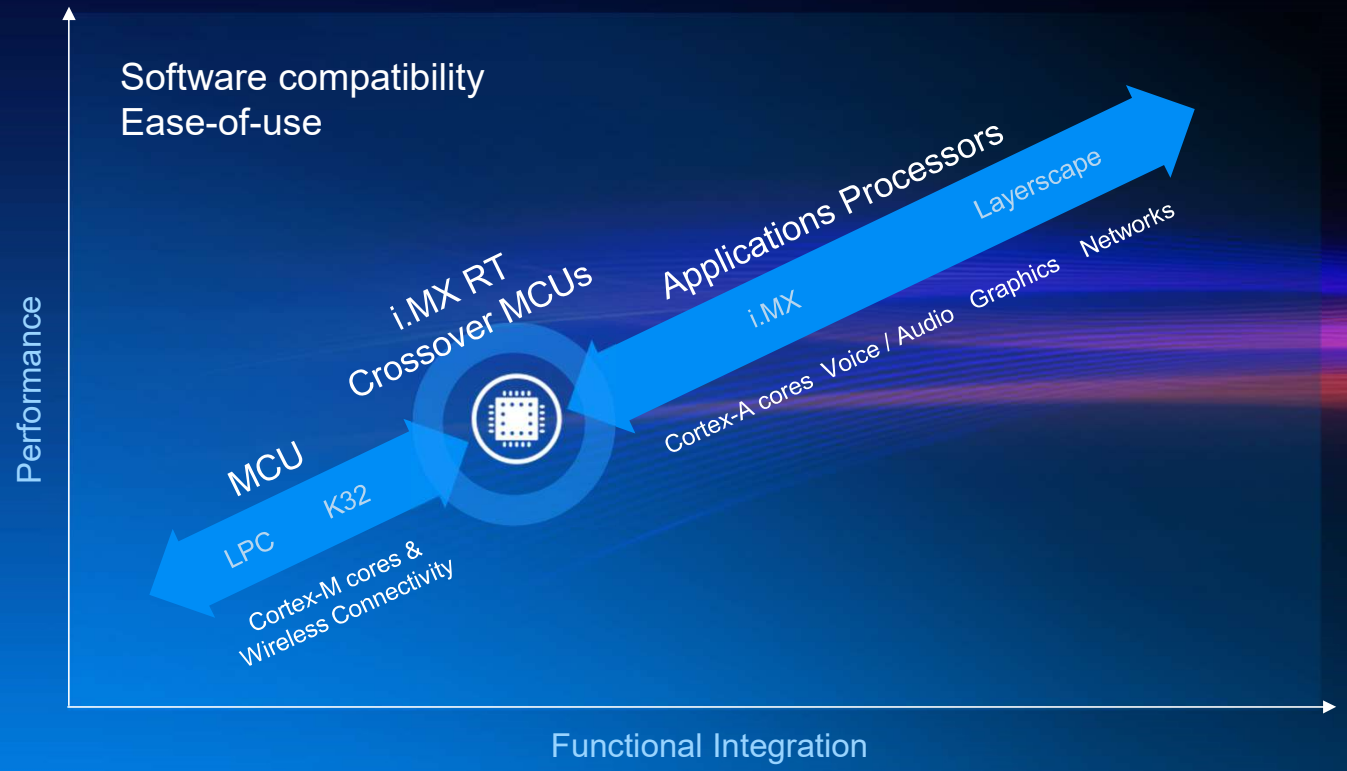
Crossover Code Challenge: Getting Started with the i.MX RT1010 EVK and MCUXpresso

Juan Mendoza, Systems Engineering Manager, NXP
Clark Jarvis, MCUXpresso Product Manager, NXP

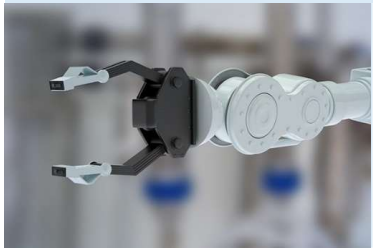
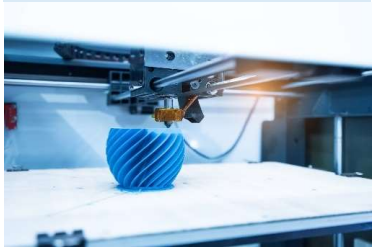
December 10, 2019

NXP

NXP Scalable Edge Processing Continuum

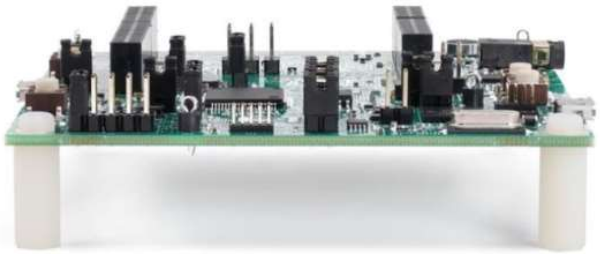


i.MX RT Crossover MCUs – Target Applications

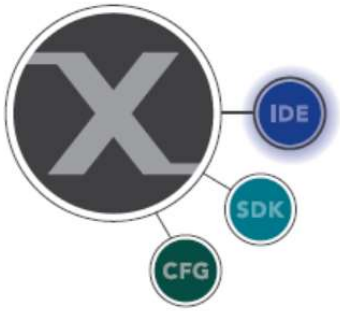
Voice & Audio	Consumer & Healthcare	Home & Building Automation	Industrial Computing	Motor Control & Power Conversion
High-end, consumer audio devices, including specialty equipment such as: Home Audio Professional microphone Guitar pedals Audio Tuners	Wearables Smart appliances TVs Game Controllers Mobile patient care, Blood pressure monitor Activity & wellness monitor Exercise equipment with display	HVAC climate control Security Lighting control panels IoT gateways Smoke Detectors Thermostats	EBS PLCs Factory automation Test and measurement HMI control assembly line robotics QR Readers Barcode Scanners	3D printers Thermal printers Unmanned autonomous vehicles Robotic vacuum cleaners Drones
				

hackster.io
AN AVNET COMMUNITY

Objective: Develop example software code for multiple features of the i.MX RT1010 crossover MCU.



+

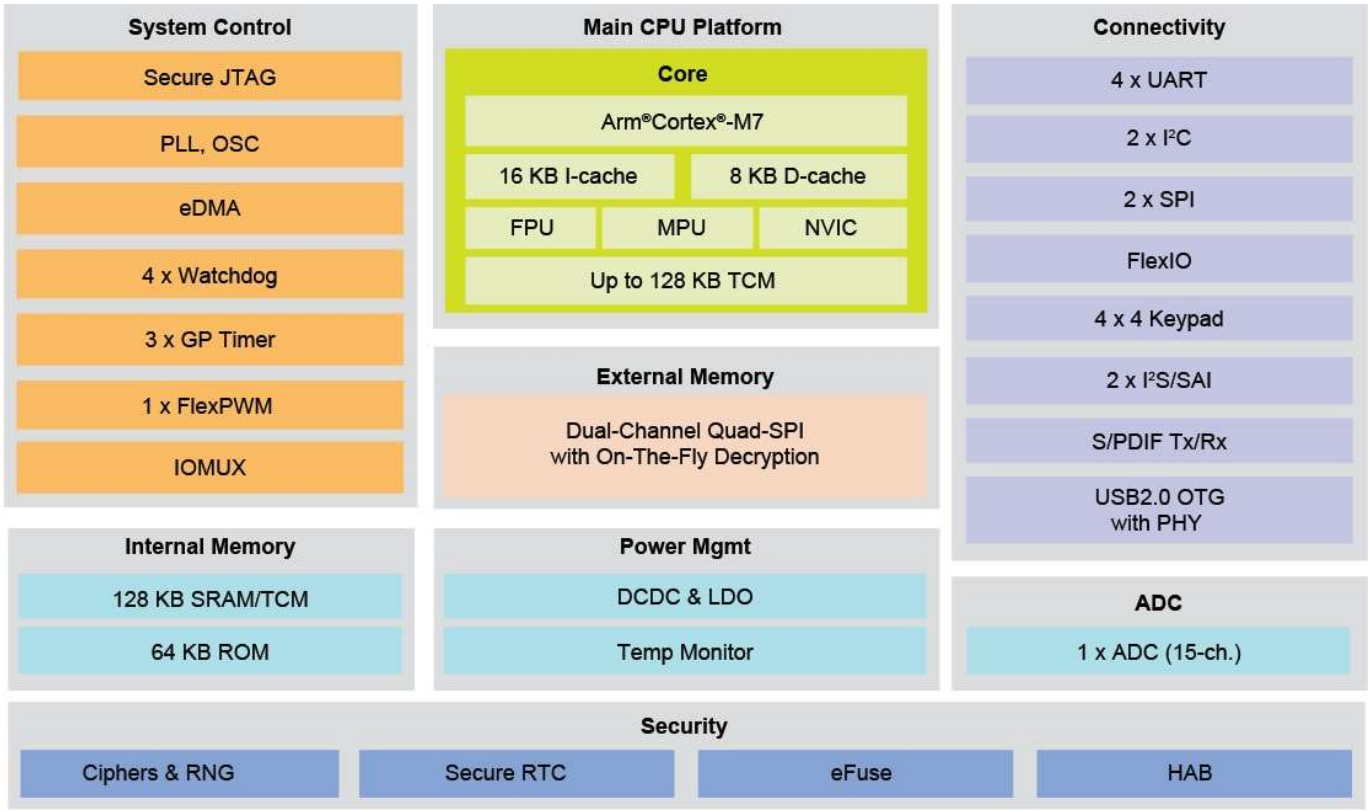


Features of the i.MX RT1010 Crossover MCU and i.MX RT1010 EVK

Juan Mendoza, Systems Engineering Manager, NXP

i.MX RT1010 SoC

- Features
  - 500Mhz Cortex M7
  - 128K SRAM
  - FlexIO (Displays)
  - Security
  - Rich Audio Features
  - Motor Control



i.MX RT1010 Evaluation Kit (MIMXRT1010-EVK)

FEATURES

Memory
- 128Mb QSPI Flash

Audio
- Wolfson Stereo Codec
- 3.5mm headphone jack
- Microphone
- L/R Speaker out connections
- S/PDIF (coax, not populated)

Connectivity
- HS USB
- Arduino® interface
- Motor Control Expansion port

Software
- MCUXpresso SDK with optional Amazon FreeRTOS support
- MCUXpresso IDE
- MCUXpresso Configuration Tools
- Arm Mbed™ OS
- Zephyr® OS

RESOURCES

- **SoC reference material**
  - IMXRT1010RM
  - Documentation Application-Notes
- **Users Guides**
  - Quick Start Guide – i.MX RT1010 Evaluation Kit
  - MIMXRT1010 Evaluation Kit Hardware User's Guide
  - Getting Started with i.MX RT1010 Evaluation Kit
- **Design Resources** – i.MX RT1010 EVK Design Files, Schematics

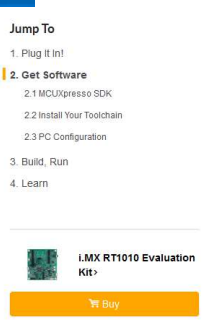
How To Get Started with MCUXpresso Software and Tools on i.MX RT1010 EVK

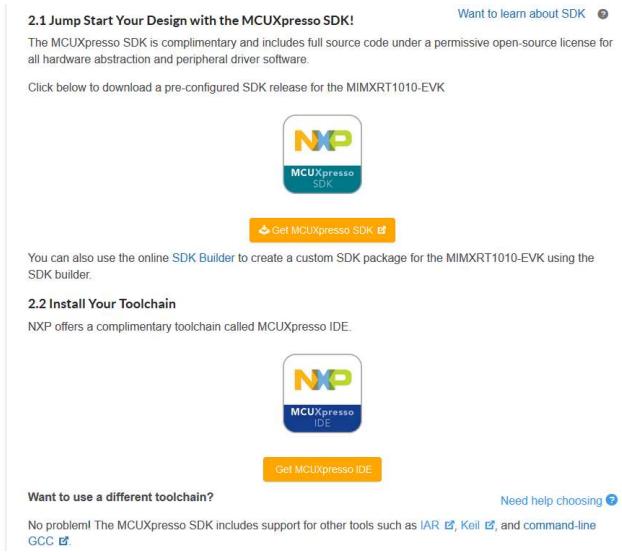
Clark Jarvis, MCUXpresso Product Manager, NXP

MCUXpresso SW and Tools

Getting Started Guide for i.MX RT1010

https://www.nxp.com/MIMXRT1010-EVK





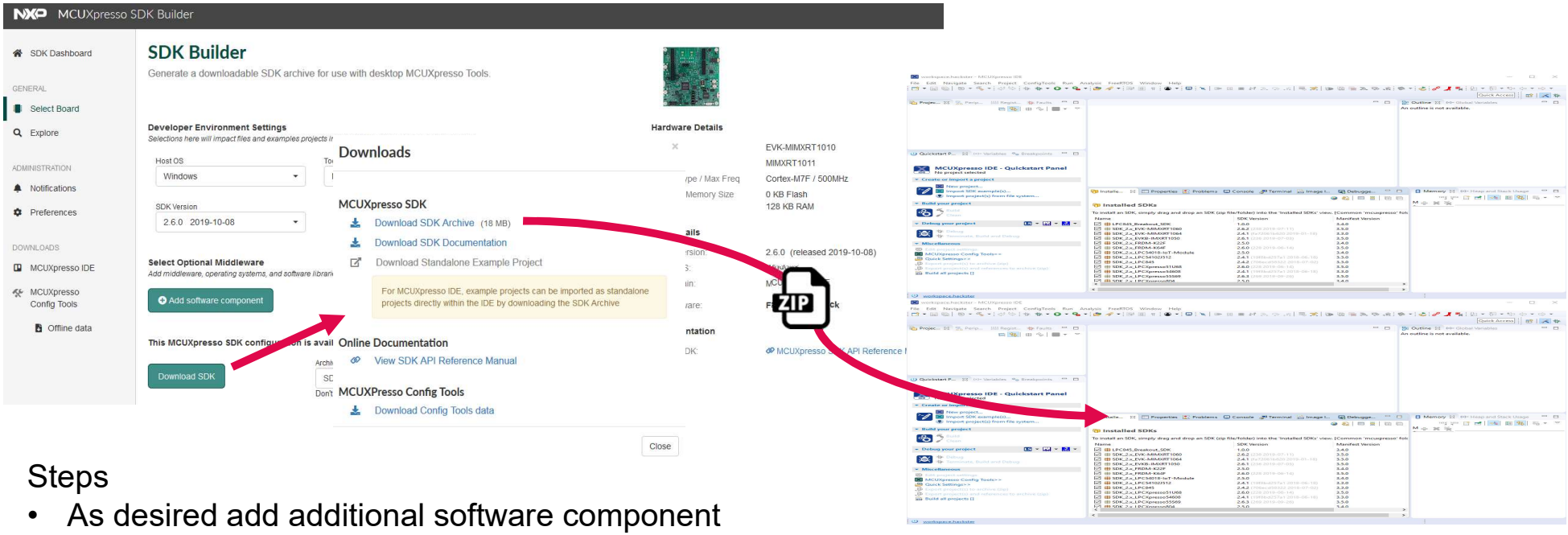
Includes:

- Getting started videos
- Links to MCUXpresso IDE and MCUXpresso SDK
- Basic examples walk-through

MCUXpresso SW and Tools

Download and Install SDK for i.MX RT1010

https://mcuxpresso.nxp.com/en/select?device=EVK-MIMXRT1010



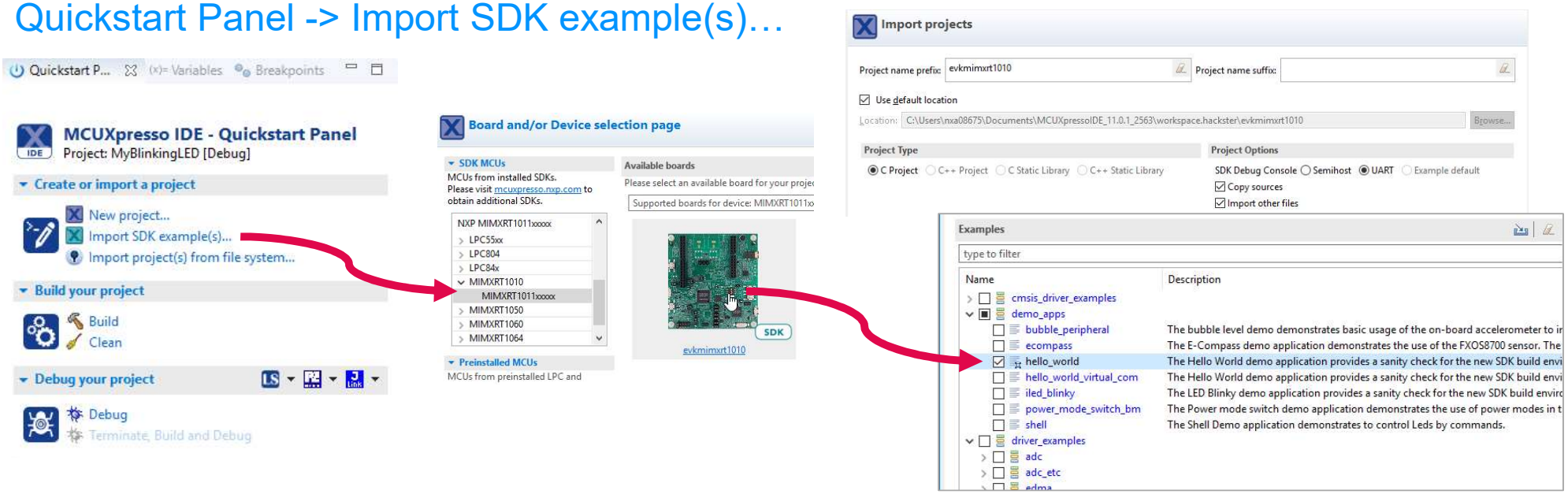
Steps

- As desired add additional software component
- Download SDK and Documentation
- Drag-and-Drop downloaded archive into IDE

MCUXpresso SW and Tools

Import example application

Quickstart Panel -> Import SDK example(s)…

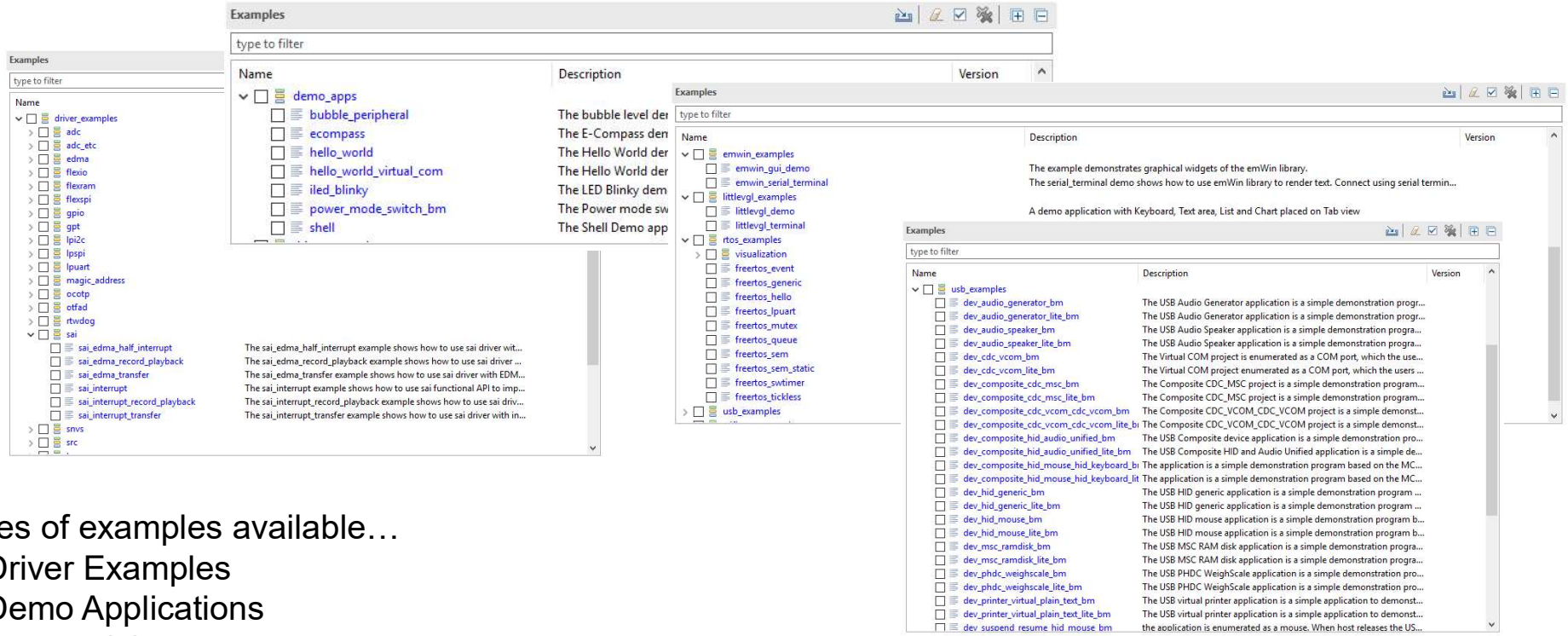


Steps

- Import (copy) an existing project based on MIMXRT1010-EVK
- Build and Debug basic “Hello World” application
- Step-though application and observed console output

MCUXpresso SW and Tools

Library of example application



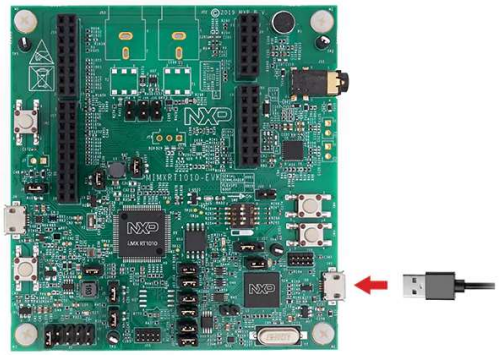
Types of examples available…

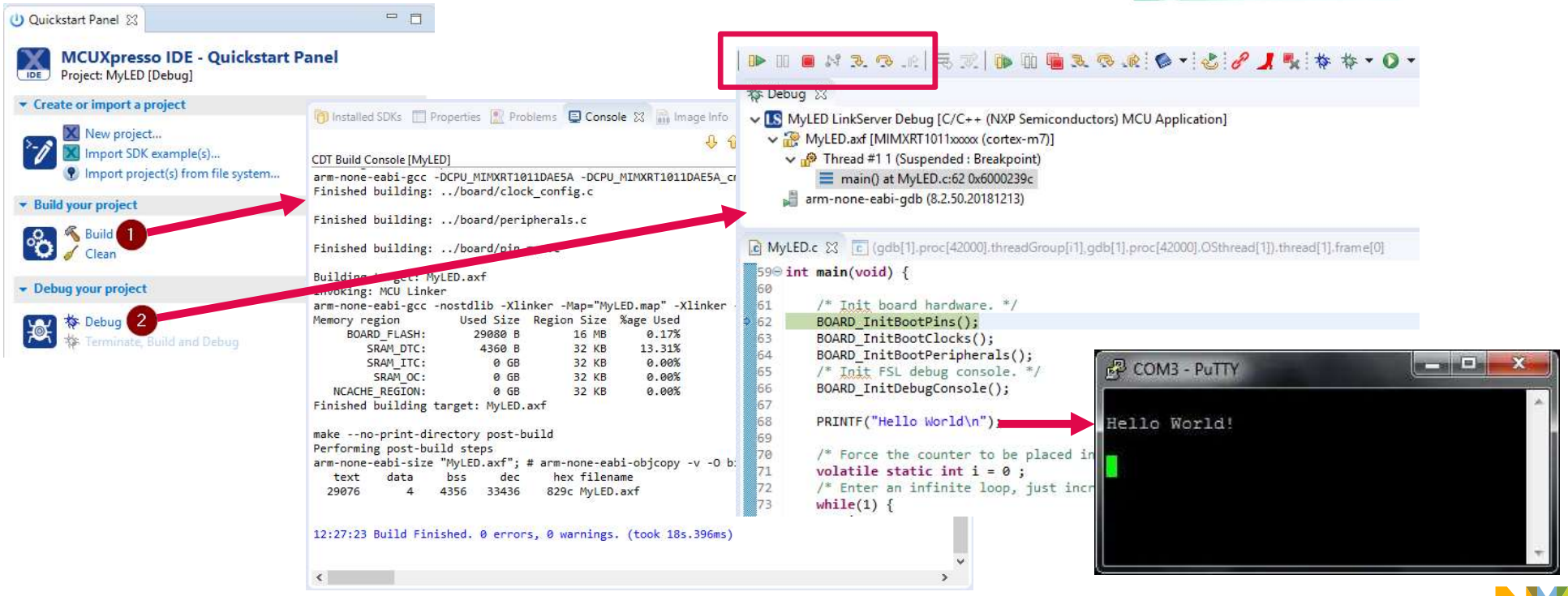
- Driver Examples
- Demo Applications
- FreeRTOS Examples
- Software Specific (Graphics, Wi-Fi, USB)

MCUXpresso SW and Tools

Build / Debug Example Application in IDE

Flash and Debug Application on Board

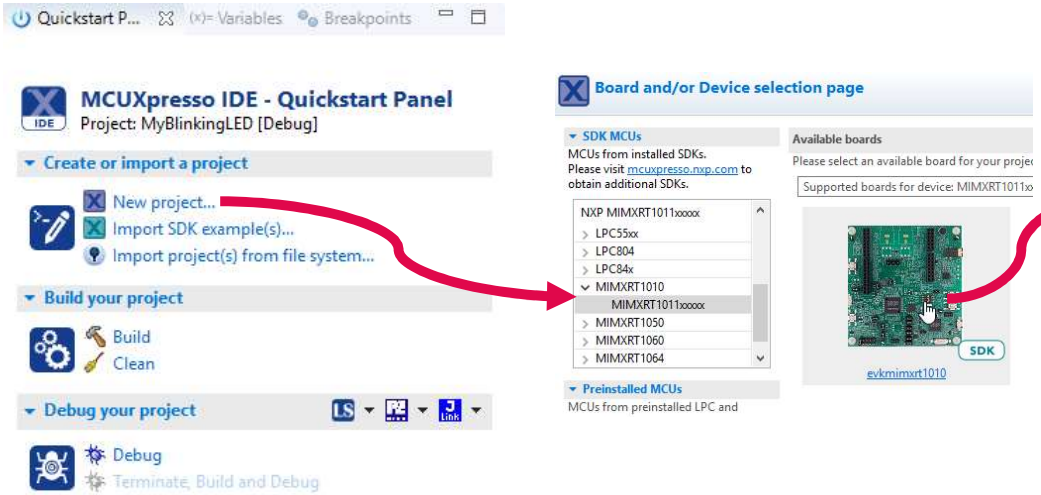


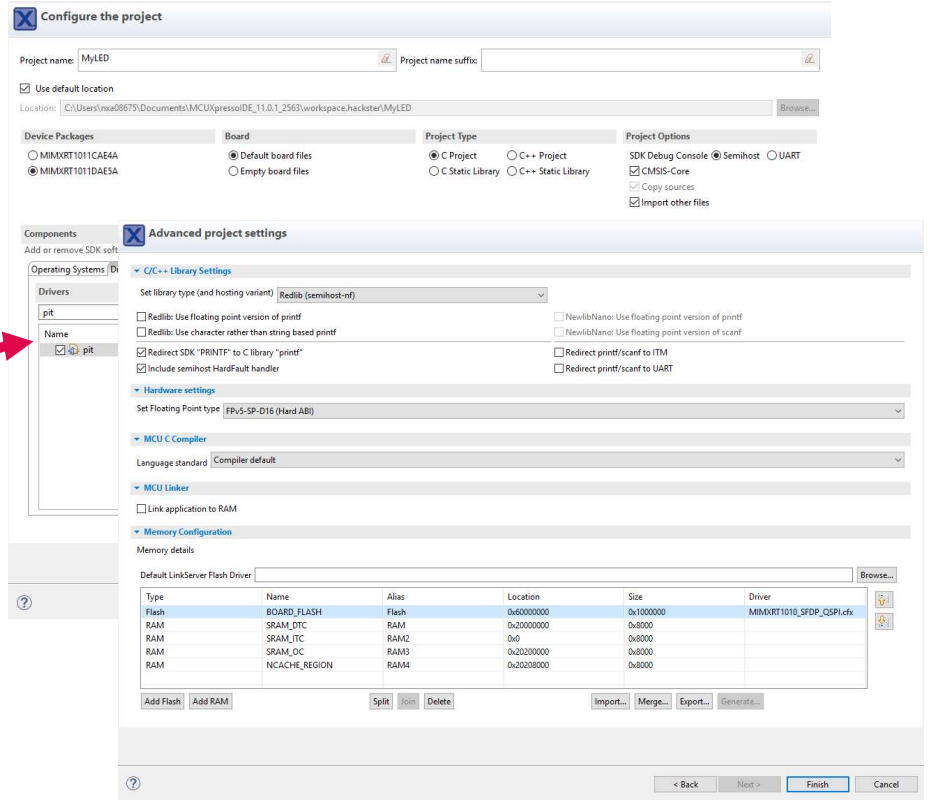


MCUXpresso SW and Tools

Create new IDE project

Quickstart Panel -> New Project…





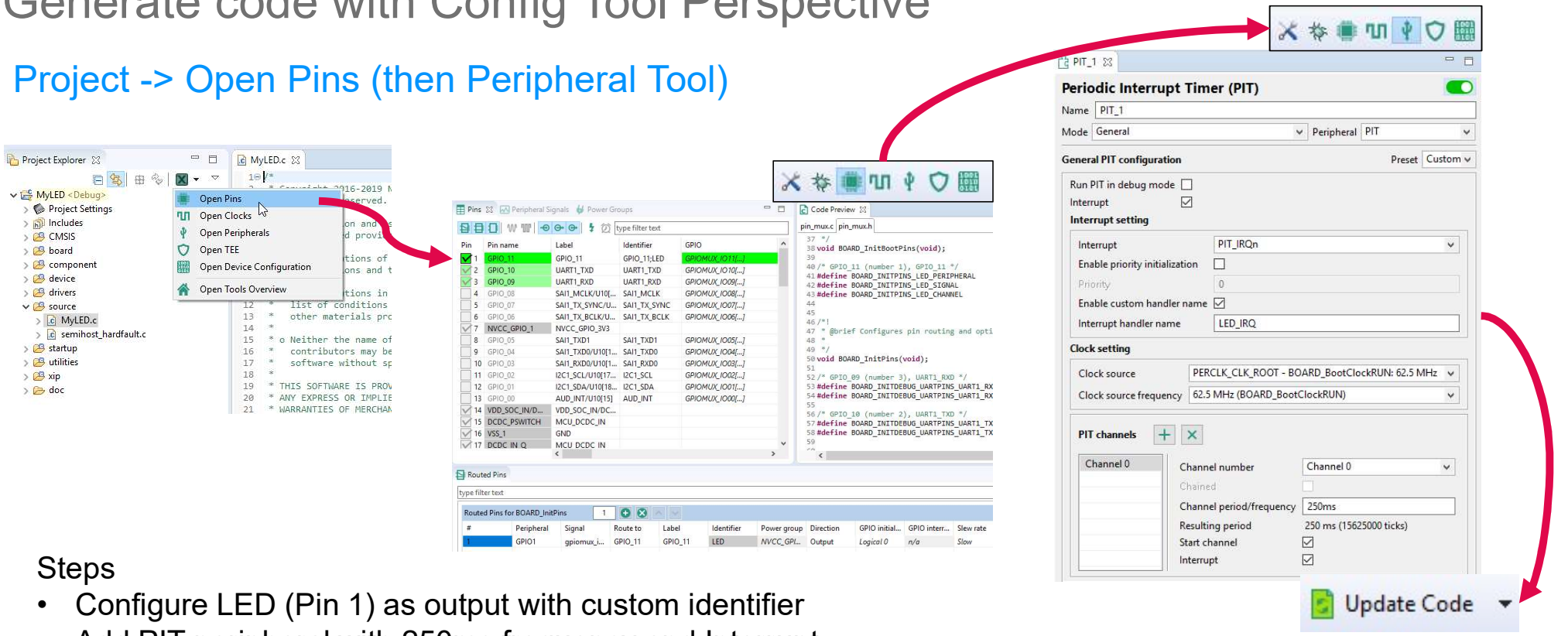
Steps

- Create a new project based on MIMXRT1010-EVK
- Add SDK driver for Periodic Interrupt Timer (PIT)
- Review new project settings

MCUXpresso SW and Tools

Generate code with Config Tool Perspective

Project -> Open Pins (then Peripheral Tool)



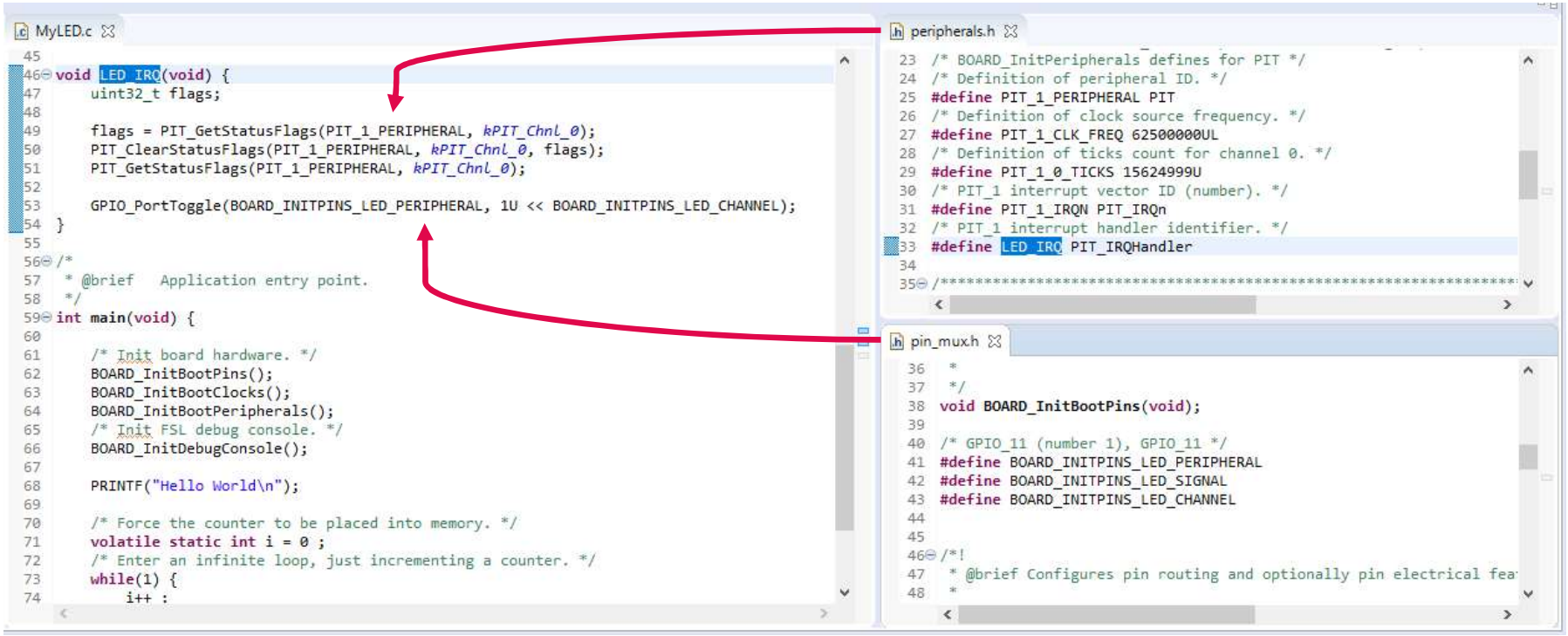
Steps

- Configure LED (Pin 1) as output with custom identifier
- Add PIT peripheral with 250ms frequency and Interrupt
- “Update Code” to generate configuration files and return to Development Perspective

MCUXpresso SW and Tools

Create an interrupt function for the LED

Create IRQ function to Blink LED



MyLED.c

```
45
46 void LED_IRQ(void) {
47     uint32_t flags;
48
49     flags = PIT_GetStatusFlags(PIT_1_PERIPHERAL, kPIT_Chnl_0);
50     PIT_ClearStatusFlags(PIT_1_PERIPHERAL, kPIT_Chnl_0, flags);
51     PIT_GetStatusFlags(PIT_1_PERIPHERAL, kPIT_Chnl_0);
52
53     GPIO_PortToggle(BOARD_INITPINS_LED_PERIPHERAL, 1U << BOARD_INITPINS_LED_CHANNEL);
54 }
55
56 /*
57  * @brief   Application entry point.
58  */
59 int main(void) {
60
61     /* Init board hardware. */
62     BOARD_InitBootPins();
63     BOARD_InitBootClocks();
64     BOARD_InitBootPeripherals();
65     /* Init FSL debug console. */
66     BOARD_InitDebugConsole();
67
68     PRINTF("Hello World\n");
69
70     /* Force the counter to be placed into memory. */
71     volatile static int i = 0 ;
72     /* Enter an infinite loop, just incrementing a counter. */
73     while(1) {
74         i++ ;
```

peripherals.h

```
23 /* BOARD_InitPeripherals defines for PIT */
24 /* Definition of peripheral ID. */
25 #define PIT_1_PERIPHERAL PIT
26 /* Definition of clock source frequency. */
27 #define PIT_1_CLK_FREQ 62500000UL
28 /* Definition of ticks count for channel 0. */
29 #define PIT_1_0_TICKS 15624999U
30 /* PIT_1 interrupt vector ID (number). */
31 #define PIT_1_IRQN PIT_IRQn
32 /* PIT_1 interrupt handler identifier. */
33 #define LED_IRQ PIT_IRQHandler
34
35 /*****************************************************************
```

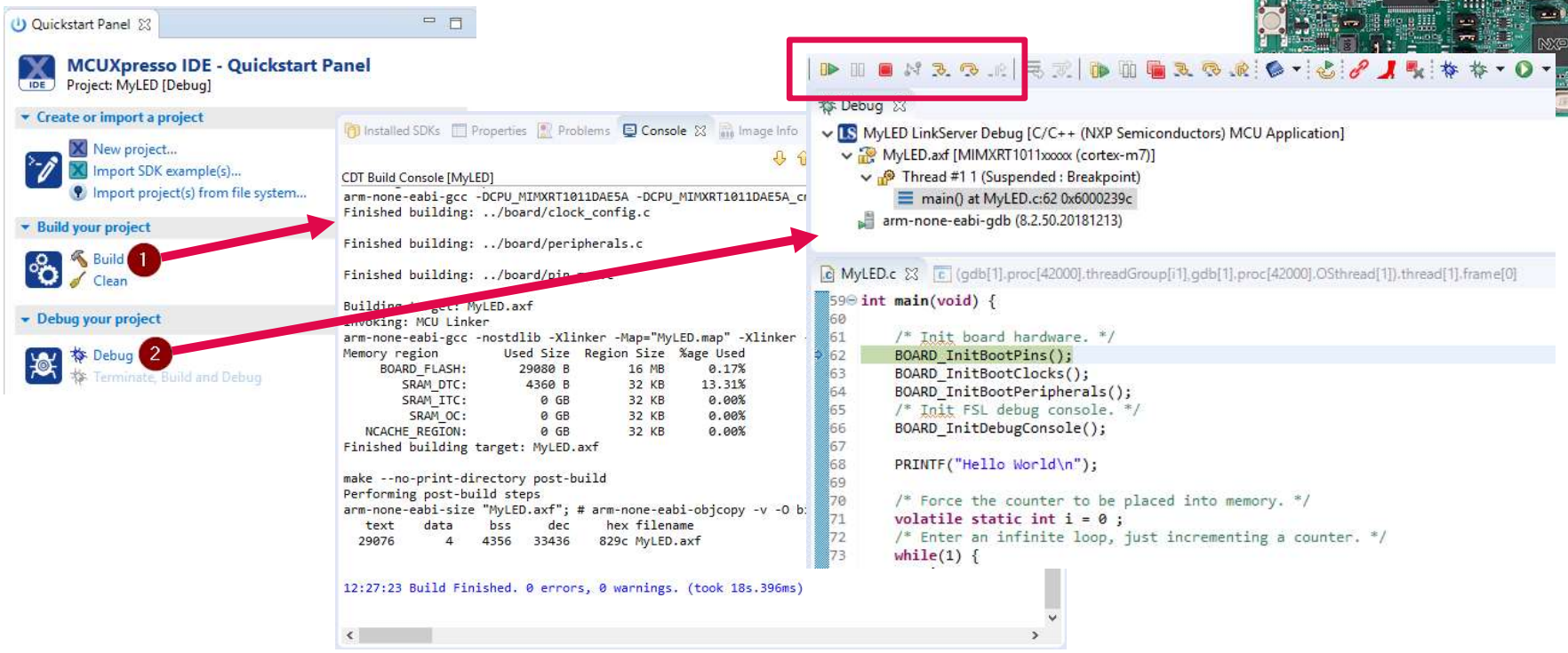
pin_mux.h

```
36  *
37  */
38 void BOARD_InitBootPins(void);
39
40 /* GPIO_11 (number 1), GPIO_11 */
41 #define BOARD_INITPINS_LED_PERIPHERAL
42 #define BOARD_INITPINS_LED_SIGNAL
43 #define BOARD_INITPINS_LED_CHANNEL
44
45
46 /*!
47  * @brief Configures pin routing and optionally pin electrical fea
48  *
```

MCUXpresso SW and Tools

Build / Debug MyLED Application in IDE

Flash and Debug Application on Board



MCUXpresso Software and Tools
Additional Resources

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
- MCUXpresso SDK Builder – https://mcuxpresso.nxp.com

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

Supported Devices

- Supported Devices Table (Community Doc)

Get Started Now

- **Pro Tips:** use the examples project provided with the SDK:
  - **Device examples** are excellent tutorial projects for how to use the respective peripheral for a basic use case.
  - **Demo Applications** provide examples of multiple drivers, middleware, and other available code packages working together.
  - Other **software specific examples** provide basic use cases of a particular software package that is included directly within the SDK as an enabling software technology.

- Register for the contest - https://www.hackster.io/contests/nxpcrossover

- How to post to GitHub (general guidelines) - https://help.github.com/en/github/getting-started-with-github/create-a-repo

- Get technical help –
  - https://www.hackster.io/contests/nxpcrossover
  - https://community.nxp.com/community/imxrt

Questions?


NXP
SECURE CONNECTIONS
FOR A SMARTER WORLD