NXP Semiconductors
Application Note

Document Number: AN12875
Rev. 0, May 2020

Getting started with the MPC5775B-EVB and MPC5775E-EVB

by: **NXP Semiconductors**

Contents

1 Introduction

The MPC5775B and MPC5775E are automotive microcontrollers that are intended for use in battery management and inverter applications. Both devices are derivative versions of the MPC5777C, which was primarily intended for combustion engines and higher end combined battery management and inverter control applications.

This quick start guide shows an overview of the MPC5775B-EVB and MPC5775E-EVB evaluation boards and walks through installing the NXP S32 Design Studio installation and creation of a first simple application.

The MPC5775E and MPC577B Evaluation Boards (EVB) both feature additional NXP devices. Both boards include the TJA1100 Automotive Ethernet PHY and the MC33FS6520 System Basic Chip (SBC). The SBC contains the power supplies necessary for the device, as well as a CAN PHY and a LIN PHY. A NXP TJA1145T/FD CAN physical interface is also provided.

The MPC5775B also includes a MC33664 Isolated Network High Speed Transceiver that is intended for use with the MC3377x Battery Cell Controller.

2 MPC5775B and MPC5775E overview

Both the MPC5775B and the MPC5775E include two functional e200z7 cores (Power Architecture®), one of which is a lock-step pair of cores. Both also include 512K of internal SRAM and 4MB of flash memory. The table below shows the major differences between the MPC5775B and the MPC5775E.

Table 1. MPC5775E and MPC5775B major differences

Feature	MPC5775E	MPC5775B
Maximum operating frequency	264 MHz	220 MHz
eTPU quantity	3	0
eTPU channels	96	0
eTPU code memory	24KB + 12KB	N/A
eTPU data memory	6KB + 3KB	N/A
ADC quantity	4	2[1]
ADC input pins	70	40
Sigma- Delta ADC	Yes	No

[1] EQADCA only

The MPC5775E is intended for inverter applications. The following figure shows the block diagram of the MPC5775E.

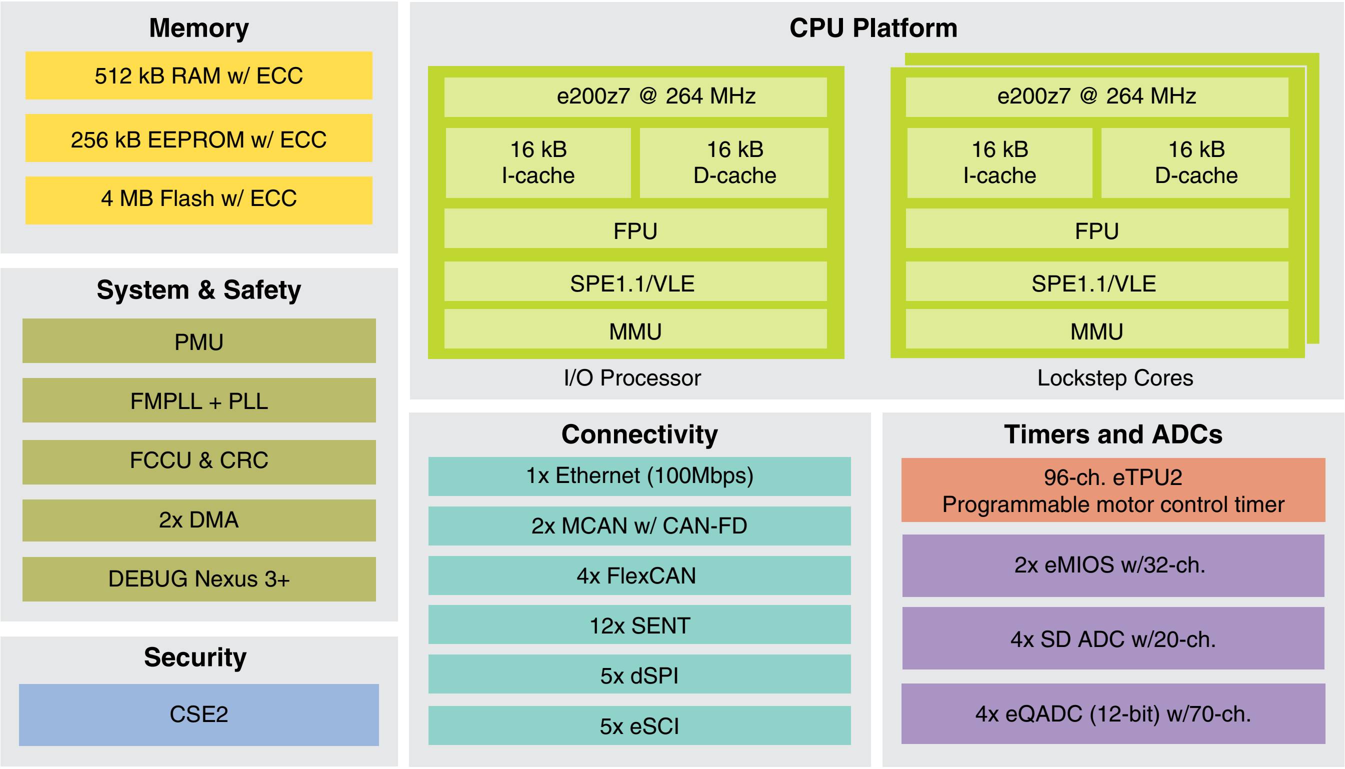


Figure 1. MPC5775E block diagram

The MPC5775B is intended for battery management applications. The following figure shows the block diagram of the MPC5775B.

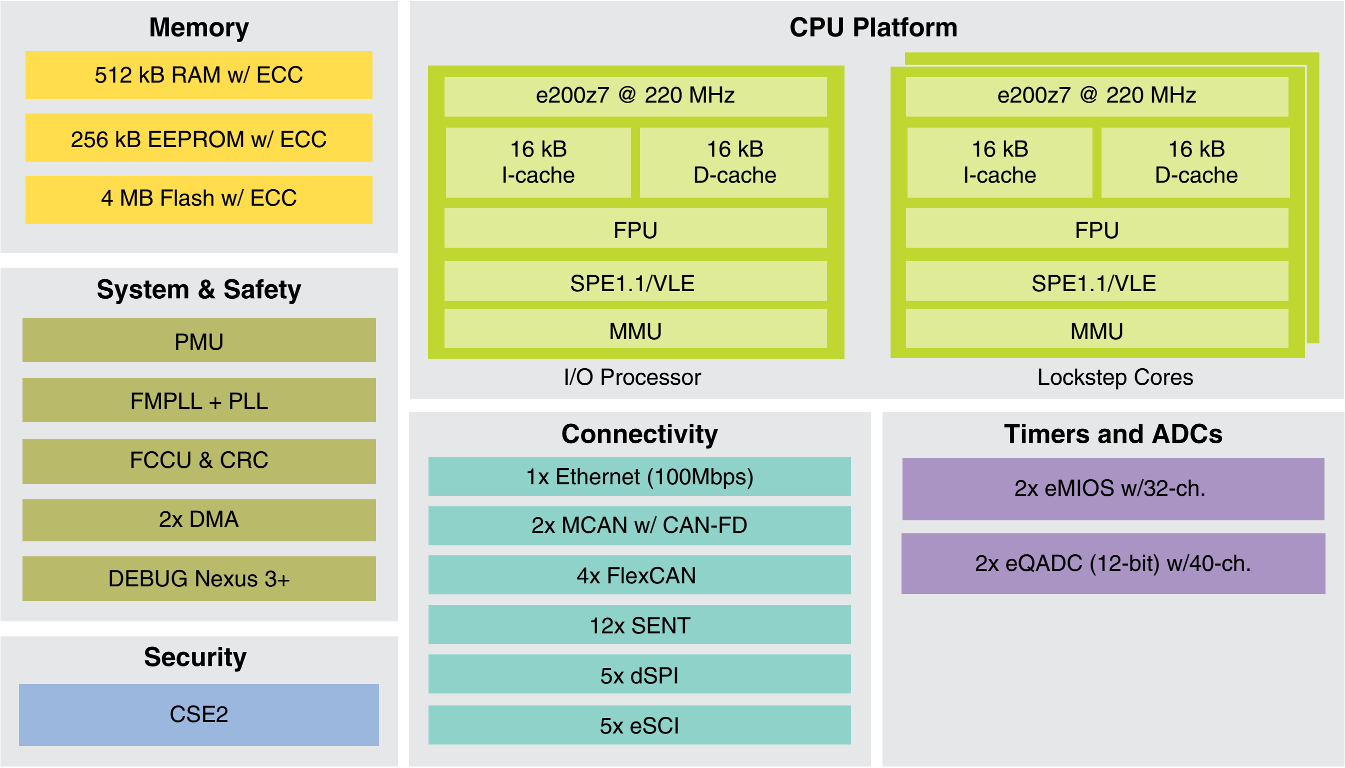


Figure 2. MPC5775B block diagram

2.1 EVB part placement

The figure below shows the overall layout of the MPC5777E and MPC5775B EVBs, highlighting the different connectors.

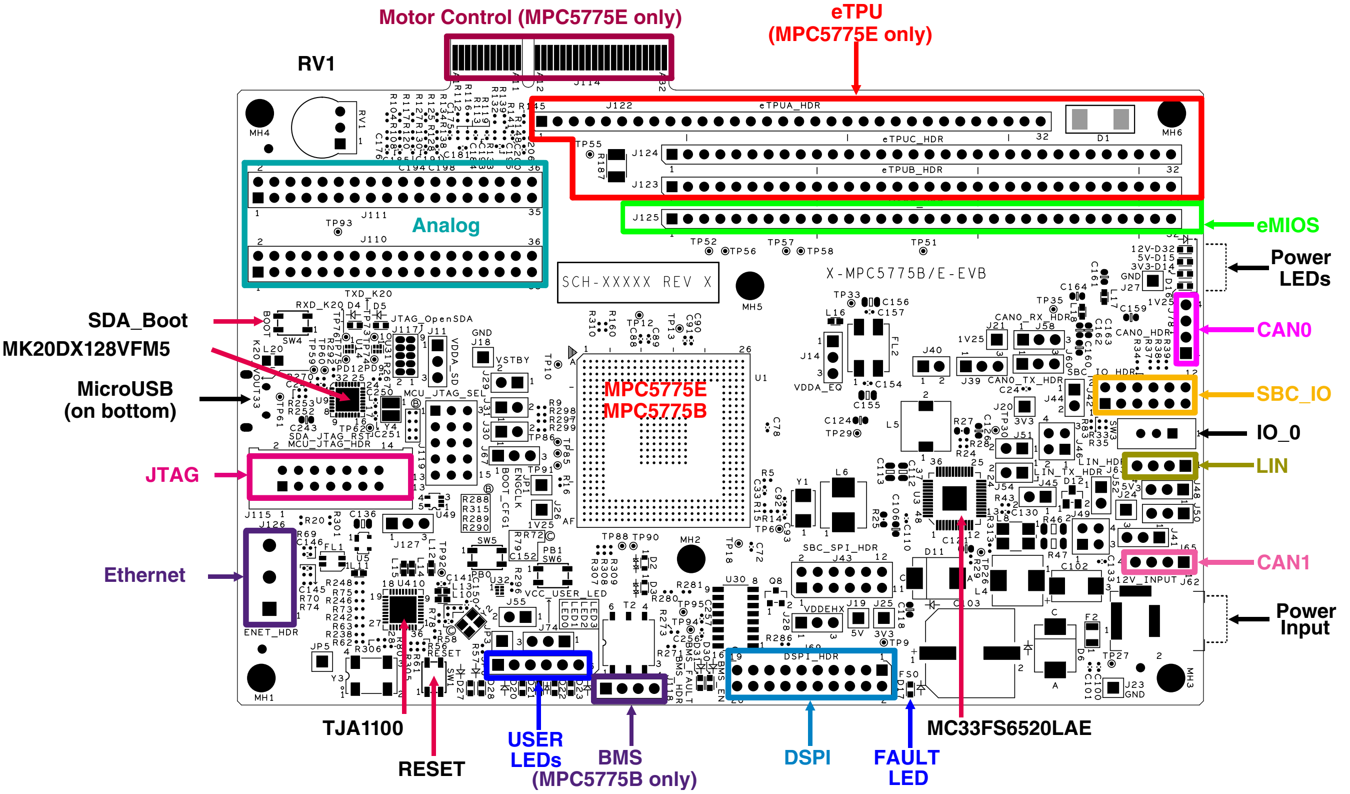


Figure 3. EVB connectors

3 MPC5775B-EVB and MPC5775E-EVB evaluation board overview

Features of the MPC5775B-EVB and MPC5775E-EVB boards are:

- Requires 12 V external DC power supply (included in kit)
- MC33FS6520LAE System Basis Chip (SBC) for the board power supply
- SBC CAN physical interface (selectable [J48/J50] between FlexCAND/MCAN1 [GPIO246 and GPIO247] or FLEXCANC [GPIO87 and GPIO88])
  - Option to select CAN termination (J49)
- SBC LIN physical interface connected to RXDC/TXDC (J51/J54 connected by default).
  - Master or slave mode supported (J52 VSUP connection)
- TJA1145T/FD CAN Physical interface (selectable between FlexCANA/MCAN0 [GPIO83 and GPIO84] and FlexCANB [GPIO85 and GPIO86])
- TJA1100 Automotive Ethernet PHY (physical interface)
- Debug selectable between external debug connection or on-board OpenSDA (JTAG to USB interface)
- eMIOS header pins
- ADC header pins
- DSPI header pins
- **MPC5775B-EVB only:** MC33664 Battery Management System (BMS) interface
- **MPC5775E-EVB only:** eTPU header pins
- **MPC5775E-EVB only:** Motor Control connector (PCIe X4 style connector)

A block diagram of the MPC5775B-EVB and MPC5775E-EVB evaluation boards is shown below.

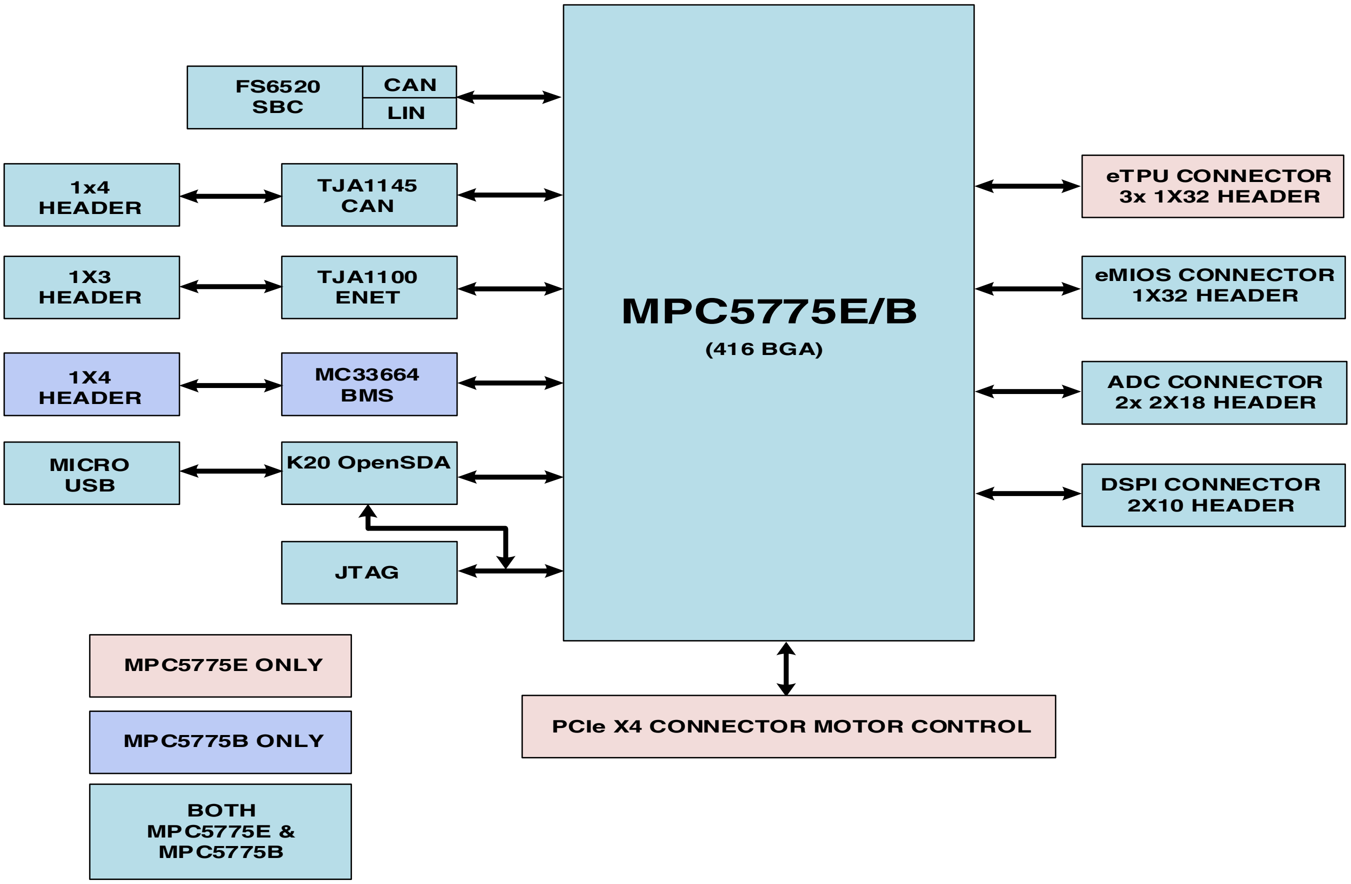


Figure 4. MPC5775B-EVB and MPC5775E EVB-EVB block diagram

3.1 Common connectors

This section shows the pin-outs of the different headers that are the same on the MPC5775B-EVB and the MPC5775E-EVB. These headers give access to signals from the MPC5775B or MPC5775E.

3.1.1 LED header pin-out

The following table shows the pin-out of the LED connector.

Table 2. LED header (JP6)

Pin	Connection
1	Ground
2	Green LED
3	Red LED
4	Blue LED
5	Orange LED
6	JP74 (Selects either 5V [pins 1-2] or 3.3V [pins 2-3])

3.1.2 CAN header pin-out

The following tables shows the pin-outs of the CAN headers. The physical interface (PHY) for CAN0 connector is a TJA1145T/FD and can be connected to either FlexCAN_A/MCAN0 or FlexCAN_B (via jumpers J58 and J60[1]). The following table shows the pin-out of the CAN1 header.

Table 3. CAN0 header (J78)

Header pin	Signal
1	5V
2	CAN1_0H
3	CAN1_0L
4	Ground

A second CAN PHY is available inside the MC33F6520LDAE System Basis Chip (SBC). It can be connected to either FlexCAN_D/MCAN1 or FlexCAN_C (via jumpers J48 and J50[2]). The following table shows the pin-out of the CAN0 header.

Table 4. CAN1 header (J65)

Header pin	Signal
1	5V
2	CAN1H
3	CAN1L
4	Ground

3.1.3 LIN header pin-out

The SBC also includes a LIN PHY. This is connected to LINFlex_C (J51 and J54) and is available on J63. The pin-out of J63 is shown in the following table. The LIN connector can be configured to master or slave operation via J52 (default is master).

Table 5. LIN header (J63)

Header pin	Signal
1	Ground
2	Ground
3	VSUP
4	LIN

[1] Both connectors must be configured to the same CAN interface.

[2] Both connectors must be configured to the same CAN interface.

3.1.4 Ethernet Connector

The table below shows the pin-out of the Ethernet Connector (J69).

Table 6. Ethernet Connector (J69)

Connector pin	Signal
1	Ground
2	TRX_CONN_P
3	TRX_CONN_N

3.1.5 eMIOS header pin-out

The table below shows the pin-out of the enhanced Modular Input/Output System (eMIOS) header.

Table 7. eMIOS connector pin-out

Connector pin	Signal
1	eMIOS0
2	eMIOS1
3	eMIOS2
4	eMIOS3
5	eMIOS4
6	eMIOS5
7	eMIOS6
8	eMIOS7
9	eMIOS8
10	eMIOS9
11	eMIOS10
12	eMIOS11
13	eMIOS12
14	eMIOS13
15	eMIOS14
16	eMIOS15
17	eMIOS16
18	eMIOS17
19	eMIOS18
20	eMIOS19
21	eMIOS20
22	eMIOS21

Table continues on the next page...

Table 7. eMIOS connector pin-out (continued)

Connector pin	Signal
23	eMIOS22
24	eMIOS23
25	eMIOS24
26	eMIOS25
27	eMIOS26
28	eMIOS27
29	eMIOS28
30	eMIOS29
31	eMIOS30
32	eMIOS31

3.1.6 ADC header pin-out

Two (2) headers are provided for Analog-to-digital pins as shown in the two tables below.

Table 8. ADC header (J110)

Signal	Header pin	Header pin	Signal
HDR_ANA16_SDB0	1	2	ANB8
ANA17_SDB1	3	4	ANB9
ANA18_SDB2	5	6	ANB10
ANA19_SDB3	7	8	ANB11
HDR_ANA20_SDC0	9	10	ANB12
ANA21_SDC1	11	12	ANB13
ANA22_SDC2	13	14	ANB14
ANA23_SDC3	15	16	ANB15
HDR_ANB0_SDD0	17	18	ANB16
HDR_ANB1_SDD1	19	20	ANB17
ANB2_SDD2	21	22	ANB18
ANB3_SDD3	23	24	ANB19
ANB4_SDD4	25	26	ANB20
ANB5_SDD5	27	28	ANB21
ANB6_SDD6	29	30	ANB22
ANB7_SDD7	31	32	ANB23
Ground	33	34	Ground
Ground	35	36	Ground

Table 9. ADC header (J111)

Signal	Header pin	Header pin	Signal
HDR_AN24	1	2	HDR_ANA0_SDA0
HDR_AN25	3	4	HDR_ANA1_SDA1
HDR_AN26	5	6	ANA2_SDA2
HDR_AN27	7	8	HDR_ANA3_SDA3
HDR_AN28	9	10	HDR_ANA4
HDR_AN29	11	12	ANA5
HDR_AN30	13	14	ANA6
HDR_AN31	15	16	ANA7
HDR_AN32	17	18	ANA8
HDR_AN33	19	20	ANA9
AN34	21	22	ANA10
AN35	23	24	ANA11
AN36	25	26	ANA12
AN37	27	28	ANA13
AN38	29	30	ANA14
AN39	31	32	ANA15
Ground	33	34	Ground
Ground	35	36	Ground

3.1.7 DSPI header pin-out

Many of the DSPI signals are available on the DSPI header (J69). These can be used for additional interfaces, be careful not to conflict with the on-board DPI interfaces.

NOTE

On the MPC5775B EVB, DSPI_A and DSPI_B are connected to the Battery Management System MC33664 Isolated Network High-Speed Transceiver. DSPI_A must be configured in Slave mode for the MC33664.

Table 10. DSPI header (J69)

Signal	Header pin	Header pin	Signal
DSPI_CSA1	1	2	DSPI_CSC0
DSPI_CSA4	3	4	DSPI_CSC1
DSPI_CSB1	5	6	DSPI_CSC2
DSPI_CSB2	7	8	DSPI_CSC3
DSPI_CSB3	9	10	DSPI_CSC4
DSPI_SCKA	11	12	DSPI_CSC5
DSPI_SCKB	13	14	DSPI_SOUTA
DSPI_SCKC	15	16	DSPI_SOUTB

Table continues on the next page...

Table 10. DSPI header (J69) (continued)

Signal	Header pin	Header pin	Signal
DSPI_SINA	17	18	DSPI_SOUTC
DSPI_SINB	19	20	DSPI_SINC

3.2 MPC5775B evaluation board specific connectors

This section lists the pin-outs of all of the MPC5775B EVB specific connectors.

The only additional connector is the BMS connector.

Table 11. BMS Connector (J118)

Connector pin	Function
1[1]	FAULT_IN
2	FAULT_RTN
3	RDTX_P
4	RDTX_N

[1] The connector on the initial prototypes is mounted backwards.

3.3 MPC5775E evaluation board specific headers/connectors

This section lists the pin-outs of all of the MPC5775B EVB specific connectors and headers, the eTPU headers and the Motor Control connector

The only MPC5775E specific header are the eTPU headers and the Motor Control Connector. These headers/connectors are shown in the tables below.

Table 12. eTPU headers (J122, J123, J124)

Header pin	eTPU A header (J122)	eTPU B header (J123)	eTPU C header (J124)
1	M_eTPUA0	eTPUB0	TP52
2	M_eTPUA1	eTPUB1	TP56
3	M_eTPUA2	eTPUB2	eTPUC2
4	TP55	eTPUB3	eTPUC3
5	M_eTPUA4	eTPUB4	eTPUC4
6	M_eTPUA5	eTPUB5	TP57
7	M_eTPUA6	eTPUB6	TP58
8	M_eTPUA7	eTPUB7	eTPUC8
9	M_eTPUA8	eTPUB8	eTPUC9
10	M_eTPUA9	eTPUB9	eTPUC10

Table continues on the next page...

Table 12. eTPU headers (J122, J123, J124) (continued)

Header pin	eTPU A header (J122)	eTPU B header (J123)	eTPU C header (J124)
11	M_eTPUA10	eTPUB10	eTPUC11
12	TCRCLKA	eTPUB11	TCRCLKC
13	M_eTPUA12	eTPUB12	TCRCLKB
14	M_eTPUA13	eTPUB13	Ground
15	M_eTPUA14	eTPUB14	Ground
16	M_eTPUA15	eTPUB15	eTPUC15
17	M_eTPUA16	eTPUB16	eTPUC16
18	M_eTPUA17	eTPUB17	eTPUC17
19	M_eTPUA18	eTPUB18	eTPUC18
20	M_eTPUA19	eTPUB19	Ground
21	M_eTPUA20	eTPUB20	Ground
22	M_eTPUA21	eTPUB21	eTPUC21
23	TP51	eTPUB22	eTPUC22
24	Ground	eTPUB23	eTPUC23
25	Ground	eTPUB24	eTPUC24
26	Ground	eTPUB25	eTPUC25
27	Ground	eTPUB26	eTPUC26
28	eTPUA27	eTPUB27	eTPUC27
29	eTPUA28	eTPUB28	eTPUC28
30	eTPUA29	eTPUB29	eTPUC29
31	eTPUA30	eTPUB30	eTPUC30
32	eTPUA31	eTPUB31	eTPUC31

The MPC5775E EVB supports the NXP Motor Control Driver board. It uses a PCIe x4 connector. The pin-out of this connector is shown in the table below.

Table 13. Motor Control Edge Connector (J114)

Signal	Connector pin	Connector pin	Signal
Analog reference voltage (5V)	B1	A1	Analog supply voltage (5V)
Analog ground	B2	A2	Analog ground
M_AN27	B3	A3	M_AN24
M_AN28	B4	A4	M_AN25
M_AN29	B5	A5	M_AN26
M_AN30	B6	A6	M_AN32
M_AN31	B7	A7	M_AN33
M_ANA1_SDA1	B8	A8	M_ANA0_SDA0
M_ANB1_SDD1	B9	A9	M_ANB0_SDD0
M_ANA20_SDC0	B20	A10	M_ANA16_SDB0
Analog ground	B11	A11	Analog ground

Table continues on the next page...

Table 13. Motor Control Edge Connector (J114) (continued)

Signal	Connector pin	Connector pin	Signal
SBC_5V	B12	A12	CAN 5V
Ground	B13	A13	Ground
M_eTPUC5	B14	A14	M_eTPUA16
M_eTPUC6	B15	A15	M_eTPUA17
M_eTPUC7	B16	A16	M_eTPUA18
M_eTPUA4	B17	A17	M_eTPUA19
M_eTPUA5	B18	A18	M_eTPUA20
M_eTPUA6	B19	A19	M_eTPUA21
M_IRQ11	B20	A20	M_eTPUA14
DSPI_SINB	B21	A21	M_eTPUA15
DSPI_SOUTB	B22	A22	M_eTPUA7
DSPI_SCKB	B23	A23	M_eTPUA8
DSPI_CS3	B24	A24	M_eTPUA9
	B25	A25	M_eTPUA10
TXDB	B26	A26	M_IRQ0
RXDB	B27	A27	M_IRQ1
M_eTPUA12	B28	A28	M_IRQ10
M_eTPUA13	B29	A29	M_IRQ13
M_IRQ12	B30	A30	MeTPUA28
VPOWER[1]	B31	A31	M_IRQ3
Ground	B32	A32	M_IRQ4

[1] Not connected by default. R187 can be installed to connect to VSUP (protected 12V)

4 Configuring the board

This section shows how to configure the MPC5775B-EVB and MPC5775E-EVB boards for the initial test project.

1. The 12 V power supply should be connected to the board (J62).
2. Check that the power supply LEDs are illuminated. There are four (4) power supply LEDs,

Table 14. Power indicator LEDs

LED	Indication
D32	12 V
D15	5 V
D14	3V3 V
D16	1V2 V

3. Connect the microUSB connector on the board (note, the connector is on the bottom of the board) to a computer USB port.

4. The simple "hello_world" application uses the eMIOS1/GPIO180 signal to blinks an LED. On the EVB, the eMIOS1 signal is available on pin 2 of the EMIOS connector (J125). Connect a jumper wire between this pin and one of the LEDs that is available on JP6.

The following figure shows a photo of the MPC5775B EVB with a 12 V power supply connected, the microUSB connected, and the LED connection between J125 pin 2 and pin 5 of JP6 (blue LED).

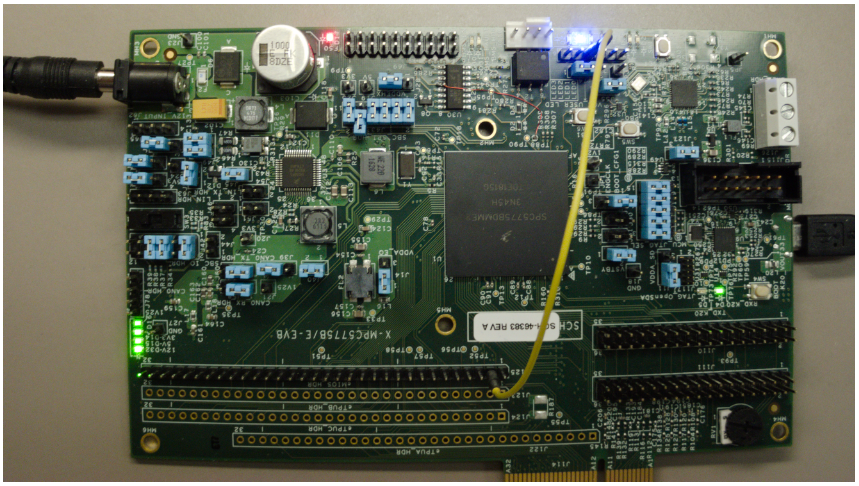
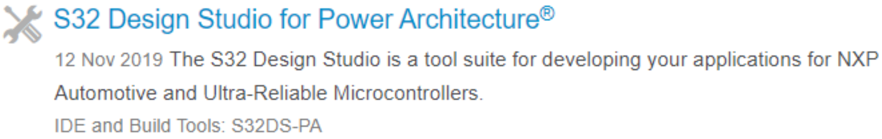


Figure 5. Configured MPC5775B-EVB

5 Getting and installing S32 Design Studio

The S32 Design Studio for Power Architecture is available for download on the NXP web site.

1. Search for "S32 Design Studio Power Architecture" to find the product page



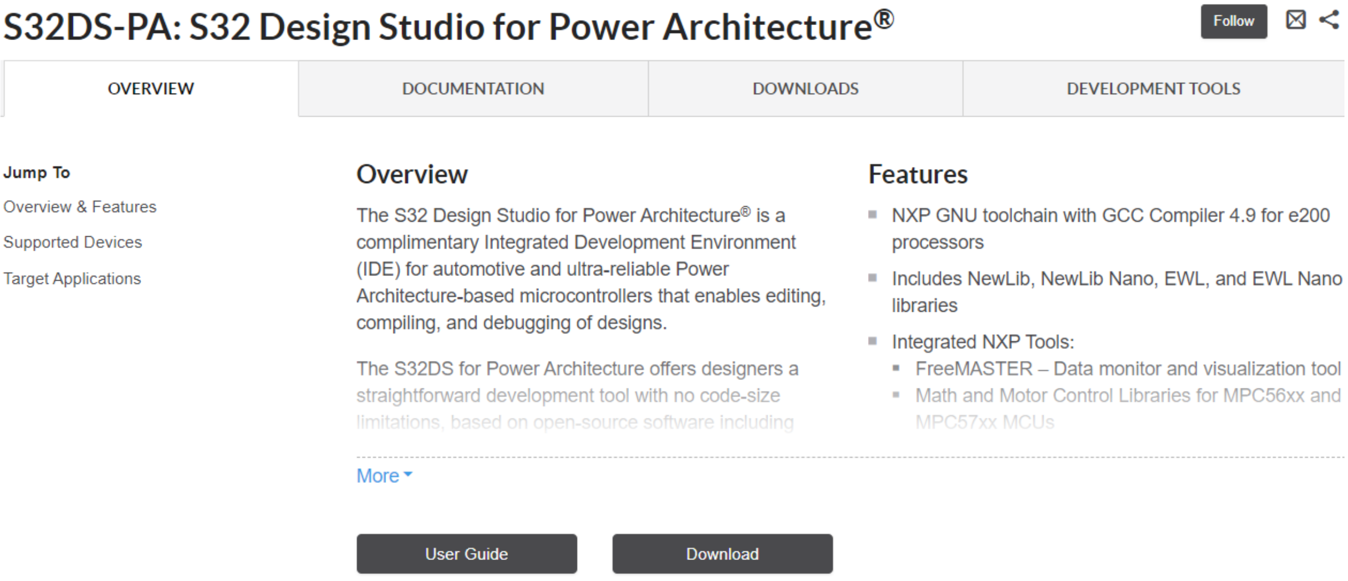
S32 Design Studio for Power Architecture®

12 Nov 2019 The S32 Design Studio is a tool suite for developing your applications for NXP Automotive and Ultra-Reliable Microcontrollers.

IDE and Build Tools: S32DS-PA

Figure 6. Search for S32 Design Studio for Power Architecture

2. Select the product page



S32DS-PA: S32 Design Studio for Power Architecture®

Overview

The S32 Design Studio for Power Architecture® is a complimentary Integrated Development Environment (IDE) for automotive and ultra-reliable Power Architecture-based microcontrollers that enables editing, compiling, and debugging of designs.

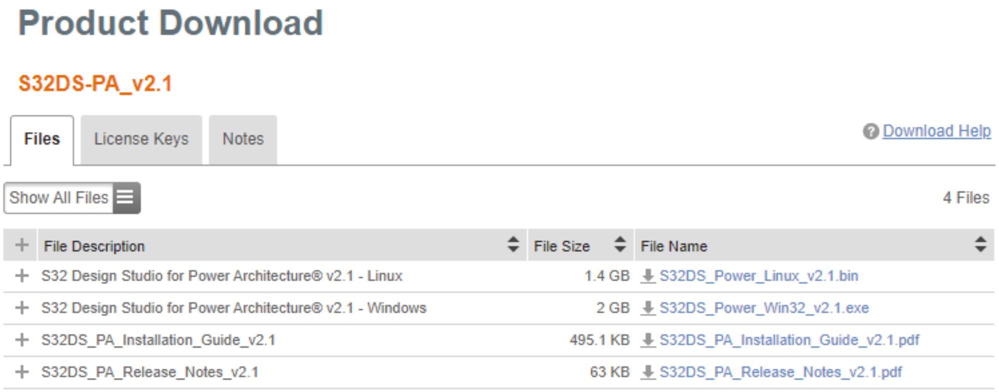
The S32DS for Power Architecture offers designers a straightforward development tool with no code-size limitations, based on open-source software including

Features

- NXP GNU toolchain with GCC Compiler 4.9 for e200 processors
- Includes NewLib, NewLib Nano, EWL, and EWL Nano libraries
- Integrated NXP Tools:
  - FreeMASTER – Data monitor and visualization tool
  - Math and Motor Control Libraries for MPC56xx and MPC57xx MCUs

Figure 7. S32 Design Studio for Power Architecture product page

3. Go to the download page and select the proper installer (windows or Linux). Also see the "Release notes and installation guide that are also available to download.



Product Download

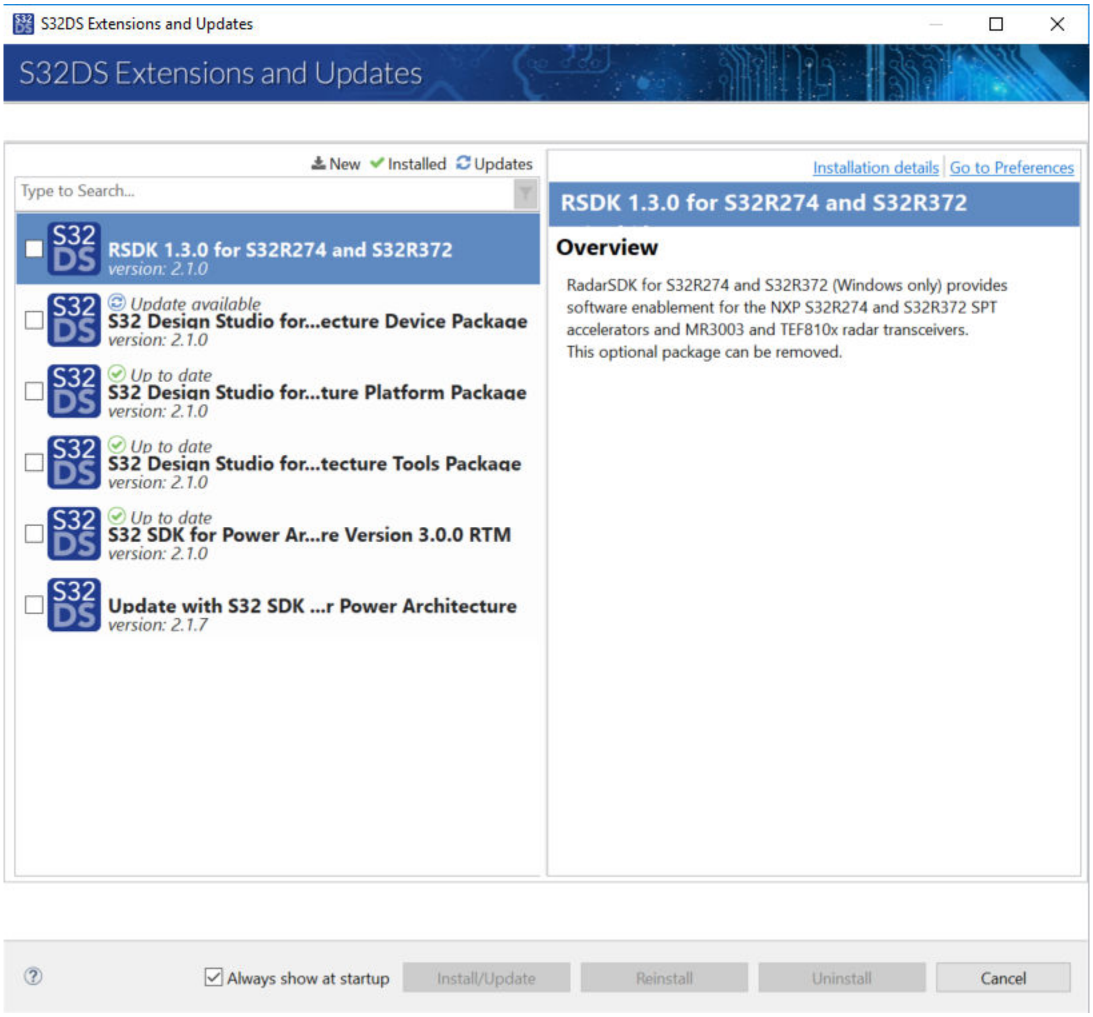
S32DS-PA_v2.1

File Description	File Size	File Name
S32 Design Studio for Power Architecture® v2.1 - Linux	1.4 GB	S32DS_Power_Linux_v2.1.bin
S32 Design Studio for Power Architecture® v2.1 - Windows	2 GB	S32DS_Power_Win32_v2.1.exe
S32DS_PA_Installation_Guide_v2.1	495.1 KB	S32DS_PA_Installation_Guide_v2.1.pdf
S32DS_PA_Release_Notes_v2.1	63 KB	S32DS_PA_Release_Notes_v2.1.pdf

Figure 8. Installer file options

4. Accept the license terms. Once the download begins, a software activation code will be sent to your registered email address.
5. Run the installer. To install both S32 Design Studio and FreeMASTER, select "Additionally Install".
6. The activation code that was emailed to your email address will need to be entered. Select online activation.

When the NXP Design Studio for Power Architecture is launched, it will check for any updates and if updates are available, give the option to download and install the updates. It also shows the packages that are up to date. In the following example, there is one (1) update and two (2) new packages available. These can be installed by selecting them and performing the install.



□ RSDK 1.3.0 for S32R274 and S32R372
version: 2.1.0

□ Update available
S32 Design Studio for...ecture Device Package
version: 2.1.0

□ Up to date
S32 Design Studio for...ture Platform Package
version: 2.1.0

□ Up to date
S32 Design Studio for...tecture Tools Package
version: 2.1.0

□ Up to date
S32 SDK for Power Ar...re Version 3.0.0 RTM
version: 2.1.0

□ Update with S32 SDK ...r Power Architecture
version: 2.1.7

RSDK 1.3.0 for S32R274 and S32R372

Overview

RadarSDK for S32R274 and S32R372 (Windows only) provides software enablement for the NXP S32R274 and S32R372 SPT accelerators and MR3003 and TEF810x radar transceivers.
This optional package can be removed.

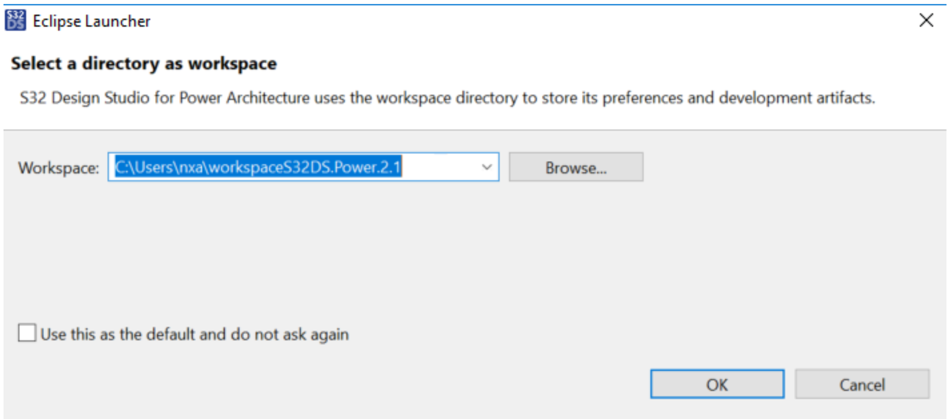
☑ Always show at startup

Figure 9. Typical update available notification

6 Configuring and building software from an example

This section walks through building and configuring a initial project.

1. Run Design Studio for Power Architecture.
2. Make or select a workspace.



Eclipse Launcher

Select a directory as workspace

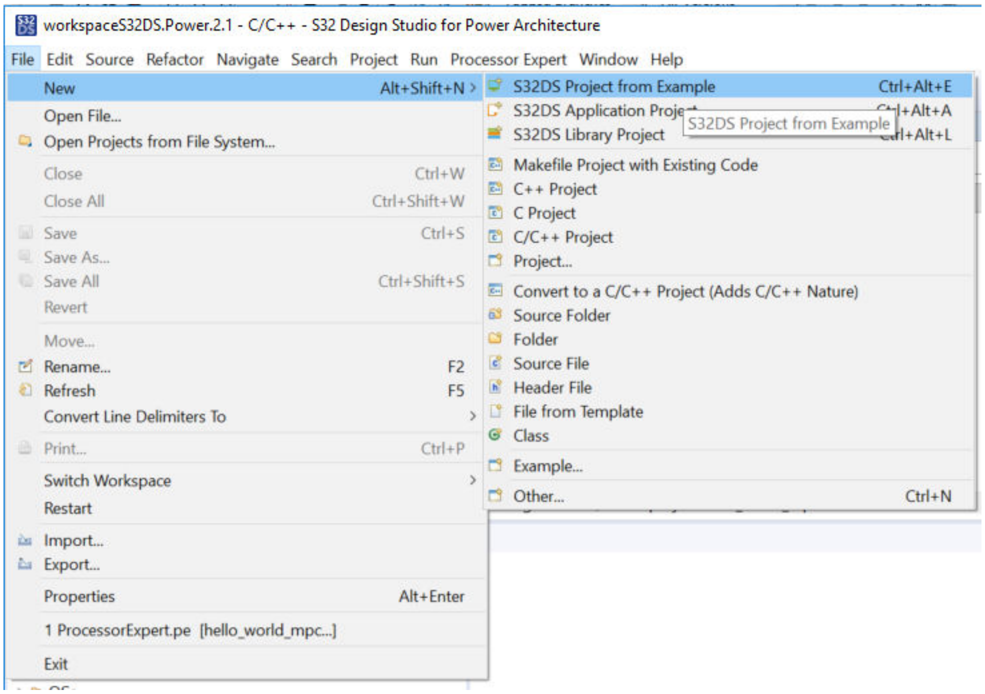
S32 Design Studio for Power Architecture uses the workspace directory to store its preferences and development artifacts.

Workspace: C:\Users\nxa\workspaceS32DS.Power.2.1

☐ Use this as the default and do not ask again

Figure 10. Set up a workspace

3. Start a "New" "S32DS Project from Example" from the File menu.

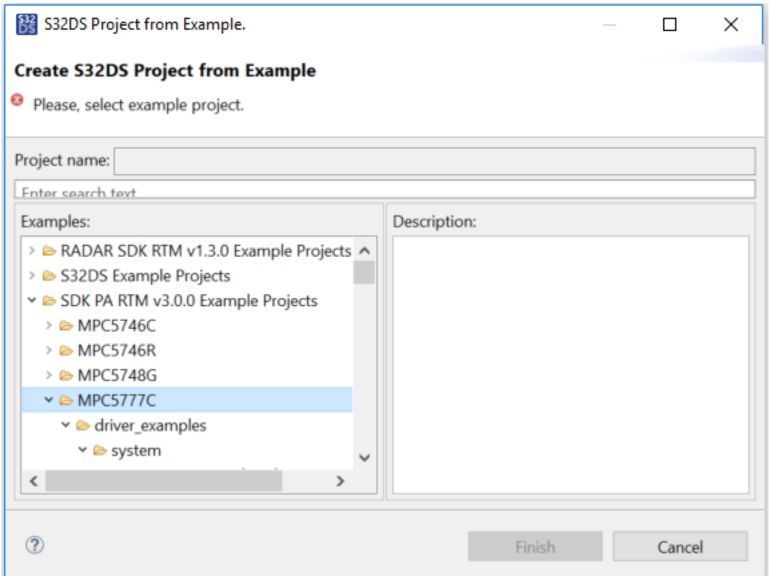


workspaceS32DS.Power.2.1 - C/C++ - S32 Design Studio for Power Architecture

File Edit Source Refactor Navigate Search Project Run Processor Expert Window Help
New Alt+Shift+N >
Open File...
Open Projects from File System...
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To >
Print... Ctrl+P
Switch Workspace >
Restart
Import...
Export...
Properties Alt+Enter
1 ProcessorExpert.pe [hello_world_mpc...]
Exit
S32DS Project from Example Ctrl+Alt+E
S32DS Application Project Ctrl+Alt+A
S32DS Project from Example
S32DS Library Project Ctrl+Alt+L
Makefile Project with Existing Code
C++ Project
C Project
C/C++ Project
Project...
Convert to a C/C++ Project (Adds C/C++ Nature)
Source Folder
Folder
Source File
Header File
File from Template
Class
Example...
Other... Ctrl+N

Figure 11. New project from an example

4. Scroll down to the "SDK PA RTM v3.0.0 Example Projects", select "MPC5777C" and then select "Hello_world_mpc5777c"

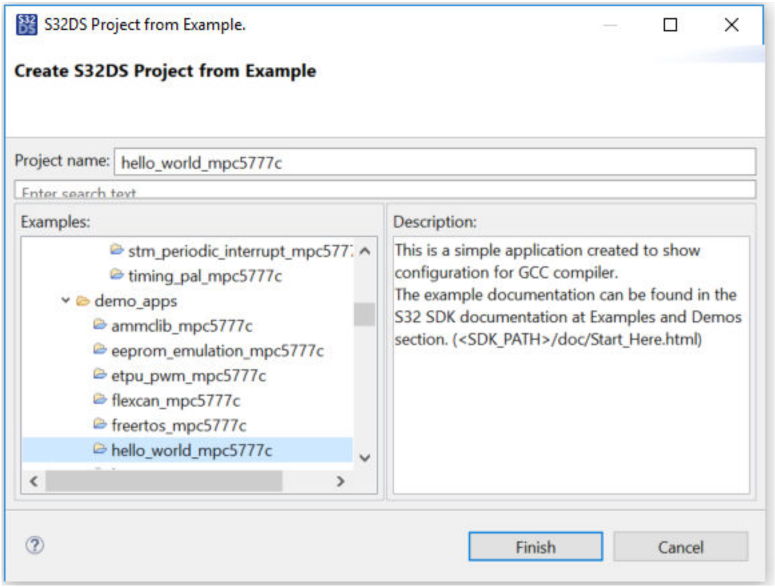

S32DS Project from Example.

Create S32DS Project from Example

Please, select example project.

Project name:

Figure 12. Scroll to MPC5777C


S32DS Project from Example.

Create S32DS Project from Example

Project name: hello_world_mpc5777c

This is a simple application created to show configuration for GCC compiler.
The example documentation can be found in the S32 SDK documentation at Examples and Demos section. (<SDK_PATH>/doc/Start_Here.html)

Figure 13. Select hello_world_mpc5777c

5. Once the initial create is complete, the project window will be similar to the below figure.

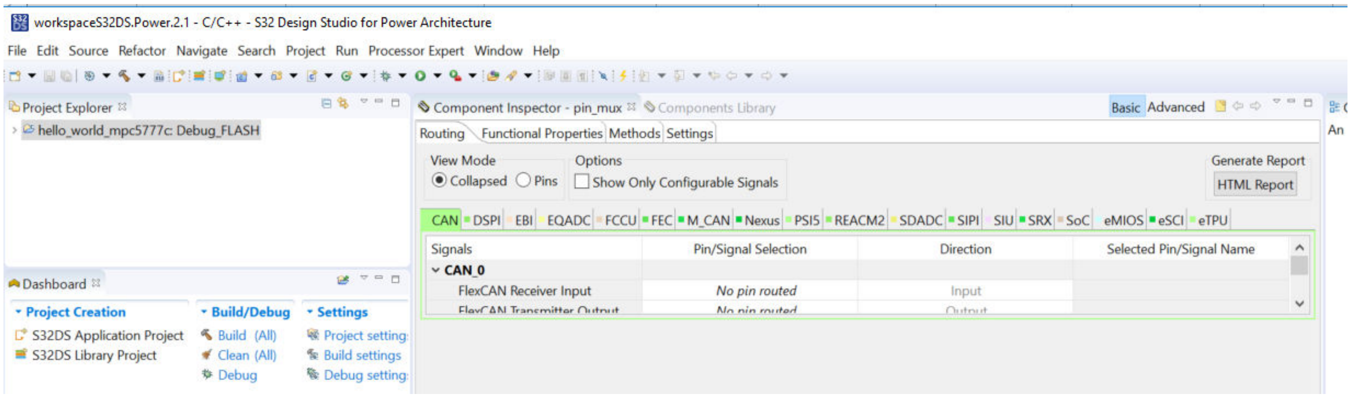

workspaceS32DS.Power.2.1 - C/C++ - S32 Design Studio for Power Architecture
File Edit Source Refactor Navigate Search Project Run Processor Expert Window Help
Project Explorer
hello_world_mpc5777c: Debug_FLASH
Dashboard
Project Creation
S32DS Application Project
S32DS Library Project
Build/Debug
Build (All)
Clean (All)
Debug
Settings
Project setting
Build settings
Debug setting
Component Inspector - pin_mux
Components Library
Basic Advanced
Routing Functional Properties Methods Settings
View Mode
● Collapsed ○ Pins
Options
□ Show Only Configurable Signals
Generate Report
HTML Report
CAN DSPI EBI EQADC FCCU FEC M_CAN Nexus PSI5 REACM2 SDADC SIPI SIU SRX SoC eMIOS eSCI eTPU
Signals
Pin/Signal Selection
Direction
Selected Pin/Signal Name
CAN_0
FlexCAN Receiver Input
No pin routed
Input
FlexCAN Transmitter Output
No pin routed
Output

Figure 14. Initial project view

6. If the MPC5775E or MPC5775B is not supported (at least through S32DS version 2.1.7), additional steps need to occur to change the MPC5777C to the MPC5775E or MPC5775B, see Change MPC5777C 516-pin project to MPC5775E or MPC5775B.
7. Select "Build all" in the "Project" menu to compile the program.

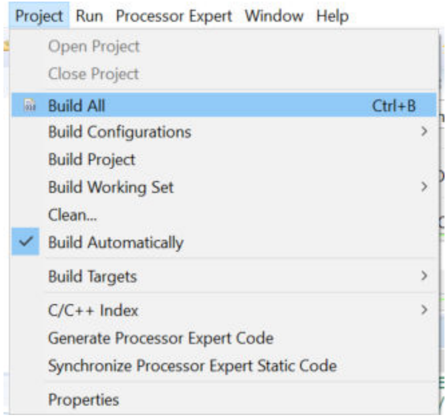


Figure 15. Build project menu

The example project is now compiled and can be executed.

7 Debugging and controlling the application

See Configuring the board for configuring the EVB for use with the example "hello_world" application.

After the program is compiled, the following steps should be performed in S32 Design Studio for Power Architecture,

1. Open "Debug Configurations" in the "Run menu.

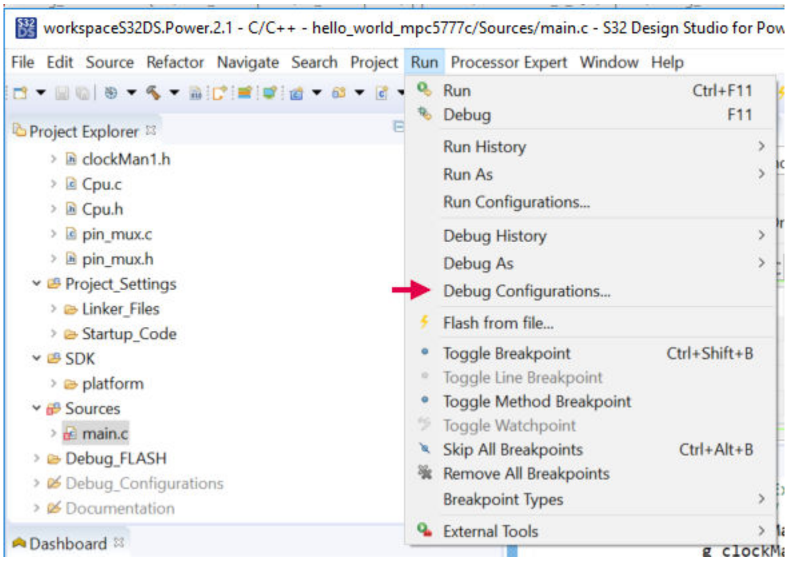
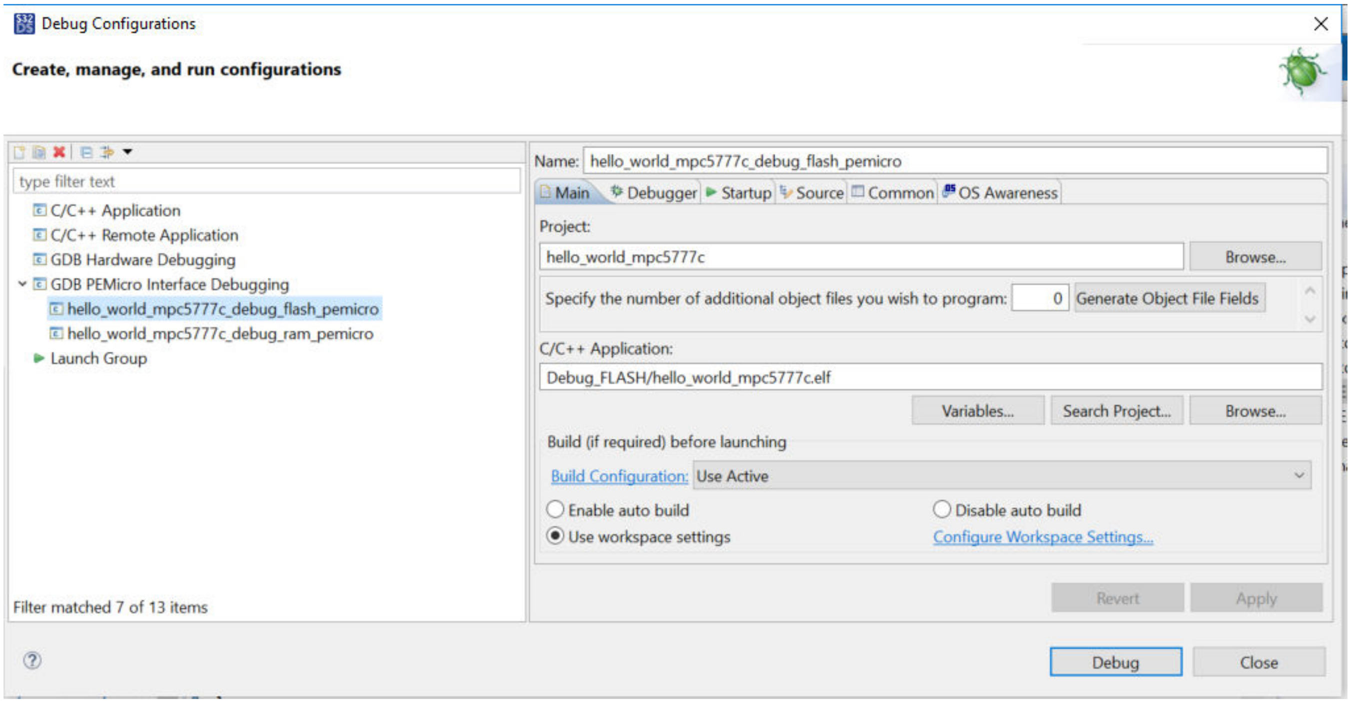


Figure 16. Select "Debug Configurations"

2. The "Debug Configurations" window allows the executable code to be selected, as well as configures the debug interface. First select the debug interface, select "GDB PEMicro Interface Debugging". Select either the flash or the SRAM image, depending on the mode selected (by default, the Flash version is compiled). Also, the proper executable is preselected.


Debug Configurations

Create, manage, and run configurations

Name: hello_world_mpc5777c_debug_flash_pemicro

Project: hello_world_mpc5777c

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

C/C++ Application: Debug_FLASH/hello_world_mpc5777c.elf

Build (if required) before launching

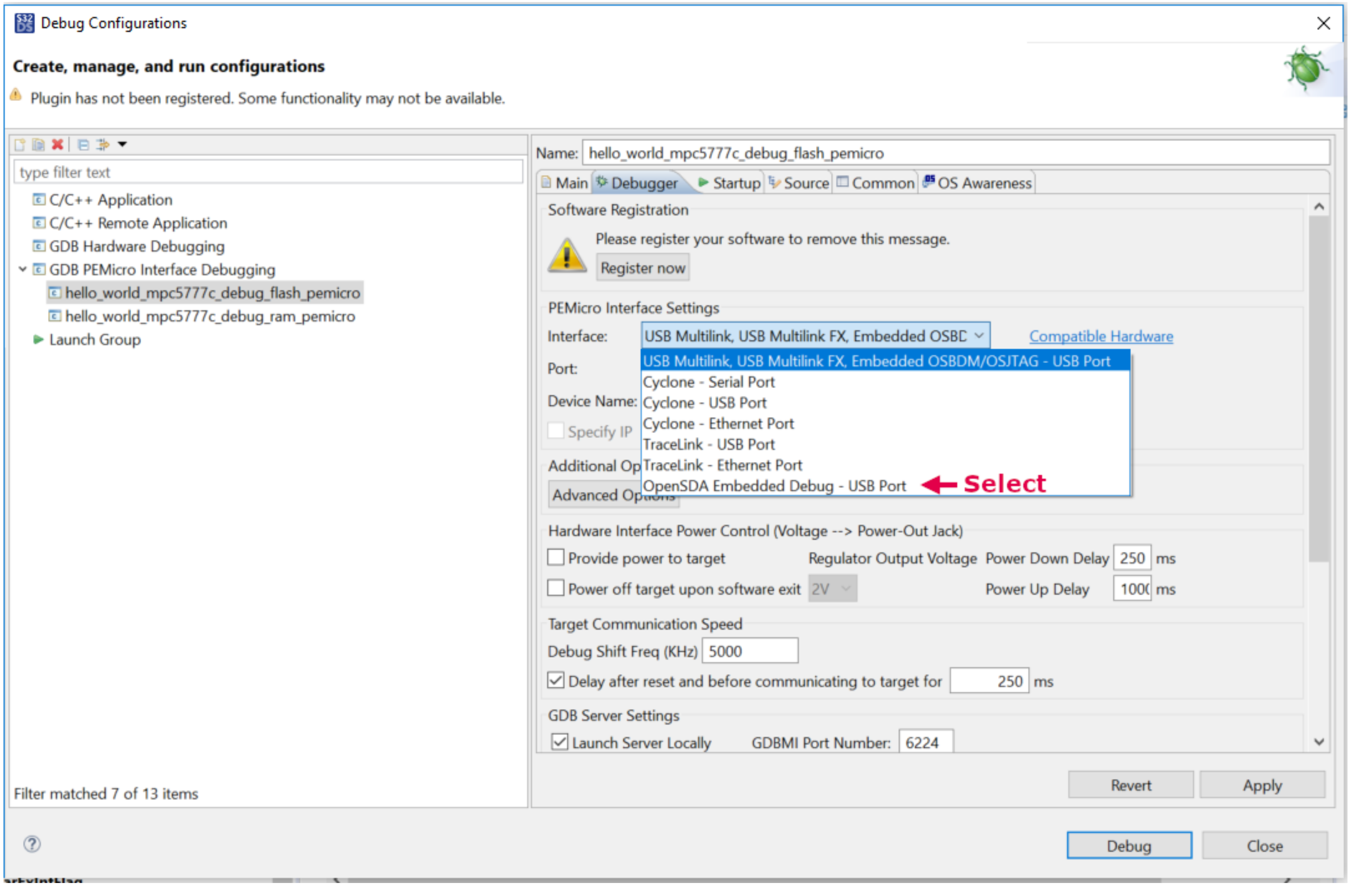
Build Configuration: Use Active

○ Enable auto build ○ Disable auto build

● Use workspace settings

Figure 17. Debug Configurations Window

3. Select the "Debugger" tab. By default, the "USB Multilink" is selected. This should be changed to "OpenSDA Embedded Debug - USB".



Debug Configurations

Create, manage, and run configurations

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

Name: hello_world_mpc5777c_debug_flash_pemicro

Software Registration

Please register your software to remove this message.

PEMicro Interface Settings

Interface: USB Multilink, USB Multilink FX, Embedded OSBD

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

□ Provide power to target Regulator Output Voltage Power Down Delay 250 ms

□ Power off target upon software exit 2V Power Up Delay 1000 ms

Target Communication Speed

Debug Shift Freq (KHz) 5000

☑ Delay after reset and before communicating to target for 250 ms

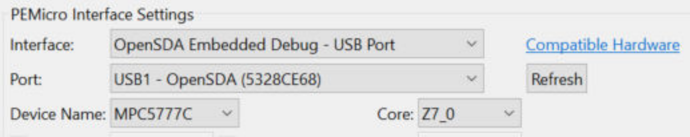
GDB Server Settings

☑ Launch Server Locally GDBMI Port Number: 6224

Filter matched 7 of 13 items

Figure 18. Select "OpenSDA Embedded Debug"

4. After selecting the "OpenSDA Embedded Debug" interface, if the EVB is connected to the computer, the "Port"should automatically be selected.



PEMicro Interface Settings

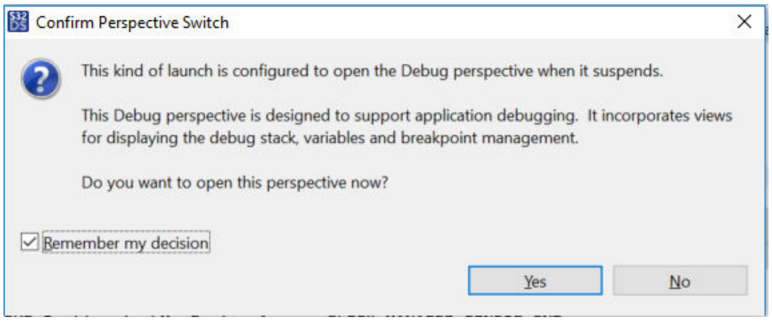
Interface: OpenSDA Embedded Debug - USB Port

Port: USB1 - OpenSDA (5328CE68)

Device Name: MPC5777C Core: Z7_0

Figure 19. Port selection

5. Click "Apply", then "Debug". The first time this is selected, DS32 will request confirmation of the change in the perspective. (The perspective is the sub-window views that will be available.) Normally, the debug perspective should always be selected when debugging, so the "Remember my decision"should be selected.



Confirm Perspective Switch

This kind of launch is configured to open the Debug perspective when it suspends.

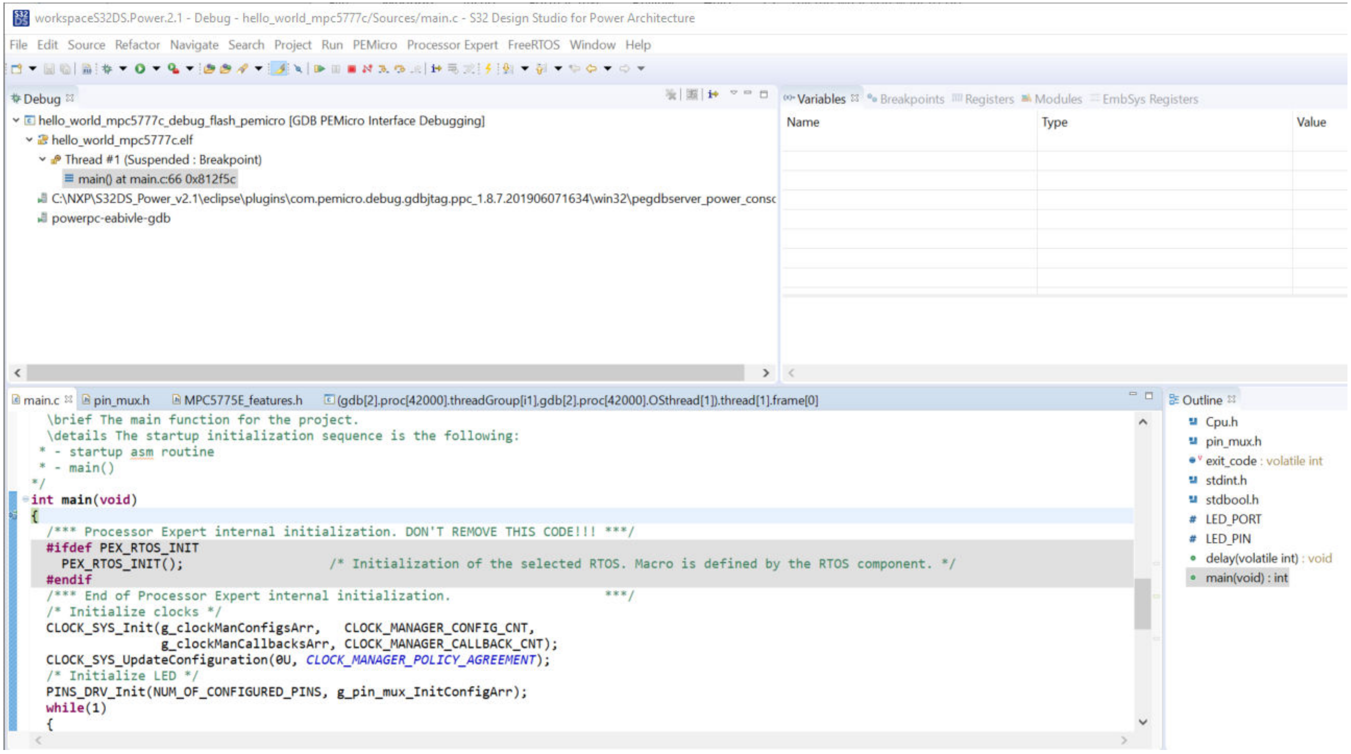
This Debug perspective is designed to support application debugging. It incorporates views for displaying the debug stack, variables and breakpoint management.

Do you want to open this perspective now?

☑ Remember my decision

Figure 20. Confirm Perspective

6. If the flash image is selected, then the example application will be programmed into the MCU internal flash and the debugger will run up to "main"of the application.


workspaceS32DS.Power.2.1 - Debug - hello_world_mpc5777c/Sources/main.c - S32 Design Studio for Power Architecture
File Edit Source Refactor Navigate Search Project Run PEMicro Processor Expert FreeRTOS Window Help
Debug
Variables
Breakpoints
Registers
Modules
EmbSys Registers
Name
Type
Value
main.c
pin_mux.h
MPC5775E_features.h
(gdb[2].proc[42000].threadGroup[i1].gdb[2].proc[42000].OSthread[1]).thread[1].frame[0]
Outline
Cpu.h
pin_mux.h
exit_code : volatile int
stdint.h
stdbool.h
LED_PORT
LED_PIN
delay(volatile int) : void
main(void) : int

```
hello_world_mpc5777c_debug_flash_pemicro [GDB PEMicro Interface Debugging]
  hello_world_mpc5777c.elf
    Thread #1 (Suspended : Breakpoint)
      main() at main.c:66 0x812f5c
  C:\NXP\S32DS_Power_v2.1\eclipse\plugins\com.pemicro.debug.gdbjtag.ppc_1.8.7.201906071634\win32\pegdbserver_power_consc
  powerpc-eabivle-gdb
```

```
  \brief The main function for the project.
  \details The startup initialization sequence is the following:
 * - startup asm routine
 * - main()
*/
int main(void)
{
  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ***/
  #ifdef PEX_RTOS_INIT
    PEX_RTOS_INIT();                   /* Initialization of the selected RTOS. Macro is defined by the RTOS component. */
  #endif
  /*** End of Processor Expert internal initialization.                    ***/
  /* Initialize clocks */
  CLOCK_SYS_Init(g_clockManConfigsArr,   CLOCK_MANAGER_CONFIG_CNT,
                 g_clockManCallbacksArr, CLOCK_MANAGER_CALLBACK_CNT);
  CLOCK_SYS_UpdateConfiguration(0U, CLOCK_MANAGER_POLICY_AGREEMENT);
  /* Initialize LED */
  PINS_DRV_Init(NUM_OF_CONFIGURED_PINS, g_pin_mux_InitConfigArr);
  while(1)
  {
```

Figure 21. Initial debugger window

7. From this point, instructions can be single-stepped (Step Into), functions stepped over (Step Over), run (Resume), stopped (Terminate) or breakpoints set.

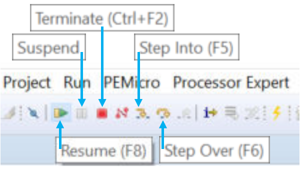
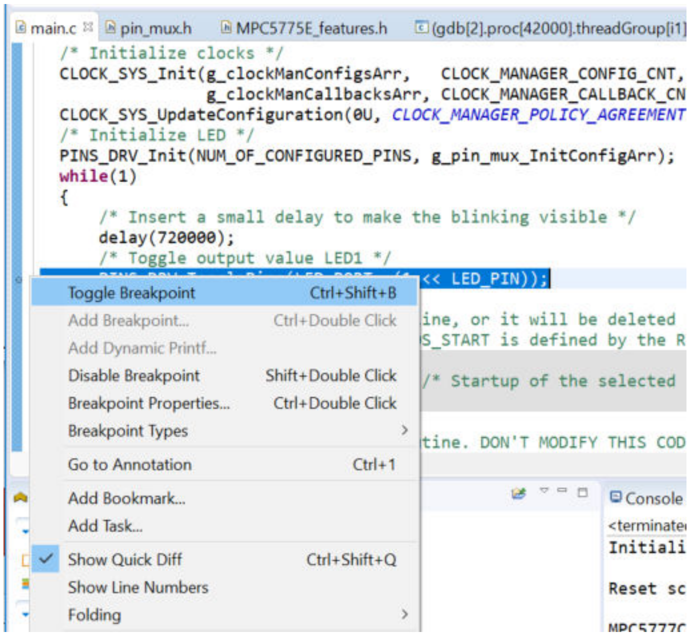


Figure 22. DS32 debug controls

8. A breakpoint can be set by right-clicking on the line of code where the breakpoint is desired and selecting "Toggle Breakpoint", as shown in the following figure.


main.c
pin_mux.h
MPC5775E_features.h
(gdb[2].proc[42000].threadGroup[i1]
Toggle Breakpoint Ctrl+Shift+B
Add Breakpoint... Ctrl+Double Click
Add Dynamic Printf...
Disable Breakpoint Shift+Double Click
Breakpoint Properties... Ctrl+Double Click
Breakpoint Types
Go to Annotation Ctrl+1
Add Bookmark...
Add Task...
Show Quick Diff Ctrl+Shift+Q
Show Line Numbers
Folding
Console
<terminate

```
/* Initialize clocks */
CLOCK_SYS_Init(g_clockManConfigsArr,   CLOCK_MANAGER_CONFIG_CNT,
               g_clockManCallbacksArr, CLOCK_MANAGER_CALLBACK_CN
CLOCK_SYS_UpdateConfiguration(0U, CLOCK_MANAGER_POLICY_AGREEMENT
/* Initialize LED */
PINS_DRV_Init(NUM_OF_CONFIGURED_PINS, g_pin_mux_InitConfigArr);
while(1)
{
    /* Insert a small delay to make the blinking visible */
    delay(720000);
    /* Toggle output value LED1 */
                                  << LED_PIN));
                                  ine, or it will be deleted
                                  S_START is defined by the R
                                  /* Startup of the selected
                                  tine. DON'T MODIFY THIS COD
```

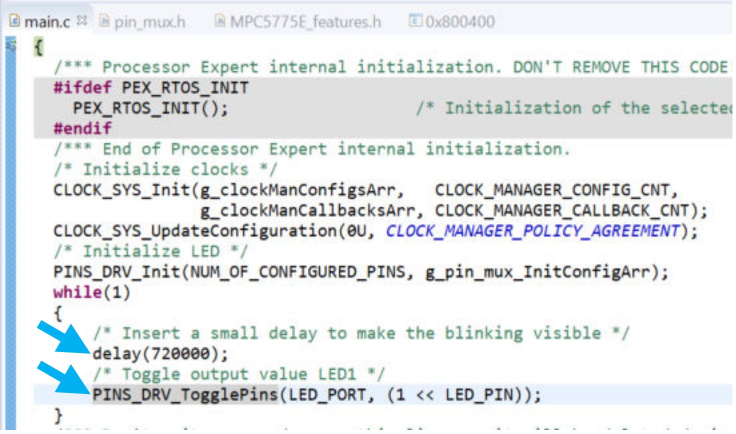
```
Initiali
Reset sc
MPC5777C
```

Figure 23. Setting a breakpoint

7.1 Breakpoint special cases

In some cases, based on the optimization level selected for the S32 Design Studio compiler, breakpoints may not be recognized by the debugger. This also depends on the exact placement of the breakpoint. In these cases, the breakpoint may be recognized the first time set, but will not stop the device if the code is continued and the address of the breakpoint is again reached.

In the "hello_word" example, this can occur if a breakpoint is set at either of the two "C" statements inside the main loop.



```
{
  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE
  #ifdef PEX_RTOS_INIT
    PEX_RTOS_INIT();                  /* Initialization of the selecte
  #endif
  /*** End of Processor Expert internal initialization.
  /* Initialize clocks */
  CLOCK_SYS_Init(g_clockManConfigsArr,   CLOCK_MANAGER_CONFIG_CNT,
                 g_clockManCallbacksArr, CLOCK_MANAGER_CALLBACK_CNT);
  CLOCK_SYS_UpdateConfiguration(0U, CLOCK_MANAGER_POLICY_AGREEMENT);
  /* Initialize LED */
  PINS_DRV_Init(NUM_OF_CONFIGURED_PINS, g_pin_mux_InitConfigArr);
  while(1)
  {
      /* Insert a small delay to make the blinking visible */
      delay(720000);
      /* Toggle output value LED1 */
      PINS_DRV_TogglePins(LED_PORT, (1 << LED_PIN));
  }
```

Figure 24. "hello_world" example listing

This can be worked around by setting the breakpoints in a disassembly window of the code. To show the disassembly view, go to the Disassembly menu item under with Window menu item Perspective.

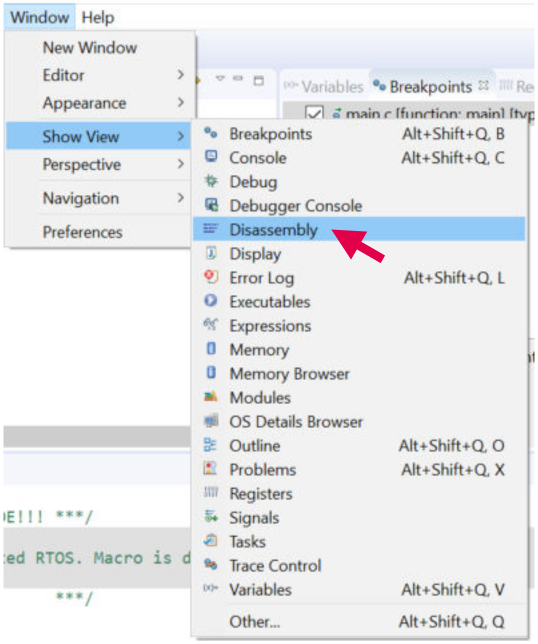
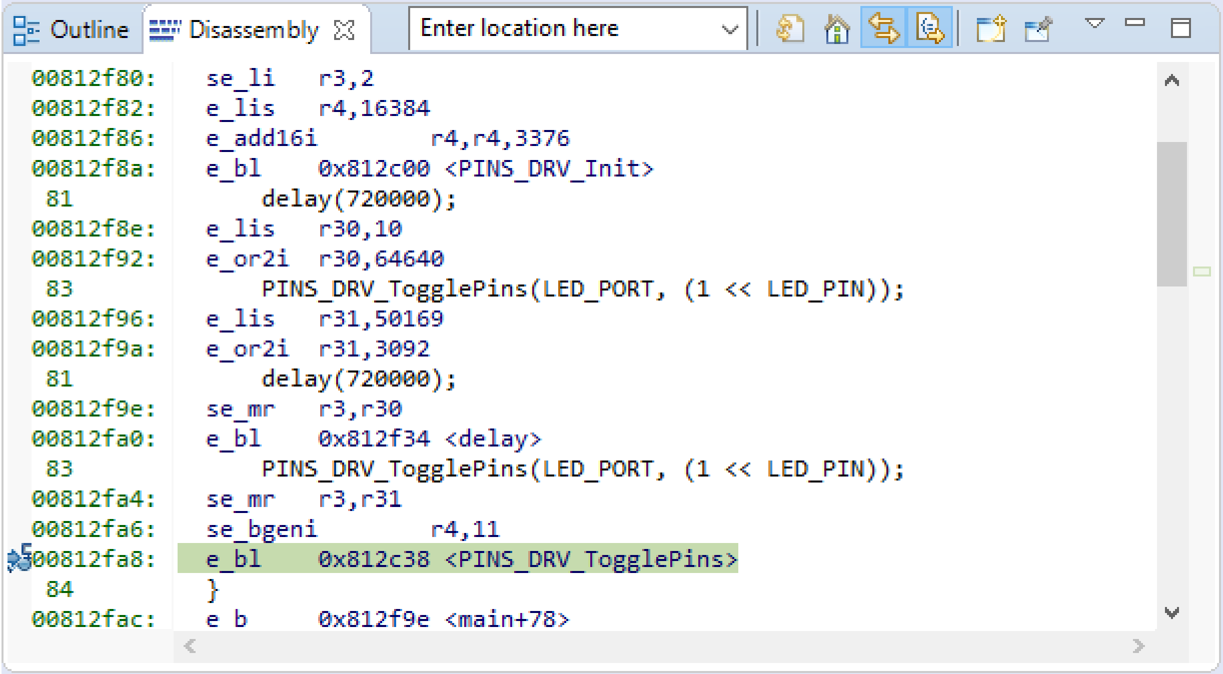


Figure 25. Opening the disassembly window

In the disassembly window, scroll down to address 0x0x0081_2FA8 and double click next to the assembly instruction to place a breakpoint. It appears in the breakpoint window. This is at the assembly instruction that toggles the pin.



```
00812f80:   se_li   r3,2
00812f82:   e_lis   r4,16384
00812f86:   e_add16i        r4,r4,3376
00812f8a:   e_bl    0x812c00 <PINS_DRV_Init>
 81             delay(720000);
00812f8e:   e_lis   r30,10
00812f92:   e_or2i  r30,64640
 83             PINS_DRV_TogglePins(LED_PORT, (1 << LED_PIN));
00812f96:   e_lis   r31,50169
00812f9a:   e_or2i  r31,3092
 81             delay(720000);
00812f9e:   se_mr   r3,r30
00812fa0:   e_bl    0x812f34 <delay>
 83             PINS_DRV_TogglePins(LED_PORT, (1 << LED_PIN));
00812fa4:   se_mr   r3,r31
00812fa6:   se_bgeni        r4,11
00812fa8:   e_bl    0x812c38 <PINS_DRV_TogglePins>
 84         }
00812fac:   e_b     0x812f9e <main+78>
```

Figure 26. Disassembly window with a breakpoint

Appendix A References

The following documentation is available to support the MPC5775B and the MPC5775E.

Table A-1. MPC5775B and Reference information major differences

Document Number	Description	Location
MPC5775E RM	MPC5775E/MPC5775B Reference Manual	NXP web site
MPC5775E	MPC577E/MPC5775B	NXP web site
MPC5775B-EVB	MPC5775B-EVB product page	NXP web site
MPC5775E-EVB	MPC577E-EVB product page	NXP web site
OpenSDA	OpenSDA User's guide	NXP web site
S32DS	S32 Design Studio for Power Architecture v2.1 - Windows/ Linux	NXP web site
MC33FS6520LAE	MC33FS6520 System Basis Chip (Power supply and drivers) Data Sheet	NXP web site[1]
FS65SBC-SDK-SW	FS6500/FS4500 Generic Embedded Software Driver (Software Development Kit)	NXP web site
MC33664_SDS	Isolated Network High-Speed Transceiver Short Data Sheet	NXP web site
MC33664	Isolated Network High-Speed Transceiver Full Data Sheet	NXP DocStore web site[2]
EMBEDDED-SW-BCCyes	MC33771B/MC33772B Embedded Cell Controller Software Driver	NXP web site
HM2102NL	Pulse Electronics HM2102NL Dual BMS transformer	Pulse Electronics web site
HM2103NL	Pulse Electronics HM2102NL Single BMS transformer	Pulse Electronics web site

[1] Requires NXP Sales or FAE approval for access.

[2] Requires registration and approval for access to full data sheet

Appendix B OpenSDA interface and JTAG

The MPC57775B and MPC5775E EVBs include an on-chip USB to JTAG converter (OpenSDA). This allows a computer to connect via a USB interface to the JTAG port of the MCU. The OpenSDA interface is built around the NXP MK20DX128VFM5 (Kinetis K20 50 MHz, Full Speed USB, 160KB Flash Microcontroller based on the Arm Cortex-M4 Core). This interface is programmed with a boot-loader and OpenSDA firmware that communicates between the computer host and the MCU over the JTAG interface. This interface is supported by the NXP S32 Design Studio software.

The boot-loader supports updates of the CMSIS-DAP firmware by connecting to a computer’s USB port and powering up the OpenSDA (plug in the microUSB connector) with SW4 pushed. An LED (D4))will start blinking. A drag and drop of the updated firmware onto the board will update the firmware. J116 allows JTAG access to the K20 MCU.

By default, OpenSDA is selected. If an external JTAG interface is required, then the J119 jumpers need to be changed to use J115 as the JTAG connection to the debugger/tool.

Table B-1. J119 JTAG pin selection (OpenSDA or external tool)

Signal	For the 14-pin JTAG external connector (J115), jumper setting	OpenSDA setting
TDI	connect pins 1 to 2	connect pins 2 to 3
TDO	4 to 5	5 to 6
TCK	7 to 8	8 to 9
TMS	10 to 11	11 to 12
RST_HDR_B	13 to 14	14 to 15

Additionally, the Open SDA interface supports a USB to serial interface that is connected to eSCI_A, pins RXDA/GPIO90 and TXDA/GPIO89.

This UART can be used even when the Open SDA JTAG interface is not used.

Appendix C BMS usage

Circuitry is provided on the MPC5775B-EVB to support a Battery Management System interface. The hardware consists of a driver (MC33664ATL1EG) that converts two SPI channels into to a 2-wire high speed (2Mbps) isolated communication channel using pulse bit information. This interface connects to a battery controller, such as the MC3377x battery cell controller, frequently used in hybrid vehicle batteries. DSPI_B Chip Select 0 is used for the master channel (DSPI_0) and DSPI_A is used in slave mode for the receive channel. Slave mode requires that Chip Select 0 (slave Select) be used on DSPI_A. DSPI_B could be utilized in the system for other SPI peripherals using different chip selects.

C.1 BMS overview

The Battery management system (BMS) consists of an NXP MC33664ATL1EG Isolated network high-speed transceiver (located on the MPC5775B-EVB board) to communicate to one or more MC33771B 14-channel Li-ion Battery Cell Controller ICs or MC33772B 6-channel Li-ion Battery Cell Controller ICs. This is shown in the following figure.

The MC33664 supports 2 Mbps isolated Network Communication rate (supports a transformer based physical layer) that supports a maximum of 5 Meters and 15 node systems. It uses a dual SPI architecture for message confirmation and communication (shown in the figure below). The MC33661 and MC33772B support the following features:

- 7 to 13 (MC33771B) or 3 to 6 cell (MC33772B) cells voltage measurement channels, expandable by daisy-chaining multiple cell controllers
- Total Stack Voltage Measurement
- Single chip 48 V battery control scalable to > 1000 V
- ASIL-C functional safety compliant
- 300 mA cell balancing transistors and 0.5% current sensors
- Isolated 2 Mbps differential communication or 4.0 Mbps SPI
- >2.5x higher transformer coupled daisy chain isolation (3750 V)
- Synchronized cell I/V measurement with coulomb counter
- 2 mV voltage measurement accuracy
- 65µs one shot synchronized cell impedance determination
- Fast data acquisition: 3.6 ms for 96 cells, 4.5 ms for 112 cells
- Functional verification and diagnostics supporting ISO26262
- Automotive robustness: ESD, EMC, Hot plug, AEC Q-100
- 9.6 V ≤ V PWR ≤ 61.6 V operation, 70V transient
- 7 to 14x differential cell voltage + stack voltage measurement
- 7x ADC + GPIO + temperature sensor Inputs
- Low power modes
- 64 pin QFP package
- Low-level drivers to simplify SW development
- Integrated Sleep Mode Over/Under Voltage & Temperature Monitoring
- Over/Under Voltage, Over/Under Temperature Fault Verification
- On-board Passive Cell Balancing with Diagnostics and balancing timers
- Open Cell Terminal Detection
- Current sensor with ±0.5% accuracy from mA to kA
- Coulomb Counter (in low-power mode as well)

This is shown in the following figure.

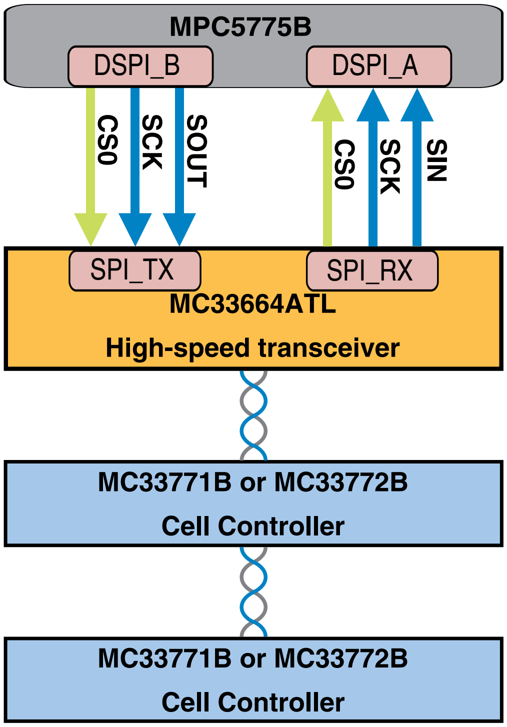


Figure C-1. Typical BMS system

C.2 Hardware set up for the BMS example

Additional hardware is required to run the BMS example software. NXP manufactures multiple boards that can be used for the cell controller, as well as a configurable "battery" pack. The cell controller boards available are:

Table C-1. Cell controller board options

Part number	Name	Description
FRDM33771BTPLEVB	Evaluation board for the MPC33771 with isolated daisy-chain communication	The FRDM33771BTPLEVB Evaluation Board (EVB) features the MC33771, a 14-channel battery cell controller for automotive and industrial Li-ion battery applications. It supports cell voltage measurement, passive cell balancing, GPIOs, external EEPROM, and fault detection pin report.
FRDM33772BTPLEVB	Evaluation board for the MPC33772 with isolated daisy-chain communication	The FRDM33772BTPLEVB Evaluation Board (EVB) features the MC33772, a 6-channel battery cell controller for automotive and industrial Li-ion battery applications. It supports cell voltage measurement, passive cell balancing, GPIOs, external EEPROM, and fault detection pin report.

NXP provides multiple options for a battery cell. The first option is a configuration battery pack "battery" pack is required. Since it is sometimes hard to emulate a real multi-cell battery with faults and varying voltages, NXP also sells a Configurable Battery pack.

Table C-2. Battery cell simulators

Part number	Name	Description
BATT-14AAAPACK	Configurable Battery Pack	The BATT-14AAAPACK can be configured for 3 to 14 battery cells using common AAA (also known as LR3 or 10440) batteries that are connected in series with an optional current sense resistor. Configuration jumpers and a charging port is provided.
BATT-6EMULATOR[1]	6-cell Battery Pack	The BATT-6EMULATOR can emulate a multi-cell battery pack that can be used with the FRDM33772BTPLEVB. It is a 6-cell emulator and up to 3 can be connected in series. The voltage of each cell can be adjusted with a potentiometer, as well as a potentiometer for controlling the shunt current. a cell voltage of 1.26 to ~4.8 V is supported. The shunt resistor supports a voltage of ±150 mV.
BATT-14EMULATOR[1]	14-cell Battery Pack	The BATT-14EMULATOR can emulate a multi-cell battery pack that can be used with the FRDM33771BTPLEVB. It is a 6-cell emulator and up to 3 can be connected in series. The voltage of each cell can be adjusted with a potentiometer, as well as a potentiometer for controlling the shunt current. A cell voltage of 1.26 to ~4.8 V is supported. The shunt resistor supports a voltage of ±150 mV.

[1] A separate power supply must be ordered for this kit. The power supply should be 12V, 0.5 A, 3.5mm jack, and positive center.

NXP provides a driver for the BMS interface. Currently version 1.1 is available for the "MC33771B&MC33772B Battery Cell Controllers - Software Driver".

In addition, some hardware connections are required. See MPC5775B evaluation board specific connectors for the pin-out of the BMS connector and wires.

C.3 BMS pin usage

The table below shows the pin usage for the BMS interface. It shows both the MPC5775B pin and the MC33664A pins.

Table C-3. BMS Pin Usage

MPC5775B					MC33664ATL1EG	
Function	Operating mode	Pin function	Ball location	Direction	Pin function	Pin location
DSPI_A SCK	Slave mode (Receive)	SCKA/GPIO93	AD8	Output	SCLK_RX	5
DSPI_A SIN		SINA/GPIO94	AF7	Input	DATA_RX	7
DSPI_A SOUT		(SOUTA/ GPIO95)	— (AD7)	—	—	—
DSPI_A CS0		PCSA0/GPIO96	AE6	Output	CS_RX	6
DSPI_B SCK	Master mode (Transmit)	SCKB/GPIO102	AE8	Output	SCLK_TX	1
DSPI_B SIN		—	— (AE9)	—	—	—
DSPI_B SOUT		SOUTB/ GPIO104	AF9	Output	DATA_TX	3
DSPI_B CS0		PCSB0/ GPIO105	AD9	Input	CS_TX_b	2

Table continues on the next page...

Table C-3. BMS Pin Usage (continued)

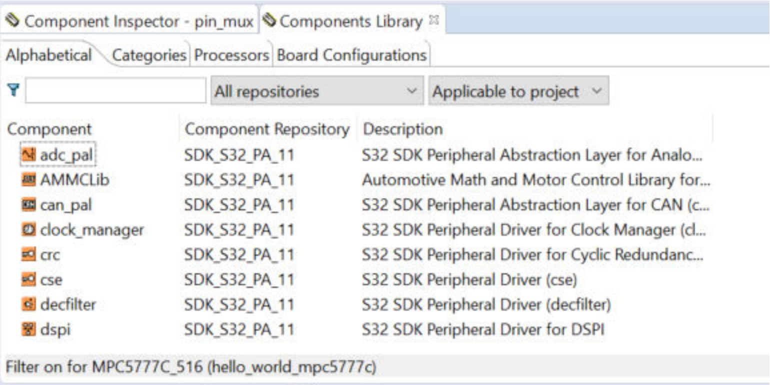
MPC5775B					MC33664ATL1EG	
Function	Operating mode	Pin function	Ball location	Direction	Pin function	Pin location
EN	BMS Enable	PCSA5/ GPIO101	AD6 (C25)	Output	EN	4
INTB	Interrupt	ETPUC14/IRQ5/ GPIO455	G26	Input	INT_b	8

Appendix D Change MPC5777C 516-pin project to MPC5775E or MPC5775B

Currently S32 Design Studio for Power Architecture (V2.1) only supports the MPC5777C in the 516 Ball Grid Array (BGA). It does not directly support the 416 BGA MPC5775E or MPC5775B.

After following the steps in Configuring and building software from an example , the following steps need to be performed.

1. Select the "Components Library" tab.

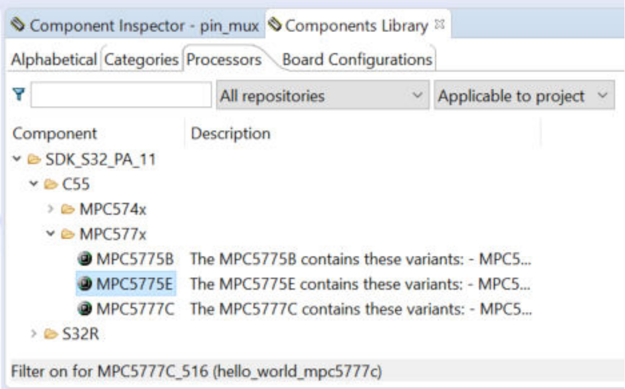


Component	Component Repository	Description
adc_pal	SDK_S32_PA_11	S32 SDK Peripheral Abstraction Layer for Analo...
AMMCLib	SDK_S32_PA_11	Automotive Math and Motor Control Library for...
can_pal	SDK_S32_PA_11	S32 SDK Peripheral Abstraction Layer for CAN (c...
clock_manager	SDK_S32_PA_11	S32 SDK Peripheral Driver for Clock Manager (cl...
crc	SDK_S32_PA_11	S32 SDK Peripheral Driver for Cyclic Redundanc...
cse	SDK_S32_PA_11	S32 SDK Peripheral Driver (cse)
decfilter	SDK_S32_PA_11	S32 SDK Peripheral Driver (decfilter)
dspi	SDK_S32_PA_11	S32 SDK Peripheral Driver for DSPI

Filter on for MPC5777C_516 (hello_world_mpc5777c)

Figure D-1. Components Library view

2. Select the "Processors" tab and select either the MPC5775E or the MPC5775B.

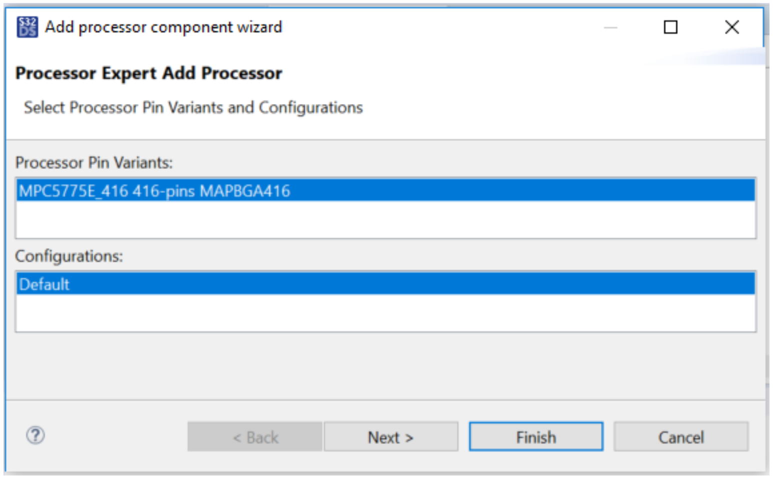


Component	Description
SDK_S32_PA_11	
C55	
MPC574x	
MPC577x	
MPC5775B	The MPC5775B contains these variants: - MPC5...
MPC5775E	The MPC5775E contains these variants: - MPC5...
MPC5777C	The MPC5777C contains these variants: - MPC5...
S32R	

Filter on for MPC5777C_516 (hello_world_mpc5777c)

Figure D-2. Processors Tab view

3. This opens a new window for adding the "Processor". select "Next"and then "Finish".



Add processor component wizard

Processor Expert Add Processor

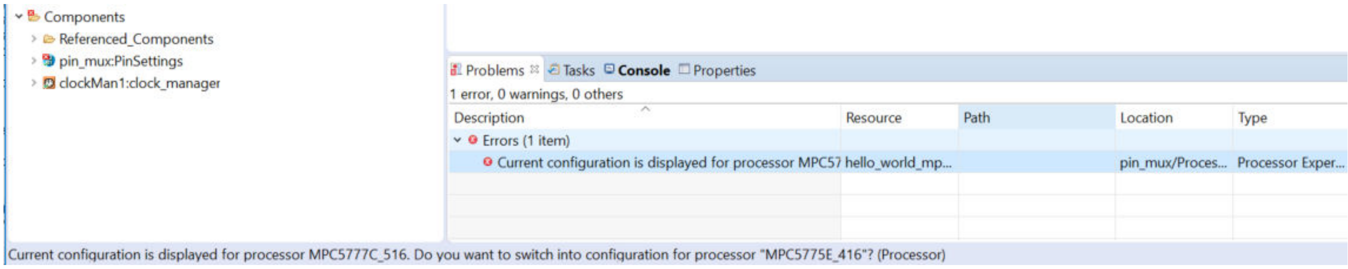
Select Processor Pin Variants and Configurations

Processor Pin Variants: MPC5775E_416 416-pins MAPBGA416

Configurations: Default

Figure D-3. Add processor

4. Selecting this processor generates errors.



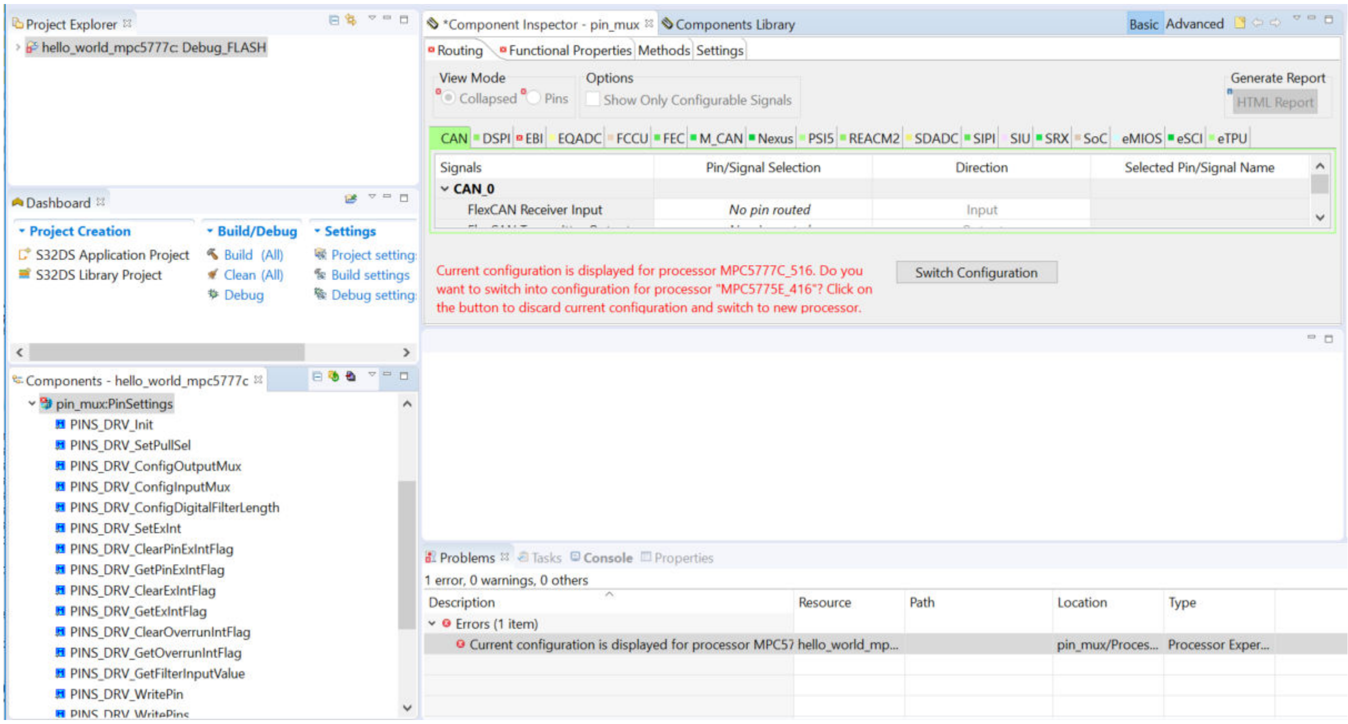
1 error, 0 warnings, 0 others

Description	Resource	Path	Location	Type
Errors (1 item)				
Current configuration is displayed for processor MPC57	hello_world_mp...		pin_mux/Proces...	Processor Exper...

Current configuration is displayed for processor MPC5777C_516. Do you want to switch into configuration for processor "MPC5775E_416"? (Processor)

Figure D-4. Processor error

5. To fix this error, the "Component Inspector" needs to be opened, which will show the source of the error.



Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
CAN_0			
FlexCAN Receiver Input	*No pin routed*	Input	

Current configuration is displayed for processor MPC5777C_516. Do you want to switch into configuration for processor "MPC5775E_416"? Click on the button to discard current configuration and switch to new processor.

1 error, 0 warnings, 0 others

Description	Resource	Path	Location	Type
Errors (1 item)				
Current configuration is displayed for processor MPC57	hello_world_mp...		pin_mux/Proces...	Processor Exper...

Figure D-5. Component Inspector

6. Selecting "Switch Configuration" will resolve the disconnects in the pin mapping between the MPC5777C 516-pin processor and the 416-pin MPC5775E or MPC5775B.
7. Select "Build all" in the "Project" menu to compile the program.

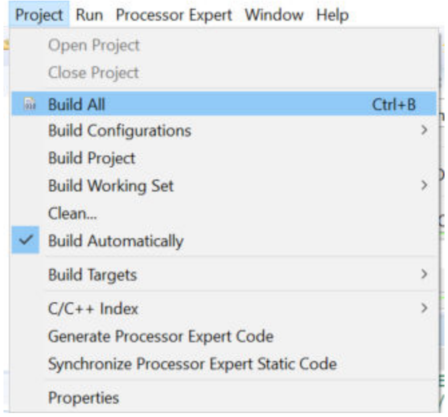


Figure D-6. Build project menu

The example project is now compiled and can be executed.

Appendix E S32 Design Studio debug connection messages

Below is a listing of the start-up messages when loading the "hello_world" example into the MPC5775E or MPC5775B EVB. This shows the flash programming and then the part initialization that is performed prior to executing to "main".

```
Connection from "127.0.0.1" via 127.0.0.1. Connection from port "62976" to 7224
Copyright 2018 P&E Microcomputer Systems,Inc.
Command Line :C:\NXP\S32DS_Power_v2.1\eclipse\plug-ins
\com.pemicro.debug.gdbjtag.ppc_1.8.7.201906071634\win32\pegdbserver_power_console -
device=MPC5777C -startserver -singlesession -serverport=7224 -gdbmiport=6224 -
interface=OPENSDA -speed=5000 -port=USB1´

CMD>RE

Initializing.

Device IDCODE is $00000377
Device ID revision is $00000001
MPC5777C Device detected.
Target has been RESET and is active.
CMD>CM C:\NXP\S32DS_Power_v2.1\eclipse\plugins
\com.pemicro.debug.gdbjtag.ppc_1.8.7.201906071634\win32\gdi\P&E
\nxp_mpc5777c_1x32x2048k_cflash_highspeed.pcp

Initializing.

Device IDCODE is $00000377
Device ID revision is $00000001
MPC5777C Device detected.
Initialized.

;version 1.04, 12/13/2016, Copyright P&E Microcomputer Systems, www.pemicro.com
[5777C_8meg_highspeed]

;device NXP, MPC5777C, 1x32x2048k, desc=CFlash_highspeed

;begin_cs device=$30800000, length=$00800000, ram=$00300000

Loading programming algorithm ...
Done.
```

```
Programming sequency is : erase, blank check, program, and verify {default}
CMD>VC
Verifying object file CRC-16 to device ranges ...
   block 00800000-00800007 ...
Ok.
   block 00800400-00800617 ...
Ok.
   block 00810000-00810133 ...
Ok.
   block 00811000-00811803 ...
Ok.
   block 00811810-0081331B ...
Ok.
   Checksum Verification Successful. (Cumulative CRC-16=$A97E)
Application verified in memory. No need to reprogram.

CMD>RE

Initializing.

Device IDCODE is $00000377
Device ID revision is $00000001
MPC5777C Device detected.
Target has been RESET and is active.

Device IDCODE is $00000377
Device ID revision is $00000001

Starting reset script (C:\NXP\S32DS_Power_v2.1\eclipse\plugins
\com.pemicro.debug.gdbjtag.ppc_1.8.7.201906071634\win32\gdi\P&E\s32e200_mpc5777c.mac) ...
REM This script is compatible with MPC5777C
devices.

REM Clean GPRs to remove residual data after using
algorithm

REM Setup MMU for for Periph B
Modules

REM Base address =
$FFE0_0000

REM TLB0, 2 MByte Memory Space, Guarded, Don't Cache, All
Access

Once Status Register Result = $0209
REM Set up MMU to put internal Flash at
0...

REM Virtual address 0x0 -> Physical address =
$0000_0000

REM TLB1, 16 MByte Memory Space, Not Guarded, Cachable, All
Access

Once Status Register Result = $0209
REM Set up MMU for External
Memory

REM Base address =
$2000_0000

REM TLB2, 16 MByte Memory Space, Not Guarded, Cachable, All
Access

Once Status Register Result = $0209
REM Set up MMU for Internal
SRAM

REM Base address =
$4000_0000

REM TLB3, 512 KByte Memory Space, Not Guarded, Don't Cache, All
Access

Once Status Register Result = $0209
REM Set up MMU for Periph A
Modules

REM Base address =
$C3E0_0000

REM TLB4, 2 MByte Memory Space, Guarded, Don't Cache, All
Access

Once Status Register Result = $0209
REM Initialize all of the Main SRAM -
512KB

Initializing RAM from $40000000 to $4007FFFF.

Reset script (C:\NXP\S32DS_Power_v2.1\eclipse\plugins
\com.pemicro.debug.gdbjtag.ppc_1.8.7.201906071634\win32\gdi\P&E\s32e200_mpc5777c.mac)
completed.

MPC5777C Device detected.
```