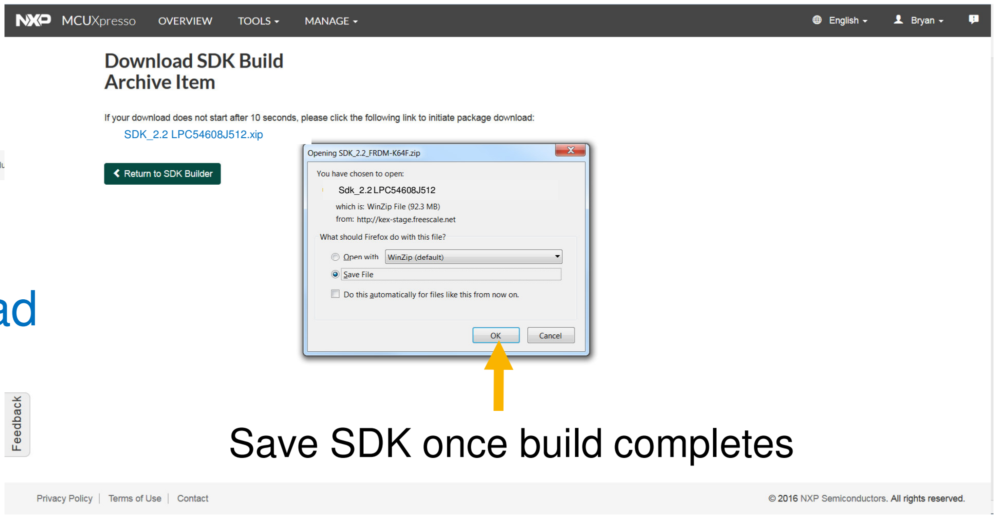
1. DEFINITIONS

1.1. For NXP the term "Affiliate" includes entities Controlled by NXP where "Control" means direct or indirect beneficial ownership of more than fifty percent (50%) of the voting stock, or decision-making authority in the event that there is no voting stock, in another entity.

Click to Agree to terms and conditions

SDK_2.2_LPC54608J512.zip

SDK download will begin



Download SDK Build Archive Item

If your download does not start after 10 seconds, please click the following link to initiate package download:

SDK_2.2 LPC54608J512.xip

Opening SDK_2.2_FRDM-K64F.zip

You have chosen to open:

Sdk_2.2 LPC54608J512

which is: WinZip File (92.3 MB)

from: http://kex-stage.freescale.net

What should Firefox do with this file?

Open with WinZip (default)

[✓ Save File]

Do this automatically for files like this from now on.

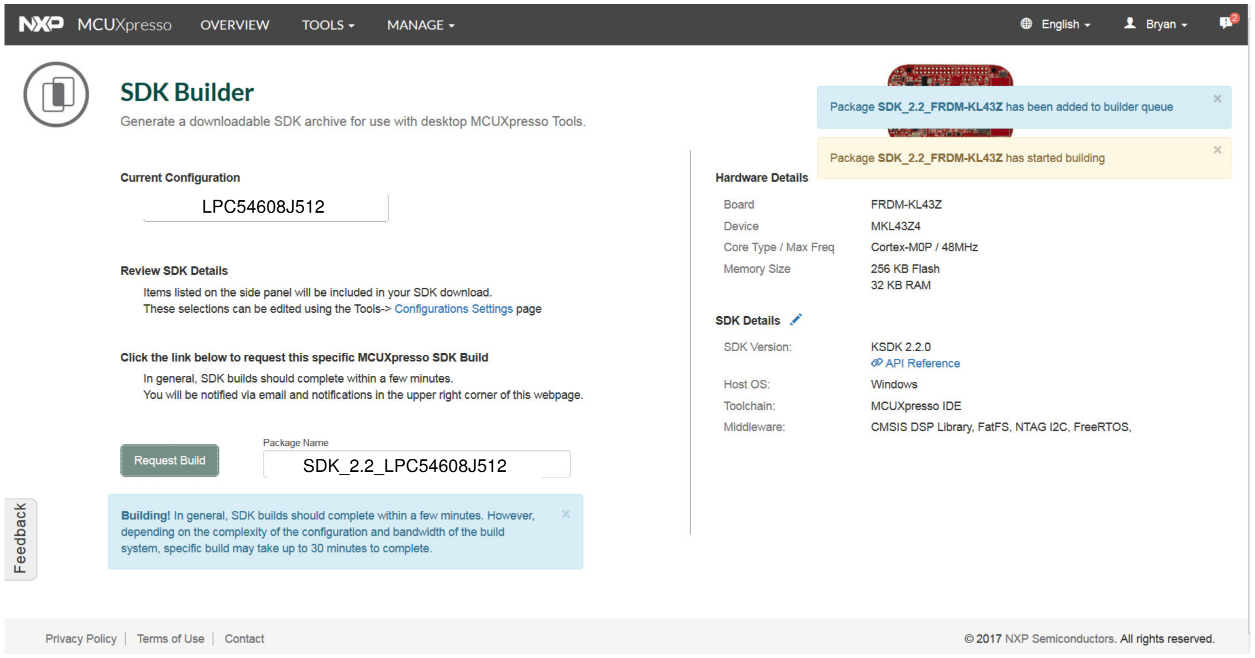
© 2016 NXP Semiconductors. All rights reserved.

Save SDK once build completes

Request Build

- In some occasions, if the SDK configuration has not previously been built, “Request Build” will be displayed in place of “Download Now”

- An email notification with direct link will be sent once the build is finished



SDK Builder

Generate a downloadable SDK archive for use with desktop MCUXpresso Tools.

Package **SDK_2.2_FRDM-KL43Z** has been added to builder queue

Package **SDK_2.2_FRDM-KL43Z** has started building

Current Configuration

LPC54608J512

Review SDK Details

Items listed on the side panel will be included in your SDK download.
These selections can be edited using the Tools-> Configurations Settings page

Click the link below to request this specific MCUXpresso SDK Build

In general, SDK builds should complete within a few minutes.
You will be notified via email and notifications in the upper right corner of this webpage.

Request Build

Package Name: SDK_2.2_LPC54608J512

Building! In general, SDK builds should complete within a few minutes. However, depending on the complexity of the configuration and bandwidth of the build system, specific build may take up to 30 minutes to complete.

Hardware Details

Board	FRDM-KL43Z
Device	MKL43Z4
Core Type / Max Freq	Cortex-M0P / 48MHz
Memory Size	256 KB Flash 32 KB RAM

SDK Details

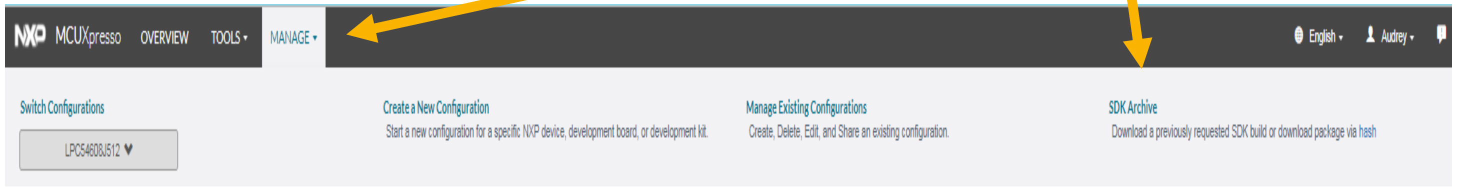
SDK Version:	KSDK 2.2.0 API Reference
Host OS:	Windows
Toolchain:	MCUXpresso IDE
Middleware:	CMSIS DSP Library, FatFS, NTAG I2C, FreeRTOS,

Privacy Policy | Terms of Use | Contact

© 2017 NXP Semiconductors. All rights reserved.

Build Archive

Access SDK Archive from Manage menu

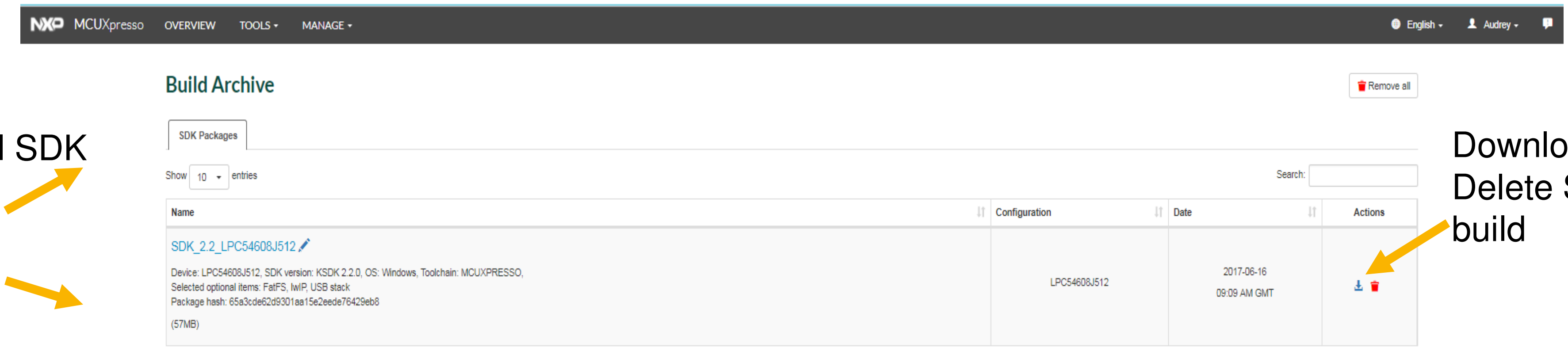


Create a New Configuration
Start a new configuration for a specific NXP device, development board, or development kit.

Manage Existing Configurations
Create, Delete, Edit, and Share an existing configuration.

SDK Archive
Download a previously requested SDK build or download package via hash

Shows all SDK builds

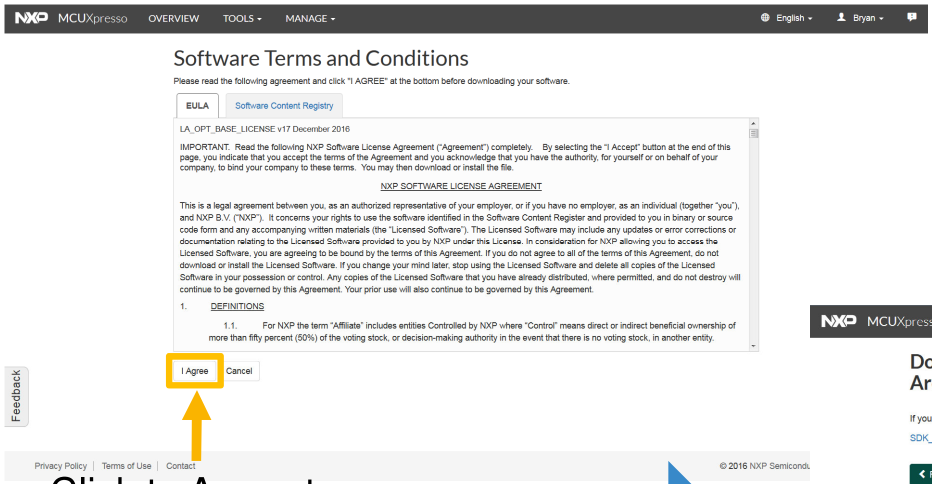


Build Archive

Name	Configuration	Date	Actions
SDK_2.2_LPC54608J512 Device: LPC54608J512, SDK version: KSDK 2.2.0, OS: Windows, Toolchain: MCUXPRESSO, Selected optional items: FatFS, lwIP, USB stack Package hash: 65a3cde62d9301aa15e2eede76429eb8 (57MB)	LPC54608J512	2017-06-16 09:09 AM GMT	

Download/ Delete SDK build

Download SDK



Software Terms and Conditions

Please read the following agreement and click "I AGREE" at the bottom before downloading your software.

EULA | Software Content Registry

LA_OPT_BASE_LICENSE v17 December 2016

IMPORTANT. Read the following NXP Software License Agreement ("Agreement") completely. By selecting the "I Accept" button at the end of this page, you indicate that you accept the terms of the Agreement and you acknowledge that you have the authority, for yourself or on behalf of your company, to bind your company to these terms. You may then download or install the file.

NXP SOFTWARE LICENSE AGREEMENT

This is a legal agreement between you, as an authorized representative of your employer, or if you have no employer, as an individual (together "you"), and NXP B.V. ("NXP"). It concerns your rights to use the software identified in the Software Content Register and provided to you in binary or source code form and any accompanying written materials (the "Licensed Software"). The Licensed Software may include any updates or error corrections or documentation relating to the Licensed Software provided to you by NXP under this License. In consideration for NXP allowing you to access the Licensed Software, you are agreeing to be bound by the terms of this Agreement. If you do not agree to all of the terms of this Agreement, do not download or install the Licensed Software. If you change your mind later, stop using the Licensed Software and delete all copies of the Licensed Software in your possession or control. Any copies of the Licensed Software that you have already distributed, where permitted, and do not destroy will continue to be governed by this Agreement. Your prior use will also continue to be governed by this Agreement.

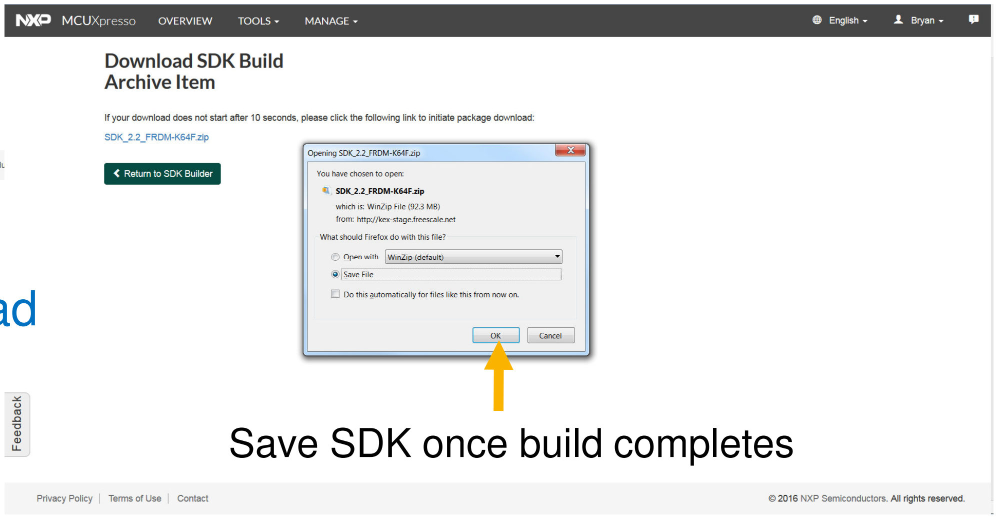
1. DEFINITIONS

1.1. For NXP the term "Affiliate" includes entities Controlled by NXP where "Control" means direct or indirect beneficial ownership of more than fifty percent (50%) of the voting stock, or decision-making authority in the event that there is no voting stock, in another entity.

I Agree | Cancel

Click to Agree to terms and conditions

SDK download will begin


MCUXpresso
OVERVIEW
TOOLS
MANAGE
English
Bryan
Return to SDK Builder
Feedback
Privacy Policy | Terms of Use | Contact
© 2016 NXP Semiconductors. All rights reserved.

Download SDK Build Archive Item

If your download does not start after 10 seconds, please click the following link to initiate package download:

SDK_2.2_FRDM-K64F.zip

Opening SDK_2.2_FRDM-K64F.zip

You have chosen to open:

SDK_2.2_FRDM-K64F.zip

which is: WinZip File (92.3 MB)

from: http://kex-stage.freescale.net

What should Firefox do with this file?

○ Open with WinZip (default)

[✓ Save File]

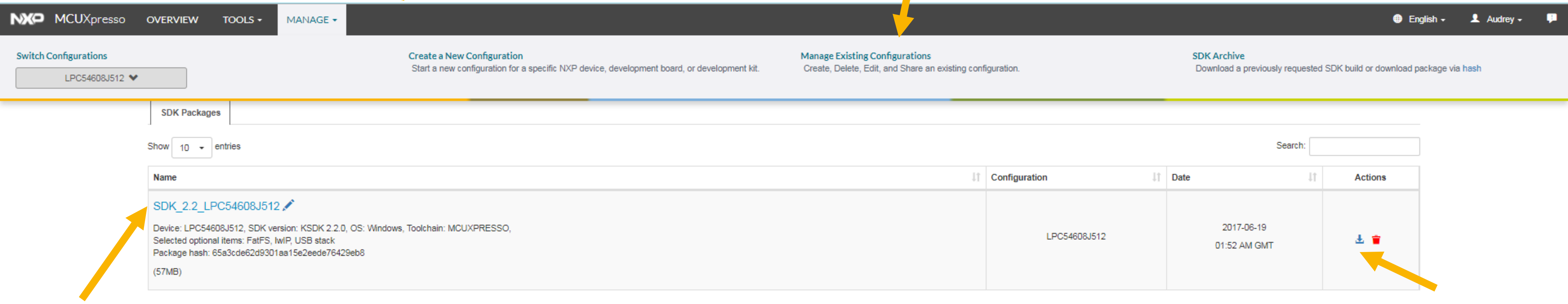
☐ Do this automatically for files like this from now on.

OK | Cancel

Save SDK once build completes

Configurations Archive

Access Configuration Archive from Manage menu



Create a New Configuration Start a new configuration for a specific NXP device, development board, or development kit.

Manage Existing Configurations Create, Delete, Edit, and Share an existing configuration.

SDK Archive Download a previously requested SDK build or download package via hash

SDK Packages

Show 10 entries Search:

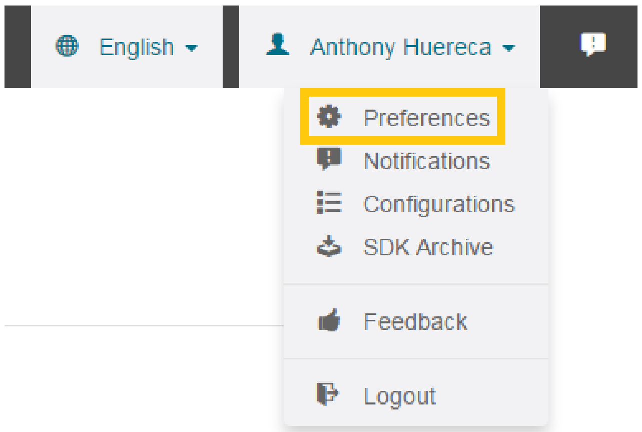
Name	Configuration	Date	Actions
SDK_2.2_LPC54608J512 Device: LPC54608J512, SDK version: KSDK 2.2.0, OS: Windows, Toolchain: MCUXPRESSO, Selected optional items: FatFS, lwIP, USB stack Package hash: 65a3cde62d9301aa15e2eede76429eb8 (57MB)	LPC54608J512	2017-06-19 01:52 AM GMT	

Current configuration

Upload a configuration

Preferences

- First time users may see an error if they have not filled out profile in “Preferences" as required for export control compliance
  - Name
  - Company
  - Country
  - Project Description



MCUXPRESSO SDK STRUCTURE

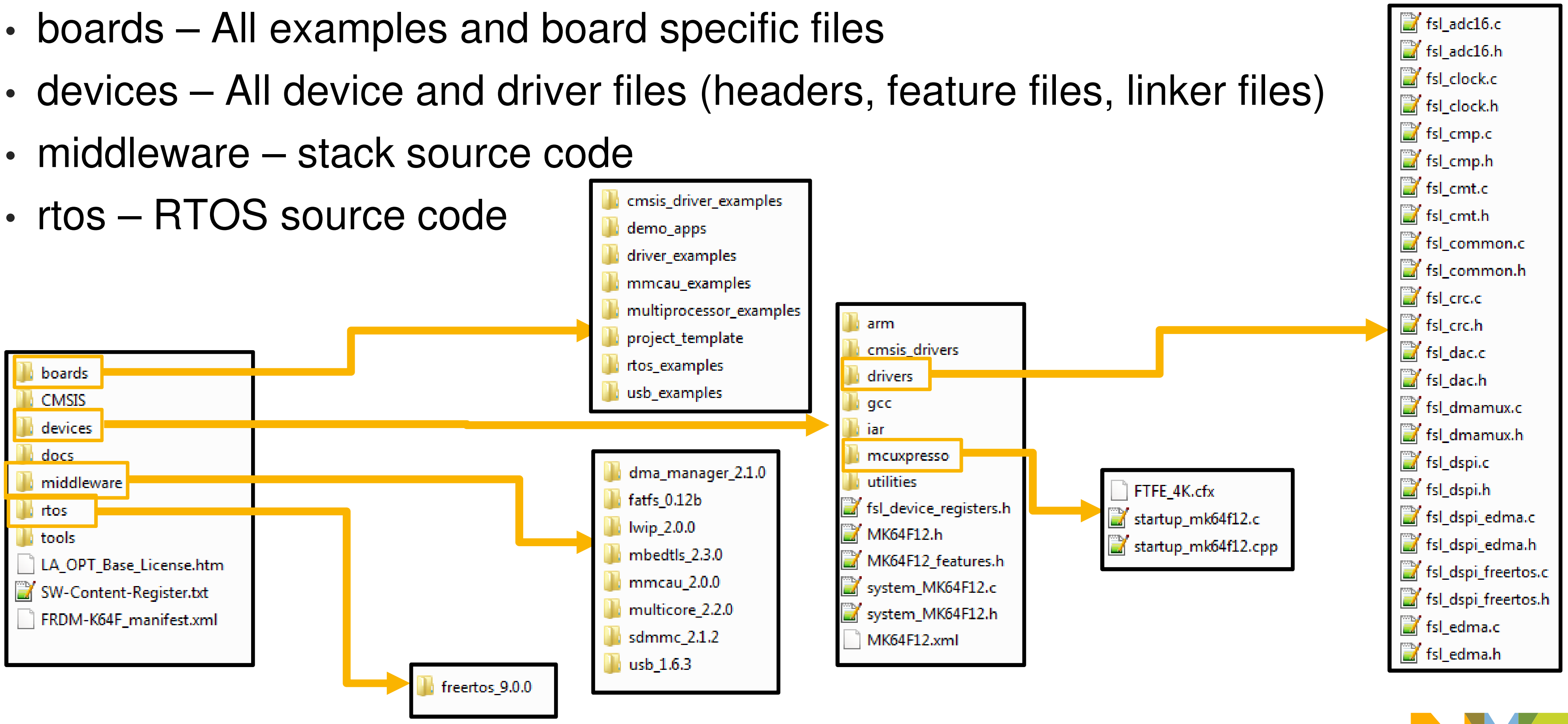
Zip or Unzip an SDK package

- SDK packages are downloaded as .zip files
- When using 3rd party IDEs, the SDK package must be unzipped
- For SDK support in the MCUXpresso Config Tools, the SDK package must also be unzipped

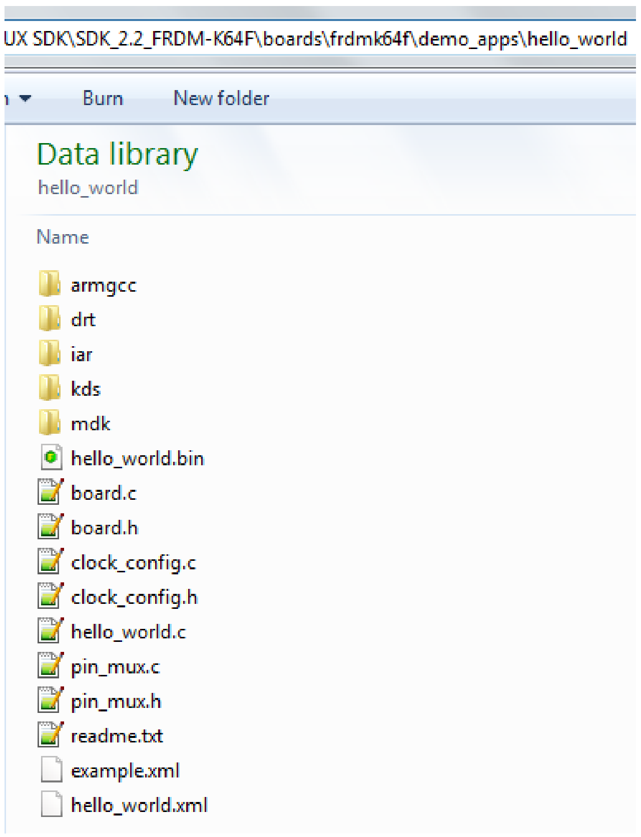
- MCUXpresso IDE can import SDK packages in either zipped or unzipped format.
  - Zipped SDKs:
    - When creating new projects or importing example projects, SDK source files are copied into the workspace (no linked references).
  - Unzipped SDKs:
    - When creating new projects or importing example projects, SDK source files can be copied into the workspace **or** referenced directly (linked references).
    - Requires additional time to unzip (one-time).
    - Provides speed improvement when many examples are imported to the workspace.

MCUXpresso SDK File Structure

- boards – All examples and board specific files
- devices – All device and driver files (headers, feature files, linker files)
- middleware – stack source code
- rtos – RTOS source code



MCUX Expresso SDK File Structure - Examples



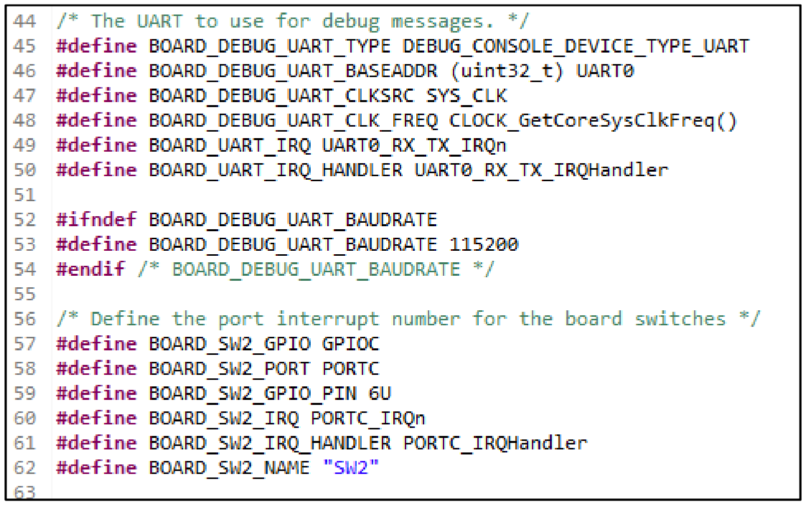
UX SDK\SDK_2.2_FRDM-K64F\boards\frdmk64f\demo_apps\hello_world

```
armgcc
drt
iar
kds
mdk
hello_world.bin
board.c
board.h
clock_config.c
clock_config.h
hello_world.c
pin_mux.c
pin_mux.h
readme.txt
example.xml
hello_world.xml
```

- Each example application has its own unique copy of the board, pin_mux, and clock_config files.
- Also each example also contains a pre-compiled .bin file for easy drag-and-drop programming
- Readme.txt contains instructions on how to run the demo and pins used

MCUXpresso File Structure - Examples

- Most configuration settings are in **board.h** file
  - UART module
  - UART baud
  - GPIO pins defined

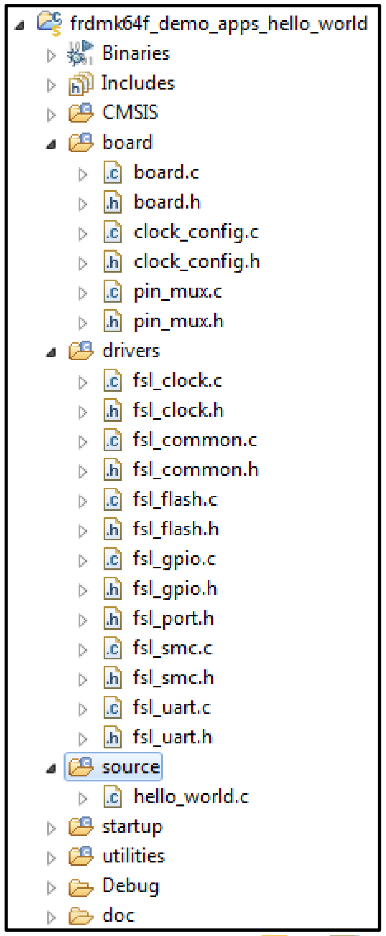


```
/* The UART to use for debug messages. */
#define BOARD_DEBUG_UART_TYPE DEBUG_CONSOLE_DEVICE_TYPE_UART
#define BOARD_DEBUG_UART_BASEADDR (uint32_t) UART0
#define BOARD_DEBUG_UART_CLKSRC SYS_CLK
#define BOARD_DEBUG_UART_CLK_FREQ CLOCK_GetCoreSysClkFreq()
#define BOARD_UART_IRQ UART0_RX_TX_IRQn
#define BOARD_UART_IRQ_HANDLER UART0_RX_TX_IRQHandler

#ifndef BOARD_DEBUG_UART_BAUDRATE
#define BOARD_DEBUG_UART_BAUDRATE 115200
#endif /* BOARD_DEBUG_UART_BAUDRATE */

/* Define the port interrupt number for the board switches */
#define BOARD_SW2_GPIO GPIOC
#define BOARD_SW2_PORT PORTC
#define BOARD_SW2_GPIO_PIN 6U
#define BOARD_SW2_IRQ PORTC_IRQn
#define BOARD_SW2_IRQ_HANDLER PORTC_IRQHandler
#define BOARD_SW2_NAME "SW2"
```

- Default UART pins defined in pin_mux.c in BOARD_InitPins().

MCUXpresso SDK Projects

- All source files are included in the example application projects
- Drivers are found under the **drivers** folder
- Board specific files under the **board** folder
- Application specific files under **source** folder



```
frdmk64f_demo_apps_hello_world
  Binaries
  Includes
  CMSIS
  board
    board.c
    board.h
    clock_config.c
    clock_config.h
    pin_mux.c
    pin_mux.h
  drivers
    fsl_clock.c
    fsl_clock.h
    fsl_common.c
    fsl_common.h
    fsl_flash.c
    fsl_flash.h
    fsl_gpio.c
    fsl_gpio.h
    fsl_port.h
    fsl_smc.c
    fsl_smc.h
    fsl_uart.c
    fsl_uart.h
  source
    hello_world.c
  startup
  utilities
  Debug
  doc
```

MCUXpresso SDK Startup

- Reset_Handler found in \devices\<device>\<compiler>\startup_<device>.s
  - Called ResetISR for MCUXpresso IDE
- SystemInit() found at \devices\<device>\system_<device>.c is used to enable cache (if available) and disable the watchdog timer.
- Then jumps to main(), and three configuration functions run:
  - **BOARD_InitPins();**
  - **BOARD_BootClockRUN();**
  - **BOARD_InitDebugConsole();**

LAB 1

Lab 1 : To create a new SDK configuration online

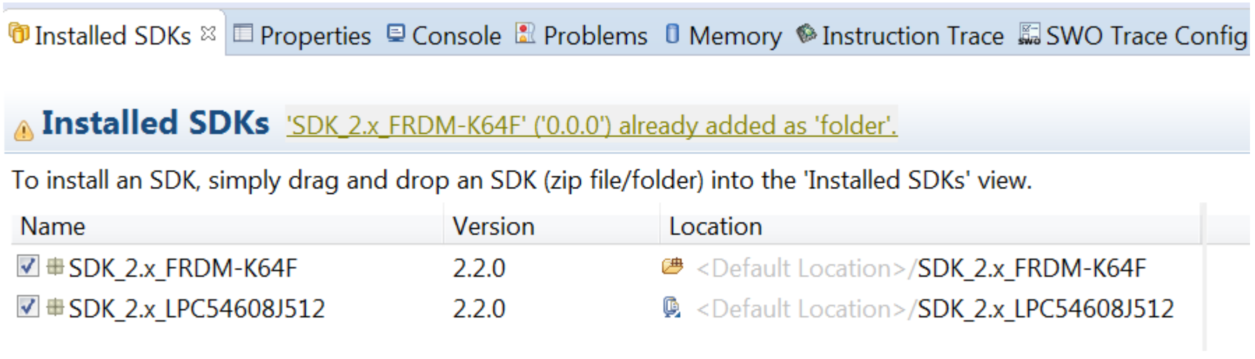
- **Pre-requisites**
  - PC running Windows/Linux/macOS
  - Internet connection
- Follow the Lab1 Hand out
- Download SDK_2.2_LPC54608J512

WALKTHROUGH INSTALLED SDK

Copy of SDK made in default path

- What happens when an SDK is dragged/dropped into the IDE?

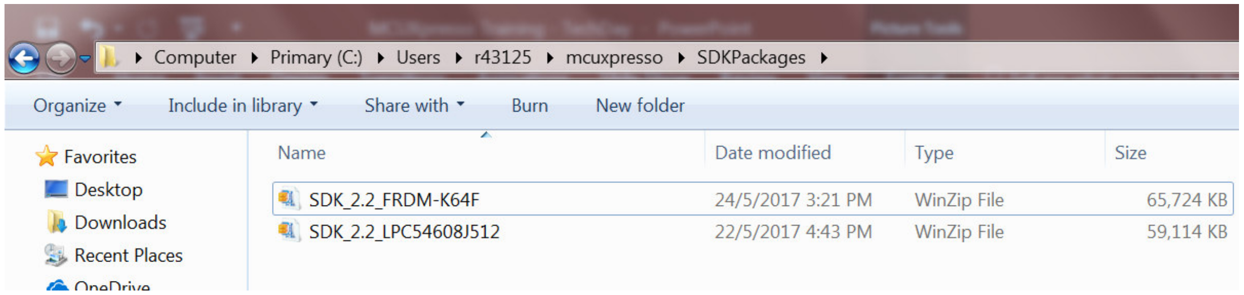
- The Drag/Drop feature creates a copy of the SDK located at **default path**: C:\Users\“user_name”\mcuxpresso\SDKPackages



Installed SDKs 'SDK_2.x_FRDM-K64F' ('0.0.0') already added as 'folder'.

To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view.

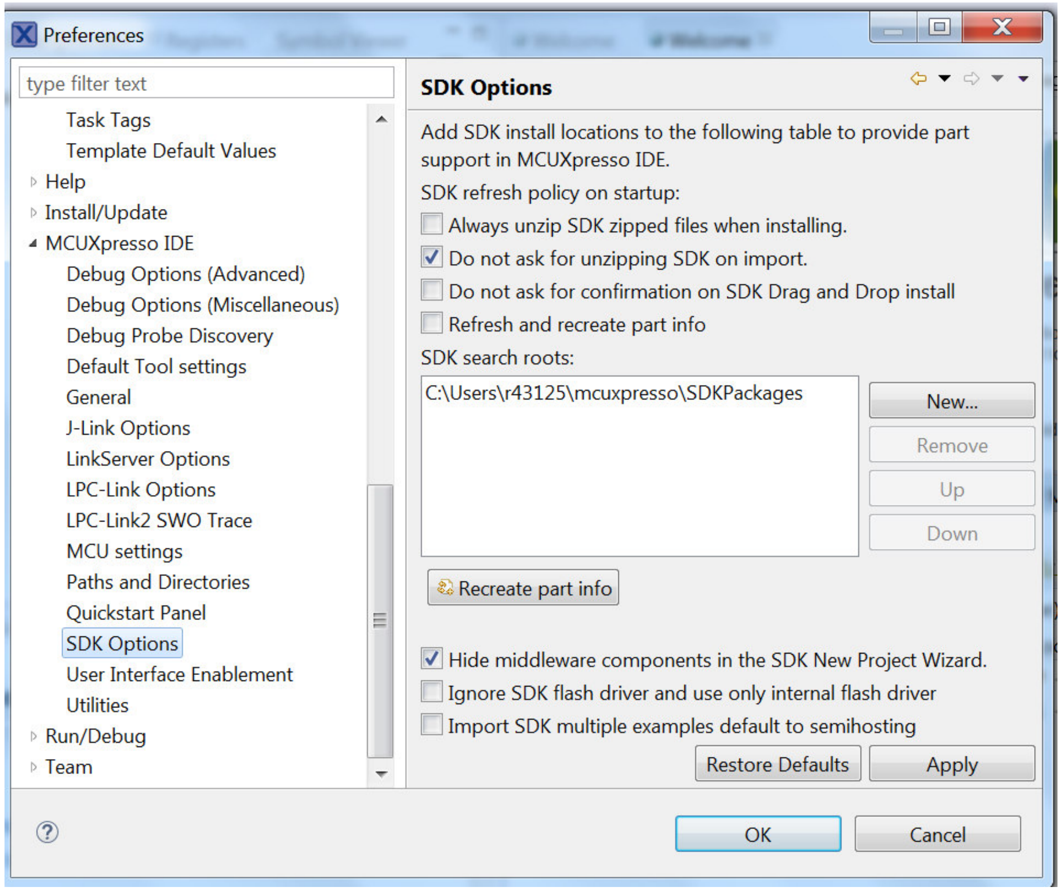
Name	Version	Location
✓ SDK_2.x_FRDM-K64F	2.2.0	<Default Location>/SDK_2.x_FRDM-K64F
✓ SDK_2.x_LPC54608J512	2.2.0	<Default Location>/SDK_2.x_LPC54608J512



Name	Date modified	Type	Size
SDK_2.2_FRDM-K64F	24/5/2017 3:21 PM	WinZip File	65,724 KB
SDK_2.2_LPC54608J512	22/5/2017 4:43 PM	WinZip File	59,114 KB

Install an SDK: Advanced

- Add paths to "SDK search roots:" for IDE to find current or future stored SDK packages
  - Window -> Preferences -> MCUXpresso IDE -> SDK Options
- SDKs can be zipped or unzipped
- For SDKs stored outside the default location:
  - "Delete SDK" function is disabled
  - Knowledge of SDKs is per workspace
- If multiple SDKs are found for the same device in various locations, you can choose which is loaded by reordering list (top has priority)
- Note: default location for drag/drop: C:\Users\"user_name"\mcuxpresso\SDKPackages



SDK Options

Add SDK install locations to the following table to provide part support in MCUXpresso IDE.

SDK refresh policy on startup:

- [ ] Always unzip SDK zipped files when installing.
- [✓] Do not ask for unzipping SDK on import.
- [ ] Do not ask for confirmation on SDK Drag and Drop install
- [ ] Refresh and recreate part info

SDK search roots:

C:\Users\r43125\mcuxpresso\SDKPackages

- [✓] Hide middleware components in the SDK New Project Wizard.
- [ ] Ignore SDK flash driver and use only internal flash driver
- [ ] Import SDK multiple examples default to semihosting

AGENDA

- **MCUXpresso Software And Tools Overview**
- **MCUXpresso SDK**
  - Web Builder
  - File Structure
- **MCUXpresso IDE**
  - Importing/Building
  - Debugging
- **MCUXpresso Config Tool**
  - Project Cloner
  - Pins Tool
  - Clocks Tool
- **LPC54608 LCD Lab, Key API and EmWin Demo**

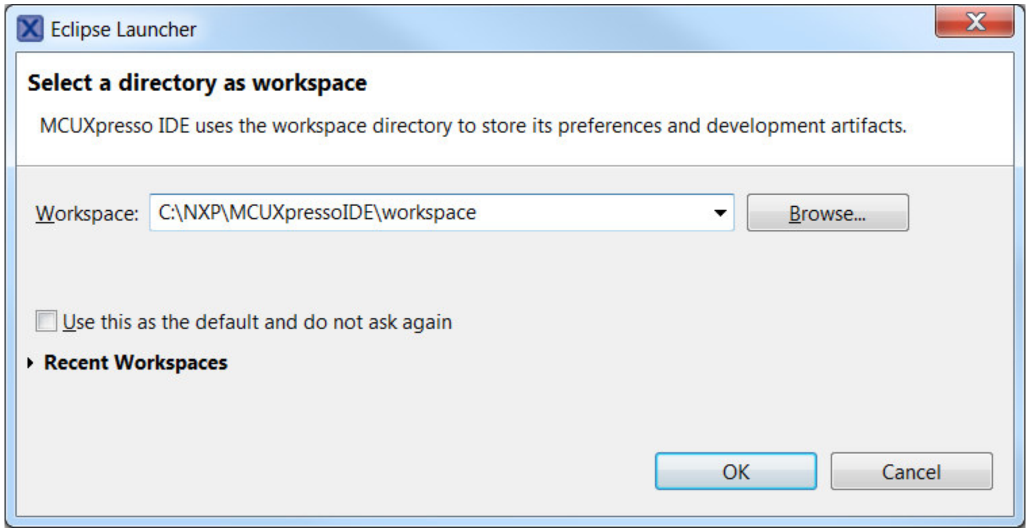
MCUXPRESSO IDE

Open MCUXpresso IDE

- Open MCUXpresso IDE on your system

- At the dialog box, enter a location for your workspace then click OK
  - Example) C:\NXP\MCUXpressoIDE\workspace

- Note: A workspace is a directory used to store projects that you want to actively work on during the IDE session



Select a directory as workspace

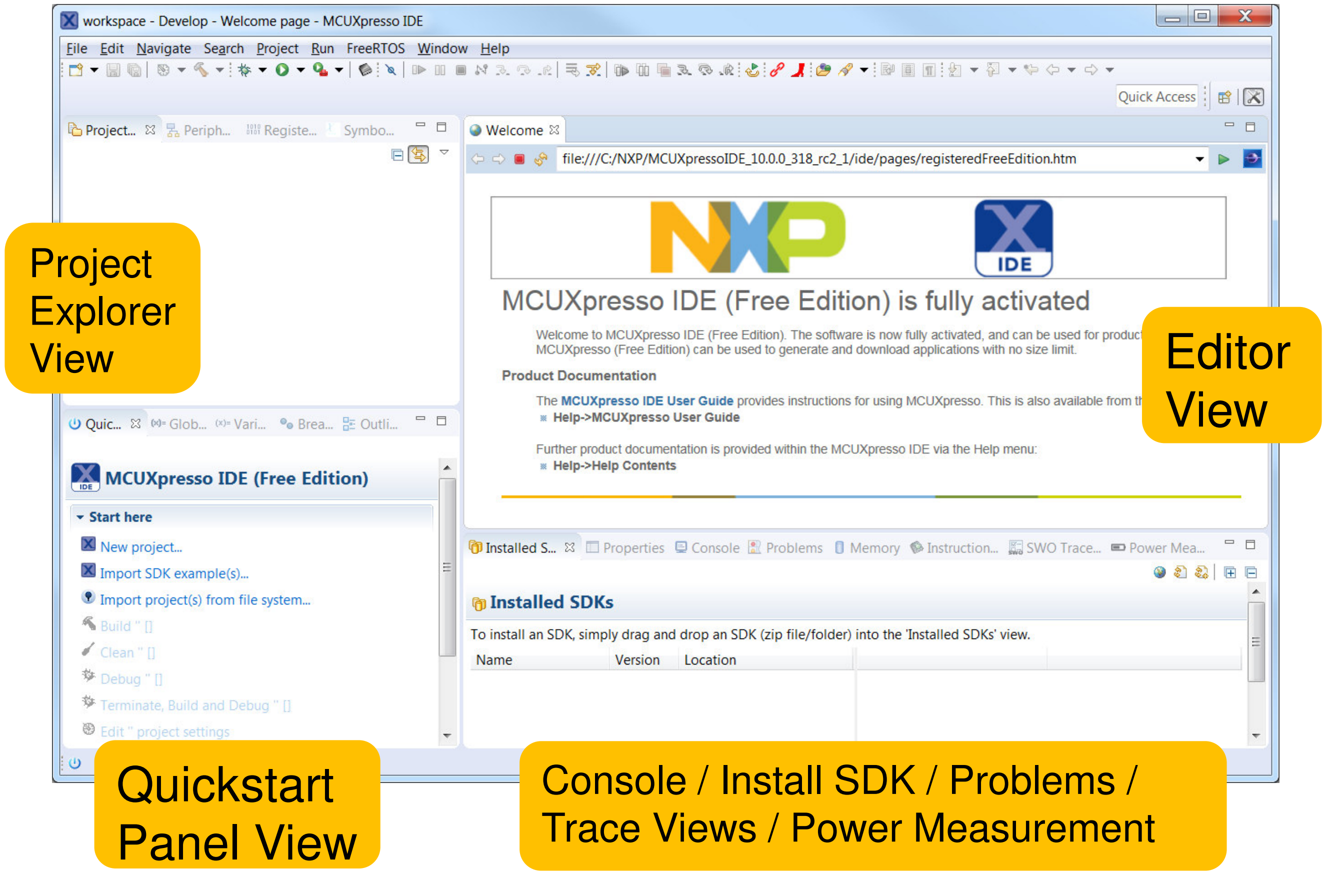
MCUXpresso IDE uses the workspace directory to store its preferences and development artifacts.

Workspace: C:\NXP\MCUXpressoIDE\workspace

www.nxp.com/mcuxpresso/ide

Develop Perspective

- MCUXpresso IDE will startup in a new workspace with no projects in the Develop Perspective
- A "perspective" is a collection of different "views"
- The Develop perspective provides a single combined project management and debugging view
- In addition to the default Develop perspective, the MCUXpresso IDE also supports traditional Eclipse C/C++ and Debug perspectives


workspace - Develop - Welcome page - MCUXpresso IDE
File Edit Navigate Search Project Run FreeRTOS Window Help
Quick Access
Project... Periph... Registe... Symbo...
Welcome
file:///C:/NXP/MCUXpressoIDE_10.0.0_318_rc2_1/ide/pages/registeredFreeEdition.htm
Quic... Glob... Vari... Brea... Outli...
MCUXpresso IDE (Free Edition)
Start here
New project...
Import SDK example(s)...
Import project(s) from file system...
Build '' []
Clean '' []
Debug '' []
Terminate, Build and Debug '' []
Edit '' project settings
Installed S... Properties Console Problems Memory Instruction... SWO Trace... Power Mea...
Name
Version
Location
Project Explorer View
Editor View
Quickstart Panel View
Console / Install SDK / Problems / Trace Views / Power Measurement

MCUXpresso IDE (Free Edition) is fully activated

Welcome to MCUXpresso IDE (Free Edition). The software is now fully activated, and can be used for product
MCUXpresso (Free Edition) can be used to generate and download applications with no size limit.

Product Documentation

The **MCUXpresso IDE User Guide** provides instructions for using MCUXpresso. This is also available from th
- **Help->MCUXpresso User Guide**

Further product documentation is provided within the MCUXpresso IDE via the Help menu:
- **Help->Help Contents**

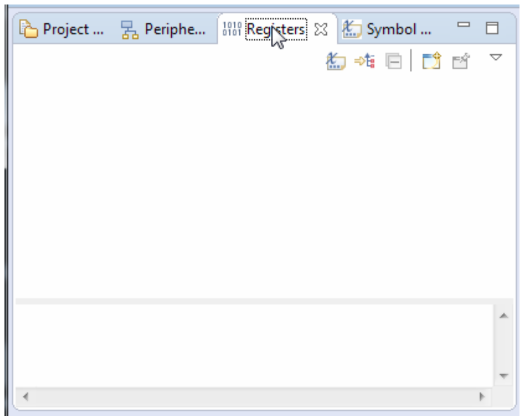
Installed SDKs

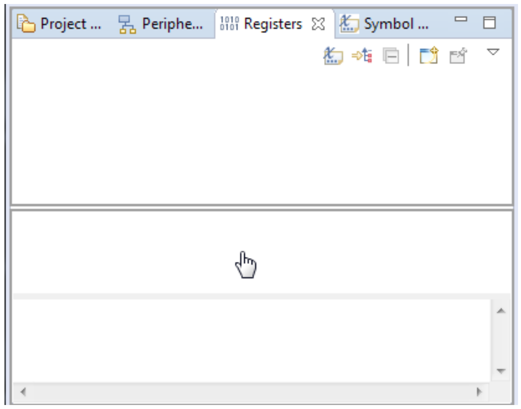
To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view.

Changing the Layout of the Develop Perspective

- Layout of views within a perspective can be tailored to meet your personal needs
- For example, if we wanted to have the Registers view always visible…

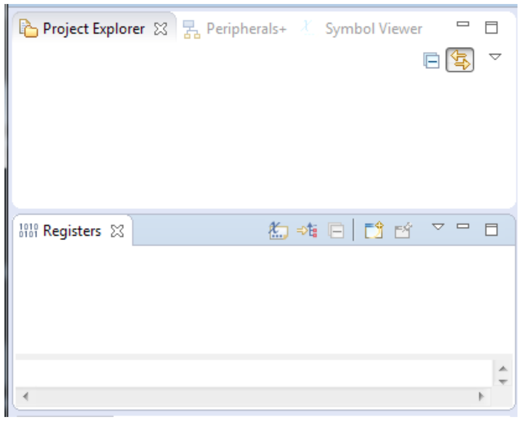
Click and hold down on the View you want to move

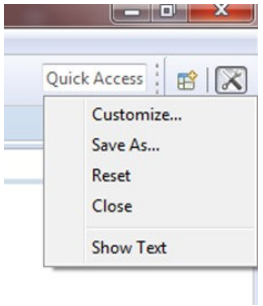




Continue to hold down and drag the cursor to the location you want to view to be displayed

Then release the mouse click, and the view will be placed at the required position

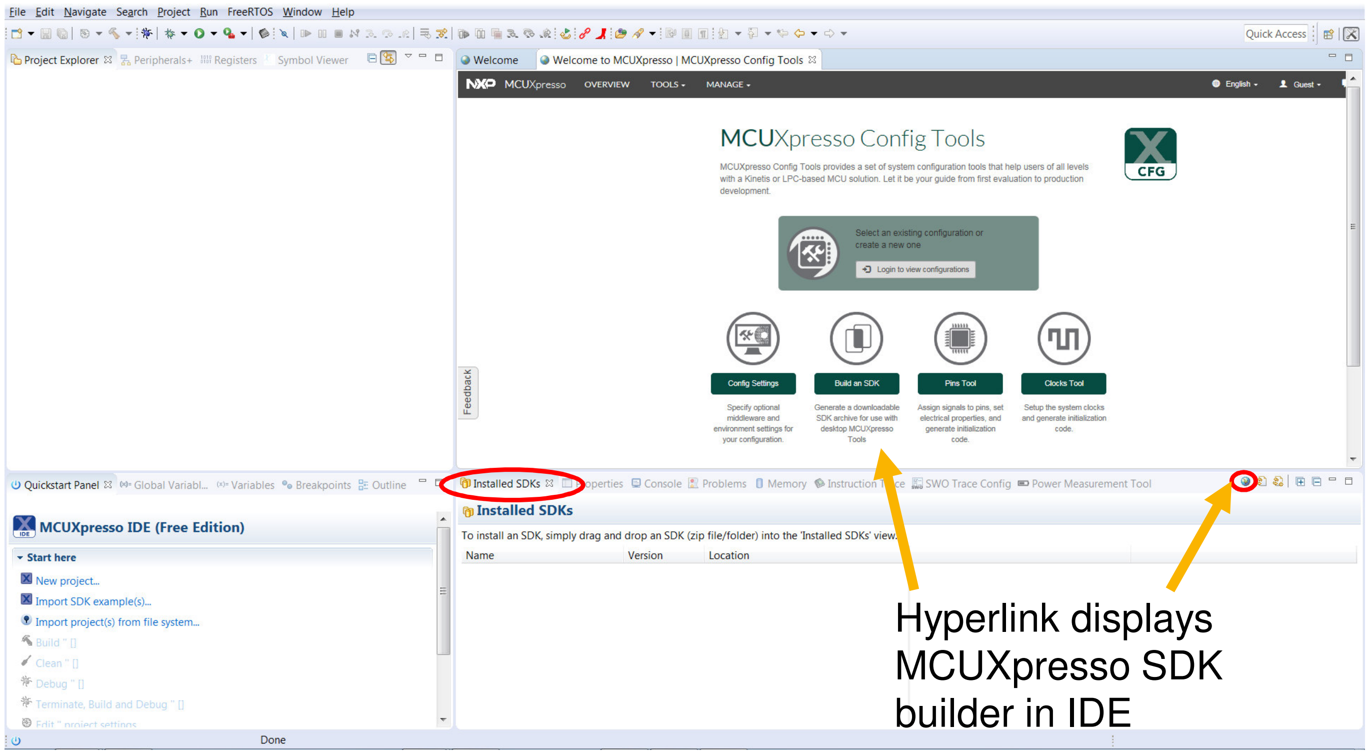




Right click on the Perspective button (top right of IDE window) to reset the layout back to the default

Installing an SDK in the IDE

- Part support is added by installing MCUXpresso SDKs into the IDE
- Allows example projects and driver examples from SDK to be easily imported
- New project generation based on board or processor in SDK
- The IDE is only compatible with SDKs built for MCUXpresso



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.

Select an existing configuration or create a new one

Login to view configurations

Specify optional middleware and environment settings for your configuration.

Generate a downloadable SDK archive for use with desktop MCUXpresso Tools

Assign signals to pins, set electrical properties, and generate initialization code.

Setup the system clocks and generate initialization code.

Installed SDKs

To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view.

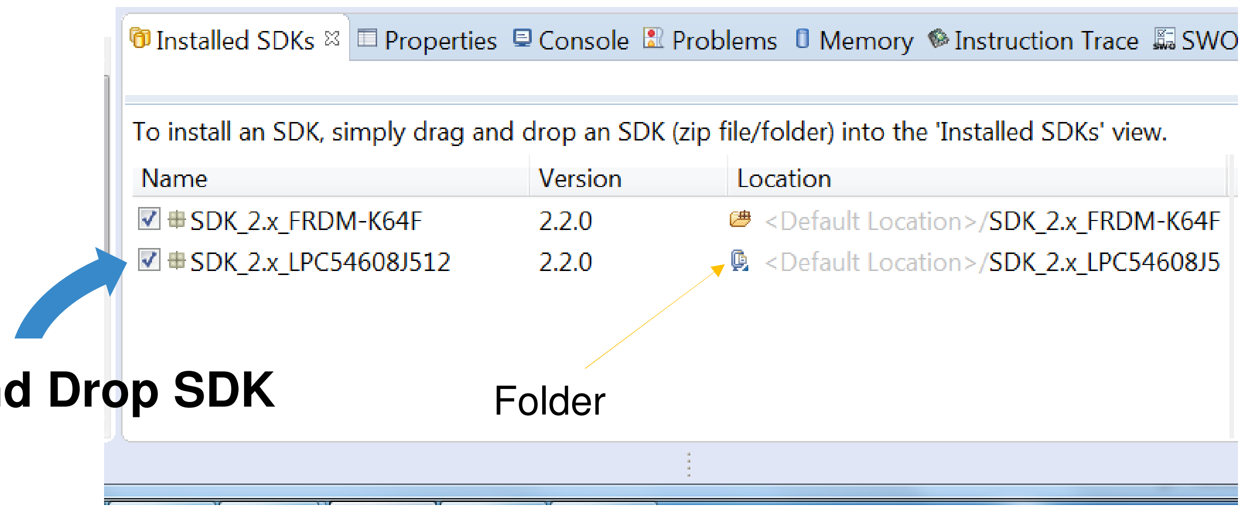
Hyperlink displays MCUXpresso SDK builder in IDE

Install an SDK: Drag and Drop

- Drag/Drop SDK packages directly into the IDE in the **Installed SDKs** view

- Can drag SDK as folder or zip (archive). IDE uses separate icon for each type

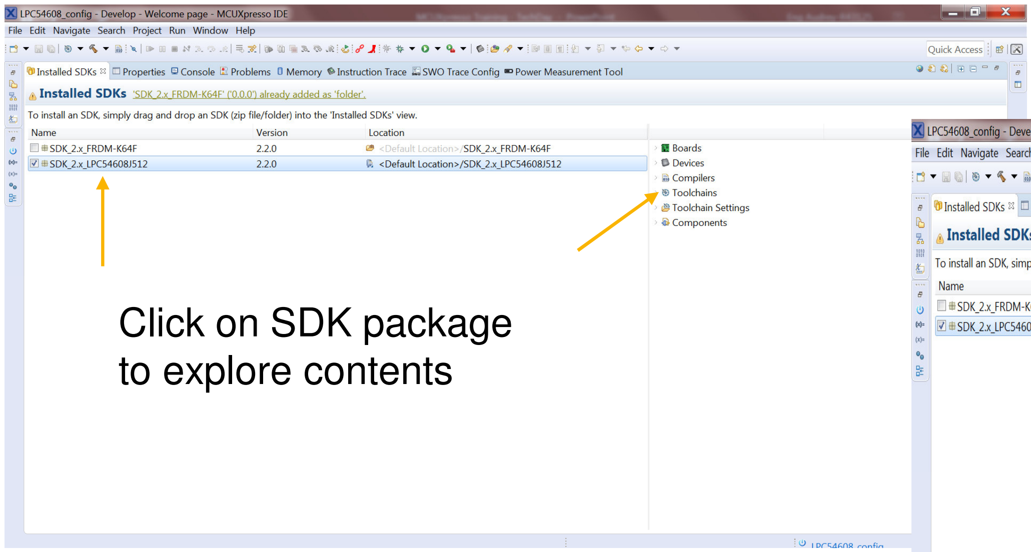
- SDKs installed in the default location are shared across workspaces



To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view.

Name	Version	Location
✓ SDK_2.x_FRDM-K64F	2.2.0	<Default Location>/SDK_2.x_FRDM-K64F
✓ SDK_2.x_LPC54608J512	2.2.0	<Default Location>/SDK_2.x_LPC54608J5

Inspect SDK



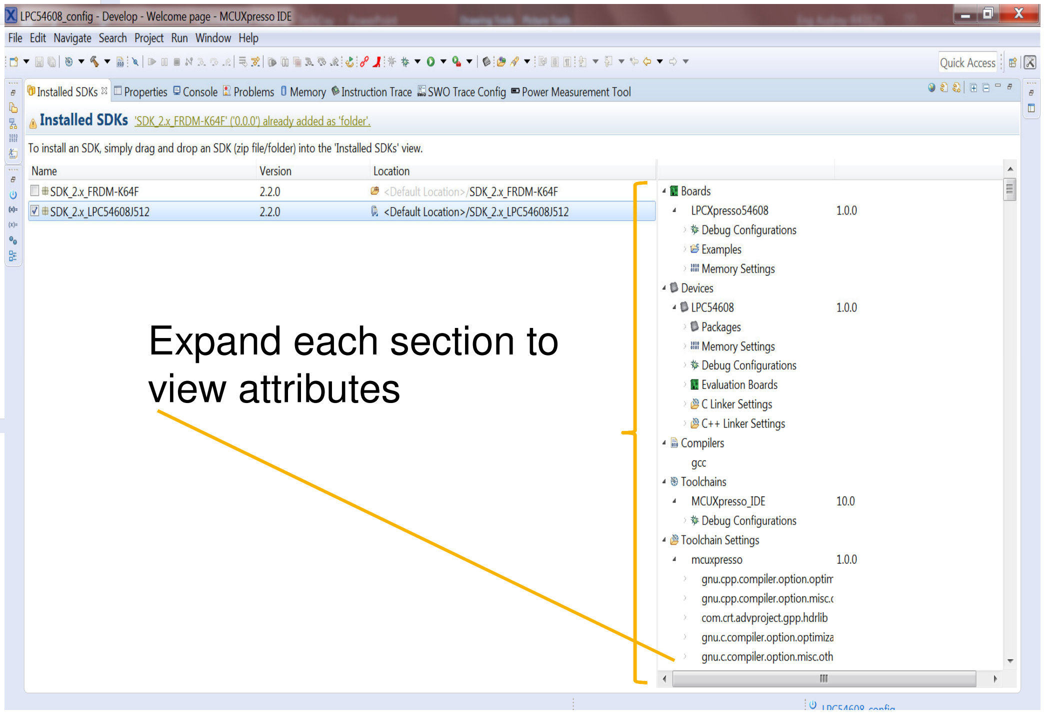
Installed SDKs 'SDK_2.x_FRDM-K64F' ('0.0.0') already added as 'folder'.

To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view.

Name	Version	Location
☐ SDK_2.x_FRDM-K64F	2.2.0	<Default Location>/SDK_2.x_FRDM-K64F
☑ SDK_2.x_LPC54608J512	2.2.0	<Default Location>/SDK_2.x_LPC54608J512

- Boards
- Devices
- Compilers
- Toolchains
- Toolchain Settings
- Components

Click on SDK package to explore contents



Installed SDKs 'SDK_2.x_FRDM-K64F' ('0.0.0') already added as 'folder'.

To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view.

Name	Version	Location
☐ SDK_2.x_FRDM-K64F	2.2.0	<Default Location>/SDK_2.x_FRDM-K64F
☑ SDK_2.x_LPC54608J512	2.2.0	<Default Location>/SDK_2.x_LPC54608J512

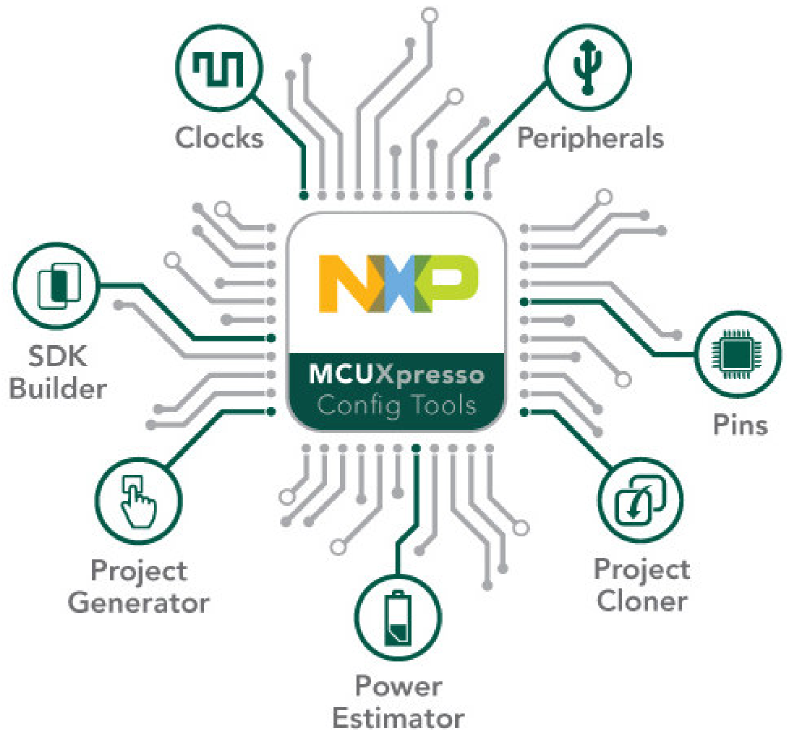
```
Boards
  LPCXpresso54608           1.0.0
    Debug Configurations
    Examples
    Memory Settings
Devices
  LPC54608                  1.0.0
    Packages
    Memory Settings
    Debug Configurations
    Evaluation Boards
    C Linker Settings
    C++ Linker Settings
Compilers
  gcc
Toolchains
  MCUXpresso_IDE            10.0
    Debug Configurations
Toolchain Settings
  mcuxpresso                1.0.0
    gnu.cpp.compiler.option.optim
    gnu.cpp.compiler.option.misc.c
    com.crt.advproject.gpp.hdrlib
    gnu.c.compiler.option.optimiza
    gnu.c.compiler.option.misc.oth
```

Expand each section to view attributes

Learn more at: www.nxp.com/mcuxpresso/config

MCUXpresso Config Tools

Integrated configuration and development tools for LPC and Kinetis MCUs



MCUXpresso Config Tools is a suite of evaluation and configuration tools that helps guide users from first evaluation to production software development.

SDK Builder packages custom SDKs based on user selections of MCU, evaluation board, and optional software components.

Pins, **Clocks**, and **Peripheral** tools generate initialization C code for custom board support. Features validation of inputs and cross-tool conflict resolution.

Project Generator creates new SDK projects with generated Pins and Clocks source files.

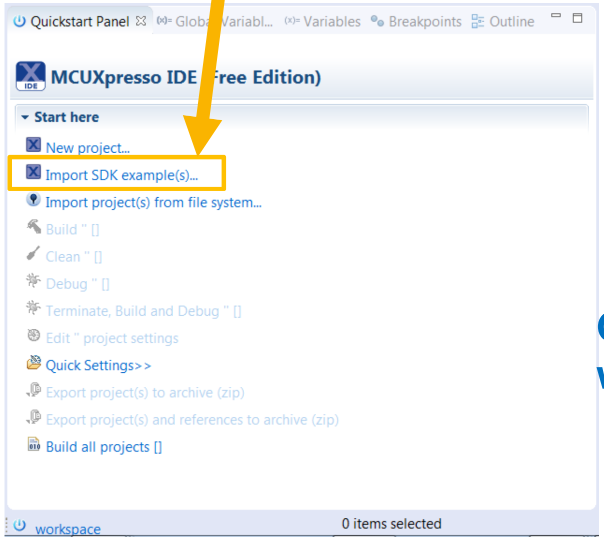
Project Cloning creates a standalone SDK project based on a example application available within SDK release.

Power Estimation tool provides energy and battery-life estimates based on a user's application model.
Available as a standalone tool for select devices.

MCUXPRESSO IDE IMPORTING/BUILDING

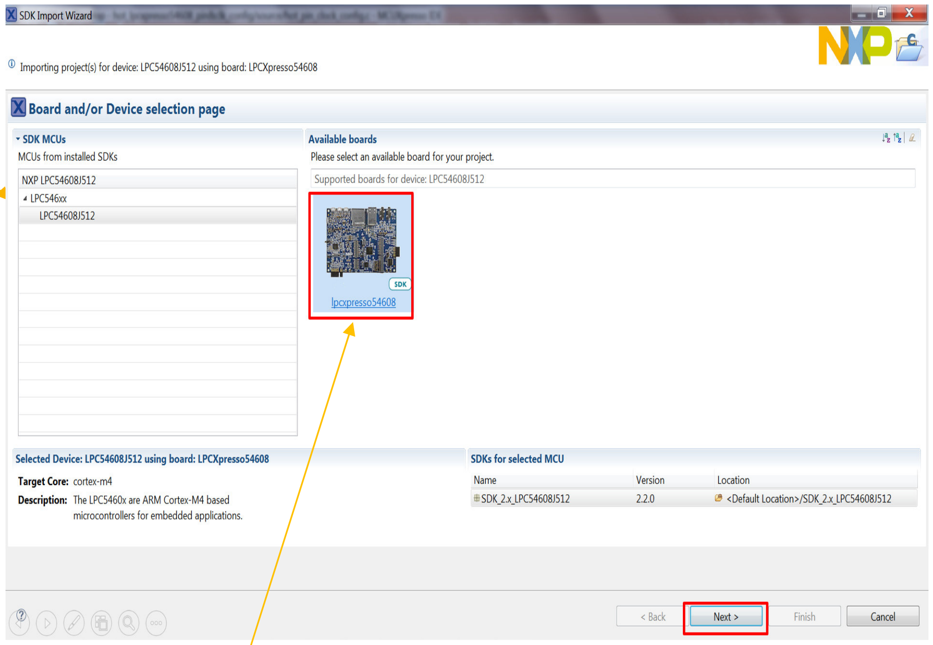
Import an SDK Example into the workspace

1. Click “Import SDK examples…” from Quickstart panel



Processors from installed SDKs

Opens selection wizard



Importing project(s) for device: LPC54608J512 using board: LPCXpresso54608

MCUs from installed SDKs

NXP LPC54608J512
LPC546xx
LPC54608J512

Please select an available board for your project.

Supported boards for device: LPC54608J512

Selected Device: LPC54608J512 using board: LPCXpresso54608

Target Core: cortex-m4

Description: The LPC5460x are ARM Cortex-M4 based microcontrollers for embedded applications.

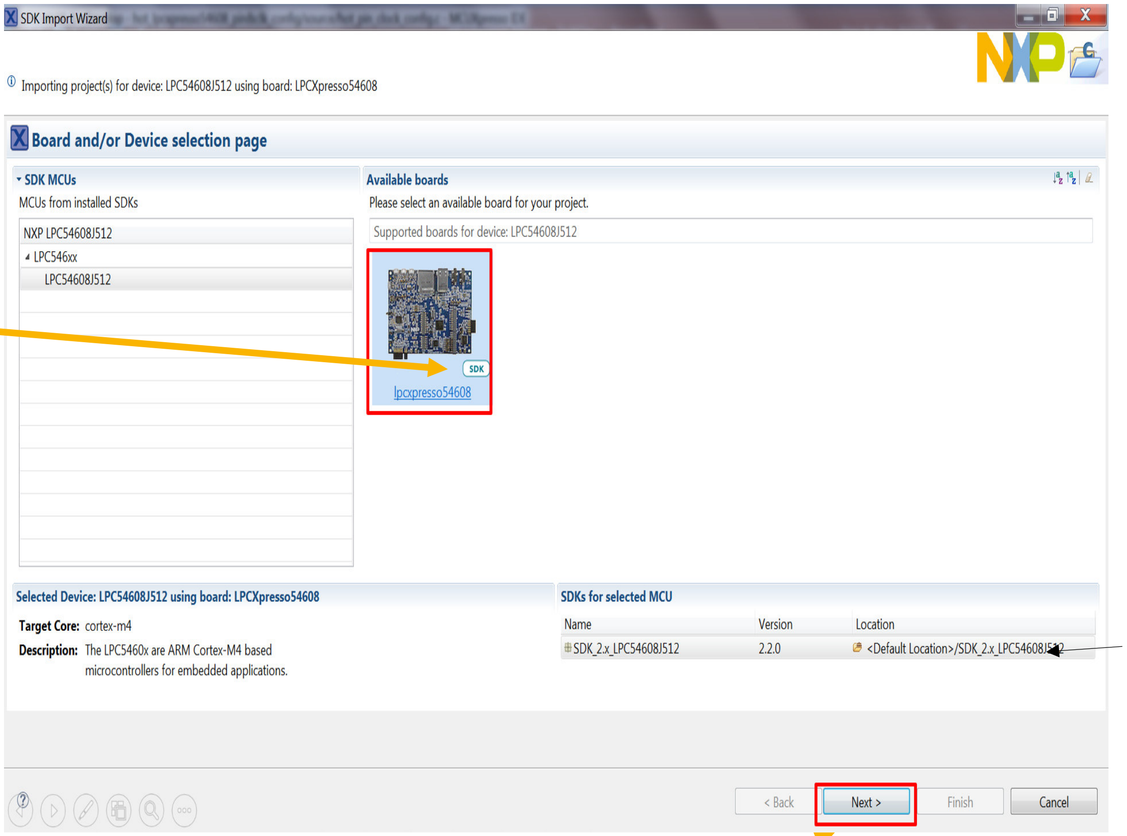
SDKs for selected MCU

Name	Version	Location
SDK_2.x_LPC54608J512	2.2.0	<Default Location>/SDK_2.x_LPC54608J512

- SDK examples are board specific

Boards from installed SDKs and preinstalled LPC boards

Import an SDK Example into the workspace



SDK Import Wizard

Importing project(s) for device: LPC54608J512 using board: LPCXpresso54608

Board and/or Device selection page

Selected Device: LPC54608J512 using board: LPCXpresso54608

Target Core: cortex-m4

Description: The LPC5460x are ARM Cortex-M4 based microcontrollers for embedded applications.

SDKs for selected MCU

Name	Version	Location
SDK_2.x_LPC54608J512	2.2.0	<Default Location>/SDK_2.x_LPC54608J512

2. Click on board image to import an SDK example

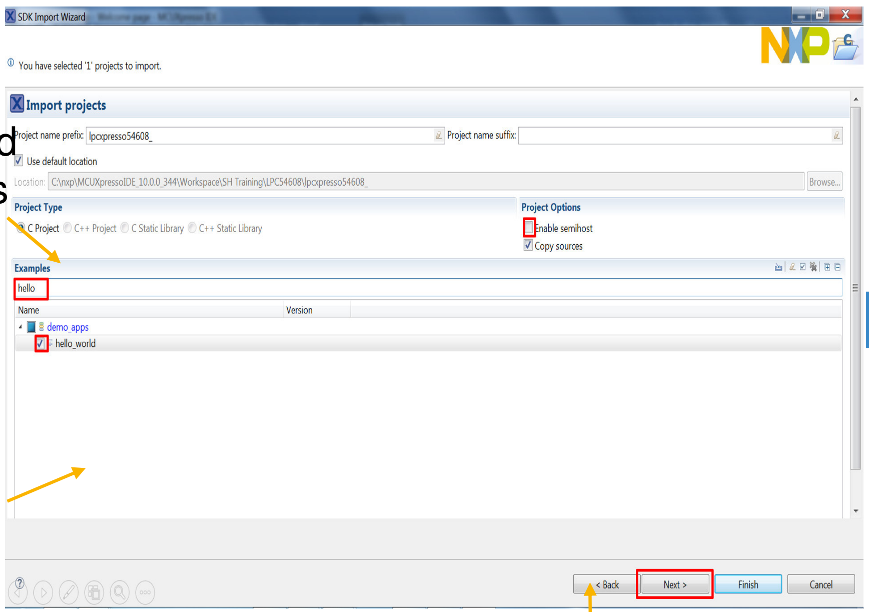
Installed SDK for selected board

3. Select Next to continue

SDK Example Import Wizard

4. Expand examples

5. Select project



You have selected '1' projects to import.

Import projects

Project name prefix: lpcxpresso54608_ Project name suffix:

☑ Use default location

Location: C:\nxp\MCUXpressoIDE_10.0.0_344\Workspace\SH Training\LPC54608\lpcxpresso54608_

Project Type

● C Project ○ C++ Project ○ C Static Library ○ C++ Static Library

Project Options

☐ Enable semihost

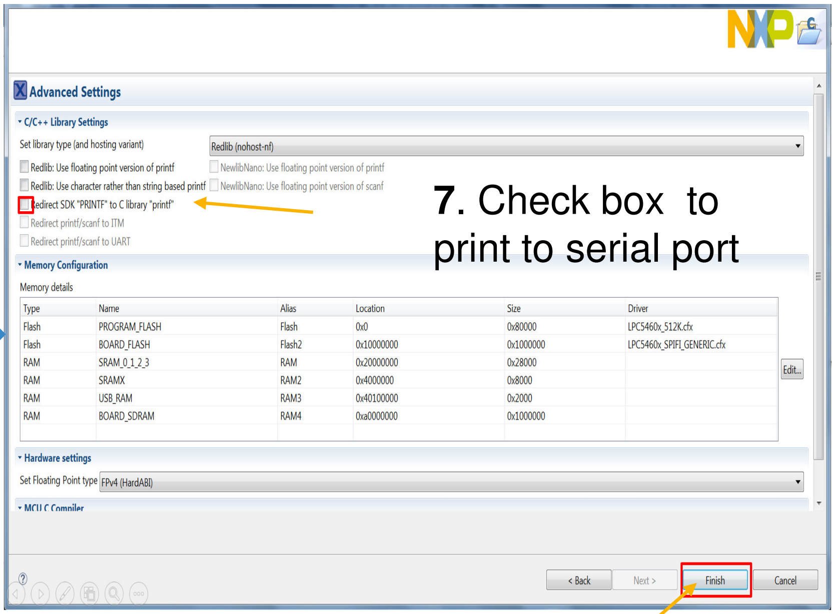
☑ Copy sources

Examples

hello

Name	Version
demo_apps	
☑ hello_world	

6. Click Next



Advanced Settings

C/C++ Library Settings

Set library type (and hosting variant): Redlib (nohost-nf)

☐ Redlib: Use floating point version of printf ☐ NewlibNano: Use floating point version of printf

☐ Redlib: Use character rather than string based printf ☐ NewlibNano: Use floating point version of scanf

☐ Redirect SDK "PRINTF" to C library "printf"

☐ Redirect printf/scanf to ITM

☐ Redirect printf/scanf to UART

Memory Configuration

Memory details

Type	Name	Alias	Location	Size	Driver
Flash	PROGRAM_FLASH	Flash	0x0	0x80000	LPC5460x_512K.cfx
Flash	BOARD_FLASH	Flash2	0x10000000	0x1000000	LPC5460x_SPIFI_GENERIC.cfx
RAM	SRAM_0_1_2_3	RAM	0x20000000	0x28000	
RAM	SRAMX	RAM2	0x4000000	0x8000	
RAM	USB_RAM	RAM3	0x40100000	0x2000	
RAM	BOARD_SDRAM	RAM4	0xa0000000	0x1000000	

Hardware settings

Set Floating Point type: FPv4 (HardABI)

MCU C Compiler

7. Check box to print to serial port

8. Click Finish

Copy Sources

- If copy sources is selected, files needed for example project are copied from the installed SDK into the project folder located in your workspace

- If the SDK was zipped this option would be selected automatically and greyed out

- If Copy Sources is not selected, SDK source files used in the project are linked directly from the installed SDK

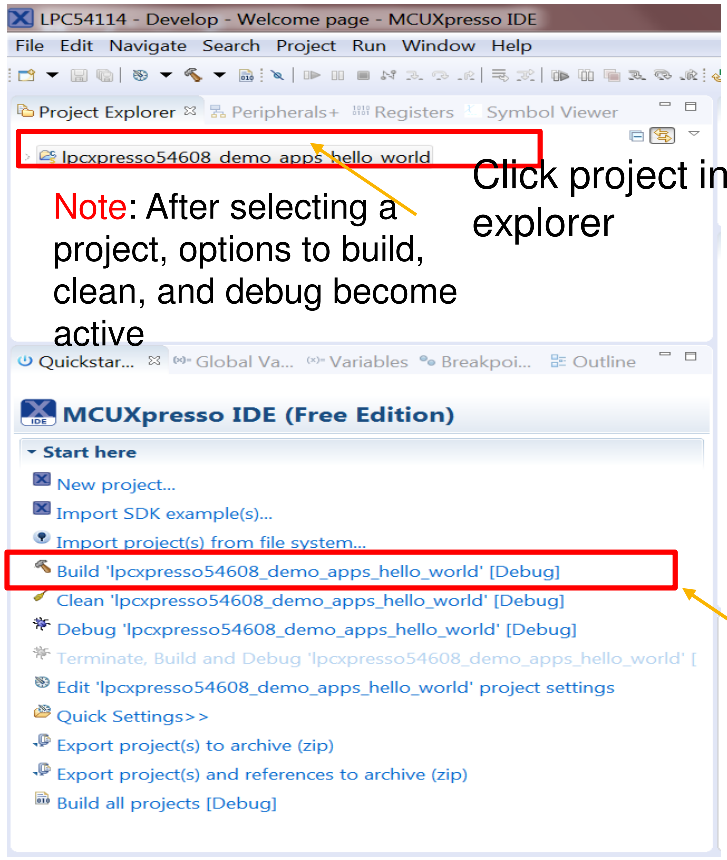
- **NOTE:** Linking sources will modify the installed SDK

Sharing Projects

- If a project is built using part support from an SDK and is then exported – for example to share the project with a colleague who also uses MCUXpresso IDE, then the colleague must also install an SDK providing part support for the project's MCU.

- **Note**: Because device support is included in the SDK, it is recommended that any required SDKs are installed before a project requiring SDK part support is imported. However, if this is not done beforehand, simply select the imported project in the project explorer and right click and select: *C/C++ Build -> MCU settings* ensure the correct MCU is selected and click **Refresh MCU Cache**.

Building an Example

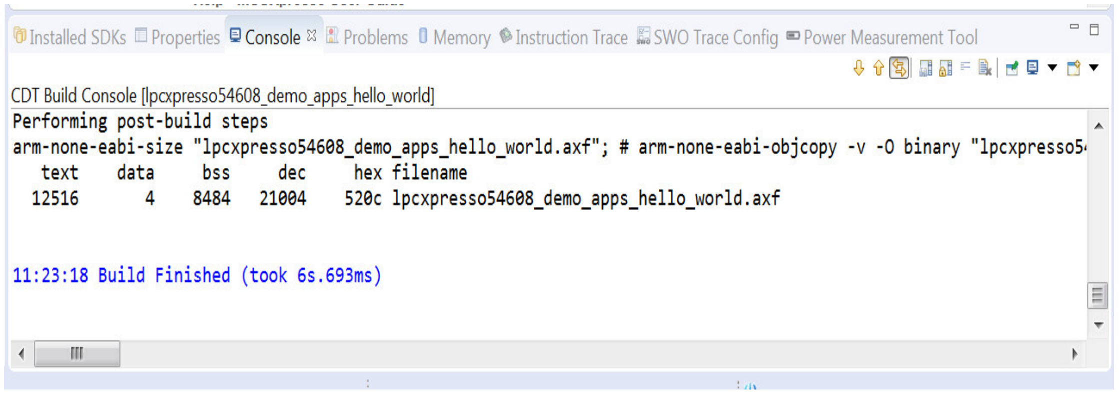

LPC54114 - Develop - Welcome page - MCUXpresso IDE
File Edit Navigate Search Project Run Window Help
Project Explorer
Peripherals+
Registers
Symbol Viewer
lpcxpresso54608_demo_apps_hello_world
Quickstar...
Global Va...
Variables
Breakpoi...
Outline
MCUXpresso IDE (Free Edition)
Start here
New project...
Import SDK example(s)...
Import project(s) from file system...
Build 'lpcxpresso54608_demo_apps_hello_world' [Debug]
Clean 'lpcxpresso54608_demo_apps_hello_world' [Debug]
Debug 'lpcxpresso54608_demo_apps_hello_world' [Debug]
Terminate, Build and Debug 'lpcxpresso54608_demo_apps_hello_world' [
Edit 'lpcxpresso54608_demo_apps_hello_world' project settings
Quick Settings>>
Export project(s) to archive (zip)
Export project(s) and references to archive (zip)
Build all projects [Debug]

Click project in explorer

Note: After selecting a project, options to build, clean, and debug become active

Click Build

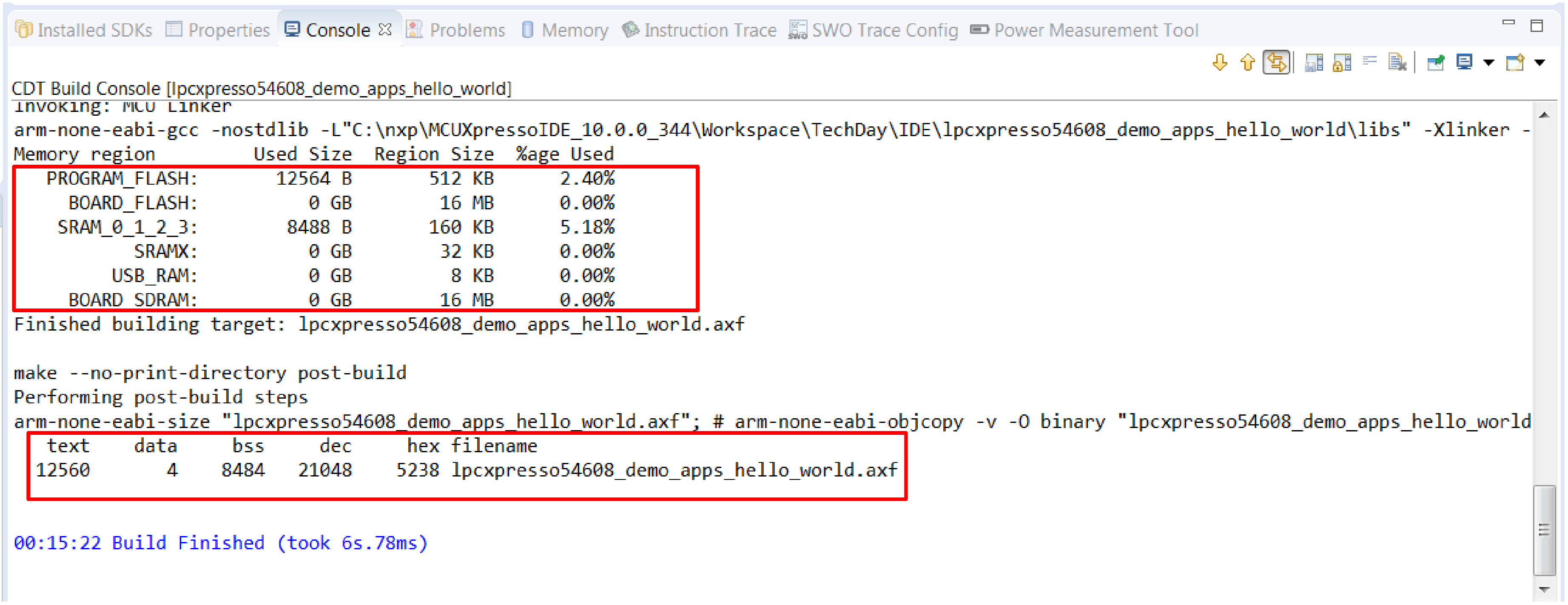
View Build Status in Console



CDT Build Console [lpcxpresso54608_demo_apps_hello_world]

```
Performing post-build steps
arm-none-eabi-size "lpcxpresso54608_demo_apps_hello_world.axf"; # arm-none-eabi-objcopy -v -O binary "lpcxpresso5
   text	   data	    bss	    dec	    hex	filename
  12516	      4	   8484	  21004	   520c	lpcxpresso54608_demo_apps_hello_world.axf


11:23:18 Build Finished (took 6s.693ms)
```

Memory Usage in Build Console



CDT Build Console [lpcxpresso54608_demo_apps_hello_world]

```
Invoking: MCU Linker
arm-none-eabi-gcc -nostdlib -L"C:\nxp\MCUXpressoIDE_10.0.0_344\Workspace\TechDay\IDE\lpcxpresso54608_demo_apps_hello_world\libs" -Xlinker -
Memory region         Used Size  Region Size  %age Used
   PROGRAM_FLASH:        12564 B       512 KB      2.40%
     BOARD_FLASH:          0 GB        16 MB      0.00%
    SRAM_0_1_2_3:        8488 B       160 KB      5.18%
           SRAMX:          0 GB        32 KB      0.00%
         USB_RAM:          0 GB         8 KB      0.00%
     BOARD_SDRAM:          0 GB        16 MB      0.00%
Finished building target: lpcxpresso54608_demo_apps_hello_world.axf

make --no-print-directory post-build
Performing post-build steps
arm-none-eabi-size "lpcxpresso54608_demo_apps_hello_world.axf"; # arm-none-eabi-objcopy -v -O binary "lpcxpresso54608_demo_apps_hello_world
   text    data     bss     dec     hex filename
  12560       4    8484   21048    5238 lpcxpresso54608_demo_apps_hello_world.axf


00:15:22 Build Finished (took 6s.78ms)
```

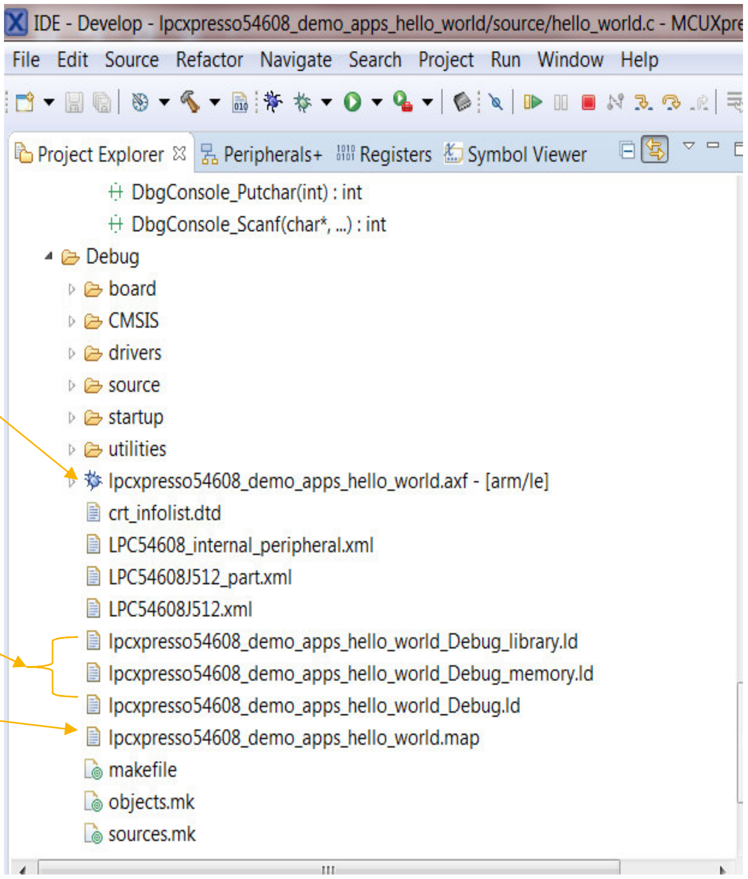
- **text** - shows the code and read-only data in your application (in decimal)
- **data** - shows the read-write data in your application (in decimal)
- **bss** - show the zero initialized ('bss' and 'common') data in your application (in decimal)
- **dec** - total of 'text' + 'data' + 'bss' (in decimal)
- **hex** - hexadecimal equivalent of 'dec'

Build Results

- Link step will generate an AXF file
  - Standard ARM Executable Format – ELF/DWARF
  - MCUXpresso IDE can directly download to target
  - Post build step can be used to convert to other formats, such as binary or hex (using arm-none-eabi-objcopy)

- Linker scripts, controlling placement of code and data in memory, generated automatically by IDE

- MAP file generated by linker can be very useful too
  - Shows where code and data has been placed, and sizes of individual sections

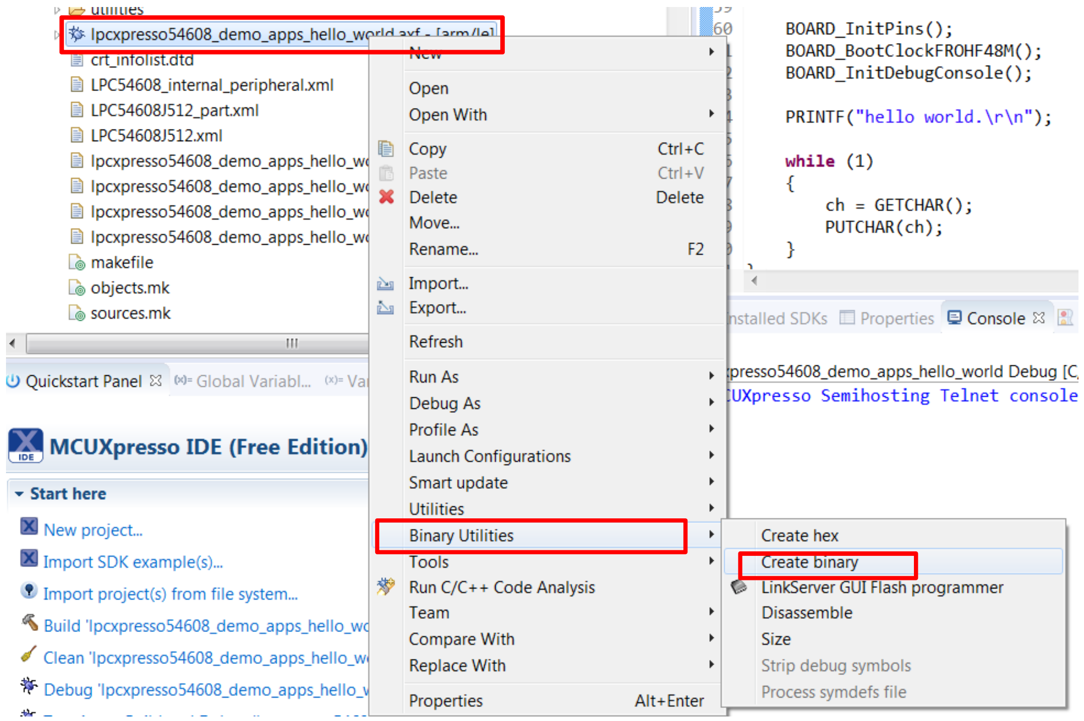


IDE - Develop - lpcxpresso54608_demo_apps_hello_world/source/hello_world.c - MCUXpre
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
Peripherals+
Registers
Symbol Viewer

```
DbgConsole_Putchar(int) : int
DbgConsole_Scanf(char*, ...) : int
Debug
  board
  CMSIS
  drivers
  source
  startup
  utilities
  lpcxpresso54608_demo_apps_hello_world.axf - [arm/le]
  crt_infolist.dtd
  LPC54608_internal_peripheral.xml
  LPC54608J512_part.xml
  LPC54608J512.xml
  lpcxpresso54608_demo_apps_hello_world_Debug_library.ld
  lpcxpresso54608_demo_apps_hello_world_Debug_memory.ld
  lpcxpresso54608_demo_apps_hello_world_Debug.ld
  lpcxpresso54608_demo_apps_hello_world.map
  makefile
  objects.mk
  sources.mk
```

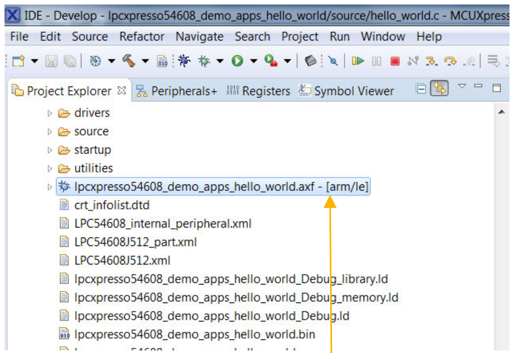
Create Binary

- Useful for drag-and-drop programming via OpenSDA
- Right Click on .axf file: Binary Utilities -> Create Binary



```
BOARD_InitPins();
BOARD_BootClockFROHF48M();
BOARD_InitDebugConsole();

PRINTF("hello world.\r\n");

while (1)
{
    ch = GETCHAR();
    PUTCHAR(ch);
}
```

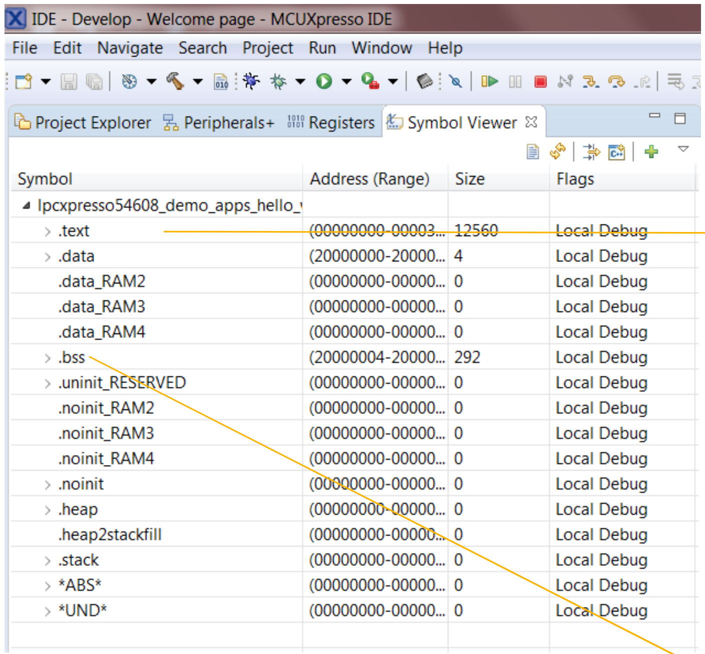
Symbol Viewer


IDE - Develop - lpcxpresso54608_demo_apps_hello_world/source/hello_world.c - MCUXpress

File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
Peripherals+
Registers
Symbol Viewer

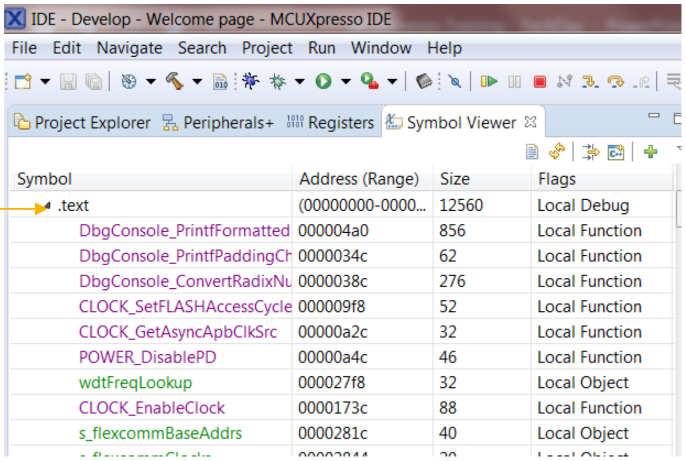
- drivers
- source
- startup
- utilities
- lpcxpresso54608_demo_apps_hello_world.axf - [arm/le]
- crt_infolist.dtd
- LPC54608_internal_peripheral.xml
- LPC54608J512_part.xml
- LPC54608J512.xml
- lpcxpresso54608_demo_apps_hello_world_Debug_library.ld
- lpcxpresso54608_demo_apps_hello_world_Debug_memory.ld
- lpcxpresso54608_demo_apps_hello_world_Debug.ld
- lpcxpresso54608_demo_apps_hello_world.bin

- Right-click .axf file in explorer, then select Tools > View Symbols

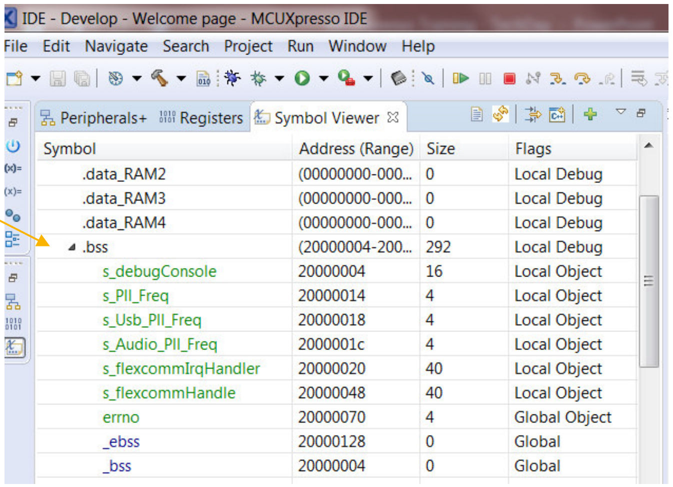

IDE - Develop - Welcome page - MCUXpresso IDE

Symbol	Address (Range)	Size	Flags
lpcxpresso54608_demo_apps_hello_			
.text	(00000000-00003...	12560	Local Debug
.data	(20000000-20000...	4	Local Debug
.data_RAM2	(00000000-00000...	0	Local Debug
.data_RAM3	(00000000-00000...	0	Local Debug
.data_RAM4	(00000000-00000...	0	Local Debug
.bss	(20000004-20000...	292	Local Debug
.uninit_RESERVED	(00000000-00000...	0	Local Debug
.noinit_RAM2	(00000000-00000...	0	Local Debug
.noinit_RAM3	(00000000-00000...	0	Local Debug
.noinit_RAM4	(00000000-00000...	0	Local Debug
.noinit	(00000000-00000...	0	Local Debug
.heap	(00000000-00000...	0	Local Debug
.heap2stackfill	(00000000-00000...	0	Local Debug
.stack	(00000000-00000...	0	Local Debug
ABS	(00000000-00000...	0	Local Debug
UND	(00000000-00000...	0	Local Debug

- Symbol viewer will move to front of view


IDE - Develop - Welcome page - MCUXpresso IDE

Symbol	Address (Range)	Size	Flags
.text	(00000000-0000...	12560	Local Debug
DbgConsole_PrintfFormatted	000004a0	856	Local Function
DbgConsole_PrintfPaddingCh	0000034c	62	Local Function
DbgConsole_ConvertRadixNu	0000038c	276	Local Function
CLOCK_SetFLASHAccessCycle	000009f8	52	Local Function
CLOCK_GetAsyncApbClkSrc	00000a2c	32	Local Function
POWER_DisablePD	00000a4c	46	Local Function
wdtFreqLookup	000027f8	32	Local Object
CLOCK_EnableClock	0000173c	88	Local Function
s_flexcommBaseAddrs	0000281c	40	Local Object


IDE - Develop - Welcome page - MCUXpresso IDE

Symbol	Address (Range)	Size	Flags
.data_RAM2	(00000000-000...	0	Local Debug
.data_RAM3	(00000000-000...	0	Local Debug
.data_RAM4	(00000000-000...	0	Local Debug
.bss	(20000004-200...	292	Local Debug
s_debugConsole	20000004	16	Local Object
s_Pll_Freq	20000014	4	Local Object
s_Usb_Pll_Freq	20000018	4	Local Object
s_Audio_Pll_Freq	2000001c	4	Local Object
s_flexcommIrqHandler	20000020	40	Local Object
s_flexcommHandle	20000048	40	Local Object
errno	20000070	4	Global Object
_ebss	20000128	0	Global
_bss	20000004	0	Global

- Expand each section to examine its symbols

New Project for Board (Defaults)

Project Name: Automatically named but can be modified.

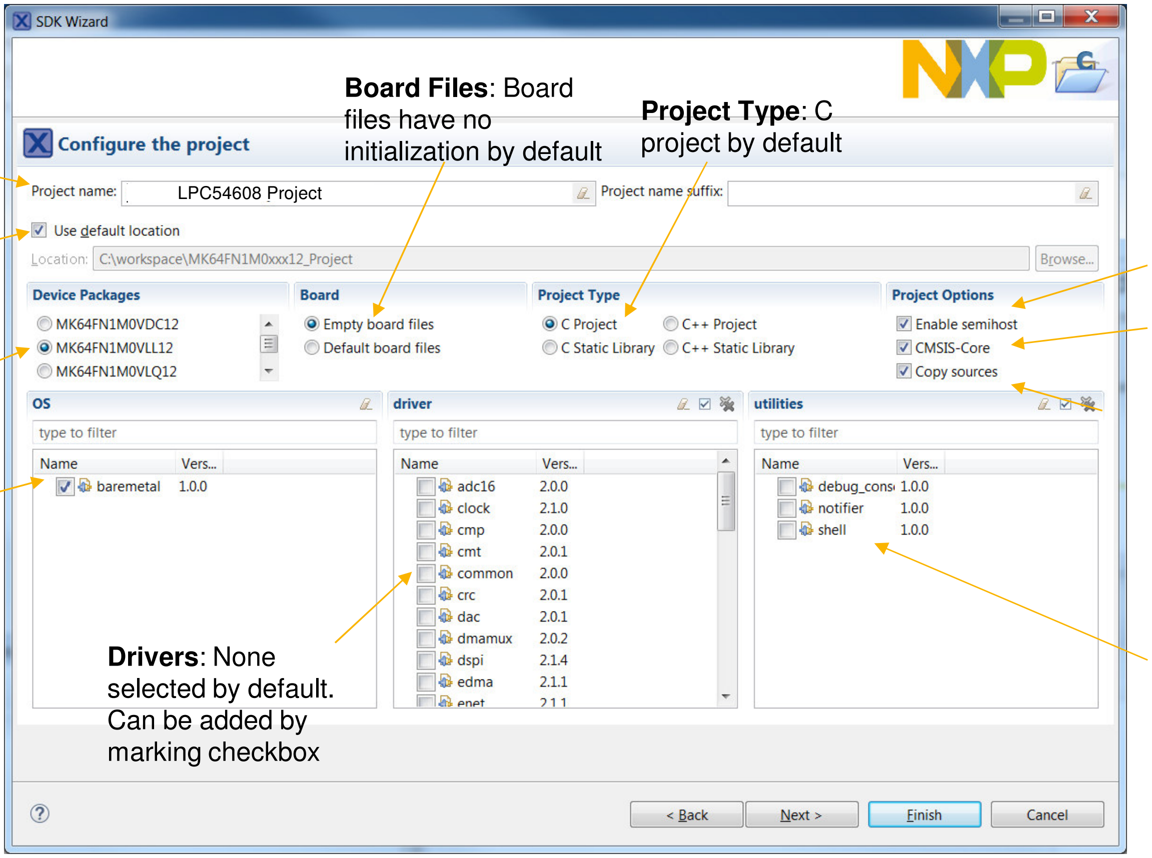
Project Location: Defaults to current workspace

Packages: Package of processor on board selected

OS: Baremetal selected by default

Board Files: Board files have no initialization by default

Project Type: C project by default



SDK Wizard

Configure the project

Project name: LPC54608 Project   Project name suffix:

[✓] Use default location

Location: C:\workspace\MK64FN1M0xxx12_Project   Browse...

Device Packages	Board	Project Type		Project Options
○ MK64FN1M0VDC12	● Empty board files	● C Project	○ C++ Project	[✓] Enable semihost
● MK64FN1M0VLL12	○ Default board files	○ C Static Library	○ C++ Static Library	[✓] CMSIS-Core
○ MK64FN1M0VLQ12				[✓] Copy sources

OS

Name	Vers...
[✓] baremetal	1.0.0

driver

Name	Vers...
[ ] adc16	2.0.0
[ ] clock	2.1.0
[ ] cmp	2.0.0
[ ] cmt	2.0.1
[ ] common	2.0.0
[ ] crc	2.0.1
[ ] dac	2.0.1
[ ] dmamux	2.0.2
[ ] dspi	2.1.4
[ ] edma	2.1.1
[ ] enet	2.1.1

utilities

Name	Vers...
[ ] debug_cons	1.0.0
[ ] notifier	1.0.0
[ ] shell	1.0.0

Drivers: None selected by default. Can be added by marking checkbox

Semihosting enabled by default

CMSIS-Core files are added by default

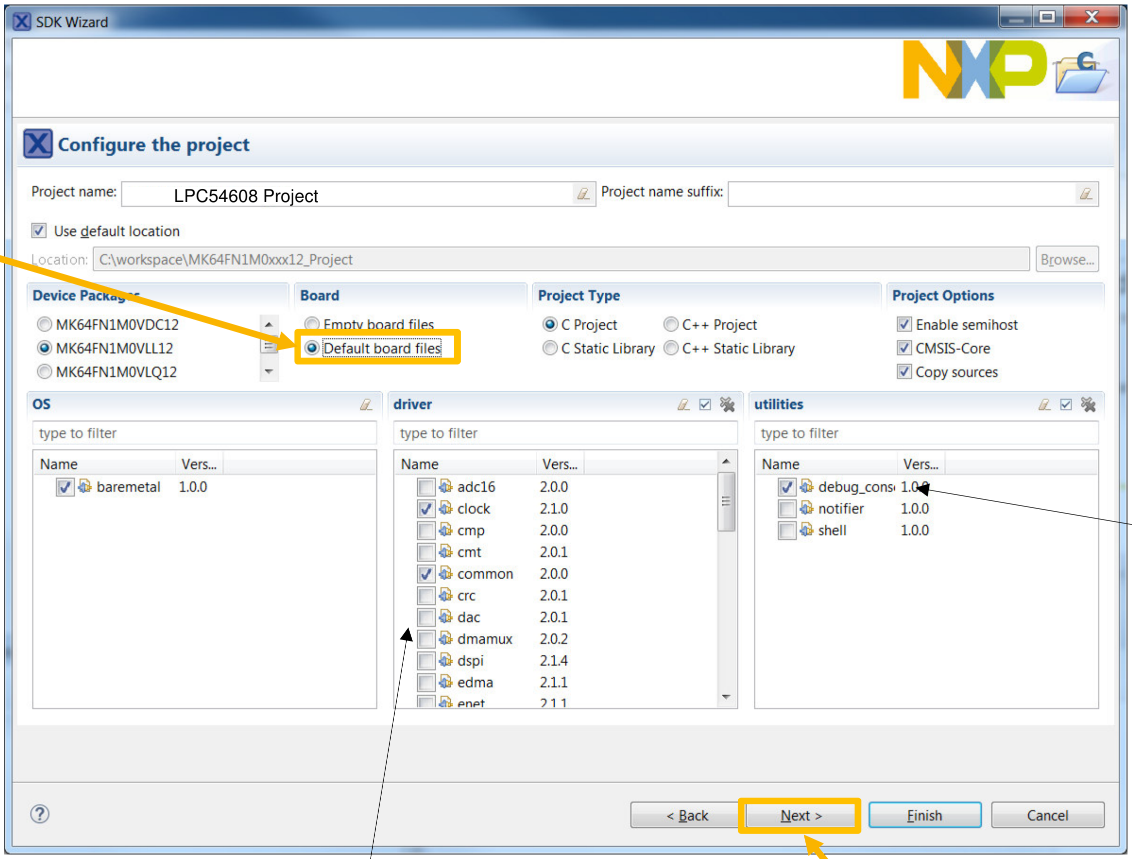
Source files are copied by default. De-select for linked references (only available if SDK is unzipped)

Utilities: None selected by default. Can be added by marking checkbox

New Project for Board

4. Select Default Board files

Adds initialization code in board files for pins, clocks, and debug console


SDK Wizard

Configure the project

Project name: LPC54608 Project
Project name suffix:

[✓] Use default location

Location: C:\workspace\MK64FN1M0xxx12_Project

Device Packages	Board	Project Type	Project Options
○ MK64FN1M0VDC12	○ Empty board files	● C Project ○ C++ Project	[✓] Enable semihost
● MK64FN1M0VLL12	● Default board files	○ C Static Library ○ C++ Static Library	[✓] CMSIS-Core
○ MK64FN1M0VLQ12			[✓] Copy sources

OS

	Name	Vers...
✓	baremetal	1.0.0

driver

	Name	Vers...
	adc16	2.0.0
✓	clock	2.1.0
	cmp	2.0.0
	cmt	2.0.1
✓	common	2.0.0
	crc	2.0.1
	dac	2.0.1
	dmamux	2.0.2
	dspi	2.1.4
	edma	2.1.1
	enet	2.1.1

utilities

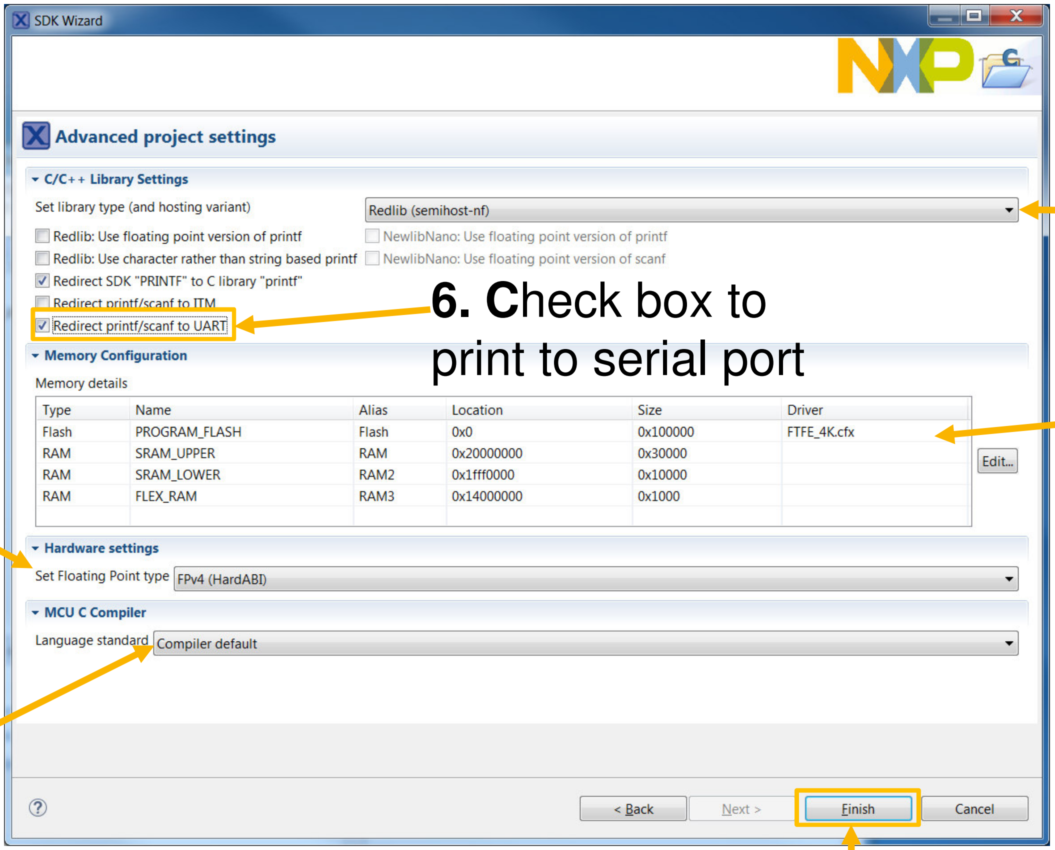
	Name	Vers...
✓	debug_cons	1.0.0
	notifier	1.0.0
	shell	1.0.0

Adds debug_console support for board

Adds drivers to support initialization

5. Select Next to continue

New Project Advanced Settings



Set library type (and hosting variant): Redlib (semihost-nf)

- [ ] Redlib: Use floating point version of printf
- [ ] NewlibNano: Use floating point version of printf
- [ ] Redlib: Use character rather than string based printf
- [ ] NewlibNano: Use floating point version of scanf
- [x] Redirect SDK "PRINTF" to C library "printf"
- [ ] Redirect printf/scanf to ITM
- [x] Redirect printf/scanf to UART

Memory details

Type	Name	Alias	Location	Size	Driver
Flash	PROGRAM_FLASH	Flash	0x0	0x100000	FTFE_4K.cfx
RAM	SRAM_UPPER	RAM	0x20000000	0x30000	
RAM	SRAM_LOWER	RAM2	0x1fff0000	0x10000	
RAM	FLEX_RAM	RAM3	0x14000000	0x1000	

Set Floating Point type: FPv4 (HardABI)

Language standard: Compiler default

Library Selection

6. Check box to print to serial port

Memory Configuration

Floating Point Setup

Compiler selection

7. Click Finish

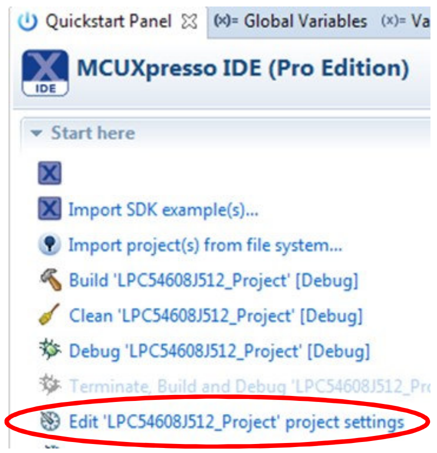
Changing Project Settings

Open the Properties for the “myproj” project via the Quickstart Panel

Switch to Compiler Optimization settings

Settings are for specified Build Configuration

Change optimization level, click OK, then trigger a build



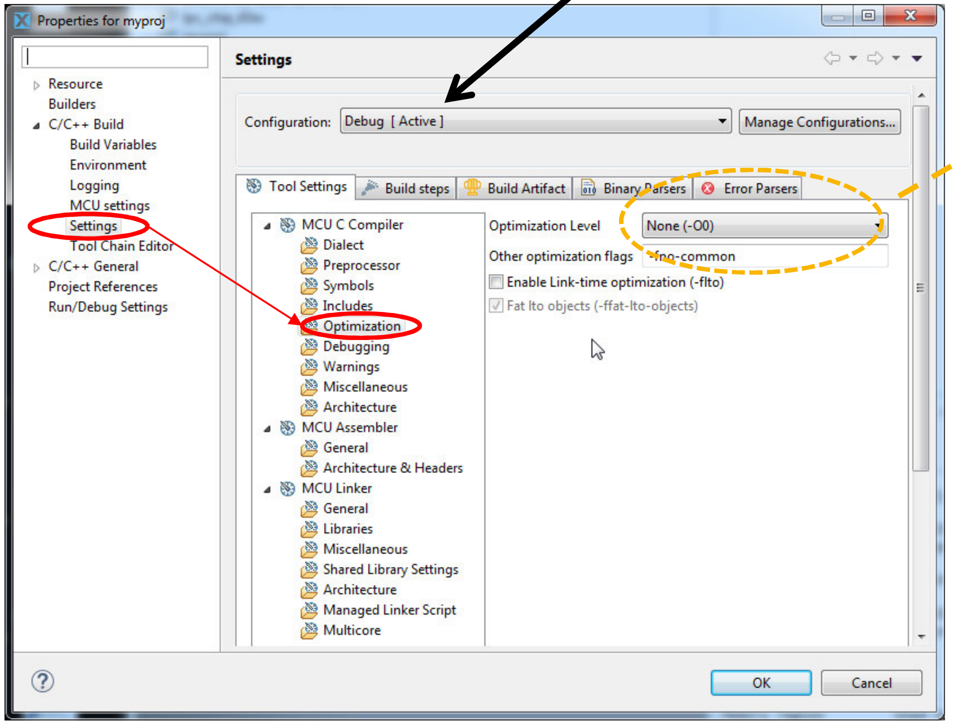
Quickstart Panel | Global Variables

MCUXpresso IDE (Pro Edition)

Start here

- Import SDK example(s)...
- Import project(s) from file system...
- Build 'LPC54608J512_Project' [Debug]
- Clean 'LPC54608J512_Project' [Debug]
- Debug 'LPC54608J512_Project' [Debug]
- Terminate, Build and Debug 'LPC54608J512_Pr
- Edit 'LPC54608J512_Project' project settings

Or use “Properties” entry on Project Explorer right-click menu or press Alt-Menu (Windows)



Properties for myproj

Settings

Configuration: Debug [Active] | Manage Configurations...

Tool Settings | Build steps | Build Artifact | Binary Parsers | Error Parsers

- Resource
- Builders
- C/C++ Build
  - Build Variables
  - Environment
  - Logging
  - MCU settings
  - Settings
  - Tool Chain Editor
- C/C++ General
- Project References
- Run/Debug Settings

- MCU C Compiler
  - Dialect
  - Preprocessor
  - Symbols
  - Includes
  - Optimization
  - Debugging
  - Warnings
  - Miscellaneous
  - Architecture
- MCU Assembler
  - General
  - Architecture & Headers
- MCU Linker
  - General
  - Libraries
  - Miscellaneous
  - Shared Library Settings
  - Architecture
  - Managed Linker Script
  - Multicore

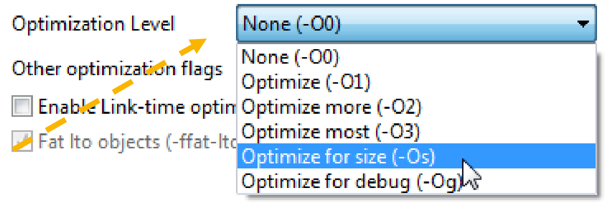
Optimization Level: None (-O0)

Other optimization flags: -fno-common

☐ Enable Link-time optimization (-flto)

☑ Fat lto objects (-ffat-lto-objects)

OK | Cancel



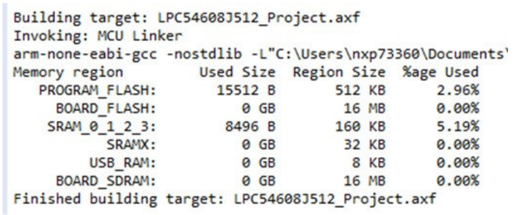
Optimization Level: None (-O0)

Other optimization flags

☐ Enable Link-time optim

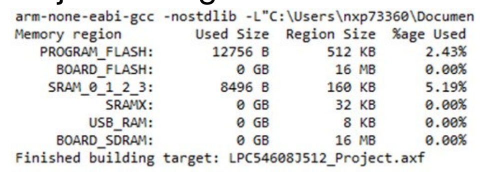
☑ Fat lto objects (-ffat-lto

Default Build at -O0



```
Building target: LPC54608J512_Project.axf
Invoking: MCU Linker
arm-none-eabi-gcc -nostdlib -L"C:\Users\nxp73360\Documents'
Memory region         Used Size  Region Size  %age Used
   PROGRAM_FLASH:       15512 B       512 KB      2.96%
     BOARD_FLASH:          0 GB        16 MB      0.00%
     SRAM_0_1_2_3:       8496 B       160 KB      5.19%
           SRAMX:          0 GB        32 KB      0.00%
         USB_RAM:          0 GB         8 KB      0.00%
     BOARD_SDRAM:          0 GB        16 MB      0.00%
Finished building target: LPC54608J512_Project.axf
```

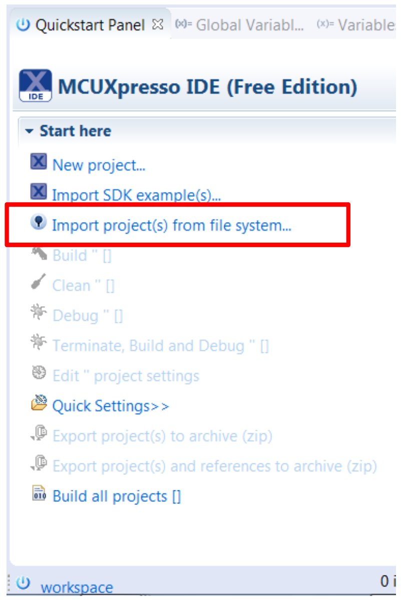
Project changed to -Os



```
arm-none-eabi-gcc -nostdlib -L"C:\Users\nxp73360\Documen
Memory region         Used Size  Region Size  %age Used
   PROGRAM_FLASH:       12756 B       512 KB      2.43%
     BOARD_FLASH:          0 GB        16 MB      0.00%
     SRAM_0_1_2_3:       8496 B       160 KB      5.19%
           SRAMX:          0 GB        32 KB      0.00%
         USB_RAM:          0 GB         8 KB      0.00%
     BOARD_SDRAM:          0 GB        16 MB      0.00%
Finished building target: LPC54608J512_Project.axf
```

Note : We have only changed the application, not library projects

Import Project from File System

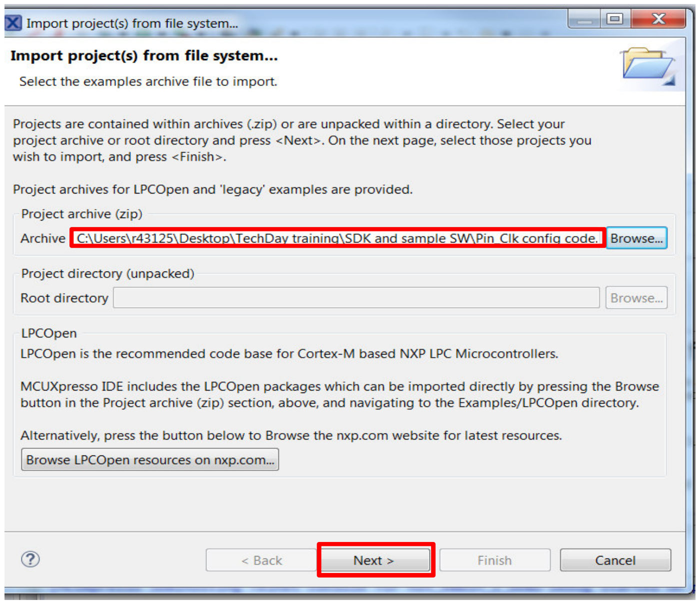

Quickstart Panel ☒ Global Variabl... Variable

MCUXpresso IDE (Free Edition)

Start here

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

workspace


Import project(s) from file system...

Import project(s) from file system...

Select the examples archive file to import.

Projects are contained within archives (.zip) or are unpacked within a directory. Select your project archive or root directory and press <Next>. On the next page, select those projects you wish to import, and press <Finish>.

Project archives for LPCOpen and 'legacy' examples are provided.

Project archive (zip)

Archive C:\Users\r43125\Desktop\TechDay training\SDK and sample SW\Pin_Clk config code. Browse...

Project directory (unpacked)

Root directory Browse...

LPCOpen

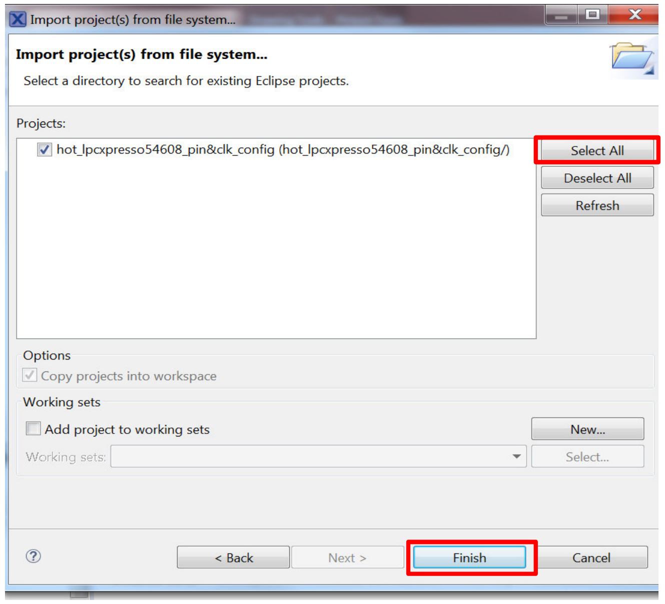
LPCOpen is the recommended code base for Cortex-M based NXP LPC Microcontrollers.

MCUXpresso IDE includes the LPCOpen packages which can be imported directly by pressing the Browse button in the Project archive (zip) section, above, and navigating to the Examples/LPCOpen directory.

Alternatively, press the button below to Browse the nxp.com website for latest resources.

Browse LPCOpen resources on nxp.com...

< Back | Next > | Finish | Cancel


Import project(s) from file system...

Import project(s) from file system...

Select a directory to search for existing Eclipse projects.

Projects:

☑ hot_lpcxpresso54608_pin&clk_config (hot_lpcxpresso54608_pin&clk_config/)

Select All | Deselect All | Refresh

Options

☑ Copy projects into workspace

Working sets

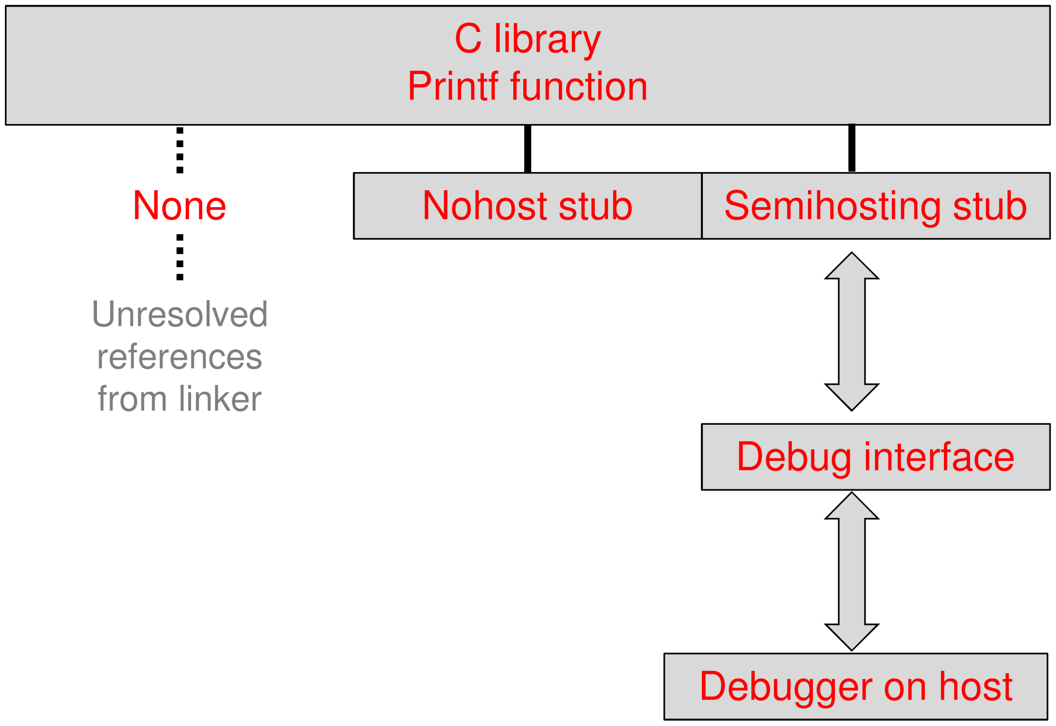
☐ Add project to working sets | New...

Working sets: | Select...

< Back | Next > | Finish | Cancel

Library Variants

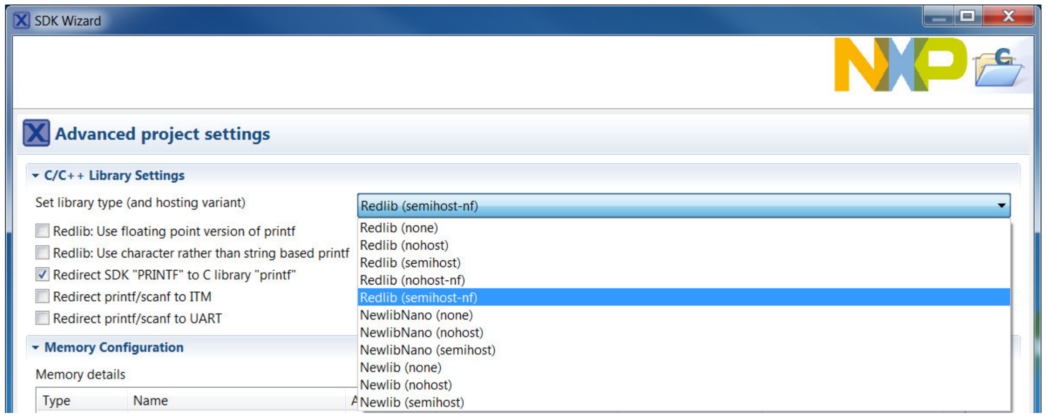
- Libraries are provided in number of variants, with different underlying “stub” providing support functions:
  - None
    - Smallest footprint. Excludes low-level file I/O
    - For Newlib, excludes memory handling functions
  - Nohost and Nohost-nf
    - Provides memory handling functions and some file I/O.
    - However, it assumes no host, and so file I/O will do nothing
  - Semihost-nf (no files)
    - Reldlib only
    - Similar to Semihost but only supports 3 standard built in streams (stdin, stdout, stderr)
    - Reduces memory overhead, but application cannot open files
  - Semihost
    - Full functionality
    - I/O resources are on the host side



- More C library information at:
  - http://community.nxp.com (MCUXpresso IDE FAQs)

C/C++ Library Selection

- C projects
  - Default to Redlib
  - C90 library, with some C99 extensions
  - Optimized for code size
  - Select use of integer printf in wizard
- C++
  - Default to Newlib
  - Provides C++ support, plus full C99
  - Can switch C projects to use Newlib if required
- MCUXpresso also supports “Newlib-Nano”
  - Code size optimized version of Newlib
  - Can switch C or C++ projects to use this
  - Integer only printf by default – enable floating point in Linker options


SDK Wizard

Advanced project settings

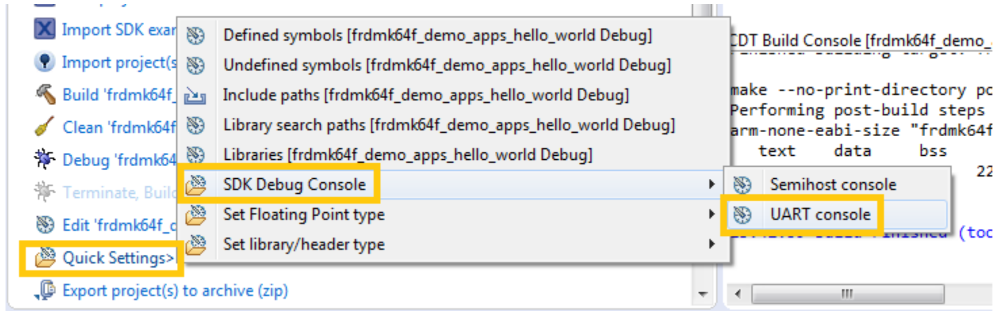
C/C++ Library Settings

Set library type (and hosting variant): Redlib (semihost-nf)

- [ ] Redlib: Use floating point version of printf
- [ ] Redlib: Use character rather than string based printf
- [x] Redirect SDK "PRINTF" to C library "printf"
- [ ] Redirect printf/scanf to ITM
- [ ] Redirect printf/scanf to UART

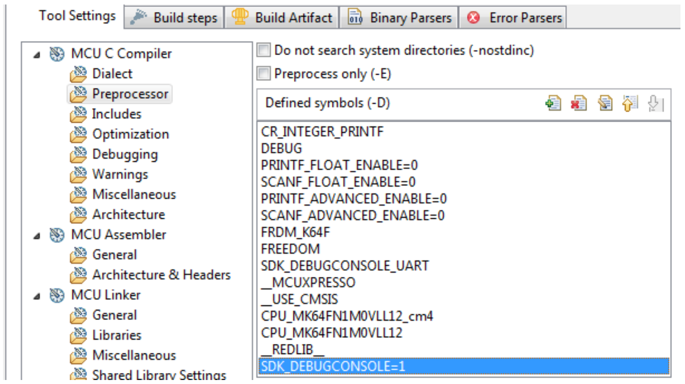
Change to UART Console

- If user forgets to check the box to redirect printf/scanf to UART, can change in project via the Quick Settings:


Import SDK exar
Import project(s
Build 'frdmk64f_
Clean 'frdmk64f
Debug 'frdmk64
Terminate, Build
Edit 'frdmk64f_c
Quick Settings>
Export project(s) to archive (zip)
Defined symbols [frdmk64f_demo_apps_hello_world Debug]
Undefined symbols [frdmk64f_demo_apps_hello_world Debug]
Include paths [frdmk64f_demo_apps_hello_world Debug]
Library search paths [frdmk64f_demo_apps_hello_world Debug]
Libraries [frdmk64f_demo_apps_hello_world Debug]
SDK Debug Console
Set Floating Point type
Set library/header type
Semihost console
UART console
CDT Build Console [frdmk64f_demo_

```
make --no-print-directory pc
Performing post-build steps
arm-none-eabi-size "frdmk64f
   text    data     bss
                      22
```

- Verify setting in Project Settings->Preprocessor that SDK_DEBUGCONSOLE=1



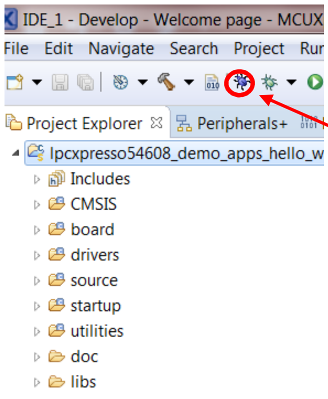
☐ Do not search system directories (-nostdinc)
☐ Preprocess only (-E)

Defined symbols (-D)

```
CR_INTEGER_PRINTF
DEBUG
PRINTF_FLOAT_ENABLE=0
SCANF_FLOAT_ENABLE=0
PRINTF_ADVANCED_ENABLE=0
SCANF_ADVANCED_ENABLE=0
FRDM_K64F
FREEDOM
SDK_DEBUGCONSOLE_UART
__MCUXPRESSO
__USE_CMSIS
CPU_MK64FN1M0VLL12_cm4
CPU_MK64FN1M0VLL12
__REDLIB__
SDK_DEBUGCONSOLE=1
```

MCUXPRESSO IDE DEBUG

Start a Debug Session



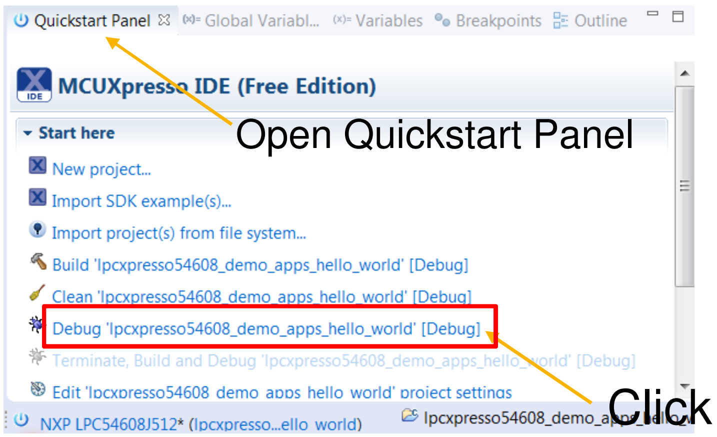
IDE_1 - Develop - Welcome page - MCUX

File Edit Navigate Search Project Rur

Project Explorer ☒ Peripherals+

- lpcxpresso54608_demo_apps_hello_w
  - Includes
  - CMSIS
  - board
  - drivers
  - source
  - startup
  - utilities
  - doc
  - libs

Note: Do not use this icon,
Use the Quickstart Panel instead



Quickstart Panel ☒ Global Variabl... Variables Breakpoints Outline

MCUXpresso IDE (Free Edition)

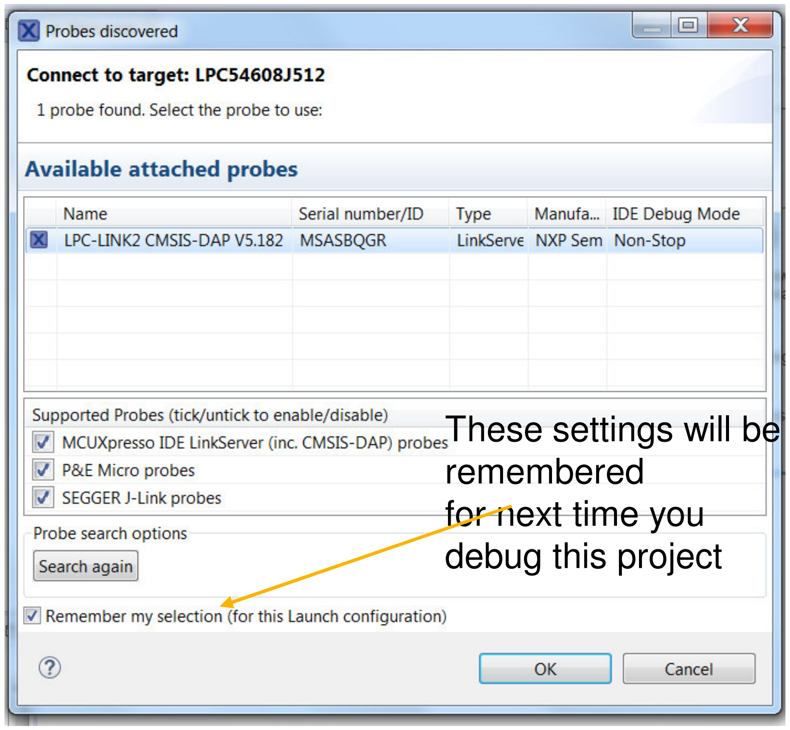
Start here

- New project...
- Import SDK example(s)...
- Import project(s) from file system...
- Build 'lpcxpresso54608_demo_apps_hello_world' [Debug]
- Clean 'lpcxpresso54608_demo_apps_hello_world' [Debug]
- Debug 'lpcxpresso54608_demo_apps_hello_world' [Debug]
- Terminate, Build and Debug 'lpcxpresso54608_demo_apps_hello_world' [Debug]
- Edit 'lpcxpresso54608_demo_apps_hello_world' project settings

NXP LPC54608J512* (lpcxpresso...ello_world) lpcxpresso54608_demo_apps_hello_w

Open Quickstart Panel

Click Debug



Probes discovered

Connect to target: LPC54608J512

1 probe found. Select the probe to use:

Available attached probes

Name	Serial number/ID	Type	Manufa...	IDE Debug Mode
LPC-LINK2 CMSIS-DAP V5.182	MSASBQGR	LinkServe	NXP Sem	Non-Stop

Supported Probes (tick/untick to enable/disable)

- ☑ MCUXpresso IDE LinkServer (inc. CMSIS-DAP) probes
- ☑ P&E Micro probes
- ☑ SEGGER J-Link probes

Probe search options

Search again

☑ Remember my selection (for this Launch configuration)

OK Cancel

These settings will be remembered for next time you debug this project

Note: By default, selecting “Debug” will trigger a build before the debug session is launched, so it is not required to run a “build” first

Debug Perspective

Registers and **Peripherals** View

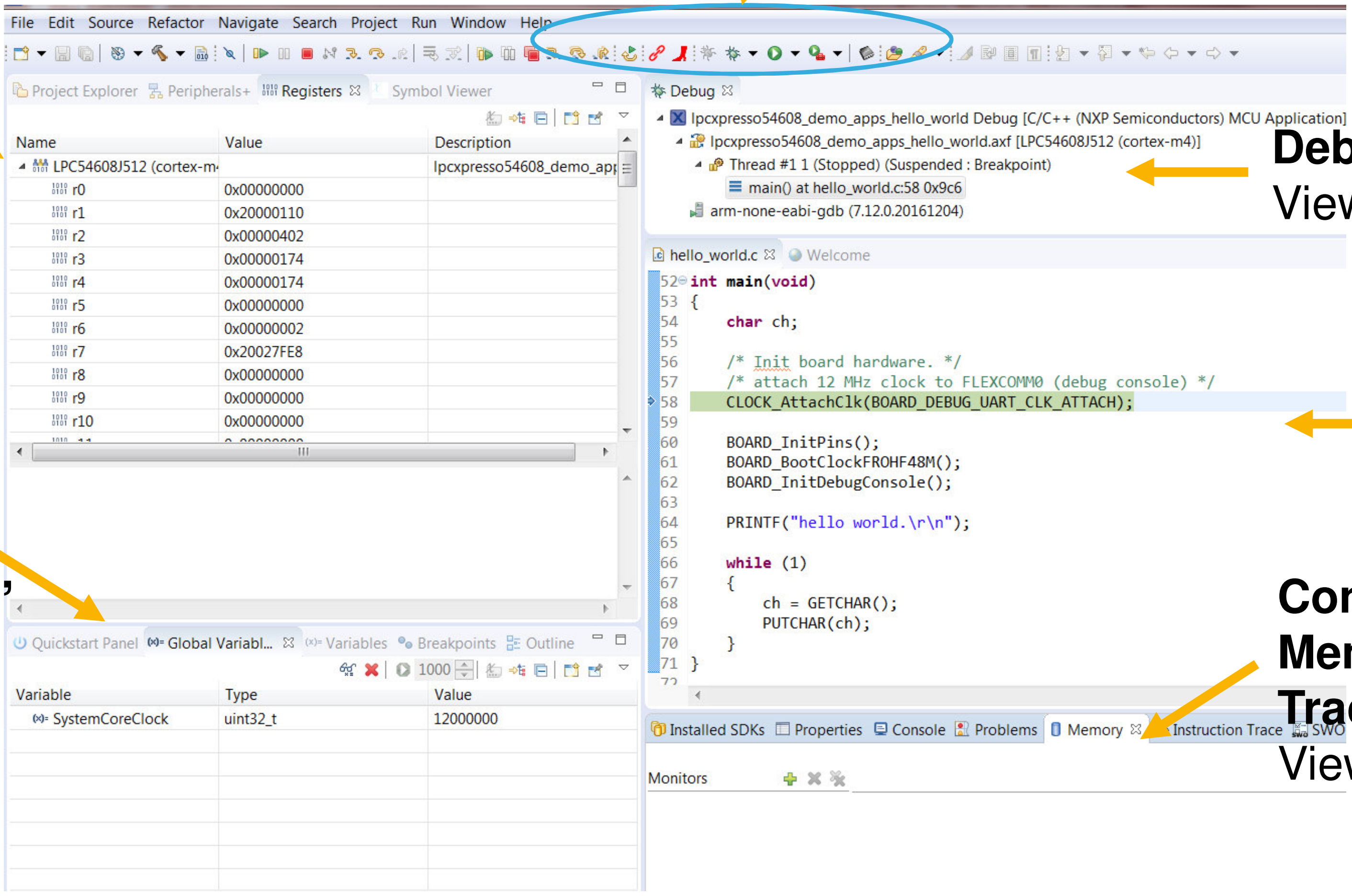
Variables, Expressions, and **Breakpoints** View

Run Controls

Debug View

Editor View

Console, Memory, and **Trace** Views



Name	Value	Description
LPC54608J512 (cortex-m4		lpcxpresso54608_demo_app
r0	0x00000000	
r1	0x20000110	
r2	0x00000402	
r3	0x00000174	
r4	0x00000174	
r5	0x00000000	
r6	0x00000002	
r7	0x20027FE8	
r8	0x00000000	
r9	0x00000000	
r10	0x00000000	

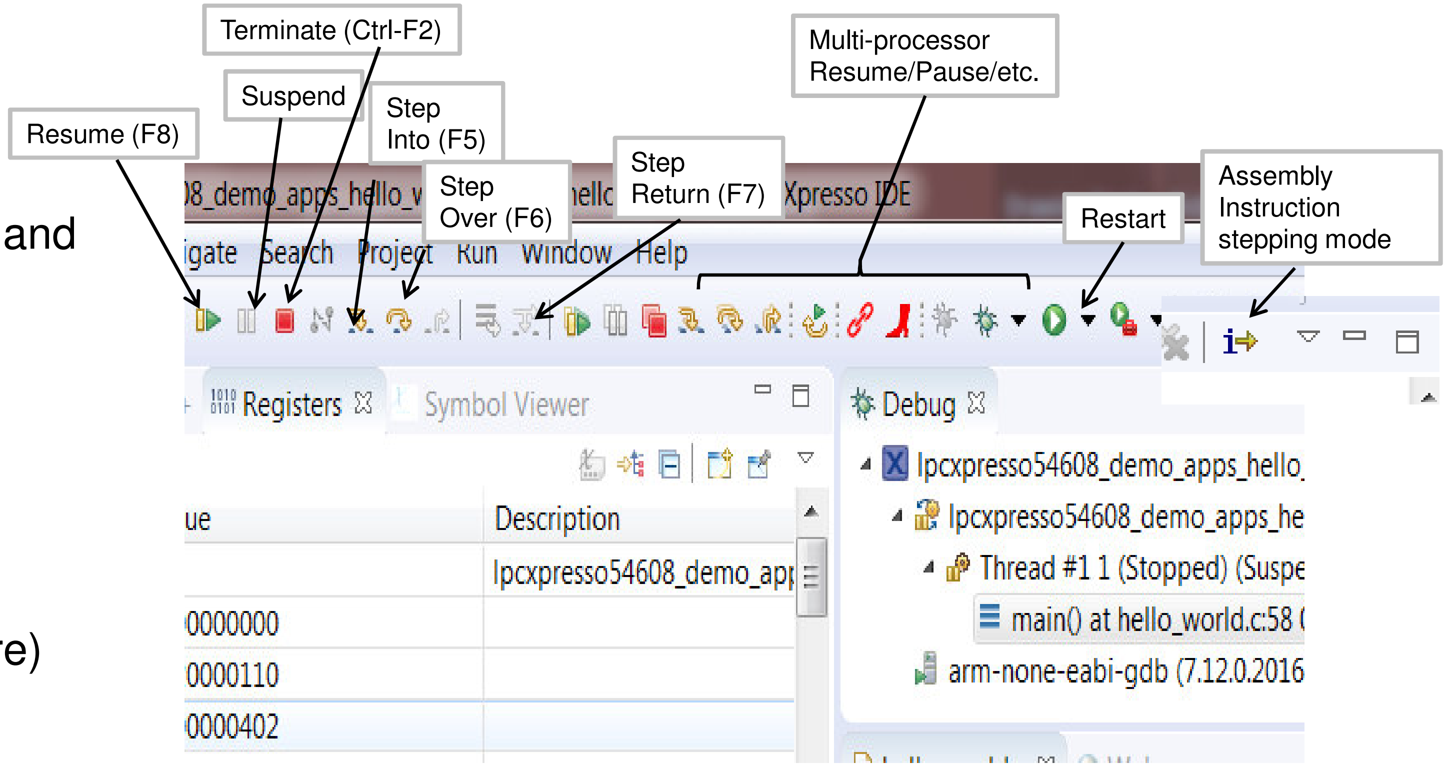
```
lpcxpresso54608_demo_apps_hello_world Debug [C/C++ (NXP Semiconductors) MCU Application]
  lpcxpresso54608_demo_apps_hello_world.axf [LPC54608J512 (cortex-m4)]
    Thread #1 1 (Stopped) (Suspended : Breakpoint)
      main() at hello_world.c:58 0x9c6
  arm-none-eabi-gdb (7.12.0.20161204)
```

```c
int main(void)
{
    char ch;

    /* Init board hardware. */
    /* attach 12 MHz clock to FLEXCOMM0 (debug console) */
    CLOCK_AttachClk(BOARD_DEBUG_UART_CLK_ATTACH);

    BOARD_InitPins();
    BOARD_BootClockFROHF48M();
    BOARD_InitDebugConsole();

    PRINTF("hello world.\r\n");

    while (1)
    {
        ch = GETCHAR();
        PUTCHAR(ch);
    }
}
```

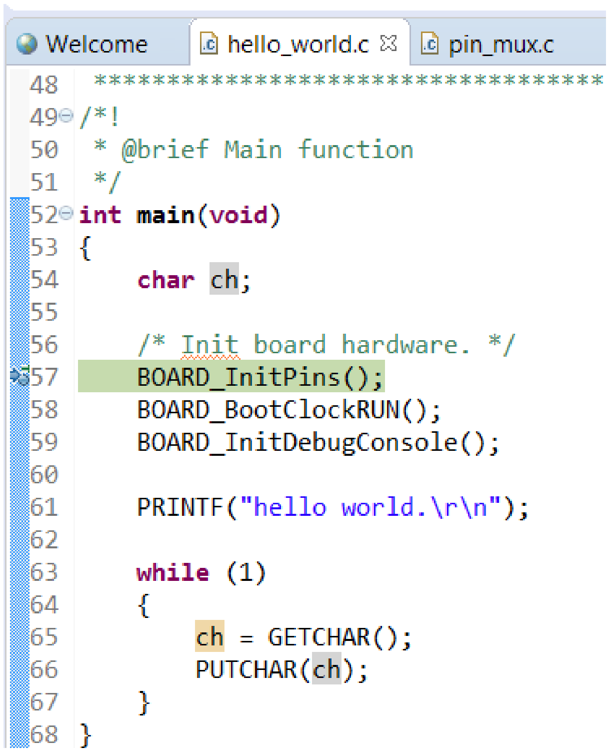
Variable	Type	Value
SystemCoreClock	uint32_t	12000000

Stopped At Main()

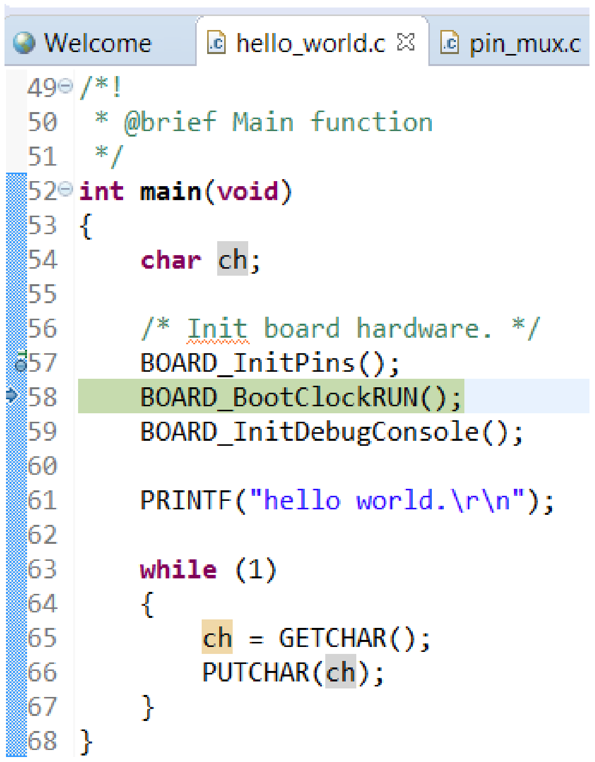
- Image downloaded to flash and execution started
  - Default breakpoint set on function main()
- Debug View displayed automatically
  - Shows / controls current scope and target (multicore)
  - Run controls are on main toolbar
- **But before you begin to run the code …**


Terminate (Ctrl-F2)
Suspend
Step Into (F5)
Resume (F8)
Step Over (F6)
Step Return (F7)
Multi-processor Resume/Pause/etc.
Restart
Assembly Instruction stepping mode
Registers
Symbol Viewer
Debug
Description
lpcxpresso54608_demo_apps_hello_
lpcxpresso54608_demo_apps_he
Thread #1 1 (Stopped) (Suspe
main() at hello_world.c:58
arm-none-eabi-gdb (7.12.0.2016

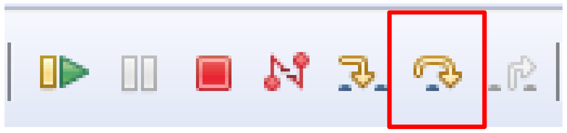
Debug: Step Over



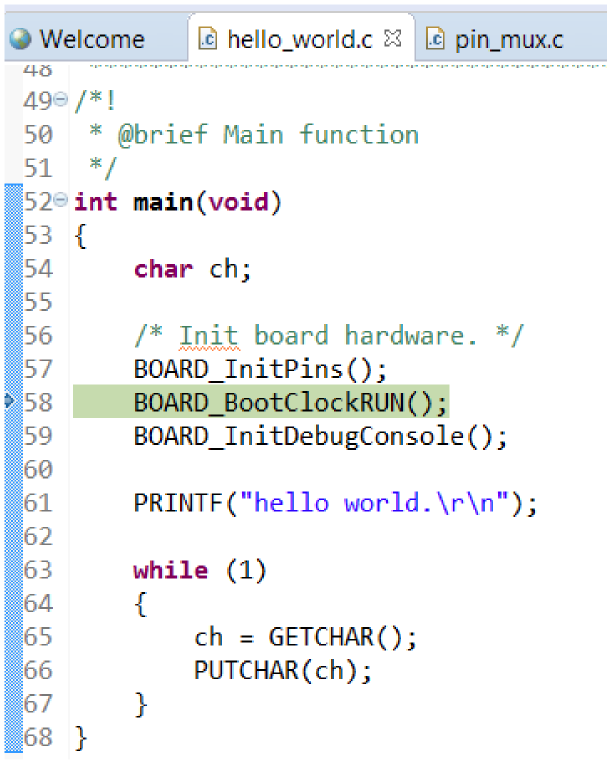
```
 ***********************************
/*!
 * @brief Main function
 */
int main(void)
{
    char ch;

    /* Init board hardware. */
    BOARD_InitPins();
    BOARD_BootClockRUN();
    BOARD_InitDebugConsole();

    PRINTF("hello world.\r\n");

    while (1)
    {
        ch = GETCHAR();
        PUTCHAR(ch);
    }
}
```



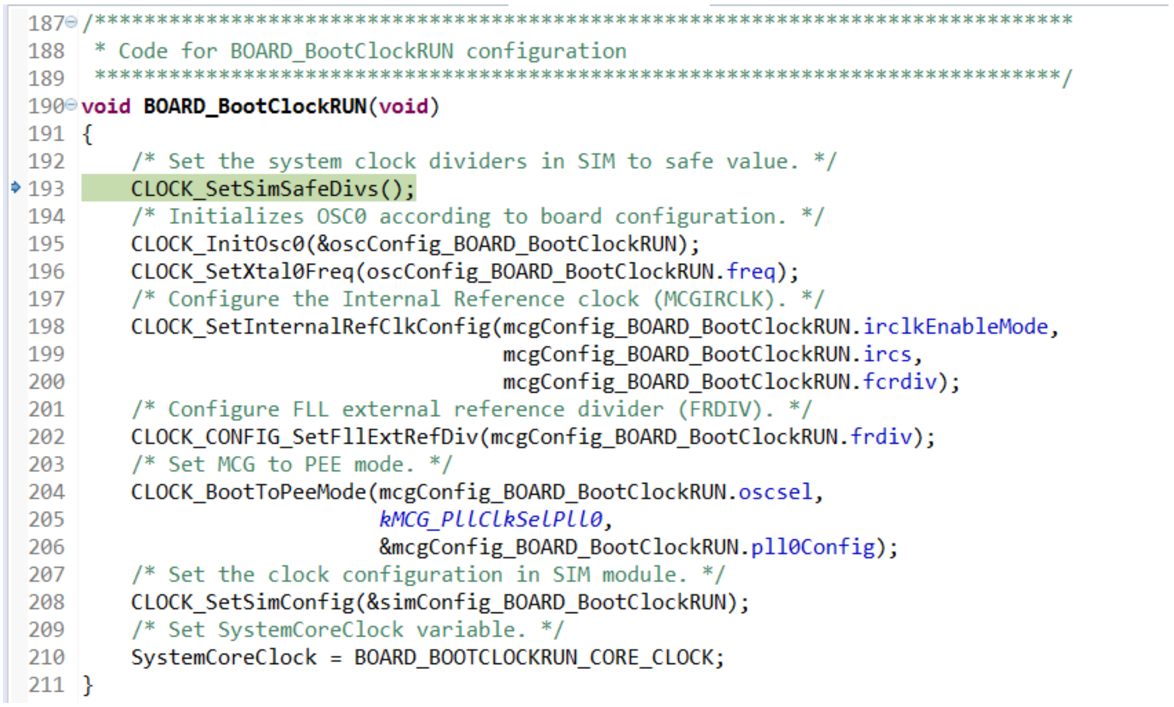
```
/*!
 * @brief Main function
 */
int main(void)
{
    char ch;

    /* Init board hardware. */
    BOARD_InitPins();
    BOARD_BootClockRUN();
    BOARD_InitDebugConsole();

    PRINTF("hello world.\r\n");

    while (1)
    {
        ch = GETCHAR();
        PUTCHAR(ch);
    }
}
```



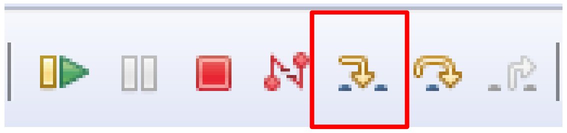
Debug: Step Into Function



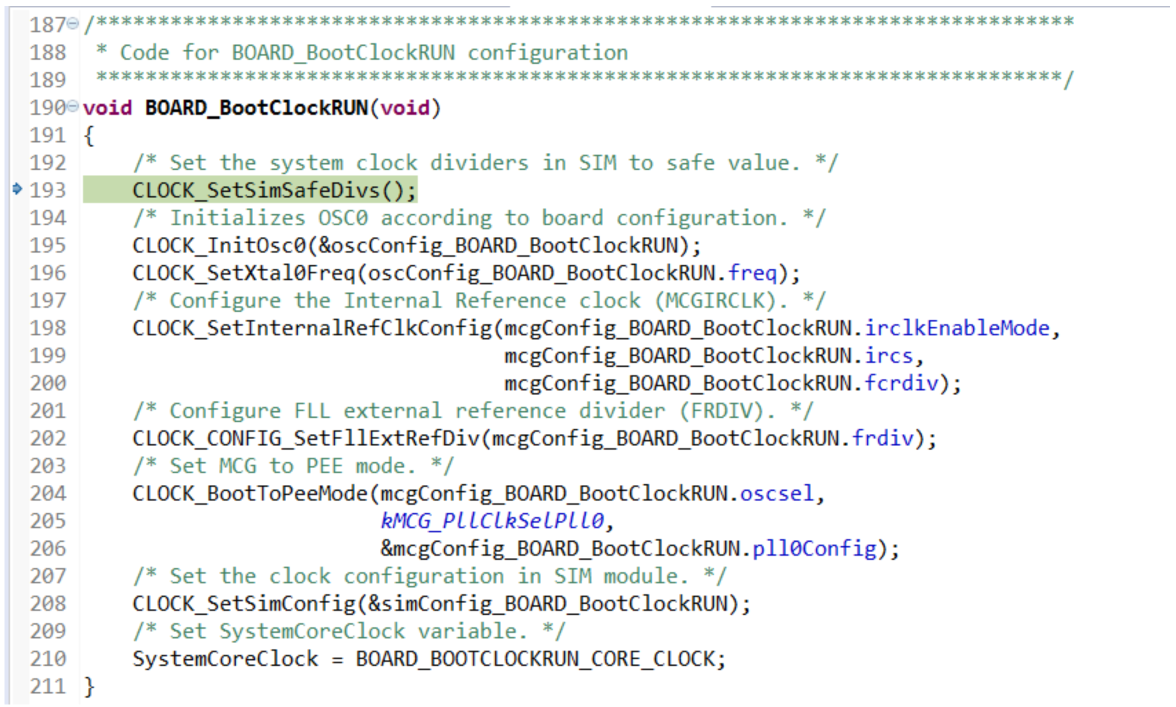
```
/*!
 * @brief Main function
 */
int main(void)
{
    char ch;

    /* Init board hardware. */
    BOARD_InitPins();
    BOARD_BootClockRUN();
    BOARD_InitDebugConsole();

    PRINTF("hello world.\r\n");

    while (1)
    {
        ch = GETCHAR();
        PUTCHAR(ch);
    }
}
```



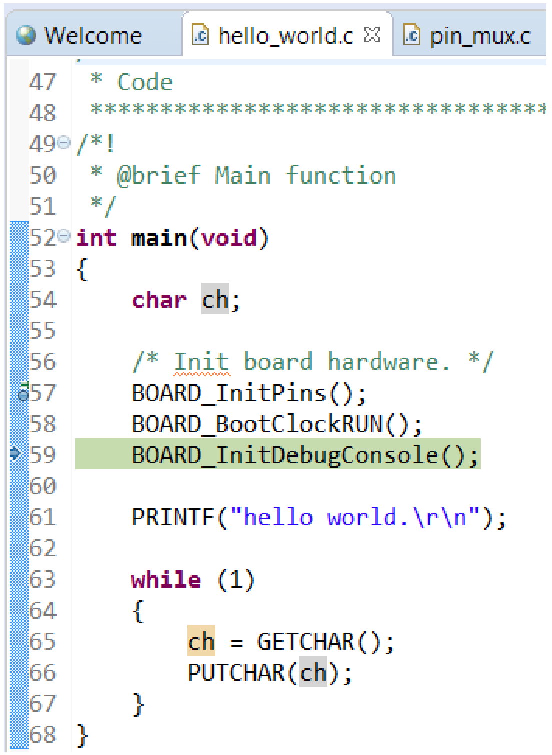
```
/*******************************************************************************
 * Code for BOARD_BootClockRUN configuration
 ******************************************************************************/
void BOARD_BootClockRUN(void)
{
    /* Set the system clock dividers in SIM to safe value. */
    CLOCK_SetSimSafeDivs();
    /* Initializes OSC0 according to board configuration. */
    CLOCK_InitOsc0(&oscConfig_BOARD_BootClockRUN);
    CLOCK_SetXtal0Freq(oscConfig_BOARD_BootClockRUN.freq);
    /* Configure the Internal Reference clock (MCGIRCLK). */
    CLOCK_SetInternalRefClkConfig(mcgConfig_BOARD_BootClockRUN.irclkEnableMode,
                                  mcgConfig_BOARD_BootClockRUN.ircs,
                                  mcgConfig_BOARD_BootClockRUN.fcrdiv);
    /* Configure FLL external reference divider (FRDIV). */
    CLOCK_CONFIG_SetFllExtRefDiv(mcgConfig_BOARD_BootClockRUN.frdiv);
    /* Set MCG to PEE mode. */
    CLOCK_BootToPeeMode(mcgConfig_BOARD_BootClockRUN.oscsel,
                        kMCG_PllClkSelPll0,
                        &mcgConfig_BOARD_BootClockRUN.pll0Config);
    /* Set the clock configuration in SIM module. */
    CLOCK_SetSimConfig(&simConfig_BOARD_BootClockRUN);
    /* Set SystemCoreClock variable. */
    SystemCoreClock = BOARD_BOOTCLOCKRUN_CORE_CLOCK;
}
```



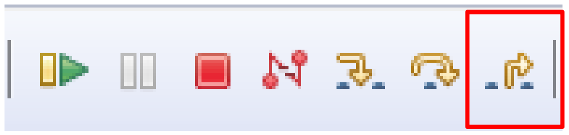
Debug: Step Return



```
/*****************************************************************************
 * Code for BOARD_BootClockRUN configuration
 ****************************************************************************/
void BOARD_BootClockRUN(void)
{
    /* Set the system clock dividers in SIM to safe value. */
    CLOCK_SetSimSafeDivs();
    /* Initializes OSC0 according to board configuration. */
    CLOCK_InitOsc0(&oscConfig_BOARD_BootClockRUN);
    CLOCK_SetXtal0Freq(oscConfig_BOARD_BootClockRUN.freq);
    /* Configure the Internal Reference clock (MCGIRCLK). */
    CLOCK_SetInternalRefClkConfig(mcgConfig_BOARD_BootClockRUN.irclkEnableMode,
                                  mcgConfig_BOARD_BootClockRUN.ircs,
                                  mcgConfig_BOARD_BootClockRUN.fcrdiv);
    /* Configure FLL external reference divider (FRDIV). */
    CLOCK_CONFIG_SetFllExtRefDiv(mcgConfig_BOARD_BootClockRUN.frdiv);
    /* Set MCG to PEE mode. */
    CLOCK_BootToPeeMode(mcgConfig_BOARD_BootClockRUN.oscsel,
                        kMCG_PllClkSelPll0,
                        &mcgConfig_BOARD_BootClockRUN.pll0Config);
    /* Set the clock configuration in SIM module. */
    CLOCK_SetSimConfig(&simConfig_BOARD_BootClockRUN);
    /* Set SystemCoreClock variable. */
    SystemCoreClock = BOARD_BOOTCLOCKRUN_CORE_CLOCK;
}
```



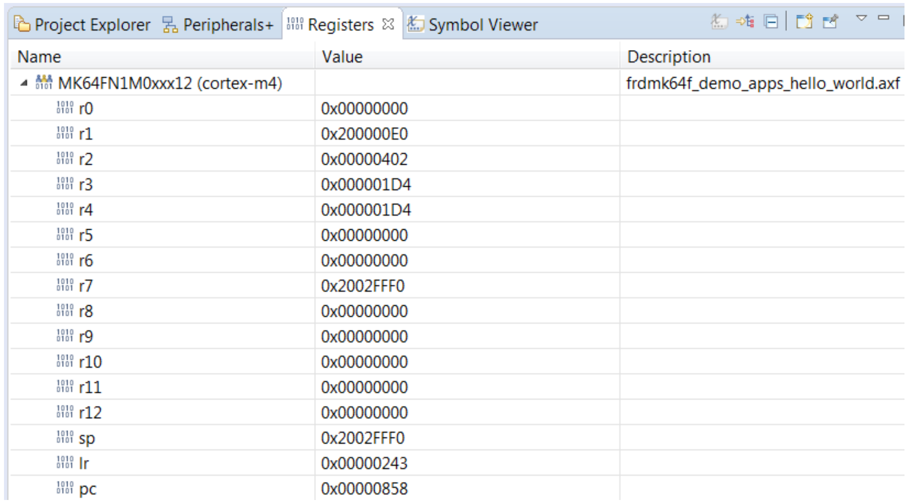
```
 * Code
 **********************************
/*!
 * @brief Main function
 */
int main(void)
{
    char ch;

    /* Init board hardware. */
    BOARD_InitPins();
    BOARD_BootClockRUN();
    BOARD_InitDebugConsole();

    PRINTF("hello world.\r\n");

    while (1)
    {
        ch = GETCHAR();
        PUTCHAR(ch);
    }
}
```



Registers, Local Variables, and Memory Views

Registers View:

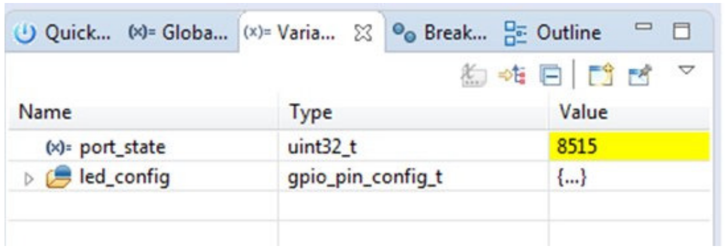
- CPU registers are displayed. Will highlight in yellow when contents update



Name	Value	Description
MK64FN1M0xxx12 (cortex-m4)		frdmk64f_demo_apps_hello_world.axf
r0	0x00000000	
r1	0x200000E0	
r2	0x00000402	
r3	0x000001D4	
r4	0x000001D4	
r5	0x00000000	
r6	0x00000000	
r7	0x2002FFF0	
r8	0x00000000	
r9	0x00000000	
r10	0x00000000	
r11	0x00000000	
r12	0x00000000	
sp	0x2002FFF0	
lr	0x00000243	
pc	0x00000858	

Variables View:

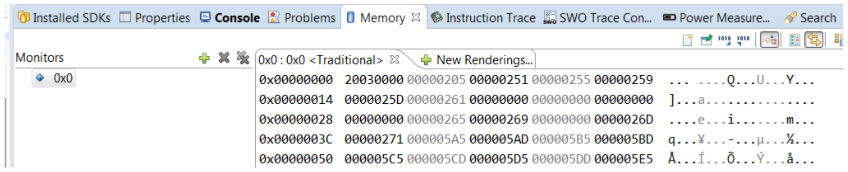
- In-scope local variables displayed
- Locals displayed will change as move up and down the call stack



Name	Type	Value
port_state	uint32_t	8515
led_config	gpio_pin_config_t	{...}

Memory View:

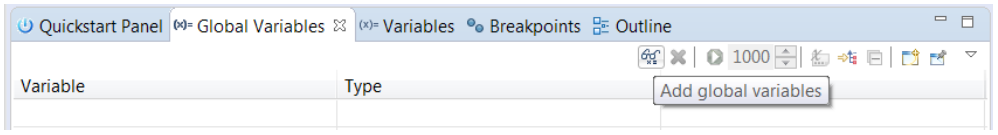
- Add address to display view of memory contents starting at that location



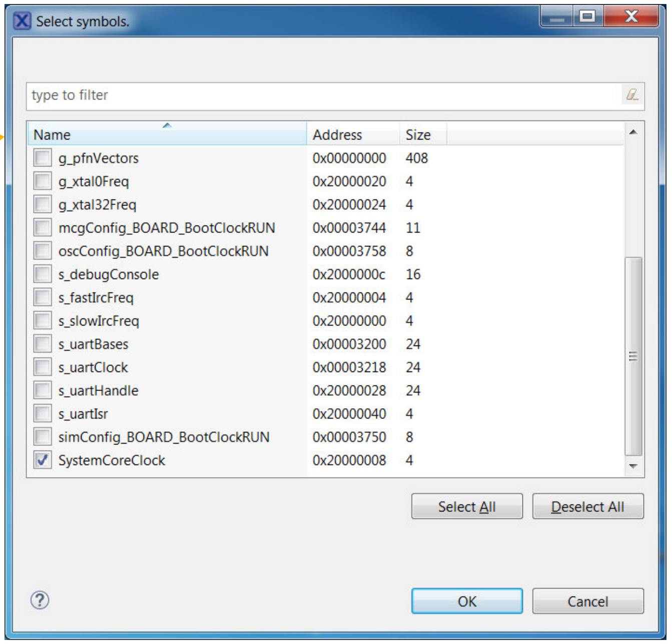
```
0x00000000  20030000 00000205 00000251 00000255 00000259  ... ....Q...U...Y...
0x00000014  0000025D 00000261 00000000 00000000 00000000  ]...a...............
0x00000028  00000000 00000265 00000269 00000000 0000026D  ....e...i.......m...
0x0000003C  00000271 000005A5 000005AD 000005B5 000005BD  q...¥...-...µ...½...
0x00000050  000005C5 000005CD 000005D5 000005DD 000005E5  Å...Í...Õ...Ý...å...
```

Add a Global Variable

1) Switch to the Global Variables View and click on the “Add global variables” button

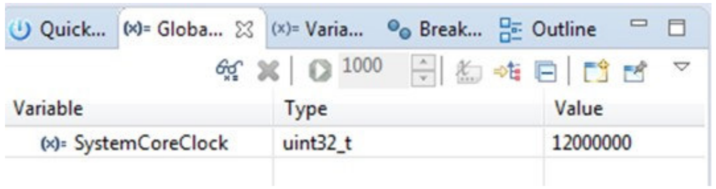


Variable	Type



Name	Address	Size
☐ g_pfnVectors	0x00000000	408
☐ g_xtal0Freq	0x20000020	4
☐ g_xtal32Freq	0x20000024	4
☐ mcgConfig_BOARD_BootClockRUN	0x00003744	11
☐ oscConfig_BOARD_BootClockRUN	0x00003758	8
☐ s_debugConsole	0x2000000c	16
☐ s_fastIrcFreq	0x20000004	4
☐ s_slowIrcFreq	0x20000000	4
☐ s_uartBases	0x00003200	24
☐ s_uartClock	0x00003218	24
☐ s_uartHandle	0x20000028	24
☐ s_uartIsr	0x20000040	4
☐ simConfig_BOARD_BootClockRUN	0x00003750	8
✓ SystemCoreClock	0x20000008	4

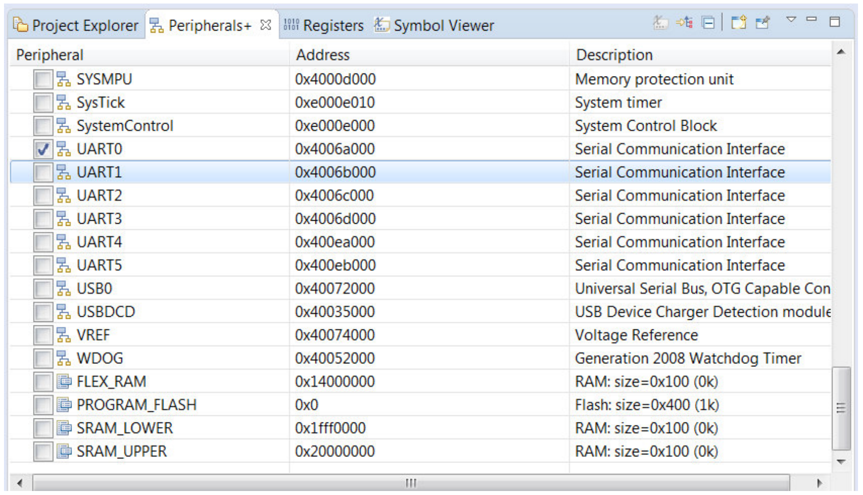
2) Scroll down and select “SystemCoreClock”, which will hold the main CPU clock speed



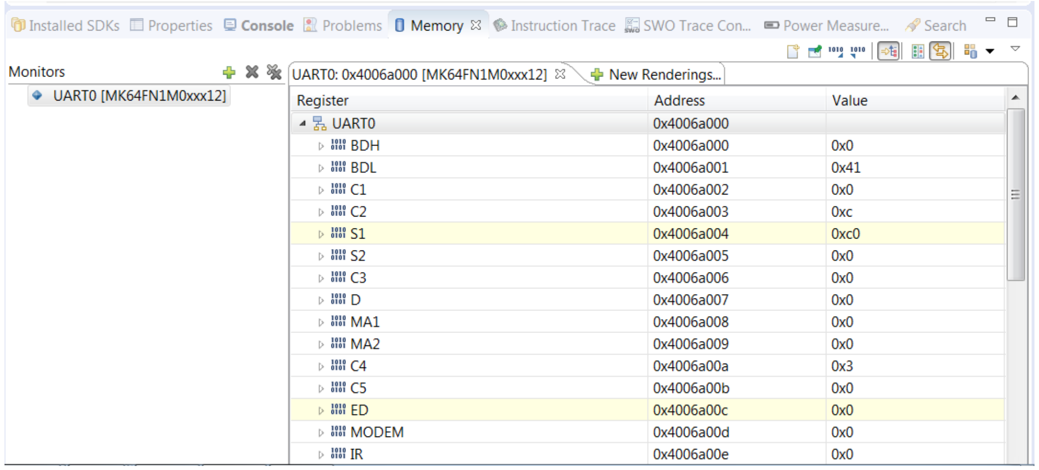
Variable	Type	Value
SystemCoreClock	uint32_t	12000000

3) SystemCoreClock global is now visible in the Expressions View

Peripherals View



Peripheral	Address	Description
SYSMPU	0x4000d000	Memory protection unit
SysTick	0xe000e010	System timer
SystemControl	0xe000e000	System Control Block
✓ UART0	0x4006a000	Serial Communication Interface
UART1	0x4006b000	Serial Communication Interface
UART2	0x4006c000	Serial Communication Interface
UART3	0x4006d000	Serial Communication Interface
UART4	0x400ea000	Serial Communication Interface
UART5	0x400eb000	Serial Communication Interface
USB0	0x40072000	Universal Serial Bus, OTG Capable Con
USBDCD	0x40035000	USB Device Charger Detection module
VREF	0x40074000	Voltage Reference
WDOG	0x40052000	Generation 2008 Watchdog Timer
FLEX_RAM	0x14000000	RAM: size=0x100 (0k)
PROGRAM_FLASH	0x0	Flash: size=0x400 (1k)
SRAM_LOWER	0x1fff0000	RAM: size=0x100 (0k)
SRAM_UPPER	0x20000000	RAM: size=0x100 (0k)



Register	Address	Value
UART0	0x4006a000	
BDH	0x4006a000	0x0
BDL	0x4006a001	0x41
C1	0x4006a002	0x0
C2	0x4006a003	0xc
S1	0x4006a004	0xc0
S2	0x4006a005	0x0
C3	0x4006a006	0x0
D	0x4006a007	0x0
MA1	0x4006a008	0x0
MA2	0x4006a009	0x0
C4	0x4006a00a	0x3
C5	0x4006a00b	0x0
ED	0x4006a00c	0x0
MODEM	0x4006a00d	0x0
IR	0x4006a00e	0x0

- In Peripherals View, click checkbox next to peripheral to select it

- This will open the peripheral in the Memory View
- Expand the peripheral to see details of registers

Sharing Projects

- If a project is built using part support from an SDK and is then exported – for example to share the project with a colleague who also uses MCUXpresso IDE, then the colleague must also install an SDK providing part support for the project's MCU.

- **Note**: it is recommended that any required SDKs are installed before a project requiring SDK part support is imported. However, if this is not done, simply select the imported project in the project explorer and right click and select: *C/C++ Build -> MCU settings* ensure the correct MCU is selected and click **Refresh MCU Cache**.

LAB 2

Lab 2 : To import SDK example and run in MCUXpressor

- **Pre-requisites**
- Boards
  - OM13092(LPCXpresso54608)
- Software
  - SDK_2.2_LPC54608J512 : https://mcuxpresso.nxp.com/en/welcome
  - MCUXpresso IDE: http://nxp.com/mcuxpresso/ide
  - Terminal Software (like TeraTerm or PuTTY)
  - mbed Serial Driver: https://developer.mbed.org/handbook/Windows-serial-configuration
    - Need to install with the board plugged in. Only need to do once per computer.
- Follow Lab 2 instruction

AGENDA

- **MCUXpresso Software And Tools Overview**
- **MCUXpresso SDK**
  - Web Builder
  - File Structure
- **MCUXpresso IDE**
  - Importing/Building
  - Debugging
- **MCUXpresso Config Tool**
  - Project Cloner
  - Pins Tool
  - Clocks Tool
- **LPC54608 LCD Lab, Key API and EmWin Demo**

MCUXPRESSO CONFIG TOOLS

Configurations

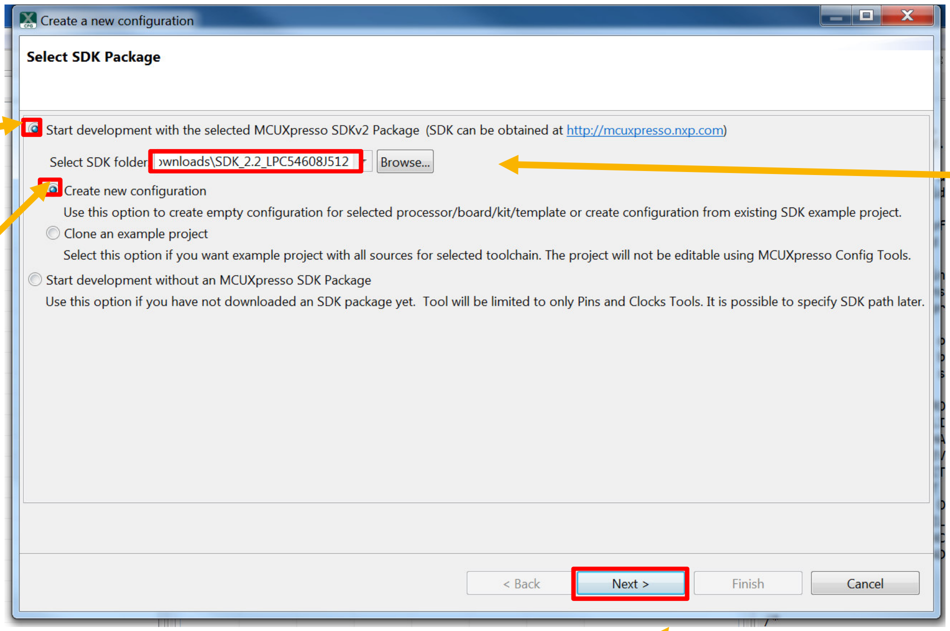
- What is a configuration?
  - A group of settings used across the MCUXpresso configuration tools (Pins, Clocks, and Project Generator)
- SDK provides configurations to start development with specific to the board or processor.
- Users can import a configuration, modify clocks and pin settings, and export the configuration.
- If no SDK is selected, default configurations for boards and processors are available.
- Configurations can be saved and shared as a .mex file

MCUXPRESSO CONFIG TOOLS PROJECT CLONE

Configuration Tool Wizard

Select an SDK to build configurations

Start a configuration



Create a new configuration

Select SDK Package

(●) Start development with the selected MCUXpresso SDKv2 Package (SDK can be obtained at http://mcuxpresso.nxp.com)

Select SDK folder ›wnloads\SDK_2.2_LPC54608J512 Browse...

(●) Create new configuration

Use this option to create empty configuration for selected processor/board/kit/template or create configuration from existing SDK example project.

( ) Clone an example project

Select this option if you want example project with all sources for selected toolchain. The project will not be editable using MCUXpresso Config Tools.

( ) Start development without an MCUXpresso SDK Package

Use this option if you have not downloaded an SDK package yet. Tool will be limited to only Pins and Clocks Tools. It is possible to specify SDK path later.

Point to SDK installation folder (unzipped)

Once finished, select Continue

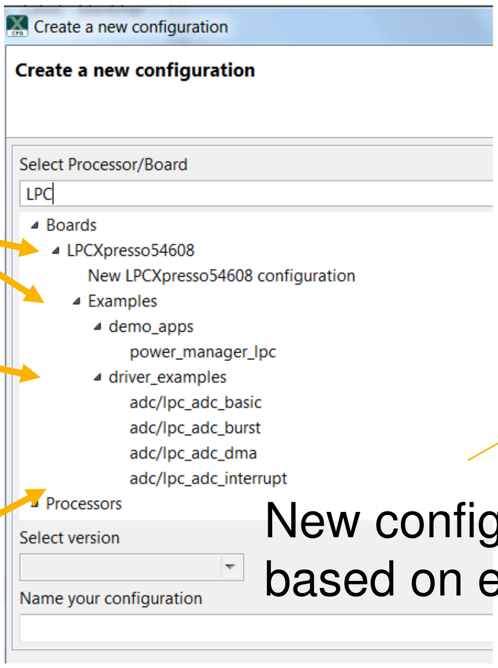
Creating a New Configuration using SDK

- New Board Configuration
  - Board-specific pin initialization (e.g. UART, LEDs, etc.)
  - Board-specific clock initialization (e.g. External OSC)

- Example Based Board Configuration
  - Pin initialization specific to board and example (e.g. I2C, SPI, etc.)
  - Clock initialization specific to board and examples

- Processor Based Board Configuration
  - Empty pin initialization
  - Supports reset clock configuration

Create a New Configuration



Create a new configuration

Create a new configuration

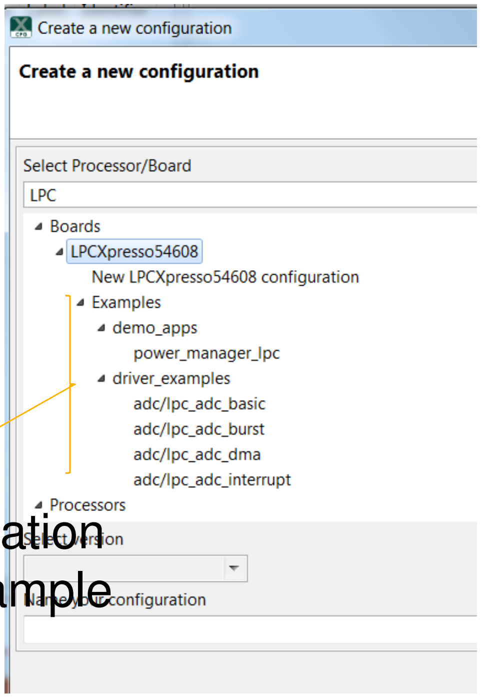
Select Processor/Board

LPC

- Boards
  - LPCXpresso54608
    - New LPCXpresso54608 configuration
    - Examples
      - demo_apps
        - power_manager_lpc
      - driver_examples
        - adc/lpc_adc_basic
        - adc/lpc_adc_burst
        - adc/lpc_adc_dma
        - adc/lpc_adc_interrupt
- Processors

Select version

Name your configuration



Create a new configuration

Create a new configuration

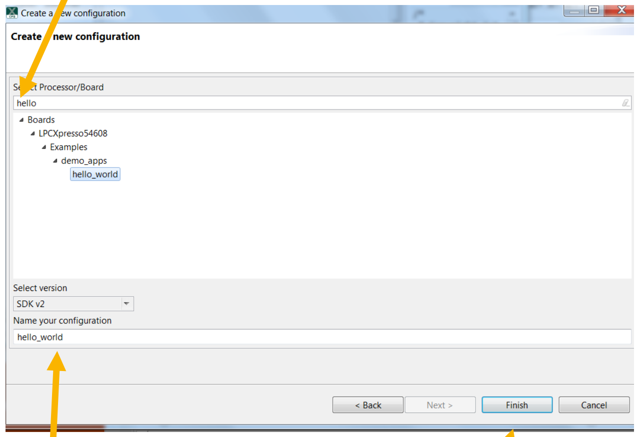
Select Processor/Board

LPC

- Boards
  - LPCXpresso54608
    - New LPCXpresso54608 configuration
    - Examples
      - demo_apps
        - power_manager_lpc
      - driver_examples
        - adc/lpc_adc_basic
        - adc/lpc_adc_burst
        - adc/lpc_adc_dma
        - adc/lpc_adc_interrupt
- Processors

Select version

Name your configuration



Create a new configuration

Create new configuration

Select Processor/Board

hello

- Boards
  - LPCXpresso54608
    - Examples
      - demo_apps
        - hello_world

Select version

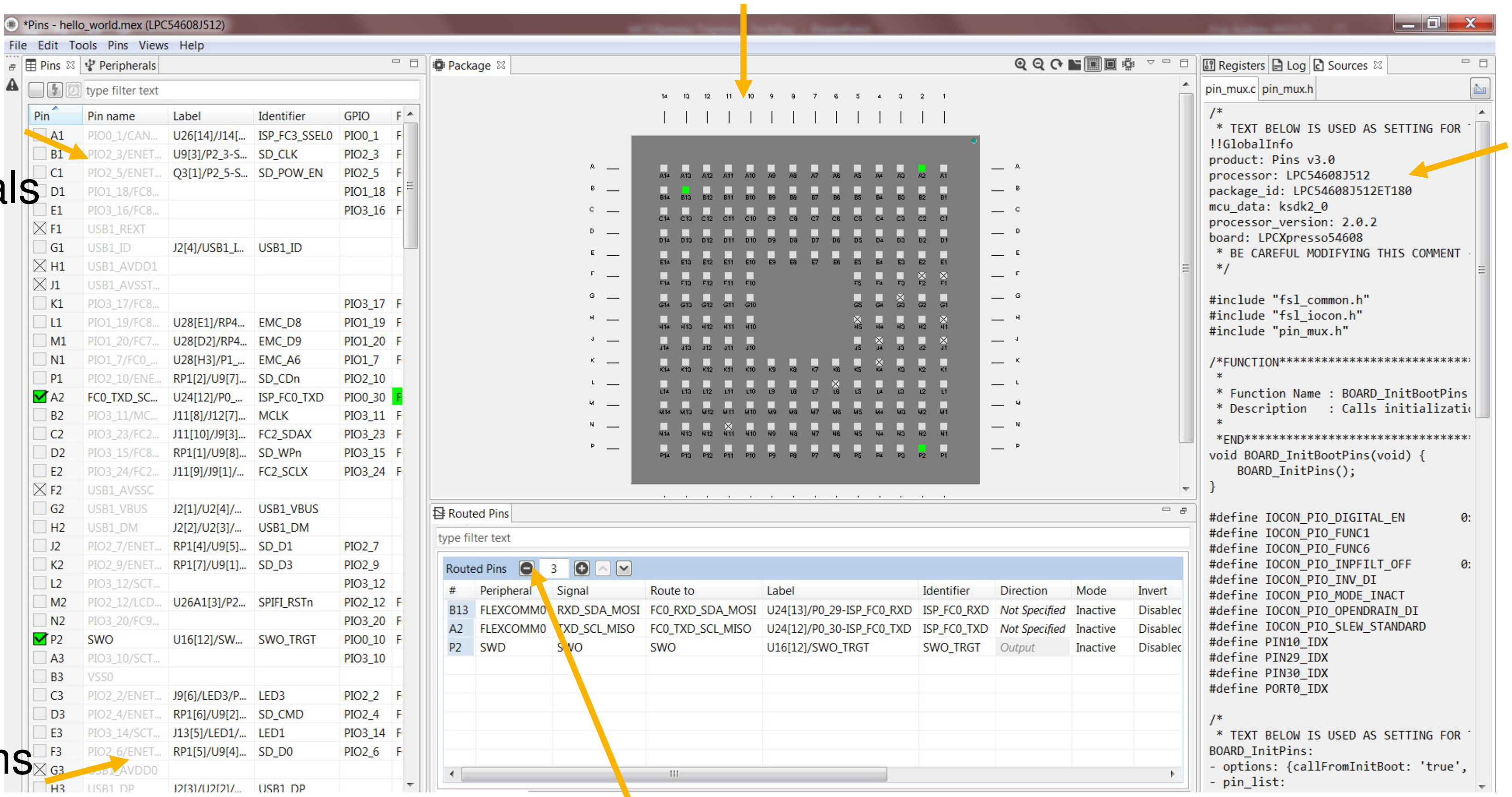
SDK v2

Name your configuration

hello_world

MCUXPRESSO CONFIG TOOLS PINS TOOL

Pins Tool Views



Pin	Pin name	Label	Identifier	GPIO	F
A1	PIO0_1/CAN...	U26[14]/J14[...	ISP_FC3_SSEL0	PIO0_1	F
B1	PIO2_3/ENET...	U9[3]/P2_3-S...	SD_CLK	PIO2_3	F
C1	PIO2_5/ENET...	Q3[1]/P2_5-S...	SD_POW_EN	PIO2_5	F
D1	PIO1_18/FC8...			PIO1_18	F
E1	PIO3_16/FC8...			PIO3_16	F
F1	USB1_REXT				
G1	USB1_ID	J2[4]/USB1_I...	USB1_ID		
H1	USB1_AVDD1				
J1	USB1_AVSST...				
K1	PIO3_17/FC8...			PIO3_17	F
L1	PIO1_19/FC8...	U28[E1]/RP4...	EMC_D8	PIO1_19	F
M1	PIO1_20/FC7...	U28[D2]/RP4...	EMC_D9	PIO1_20	F
N1	PIO1_7/FC0_...	U28[H3]/P1_...	EMC_A6	PIO1_7	F
P1	PIO2_10/ENE...	RP1[2]/U9[7]...	SD_CDn	PIO2_10	
✓ A2	FC0_TXD_SC...	U24[12]/P0_...	ISP_FC0_TXD	PIO0_30	F
B2	PIO3_11/MC...	J11[8]/J12[7]...	MCLK	PIO3_11	F
C2	PIO3_23/FC2...	J11[10]/J9[3]...	FC2_SDAX	PIO3_23	F
D2	PIO3_15/FC8...	RP1[1]/U9[8]...	SD_WPn	PIO3_15	F
E2	PIO3_24/FC2...	J11[9]/J9[1]/...	FC2_SCLX	PIO3_24	F
F2	USB1_AVSSC				
G2	USB1_VBUS	J2[1]/U2[4]/...	USB1_VBUS		
H2	USB1_DM	J2[2]/U2[3]/...	USB1_DM		
J2	PIO2_7/ENET...	RP1[4]/U9[5]...	SD_D1	PIO2_7	
K2	PIO2_9/ENET...	RP1[7]/U9[1]...	SD_D3	PIO2_9	
L2	PIO3_12/SCT...			PIO3_12	
M2	PIO2_12/LCD...	U26A1[3]/P2...	SPIFI_RSTn	PIO2_12	F
N2	PIO3_20/FC9...			PIO3_20	F
✓ P2	SWO	U16[12]/SW...	SWO_TRGT	PIO0_10	F
A3	PIO3_10/SCT...			PIO3_10	
B3	VSS0				
C3	PIO2_2/ENET...	J9[6]/LED3/P...	LED3	PIO2_2	F
D3	PIO2_4/ENET...	RP1[6]/U9[2]...	SD_CMD	PIO2_4	F
E3	PIO3_14/SCT...	J13[5]/LED1/...	LED1	PIO3_14	F
F3	PIO2_6/ENET...	RP1[5]/U9[4]...	SD_D0	PIO2_6	F
G3	USB1_AVDD0				
H3	USB1_DP	J2[3]/U2[2]/...	USB1_DP		

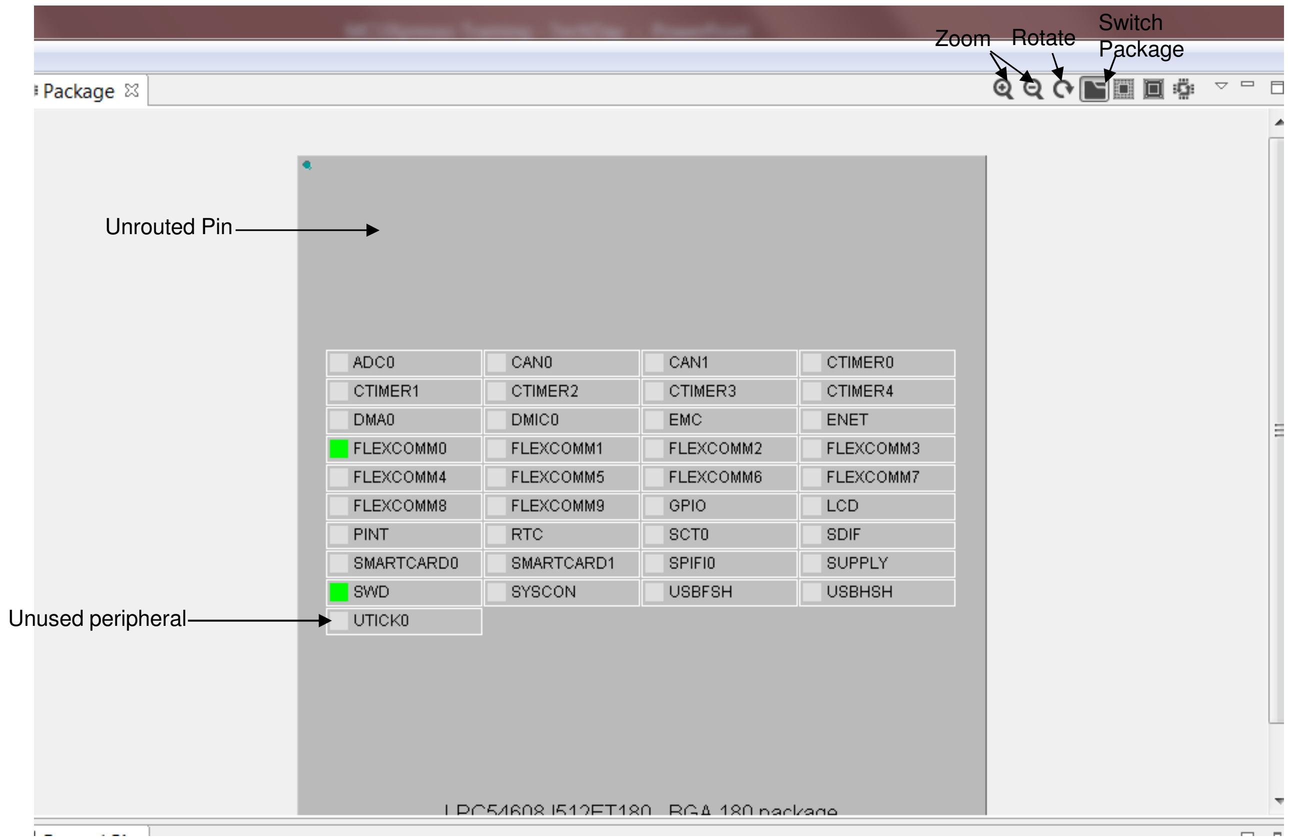
Routed Pins 3

#	Peripheral	Signal	Route to	Label	Identifier	Direction	Mode	Invert
B13	FLEXCOMM0	RXD_SDA_MOSI	FC0_RXD_SDA_MOSI	U24[13]/P0_29-ISP_FC0_RXD	ISP_FC0_RXD	*Not Specified*	Inactive	Disablec
A2	FLEXCOMM0	TXD_SCL_MISO	FC0_TXD_SCL_MISO	U24[12]/P0_30-ISP_FC0_TXD	ISP_FC0_TXD	*Not Specified*	Inactive	Disablec
P2	SWD	SWO	SWO	U16[12]/SWO_TRGT	SWO_TRGT	*Output*	Inactive	Disablec

```
/*
 * TEXT BELOW IS USED AS SETTING FOR
!!GlobalInfo
product: Pins v3.0
processor: LPC54608J512
package_id: LPC54608J512ET180
mcu_data: ksdk2_0
processor_version: 2.0.2
board: LPCXpresso54608
 * BE CAREFUL MODIFYING THIS COMMENT
 */

#include "fsl_common.h"
#include "fsl_iocon.h"
#include "pin_mux.h"

/*FUNCTION*****************************
 *
 * Function Name : BOARD_InitBootPins
 * Description   : Calls initializati
 *
 *END**********************************
void BOARD_InitBootPins(void) {
    BOARD_InitPins();
}

#define IOCON_PIO_DIGITAL_EN        0:
#define IOCON_PIO_FUNC1
#define IOCON_PIO_FUNC6
#define IOCON_PIO_INPFILT_OFF       0:
#define IOCON_PIO_INV_DI
#define IOCON_PIO_MODE_INACT
#define IOCON_PIO_OPENDRAIN_DI
#define IOCON_PIO_SLEW_STANDARD
#define PIN10_IDX
#define PIN29_IDX
#define PIN30_IDX
#define PORT0_IDX

/*
 * TEXT BELOW IS USED AS SETTING FOR
BOARD_InitPins:
- options: {callFromInitBoot: 'true',
- pin_list:
```

Package View

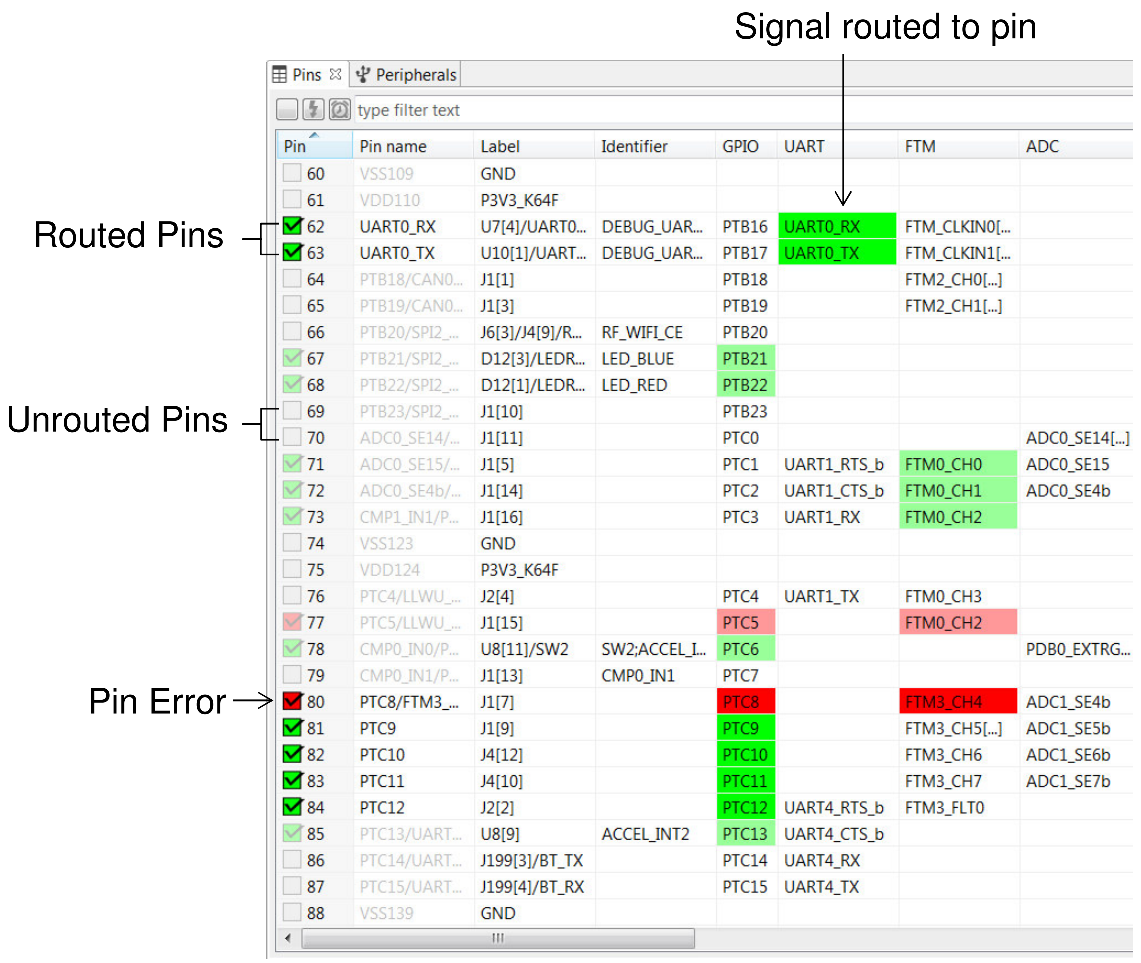
- Provides overview of package
- Available peripherals are indicated inside the package
- Pin names are listed next to each pin
- Color Coding:
  - Green indicates pin/peripheral is routed
  - Yellow indicates pin/peripheral is currently selected
  - Red indicates an error
  - Light grey indicates pin/peripheral is available but is not currently routed



⬜ ADC0	⬜ CAN0	⬜ CAN1	⬜ CTIMER0
⬜ CTIMER1	⬜ CTIMER2	⬜ CTIMER3	⬜ CTIMER4
⬜ DMA0	⬜ DMIC0	⬜ EMC	⬜ ENET
🟩 FLEXCOMM0	⬜ FLEXCOMM1	⬜ FLEXCOMM2	⬜ FLEXCOMM3
⬜ FLEXCOMM4	⬜ FLEXCOMM5	⬜ FLEXCOMM6	⬜ FLEXCOMM7
⬜ FLEXCOMM8	⬜ FLEXCOMM9	⬜ GPIO	⬜ LCD
⬜ PINT	⬜ RTC	⬜ SCT0	⬜ SDIF
⬜ SMARTCARD0	⬜ SMARTCARD1	⬜ SPIFI0	⬜ SUPPLY
🟩 SWD	⬜ SYSCON	⬜ USBFSH	⬜ USBHSH
⬜ UTICK0			

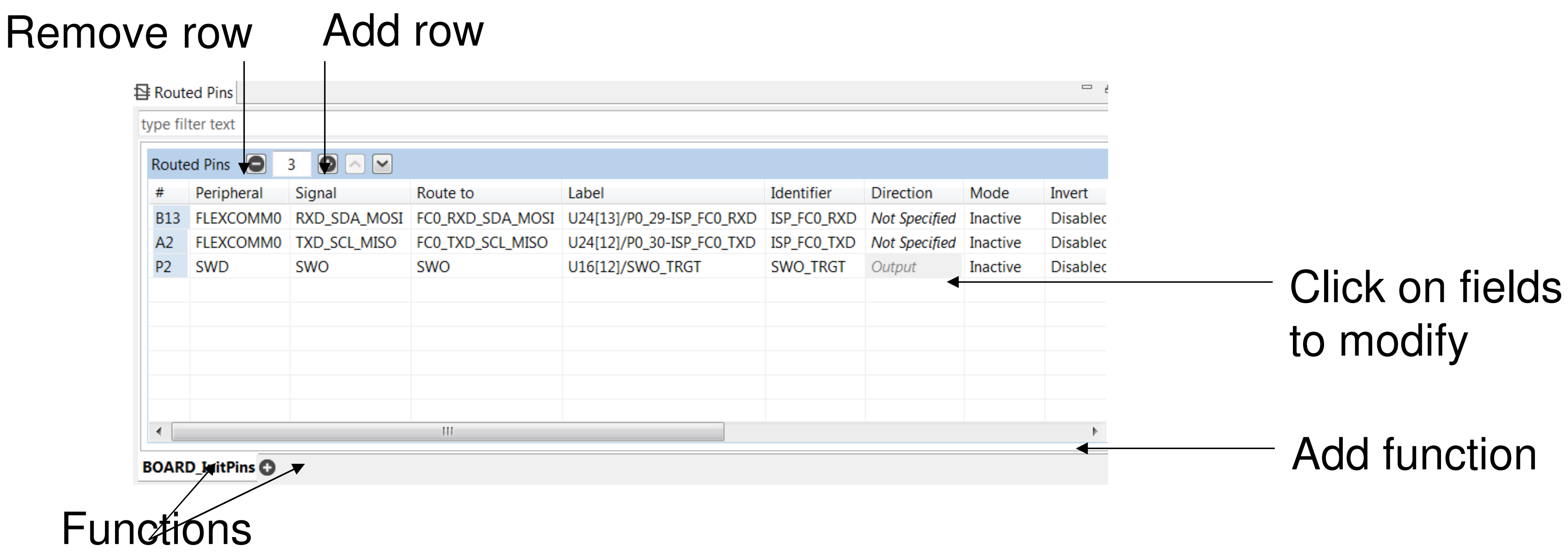
Pins View

- Provides a table of all the pins for the device / package
- Pins are listed by number and name
- Signals available for each pin are listed in columns across the pin's row
- The checkbox next to the pin name indicates whether the pin is routed



Pin	Pin name	Label	Identifier	GPIO	UART	FTM	ADC
☐ 60	VSS109	GND					
☐ 61	VDD110	P3V3_K64F					
✓ 62	UART0_RX	U7[4]/UART0...	DEBUG_UAR...	PTB16	UART0_RX	FTM_CLKIN0[...	
✓ 63	UART0_TX	U10[1]/UART...	DEBUG_UAR...	PTB17	UART0_TX	FTM_CLKIN1[...	
☐ 64	PTB18/CAN0...	J1[1]		PTB18		FTM2_CH0[...]	
☐ 65	PTB19/CAN0...	J1[3]		PTB19		FTM2_CH1[...]	
☐ 66	PTB20/SPI2_...	J6[3]/J4[9]/R...	RF_WIFI_CE	PTB20			
✓ 67	PTB21/SPI2_...	D12[3]/LEDR...	LED_BLUE	PTB21			
✓ 68	PTB22/SPI2_...	D12[1]/LEDR...	LED_RED	PTB22			
☐ 69	PTB23/SPI2_...	J1[10]		PTB23			
☐ 70	ADC0_SE14/...	J1[11]		PTC0			ADC0_SE14[...]
✓ 71	ADC0_SE15/...	J1[5]		PTC1	UART1_RTS_b	FTM0_CH0	ADC0_SE15
✓ 72	ADC0_SE4b/...	J1[14]		PTC2	UART1_CTS_b	FTM0_CH1	ADC0_SE4b
✓ 73	CMP1_IN1/P...	J1[16]		PTC3	UART1_RX	FTM0_CH2	
☐ 74	VSS123	GND					
☐ 75	VDD124	P3V3_K64F					
☐ 76	PTC4/LLWU_...	J2[4]		PTC4	UART1_TX	FTM0_CH3	
✓ 77	PTC5/LLWU_...	J1[15]		PTC5		FTM0_CH2	
✓ 78	CMP0_IN0/P...	U8[11]/SW2	SW2;ACCEL_I...	PTC6			PDB0_EXTRG...
☐ 79	CMP0_IN1/P...	J1[13]	CMP0_IN1	PTC7			
✓ 80	PTC8/FTM3_...	J1[7]		PTC8		FTM3_CH4	ADC1_SE4b
✓ 81	PTC9	J1[9]		PTC9		FTM3_CH5[...]	ADC1_SE5b
✓ 82	PTC10	J4[12]		PTC10		FTM3_CH6	ADC1_SE6b
✓ 83	PTC11	J4[10]		PTC11		FTM3_CH7	ADC1_SE7b
✓ 84	PTC12	J2[2]		PTC12	UART4_RTS_b	FTM3_FLT0	
✓ 85	PTC13/UART...	U8[9]	ACCEL_INT2	PTC13	UART4_CTS_b		
☐ 86	PTC14/UART...	J199[3]/BT_TX		PTC14	UART4_RX		
☐ 87	PTC15/UART...	J199[4]/BT_RX		PTC15	UART4_TX		
☐ 88	VSS139	GND					

Routed Pins View

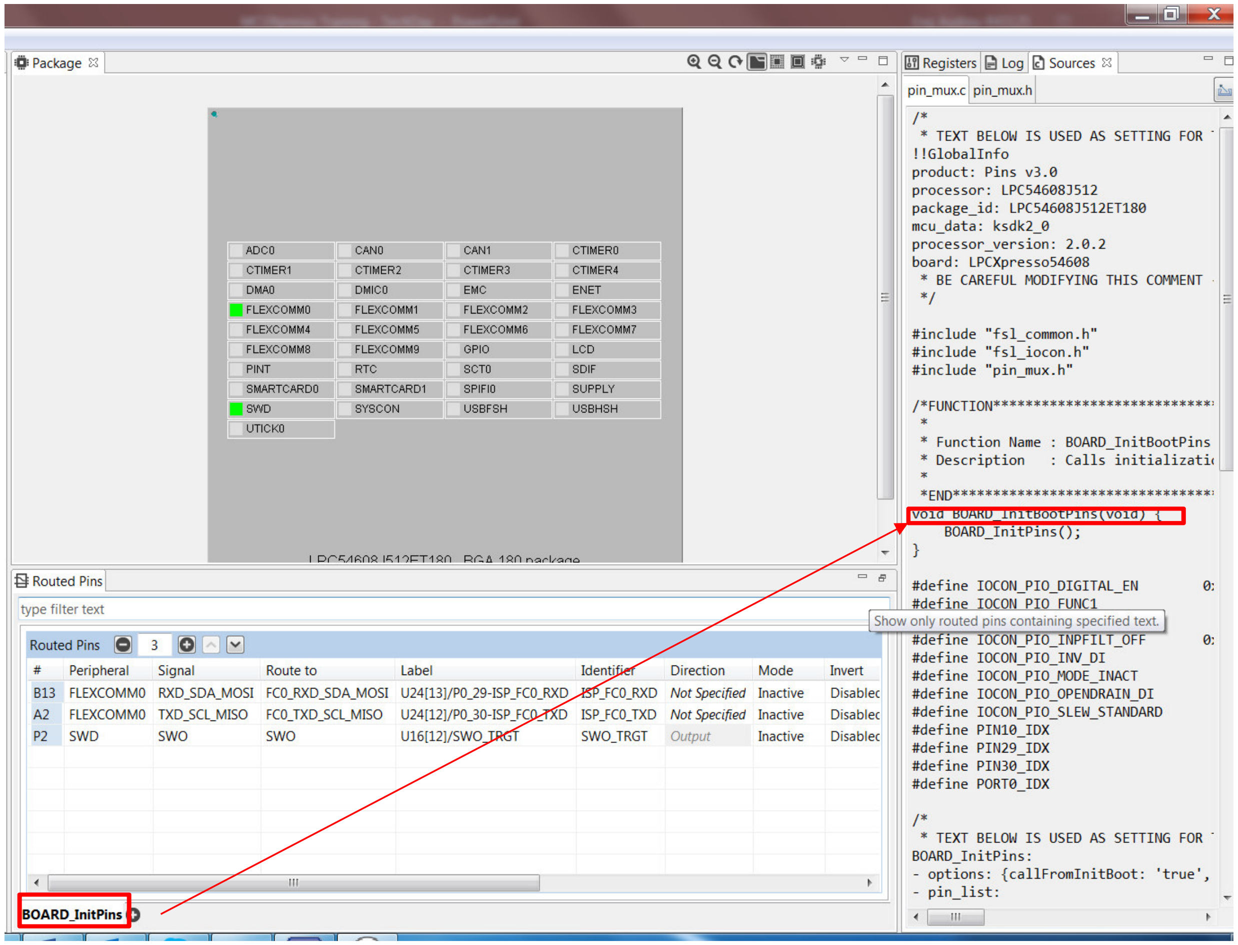


#	Peripheral	Signal	Route to	Label	Identifier	Direction	Mode	Invert
B13	FLEXCOMM0	RXD_SDA_MOSI	FC0_RXD_SDA_MOSI	U24[13]/P0_29-ISP_FC0_RXD	ISP_FC0_RXD	*Not Specified*	Inactive	Disablec
A2	FLEXCOMM0	TXD_SCL_MISO	FC0_TXD_SCL_MISO	U24[12]/P0_30-ISP_FC0_TXD	ISP_FC0_TXD	*Not Specified*	Inactive	Disablec
P2	SWD	SWO	SWO	U16[12]/SWO_TRGT	SWO_TRGT	*Output*	Inactive	Disablec

- View shows table of every routed pin
- 'Functions' are used to group a set of routed pins
- Functions create code which can be called by the application.

Functions

- Functions in Routed Pins View are used to generate functions in source code
- Pins that are routed in the Routed pins view will be initialized in the corresponding function in the source code



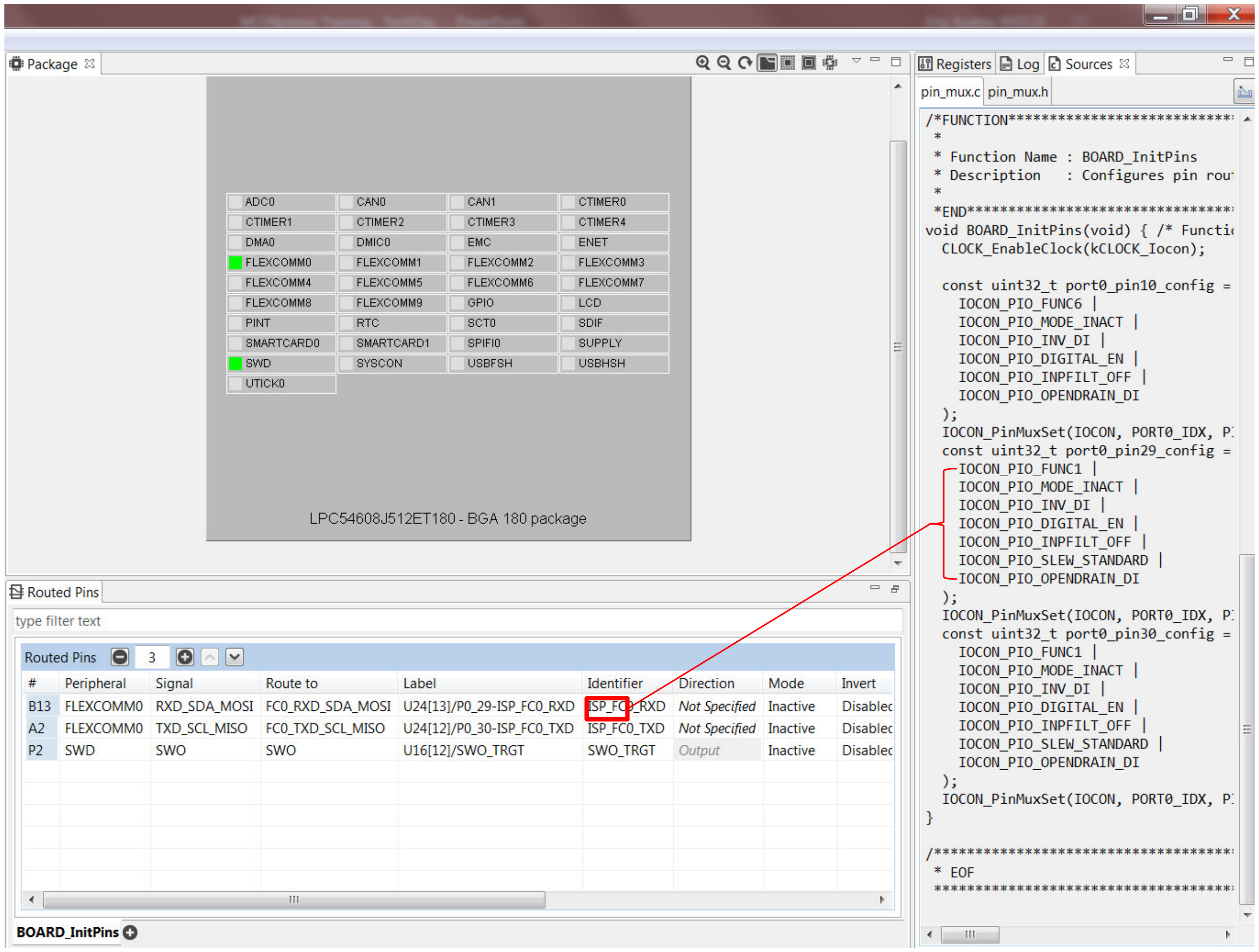
Routed Pins 3

#	Peripheral	Signal	Route to	Label	Identifier	Direction	Mode	Invert
B13	FLEXCOMM0	RXD_SDA_MOSI	FC0_RXD_SDA_MOSI	U24[13]/P0_29-ISP_FC0_RXD	ISP_FC0_RXD	*Not Specified*	Inactive	Disablec
A2	FLEXCOMM0	TXD_SCL_MISO	FC0_TXD_SCL_MISO	U24[12]/P0_30-ISP_FC0_TXD	ISP_FC0_TXD	*Not Specified*	Inactive	Disablec
P2	SWD	SWO	SWO	U16[12]/SWO_TRGT	SWO_TRGT	*Output*	Inactive	Disablec

```
/*
 * TEXT BELOW IS USED AS SETTING FOR
!!GlobalInfo
product: Pins v3.0
processor: LPC54608J512
package_id: LPC54608J512ET180
mcu_data: ksdk2_0
processor_version: 2.0.2
board: LPCXpresso54608
 * BE CAREFUL MODIFYING THIS COMMENT
 */

#include "fsl_common.h"
#include "fsl_iocon.h"
#include "pin_mux.h"

/*FUNCTION*****************************
 *
 * Function Name : BOARD_InitBootPins
 * Description   : Calls initializati
 *
 *END**********************************
void BOARD_InitBootPins(void) {
    BOARD_InitPins();
}

#define IOCON_PIO_DIGITAL_EN        0:
#define IOCON_PIO_FUNC1
#define IOCON_PIO_INPFILT_OFF       0:
#define IOCON_PIO_INV_DI
#define IOCON_PIO_MODE_INACT
#define IOCON_PIO_OPENDRAIN_DI
#define IOCON_PIO_SLEW_STANDARD
#define PIN10_IDX
#define PIN29_IDX
#define PIN30_IDX
#define PORT0_IDX

/*
 * TEXT BELOW IS USED AS SETTING FOR
BOARD_InitPins:
- options: {callFromInitBoot: 'true',
- pin_list:
```

Labels and Identifiers

- Board and kit configurations have predefined pin labels and identifiers
- Label
  - Can be defined for any pin for easy identification
- Identifier
  - Used to generate the #define in the pin_mux.h file
  - Can be modified in Pins View



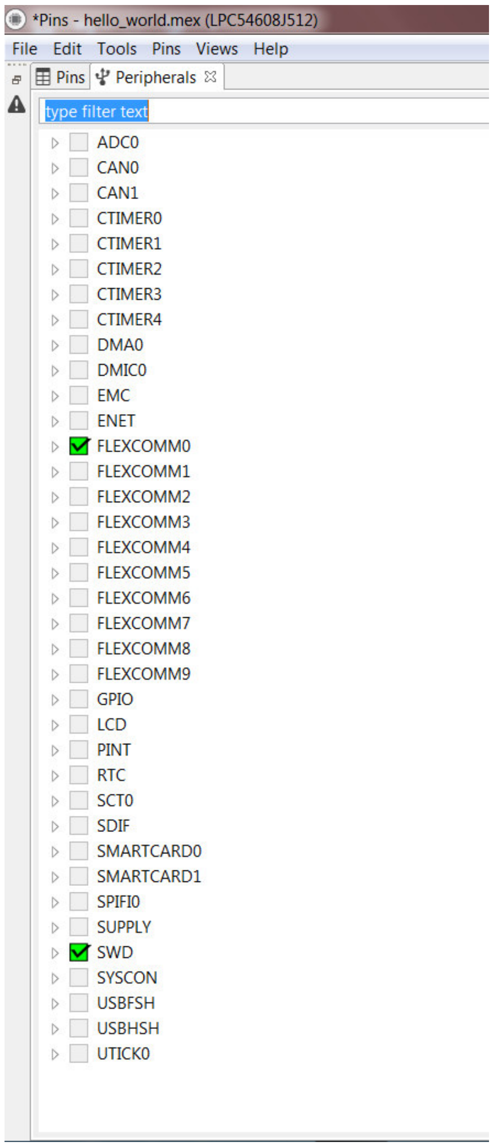
Routed Pins 3

#	Peripheral	Signal	Route to	Label	Identifier	Direction	Mode	Invert
B13	FLEXCOMM0	RXD_SDA_MOSI	FC0_RXD_SDA_MOSI	U24[13]/P0_29-ISP_FC0_RXD	ISP_FC0_RXD	*Not Specified*	Inactive	Disablec
A2	FLEXCOMM0	TXD_SCL_MISO	FC0_TXD_SCL_MISO	U24[12]/P0_30-ISP_FC0_TXD	ISP_FC0_TXD	*Not Specified*	Inactive	Disablec
P2	SWD	SWO	SWO	U16[12]/SWO_TRGT	SWO_TRGT	*Output*	Inactive	Disablec

```
/*FUNCTION***************************
 *
 * Function Name : BOARD_InitPins
 * Description   : Configures pin rou
 *
 *END*********************************
void BOARD_InitPins(void) { /* Functi
  CLOCK_EnableClock(kCLOCK_Iocon);

  const uint32_t port0_pin10_config =
    IOCON_PIO_FUNC6 |
    IOCON_PIO_MODE_INACT |
    IOCON_PIO_INV_DI |
    IOCON_PIO_DIGITAL_EN |
    IOCON_PIO_INPFILT_OFF |
    IOCON_PIO_OPENDRAIN_DI
  );
  IOCON_PinMuxSet(IOCON, PORT0_IDX, P
  const uint32_t port0_pin29_config =
    IOCON_PIO_FUNC1 |
    IOCON_PIO_MODE_INACT |
    IOCON_PIO_INV_DI |
    IOCON_PIO_DIGITAL_EN |
    IOCON_PIO_INPFILT_OFF |
    IOCON_PIO_SLEW_STANDARD |
    IOCON_PIO_OPENDRAIN_DI
  );
  IOCON_PinMuxSet(IOCON, PORT0_IDX, P
  const uint32_t port0_pin30_config =
    IOCON_PIO_FUNC1 |
    IOCON_PIO_MODE_INACT |
    IOCON_PIO_INV_DI |
    IOCON_PIO_DIGITAL_EN |
    IOCON_PIO_INPFILT_OFF |
    IOCON_PIO_SLEW_STANDARD |
    IOCON_PIO_OPENDRAIN_DI
  );
  IOCON_PinMuxSet(IOCON, PORT0_IDX, P
}

/************************************
 * EOF
 ************************************
```

Peripherals View

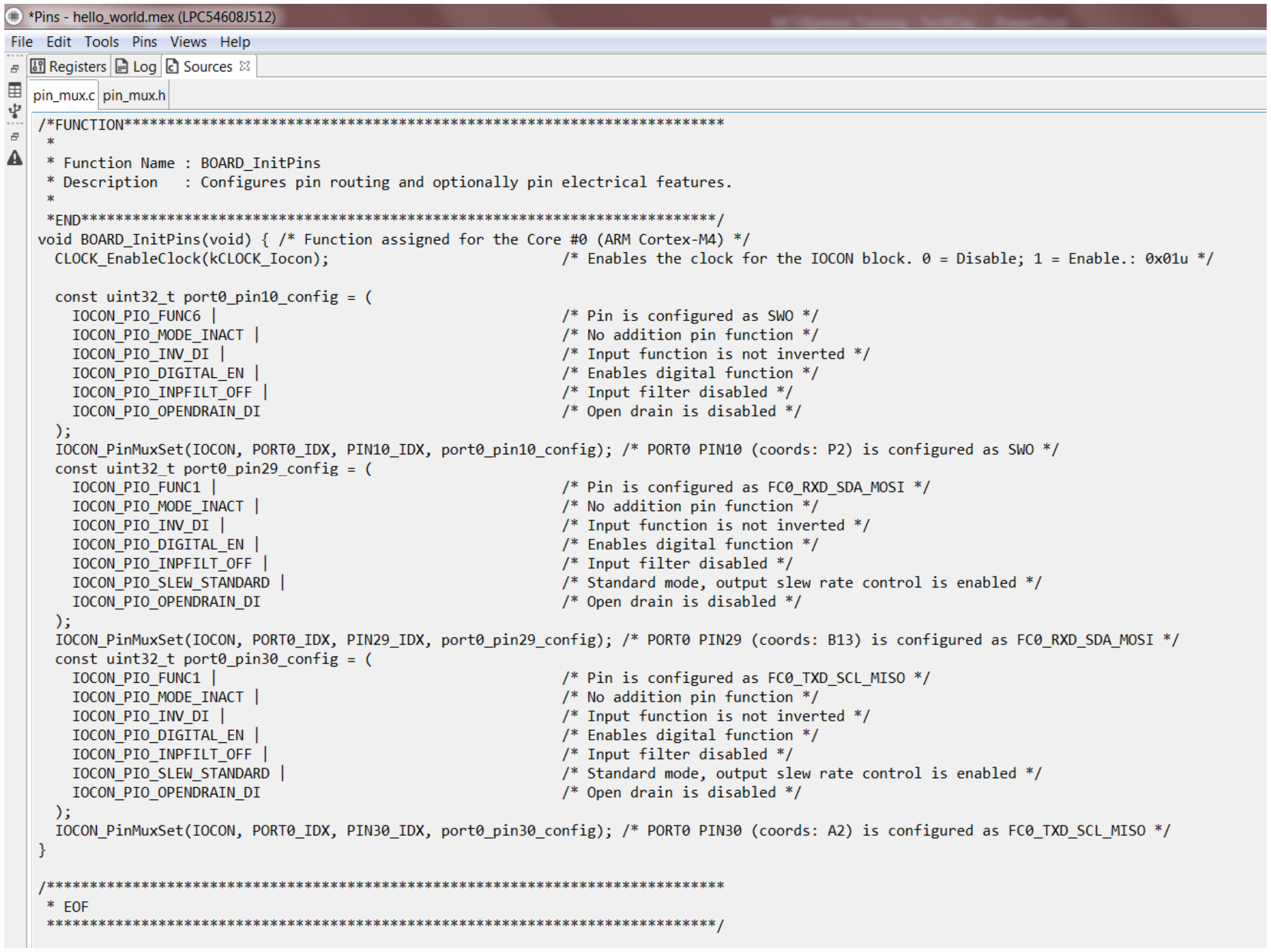
- Displays list of all peripherals of device
- Expand peripheral to see available signals
- Peripherals with marks in checkbox have signals routed


*Pins - hello_world.mex (LPC54608J512)

- [ ] ADC0
- [ ] CAN0
- [ ] CAN1
- [ ] CTIMER0
- [ ] CTIMER1
- [ ] CTIMER2
- [ ] CTIMER3
- [ ] CTIMER4
- [ ] DMA0
- [ ] DMIC0
- [ ] EMC
- [ ] ENET
- [✓] FLEXCOMM0
- [ ] FLEXCOMM1
- [ ] FLEXCOMM2
- [ ] FLEXCOMM3
- [ ] FLEXCOMM4
- [ ] FLEXCOMM5
- [ ] FLEXCOMM6
- [ ] FLEXCOMM7
- [ ] FLEXCOMM8
- [ ] FLEXCOMM9
- [ ] GPIO
- [ ] LCD
- [ ] PINT
- [ ] RTC
- [ ] SCT0
- [ ] SDIF
- [ ] SMARTCARD0
- [ ] SMARTCARD1
- [ ] SPIFI0
- [ ] SUPPLY
- [✓] SWD
- [ ] SYSCON
- [ ] USBFSH
- [ ] USBHSH
- [ ] UTICK0

Sources View

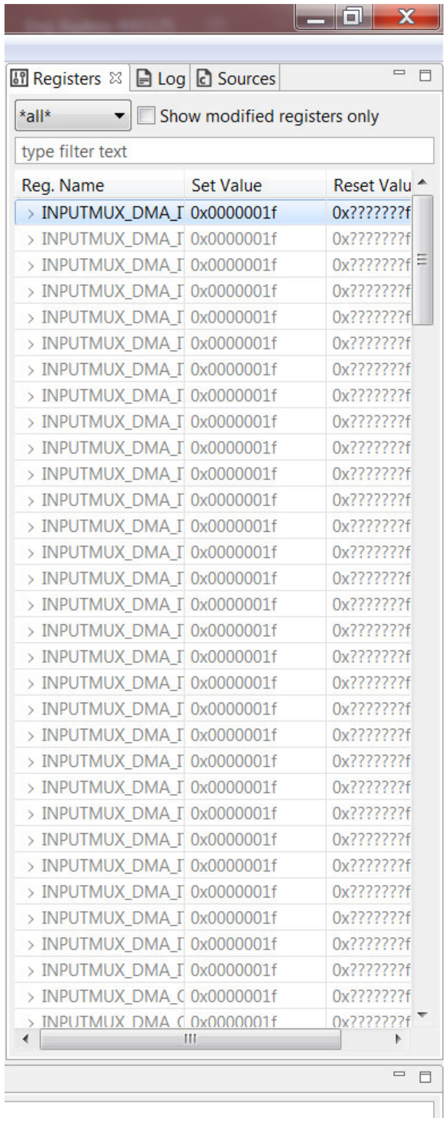
- The pins tool modifies and creates code in pin_mux.c and pin_mux.h files
- The pin configuration info is stored in YAML format
- YAML code is not intended for user modification


*Pins - hello_world.mex (LPC54608J512)
File Edit Tools Pins Views Help
Registers Log Sources
pin_mux.c pin_mux.h

```
/*FUNCTION**********************************************************************
 *
 * Function Name : BOARD_InitPins
 * Description   : Configures pin routing and optionally pin electrical features.
 *
 *END**************************************************************************/
void BOARD_InitPins(void) { /* Function assigned for the Core #0 (ARM Cortex-M4) */
  CLOCK_EnableClock(kCLOCK_Iocon);                           /* Enables the clock for the IOCON block. 0 = Disable; 1 = Enable.: 0x01u */

  const uint32_t port0_pin10_config = (
    IOCON_PIO_FUNC6 |                                        /* Pin is configured as SWO */
    IOCON_PIO_MODE_INACT |                                   /* No addition pin function */
    IOCON_PIO_INV_DI |                                       /* Input function is not inverted */
    IOCON_PIO_DIGITAL_EN |                                   /* Enables digital function */
    IOCON_PIO_INPFILT_OFF |                                  /* Input filter disabled */
    IOCON_PIO_OPENDRAIN_DI                                   /* Open drain is disabled */
  );
  IOCON_PinMuxSet(IOCON, PORT0_IDX, PIN10_IDX, port0_pin10_config); /* PORT0 PIN10 (coords: P2) is configured as SWO */
  const uint32_t port0_pin29_config = (
    IOCON_PIO_FUNC1 |                                        /* Pin is configured as FC0_RXD_SDA_MOSI */
    IOCON_PIO_MODE_INACT |                                   /* No addition pin function */
    IOCON_PIO_INV_DI |                                       /* Input function is not inverted */
    IOCON_PIO_DIGITAL_EN |                                   /* Enables digital function */
    IOCON_PIO_INPFILT_OFF |                                  /* Input filter disabled */
    IOCON_PIO_SLEW_STANDARD |                                /* Standard mode, output slew rate control is enabled */
    IOCON_PIO_OPENDRAIN_DI                                   /* Open drain is disabled */
  );
  IOCON_PinMuxSet(IOCON, PORT0_IDX, PIN29_IDX, port0_pin29_config); /* PORT0 PIN29 (coords: B13) is configured as FC0_RXD_SDA_MOSI */
  const uint32_t port0_pin30_config = (
    IOCON_PIO_FUNC1 |                                        /* Pin is configured as FC0_TXD_SCL_MISO */
    IOCON_PIO_MODE_INACT |                                   /* No addition pin function */
    IOCON_PIO_INV_DI |                                       /* Input function is not inverted */
    IOCON_PIO_DIGITAL_EN |                                   /* Enables digital function */
    IOCON_PIO_INPFILT_OFF |                                  /* Input filter disabled */
    IOCON_PIO_SLEW_STANDARD |                                /* Standard mode, output slew rate control is enabled */
    IOCON_PIO_OPENDRAIN_DI                                   /* Open drain is disabled */
  );
  IOCON_PinMuxSet(IOCON, PORT0_IDX, PIN30_IDX, port0_pin30_config); /* PORT0 PIN30 (coords: A2) is configured as FC0_TXD_SCL_MISO */
}

/*******************************************************************************
 * EOF
 ******************************************************************************/
```

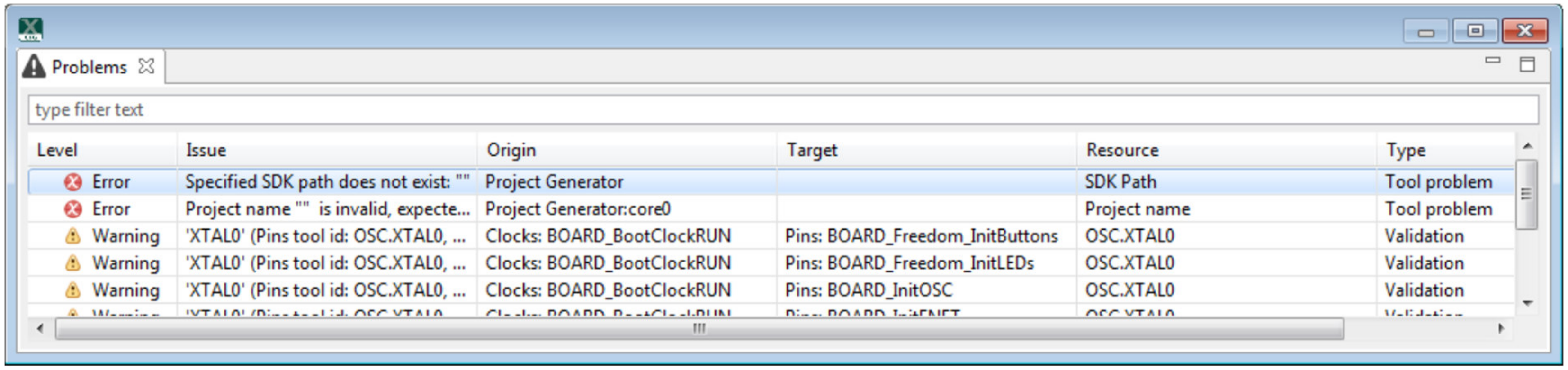
Register View

- Shows Register name, current value, and value at Reset
- Recently updated fields are highlighted in yellow



Reg. Name	Set Value	Reset Valu
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_I	0x0000001f	0x???????f
INPUTMUX_DMA_(	0x0000001f	0x???????f
INPUTMUX_DMA_(	0x0000001f	0x???????f

Problems View


Problems

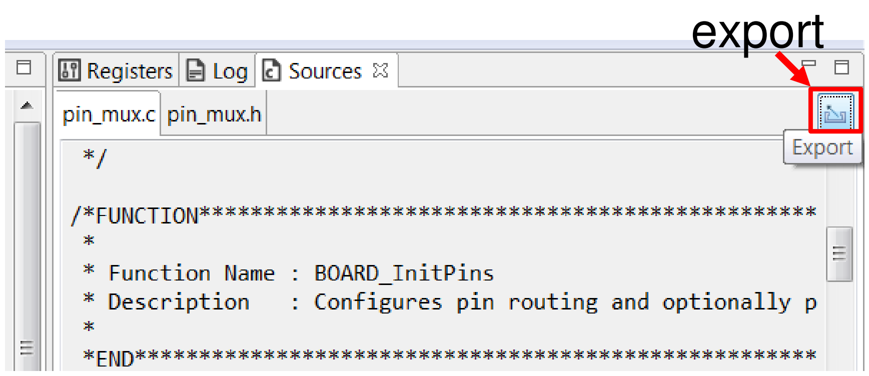
type filter text

Level	Issue	Origin	Target	Resource	Type
Error	Specified SDK path does not exist: ""	Project Generator		SDK Path	Tool problem
Error	Project name "" is invalid, expecte...	Project Generator:core0		Project name	Tool problem
Warning	'XTAL0' (Pins tool id: OSC.XTAL0, ...	Clocks: BOARD_BootClockRUN	Pins: BOARD_Freedom_InitButtons	OSC.XTAL0	Validation
Warning	'XTAL0' (Pins tool id: OSC.XTAL0, ...	Clocks: BOARD_BootClockRUN	Pins: BOARD_Freedom_InitLEDs	OSC.XTAL0	Validation
Warning	'XTAL0' (Pins tool id: OSC.XTAL0, ...	Clocks: BOARD_BootClockRUN	Pins: BOARD_InitOSC	OSC.XTAL0	Validation
Warning	'XTAL0' (Pins tool id: OSC.XTAL0	Clocks: BOARD_BootClockRUN	Pins: BOARD_InitENET	OSC.XTAL0	Validation

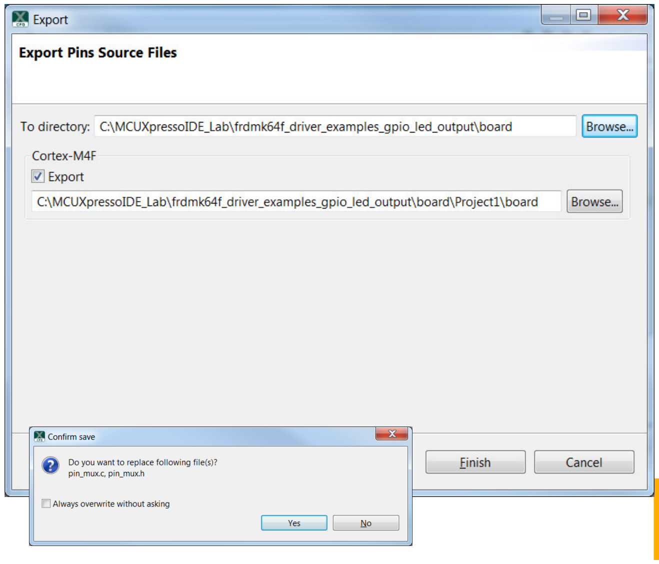
- **Level** – Lists the severity of the problem: Information, Warning, or Error.
- **Issue** – Description of the problem.
- **Origin** – Information on the dependency source.
- **Target** – Lists the tool that handled the dependency and where it should be fulfilled.
- **Resource** – Lists the resource which is related to the problem,. For example, the signal name, the clock signal, and so on.
- **Type** – The type of the problem. It is either the validation that is

Export Source Files

- Export a configuration to a pin_mux.c and pin_mux.h file
- Includes functions with initialization for routed pins in configurations
- Accessed from menu:
  - File->Export
  - Can also access by selecting the export icon in the Sources view
- Select directory to export pin_mux.c and pin_mux.h



```
 */

/*FUNCTION**********************************************************
 *
 * Function Name : BOARD_InitPins
 * Description   : Configures pin routing and optionally p
 *
 *END**************************************************************
```



Export Pins Source Files

To directory: C:\MCUXpressoIDE_Lab\frdmk64f_driver_examples_gpio_led_output\board

Cortex-M4F

☑ Export

C:\MCUXpressoIDE_Lab\frdmk64f_driver_examples_gpio_led_output\board\Project1\board

Do you want to replace following file(s)?
pin_mux.c, pin_mux.h

☐ Always overwrite without asking

MCUXPRESSO CONFIG TOOLS CLOCK TOOL

Clocks Tool Views

Clocks Table

Clocks Diagram

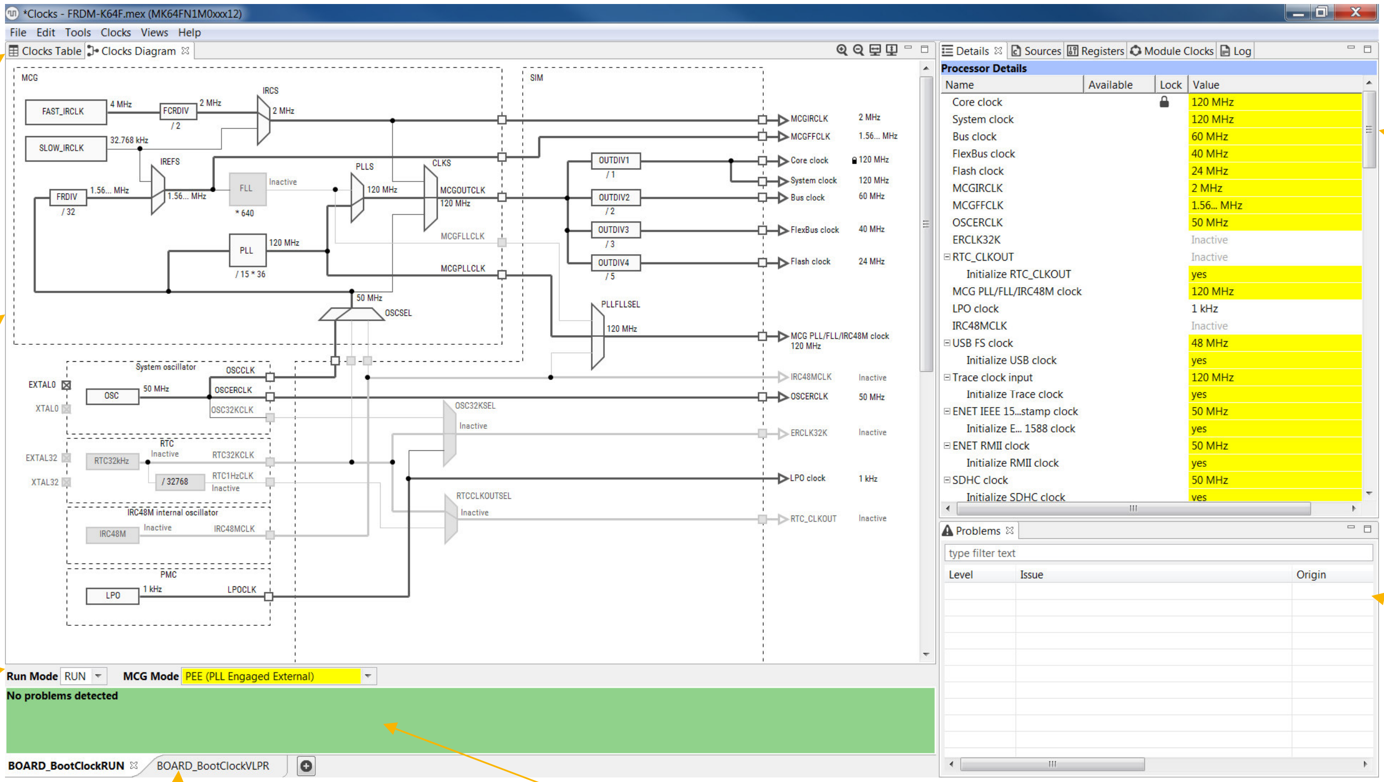
Global Settings

Clock Configuration tabs

Status bar

Details View Selected

Problems View



Run Mode RUN   MCG Mode PEE (PLL Engaged External)

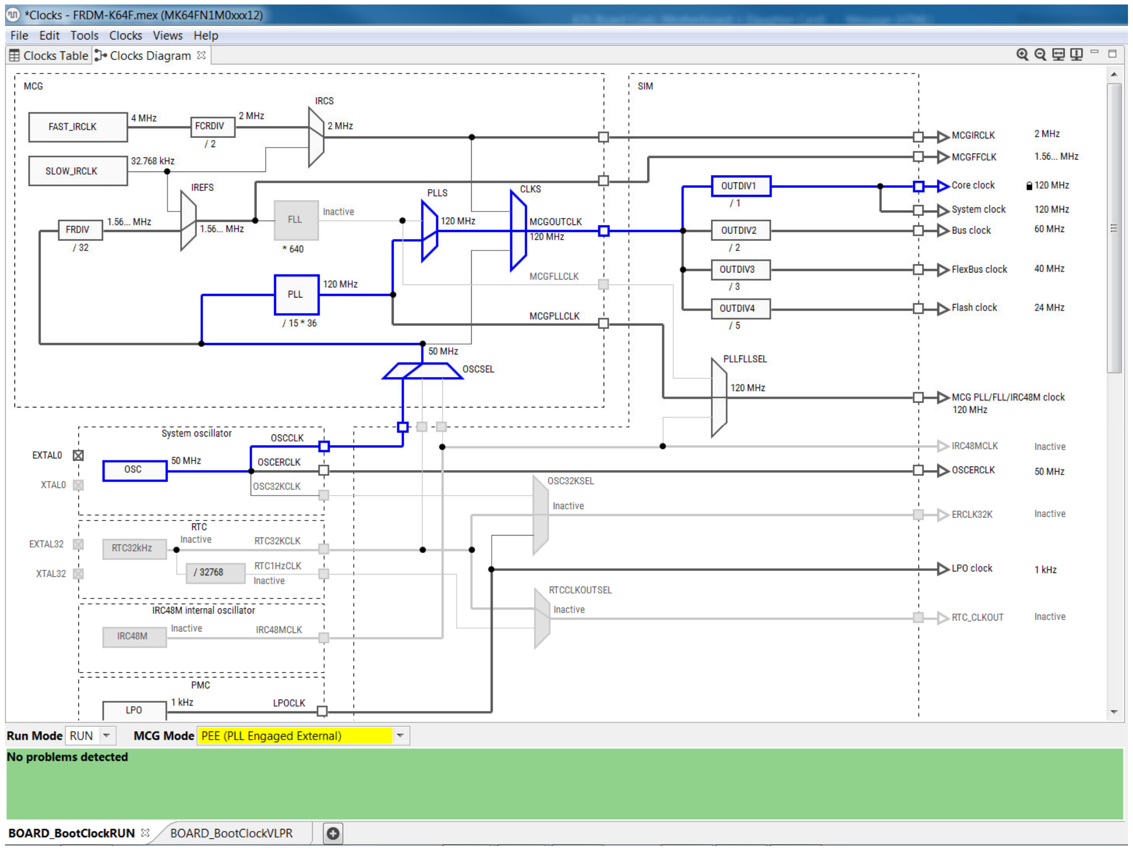
No problems detected

Processor Details

Name	Available	Lock	Value
Core clock		🔒	120 MHz
System clock			120 MHz
Bus clock			60 MHz
FlexBus clock			40 MHz
Flash clock			24 MHz
MCGIRCLK			2 MHz
MCGFFCLK			1.56... MHz
OSCERCLK			50 MHz
ERCLK32K			Inactive
⊟ RTC_CLKOUT			Inactive
Initialize RTC_CLKOUT			yes
MCG PLL/FLL/IRC48M clock			120 MHz
LPO clock			1 kHz
IRC48MCLK			Inactive
⊟ USB FS clock			48 MHz
Initialize USB clock			yes
⊟ Trace clock input			120 MHz
Initialize Trace clock			yes
⊟ ENET IEEE 15...stamp clock			50 MHz
Initialize E... 1588 clock			yes
⊟ ENET RMII clock			50 MHz
Initialize RMII clock			yes
⊟ SDHC clock			50 MHz
Initialize SDHC clock			yes

Clocks Diagram View

- Provides a block diagram of the clock generation for the device
- Solid lines indicate connections between elements
  - Dark lines indicate currently active clock paths
  - Gray lines indicate inactive clock paths
- Clock settings can be edited within the diagram
- When an active clock output is selected (e.g. Core clock), the clock path highlights in blue



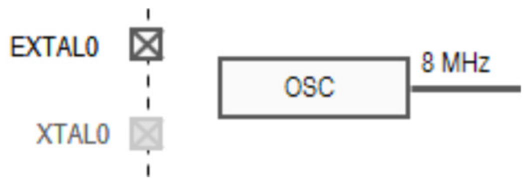
Run Mode RUN **MCG Mode** PEE (PLL Engaged External)

No problems detected

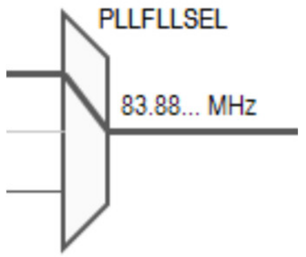
BOARD_BootClockRUN BOARD_BootClockVLPR

Elements of Clock Diagram

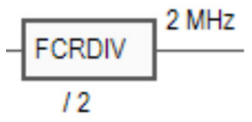
- **Clock Source**
  - Provides a clock frequency



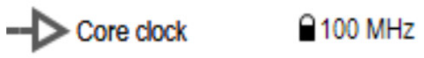
- **Multiplexer**
  - Selects between clock options



- **Prescaler**
  - Divides clock frequency

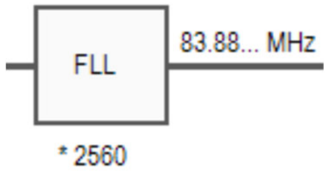


- **Clock Output**
  - Marks the clock signal output

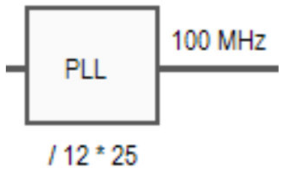


Elements of Clock Diagram (cont)

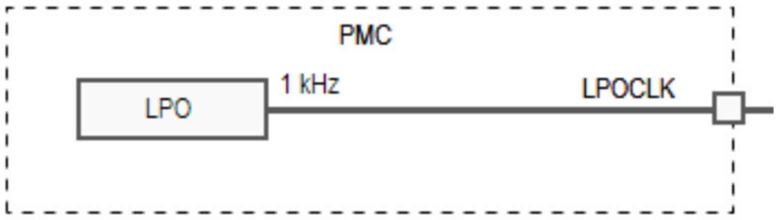
- **FLL (Frequency Locked Loop)**
  - Multiplies an incoming frequency by a given factor



- **PLL (Phase Locked Loop)**
  - Contains pre-divider and thus is able to divide/multiply with a given value.

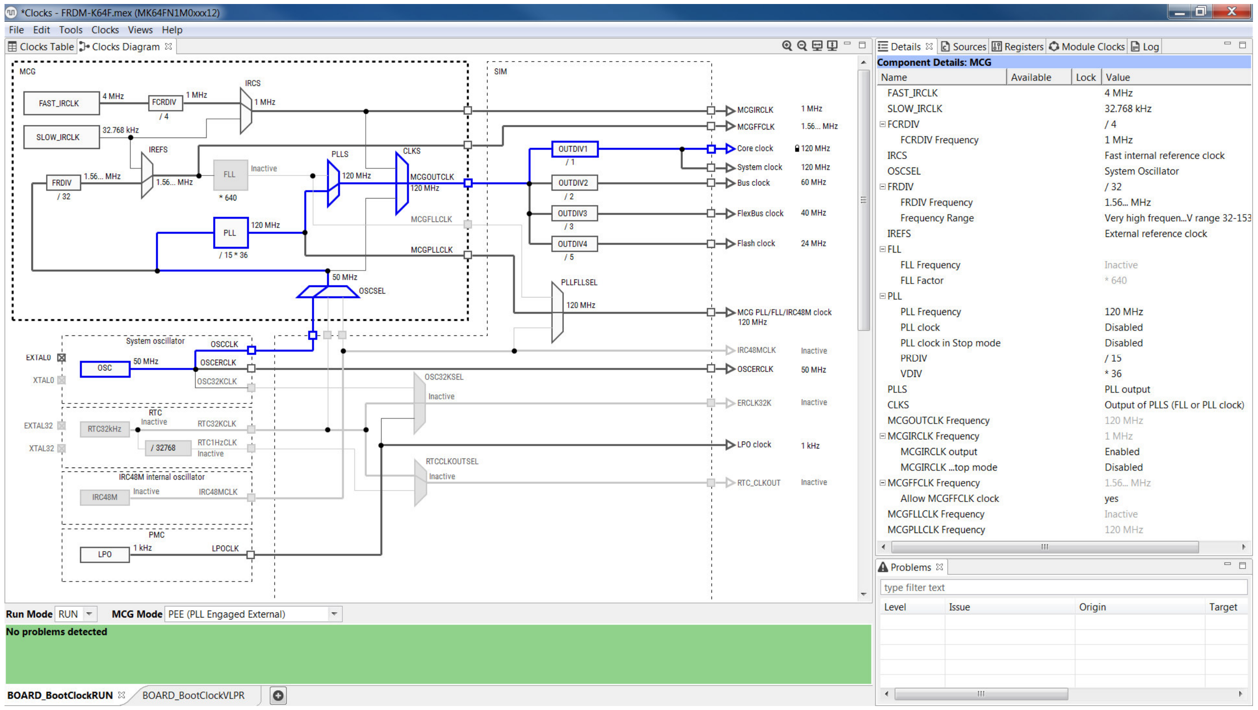


- **Clock Component**
  - Group of clock elements surrounded with a border. The clock component usually corresponds to the processor modules or peripherals.



Details View: Overview

- The details for any element in the block diagram are displayed in the Details view when selected in the diagram
- The MCG component is selected in the figure to the right. The details for clocks, mux settings, and dividers in the MCG are shown in the details view



Component Details: MCG

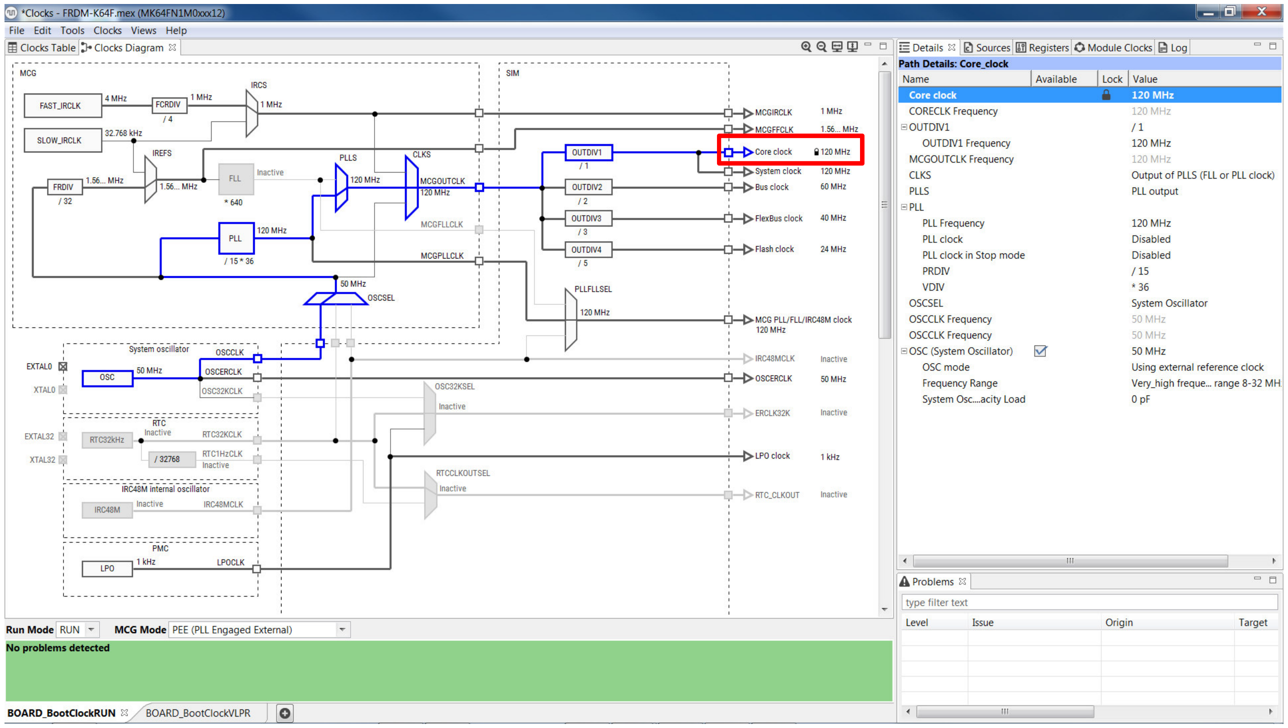
Name	Available	Lock	Value
FAST_IRCLK			4 MHz
SLOW_IRCLK			32.768 kHz
FCRDIV			/ 4
FCRDIV Frequency			1 MHz
IRCS			Fast internal reference clock
OSCSEL			System Oscillator
FRDIV			/ 32
FRDIV Frequency			1.56... MHz
Frequency Range			Very high frequen...V range 32-153
IREFS			External reference clock
FLL			
FLL Frequency			Inactive
FLL Factor			* 640
PLL			
PLL Frequency			120 MHz
PLL clock			Disabled
PLL clock in Stop mode			Disabled
PRDIV			/ 15
VDIV			* 36
PLLS			PLL output
CLKS			Output of PLLS (FLL or PLL clock)
MCGOUTCLK Frequency			120 MHz
MCGIRCLK Frequency			1 MHz
MCGIRCLK output			Enabled
MCGIRCLK ...top mode			Disabled
MCGFFCLK Frequency			1.56... MHz
Allow MCGFFCLK clock			yes
MCGFLLCLK Frequency			Inactive
MCGPLLCLK Frequency			120 MHz

type filter text

Level	Issue	Origin	Target

No problems detected

Details View: Clock Path Details



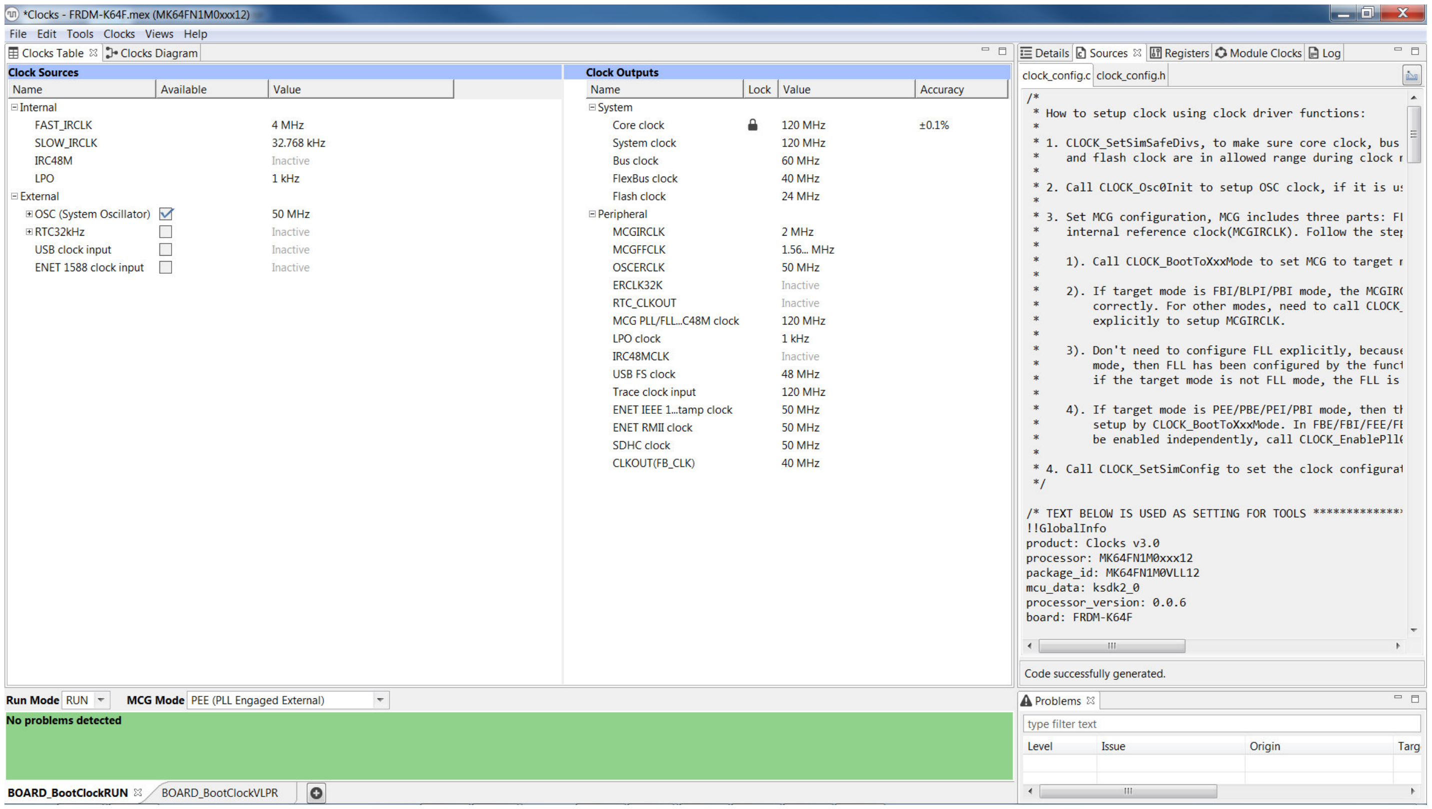
Path Details: Core_clock

Name	Available	Lock	Value
Core clock		🔒	120 MHz
CORECLK Frequency			120 MHz
⊟ OUTDIV1			/ 1
OUTDIV1 Frequency			120 MHz
MCGOUTCLK Frequency			120 MHz
CLKS			Output of PLLS (FLL or PLL clock)
PLLS			PLL output
⊟ PLL			
PLL Frequency			120 MHz
PLL clock			Disabled
PLL clock in Stop mode			Disabled
PRDIV			/ 15
VDIV			* 36
OSCSEL			System Oscillator
OSCCLK Frequency			50 MHz
OSCCLK Frequency			50 MHz
⊟ OSC (System Oscillator)	✓		50 MHz
OSC mode			Using external reference clock
Frequency Range			Very_high freque... range 8-32 MH
System Osc....acity Load			0 pF

- Clock path details are shown in the Details View when a clock output is selected (in Clock Diagram or Clock Table)
- The Details view shows information for each element in the clock path (e.g. OSC, PLL, mux selections, divider settings)

Clocks Table View

- Provides overview of the clocking system and its current state
- Available clock sources are shown in the left panel
- Clock outputs and their current frequencies are shown in the right panel



Clock Sources

Name	Available	Value
Internal		
FAST_IRCLK		4 MHz
SLOW_IRCLK		32.768 kHz
IRC48M		Inactive
LPO		1 kHz
External		
OSC (System Oscillator)	✓	50 MHz
RTC32kHz	☐	Inactive
USB clock input	☐	Inactive
ENET 1588 clock input	☐	Inactive

Clock Outputs

Name	Lock	Value	Accuracy
System			
Core clock	🔒	120 MHz	±0.1%
System clock		120 MHz	
Bus clock		60 MHz	
FlexBus clock		40 MHz	
Flash clock		24 MHz	
Peripheral			
MCGIRCLK		2 MHz	
MCGFFCLK		1.56... MHz	
OSCERCLK		50 MHz	
ERCLK32K		Inactive	
RTC_CLKOUT		Inactive	
MCG PLL/FLL...C48M clock		120 MHz	
LPO clock		1 kHz	
IRC48MCLK		Inactive	
USB FS clock		48 MHz	
Trace clock input		120 MHz	
ENET IEEE 1...tamp clock		50 MHz	
ENET RMII clock		50 MHz	
SDHC clock		50 MHz	
CLKOUT(FB_CLK)		40 MHz	

```
/*
 * How to setup clock using clock driver functions:
 *
 * 1. CLOCK_SetSimSafeDivs, to make sure core clock, bus
 *    and flash clock are in allowed range during clock r
 *
 * 2. Call CLOCK_Osc0Init to setup OSC clock, if it is u:
 *
 * 3. Set MCG configuration, MCG includes three parts: Fl
 *    internal reference clock(MCGIRCLK). Follow the ste
 *
 *    1). Call CLOCK_BootToXxxMode to set MCG to target r
 *
 *    2). If target mode is FBI/BLPI/PBI mode, the MCGIR(
 *        correctly. For other modes, need to call CLOCK_
 *        explicitly to setup MCGIRCLK.
 *
 *    3). Don't need to configure FLL explicitly, becaus
 *        mode, then FLL has been configured by the funct
 *        if the target mode is not FLL mode, the FLL is
 *
 *    4). If target mode is PEE/PBE/PEI/PBI mode, then tl
 *        setup by CLOCK_BootToXxxMode. In FBE/FBI/FEE/FE
 *        be enabled independently, call CLOCK_EnablePll(
 *
 * 4. Call CLOCK_SetSimConfig to set the clock configurat
 */

/* TEXT BELOW IS USED AS SETTING FOR TOOLS *************
!!GlobalInfo
product: Clocks v3.0
processor: MK64FN1M0xxx12
package_id: MK64FN1M0VLL12
mcu_data: ksdk2_0
processor_version: 0.0.6
board: FRDM-K64F
```

Code successfully generated.

Run Mode RUN **MCG Mode** PEE (PLL Engaged External)

No problems detected

BOARD_BootClockRUN BOARD_BootClockVLPR

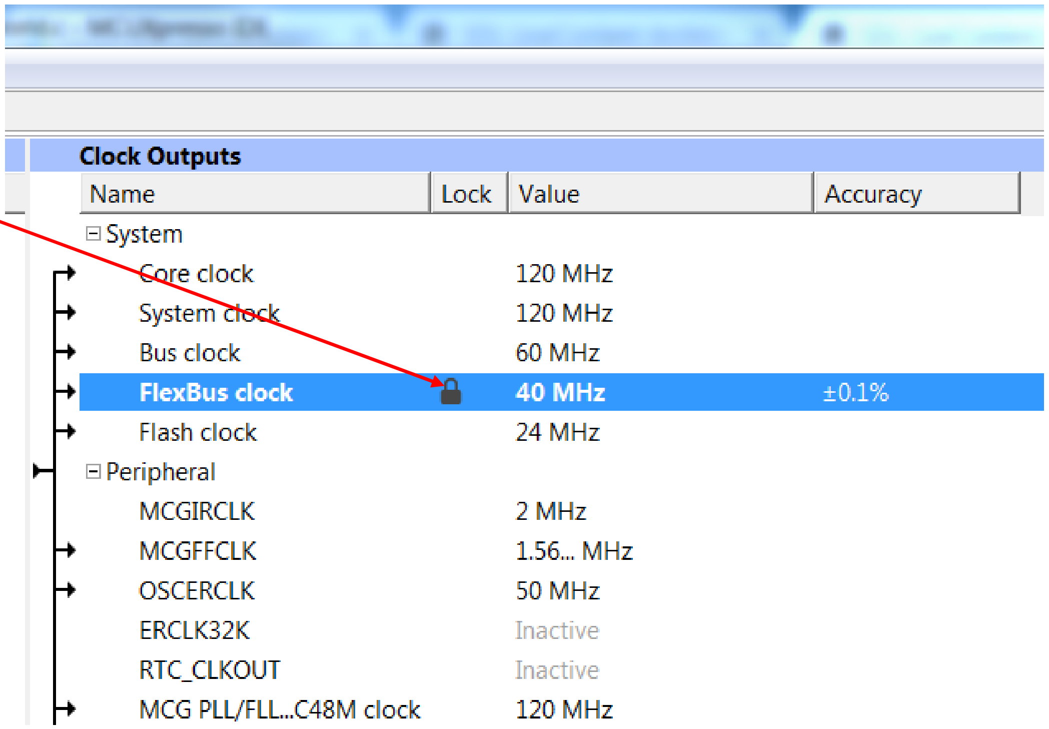
Problems

type filter text

Level	Issue	Origin	Targ

Locked Settings

- Lock Icon indicates that a setting (that may be automatically adjusted by the tool) is locked to prevent any automatic adjustment.
- If the setting can be locked, they are automatically locked when you change the value.
- To add/remove the lock manually, use the pop-up menu command **Lock/Unlock**.

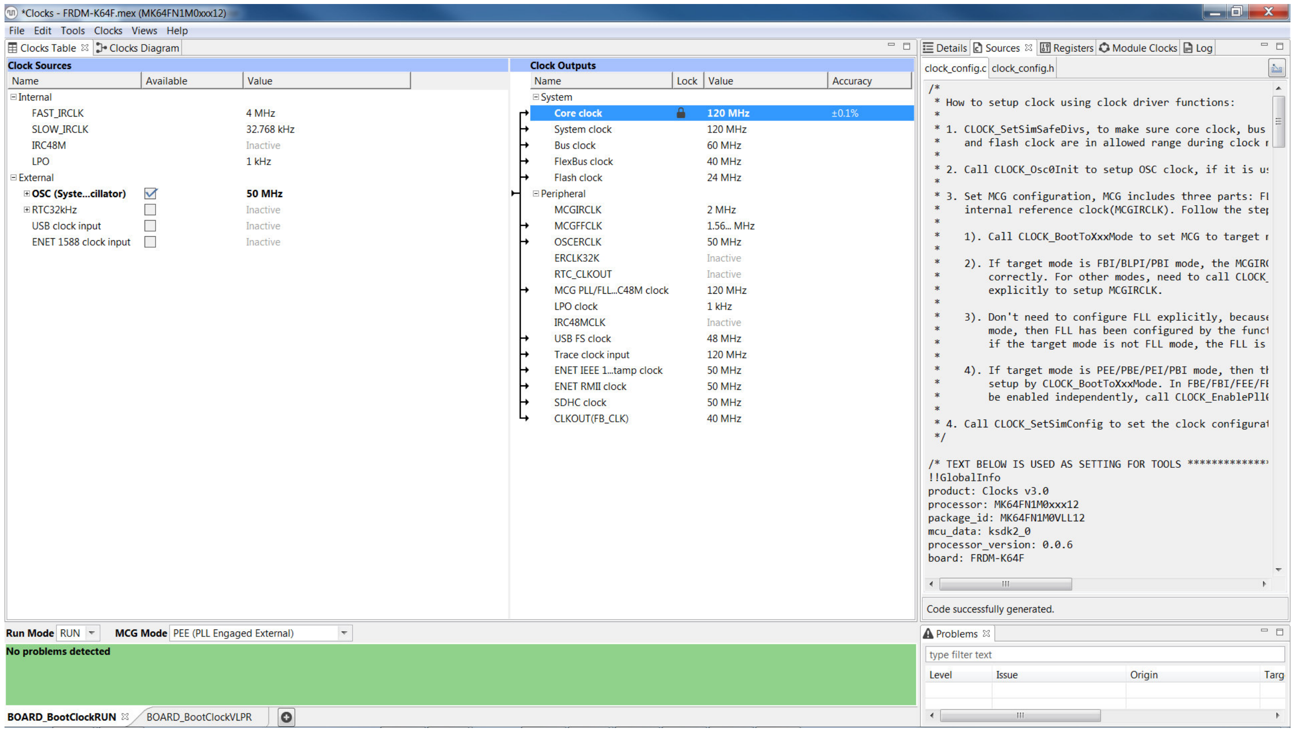


Clock Outputs

Name	Lock	Value	Accuracy
⊟ System			
Core clock		120 MHz	
System clock		120 MHz	
Bus clock		60 MHz	
FlexBus clock	🔒	**40 MHz**	±0.1%
Flash clock		24 MHz	
⊟ Peripheral			
MCGIRCLK		2 MHz	
MCGFFCLK		1.56... MHz	
OSCERCLK		50 MHz	
ERCLK32K		Inactive	
RTC_CLKOUT		Inactive	
MCG PLL/FLL...C48M clock		120 MHz	

Dependency Arrows

- Arrows between the Clock Sources and Clock Outputs indicate dependency
- Arrows lead from the clock source used for the selected output
- Arrows lead to clock outputs that are using the signal from the same clock source (as selected output)



Clock Sources

Name	Available	Value
Internal		
FAST_IRCLK		4 MHz
SLOW_IRCLK		32.768 kHz
IRC48M		Inactive
LPO		1 kHz
External		
OSC (Syste...cillator)	✓	**50 MHz**
RTC32kHz	☐	Inactive
USB clock input	☐	Inactive
ENET 1588 clock input	☐	Inactive

Clock Outputs

Name	Lock	Value	Accuracy
System			
Core clock	🔒	**120 MHz**	±0.1%
System clock		120 MHz	
Bus clock		60 MHz	
FlexBus clock		40 MHz	
Flash clock		24 MHz	
Peripheral			
MCGIRCLK		2 MHz	
MCGFFCLK		1.56... MHz	
OSCERCLK		50 MHz	
ERCLK32K		Inactive	
RTC_CLKOUT		Inactive	
MCG PLL/FLL...C48M clock		120 MHz	
LPO clock		1 kHz	
IRC48MCLK		Inactive	
USB FS clock		48 MHz	
Trace clock input		120 MHz	
ENET IEEE 1...tamp clock		50 MHz	
ENET RMII clock		50 MHz	
SDHC clock		50 MHz	
CLKOUT(FB_CLK)		40 MHz	

```
/*
 * How to setup clock using clock driver functions:
 *
 * 1. CLOCK_SetSimSafeDivs, to make sure core clock, bus
 *    and flash clock are in allowed range during clock r
 *
 * 2. Call CLOCK_Osc0Init to setup OSC clock, if it is us
 *
 * 3. Set MCG configuration, MCG includes three parts: Fl
 *    internal reference clock(MCGIRCLK). Follow the step
 *
 *    1). Call CLOCK_BootToXxxMode to set MCG to target r
 *
 *    2). If target mode is FBI/BLPI/PBI mode, the MCGIR(
 *        correctly. For other modes, need to call CLOCK_
 *        explicitly to setup MCGIRCLK.
 *
 *    3). Don't need to configure FLL explicitly, because
 *        mode, then FLL has been configured by the funct
 *        if the target mode is not FLL mode, the FLL is
 *
 *    4). If target mode is PEE/PBE/PEI/PBI mode, then th
 *        setup by CLOCK_BootToXxxMode. In FBE/FBI/FEE/FE
 *        be enabled independently, call CLOCK_EnablePll(
 *
 * 4. Call CLOCK_SetSimConfig to set the clock configurat
 */

/* TEXT BELOW IS USED AS SETTING FOR TOOLS *************
!!GlobalInfo
product: Clocks v3.0
processor: MK64FN1M0xxx12
package_id: MK64FN1M0VLL12
mcu_data: ksdk2_0
processor_version: 0.0.6
board: FRDM-K64F
```

Code successfully generated.

Run Mode RUN **MCG Mode** PEE (PLL Engaged External)

No problems detected

Problems

type filter text

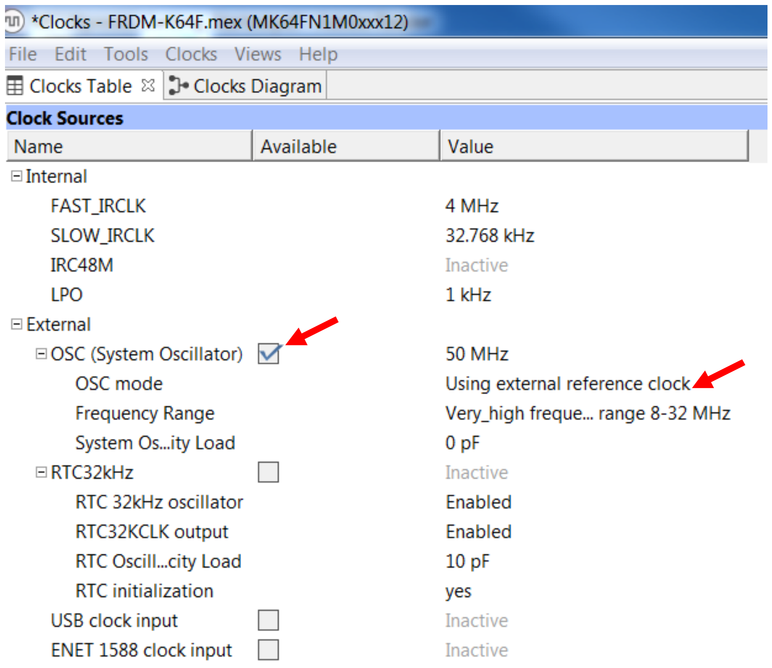
Level	Issue	Origin	Targ

BOARD_BootClockRUN BOARD_BootClockVLPR

Enabling External Clock Sources in Clocks Table View

- External clock sources can be enabled in the Clock Sources panel

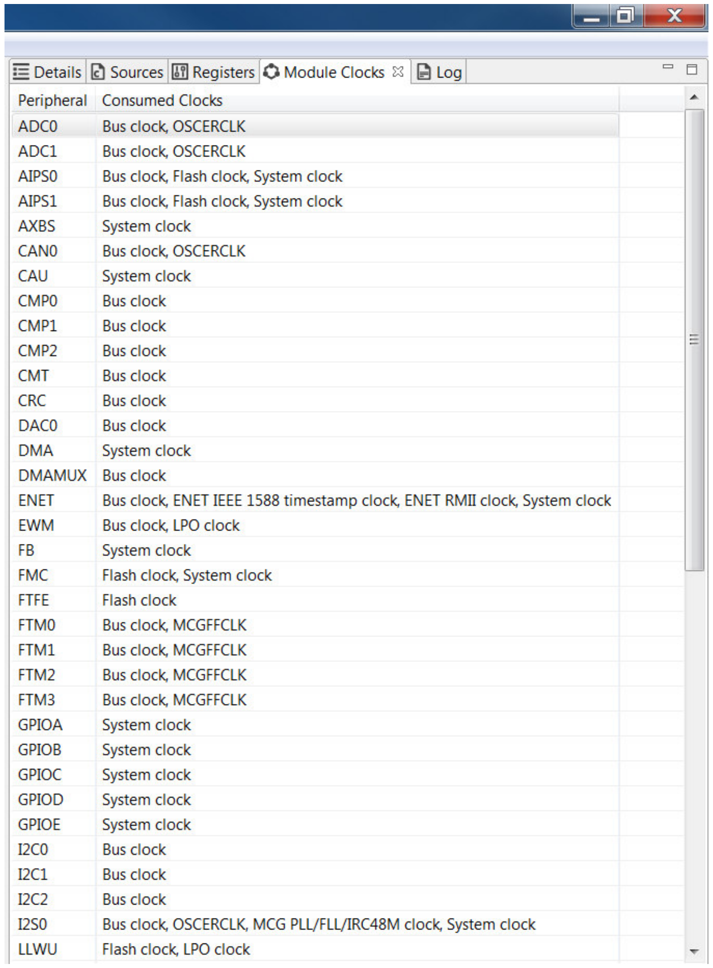
- External clock settings can also be changed in this view


*Clocks - FRDM-K64F.mex (MK64FN1M0xxx12)
File Edit Tools Clocks Views Help
Clocks Table
Clocks Diagram
Clock Sources

Name	Available	Value
⊟ Internal		
FAST_IRCLK		4 MHz
SLOW_IRCLK		32.768 kHz
IRC48M		Inactive
LPO		1 kHz
⊟ External		
⊟ OSC (System Oscillator)	✓	50 MHz
OSC mode		Using external reference clock
Frequency Range		Very_high freque... range 8-32 MHz
System Os...ity Load		0 pF
⊟ RTC32kHz	☐	Inactive
RTC 32kHz oscillator		Enabled
RTC32KCLK output		Enabled
RTC Oscill...city Load		10 pF
RTC initialization		yes
USB clock input	☐	Inactive
ENET 1588 clock input	☐	Inactive

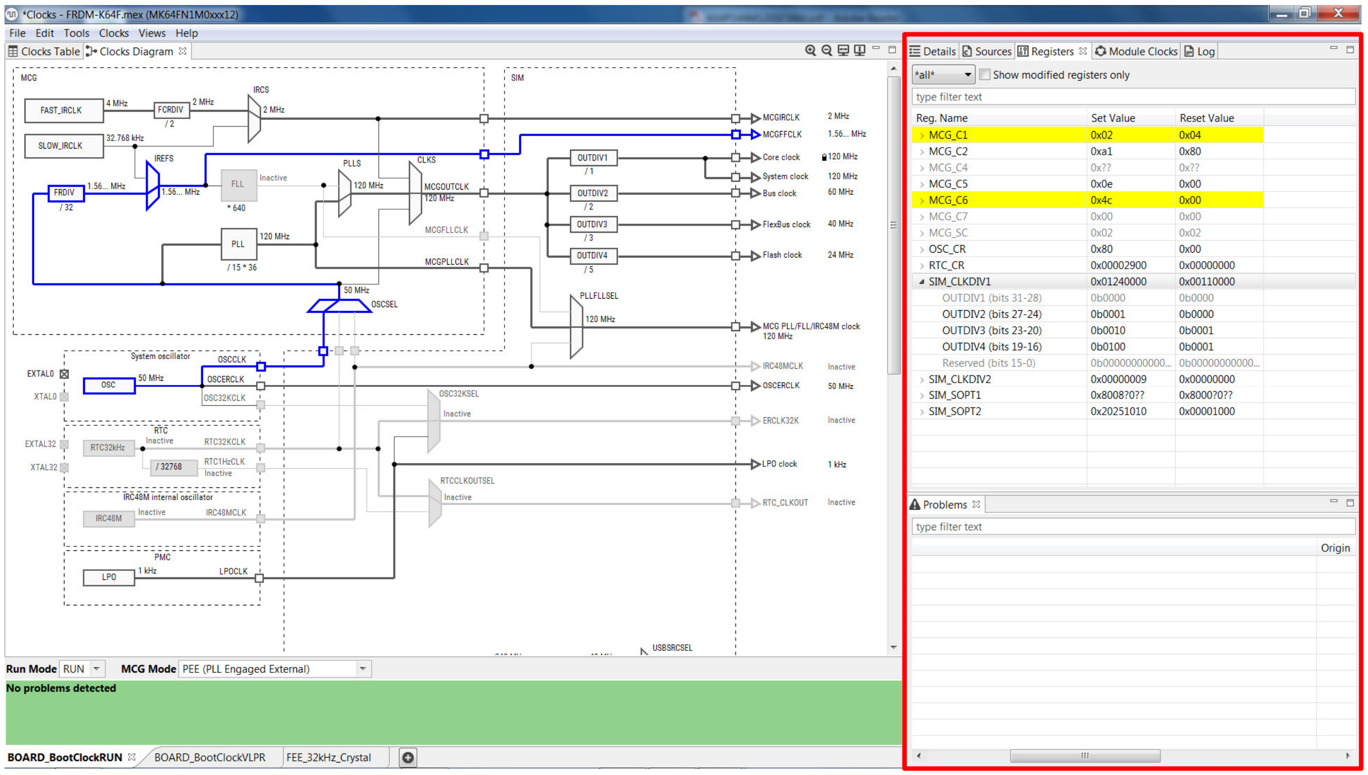
Module Clocks View

- Provides list of peripherals and currently selected clock sources



Peripheral	Consumed Clocks
ADC0	Bus clock, OSCERCLK
ADC1	Bus clock, OSCERCLK
AIPS0	Bus clock, Flash clock, System clock
AIPS1	Bus clock, Flash clock, System clock
AXBS	System clock
CAN0	Bus clock, OSCERCLK
CAU	System clock
CMP0	Bus clock
CMP1	Bus clock
CMP2	Bus clock
CMT	Bus clock
CRC	Bus clock
DAC0	Bus clock
DMA	System clock
DMAMUX	Bus clock
ENET	Bus clock, ENET IEEE 1588 timestamp clock, ENET RMII clock, System clock
EWM	Bus clock, LPO clock
FB	System clock
FMC	Flash clock, System clock
FTFE	Flash clock
FTM0	Bus clock, MCGFFCLK
FTM1	Bus clock, MCGFFCLK
FTM2	Bus clock, MCGFFCLK
FTM3	Bus clock, MCGFFCLK
GPIOA	System clock
GPIOB	System clock
GPIOC	System clock
GPIOD	System clock
GPIOE	System clock
I2C0	Bus clock
I2C1	Bus clock
I2C2	Bus clock
I2S0	Bus clock, OSCERCLK, MCG PLL/FLL/IRC48M clock, System clock
LLWU	Flash clock, LPO clock

Registers View



all ☐ Show modified registers only

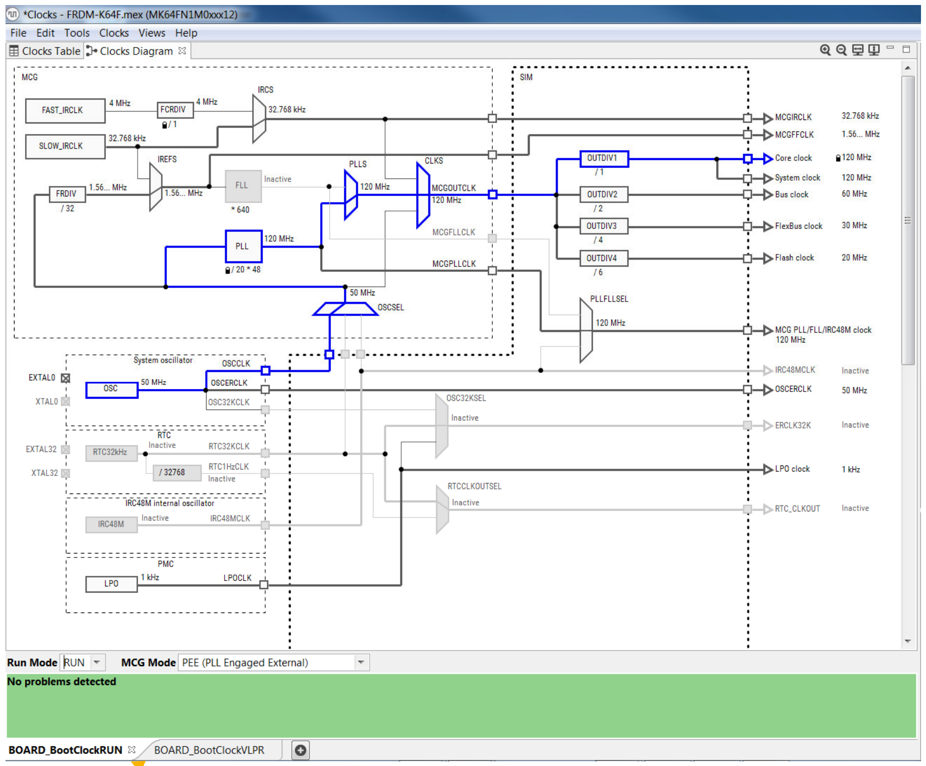
Reg. Name	Set Value	Reset Value
MCG_C1	0x02	0x04
MCG_C2	0xa1	0x80
MCG_C4	0x??	0x??
MCG_C5	0x0e	0x00
MCG_C6	0x4c	0x00
MCG_C7	0x00	0x00
MCG_SC	0x02	0x02
OSC_CR	0x80	0x00
RTC_CR	0x00002900	0x00000000
SIM_CLKDIV1	0x01240000	0x00110000
OUTDIV1 (bits 31-28)	0b0000	0b0000
OUTDIV2 (bits 27-24)	0b0001	0b0000
OUTDIV3 (bits 23-20)	0b0010	0b0001
OUTDIV4 (bits 19-16)	0b0100	0b0001
Reserved (bits 15-0)	0b00000000000...	0b00000000000...
SIM_CLKDIV2	0x00000009	0x00000000
SIM_SOPT1	0x8008?0??	0x8000?0??
SIM_SOPT2	0x20251010	0x00001000

Run Mode RUN   MCG Mode PEE (PLL Engaged External)

No problems detected

- Displays register values for current clock configuration
- Current register value and default value after reset are shown
- Recently changed registers are highlighted in yellow.

Configuration Tabs

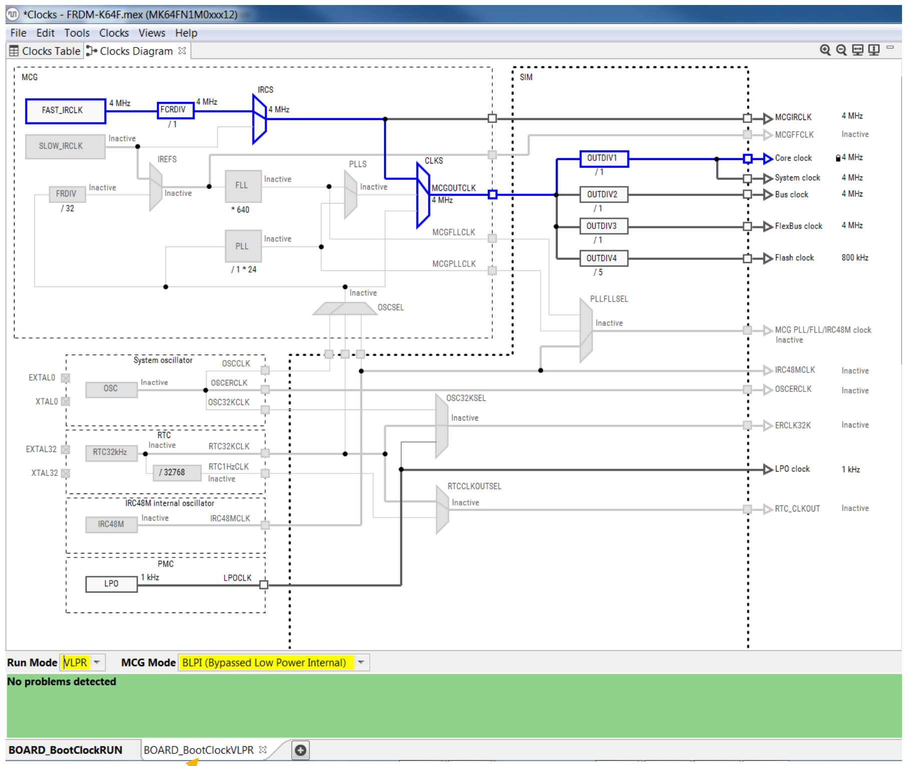


*Clocks - FRDM-K64F.mex (MK64FN1M0xxx12)

Run Mode RUN **MCG Mode** PEE (PLL Engaged External)

No problems detected

BOARD_BootClockRUN BOARD_BootClockVLPR



*Clocks - FRDM-K64F.mex (MK64FN1M0xxx12)

Run Mode VLPR **MCG Mode** BLPI (Bypassed Low Power Internal)

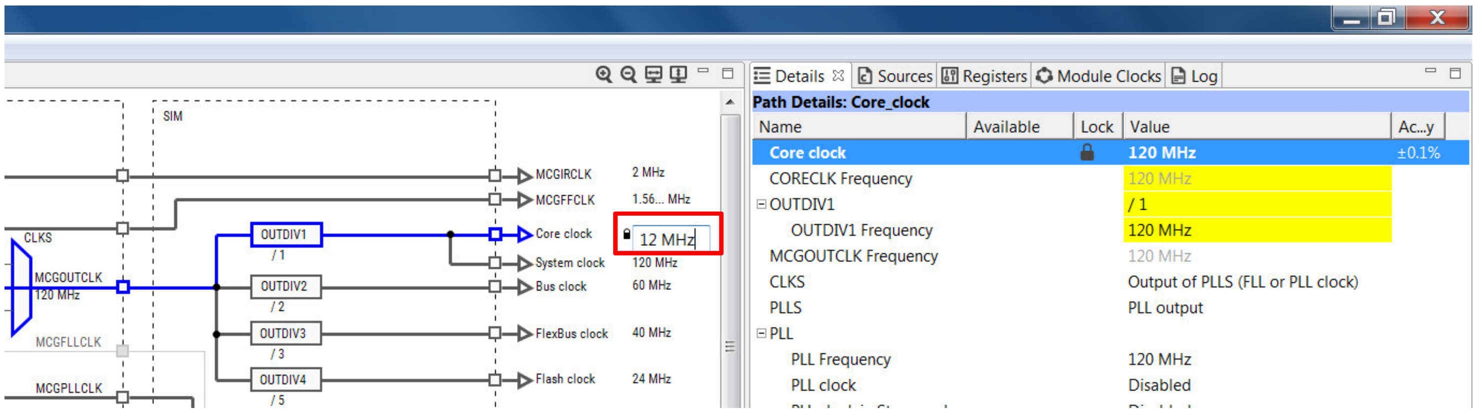
No problems detected

BOARD_BootClockRUN BOARD_BootClockVLPR

- Switch between clock configurations using the tabs

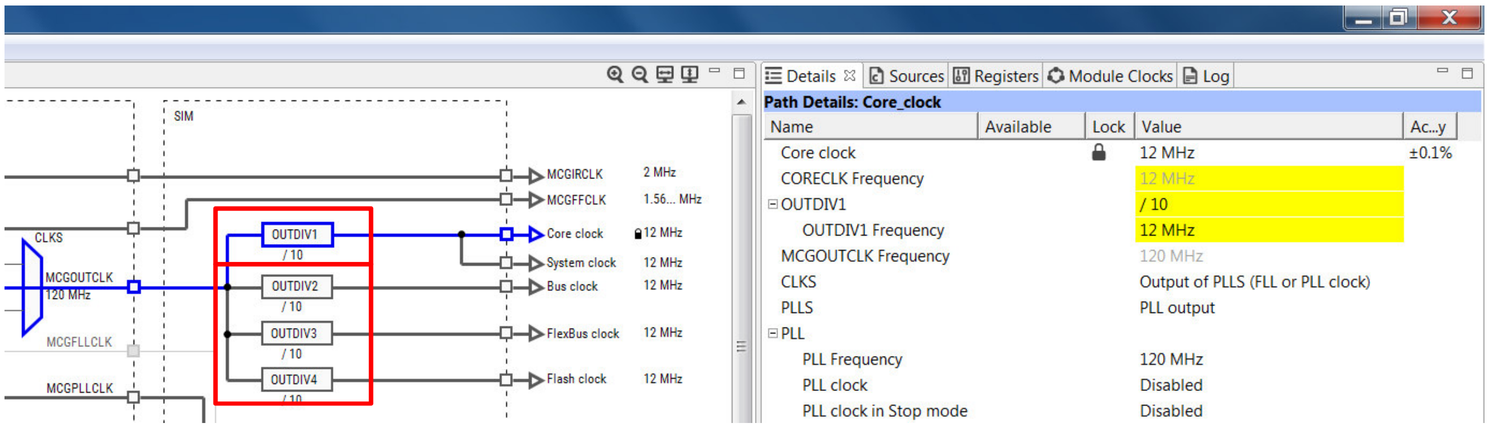
Clocks Diagram: Change Clock Output Frequency

- Clock output frequency can be typed directly into the field (e.g. core clock)
- Tool will attempt to achieve target frequency by increasing divider (e.g. OUTDIV1)
- Bus/Flexbus/Flash divide values also update to meet requirement (must be less than or equal to core clock)



Path Details: Core_clock

Name	Available	Lock	Value	Ac...y
Core clock		🔒	120 MHz	±0.1%
CORECLK Frequency			120 MHz	
⊟ OUTDIV1			/ 1	
OUTDIV1 Frequency			120 MHz	
MCGOUTCLK Frequency			120 MHz	
CLKS			Output of PLLS (FLL or PLL clock)	
PLLS			PLL output	
⊟ PLL				
PLL Frequency			120 MHz	
PLL clock			Disabled	

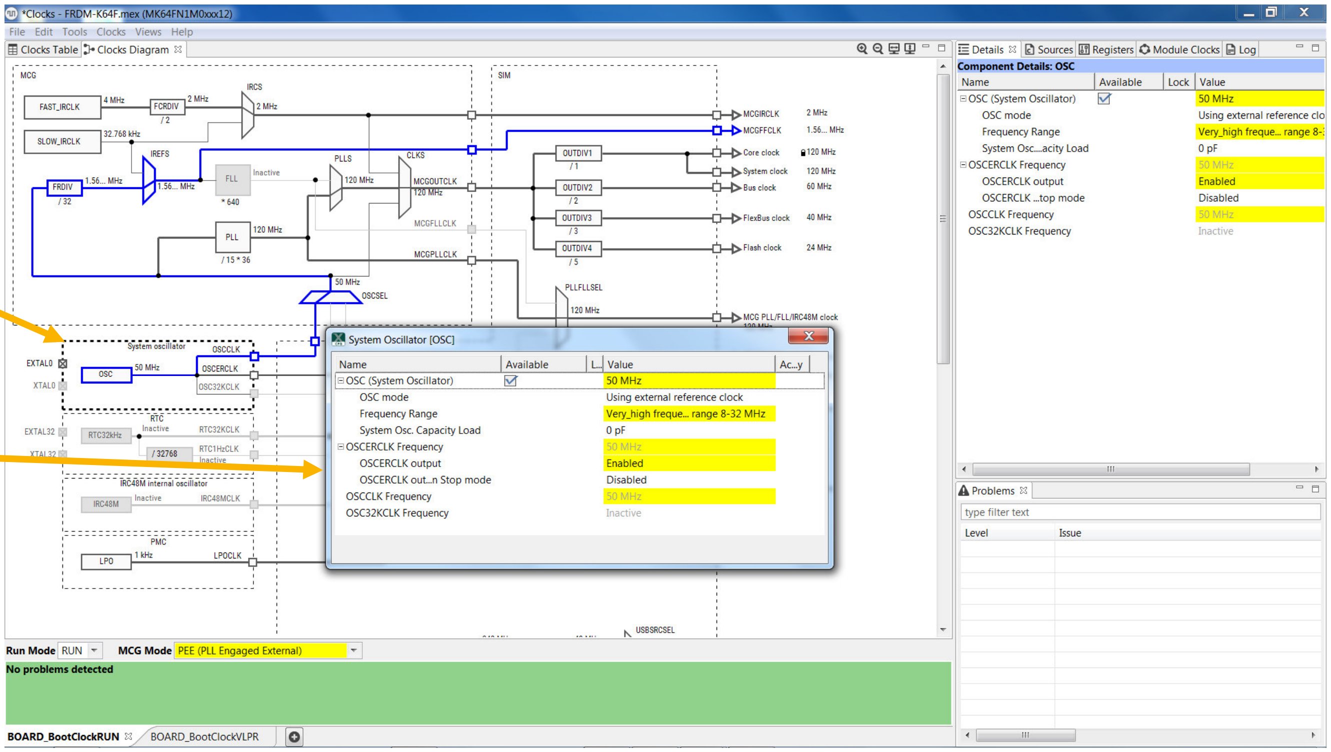


Path Details: Core_clock

Name	Available	Lock	Value	Ac...y
Core clock		🔒	12 MHz	±0.1%
CORECLK Frequency			12 MHz	
⊟ OUTDIV1			/ 10	
OUTDIV1 Frequency			12 MHz	
MCGOUTCLK Frequency			120 MHz	
CLKS			Output of PLLS (FLL or PLL clock)	
PLLS			PLL output	
⊟ PLL				
PLL Frequency			120 MHz	
PLL clock			Disabled	
PLL clock in Stop mode			Disabled	

Clocks Diagram: Component Settings

- Double click on a component in the block diagram to view/modify its configuration
- Pop-up box shows current configuration of component
- Element settings (e.g. frequency, etc.) can be modified in pop-up box



Component Details: OSC

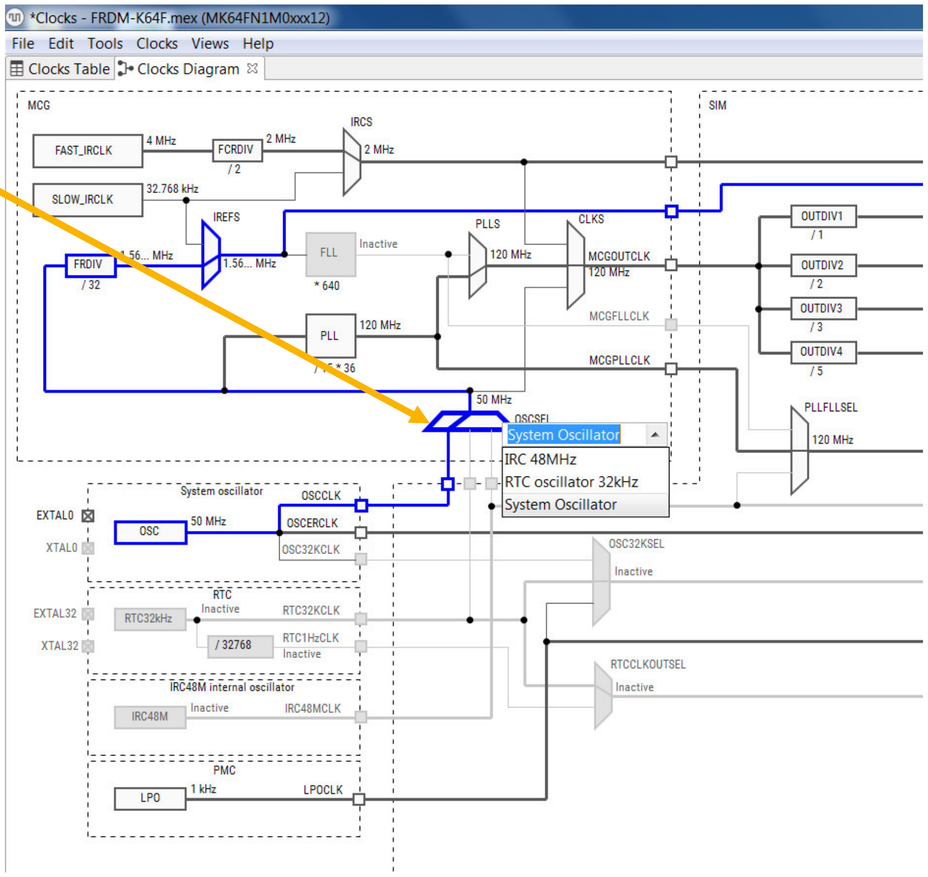
Name	Available	Lock	Value
OSC (System Oscillator)	✓		50 MHz
OSC mode			Using external reference clo
Frequency Range			Very_high freque... range 8-3
System Osc....acity Load			0 pF
OSCERCLK Frequency			50 MHz
OSCERCLK output			Enabled
OSCERCLK ...top mode			Disabled
OSCCLK Frequency			50 MHz
OSC32KCLK Frequency			Inactive

System Oscillator [OSC]

Name	Available	L...	Value	Ac...y
OSC (System Oscillator)	✓		50 MHz	
OSC mode			Using external reference clock	
Frequency Range			Very_high freque... range 8-32 MHz	
System Osc. Capacity Load			0 pF	
OSCERCLK Frequency			50 MHz	
OSCERCLK output			Enabled	
OSCERCLK out...n Stop mode			Disabled	
OSCCLK Frequency			50 MHz	
OSC32KCLK Frequency			Inactive	

Clocks Diagram: Change a Clock Mux

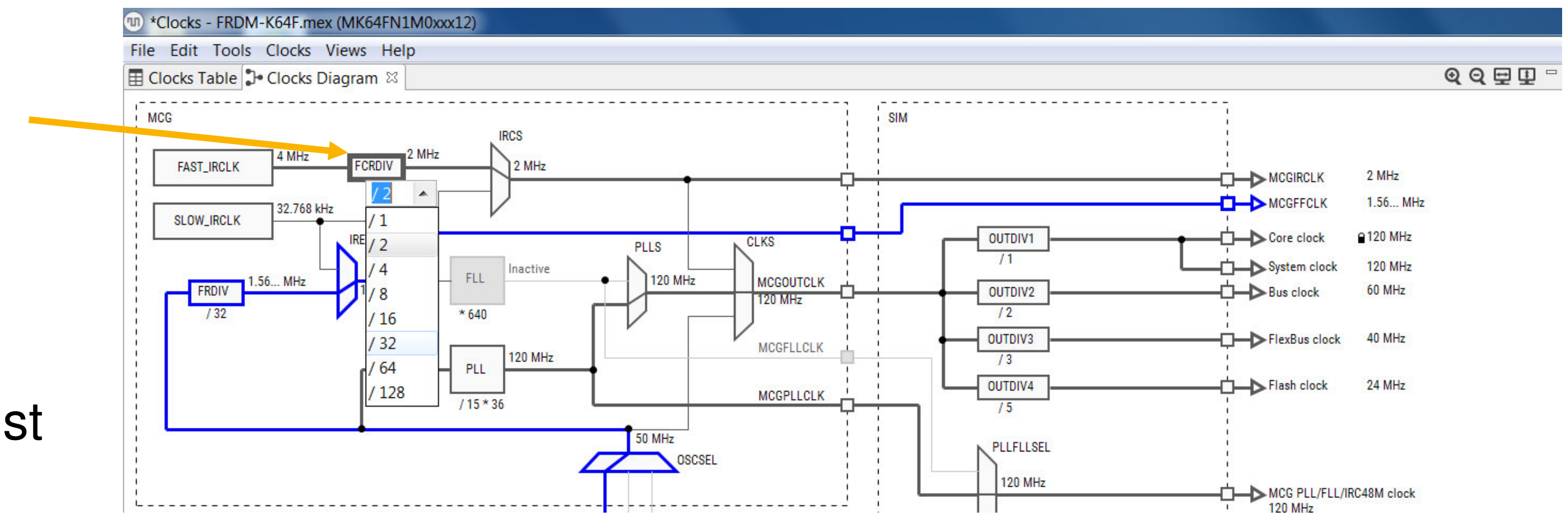
- Click once on a multiplexer for drop down of clock options
- Select from available clocks in the drop-down to change the mux
- Clock source must be enabled. Otherwise, an error will occur.



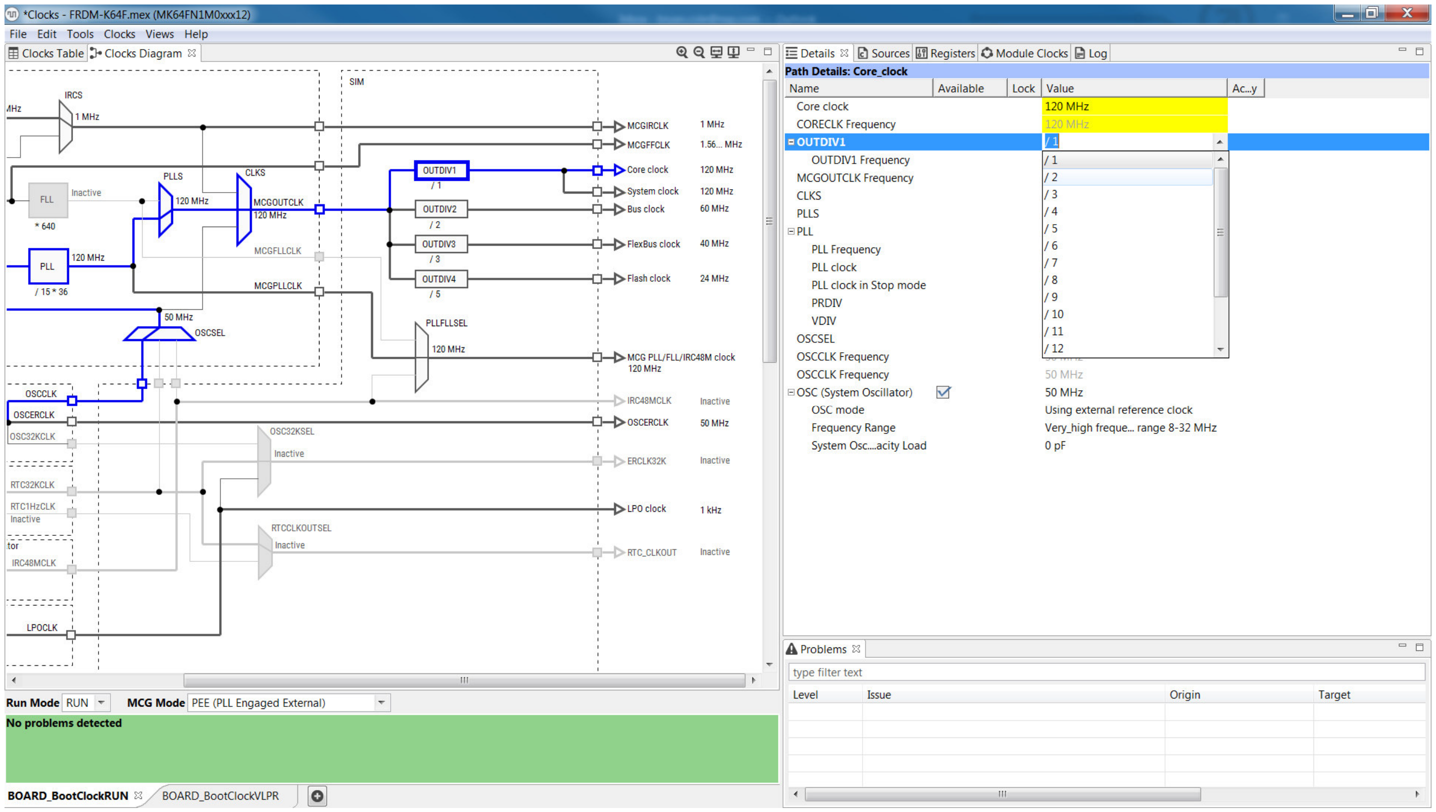
Clocks Diagram: Set a Clock Divider

- Click once on a divider for a drop-down of available divide values

- Select divide value from the list and the clock output frequency automatically updates



Details View



Run Mode RUN MCG Mode PEE (PLL Engaged External)

No problems detected

Path Details: Core_clock

Name	Available	Lock	Value	Ac...y
Core clock			120 MHz	
CORECLK Frequency			120 MHz	
OUTDIV1			/ 1	
OUTDIV1 Frequency				
MCGOUTCLK Frequency				
CLKS				
PLLS				
PLL				
PLL Frequency				
PLL clock				
PLL clock in Stop mode				
PRDIV				
VDIV				
OSCSEL				
OSCCLK Frequency				
OSCCLK Frequency			50 MHz	
OSC (System Oscillator)	✓		50 MHz	
OSC mode			Using external reference clock	
Frequency Range			Very_high freque... range 8-32 MHz	
System Osc....acity Load			0 pF	

/ 1, / 2, / 3, / 4, / 5, / 6, / 7, / 8, / 9, / 10, / 11, / 12

type filter text

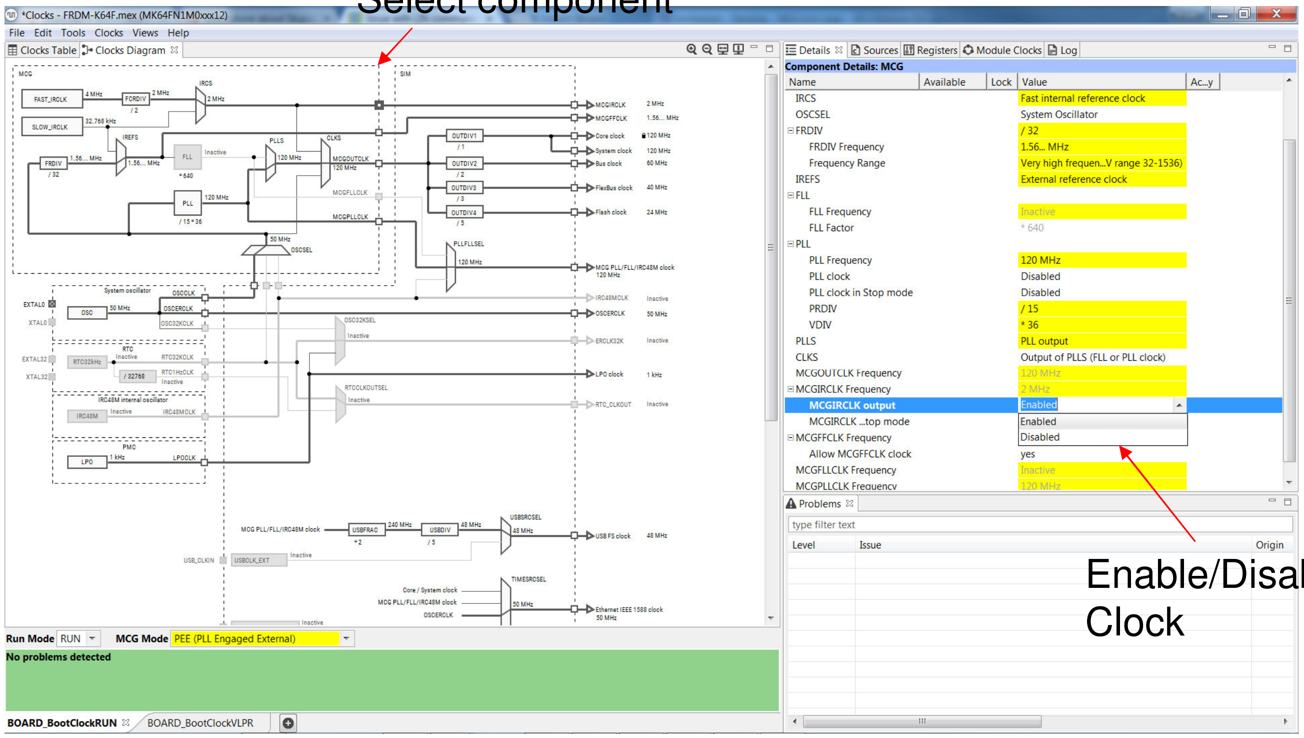
Level	Issue	Origin	Target

- Elements can be modified in the Details View

Enable/Disable A Clock Source

- Clock sources such as the Slow/Fast IRCLK and OSCERCLK have clock gates that can be enabled/disabled
- Select the component containing the clock (e.g. MCG for MCGIRCLK)
- Disable the MCGIRLCK in the Details VIew

Select component



Component Details: MCG

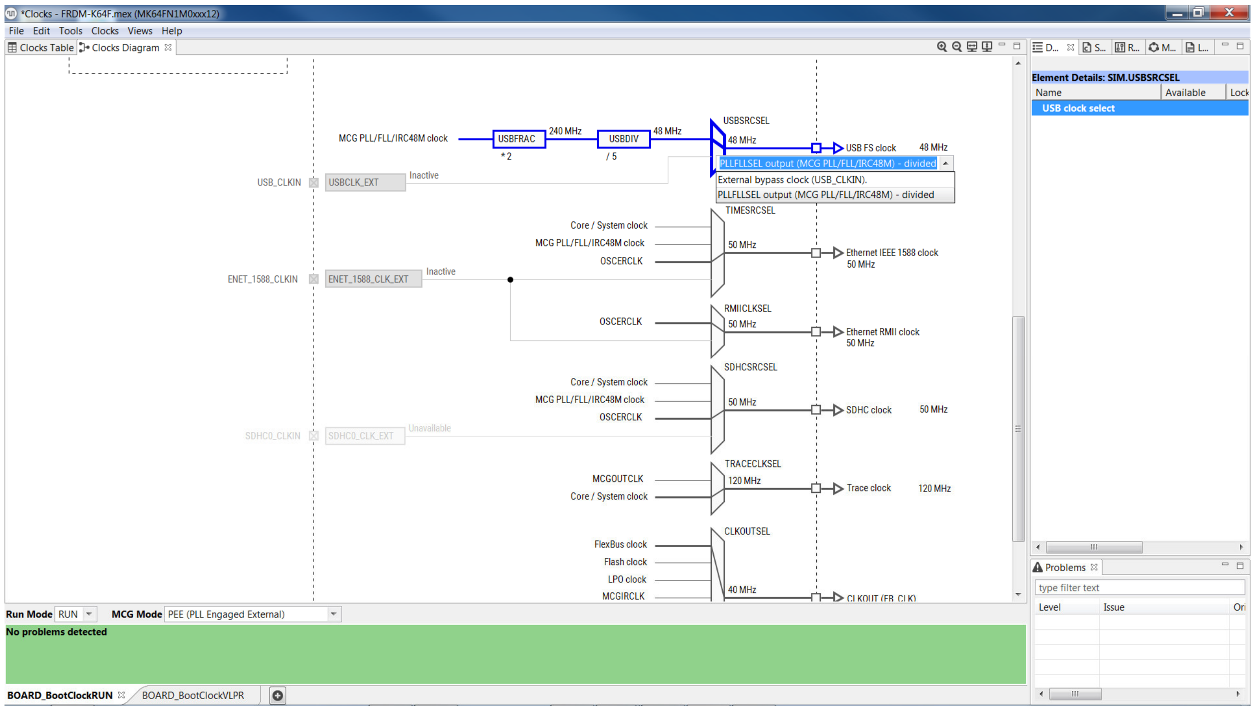
Name	Available	Lock	Value	Ac...y
IRCS			Fast internal reference clock	
OSCSEL			System Oscillator	
FRDIV			/ 32	
FRDIV Frequency			1.56... MHz	
Frequency Range			Very high frequen...V range 32-1536)	
IREFS			External reference clock	
FLL				
FLL Frequency			Inactive	
FLL Factor			* 640	
PLL				
PLL Frequency			120 MHz	
PLL clock			Disabled	
PLL clock in Stop mode			Disabled	
PRDIV			/ 15	
VDIV			* 36	
PLLS			PLL output	
CLKS			Output of PLLS (FLL or PLL clock)	
MCGOUTCLK Frequency			120 MHz	
MCGIRCLK Frequency			2 MHz	
MCGIRCLK output			Enabled (Enabled / Disabled)	
MCGIRCLK ...top mode				
MCGFFCLK Frequency				
Allow MCGFFCLK clock			yes	
MCGFLLCLK Frequency			Inactive	
MCGPLLCLK Frequency			120 MHz	

Run Mode RUN
MCG Mode PEE (PLL Engaged External)
No problems detected

Enable/Disable Clock

Peripheral Clocks in SIM

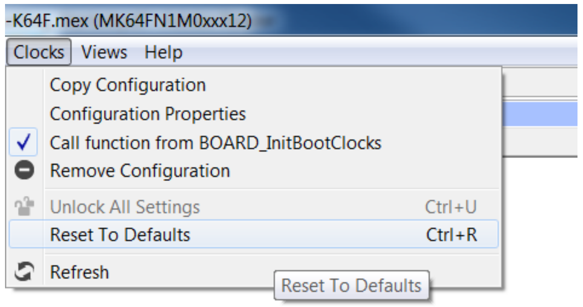
- Peripheral clock selections are often controlled through the SIM
- The multiplexers in the SIM component can be used to change the clock source for a peripheral
- The clock source must be enabled to change the mux

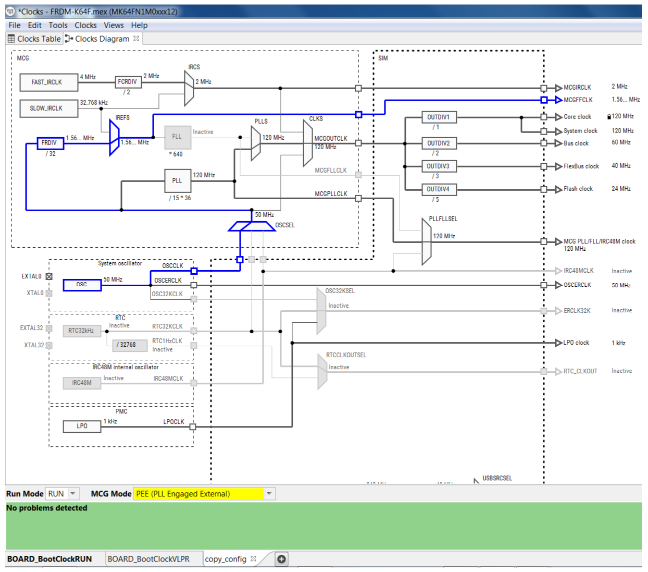


Run Mode RUN
MCG Mode PEE (PLL Engaged External)
No problems detected

Reset To Defaults

- Reset to Defaults
  - Resets clock configuration to the default reset clock configuration for the processor
  - Not the same as Board_BootClock configuration

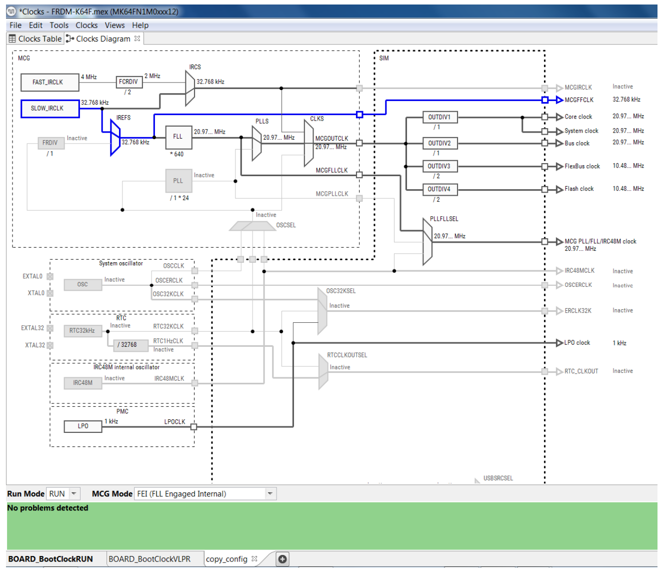




Run Mode RUN MCG Mode PEE (PLL Engaged External)

No problems detected

BOARD_BootClockRUN BOARD_BootClockVLPR copy_config



Run Mode RUN MCG Mode FEI (FLL Engaged Internal)

No problems detected

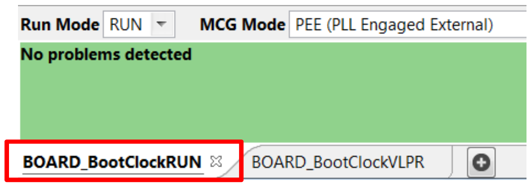
BOARD_BootClockRUN BOARD_BootClockVLPR copy_config

- PEE at 120 MHz
- FEI at 21 MHz (reset)

Copy Clock Configuration

- To Copy an existing clock configuration:

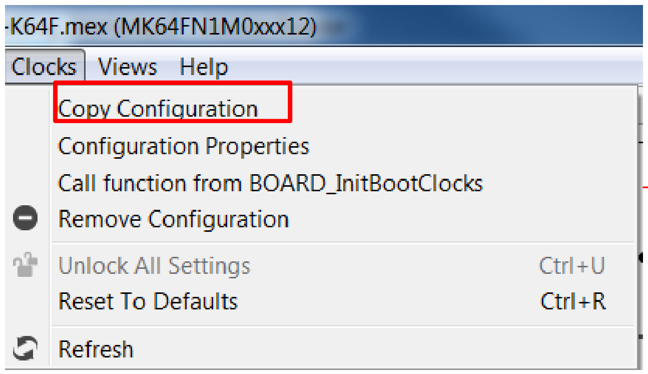
1. Select an existing configuration tab



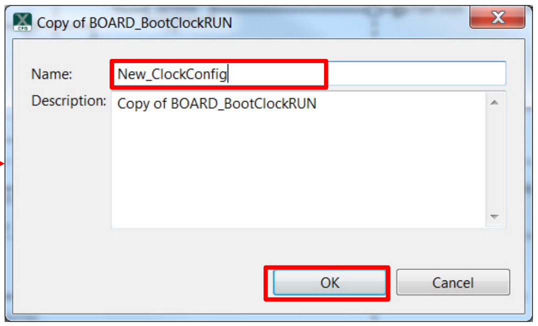
Run Mode RUN MCG Mode PEE (PLL Engaged External)

No problems detected

2. Select Clocks > Copy Configuration



3. Name the configuration. Select ok

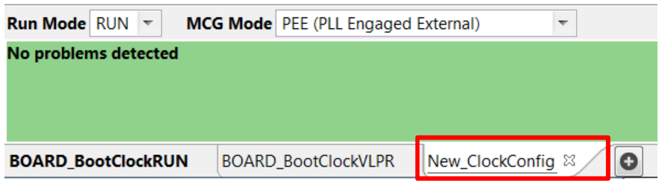


Copy of BOARD_BootClockRUN

Name: New_ClockConfig

Description: Copy of BOARD_BootClockRUN

4. Copied Configuration will be created

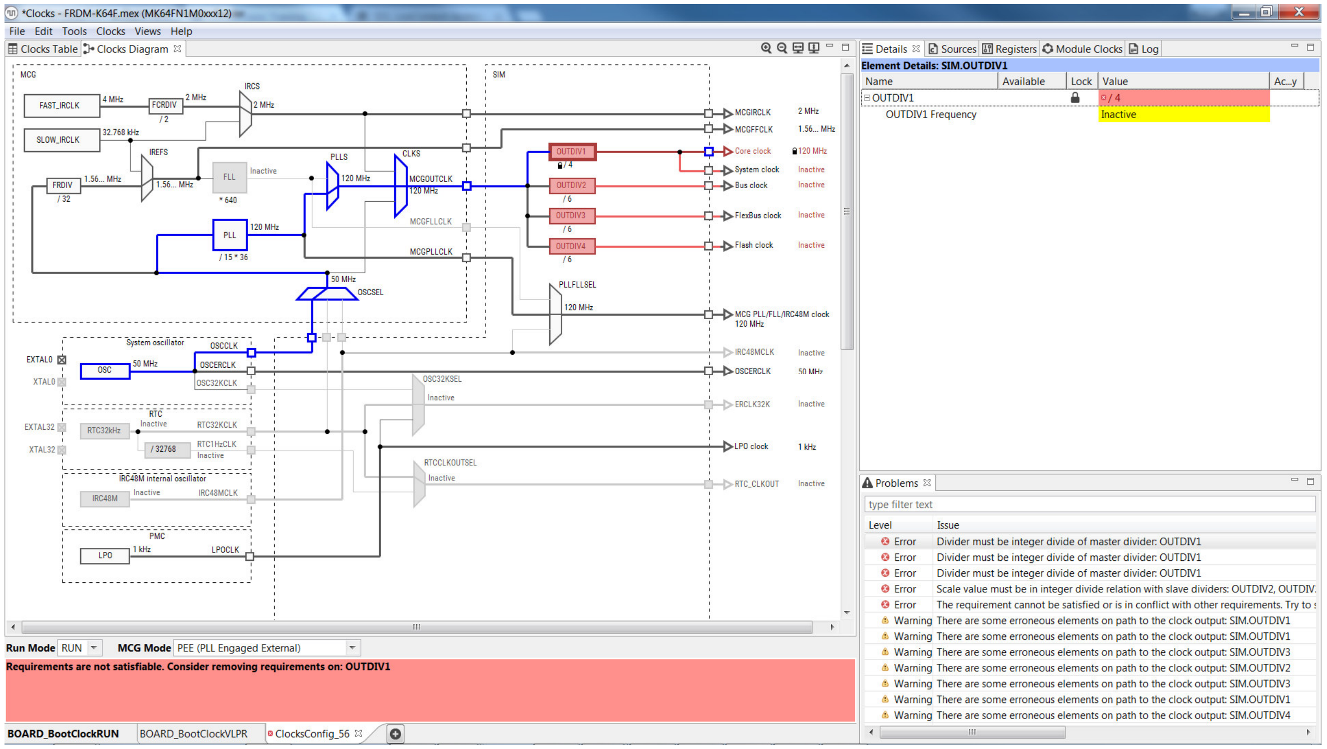


Run Mode RUN MCG Mode PEE (PLL Engaged External)

No problems detected

Configuration Errors

- Errors may occur when changing the clock configurations
- Error conditions will cause the status bar to become RED
- Potential Problems:
  - **Requirement(s) not satisfiable:** Indicates that there are one or more locked frequency or frequency constraints for which the tool is not able to find a valid settings and satisfy those requirements.
  - **Invalid settings or requirements:** [*element list*] – Indicates that the value of some settings is not valid. For example:The current state of settings is beyond the acceptable range.


*Clocks - FRDM-K64F.mex (MK64FN1M0xxx12)
File Edit Tools Clocks Views Help
Clocks Table
Clocks Diagram
Details
Sources
Registers
Module Clocks
Log
Element Details: SIM.OUTDIV1

Name	Available	Lock	Value	Ac...y
OUTDIV1		🔒	0 / 4	
OUTDIV1 Frequency			Inactive	

Problems

Level	Issue
Error	Divider must be integer divide of master divider: OUTDIV1
Error	Divider must be integer divide of master divider: OUTDIV1
Error	Divider must be integer divide of master divider: OUTDIV1
Error	Scale value must be in integer divide relation with slave dividers: OUTDIV2, OUTDIV
Error	The requirement cannot be satisfied or is in conflict with other requirements. Try to s
Warning	There are some erroneous elements on path to the clock output: SIM.OUTDIV1
Warning	There are some erroneous elements on path to the clock output: SIM.OUTDIV1
Warning	There are some erroneous elements on path to the clock output: SIM.OUTDIV3
Warning	There are some erroneous elements on path to the clock output: SIM.OUTDIV2
Warning	There are some erroneous elements on path to the clock output: SIM.OUTDIV3
Warning	There are some erroneous elements on path to the clock output: SIM.OUTDIV1
Warning	There are some erroneous elements on path to the clock output: SIM.OUTDIV4

Run Mode: RUN
MCG Mode: PEE (PLL Engaged External)

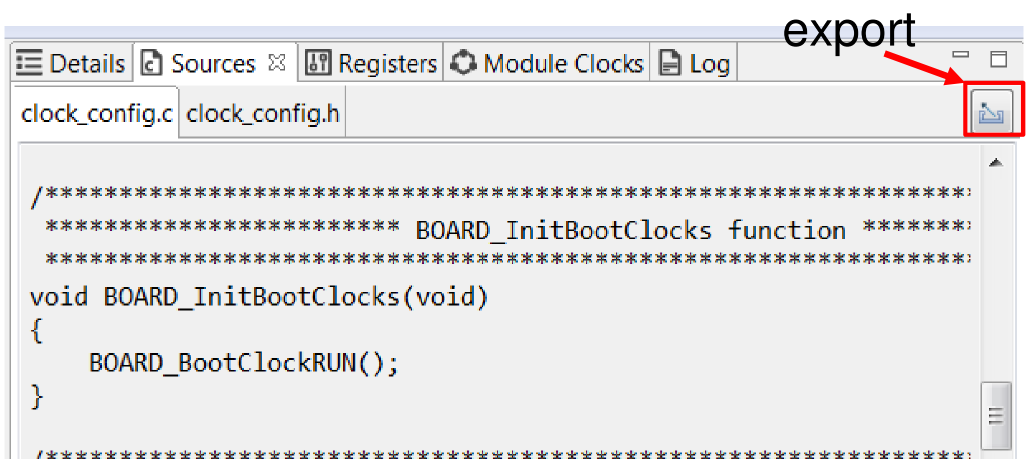
Requirements are not satisfiable. Consider removing requirements on: OUTDIV1

Export Clock Configuration

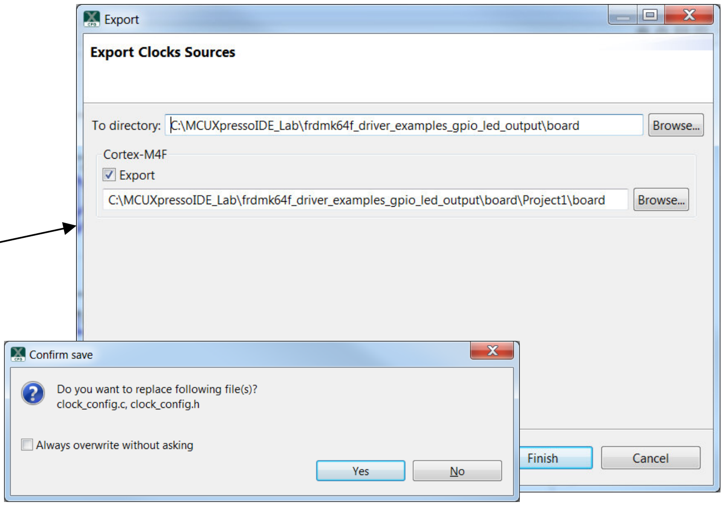
- Export a configuration to clock_config.c and clock_config.h files
- Exports source files, and generates code to initialize current clock configurations

- Access from menu:
  – File->Export
  – Can also be accessed from Export icon in Sources view

- Select the directory to export clock_config.c and clock_config.h.



```
/*********************************************************************
 ********************** BOARD_InitBootClocks function ********
 *********************************************************************
void BOARD_InitBootClocks(void)
{
    BOARD_BootClockRUN();
}

/*********************************************************************
```



Export Clocks Sources

To directory: C:\MCUXpressoIDE_Lab\frdmk64f_driver_examples_gpio_led_output\board

Cortex-M4F

☑ Export

C:\MCUXpressoIDE_Lab\frdmk64f_driver_examples_gpio_led_output\board\Project1\board

Do you want to replace following file(s)?
clock_config.c, clock_config.h

☐ Always overwrite without asking

LAB 3

Lab 2 : To use Config Tools to generate code for pins and clock configuration

- **Pre-requisites**
- Boards
  - OM13092(LPCXpresso54608)
- Software
  - MCUXpresso IDE: http://nxp.com/mcuxpresso/ide
  - MCUXpresso Config Tools v3.0 : http://nxp.com/mcuxpresso/config
  - SDK_2.2_LPCXpresso54608 (Thumbdrive)
  - Terminal Software (like TeraTerm or PuTTY)
  - mbed Serial Driver: https://developer.mbed.org/handbook/Windows-serial-configuration
  - Pin_clk config code file (Thumbdrive)
- Follow Lab 3 instruction

Outline

- Create a new configuration based on hello_world example to be use in hot_pin&clock config project.
- Modify pins and generate the code to pin_mux.c and pin_mux.h files
- Generate the clock code to clock_config.c and clock_config.h
- Export pin and clock sources to workspace by dragging the files to Board directory in the pin&clock config project
- Initiate pins and clocks in hot_pin&clock_config.c by adding “BOARD_InitPins(); and BOARD_BootClockFROHF48M();
- Build,debug and run the project.
- Result in Tera Term : Success in Pin Config and Success in Clock Config!

AGENDA

- **MCUXpresso Software And Tools Overview**
- **MCUXpresso SDK**
    - Web Builder
    - File Structure
- **MCUXpresso IDE**
    - Importing/Building
    - Debugging
- **MCUXpresso Config Tool**
    - Project Cloner
    - Pins Tool
    - Clocks Tool
- **LPC54608 LCD Lab, Key API and EmWin Demo**

LPC54608

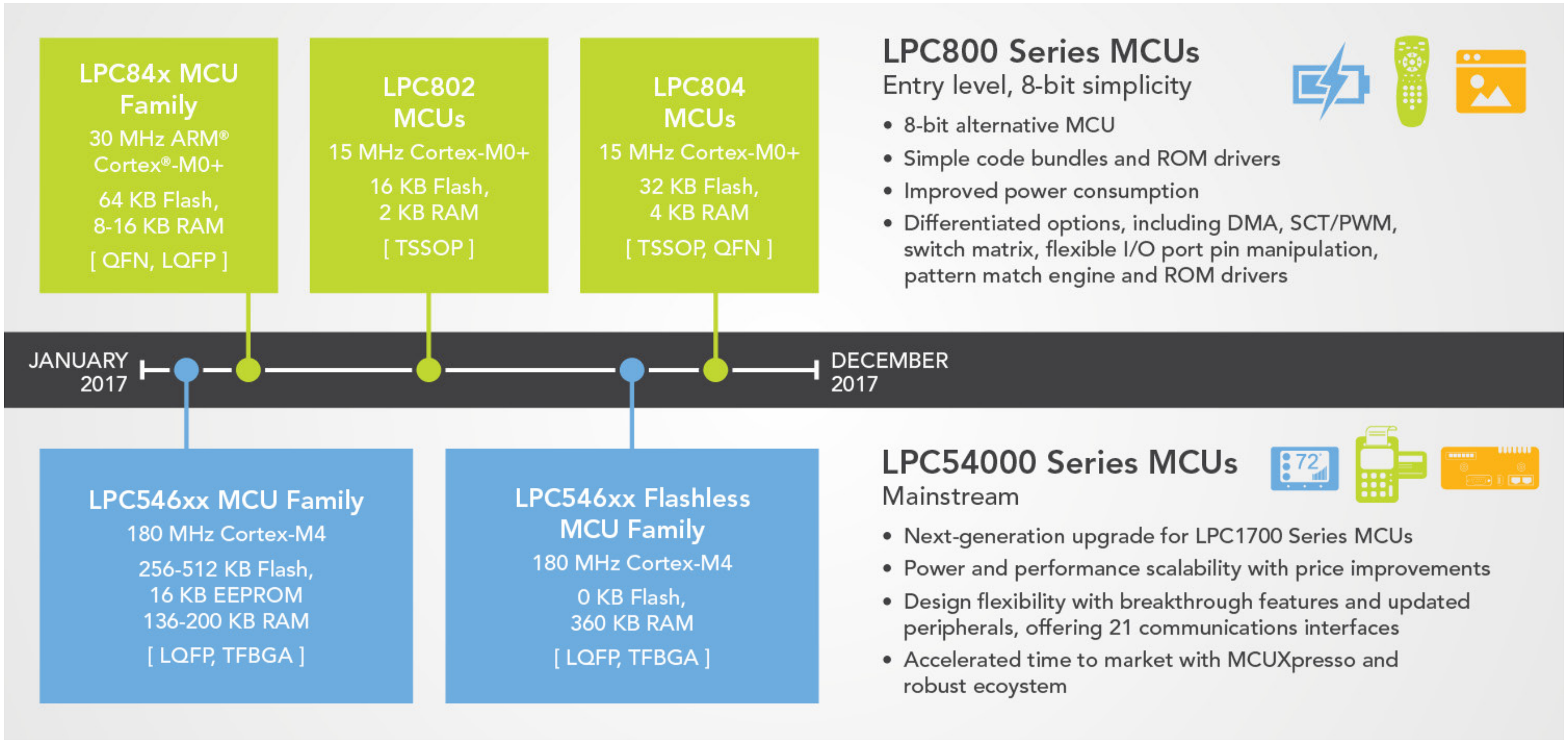
LPC546XX MCU FAMILY

SCALABLE AND POWER-EFFICIENT MULTI-MARKET MCUS

HMI & FLEXIBLE COMMUNICATION INTERFACES FOR IOT APPLICATIONS

SECURE CONNECTIONS
FOR A SMARTER WORLD

2017 LPC Roadmap



LPC84x MCU Family
30 MHz ARM® Cortex®-M0+
64 KB Flash, 8-16 KB RAM
[QFN, LQFP]

LPC802 MCUs
15 MHz Cortex-M0+
16 KB Flash, 2 KB RAM
[TSSOP]

LPC804 MCUs
15 MHz Cortex-M0+
32 KB Flash, 4 KB RAM
[TSSOP, QFN]

LPC546xx MCU Family
180 MHz Cortex-M4
256-512 KB Flash, 16 KB EEPROM
136-200 KB RAM
[LQFP, TFBGA]

LPC546xx Flashless MCU Family
180 MHz Cortex-M4
0 KB Flash, 360 KB RAM
[LQFP, TFBGA]

LPC800 Series MCUs
Entry level, 8-bit simplicity

- 8-bit alternative MCU
- Simple code bundles and ROM drivers
- Improved power consumption
- Differentiated options, including DMA, SCT/PWM, switch matrix, flexible I/O port pin manipulation, pattern match engine and ROM drivers

LPC54000 Series MCUs
Mainstream

- Next-generation upgrade for LPC1700 Series MCUs
- Power and performance scalability with price improvements
- Design flexibility with breakthrough features and updated peripherals, offering 21 communications interfaces
- Accelerated time to market with MCUXpresso and robust ecoystem

Introducing LPC54000 Series of MCU
Mainstream, Power Efficient Microcontrollers

LPC5410x

Baseline Cortex-M4

Cortex-M4F at 100 MHz
1.62 V to 3.6 V
256-512 KB Flash
104 KB RAM

Differentiating Features:
- Optional Dual Core (Cortex-M0+)
- <100uA / MHz (Cortex-M4)
- Digital Mic Subsystem

Available Now

LQFP64
CSP49

LPC5411x

FS USB
Large Internal SRAM

Cortex-M4F at 100 MHz
1.62 V to 3.6 V
128-256 KB Flash
96-192 KB RAM
FRO, FS USB

Differentiating Features:
- Optional Dual Core (Cortex-M0+)
- <80uA / MHz (Cortex-M4)
- Flexible Comm Interface
- Digital Mic Subsystem

Available Now

LQFP64
CSP49

LPC546xx

Performance & Integration

Cortex-M4F at 180 MHz
1.71 V to 3.6 V
256-512 KB Flash
136-200 KB RAM
FRO, FS/HS USB

Differentiating Features:
- 120uA / MHz (Cortex-M4)
- Flexible Comm Interfaces
- TFT-LCD Controller
- External Memory Interface
- Ethernet PTP IEE1588 v2
- Dual CAN2.0 / CAN-FD
- Digital Mic Subsystem

Available Now

LQFP208, TFBGA180 (NOW)
LQFP100, TFBGA100 (MAY)

LPC546xx Family Introduction

Power-efficiency, Advanced HMI & Flexible Comms for next-generation IoT

✓ Extremely Low Active Current with 180MHz Performance

- ARM Cortex-M4 core running up to 180MHz at 120 µA / MHz

✓ Advanced HMI & Flexible Communication Peripherals

- Up to 21 flexible communication peripherals to interface with memory, connectivity modules, and a variety of sensors
- Numerous wake-up sources, ample timers
- Integrated TFT control allows to keep the overall cost and complexity to a minimum

✓ Comprehensive Enablement

- Complimentary MCUXpresso IDE and Software Development Kit (SDK)
- Integration of Segger's emWIN Graphics Library into SDK
- Faster time to market with comprehensive development hardware and reference designs

LPC546xx Target Applications

Industrial, Building, Energy, General Embedded

- Diagnostic equipment
- Industrial control devices
- PLC
- Data Aggregator & Comms Hub
- Building control & automation
- HVAC control
- Multi-protocol bridge
- Data acquisition
- Medical/industrial grade scale
- Scanners / Mini printers

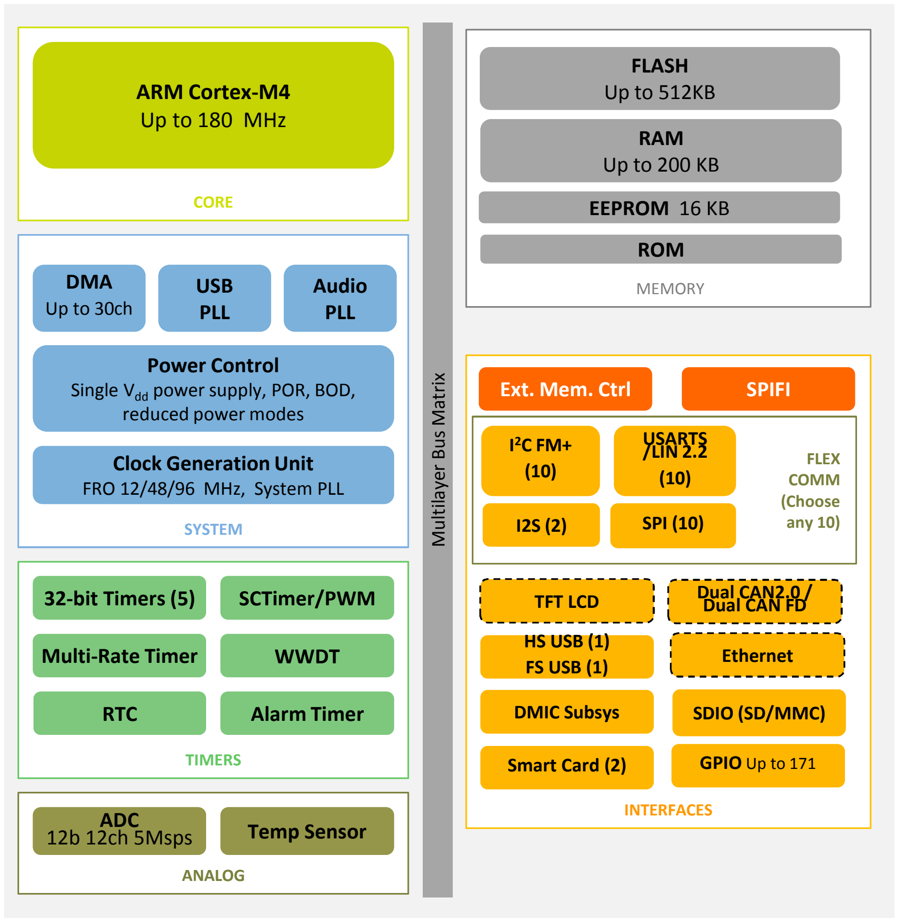
Consumer, Smart Home & Automation

- Small Appliance
- White Goods HMI
- Thermostat
- In Home Display (IHD)
- IOT gateway
- Security monitoring
- High end gaming accessories
- Fitness equipment
- Audio accessories

Automotive Aftermarket

- Satellite Radio
- Portable GPS Tracker
- Data aggregator for Infotainment/Navigation
- Fleet Management/Telematics
- Vehicle Diagnostic
- Tachograph
- OBD-II

LPC546xx Series Block Diagram



CPU
- 180MHz Cortex-M4 with floating point unit

Memory
- Up to 512 KB Flash, Up to 200 KB RAM
- 16 KB EEPROM

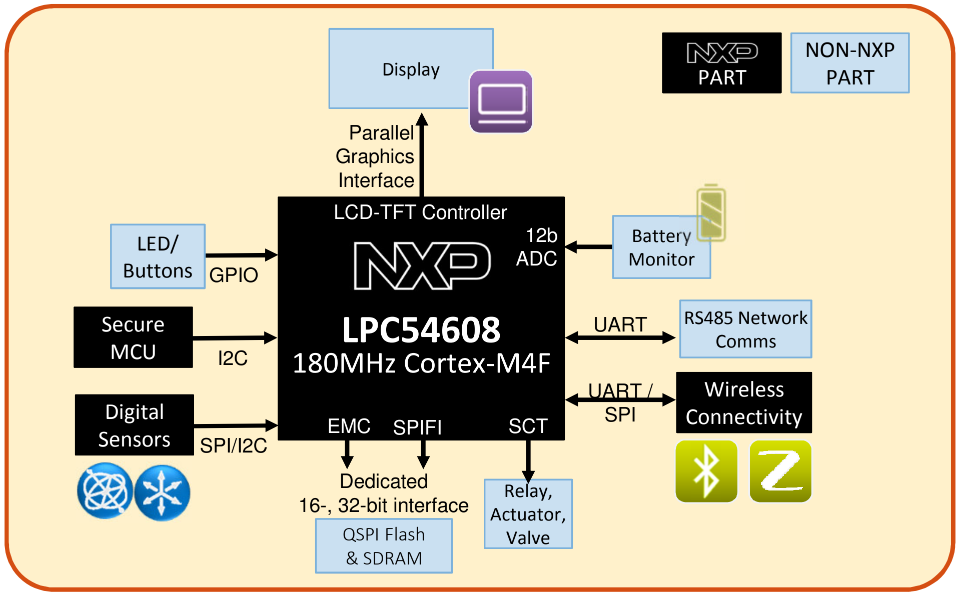
Interfaces for connectivity & sensors
- Dual CAN2.0 or CAN FD Controller Options
- XTAL-less FS USB (H/D)
- 10 SPI, 10 I2C, 10 USART, 2 I²S channels. Max 10 channels
- Graphic LCD with resolutions up to 1024x768
- 10/100 Ethernet Controller with PTP
- Stereo DMIC subsystem
  - (PDM, decimator, HW VAD)
- 1x HS USB (H/D) w/ on-chip HS PHY
- XIP from QSPI via SPIFI
- External Memory Ctrl (up to 32 bits)

Other
- Operating voltage: 1.71 to 3.6V
- Temperature range: -40 to 105 °C
- LQFP208, LQFP100
- TFBGA180, TFBGA100

Typical Application

Connected, HMI Control Panel/Edge Node in Industrial Applications

LPC546xx Family of MCUs *Combine a 180MHz Cortex-M4, for real-time performance, with its unique architecture for outstanding power-efficiency.*



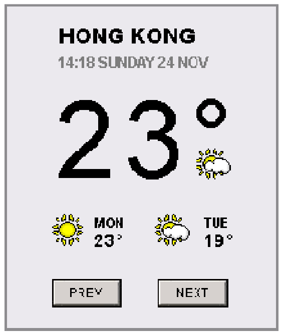
Key Features:

- 30 channel Direct Memory Access (DMA)
- Fast wake-up & mode transitions with 12 MHz Free Running Oscillator (FRO) trimmed to +/- 1% accuracy over voltage & temperature (selectable 48/96 MHz outputs)
- Code Security with Enhanced Code Read Protection (eCRP) and a 128 bit unique device serial number for identification
- Powerful, feature rich 32-bit timers, including State Configurable Timer (SCT/PWM)
- Flexible communication interface with up to 10x USARTS, I2C (supporting FM+) and SPI, along with up to two I2S
- Large availability of GPIOs (up to 171) with fast access (on AHB), DMA support of GPIO ports
- Ethernet with IEEE1588 PTP, Dual CAN supporting CAN-FD and CAN2.0
- FS & HS USB with integrated PHY
- Flexible wake-up & clock sources

LPC's Complete Offering of Graphics Solutions

Segger emWIN Provided Complementary with NXP's MCUXpresso SDK

- Offered as Library + API in C language, compatible with any RTOS (although not required)
- GUI tool builder available
- Source Code from Segger available with a per product license fee
- Free/no royalty required



Provider / Product	Type	Language	GUI Tool Builder	Business model	RTOS Required
TARA / Embedded Wizard	Source code generator	C Javascript	Yes	Developer seats Volume based product line license	Optional (any)
Draupner / TouchGFX	Library + API	C++	Yes	Free developer tools Volume based product line license	Recommended (any)
MicroEJ	Library + API	C/C++ Java	Yes	Part of MicroEJ platform Developer seat licenses Volume based licenses	Yes (MicroEJ)
expresslogic / GUIX	Library + API	C	Yes	Source code per product license	Yes (ThreadX)

Enablement Overview

Runtime Software

NXP Solutions:

MCUXpresso Software and Tools

- **IDE**
- **SDK**
- **Config Tools**

For NXP Cortex-M controllers
- Kinetis MCUs
- LPC Microcontrollers
- i.MX Application Processors

RTOS, Middleware Partners:

MICROEJ

Embedded Wizard

Comprehensive frameworks and solutions for low-power, connected, and secure embedded systems

Software Development Tools

IDE / Toolchains:

ARM KEIL Microcontroller Tools

IAR SYSTEMS

Rowley Associates

atollic

Industry leading IDE support and intuitive software configuration tools to accelerate application development

Hardware Development Tools

Evaluation Kits:

Partner Solutions

LAUTERBACH DEVELOPMENT TOOLS

Low cost hardware platforms for evaluation and application development. Partner solutions for hardware debugging solutions

Application Specific

- Graphics
- Cloud Connectivity
- Voice activation
- USB Audio
- Touch HMI
- Camera interface

Connectivity Solutions

Bluetooth SMART

Wi Fi

EtherNet

802.15.4

CAN

Software frameworks and development tools for targeted applications and certified connectivity solutions

Support

Broad Market:

- OOB Walkthroughs
- NXP Community
- Solution Designs
- Application Notes
- Schematics

High Touch:

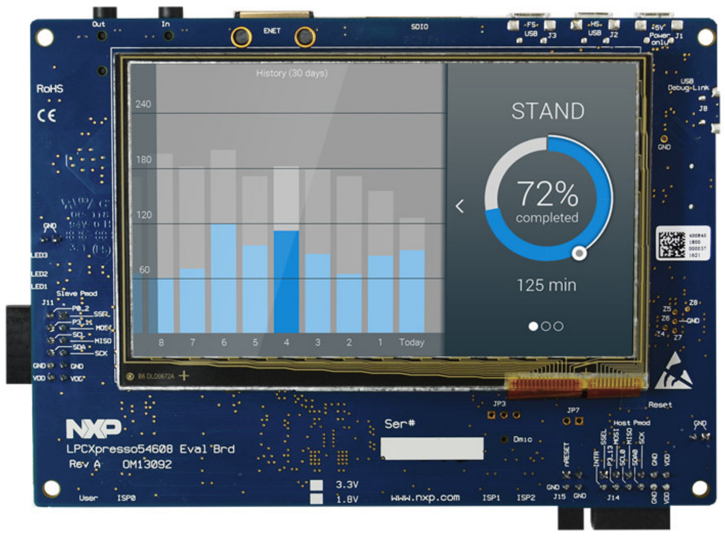
- Professional Support
- Professional Services

Get started quickly and get the support you need, when you need it

Enablement: LPC546xx Development Board

OM13092
Base Development Board
On-board Display
Available Now

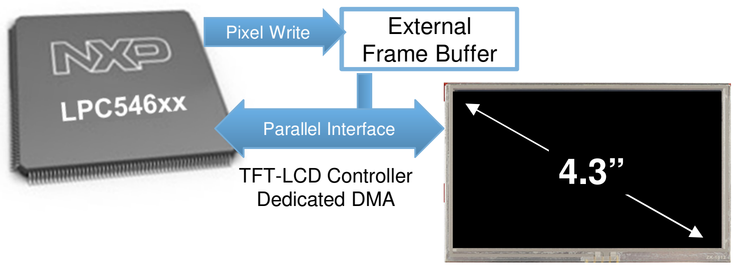
OM13094
CAN-FD Enabled Kit
CAN Physical Transceiver Shield
(no display)
Orderable March-2017



Additional (new) on-board features:
- 16MB Micron SDRAM (required for graphics)
- Ethernet (PHY, magnetics & connector)
- DMIC (Knowles Morello)
- I^{2}S connected CODEC with Line In/Out
- SD/MMC card (SDIO)
- Accelerometer on I^{2}C
- 16MB Micron QSPIFI with XIP

- **LPC54608 in BGA180 package**
  - Cortex-M4F@180MHz
- **Standard LPCXpresso features:**
  - Link2 OBD / external debug
  - Wake, ISP, Reset buttons
  - HS micro USB AB connector
  - FS micro USB AB connector
- **4.3” cap touch display (parallel interface)**
- **2 x PMod expansion connectors**
- **Expansion connectors**
  - Can support Arduino shields such as WiFi modules

Graphics – 24-bit LCD Interface Supports up to XGA



Features and Advantages

- Up to 1024x768 resolution
- 24-bit LCD interface supports 24bpp (16M colors)
- Palette table allows display of up to 256 of 64K colors
- Dedicated LCD DMA controller
- Hardware cursor support

Target Applications

- Thermostat
- Appliance/White Goods HMI
- Fitness equipment
- Industrial Panel

Enablement and Third Parties

- Free MCUXpresso IDE with SDK, configuration tools
- LPCXpresso54608 Development Board
- LCD App notes and Design recommendation
- Complementary Segger emWIN to develop GUI applications
- Additional GUI solutions from industry leading partners

LAB 4

Lab 4.1 LCD Basic

- **Pre-requiste:** Install the SDK(SDK2.2_LPCXpresso54608) from Thumb drive and import project from file system hot_LCD_1_tft16bpp.
- **Build, Debug and run** hot_LCD_1_tft16bpp.
- **Objective**：Understanding how to
  - Defines LCD parameters and use SDK APIs to initialize LCD controller, start LCD operation
  - Allocate framebuffer in SDRAM w/o having to initialize SDRAM before main()
  - Draw on framebuffer
- **Description**：Initializes LCD controller, SDRAM, and draw on framebuffer
- **Result:** 8 color stripes moving on LCD screen

Key API and codeHOT_LCD_1_tft16bpp

- Initialize SDRAM for framebuffer availability

BOARD_InitSDRAM();

- Enabled clock to LCD controller

CLOCK_SetClkDiv(**kCLOCK_DivLcdClk, 1, true**);

- Initialize LCD controller with specified parameters, including panel clock, resolution, color format, timings, framebuffer address.

LCDC_GetDefaultConfig(&lcdConfig);

lcdcInFreq = CLOCK_GetFreq(**kCLOCK_LCD**);

LCDC_Init(LCD, &lcdConfig, lcdcInFreq);

- Start LCD controller and power up LCD

LCDC_Start(LCD);

LCDC_PowerUp(LCD);

Observation and changes

- Change the color of stripes by modifying “colTab” array.
- Change panel clock frequency macro “LCD_PANEL_CLK”, suggested range no less than 4MHz.
- Learn the members of “lcdc_config_t”, how they map to LCD controller to registers.
- Check the “Flash.sct” and the definition of “**s_FB**” to see how to make framebuffer placed into SDRAM w/o involving compiler to generate “zero-init” code to framebuffer before main().
- Otherwise, Compiler will generate “zero-init” code to framebuffer before jumping to “main()”; however, w/o having SDRAM initialized, accessing SDRAM will cause hard fault.

Lab 4.2 DUAL FRAMEBUFFER

- **Pre-requiste:** Import project from file system hot_LCD_2_tft16bpp_2fb
- **Build, Debug and run** hot_LCD_2_tft16bpp_2fb.
- **Objective** : Understanding how to
  - Defines 2 framebuffers
  - Using LCD's "base address update" interrupt to safely draw to background FB.
  - Draw on framebuffer
- **Description** : Repeating drawings in main loop: first clear the screen to black, then draw color stripes. Use SysTick timer to limit draw rate.
  - If "SW5" is not pressed, then use one FB to draw,
  - if "SW5" is pressed, then waits for "base address update" IRQ, then draws on backup FB (the previous active FB), after drawing, set the next active FB to this FB.
- **Result:** If "SW5" is not pressed, then black screen and color stripes shows on screen interleaved, get flicker feeling; if "SW5" is pressed, only the rotating color stripes are shown (like HOT1).

Key API and code HOT_LCD_2_tft16bpp_fb

- Enable LCD “base address update” interrupt

LCDC_EnableInterrupts(LCD, **kLCDC_BaseAddrUpdateInterrupt);**

- IRQ handler: Get LCD interrupt flag and clear in LCD IRQ handler, set the s/w level notify ---- “s_frameAddrUpdated = true;”

void LCD_IRQHandler(void) {…}

- intStatus = **LCDC_GetEnabledInterruptsPendingStatus**(LCD);

LCDC_ClearInterruptsStatus(LCD, intStatus);

if (intStatus & **kLCDC_BaseAddrUpdateInterrupt**) {…}

- Update FB address after background FB drawing is done, and switch the active/background FB.

LCDC_SetPanelAddr(LCD, **kLCDC_UpperPanel,** (uint32_t)(pFB32)); s_actFBNdx = !s_actFBNdx;

- Background code: Wait for “s_frameAddrUpdated” to become true before drawing next frame.

while (!s_frameAddrUpdated){}

Observation with changes

- See different drawing effects when “SW5” is pressed and not pressed.
- Enter debug mode，press ”F10” to step over or “F11” to step into to analyze and check how the FBs are switched with “s_actFBNdx” variable.
- Experiments :
  - Change SysTick rate, check if it can resolve the flicker effect w/o dual-FB, and/or affects dual-FB effect.
  - Comment out the “while (!s_frameAddrUpdated){}”, see if it affects dual-FB effect.
  - Switch “stage1” and “stage2” in code, check the differences of LCD display for single FB and dual-FB respectively.

Lab 4.3 PALETTE

- **Pre-requiste:** Import project from file system hot_LCD_3_palette
- **Build, Debug and run** hot_LCD_3_palette

- **Objective** : Understanding how to
  - use palette to put framebuffer in SRAM, instead of SDRAM
  - Palette color settings

- **Description** : Draw moving rectangle periodically. Every period is synchronized to a new LCD base address update IRQ. The examples implements a rectangle draw & fill routine with 2bpp mode.

- **Result:** There is a rectangle moving smoothly and when reach a edge (either left, top, right ,bottom), it changes color and bounces.

Key API and code HOT_LCD_3_palette

- Setup palette colors

  static const uint32_t palette[] = {0x001F0000U, 0x7C0003E0U};

- Set palette buffer

  LCDC_SetPalette(APP_LCD, palette, ARRAY_SIZE(palette));

- Update FB address after background FB drawing is done, and switch the active/background FB.

  LCDC_SetPanelAddr(LCD, **kLCDC_UpperPanel,** (uint32_t)(pFB32));

  s_actFBNdx = !s_actFBNdx;

Observation with changes

- Locate palette color settings and change color to see the result. Color is RGB565 format.
- Change the moving speed of rectangle, either X or Y.

Lab 4.4 H/W CURSOR

- **Pre-requiste:** Import project from file system hot_LCD_4_cursor
- **Build, Debug and run** hot_LCD_4_cursor
- **Objective**：Understanding how to
  - Understands how to setup cursor bitmap, including transparent and XOR colors
  - Set new position of cursor synchronized with LCD vertical back porch.
- **Description**：Draw and moves cursor periodically. Every period is synchronized to a new LCD vertical back porch IRQ.
- **Result:** There is a cursor moving smoothly and when reach a edge (either left, top, right ,bottom), it bounces.

Key API and code HOT_LCD_4_cursor

- Defines the bitmap (w/ transparency and XOR “colors”) of cursor

  static const **uint8_t cursor32Img0[]**

- Configure cursor

  lcdc_config_t lcdConfig**;**

  LCDC_CursorGetDefaultConfig(&cursorConfig);

  cursorConfig.**size** = **kLCDC_CursorSize32**;

  cursorConfig.**syncMode** = **kLCDC_CursorSync**;

  cursorConfig.**image**[0] = (uint32_t *)**cursor32Img0**;

- Select cursor image number (0 to 3). LCDC supports 64x64, divided into 4 32x32 images like a “田”. As we just setup one left-top 32x32, the number is 0.

  LCDC_ChooseCSeursor(APP_LCD, 0)**;**

- Cursor update is synchronized to vertical back porch, so enabled the “vertical compare” IRQ and select vertical back porch as IRQ trigger source.

  LCDC_SetVerticalInterruptMode(APP_LCD, **kLCDC_StartOfBackPorch**);

  LCDC_EnableInterrupts(APP_LCD, **kLCDC_VerticalCompareInterrupt**);

  NVIC_EnableIRQ(**APP_LCD_IRQn**);

- Enable H/W cursor layer

  LCDC_EnableCursor(APP_LCD, true);

- Update cursor location after a new vertical back porch IRQ fired.

  LCDC_SetCursorPosition(APP_LCD, cursorPosX, cursorPosY);

Observation with changes

- cursor position update synchronized to vertical back porch, to verify if this is required,
- comment out the “while (!s_frameEndFlag){}” and uncomment “while (!s_isNewTick){}” to see the change of cursor movement (Which one is more smoother?).

EMWIN DEMO

Emwin

- **Pre-requiste:** Import project from file system Emwin_code
- **Build, Debug and run** Emwin_2_demo
- **Objective** : Segger's EmWin provides a complimentary way to develop your next GUI application

http://www.nxp.com/pages/emwin-graphics-library:EMWIN-GRAPHICS-LIBRARY

SECURE CONNECTIONS
FOR A SMARTER WORLD