SDK Migration

Lu Lin

APR.2015

External Use

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 .

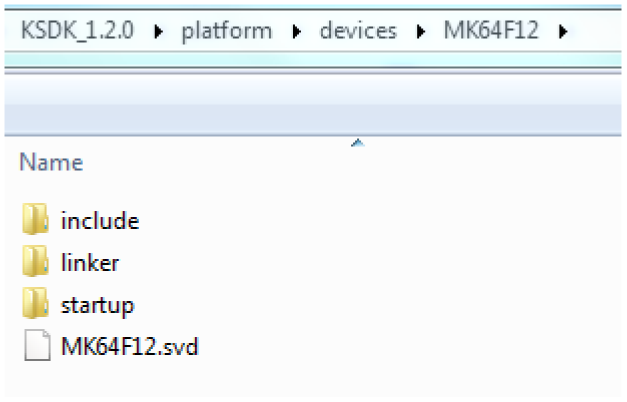
Description of Process

there are four main steps to be porting.

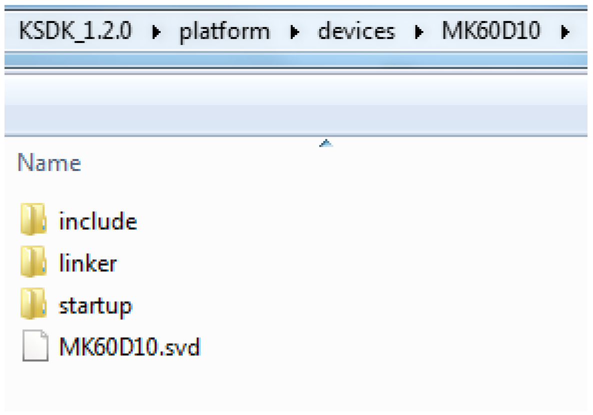
- ✓ **Create the device specific files**
- ✓ **Build the Platform Library**
- ✓ **Create the Board configuration files**
- ✓ **Modify the Projects**

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 ▸

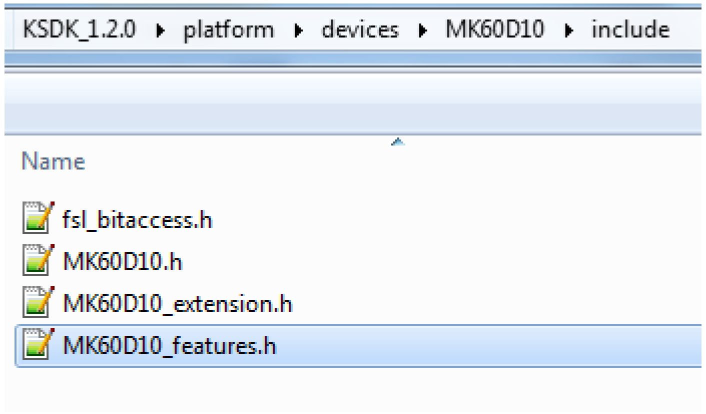
Name
include
linker
startup
MK64F12.svd


KSDK_1.2.0 ▸ platform ▸ devices ▸ MK60D10 ▸

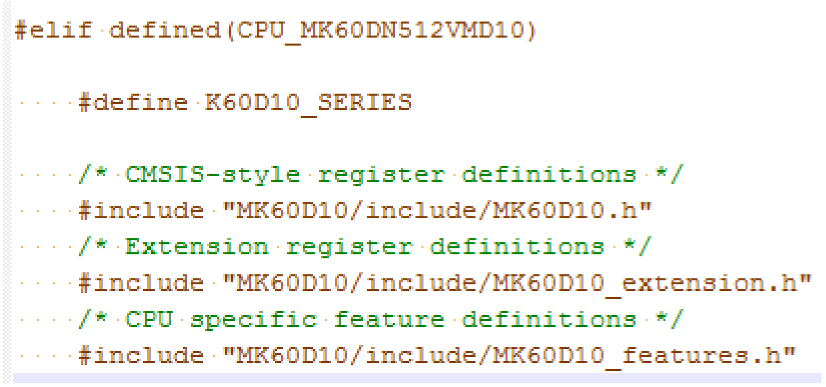
Name
include
linker
startup
MK60D10.svd

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.



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

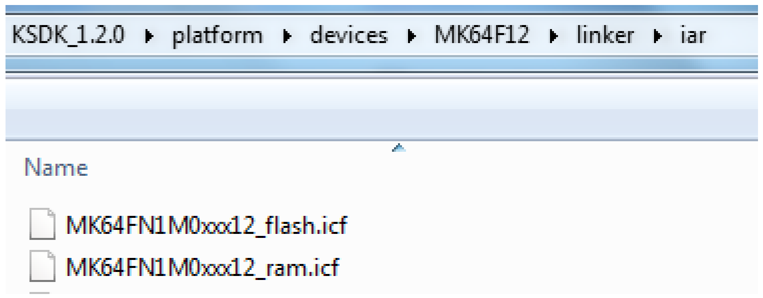


```
#elif defined(CPU_MK60DN512VMD10)

    #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

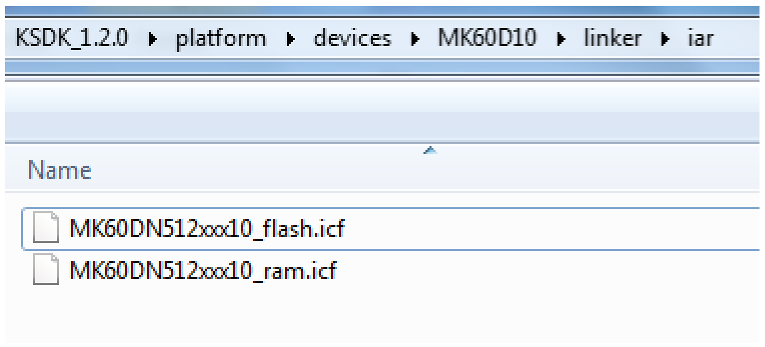
- The directory “linker” includes device specific link file for each tool chain, such as IAR, Keil and so on.
- These files should be modified according to the memory map of K60D10, take IAR as example:


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

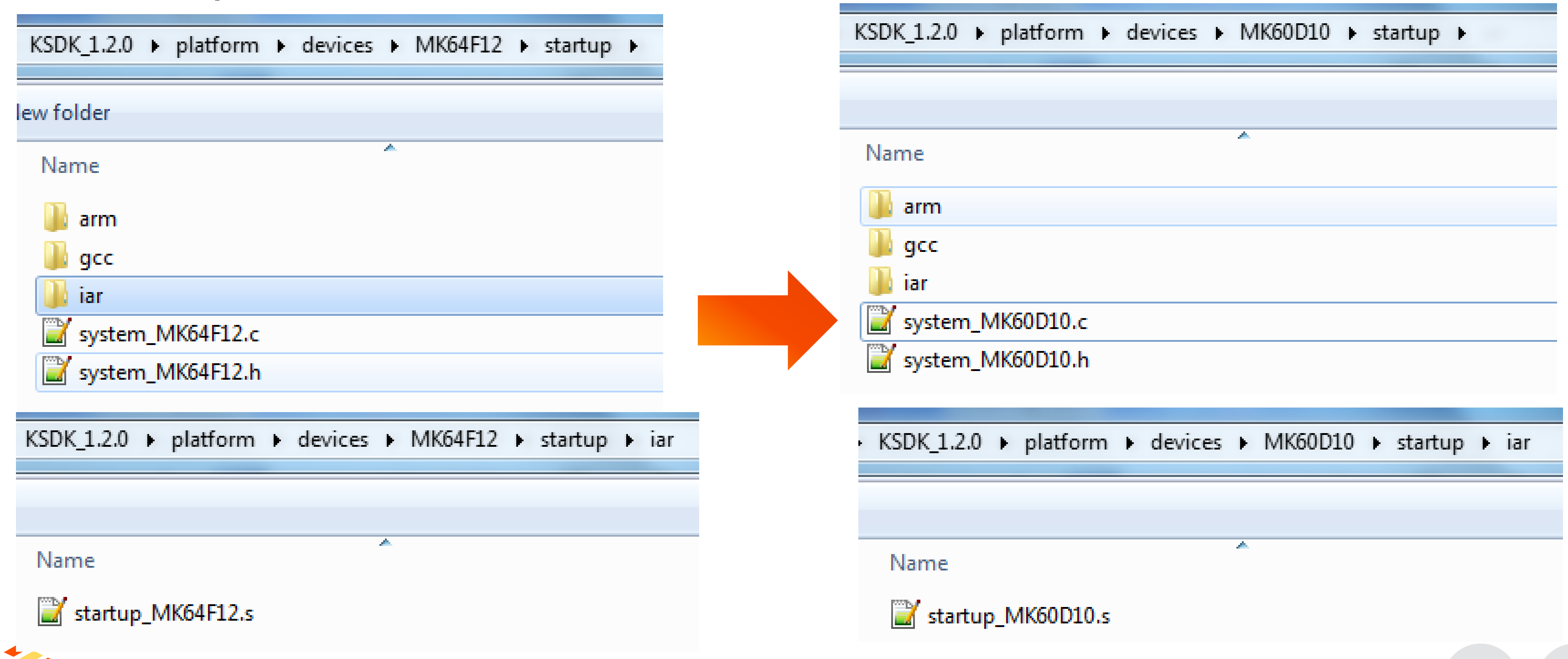
Name

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

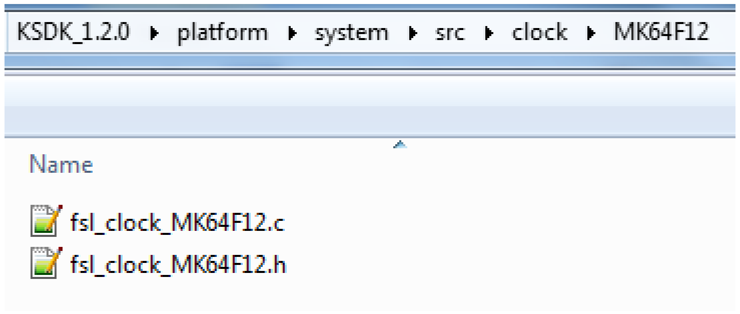
KSDK_1.2.0 ▸ platform ▸ devices ▸ MK60D10 ▸ startup ▸ iar

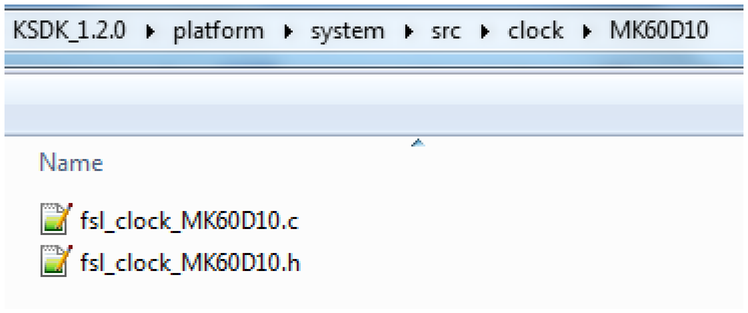
Name

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.



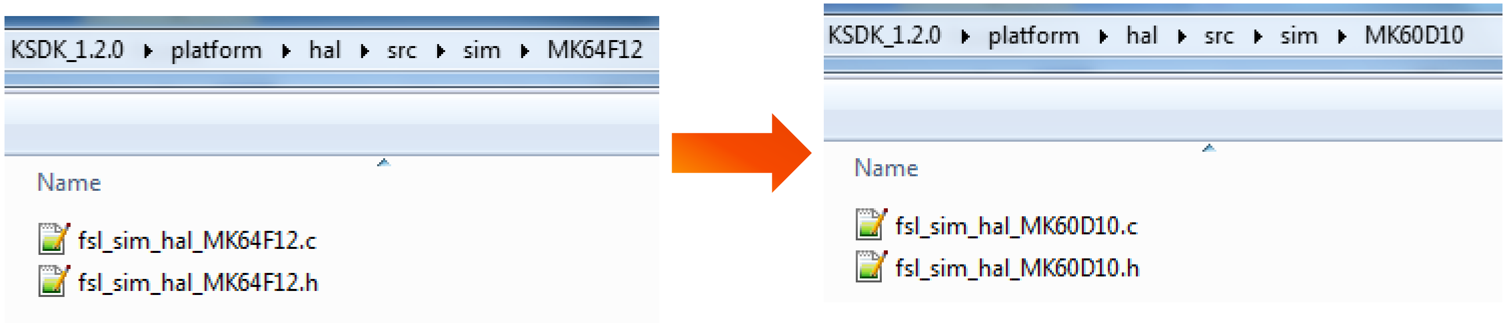


- Then add clock content of K60D10 into common file "fsl_clock_manager.h" which located at \platform\system\inc.

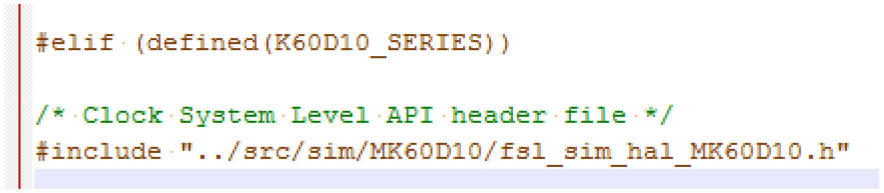
```
#elif (defined(K60D10_SERIES))

    /* 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

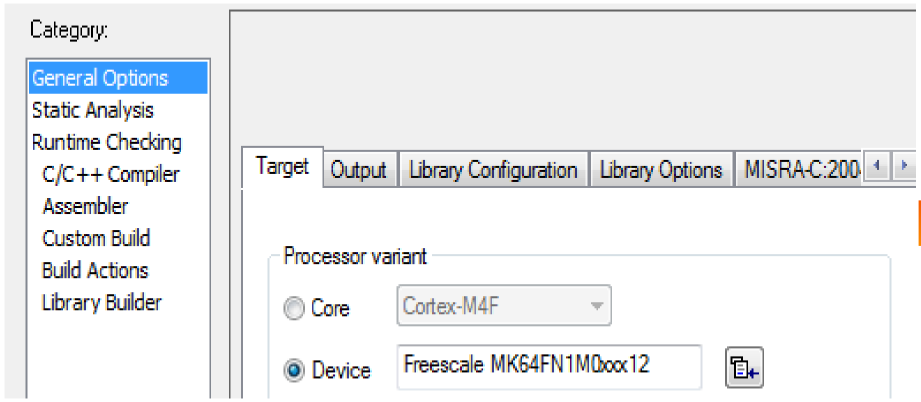
- Then add SIM content of K60D10 into common file “fsl_sim_hal.h” which is located at sdk_install_folder\platform\hal\inc



```
#elif (defined(K60D10_SERIES))

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

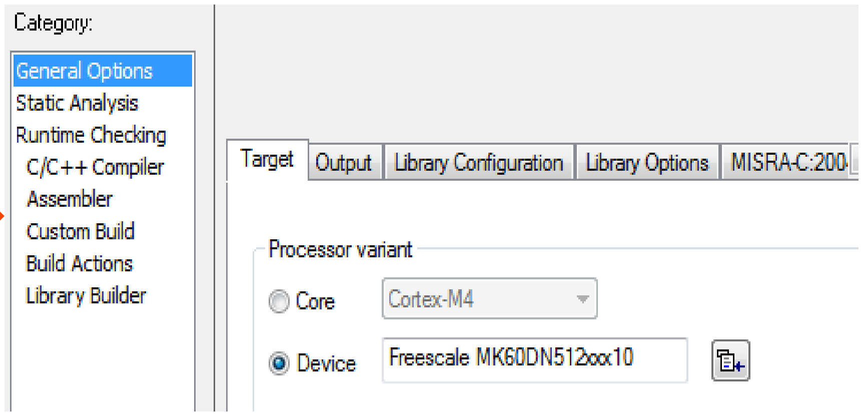
Build the Platform Library

- 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.


Category:
General Options
Static Analysis
Runtime Checking
C/C++ Compiler
Assembler
Custom Build
Build Actions
Library Builder

Target | Output | Library Configuration | Library Options | MISRA-C:200

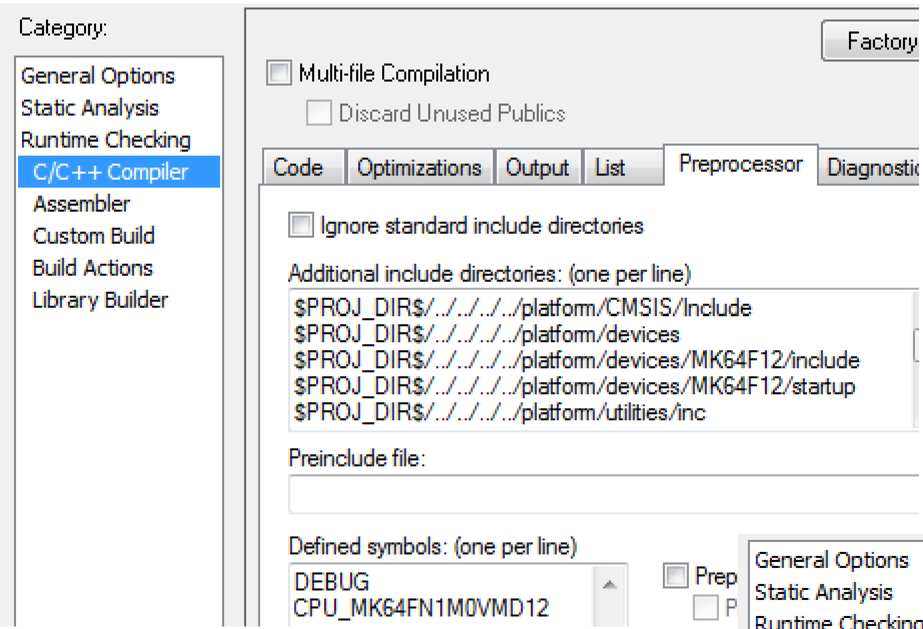
Processor variant
Core: Cortex-M4F
[✓ Device]: Freescale MK64FN1M0xxx12


Category:
General Options
Static Analysis
Runtime Checking
C/C++ Compiler
Assembler
Custom Build
Build Actions
Library Builder

Target | Output | Library Configuration | Library Options | MISRA-C:200

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

Build the Platform Library


Category:

- General Options
- Static Analysis
- Runtime Checking
- C/C++ Compiler
- Assembler
- Custom Build
- Build Actions
- Library Builder

Factory

Multi-file Compilation

Discard Unused Publics

Code | Optimizations | Output | List | Preprocessor | Diagnosti

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
```

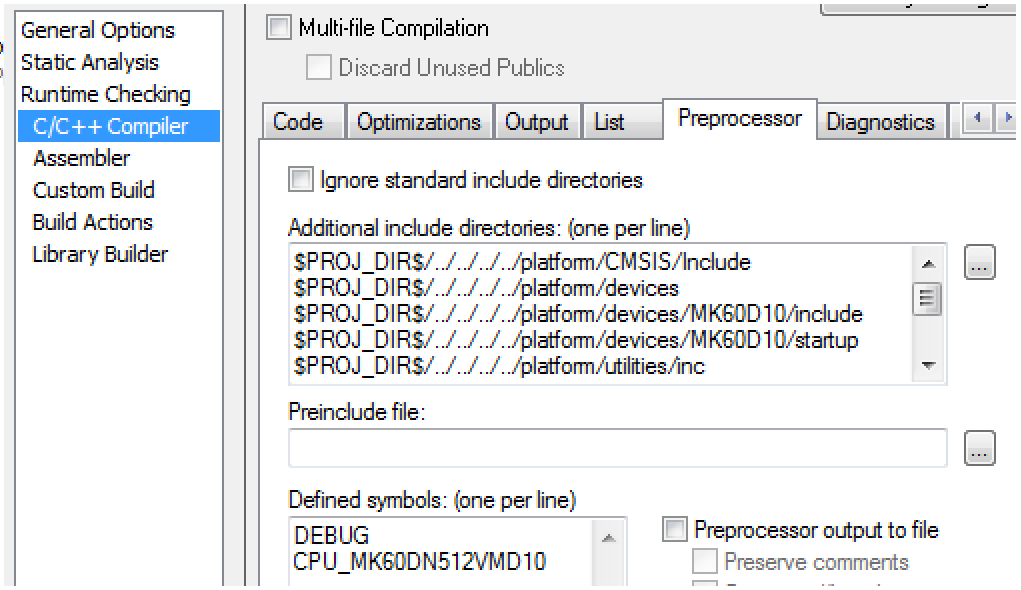
Preinclude file:

Defined symbols: (one per line)

```
DEBUG
CPU_MK64FN1M0VMD12
```

Prep

P


- General Options
- Static Analysis
- Runtime Checking
- C/C++ Compiler
- Assembler
- Custom Build
- Build Actions
- Library Builder

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/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)

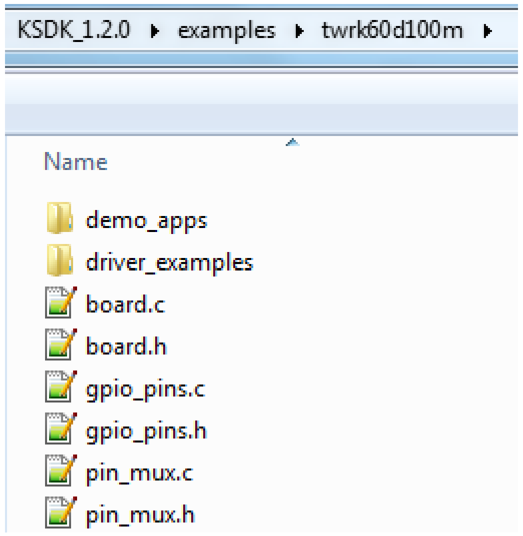
```
DEBUG
CPU_MK60DN512VMD10
```

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.

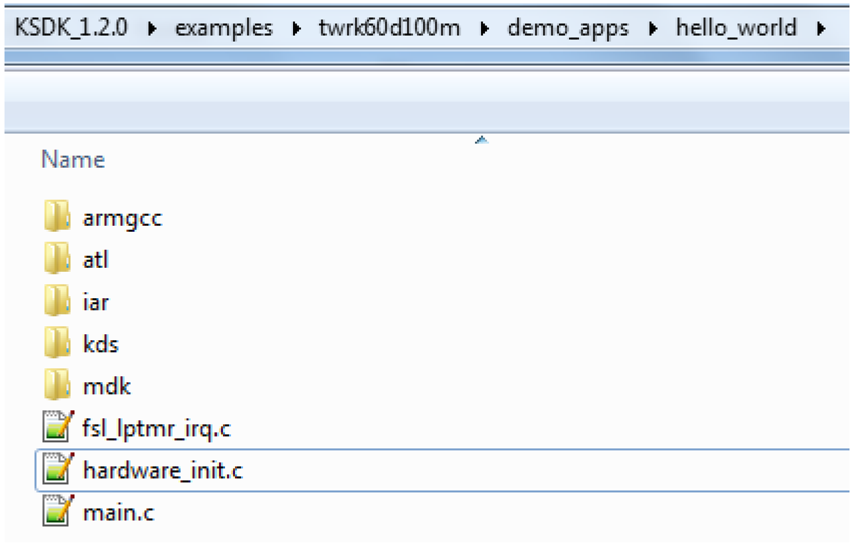

KSDK_1.2.0 ▸ examples ▸ twrk60d100m ▸

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 ▸

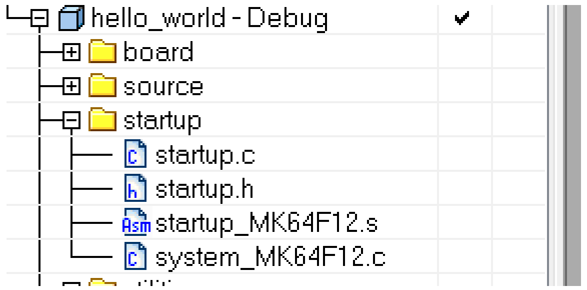
Name

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

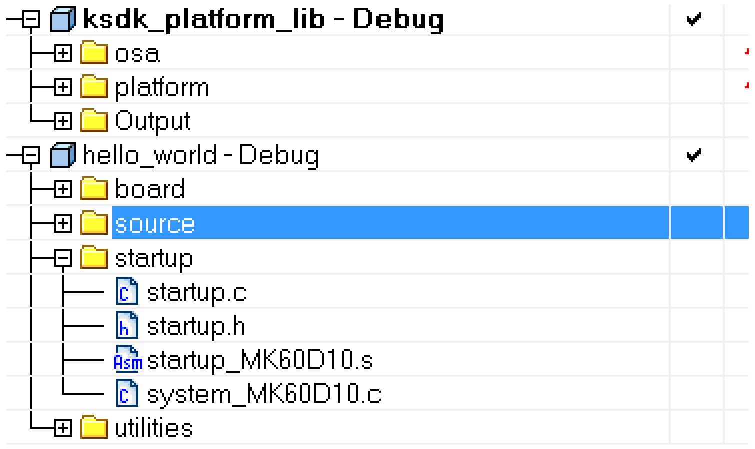
- Hardware_init.c should be modified according to schematic and reference manual.

Modify the Projects

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



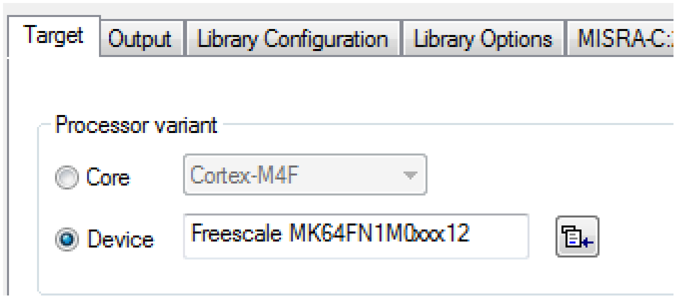
```
hello_world - Debug          ✓
├ board
├ source
├ startup
│ ├ startup.c
│ ├ startup.h
│ ├ startup_MK64F12.s
│ └ system_MK64F12.c
```



```
ksdk_platform_lib - Debug    ✓
├ osa
├ platform
└ Output
hello_world - Debug          ✓
├ board
├ source
├ startup
│ ├ startup.c
│ ├ startup.h
│ ├ startup_MK60D10.s
│ └ system_MK60D10.c
└ utilities
```

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

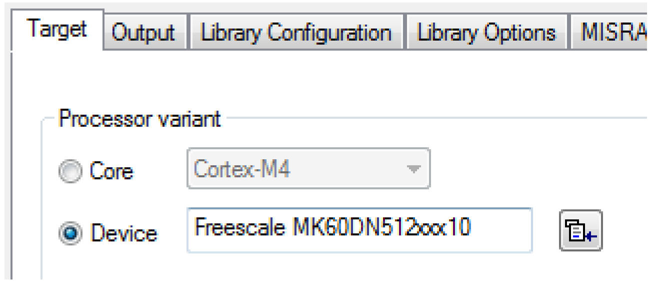
Modify the Projects



Processor variant

Core: Cortex-M4F

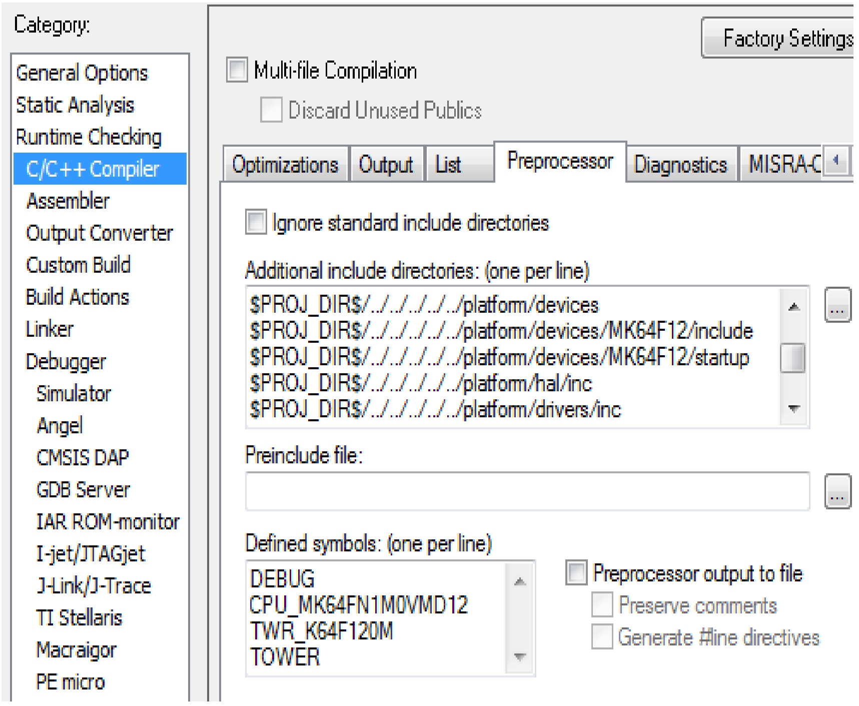
[✓ Device]: Freescale MK64FN1M0xxx12



Processor variant

Core: Cortex-M4

[✓ Device]: Freescale MK60DN512xxx10



☐ Multi-file Compilation

☐ Discard Unused Publics

☐ 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:

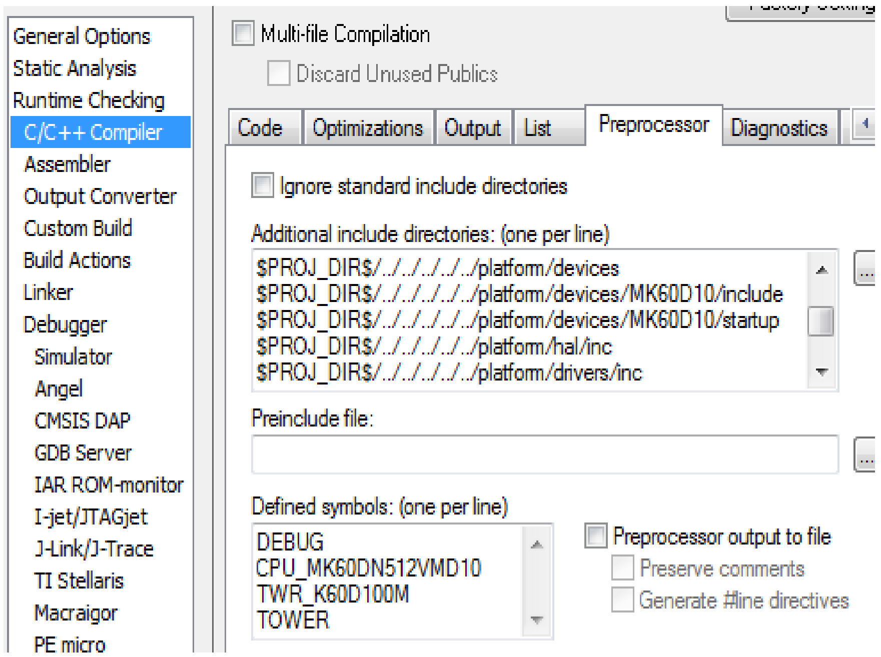
Defined symbols: (one per line)

```
DEBUG
CPU_MK64FN1M0VMD12
TWR_K64F120M
TOWER
```

☐ Preprocessor output to file

☐ Preserve comments

☐ Generate #line directives



☐ Multi-file Compilation

☐ Discard Unused Publics

☐ 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
```

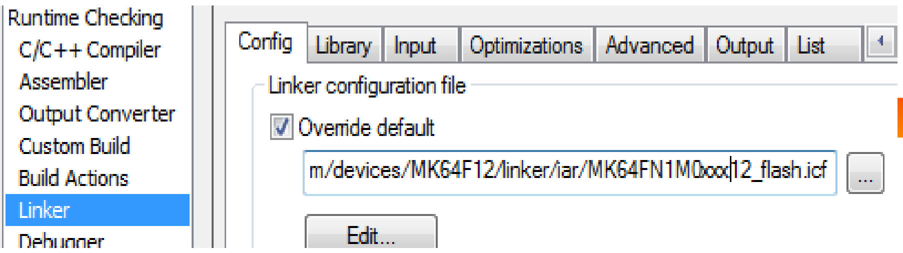
☐ Preprocessor output to file

☐ Preserve comments

☐ Generate #line directives

Modify the Projects

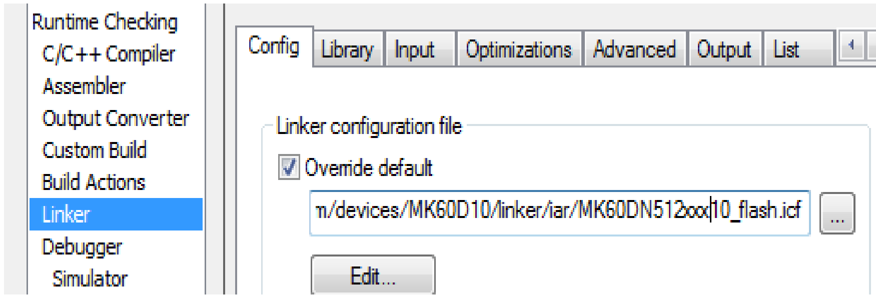
- Then change the linker file and link library.



Linker configuration file

☑ Override default

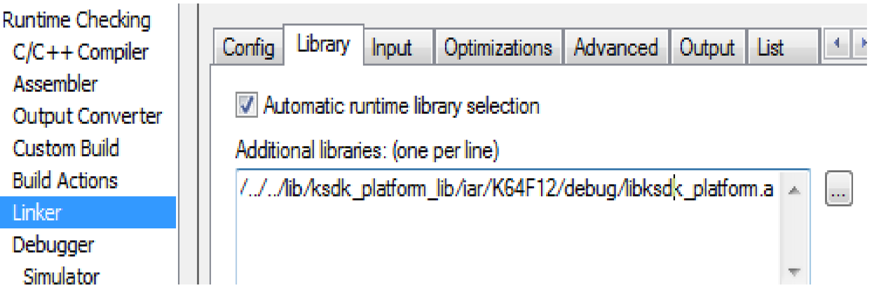
m/devices/MK64F12/linker/iar/MK64FN1M0xxx12_flash.icf



Linker configuration file

☑ Override default

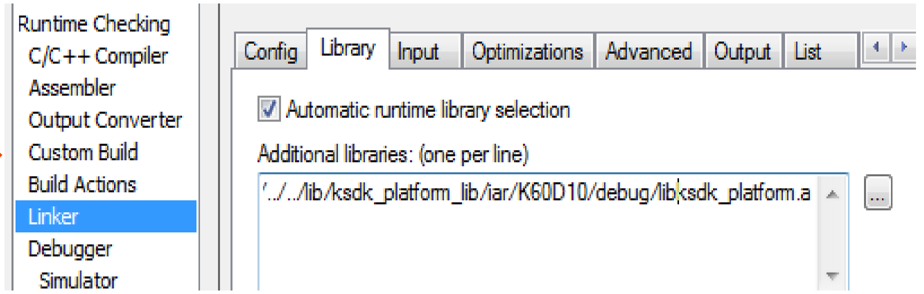
n/devices/MK60D10/linker/iar/MK60DN512xxx10_flash.icf



☑ 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 www.Freescale.com