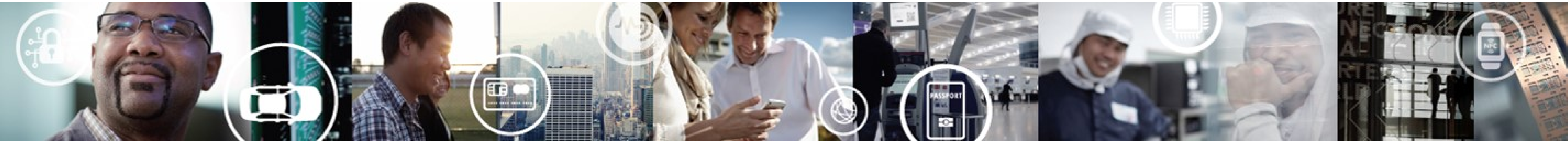
S32K116 EVB

QUICK START GUIDE

APPLIES FOR: S32K116 EVB (SCH_30003 REV B)



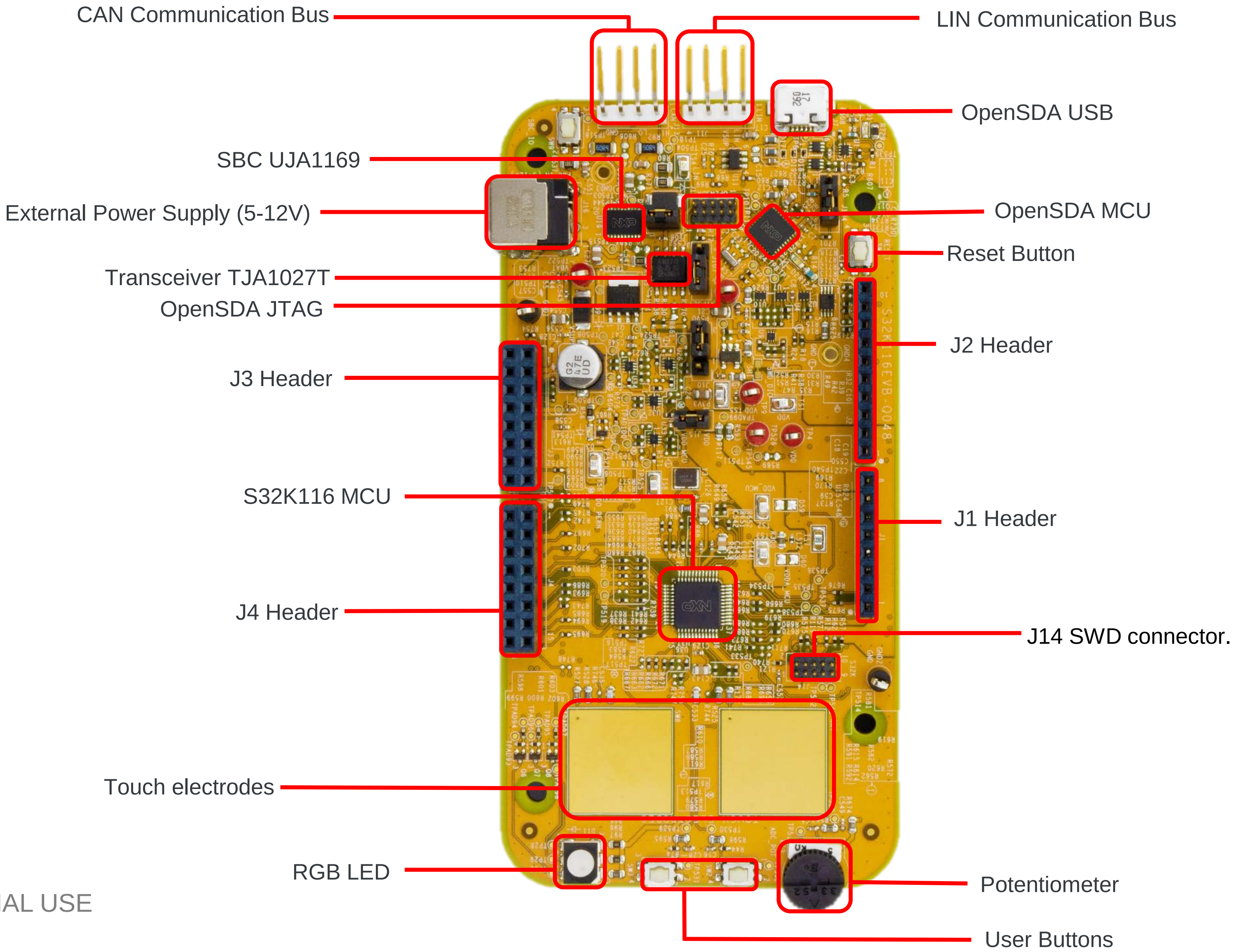
EXTERNAL USE

SECURE CONNECTIONS
FOR A SMARTER WORLD

Contents:

- Get to Know S32K116 EVB
- Out of the Box Setup
- Introduction to OpenSDA
- Creating a new S32DS project for S32K116:
  - Download
  - Create a project
  - Create a project from SDK example
- S32DS Debug basics
- Create a P&E debug configuration

Get to know S32K116-EVB



CAN Communication Bus

LIN Communication Bus

OpenSDA USB

SBC UJA1169

External Power Supply (5-12V)

OpenSDA MCU

Reset Button

Transceiver TJA1027T

OpenSDA JTAG

J2 Header

J3 Header

S32K116 MCU

J1 Header

J14 SWD connector.

J4 Header

Touch electrodes

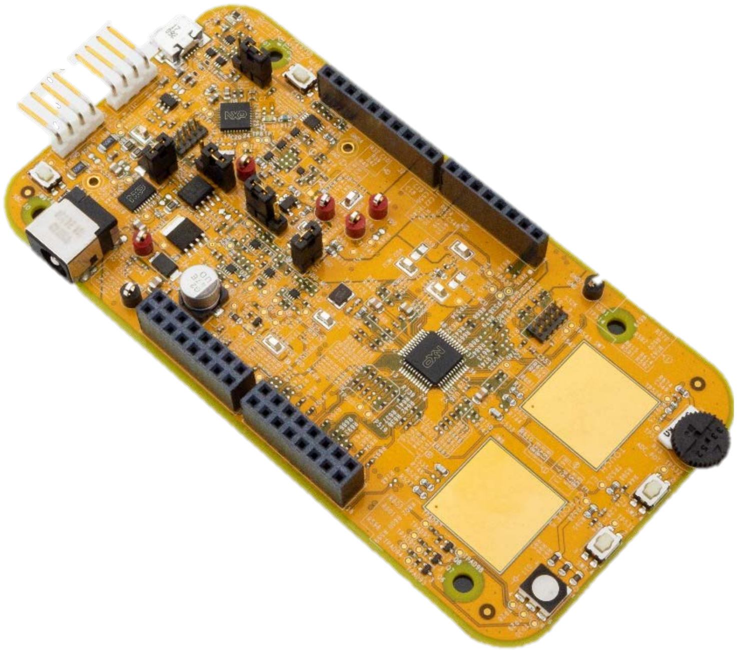
RGB LED

Potentiometer

User Buttons

S32K116 EVB Features:

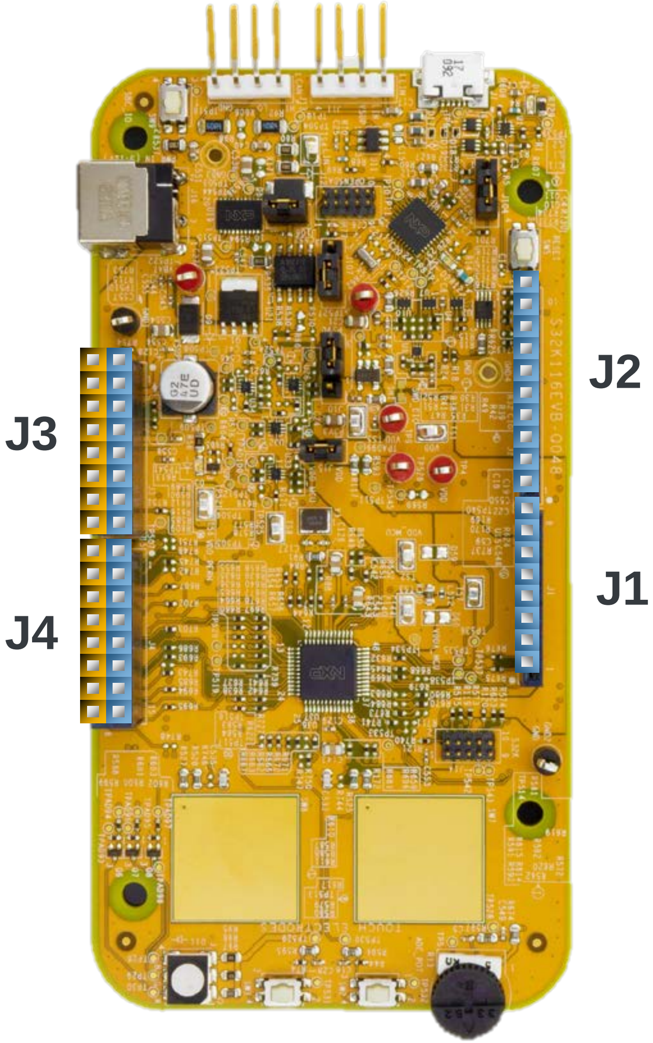
- Supports **48LQFP and 32QFN** packages
- Small form factor size 4.5” x 2.3”
- Arduino™ UNO footprint-compatible with expansion “shield” support
- Integrated open-standard serial and debug adapter (OpenSDA) with support for several industry-standard debug interfaces
- Easy access to the MCU I/O header pins for prototyping
- On-chip connectivity for CAN, LIN, UART/SCI.
- SBC UJA1169 and LIN phy TJA1027
- Potentiometer for precise voltage and analog measurement
- RGB LED
- Two push-button switches (SW2 and SW3) and two touch electrodes
- Flexible power supply options
  - microUSB
  - external 12V power supply



Header/Pinout Mapping for S32K116

PIN	PORT	FUNCTION	J3	PIN	PORT	FUNCTION
J3-02	PTB0	**GPIO**		J3-01	-	**VBAT**
J3-04	PTB1	**GPIO**		J3-03	-	**VDD_PERH**
J3-06	PTB6	**GPIO**		J3-05	PTA5	**RESET**
J3-08	PTB7	**GPIO**		J3-07	-	**3.3V**
J3-10	PTE4	**GPIO**		J3-09	-	**5V**
J3-12	PTE5	**GPIO**		J3-11	-	**GND**
J3-14	PTA11	**GPIO**		J3-13	-	**GND**
J3-16	PTD3	**GPIO**		J3-15	-	**VBAT**

PIN	PORT	FUNCTION	J4	PIN	PORT	FUNCTION
J4-02	PTC6	**GPIO**		J4-01	PTA7	**ADC0_SE3**
J4-04	PTC7	**GPIO**		J4-03	PTC8	**GPIO**
J4-06	PTC8	**GPIO**		J4-05	PTC1	**ADC0_SE9**
J4-08	PTC9	**GPIO**		J4-07	PTC14	**ADC0_SE12**
J4-10	PTD5	**GPIO**		J4-09	PTC15	**ADC0_SE13**
J4-12	PTD15	**GPIO**		J4-11	PTC16	**ADC0_SE14**
J4-14	PTD16	**GPIO**		J4-13	PTC9	**GPIO**
J4-16	PTE8	**GPIO**		J4-15	PTB13	**GPIO**



J2	PIN	PORT	FUNCTION
	J2-01	PTC2	**FTM0_CH2**
	J2-02	PTC3	**FTM0_CH3**
	J2-03	PTB5	**LPSPI0_PCS**
	J2-04	PTB4	**LPSPI0_SOUT**
	J2-05	PTB3	**LPSPI0_SIN**
	J2-06	PTB2	**LPSPI0_SCK**
	J2-07	-	**GND**
	J2-08	-	**AREF**
	J2-09	PTA1	**LPI2C0_SDA**
	J2-10	PTA0	**LPI2C0_SCL**

J1	PIN	PORT	FUNCTION
	J1-01	PTA2	**LPUART0_RX**
	J1-02	PTA3	**LPUART0_TX**
	J1-03	PTA13	**FTM1_CH7**
	J1-04	PTA12	**FTM1_CH6**
	J1-05	PTD0	**FTM0_CH2**
	J1-06	PTD1	**FTM0_CH3**
	J1-07	PTD2	**FXIO_D4**
	J1-08	PTE9	**FTM0_CH7**

🟦 Arduino compatible pins
🟧 NXP pins

*0ohm resistor is not connected

Jumper Settings

Jumper	Configuration	Description
J104	1-2	Reset signal to OpenSDA, use to enter into OpenSDA Bootloader mode
	2-3 (Default)	Reset signal direct to the MCU, use to reset S32K116.
J107	1-2 (Default)	S32K116 powered by 12V power source.
	2-3	S32K116 powered by USB micro connector.
J10	2-3 (Default)	MCU voltage 5v
	1-2	MCU voltage 3.3v
J108	1-2 (Default)	Select LIN master option
J15	1-2 (Default)	Used for current measurement

HMI mapping

Component	S32K116
Red LED	PTD16 (FTM0 CH1)
Blue LED	PTE8(FTM0 CH6)
Green LED	PTD15(FTM0 CH0)
Potentiometer	PTA7 (ADC0_SE3)
SW2	PTD3
SW3	PTD5
OpenSDA UART TX	PTB1(LPUART0_TX)
OpenSDA UART RX	PTB0(LPUART0_RX)
CAN TX	PTE5(CAN0_TX)
CAN RX	PTE4 (CAN0_RX)
LIN TX	PTC7(LPUART1_TX)
LIN RX	PTC6 (LPUART1_RX)
SBC_SCK	PTB2 (LPSPI0_SCK)
SBC_MISO	PTB3(LPSPI0_SIN)
SBC_MOSI	PTB4(LPSPI0_SOUT)
SBC_CS	PTB5(LPSPI0_PCS1)

S32K116 EVB
OUT OF THE BOX

Step 1: Power up the Board – EVB Power Supplies

- The S32K116-EVB evaluation board powers from a USB or external 12V power supply. By default 12V power is enabled with J107 (check slide 5)

- Connect the USB cable to a PC using supplied USB cable .

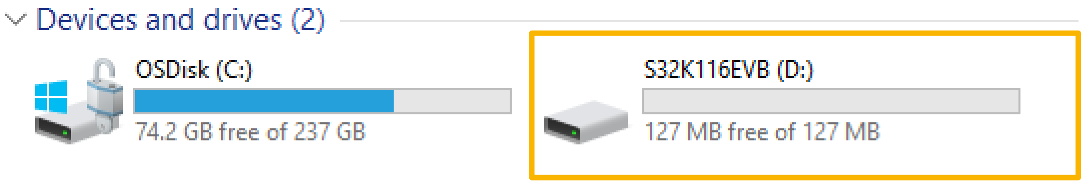
- Connect other end of USB cable (microUSB) to mini-B port on S32K116-EVB at J7

- Allow the PC to automatically configure the USB drivers if needed

- Debug is done using OpenSDA through J7

Step 2: Power up the Board – Is it powered on correctly?

- When powered through USB, LEDs D2 and D3 should light green
- Once the board is recognized, it should appear as a mass storage device in your PC with the name S32K116EVB.



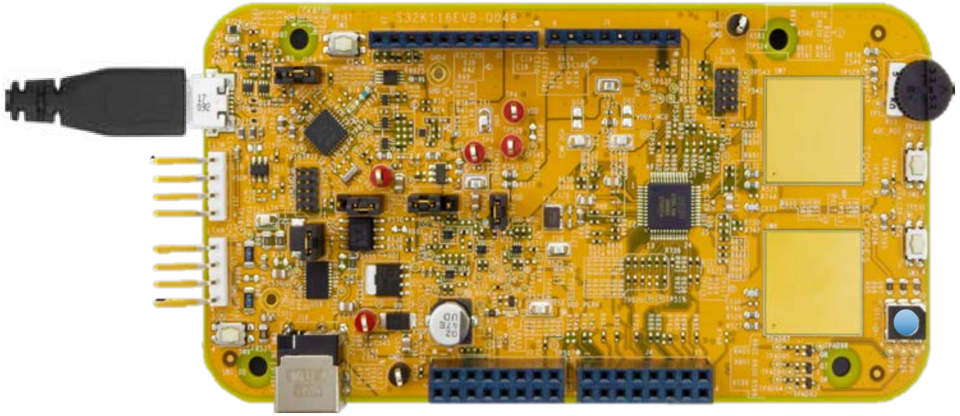
Devices and drives (2)

OSDisk (C:)
74.2 GB free of 237 GB

S32K116EVB (D:)
127 MB free of 127 MB

Step 3: Power up the Board – Is it powered on correctly?

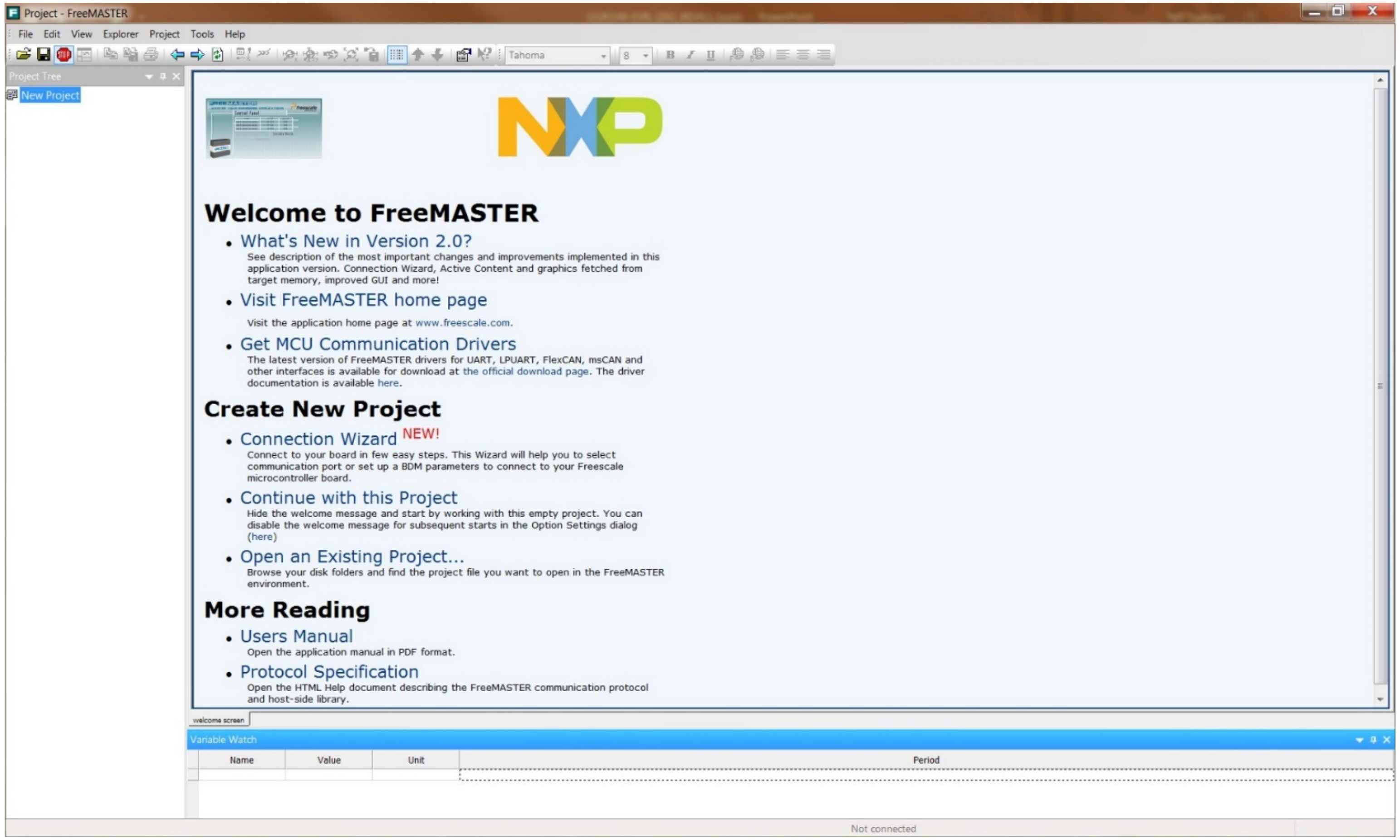
- Board is preloaded with a software, in which the red, blue and green leds will toggle at different rates.



S32K116 JUMPSTART EXPERIENCE BASED ON THE FREEMASTER TOOL

Install the FreeMASTER tool

- Download and install the FreeMASTER PC application www.nxp.com/FreeMASTER .
- Open the FreeMASTER application on your PC. You should see Welcome page:



Welcome to FreeMASTER

- What's New in Version 2.0?
  See description of the most important changes and improvements implemented in this application version. Connection Wizard, Active Content and graphics fetched from target memory, improved GUI and more!
- Visit FreeMASTER home page
  Visit the application home page at www.freescale.com.
- Get MCU Communication Drivers
  The latest version of FreeMASTER drivers for UART, LPUART, FlexCAN, msCAN and other interfaces is available for download at the official download page. The driver documentation is available here.

Create New Project

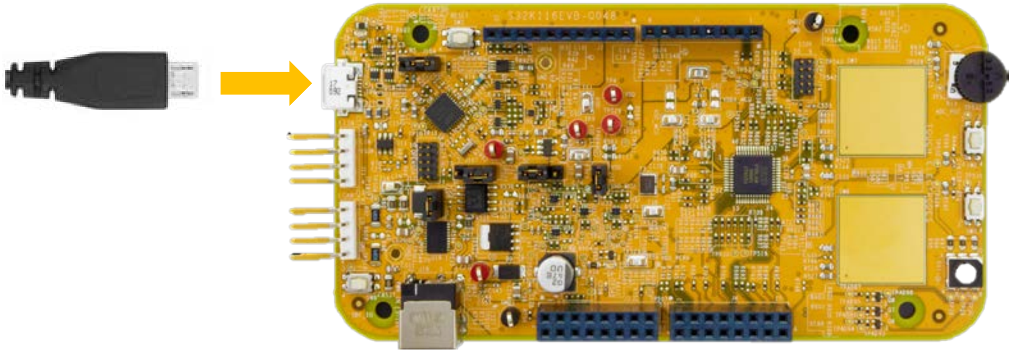
- Connection Wizard NEW!
  Connect to your board in few easy steps. This Wizard will help you to select communication port or set up a BDM parameters to connect to your Freescale microcontroller board.
- Continue with this Project
  Hide the welcome message and start by working with this empty project. You can disable the welcome message for subsequent starts in the Option Settings dialog (here)
- Open an Existing Project...
  Browse your disk folders and find the project file you want to open in the FreeMASTER environment.

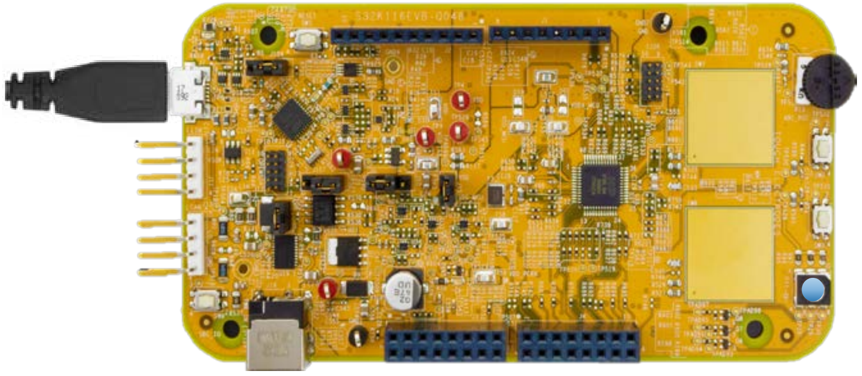
More Reading

- Users Manual
  Open the application manual in PDF format.
- Protocol Specification
  Open the HTML Help document describing the FreeMASTER communication protocol and host-side library.

Power up the EVB board

- Powers the S32K116EVB evaluation board from a USB. By default, the USB power is enabled by J07 jumper.
- Connect the USB cable to a PC and connect micro USB connector of the USB cable to micro-B port J7 on the S32K116EVB.
- Allow the PC to automatically configure the USB drivers if needed.
- When EVB is powered from USB, LEDs D2 and D3 should light green.
- The EVB board is preloaded with a software toggling the RGB LED colors periodically between RED-GREEN-BLUE.





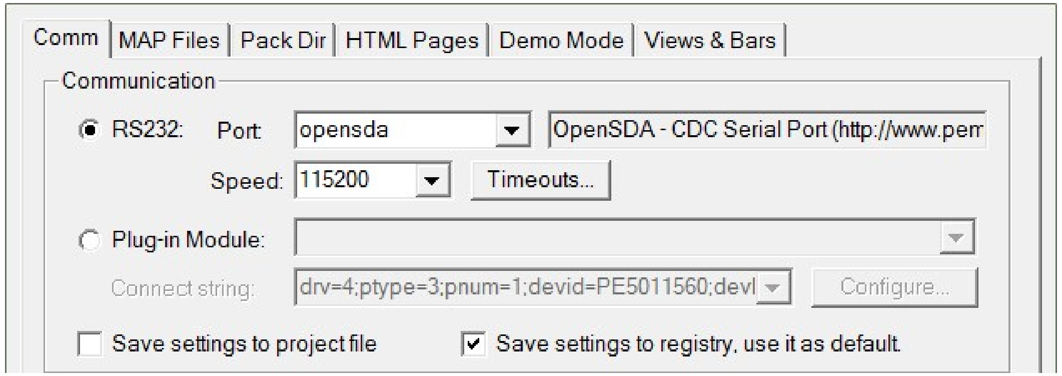
Setup serial connection in the FreeMASTER tool

Setup communication port to “OpenSDA“ and speed to 115200 b/s:

- Setup communication manualy:
  Go to: “Project > Options > Comm“

OR

- Setup communication automatically:
  Go to “Tools > Connection Wizard“



● RS232: Port: opensda | OpenSDA - CDC Serial Port (http://www.pem

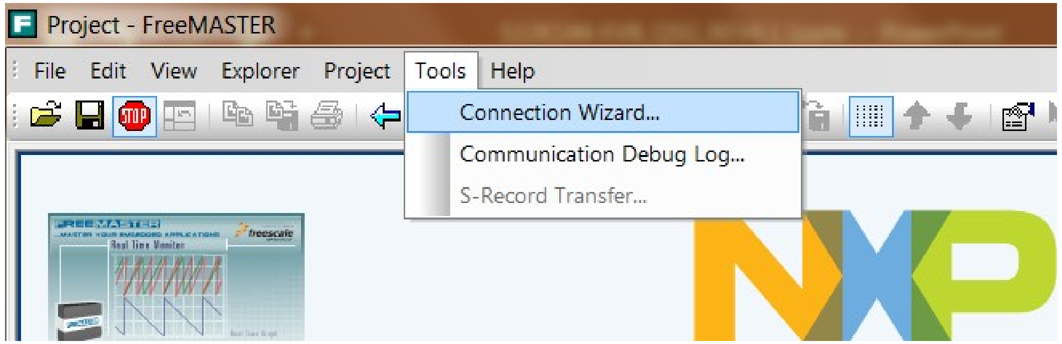
Speed: 115200 | Timeouts...

○ Plug-in Module:

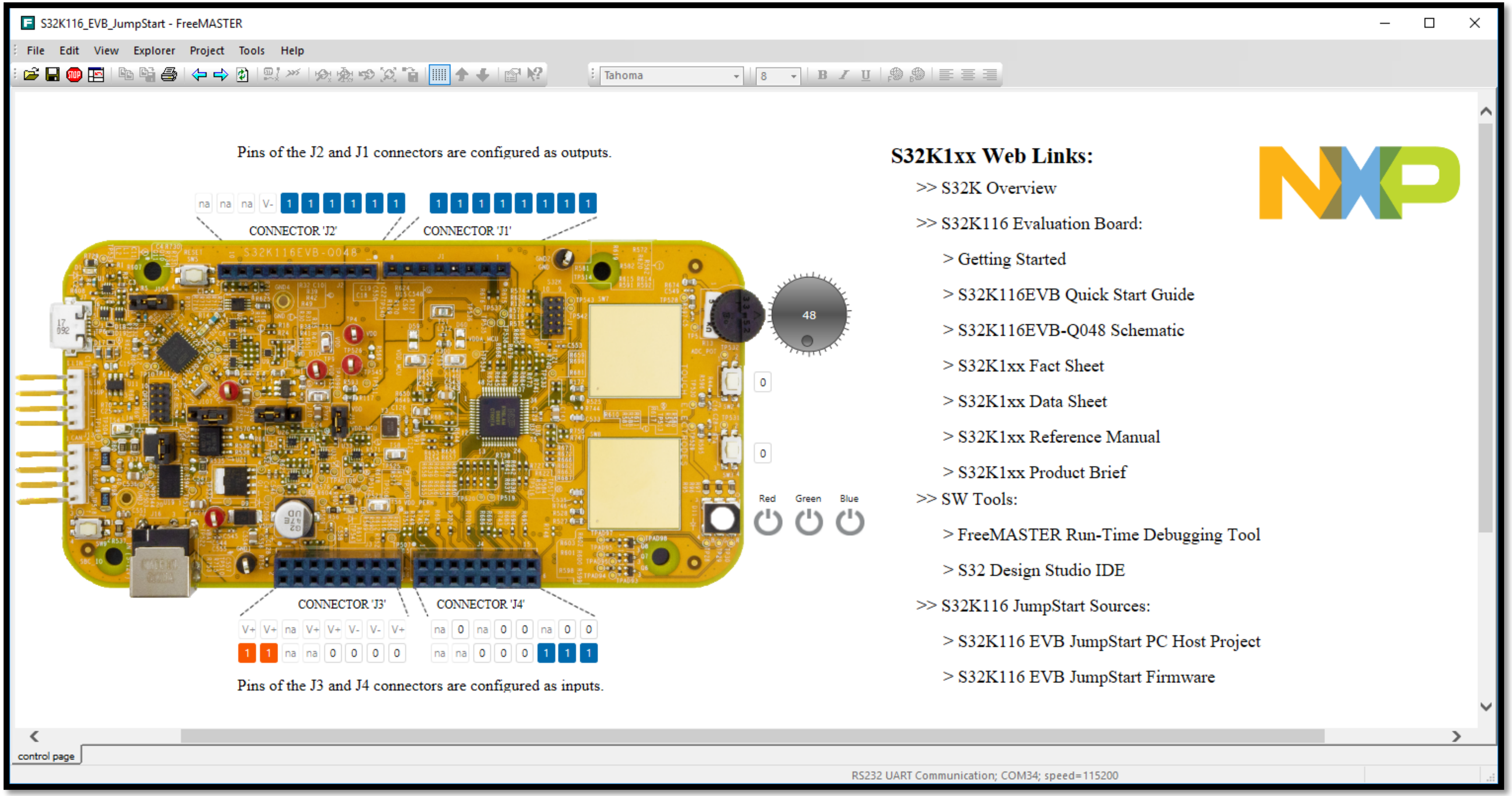
Connect string: drv=4;ptype=3;pnum=1;devid=PE5011560;devl | Configure...

□ Save settings to project file

☑ Save settings to registry, use it as default



The FreeMASTER JumpStart project is loaded



Pins of the J2 and J1 connectors are configured as outputs.

Pins of the J3 and J4 connectors are configured as inputs.

S32K1xx Web Links:

>> S32K Overview

>> S32K116 Evaluation Board:

> Getting Started

> S32K116EVB Quick Start Guide

> S32K116EVB-Q048 Schematic

> S32K1xx Fact Sheet

> S32K1xx Data Sheet

> S32K1xx Reference Manual

> S32K1xx Product Brief

>> SW Tools:

> FreeMASTER Run-Time Debugging Tool

> S32 Design Studio IDE

>> S32K116 JumpStart Sources:

> S32K116 EVB JumpStart PC Host Project

> S32K116 EVB JumpStart Firmware

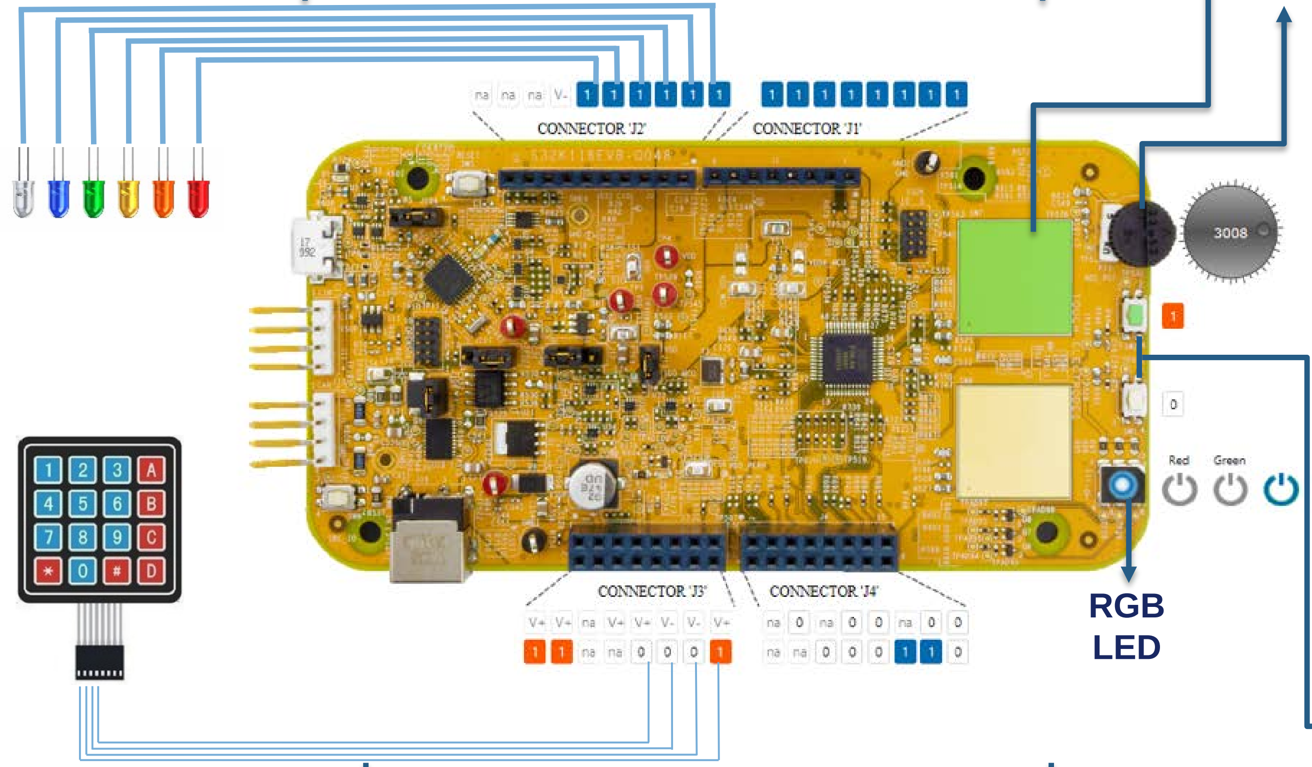
RS232 UART Communication; COM34; speed=115200

The FreeMASTER JumpStart project description

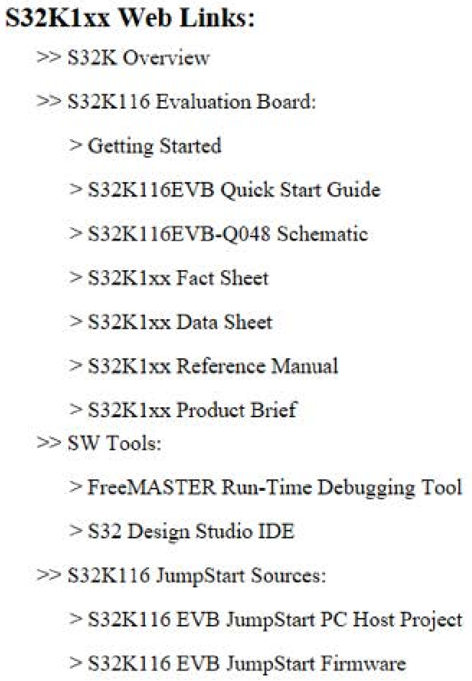
Pins of the J2 and J1 connectors are configured as outputs. By single click on each pin you can change their logical level to log0 or log1. User can connect e.g. LED diodes to these ouput pins.

Touch Sense Electrodes

Potentiometer



RGB LED



S32K1xx Web Links:

>> S32K Overview

>> S32K116 Evaluation Board:

> Getting Started

> S32K116EVB Quick Start Guide

> S32K116EVB-Q048 Schematic

> S32K1xx Fact Sheet

> S32K1xx Data Sheet

> S32K1xx Reference Manual

> S32K1xx Product Brief

>> SW Tools:

> FreeMASTER Run-Time Debugging Tool

> S32 Design Studio IDE

>> S32K116 JumpStart Sources:

> S32K116 EVB JumpStart PC Host Project

> S32K116 EVB JumpStart Firmware

Mechanical Buttons

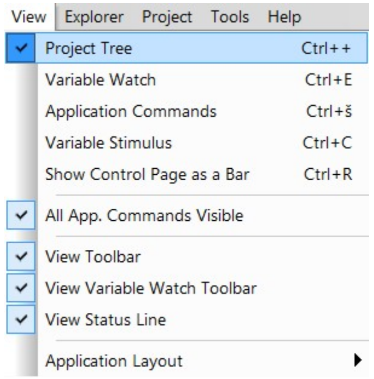
Links to S32K1xx docs:

- **Fact Sheet**
- **Data Sheet**
- **Reference Manual**
- **Product Brief**
- **S32K116EVB schematic**
- **S32K116EVB Quick Start Guide**
- **Tools:**
  - **FreeMASTER**
  - **S32 Design Studio IDE**
- **S32K116EVB JumpStart source files**

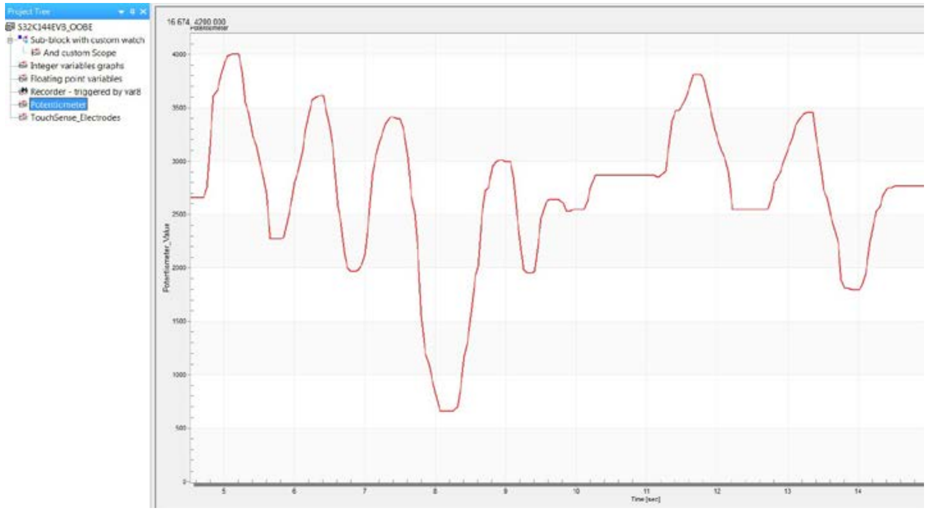
Pins of the J3 and J4 connectors are configured as inputs. Logical level (log0/log1) is visualised for all connector pins. User can connect e.g. push-button keyboard to these input pins.

The FreeMASTER JumpStart oscilloscope feature examples

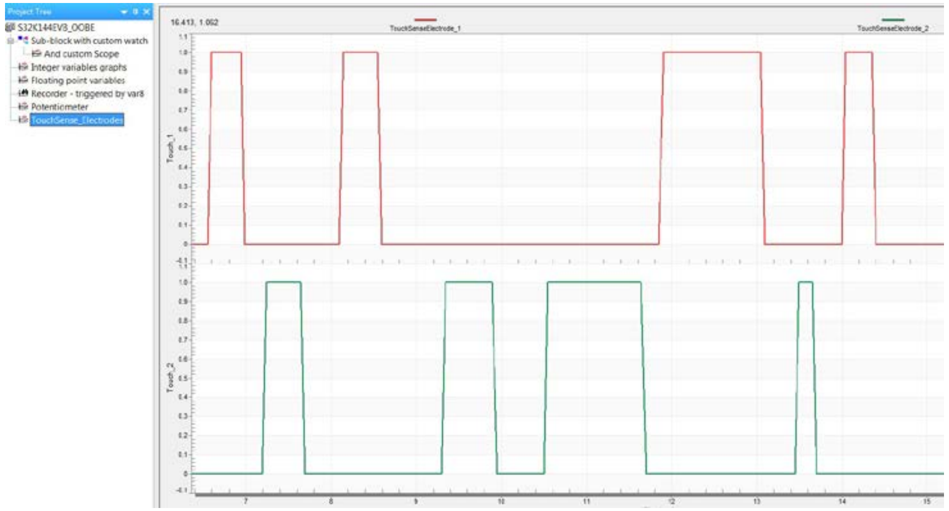
- Display main project panel “View > Project Tree“.



- Display real-time oscilloscope graph examples such as „Potentiometer“ or „Touch Sense Electrodes“.



Analog values from potentiometer.



Responses from touch sense electrodes.

INTRODUCTION TO OPENSDA

Introduction to OpenSDA: 1 of 2

OpenSDA is an open-standard serial and debug adapter. It bridges serial and debug communications between a USB host and an embedded target processor. OpenSDA software includes a flash-resident USB mass-storage device (MSD) bootloader and a collection of OpenSDA Applications. S32K116 EVB comes with the MSD Flash Programmer OpenSDA Application preinstalled. Follow these instructions to run the OpenSDA Bootloader and update or change the installed OpenSDA Application.

Enter OpenSDA Bootloader Mode

1. Unplug the USB cable if attached
2. Set J104 on position 1-2.
3. Press and hold the Reset button (SW5)
4. Plug in a USB cable (not included) between a USB host and the OpenSDA USB connector (labeled “SDA”)
5. Release the Reset button

A removable drive should now be visible in the host file system with a volume label of BOOTLOADER. You are now in OpenSDA Bootloader mode.

IMPORTANT NOTE: Follow the “Load an OpenSDA Application” instructions to update the MSD Flash Programmer on your S32K116 EVB to the latest version.

Load an OpenSDA Application

1. While in OpenSDA Bootloader mode, double-click ***SDA_INFO.HTML*** in the ***BOOTLOADER*** drive. A web browser will open the OpenSDA homepage containing the name and version of the installed Application. This information can also be read as text directly from ***SDA_INFO.HTML***
2. Locate the ***OpenSDA Applications***
3. Copy & paste or drag & drop the MSD Flash Programmer Application *to the **BOOTLOADER** drive*
4. Unplug the USB cable and plug it in again. The new OpenSDA Application should now be running and a ***S32K116 EVB*** drive should be visible in the host file system

You are now running the latest version of the MSD Flash Programmer. Use this same procedure to load other OpenSDA Applications.

Introduction to OpenSDA: 2 of 2

The MSD Flash Programmer is a composite USB application that provides a virtual serial port and an easy and convenient way to program applications into the S32K116 MCU. It emulates a FAT file system, appearing as a removable drive in the host file system with a volume label of S32K116EVB. Raw binary and Motorola S-record files that are copied to the drive are programmed directly into the flash of the S32K116 and executed automatically. The virtual serial port enumerates as a standard serial port device that can be opened with standard serial terminal applications.

Using the MSD Flash Programmer

1. Locate the .srec file of your project , file is under the Debug folder of the S32DS project.
2. Copy & paste or drag & drop one of the .srec files to the S32K116EVB drive

The new application should now be running on the S32K116 EVB. Starting with v1.03 of the MSD Flash Programmer, you can program repeatedly without the need to unplug and reattach the USB cable before reprogramming.

Drag one of the .srec code for the S32K116 EVB board over USB to reprogram the preloaded code example to another example.

NOTE: Flash programming with the MSD Flash Programmer is currently only supported on Windows operating systems. However, the virtual serial port has been successfully tested on Windows, Linux and Mac operating systems.

Using the Virtual Serial Port

1. Determine the symbolic name assigned to the S32K116EVB virtual serial port. In Windows open Device Manager and look for the COM port named “PEMicro/Freescale – CDC Serial Port”.
2. Open the serial terminal emulation program of your choice. Examples for Windows include Tera Term, PuTTY, and HyperTerminal
3. Press and release the Reset button (SW5) at anytime to restart the example application. Resetting the embedded application will not affect the connection of the virtual serial port to the terminal program.
4. It is possible to debug and communicate with the serial port at the same time, no need to stop the debug.

NOTE: Refer to the OpenSDA User’s Guide for a description of a known Windows issue when disconnecting a virtual serial port while the COM port is in use.

INSTALLING S32DS

Download S32DS

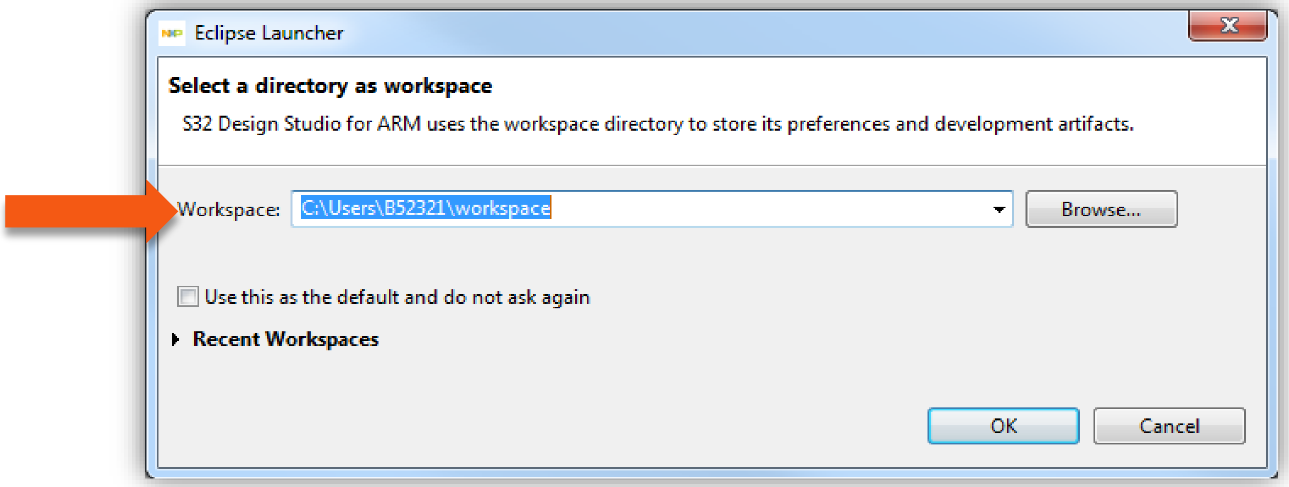
Download S32DS from:

S32DS for ARM

CREATE A NEW PROJECT IN S32 DESIGN STUDIO

Create New Project: First Time – Select a Workspace

- Start program: Click on “S32 Design Studio for ARM v2.0” icon
- Select workspace:
  - Choose default (see below example) or specify new one
  - Suggestion: Uncheck the box “Use this as the default and do not ask again”
  - Click OK



Eclipse Launcher

Select a directory as workspace

S32 Design Studio for ARM uses the workspace directory to store its preferences and development artifacts.

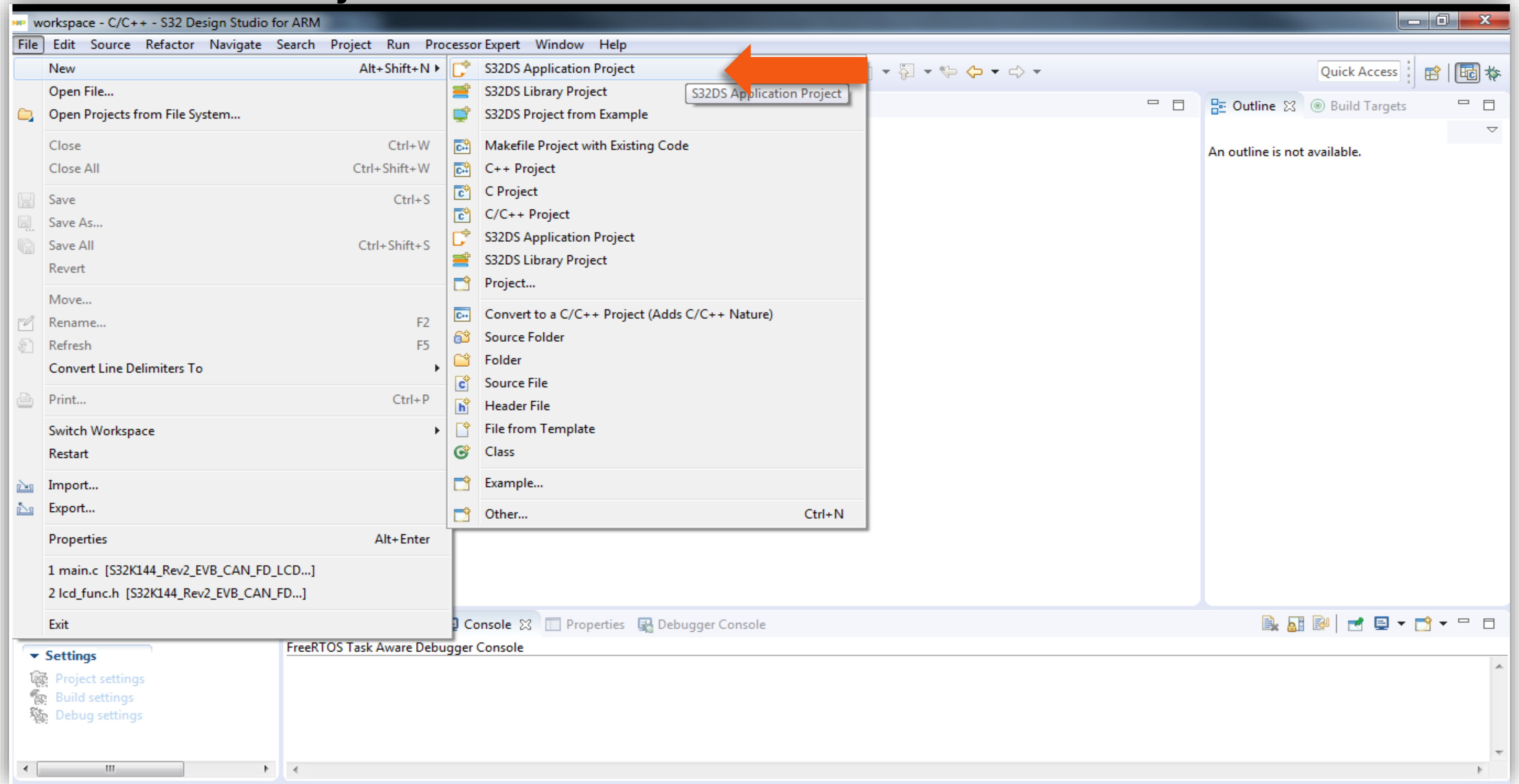
Workspace: C:\Users\B52321\workspace

□ Use this as the default and do not ask again

Recent Workspaces

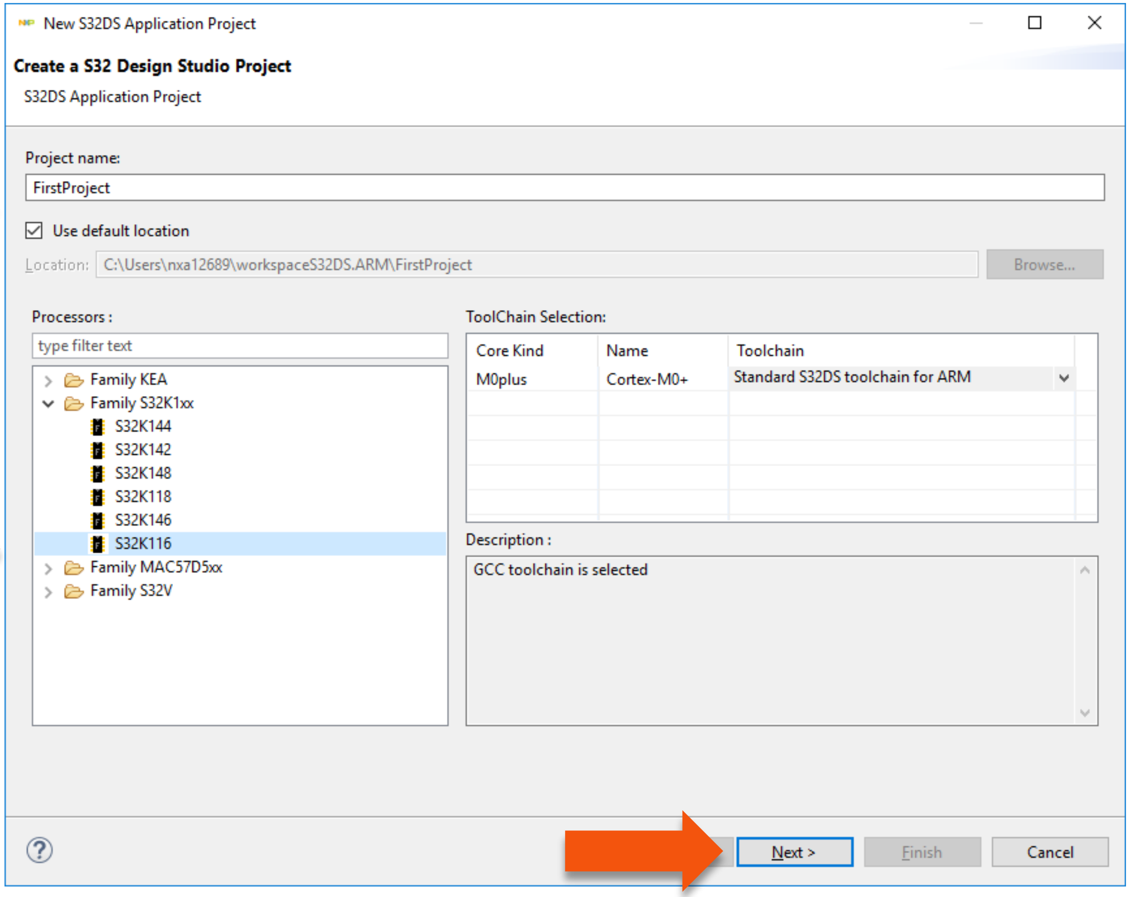
Create New Project: Top Menu Selection

- File – New –Project



Create New Project: S32DS Project

- Project Name:
  - Example: FirstProject
- Project Type:
  - Select from inside executable or library folder
- Next



New S32DS Application Project

Create a S32 Design Studio Project

S32DS Application Project

Project name: FirstProject

☑ Use default location

Location: C:\Users\nxa12689\workspaceS32DS.ARM\FirstProject

Processors :

type filter text

- Family KEA
- Family S32K1xx
  - S32K144
  - S32K142
  - S32K148
  - S32K118
  - S32K146
  - S32K116
- Family MAC57D5xx
- Family S32V

ToolChain Selection:

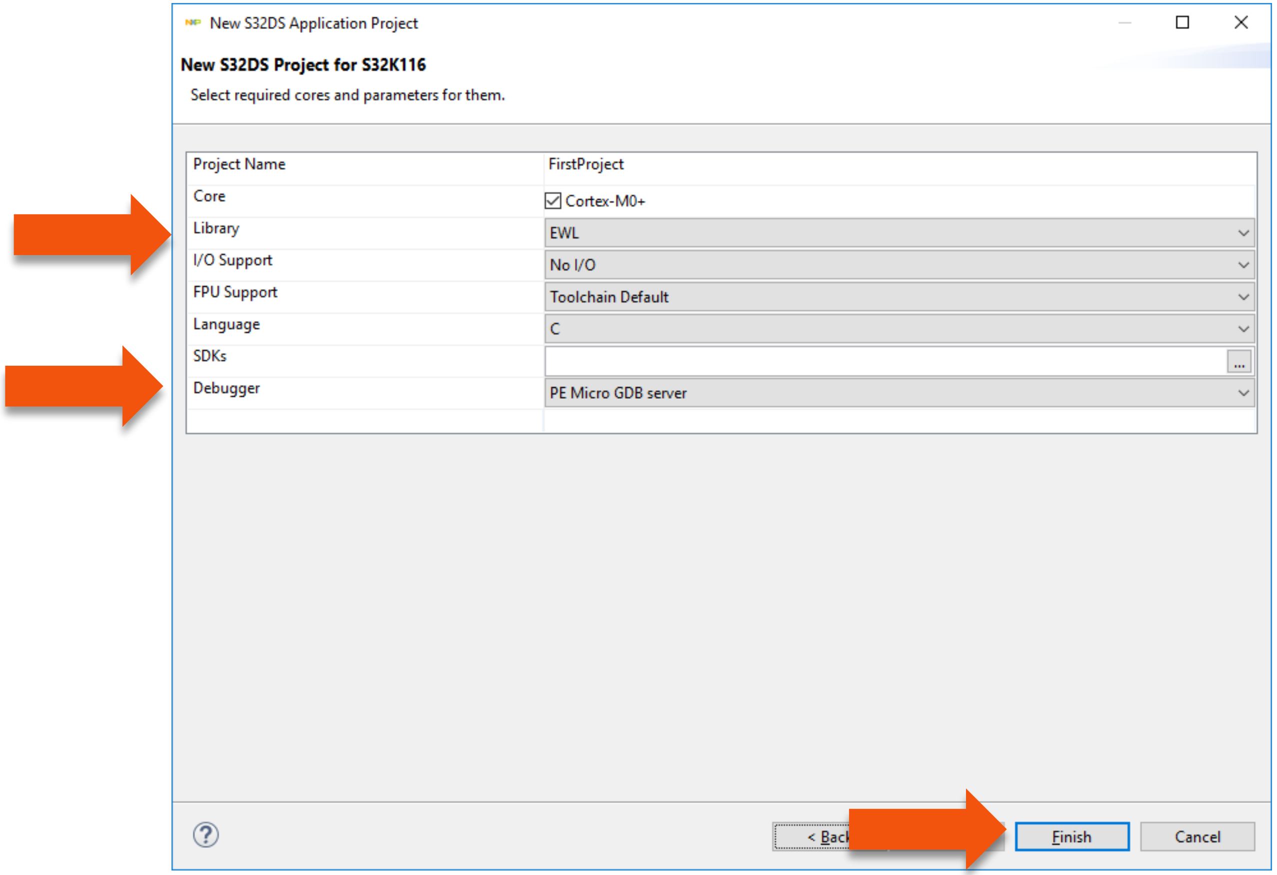
Core Kind	Name	Toolchain
M0plus	Cortex-M0+	Standard S32DS toolchain for ARM

Description :

GCC toolchain is selected

Create New Project: S32DS Project

- Select Debugger Support and Library Support
- Click Finish



New S32DS Application Project

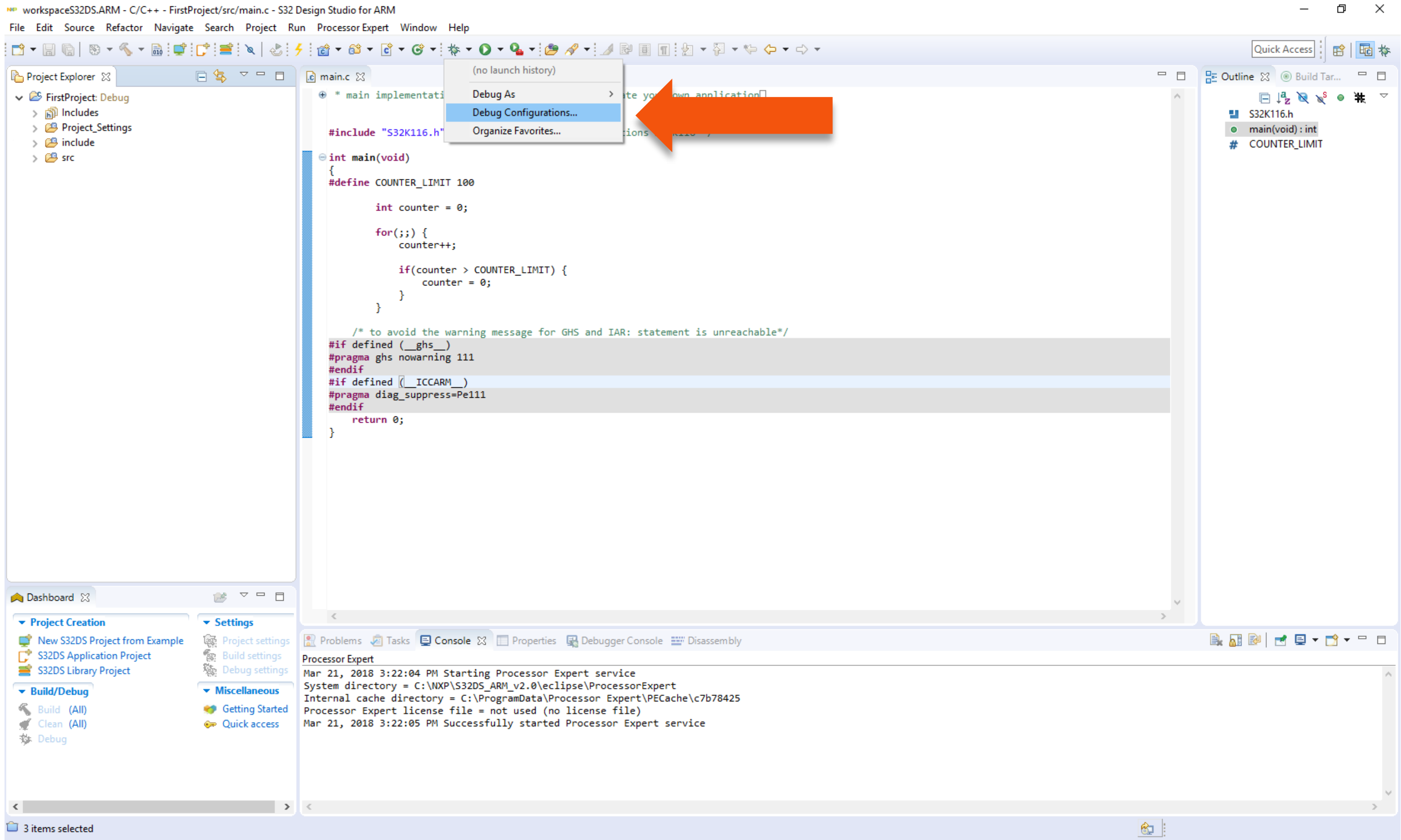
New S32DS Project for S32K116

Select required cores and parameters for them.

Project Name	FirstProject
Core	☑ Cortex-M0+
Library	EWL
I/O Support	No I/O
FPU Support	Toolchain Default
Language	C
SDKs	
Debugger	PE Micro GDB server

OpenSDA Configuration

- To Debug your project with OpenSDA, it is necessary to select the OpenSDA in the Debug Configuration.
- Select your project, and click on debug configuration

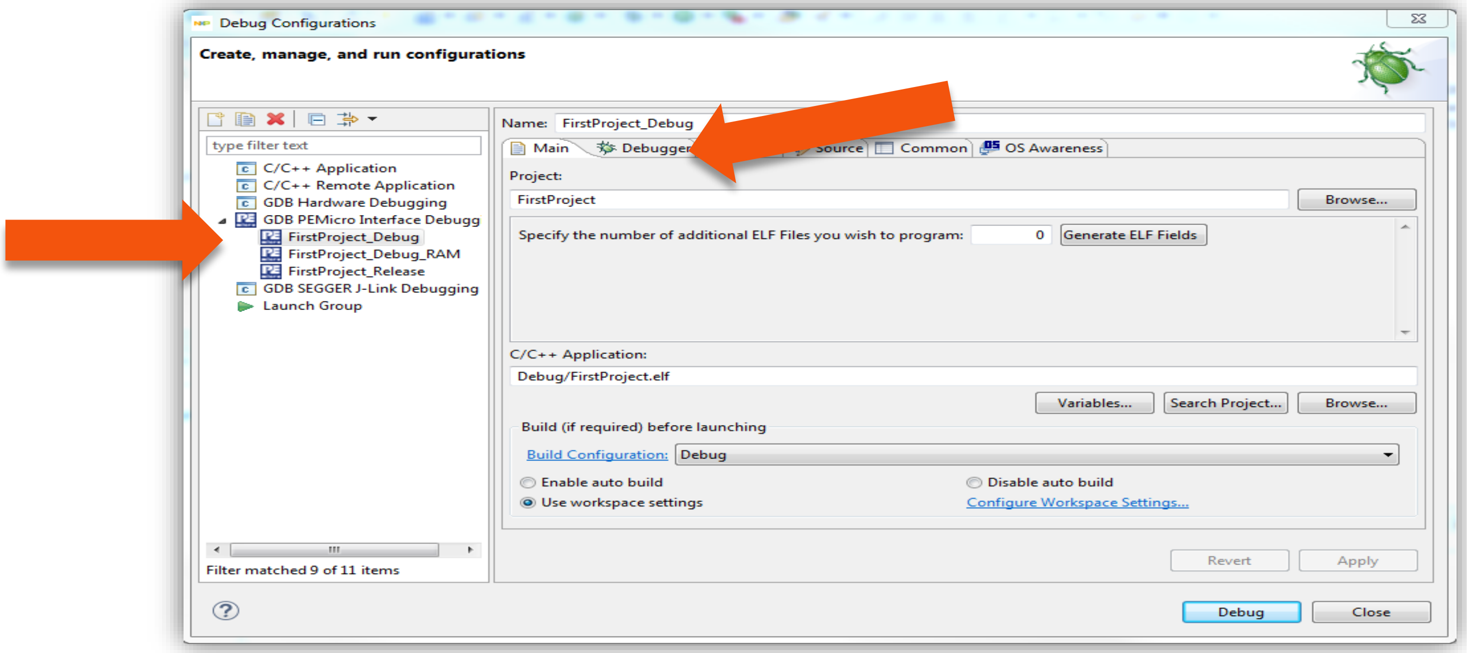


```
 * main implementati                    te your own application

#include "S32K116.h"                   tions

int main(void)
{
#define COUNTER_LIMIT 100

        int counter = 0;

        for(;;) {
            counter++;

            if(counter > COUNTER_LIMIT) {
                counter = 0;
            }
        }

    /* to avoid the warning message for GHS and IAR: statement is unreachable*/
#if defined (__ghs__)
#pragma ghs nowarning 111
#endif
#if defined (__ICCARM__)
#pragma diag_suppress=Pe111
#endif
    return 0;
}
```

```
Mar 21, 2018 3:22:04 PM Starting Processor Expert service
System directory = C:\NXP\S32DS_ARM_v2.0\eclipse\ProcessorExpert
Internal cache directory = C:\ProgramData\Processor Expert\PECache\c7b78425
Processor Expert license file = not used (no license file)
Mar 21, 2018 3:22:05 PM Successfully started Processor Expert service
```

OpenSDA Configuration

- Select the Debug configuration under GDB PEMicro Interface Debugging
- Click on Debugger tab



Debug Configurations

Create, manage, and run configurations

type filter text

- C/C++ Application
- C/C++ Remote Application
- GDB Hardware Debugging
- GDB PEMicro Interface Debugg
  - FirstProject_Debug
  - FirstProject_Debug_RAM
  - FirstProject_Release
- GDB SEGGER J-Link Debugging
- Launch Group

Filter matched 9 of 11 items

Name: FirstProject_Debug

Project:

FirstProject

Specify the number of additional ELF Files you wish to program: 0

C/C++ Application:

Debug/FirstProject.elf

Build (if required) before launching

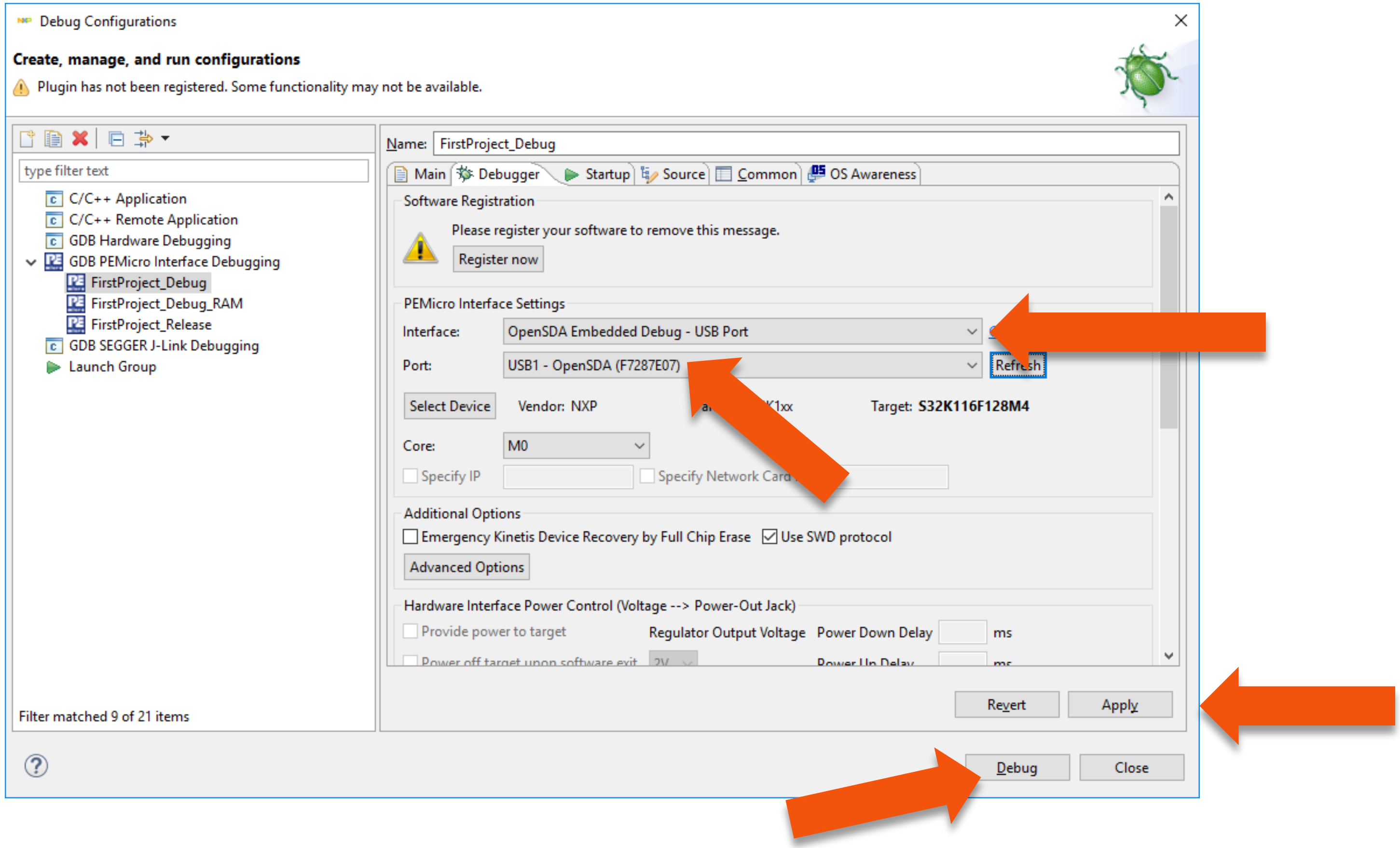
Build Configuration: Debug

○ Enable auto build ○ Disable auto build

● Use workspace settings

OpenSDA Configuration

- Select OpenSDA as the interface, if your board is plugged should appear in the Port field.
- Click Apply and debug to finish.



Debug Configurations

Create, manage, and run configurations

Plugin has not been registered. Some functionality may not be available.

type filter text

- C/C++ Application
- C/C++ Remote Application
- GDB Hardware Debugging
- GDB PEMicro Interface Debugging
  - FirstProject_Debug
  - FirstProject_Debug_RAM
  - FirstProject_Release
- GDB SEGGER J-Link Debugging
- Launch Group

Filter matched 9 of 21 items

Name: FirstProject_Debug

Software Registration

Please register your software to remove this message.

PEMicro Interface Settings

Interface: OpenSDA Embedded Debug - USB Port

Port: USB1 - OpenSDA (F7287E07)

Vendor: NXP

K1xx

Target: **S32K116F128M4**

Core: M0

□ Specify IP

□ Specify Network Card

Additional Options

□ Emergency Kinetis Device Recovery by Full Chip Erase ☑ Use SWD protocol

Hardware Interface Power Control (Voltage --> Power-Out Jack)

□ Provide power to target

Regulator Output Voltage

Power Down Delay ms

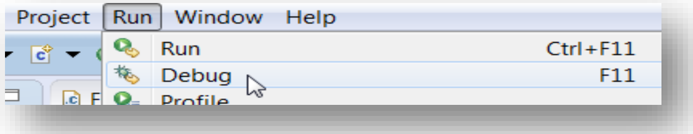
□ Power off target upon software exit 2V

Power Up Delay ms

DEBUG BASICS

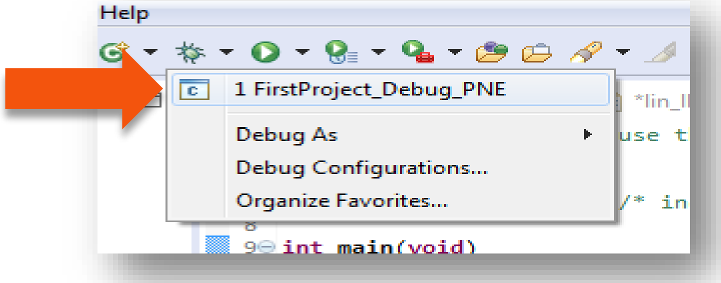
Debug Basics: Starting the Debugger

- Debug configuration is only required once. Subsequent starting of debugger does not require those steps.
- Three options to start debugger:
  - If the “Debug Configuration” has not been closed, click on “Debug” button on bottom right
  - Select Run – Debug (or hit F11)



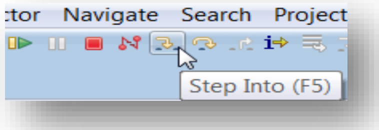
    Note: This method currently selects the desktop target (*project.*elf) and gives an error. Do not use until this is changed.

  - ***<u>Recommended Method</u>***: Click on pull down arrow for bug icon and select …_debug.elf target

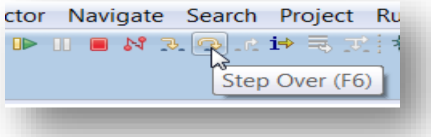


Debug Basics: Step, Run, Suspend, Resume

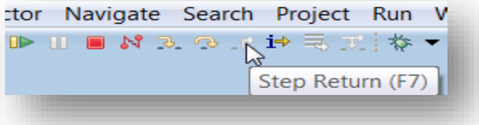
- Step Into (F5)



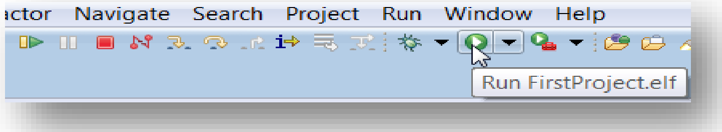
- Step Over (F6)



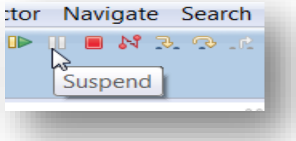
- Step Return (F7)



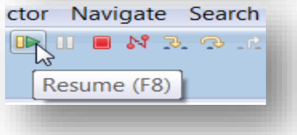
- Run



- Suspend

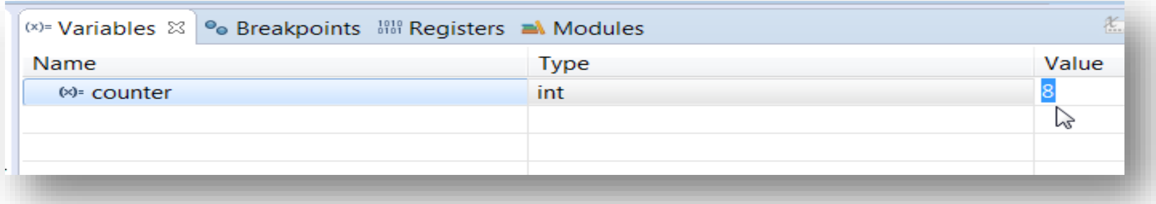


- Resume (F8)



Debug Basics: View & Alter Variables

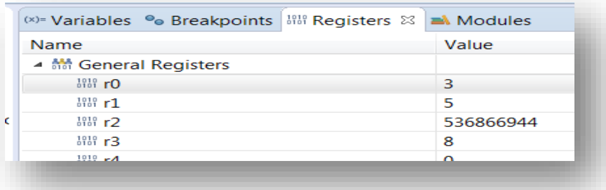
- View variables in “Variables” tab.
- Click on a value to allow typing in a different value.



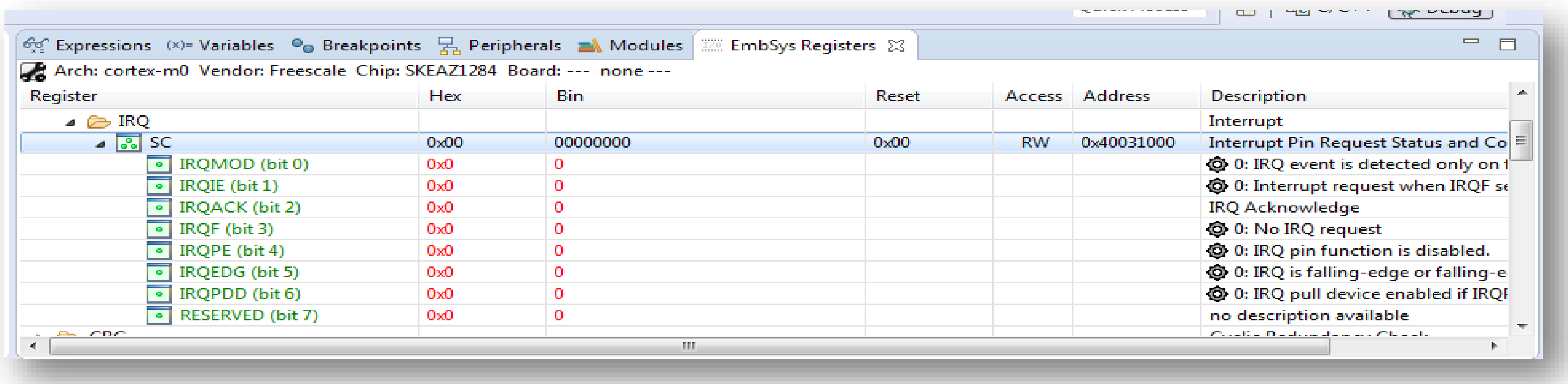
Name	Type	Value
(x)= counter	int	8

Debug Basics: View & Alter Registers

- View CPU registers in the “Registers” tab
- Click on a value to allow typing in a different value
- View peripheral registers in the EmbSys Registers tab



Name	Value
General Registers	
r0	3
r1	5
r2	536866944
r3	8
r4	0

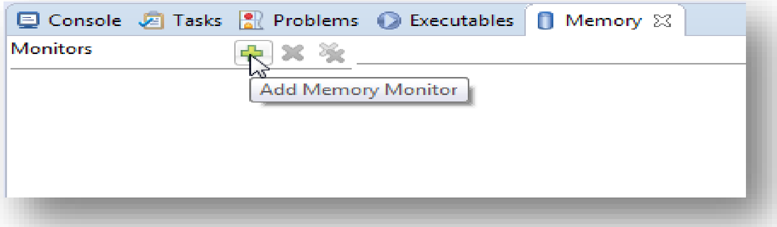


Arch: cortex-m0 Vendor: Freescale Chip: SKEAZ1284 Board: --- none ---

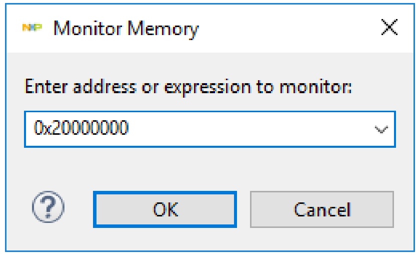
Register	Hex	Bin	Reset	Access	Address	Description
IRQ						Interrupt
SC	0x00	00000000	0x00	RW	0x40031000	Interrupt Pin Request Status and Co
IRQMOD (bit 0)	0x0	0				0: IRQ event is detected only on f
IRQIE (bit 1)	0x0	0				0: Interrupt request when IRQF se
IRQACK (bit 2)	0x0	0				IRQ Acknowledge
IRQF (bit 3)	0x0	0				0: No IRQ request
IRQPE (bit 4)	0x0	0				0: IRQ pin function is disabled.
IRQEDG (bit 5)	0x0	0				0: IRQ is falling-edge or falling-e
IRQPDD (bit 6)	0x0	0				0: IRQ pull device enabled if IRQF
RESERVED (bit 7)	0x0	0				no description available

Debug Basics: View & Alter Memory

- Add Memory Monitor



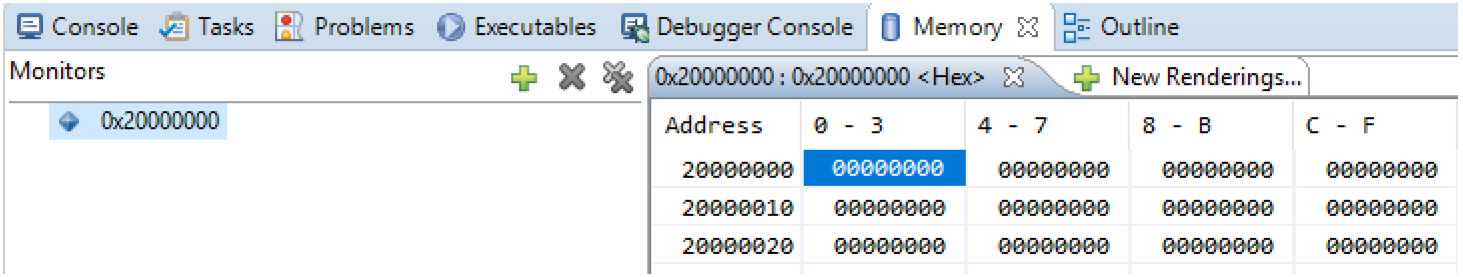
- Select Base Address to Start at : 0x20000000



Enter address or expression to monitor:

0x20000000

- View Memory



0x20000000

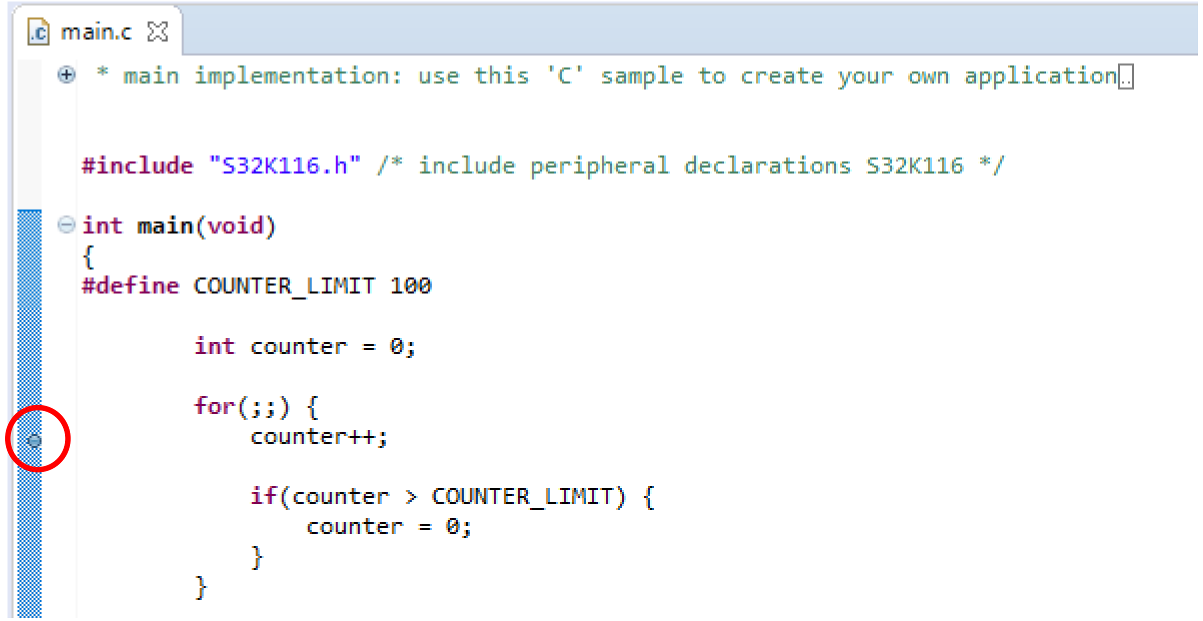
0x20000000 : 0x20000000 <Hex>

Address	0 - 3	4 - 7	8 - B	C - F
20000000	00000000	00000000	00000000	00000000
20000010	00000000	00000000	00000000	00000000
20000020	00000000	00000000	00000000	00000000

Debug Basics: Breakpoints

Add Breakpoint: Point and Click

- light blue dot represents debugger breakpoint

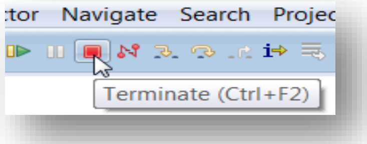


```
* main implementation: use this 'C' sample to create your own application

#include "S32K116.h" /* include peripheral declarations S32K116 */

int main(void)
{
#define COUNTER_LIMIT 100

        int counter = 0;

        for(;;) {
            counter++;

            if(counter > COUNTER_LIMIT) {
                counter = 0;
            }
        }
```

Debug Basics: Reset & Terminate Debug Session

- Reset program counter

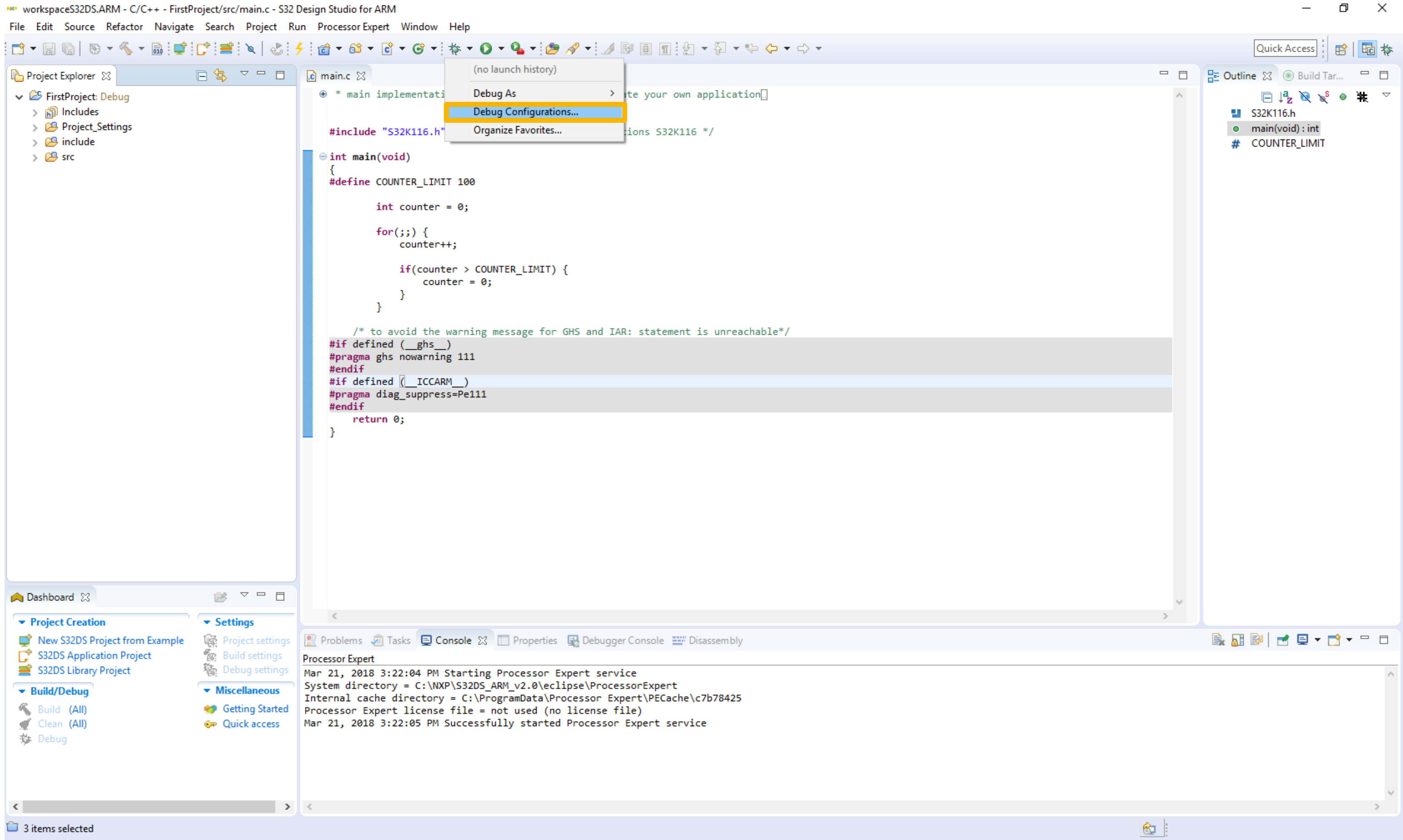
- Terminate Ctl+F2()



CREATE A P&E DEBUG CONFIGURATION (OPTIONAL)

New P&E debug configuration

- Click in debug configurations

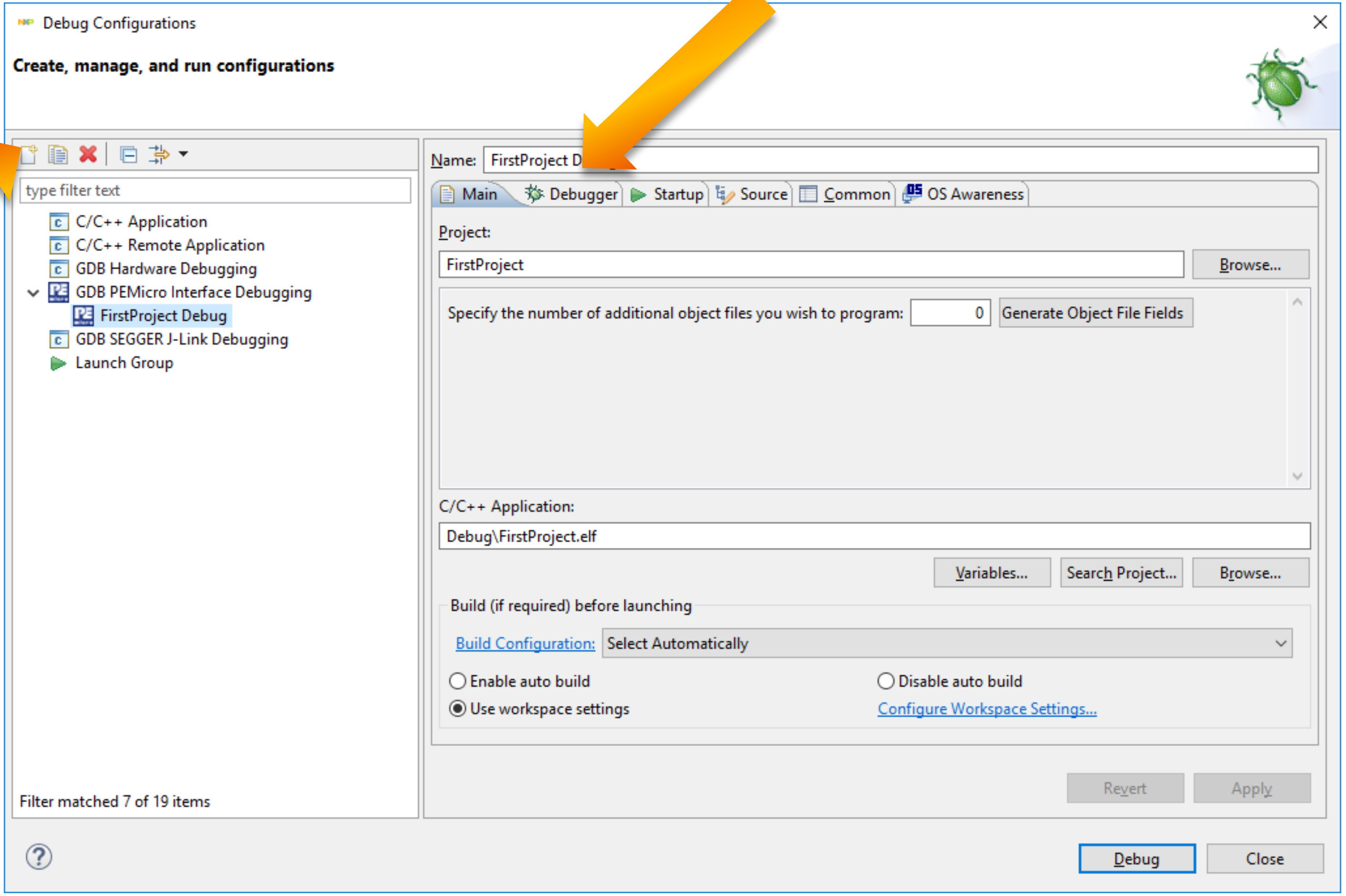


```
 * main implementati                                     te your own application

#include "S32K116.h"                                    ions S32K116 */

int main(void)
{
#define COUNTER_LIMIT 100

        int counter = 0;

        for(;;) {
            counter++;

            if(counter > COUNTER_LIMIT) {
                counter = 0;
            }
        }

    /* to avoid the warning message for GHS and IAR: statement is unreachable*/
#if defined (__ghs__)
#pragma ghs nowarning 111
#endif
#if defined (__ICCARM__)
#pragma diag_suppress=Pe111
#endif
    return 0;
}
```

```
Mar 21, 2018 3:22:04 PM Starting Processor Expert service
System directory = C:\NXP\S32DS_ARM_v2.0\eclipse\ProcessorExpert
Internal cache directory = C:\ProgramData\Processor Expert\PECache\c7b78425
Processor Expert license file = not used (no license file)
Mar 21, 2018 3:22:05 PM Successfully started Processor Expert service
```

New P&E debug configuration

- Create a new P&E launch configuration



Click on the debugger tab.

Click to create a new P&E launch

Debug Configurations

Create, manage, and run configurations

type filter text

- C/C++ Application
- C/C++ Remote Application
- GDB Hardware Debugging
- GDB PEMicro Interface Debugging
  - FirstProject Debug
- GDB SEGGER J-Link Debugging
- Launch Group

Filter matched 7 of 19 items

Name: FirstProject D

Project:

FirstProject

Specify the number of additional object files you wish to program: 0

C/C++ Application:

Debug\FirstProject.elf

Build (if required) before launching

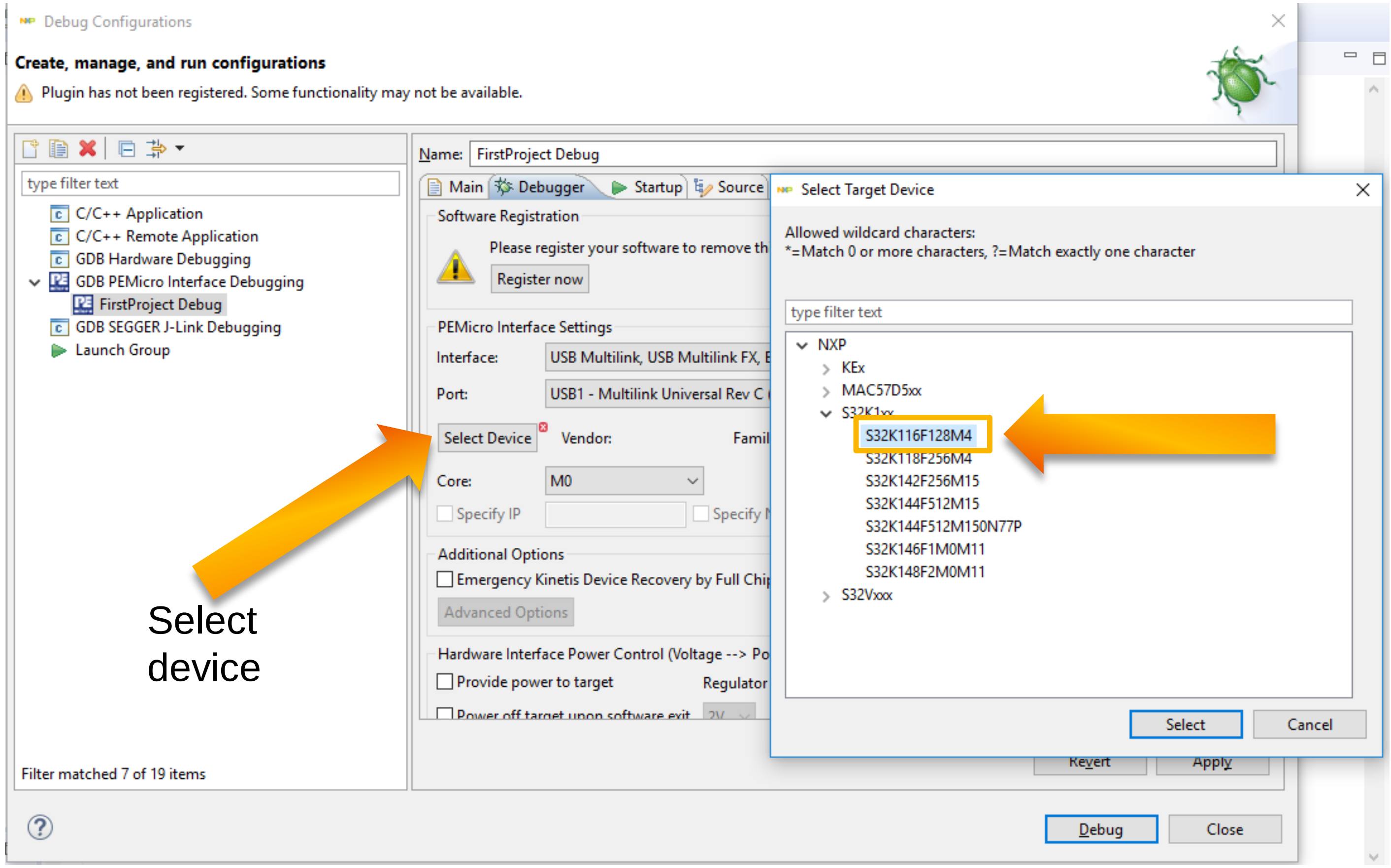
Build Configuration: Select Automatically

○ Enable auto build
○ Disable auto build
● Use workspace settings

Configure Workspace Settings...

New P&E debug configuration

- Select the device



Debug Configurations

Create, manage, and run configurations

Plugin has not been registered. Some functionality may not be available.

Name: FirstProject Debug

Software Registration

Please register your software to remove th

PEMicro Interface Settings

Interface: USB Multilink, USB Multilink FX, E

Port: USB1 - Multilink Universal Rev C

Core: M0

Additional Options

□ Emergency Kinetis Device Recovery by Full Chip

Hardware Interface Power Control (Voltage --> Po

□ Provide power to target

□ Power off target upon software exit

Filter matched 7 of 19 items

Select Target Device

Allowed wildcard characters:
*=Match 0 or more characters, ?=Match exactly one character

Select device

- Click Apply and debug your application

USEFUL LINKS