SDK Migration

Lu Lin

APR.2015

External Use

This presentation will discuss two topics

✓ Migrate SDK demos from 1.0 to 1.2

✓ Porting SDK to an unsupported chip

Migrate SDK demos from 1.0 to 1.2

Agenda

- Overview
- Description of Process
- Q&A

Overview

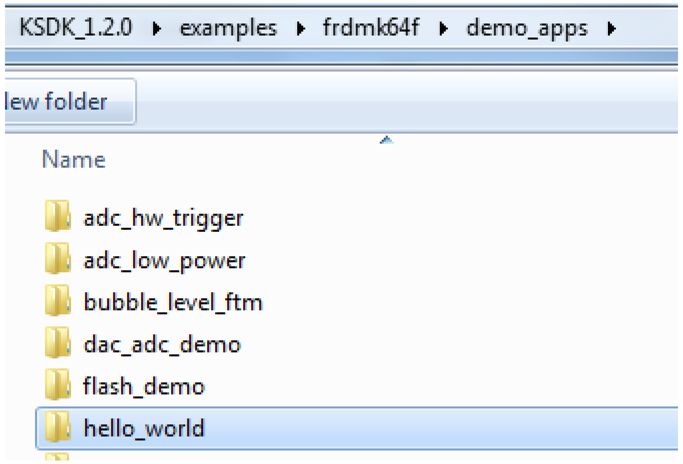
- Compared with SDK1.0, the architecture of SDK1.2 has been changed greatly. So if want to migrate demos from 1.0 to 1.2, The difference of demo directory between 1.0 and 1.2 should be understood. It can refer to SDK1.0 and SDK1.2 release note.

- Take hello_world demo of frdmk64f on IAR as example.

Description of process

- The SDK1.0 IAR project of hello_world demo is located at \demos\hello_world\iar\frdmk64f120m\.
- The demos of SDK1.2 are located at \examples\frdmk64f\demo_apps\.

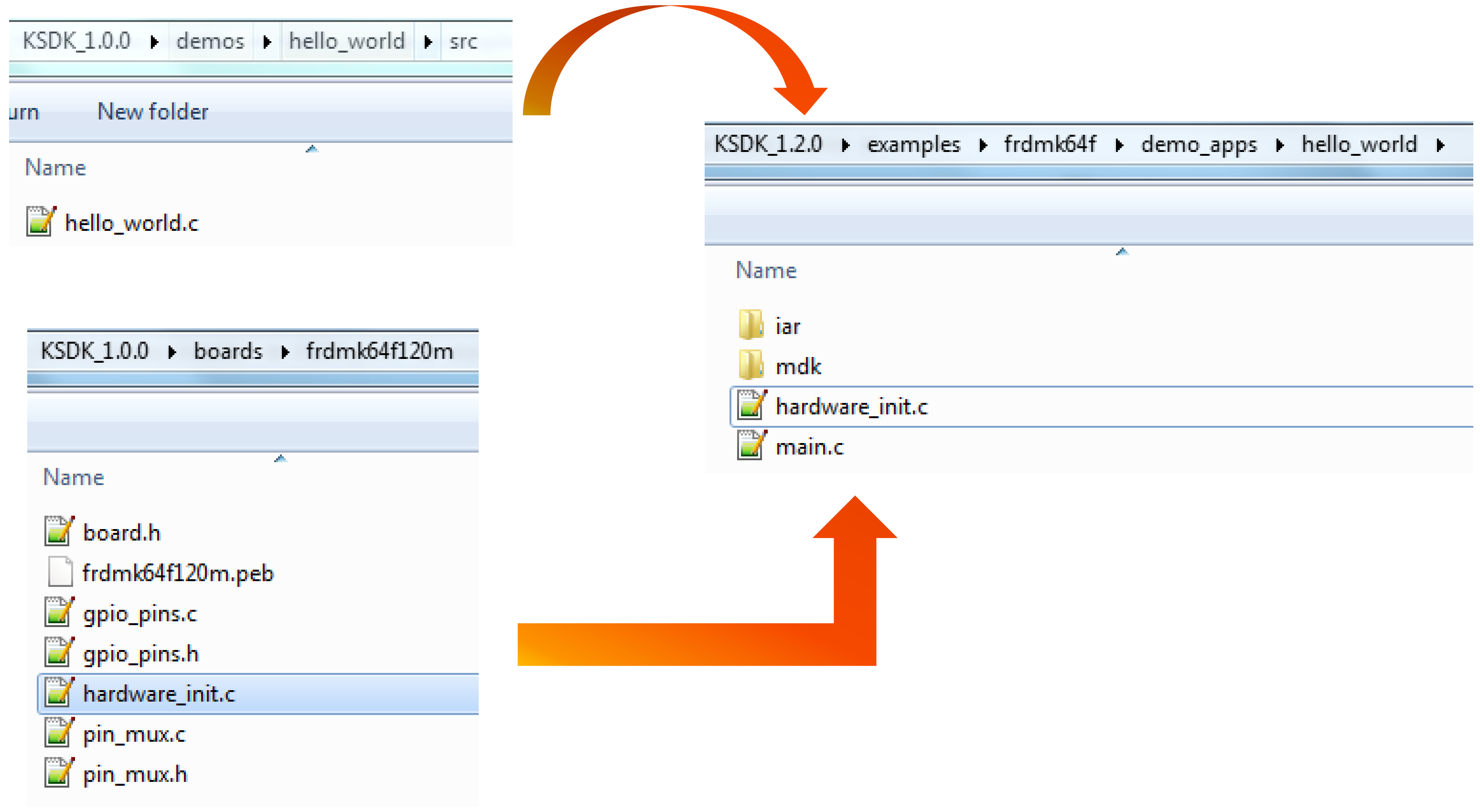
- First, we should create a new directory named “hello_world” under \examples\frdmk64f\demo_apps\



Description of process

- Create a new directory named “iar” under \examples\frdmk64f\demo_apps\hello_world\, if you want to use keil, you can also create a new directory named “mdk” under the same directory, and other tool chains can be operated with the same action.
- Copy the source file “hello_world.c” from 1.0 to 1.2, and changed the name to “main.c”, and modify it according to your needing and API reference manual of SDK1.2.
- Copy “hardware_init.c” from 1.0 to 1.2, and modify it, because in SDK1.2, hardware initialization source file is located at demo directory, it only initializes the necessary modules that this demo required.

Description of process


KSDK_1.0.0 ▸ demos ▸ hello_world ▸ src

urn New folder

Name

hello_world.c

KSDK_1.0.0 ▸ boards ▸ frdmk64f120m

Name

board.h
frdmk64f120m.peb
gpio_pins.c
gpio_pins.h
hardware_init.c
pin_mux.c
pin_mux.h

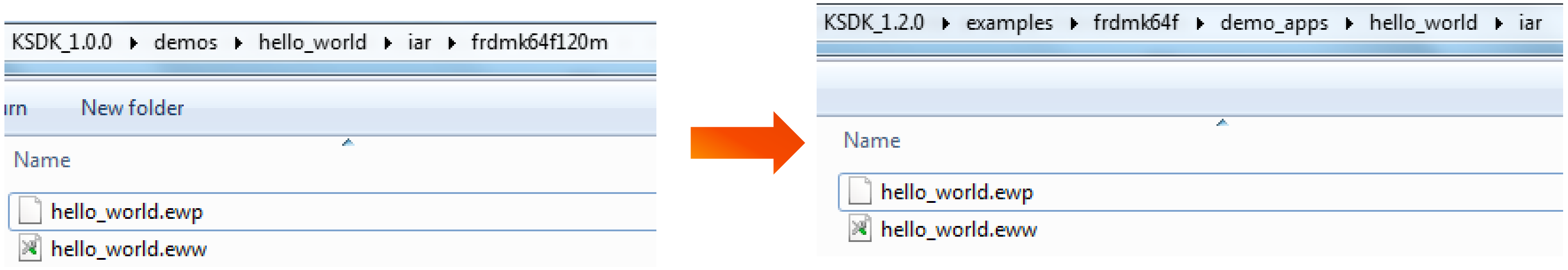
KSDK_1.2.0 ▸ examples ▸ frdmk64f ▸ demo_apps ▸ hello_world ▸

Name

iar
mdk
hardware_init.c
main.c

Description of process

- Copy IAR projects from 1.0 to 1.2


KSDK_1.0.0 ▸ demos ▸ hello_world ▸ iar ▸ frdmk64f120m

hello_world.ewp
hello_world.eww

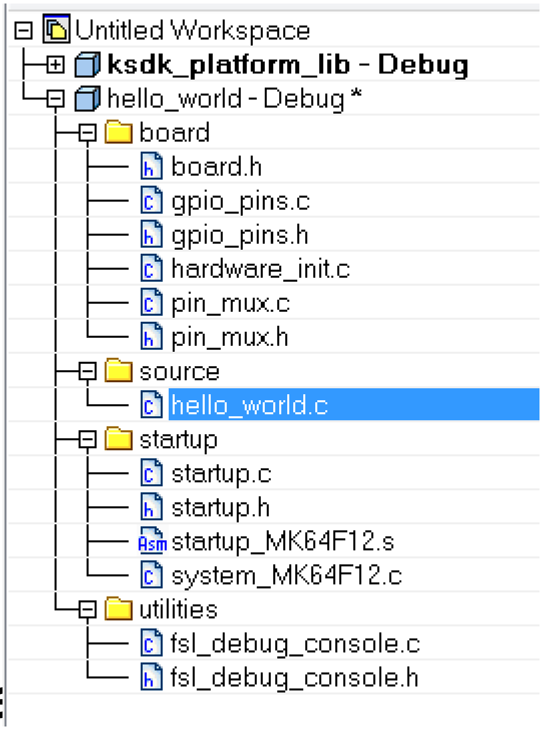
KSDK_1.2.0 ▸ examples ▸ frdmk64f ▸ demo_apps ▸ hello_world ▸ iar

hello_world.ewp
hello_world.eww

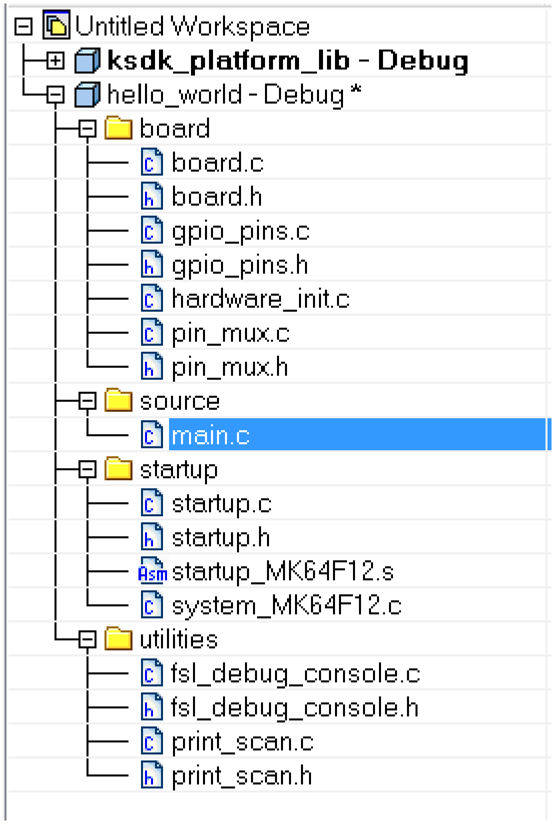
- Then open the IAR project files to change for SDK1.2
  - Relocate the platform library project.
  - Relocate the board configuration files. For SDK1.0, board configuration files are located at \boards\frdmk64f120m, and for SDK1.2, they are located at \examples\frdmk64f and hardware_init.c is located at \examples\frdmk64f\demo_apps\hello_world
  - Relocate the source files of this demo. For SDK1.0, hello_world.c is located at \demos\hello_world\src, and for SDK1.2, main.c is located at \examples\frdmk64f\demo_apps\hello_world

Description of process

- Change the IAR project files for SDK1.2
  - Relocate the startup files. For SDK1.0, these files are located at \platform\startup and \platform\startup\MK64F12, and for SDK1.2, they are located at \platform\devices and \platform\devices\MK64F12\startup
  - Relocate the utilities files. For SDK1.0, these files are located at \platform\utilities, and for SDK1.2, these files are located at \platform\utilities.



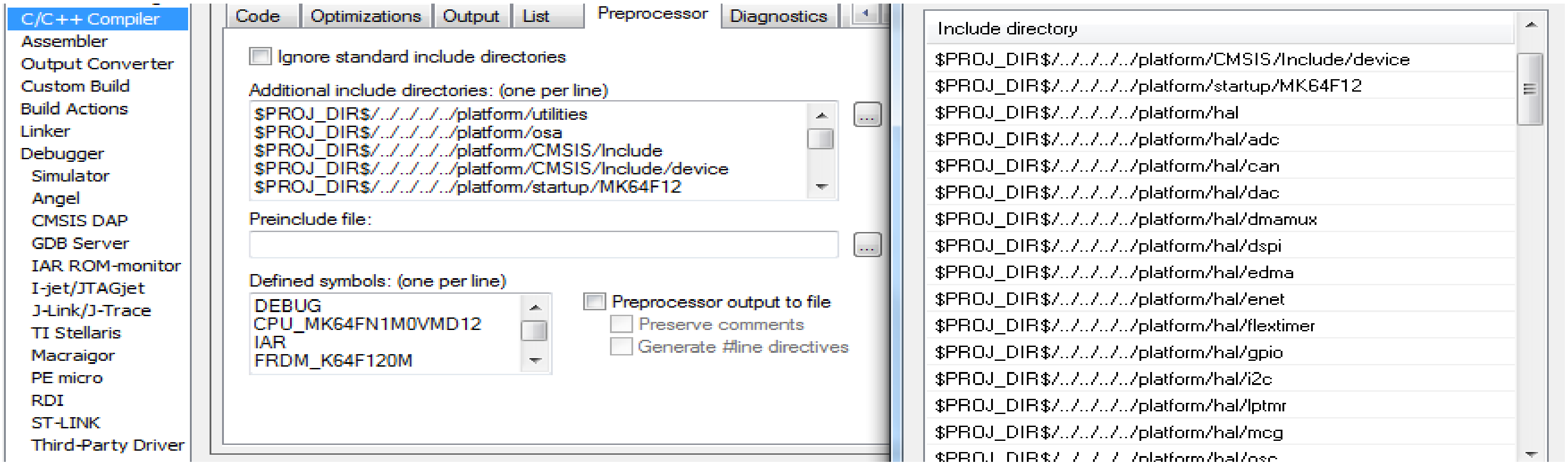
```
Untitled Workspace
├ ksdk_platform_lib - Debug
└ hello_world - Debug *
   ├ board
   │  ├ board.h
   │  ├ gpio_pins.c
   │  ├ gpio_pins.h
   │  ├ hardware_init.c
   │  ├ pin_mux.c
   │  └ pin_mux.h
   ├ source
   │  └ hello_world.c
   ├ startup
   │  ├ startup.c
   │  ├ startup.h
   │  ├ startup_MK64F12.s
   │  └ system_MK64F12.c
   └ utilities
      ├ fsl_debug_console.c
      └ fsl_debug_console.h
```



```
Untitled Workspace
├ ksdk_platform_lib - Debug
└ hello_world - Debug *
   ├ board
   │  ├ board.c
   │  ├ board.h
   │  ├ gpio_pins.c
   │  ├ gpio_pins.h
   │  ├ hardware_init.c
   │  ├ pin_mux.c
   │  └ pin_mux.h
   ├ source
   │  └ main.c
   ├ startup
   │  ├ startup.c
   │  ├ startup.h
   │  ├ startup_MK64F12.s
   │  └ system_MK64F12.c
   └ utilities
      ├ fsl_debug_console.c
      ├ fsl_debug_console.h
      ├ print_scan.c
      └ print_scan.h
```

Description of process

- Then change the option of the project.



Ignore standard include directories

Additional include directories: (one per line)

```
$PROJ_DIR$/../../../../platform/utilities
$PROJ_DIR$/../../../../platform/osa
$PROJ_DIR$/../../../../platform/CMSIS/Include
$PROJ_DIR$/../../../../platform/CMSIS/Include/device
$PROJ_DIR$/../../../../platform/startup/MK64F12
```

Preinclude file:

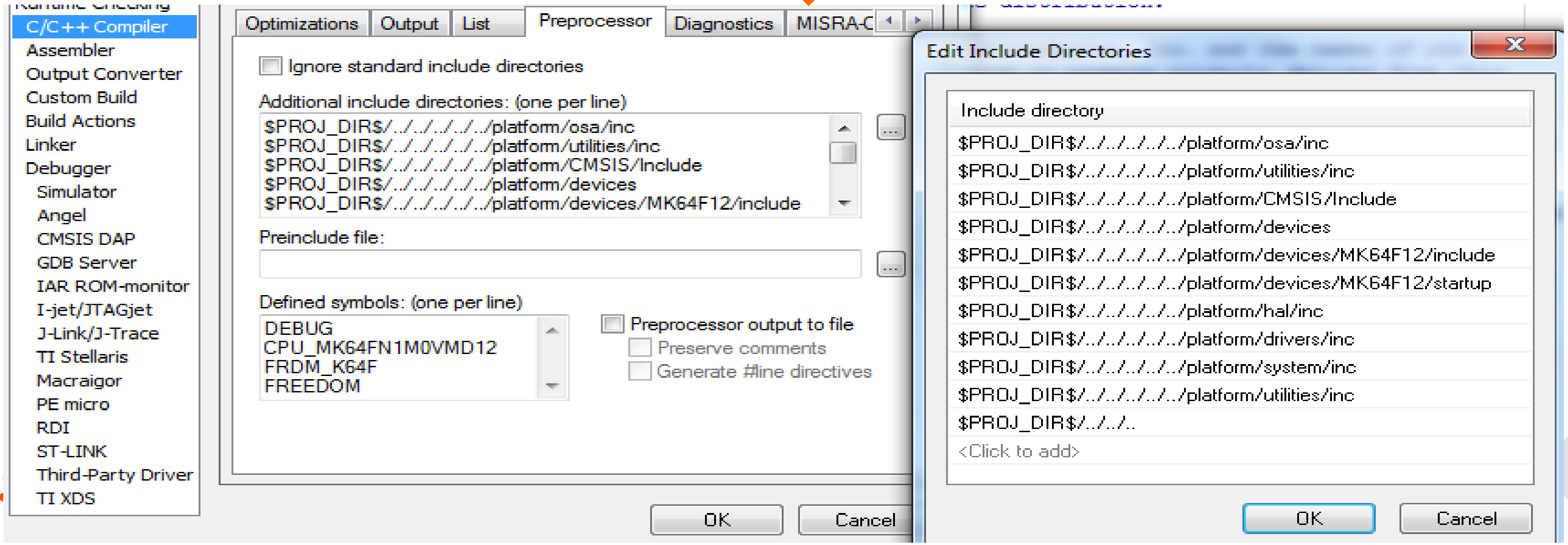
Defined symbols: (one per line)

```
DEBUG
CPU_MK64FN1M0VMD12
IAR
FRDM_K64F120M
```

Preprocessor output to file
Preserve comments
Generate #line directives

Include directory

```
$PROJ_DIR$/../../../../platform/CMSIS/Include/device
$PROJ_DIR$/../../../../platform/startup/MK64F12
$PROJ_DIR$/../../../../platform/hal
$PROJ_DIR$/../../../../platform/hal/adc
$PROJ_DIR$/../../../../platform/hal/can
$PROJ_DIR$/../../../../platform/hal/dac
$PROJ_DIR$/../../../../platform/hal/dmamux
$PROJ_DIR$/../../../../platform/hal/dspi
$PROJ_DIR$/../../../../platform/hal/edma
$PROJ_DIR$/../../../../platform/hal/enet
$PROJ_DIR$/../../../../platform/hal/flextimer
$PROJ_DIR$/../../../../platform/hal/gpio
$PROJ_DIR$/../../../../platform/hal/i2c
$PROJ_DIR$/../../../../platform/hal/lptmr
$PROJ_DIR$/../../../../platform/hal/mcg
$PROJ_DIR$/../../../../platform/hal/osc
```



Ignore standard include directories

Additional include directories: (one per line)

```
$PROJ_DIR$/../../../../../platform/osa/inc
$PROJ_DIR$/../../../../../platform/utilities/inc
$PROJ_DIR$/../../../../../platform/CMSIS/Include
$PROJ_DIR$/../../../../../platform/devices
$PROJ_DIR$/../../../../../platform/devices/MK64F12/include
```

Preinclude file:

Defined symbols: (one per line)

```
DEBUG
CPU_MK64FN1M0VMD12
FRDM_K64F
FREEDOM
```

Preprocessor output to file
Preserve comments
Generate #line directives

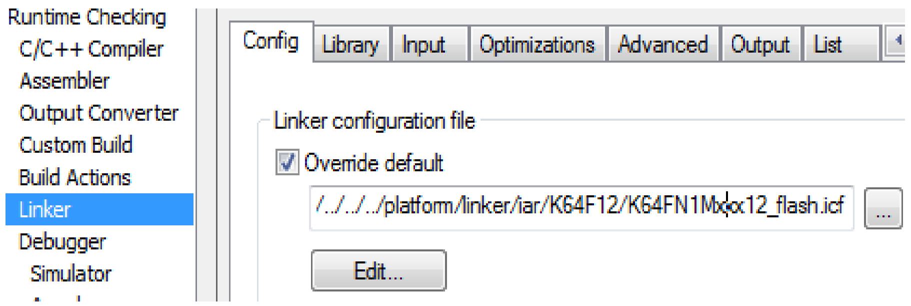
Edit Include Directories

Include directory

```
$PROJ_DIR$/../../../../../platform/osa/inc
$PROJ_DIR$/../../../../../platform/utilities/inc
$PROJ_DIR$/../../../../../platform/CMSIS/Include
$PROJ_DIR$/../../../../../platform/devices
$PROJ_DIR$/../../../../../platform/devices/MK64F12/include
$PROJ_DIR$/../../../../../platform/devices/MK64F12/startup
$PROJ_DIR$/../../../../../platform/hal/inc
$PROJ_DIR$/../../../../../platform/drivers/inc
$PROJ_DIR$/../../../../../platform/system/inc
$PROJ_DIR$/../../../../../platform/utilities/inc
$PROJ_DIR$/../../..
<Click to add>
```

Description of process

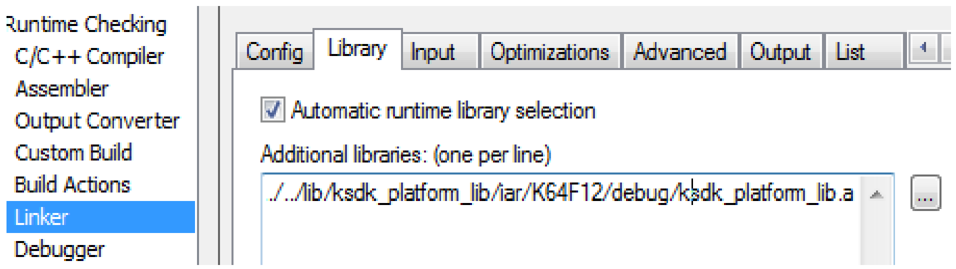
- Change the option of the project.



Linker configuration file

☑ Override default

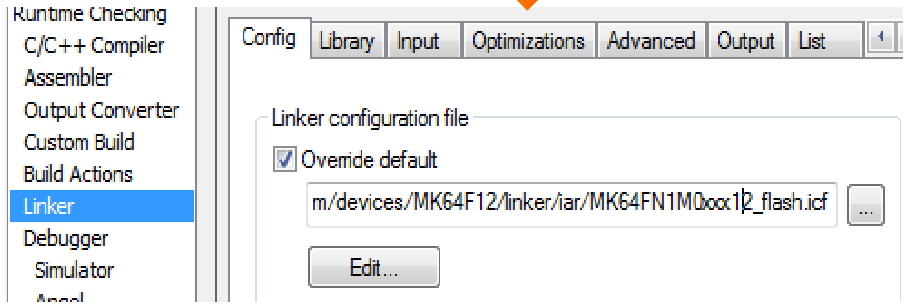
/../../../platform/linker/iar/K64F12/K64FN1Mxxx12_flash.icf



☑ Automatic runtime library selection

Additional libraries: (one per line)

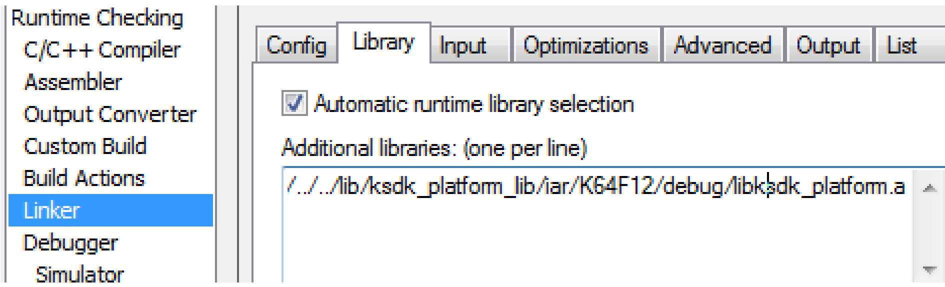
./../lib/ksdk_platform_lib/iar/K64F12/debug/ksdk_platform_lib.a



Linker configuration file

☑ Override default

m/devices/MK64F12/linker/iar/MK64FN1M0xxx12_flash.icf



☑ Automatic runtime library selection

Additional libraries: (one per line)

/../../lib/ksdk_platform_lib/iar/K64F12/debug/libksdk_platform.a

Q & A

Porting SDK to an unsupported chip

Agenda

- Overview
- Description of Process
- Q&A

Overview

- SDK package may not cover all Kinetis devices when released, but some customers may want to use our SDK on their specified Kinetis silicon that is not supported by current SDK.

- This presentation will discuss about how to do to make a successful port from a supported device to a non-supported device.

- Take K60D10 and TWR-K60D100M board as example based on SDK1.2 release package.
- The chip of TWR-K60D100M is MK60DN512VMD10 and it belongs to MK60DN512xxx10 series.
  Assuming that SDK1.2 doesn’t support K60D10 and can be found that the most similar device with K60 in SDK1.2 is K64F120M .

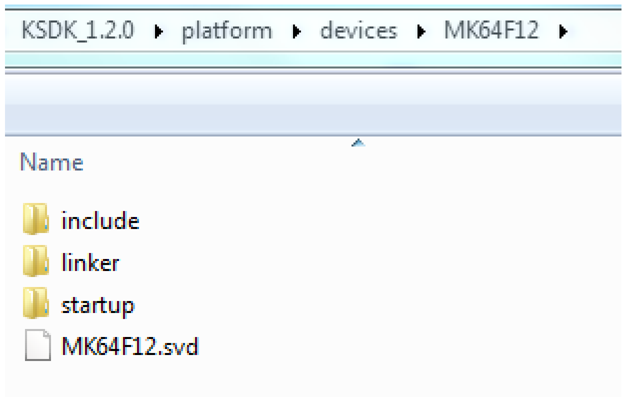
Description of Process

there are four main steps to be porting.

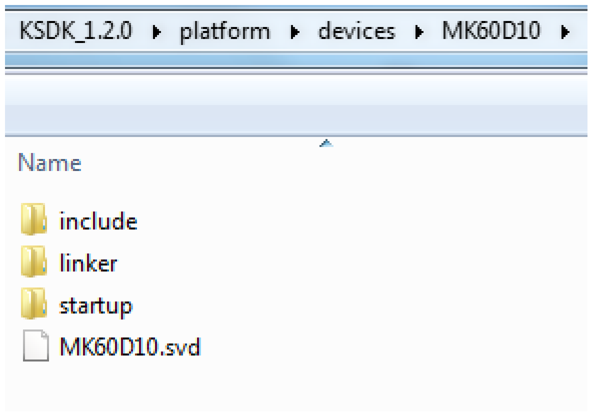
- ✓ **Create the device specific files**
- ✓ **Build the Platform Library**
- ✓ **Create the Board configuration files**
- ✓ **Modify the Projects**

Create the device specific files

- The device specific files include register definition files, feature definition files, startup files, linker files and other such as specified clock, sim files.
- Create a new directory named “MK60D10” under path sdk_install_folder\platform\devices\, then copy all files under path sdk_install_folder\platform\devices\ \MK64F12 to path sdk_install_folder\platform\devices\ MK60D10.
- Modify the name of file from “MK64F12” to “MK60D10”.


KSDK_1.2.0 ▸ platform ▸ devices ▸ MK64F12 ▸

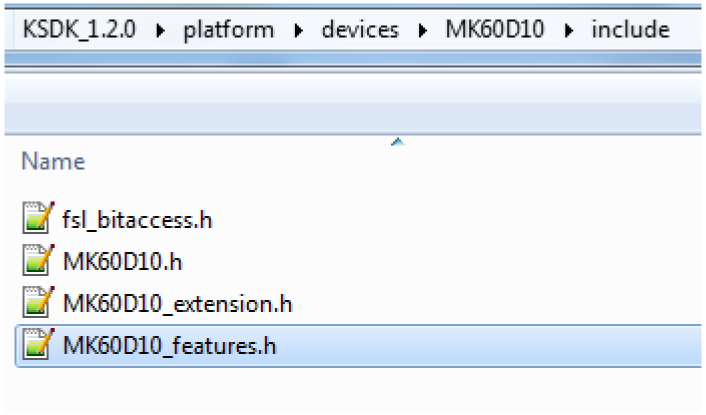
Name
- include
- linker
- startup
- MK64F12.svd


KSDK_1.2.0 ▸ platform ▸ devices ▸ MK60D10 ▸

Name
- include
- linker
- startup
- MK60D10.svd

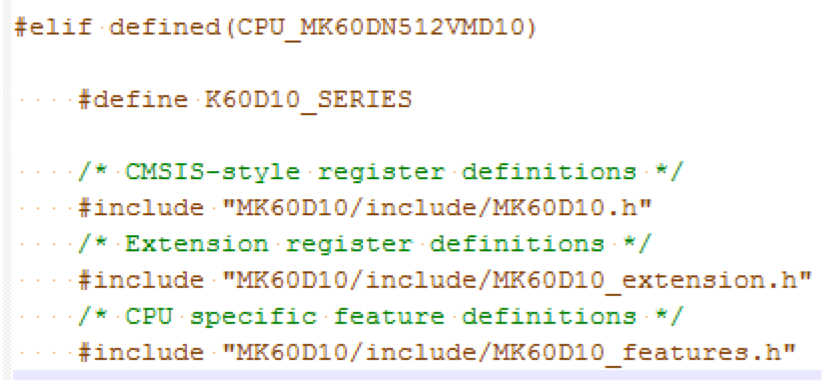
Create the device specific files

- The directory “include” includes device specific register definition, feature definition and so on
- These files should be modified according to the datasheet and reference manual of K60D10.



KSDK_1.2.0 ▸ platform ▸ devices ▸ MK60D10 ▸ include

Name
fsl_bitaccess.h
MK60D10.h
MK60D10_extension.h
MK60D10_features.h

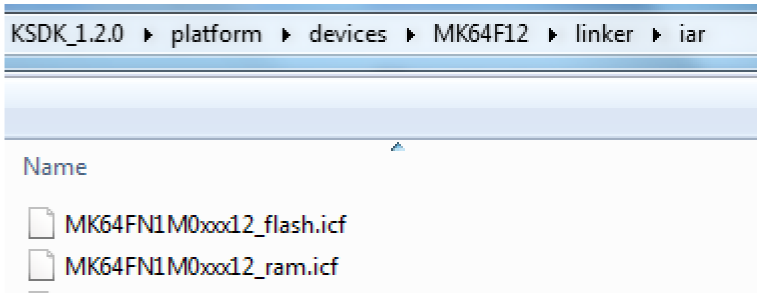


```
#elif defined(CPU_MK60DN512VMD10)

    #define K60D10_SERIES

    /* CMSIS-style register definitions */
    #include "MK60D10/include/MK60D10.h"
    /* Extension register definitions */
    #include "MK60D10/include/MK60D10_extension.h"
    /* CPU specific feature definitions */
    #include "MK60D10/include/MK60D10_features.h"
```

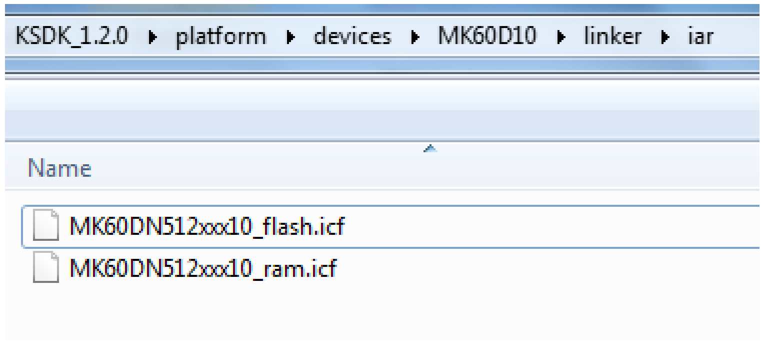
- Then Add these header files into the common file “fsl_device_registers.h” which located at \platform\devices.

Create the device specific files

- The directory “linker” includes device specific link file for each tool chain, such as IAR, Keil and so on.
- These files should be modified according to the memory map of K60D10, take IAR as example:


KSDK_1.2.0 ▸ platform ▸ devices ▸ MK64F12 ▸ linker ▸ iar

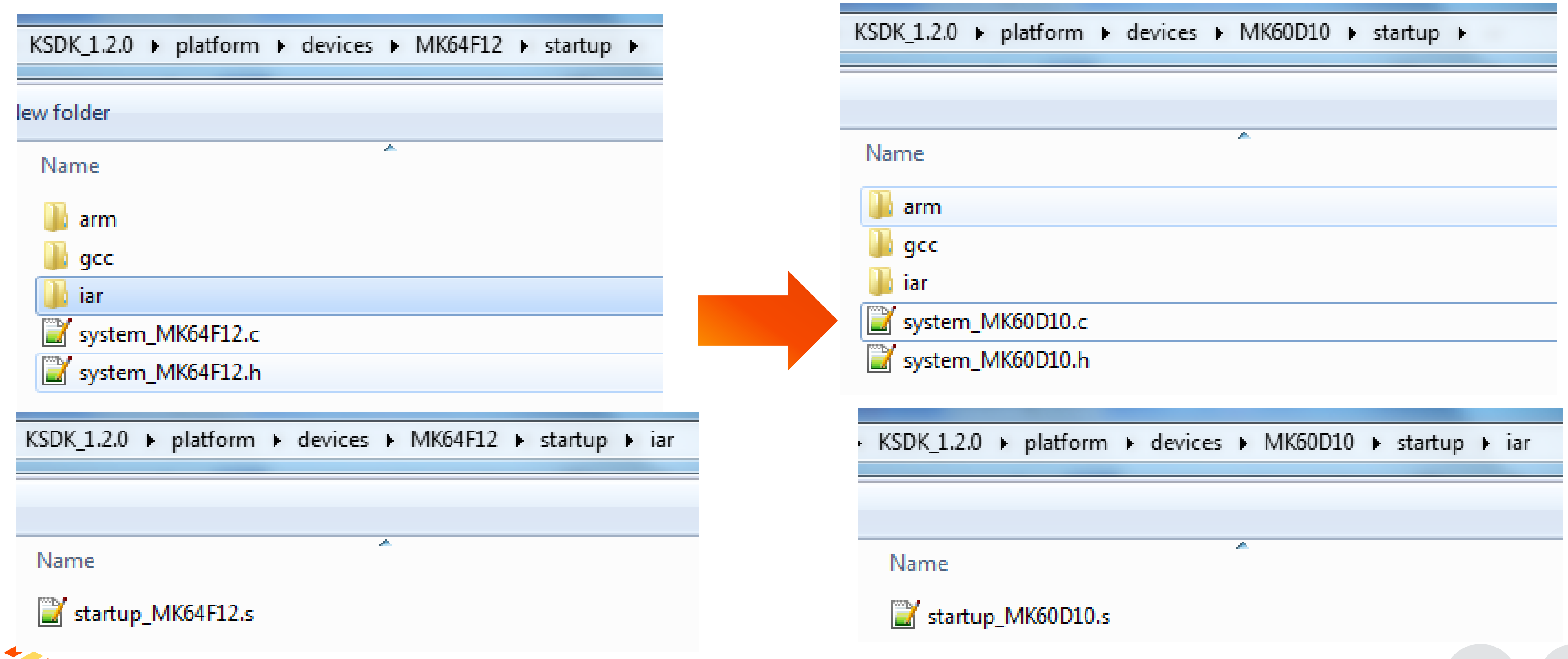
Name
MK64FN1M0xxx12_flash.icf
MK64FN1M0xxx12_ram.icf


KSDK_1.2.0 ▸ platform ▸ devices ▸ MK60D10 ▸ linker ▸ iar

Name
MK60DN512xxx10_flash.icf
MK60DN512xxx10_ram.icf

Create the device specific files

- The directory "startup" includes device specific startup files for each tool chain, such as IAR, Keil and so on.
- These files should be modified according to the reference manual of K60D10 and main modification is clock setup and interrupt vector table.


KSDK_1.2.0 ▸ platform ▸ devices ▸ MK64F12 ▸ startup ▸

lew folder

Name
- arm
- gcc
- iar
- system_MK64F12.c
- system_MK64F12.h

KSDK_1.2.0 ▸ platform ▸ devices ▸ MK64F12 ▸ startup ▸ iar

Name
- startup_MK64F12.s

KSDK_1.2.0 ▸ platform ▸ devices ▸ MK60D10 ▸ startup ▸

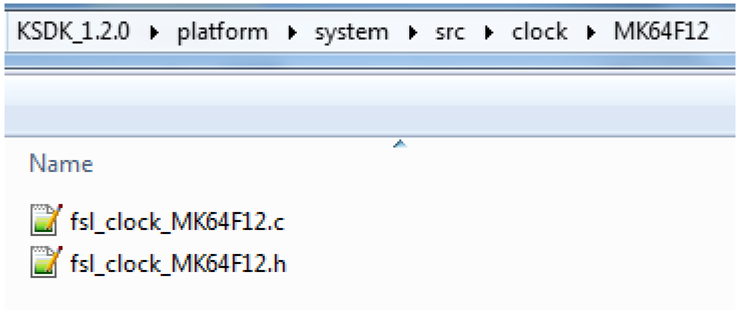
Name
- arm
- gcc
- iar
- system_MK60D10.c
- system_MK60D10.h

KSDK_1.2.0 ▸ platform ▸ devices ▸ MK60D10 ▸ startup ▸ iar

Name
- startup_MK60D10.s

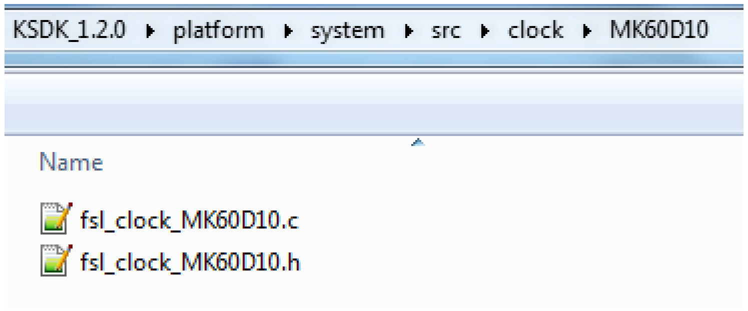
Create the device specific files

- Other driver/hal/system device specific files should be created, for example, under path sdk_install_folder\platform\system\src\clock\, each device has its own clock definition files, Copy \MK64F12 to \MK60D10 and modify these files for K60D10.


KSDK_1.2.0 ▸ platform ▸ system ▸ src ▸ clock ▸ MK64F12

Name

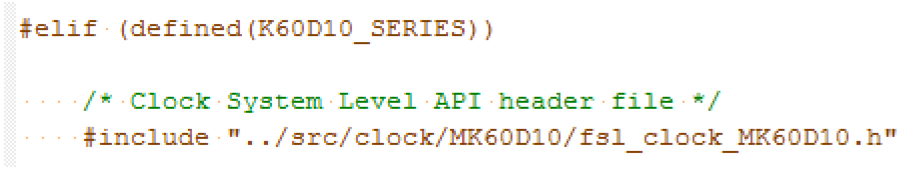
fsl_clock_MK64F12.c
fsl_clock_MK64F12.h


KSDK_1.2.0 ▸ platform ▸ system ▸ src ▸ clock ▸ MK60D10

Name

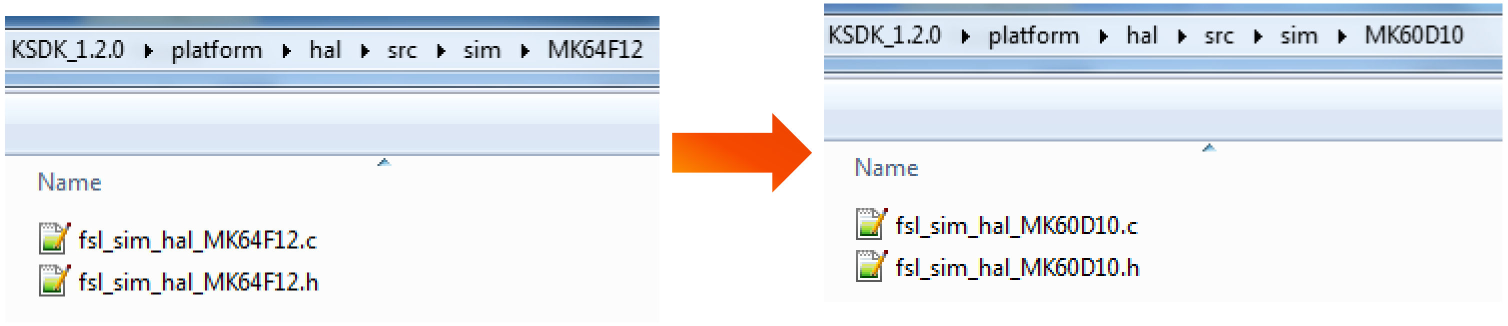
fsl_clock_MK60D10.c
fsl_clock_MK60D10.h

- Then add clock content of K60D10 into common file “fsl_clock_manager.h” which located at \platform\system\inc.


```
#elif (defined(K60D10_SERIES))

    /* Clock System Level API header file */
    #include "../src/clock/MK60D10/fsl_clock_MK60D10.h"
```

Create the device specific files

- under path sdk_install_folder\platform\hal\src\sim\, each device has its own SIM definition files, Copy \MK64F12 to \MK60D10 and modify these files for K60D10.



KSDK_1.2.0 ▸ platform ▸ hal ▸ src ▸ sim ▸ MK64F12

Name

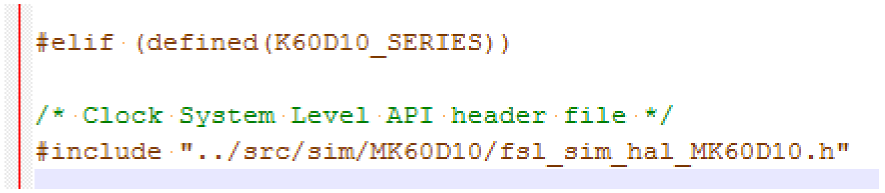
fsl_sim_hal_MK64F12.c
fsl_sim_hal_MK64F12.h

KSDK_1.2.0 ▸ platform ▸ hal ▸ src ▸ sim ▸ MK60D10

Name

fsl_sim_hal_MK60D10.c
fsl_sim_hal_MK60D10.h

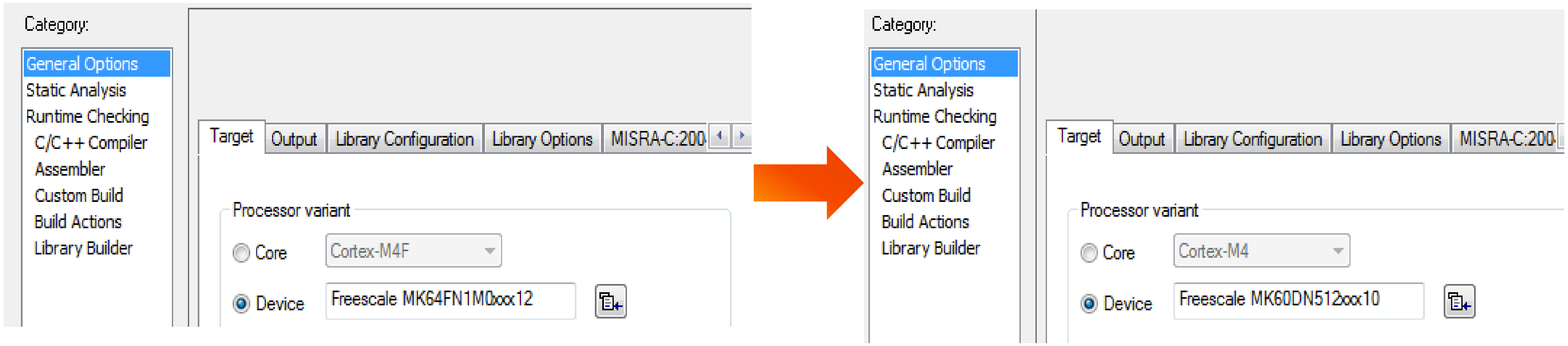
- Then add SIM content of K60D10 into common file "fsl_sim_hal.h" which is located at sdk_install_folder\platform\hal\inc



```
#elif (defined(K60D10_SERIES))

/* Clock System Level API header file */
#include "../src/sim/MK60D10/fsl_sim_hal_MK60D10.h"
```

Build the Platform Library

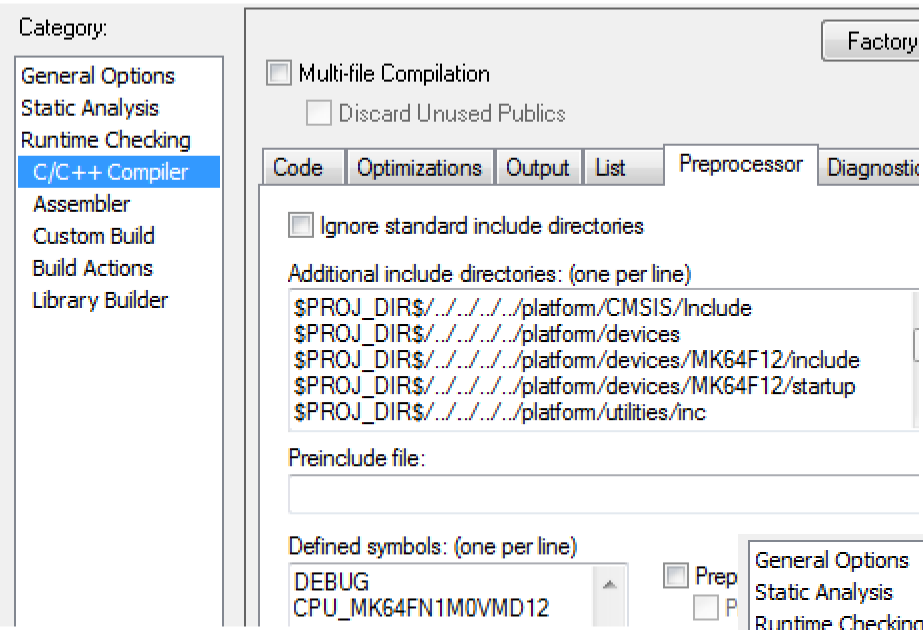
- Under directory sdk_install_folder\lib, there are platform library's project files for each tool chain.
- Take IAR project files of ksdk_platform_lib as example.Copy \lib\ksdk_platform_lib\iar\K64F12 to \lib\ksdk_platform_lib\iar\K60D10. Add modules that K60D10 had and K64F12 didn't have into the library and remove modules that K60D10 didn't have and K64F12 had.
- Then modify the project options.



Processor variant
Core: Cortex-M4F
[✓ Device]: Freescale MK64FN1M0xxx12

Processor variant
Core: Cortex-M4
[✓ Device]: Freescale MK60DN512xxx10

Build the Platform Library



Multi-file Compilation

Discard Unused Publics

Ignore standard include directories

Additional include directories: (one per line)

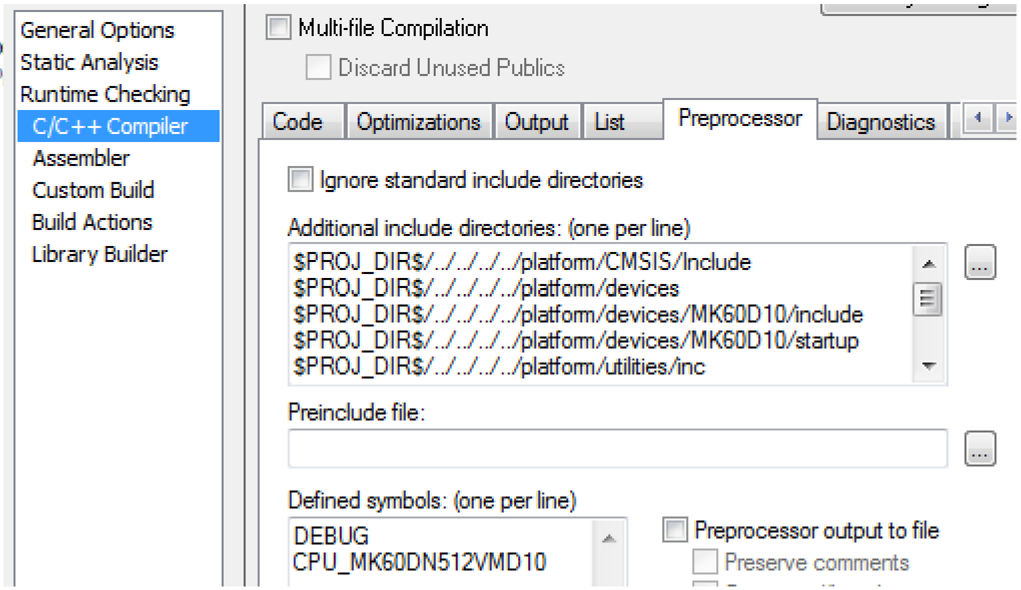
```
$PROJ_DIR$/../../../../platform/CMSIS/Include
$PROJ_DIR$/../../../../platform/devices
$PROJ_DIR$/../../../../platform/devices/MK64F12/include
$PROJ_DIR$/../../../../platform/devices/MK64F12/startup
$PROJ_DIR$/../../../../platform/utilities/inc
```

Preinclude file:

Defined symbols: (one per line)

```
DEBUG
CPU_MK64FN1M0VMD12
```

Prep



Multi-file Compilation

Discard Unused Publics

Ignore standard include directories

Additional include directories: (one per line)

```
$PROJ_DIR$/../../../../platform/CMSIS/Include
$PROJ_DIR$/../../../../platform/devices
$PROJ_DIR$/../../../../platform/devices/MK60D10/include
$PROJ_DIR$/../../../../platform/devices/MK60D10/startup
$PROJ_DIR$/../../../../platform/utilities/inc
```

Preinclude file:

Defined symbols: (one per line)

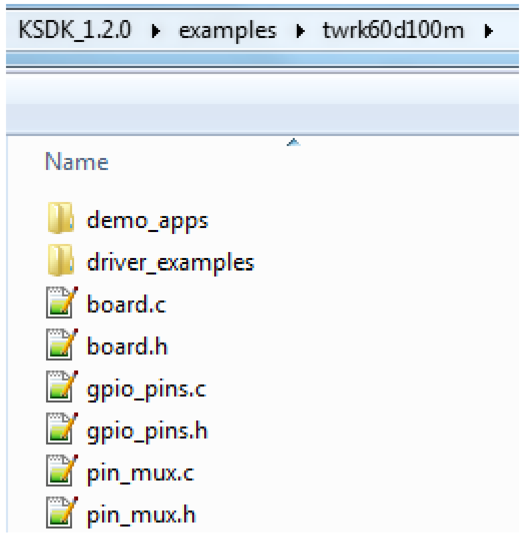
```
DEBUG
CPU_MK60DN512VMD10
```

Preprocessor output to file

Preserve comments

Create the Board configuration files

- The board configuration files are located at sdk_install_folder\examples. Copy \twrk64f120m to \twrk60d100m, and board configuration files are board.c, board.h, gpio_pins.c, gpio_pins.h, pin_mux.c and pin_mux.h
- Modify these files according to schematic and reference manual.



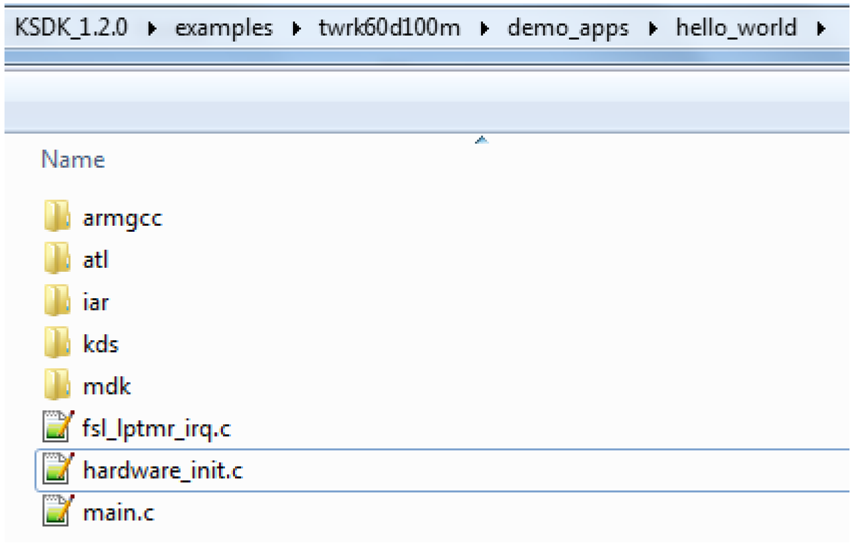
KSDK_1.2.0 ▸ examples ▸ twrk60d100m ▸

Name

- demo_apps
- driver_examples
- board.c
- board.h
- gpio_pins.c
- gpio_pins.h
- pin_mux.c
- pin_mux.h

Modify the Projects

- For SDK1.2, the demos are located at \examples\twrk60d100m\demo_apps and \examples\twrk60d100m\driver_examples
- Take hello_world demo as example.


KSDK_1.2.0 ▸ examples ▸ twrk60d100m ▸ demo_apps ▸ hello_world ▸

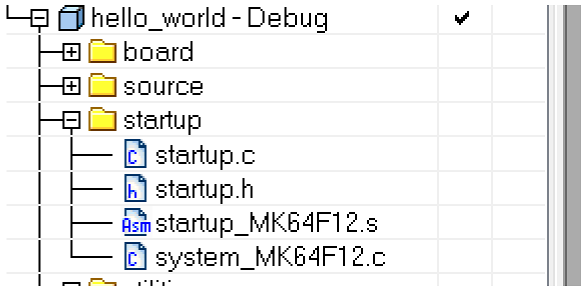
Name

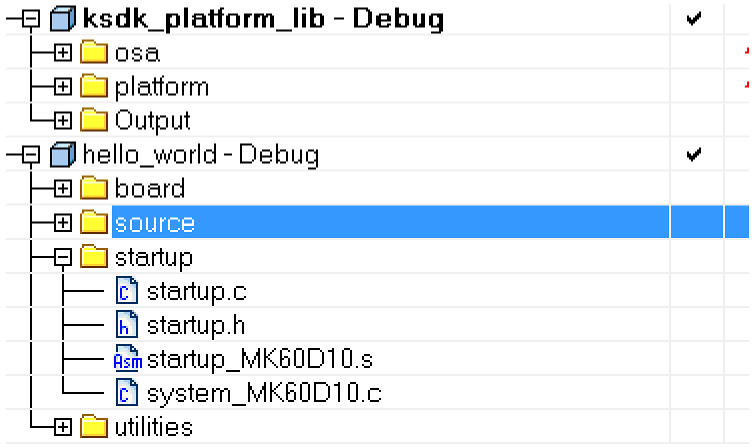
- armgcc
- atl
- iar
- kds
- mdk
- fsl_lptmr_irq.c
- hardware_init.c
- main.c

- Hardware_init.c should be modified according to schematic and reference manual.

Modify the Projects

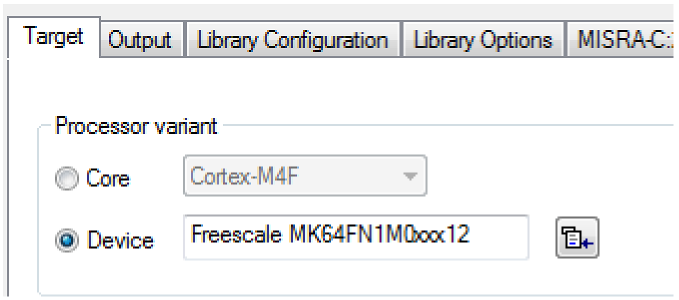
- Take IAR project as example, project option and platform library project included should be modified.





- The included platform library should be changed to the library for K60D10.
- And the startup file included should be changed to files for MK60D10.

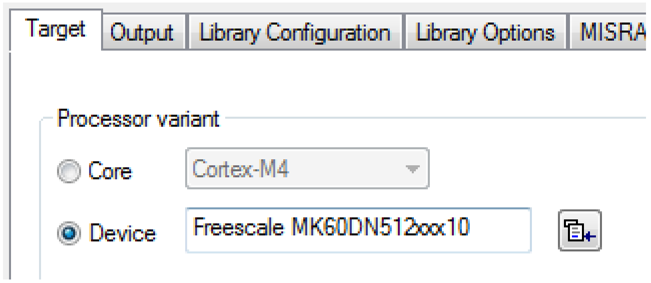
Modify the Projects


Target | Output | Library Configuration | Library Options | MISRA-C:

Processor variant

Core: Cortex-M4F

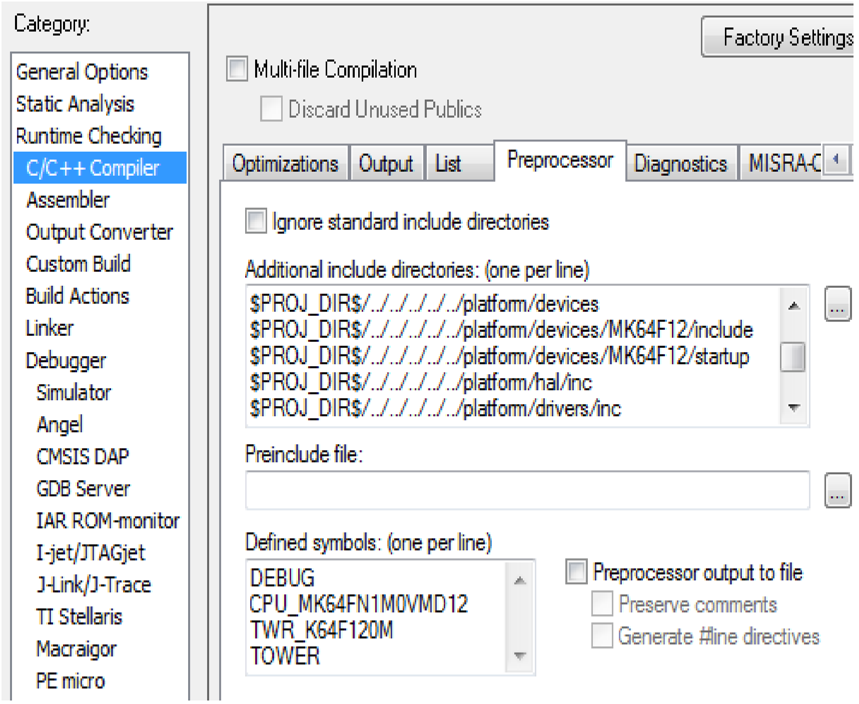
[✓ Device]: Freescale MK64FN1M0xxx12


Target | Output | Library Configuration | Library Options | MISRA

Processor variant

Core: Cortex-M4

[✓ Device]: Freescale MK60DN512xxx10


Category:

General Options
Static Analysis
Runtime Checking
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
CMSIS DAP
GDB Server
IAR ROM-monitor
I-jet/JTAGjet
J-Link/J-Trace
TI Stellaris
Macraigor
PE micro

Factory Settings

Multi-file Compilation
Discard Unused Publics

Optimizations | Output | List | Preprocessor | Diagnostics | MISRA-C

Ignore standard include directories

Additional include directories: (one per line)

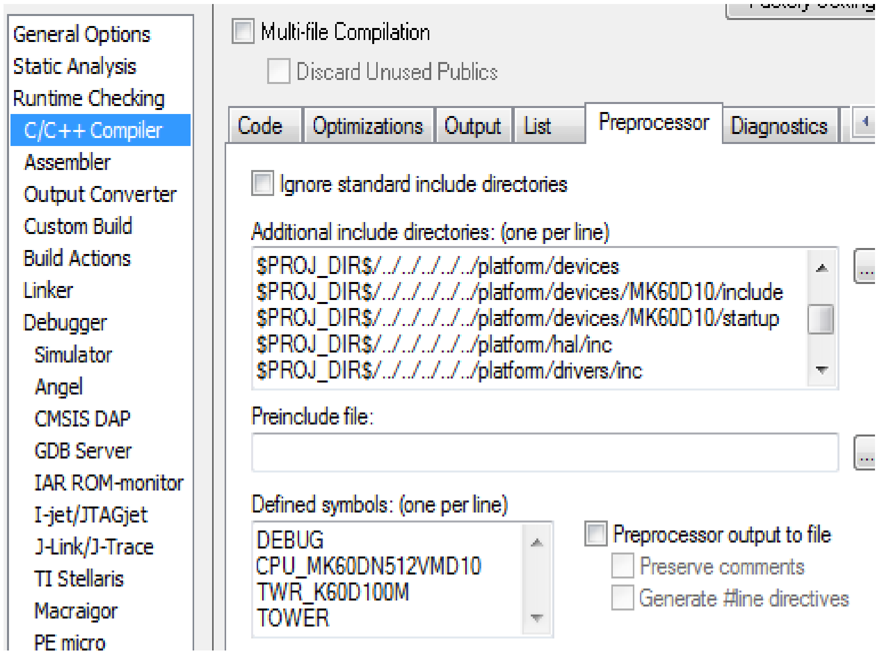
```
$PROJ_DIR$/../../../../../platform/devices
$PROJ_DIR$/../../../../../platform/devices/MK64F12/include
$PROJ_DIR$/../../../../../platform/devices/MK64F12/startup
$PROJ_DIR$/../../../../../platform/hal/inc
$PROJ_DIR$/../../../../../platform/drivers/inc
```

Preinclude file:

Defined symbols: (one per line)

```
DEBUG
CPU_MK64FN1M0VMD12
TWR_K64F120M
TOWER
```

Preprocessor output to file
Preserve comments
Generate #line directives


General Options
Static Analysis
Runtime Checking
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
CMSIS DAP
GDB Server
IAR ROM-monitor
I-jet/JTAGjet
J-Link/J-Trace
TI Stellaris
Macraigor
PE micro

Multi-file Compilation
Discard Unused Publics

Code | Optimizations | Output | List | Preprocessor | Diagnostics

Ignore standard include directories

Additional include directories: (one per line)

```
$PROJ_DIR$/../../../../../platform/devices
$PROJ_DIR$/../../../../../platform/devices/MK60D10/include
$PROJ_DIR$/../../../../../platform/devices/MK60D10/startup
$PROJ_DIR$/../../../../../platform/hal/inc
$PROJ_DIR$/../../../../../platform/drivers/inc
```

Preinclude file:

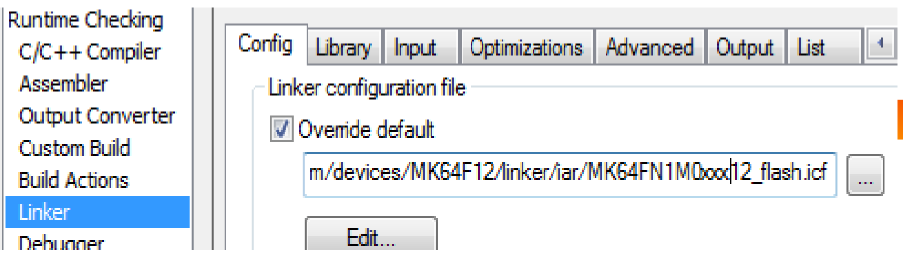
Defined symbols: (one per line)

```
DEBUG
CPU_MK60DN512VMD10
TWR_K60D100M
TOWER
```

Preprocessor output to file
Preserve comments
Generate #line directives

Modify the Projects

- Then change the linker file and link library.



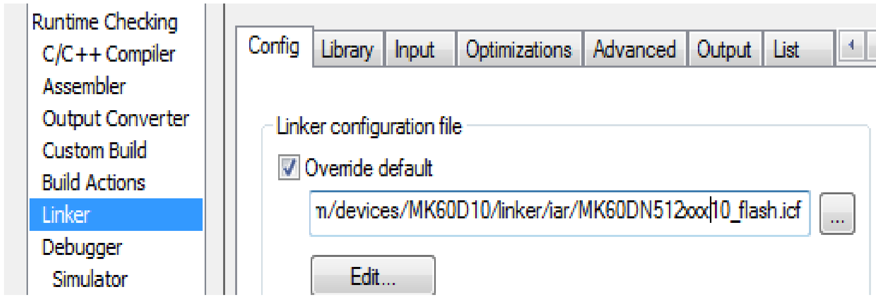
Runtime Checking
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger

Config | Library | Input | Optimizations | Advanced | Output | List

Linker configuration file

☑ Override default

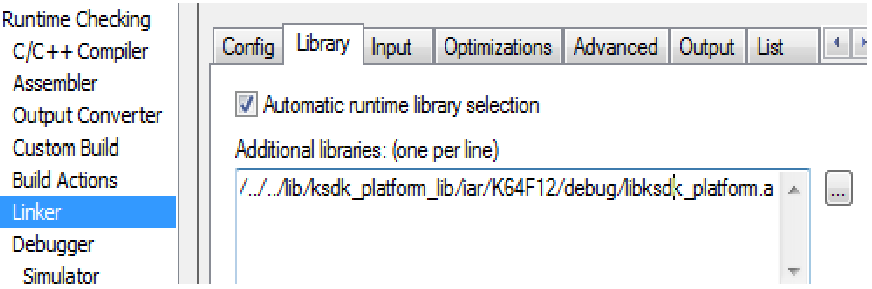
m/devices/MK64F12/linker/iar/MK64FN1M0xxx12_flash.icf



Linker configuration file

☑ Override default

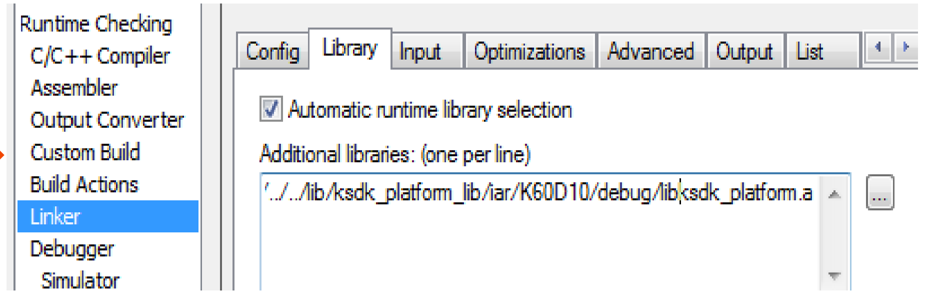
n/devices/MK60D10/linker/iar/MK60DN512xxx10_flash.icf



☑ Automatic runtime library selection

Additional libraries: (one per line)

/../../lib/ksdk_platform_lib/iar/K64F12/debug/libksdk_platform.a



☑ Automatic runtime library selection

Additional libraries: (one per line)

'../../lib/ksdk_platform_lib/iar/K60D10/debug/libksdk_platform.a

Q & A

www.Freescale.com