Kinetis SDK Demo Applications User's Guide

Freescale Semiconductor, Inc.

Document Number: KSDKDEMOUG
Rev. 1.1.0
Oct 2014

Contents

Chapter 1
ADC Hardware Trigger Demo

Chapter 2
ADC Low Power Demo

Chapter 1
ADC Hardware Trigger Demo

This demo application demonstrates how to use the ADC drivers with different hardware triggers.

1.1 Overview

This is an ADC demo application which shows how to use different hardware trigger sources to handle the ADC hardware trigger function. These trigger sources are supported:

- PIT (Periodic Interrupt Timer)
- PDB (Programmable Delay Block)
- LPTMR (Low Power Timer)

1.1.1 Trigger by PIT

The Periodic Interrupt Timer (PIT) is a period timer source and the ADC hardware trigger event. Because the PIT trigger event can only be used to trigger one of the ADC channels (channel 0 or 1), this demo uses PIT as a trigger source for the ADCx channel 0. The PIT triggers the ADC in a fixed frequency and the demo gets the ADC conversion result in the ADC Conversion Complete (COCO) interrupt. TWR-KV10-Z32 does not support PIT trigger, due to no PIT in KV10 silicon.

1.1.2 Trigger by PDB

The Programmable Delay Block (PDB) is a continuous trigger event for ADC. It uses the software trigger as the first trigger input event and turns on the PDB continuous mode to generate a period trigger source. Because the PDB can trigger different channels inside one ADC instance, this demo shows the Ping-Pong triggering which occurs by sampling the channel 0/1 with the PDB Pre-trigger A/B channel.

1.1.3 Trigger by LPTMR

The Low Power Timer (LPTMR) is a period timer source and the ADC hardware trigger event. Because the LPTMR trigger event can only be used to trigger one of the ADC channels (channel 0 or 1), this demo uses the LPTMR as a trigger source for the ADCx channel 0. The LPTMR triggers the ADC in a fixed frequency and the demo gets the ADC conversion result in the ADC Conversion Complete (COCO) interrupt.

1.1.4 Input signal for ADC

Use the DAC module to generate a sine wave as the ADC input on the DAC0_OUT pin. Because the DAC0_OUT is internally connected to the ADC0_SE23 (DAC0_OUT is a source of ADC0_SE23), there is no need to connect any external signals for this demo. For TWR-KV10Z32 Demo, it will need external sine wave due to DAC0_OUT is not coneected to ADC0_SE23 in KV10 silicon.

This demo samples the input digital signal from the ADC0_SE23 pin and records each sample point with the appropriate amplitude. After 2 period samples are complete, it prints out the rough shape of the signal wave on the debug console like a primitive oscilloscope.

1.2 Supported Platforms

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK ADC Hardware Trigger demo.

- FRDM-K22F
- FRDM-K64F
- TWR-K22F120M
- TWR-K64F120M
- TWR-KV31F120M
- TWR-KV10Z32

1.3 Getting Started

1.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

1.4 Run the demo

1. Select and open one project from the three projects available: adc_pit_trigger, adc_lptmr_trigger and adc_pdb_trigger.
2. Open the UART console on a PC.
3. Download and run the program on the target.
4. The signal waveform is displayed on the console.

1.5 Customization Options

This demo application is customizable to show different kinds of input signal waves.

1.5.1 Default configurations

The configuration macro is located in the adc_hw_trigger.h header file.

1.5.1.1 ADC configurations

1. Use ADC0 instance.
2. Use ADC_SE23 input pin as sample pin.
3. Use VREFH/L as reference voltage.

1.5.1.2 Sample frequency

The default sample rate is 20 Hz $*$ 100 / 2, which enables the demo application to get 100 samples per two periods. To change the sample rate, see the next section.

1.5.2 Configure the number of samples

Printing of the signal wave shape depends on the console size. A console can be 100x40. To get the best printing effect, align the number of samples to the console column numbers and convert the amplitude range to the [0, row - 1] range. The console column number should be same as sample numbers. Configuring the number of samples means configuring the console column size:

```
#define CHART_ROWS 30U // chart row for sampled data
#define CHART_COLS 100U // chart column for sampled data
#define NR_SAMPLES 100U // number of samples in one period
```

1.5.3 Configure the signal frequency

Change the following macro to configure the desired frequency in Hz units.

```
#define INPUT_SIGNAL_FREQ 20U // in Hz
```

1.5.4 Configure the ADC instance

Change the ADC_INST macro to configure the ADC instance you want to use.

```
#define ADC_INST 0U  // ADC instance
```


1.5.5 Configure the ADC input pin

If you do not use the DAC0_OUT as a input signal, disable the macro in the project:

```
//#USE_DAC_OUT_AS_SOURCE
```

After disabling the DAC output, configure one ADC input source pin to get the signal:

```
#define ADC_INPUT_CHAN    23U // default input signal channel
```

Chapter 2
ADC Low Power Demo

This demo application demonstrates how to use the ADC drivers in low power modes.

2.1 Overview

The ADC Low Power Demo project is a demonstration program that uses the KSDK software. The microcontroller is set to a very low power stop (VLPS) mode, and every 500 ms an interrupt wakes up the ADC module and takes the current temperature of the microcontroller. While the temperature remains within boundaries, both LEDs are off. If the temperature is higher than average, a red LED comes on. If it is lower, a blue LED(orange LED for TWR-KV10Z32) comes on. This demo provides an example to show how ADC works during a VLPS mode and a simple debugging, "golden" project.

2.2 Supported Platforms

These Freescale Freedom development platforms and Tower System modules are supported by the Kinetis software development kit ADC Low Power demo.

- FRDM-K22F
- FRDM-K64F
- TWR-K22F120M
- TWR-K64F120M
- TWR-KV31F120M
- TWR-KV10Z32

2.3 Getting Started

The ADC Low Power project is designed to work with the Tower System or in a stand alone setting.

2.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

2.3.2 Run the demo

1. Set your target board in a place where the temperature is constant.
2. Press the reset button on your development board.
3. "ADC LOW POWER DEMO" message and some instructions should be displayed on the terminal.
4. Wait until the green or white LED light turns on.
5. Increment or decrement the temperature to see the changes.

Chapter 3
BLDC Sensorless Demo

This demo application demonstrates the software portion (hardware/chip independent) of the 16-bit implementation of a sensorless three phase brushless DC (BLDC) motor control application. The demo supports both IAR and KEIL versions.

3.1 Overview

The BLDC sensorless Control Demo project is a demonstration program that uses the KSDK software. The application software uses the concept of an isolated algorithm software and hardware. This software approach enables easy porting of an application to other devices or platforms. The application software is divided in two sections: • BLDC motor control algorithm process input variables to output variables and flags. • MKV10x hardware and microprocessor serves as a bridge between hardware peripheral modules and BLDC motor control software algorithm.

3.2 Supported Platforms

This Tower System module is supported by the Kinetis software development kit ADC Low Power demo.

- TWR-KV10Z32

3.3 Getting Started

This table lists the FTM channels and MCU pins and corresponding LEDs for this demo application. This table also lists which connections should be made (if any) to ensure proper demo operation.

1. TWRMCLV3PH jumper settings

jumper	position
J2	1-2
J3	1-2
J10	2-3
J11	2-3
J12	2-3
J13	2-3
J14	1-2

2. TWR-KV10Z32 jumper settings

jumper	position	jumper	position	jumper	position
J1	2-3	**J10**	1-2	**J21**	3-4

J2	short	**J11**	open	**J22**	3-4
J3	2-3	**J12**	open	**J25**	open
J4	short	**J13**	open	**J26**	short
J5	short	**J14**	open	**J27**	short
J7	1-2	**J18**	2-3	**J28**	short
J8	1-2	**J19**	2-3	**J29**	1-2
J9	1-2	**J20**	1-2	**–**	–

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board. Note that, because of board limitations, if the power is not supplied to OpenSDA, the KV10 reset pin is in low level.

3.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Connect three phases of the BLDC motor to J5 in the TWRMCLV3PH board.
5. Supply 24 V digital power to J1 in the TWRMCLV3PH board.
6. Either press the reset button on your board or launch the debugger in the IDE to start running the demo.

For detailed instructions, see the appropriate board User's Guide.

3.3.2 Run the demo

The application can be controlled using one interface:

• Up / Down buttons on the TWR-KV10Z32 board

1. After the power supply is plugged into the TWR-MC-LV3PH, the motor is ready to run.
2. Press the reset button on the development board.
3. Pressing the Up button (SW1) increases the speed by 500 RPM. The motor starts rotating in the clockwise direction if it is not spinning, or decreases speed if the direction of the rotation is counter-clockwise.
4. Pressing the Down button (SW2) decreases the speed by 500 RPM. The motor starts rotating in the counter-clockwise direction if it is not spinning, or decreases speed if the direction of the rotation is clockwise.

5. Pressing the buttons beyond this point increases or decreases the required speed within the speed limit -5000 to 5000 RPM.
6. If both buttons are pressed for more than 2 seconds, the demonstration mode is switched on (or demonstration mode is switched off if it is on)

Chapter 4
DAC ADC Demo

This demo application demonstrates the DAC and ADC demo.

4.1 Overview

This application demonstrates how to configure the DAC and set the output on the DAC using software. It also demonstrates how to configure the ADC in 'Blocking Mode' and read ADC values.

4.2 Supported Platforms

This demo supports these Freescale Freedom development platforms and Tower System modules:

- FRDM-K22F120M(K02)
- FRDM-K64F120M
- FRDM-KL46Z48M
- TWR-K22F120M(128R\256R)
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z75M
- TWR-KV31F120M(128R\256R)

4.3 Getting Started

4.3.1 Hardware Settings

This table shows the connections that are required for each of the supported platforms:

Platform	DAC Out		ADC In	
	Pin Name	Board Location	Pin Name	Board Location
FRDM-K22F120-M(K02)	DAC0_OUT	J24-11	PTB0/ADC0_SE8	J24-2
FRDM-K64F120-M	DAC0_OUT	J4-11	PTB2/ADC0_SE12	J4-2
FRDM-KL46Z48-M	PTE30/DAC_OUT	J4-11	PTE20/DIFF_AD-C0_DP	J4-1
TWR-K22F120-M(128R\256R)	DAC0_OUT	Primary Elevator - A32	PTB0/ADC0_SE8	Primary Elevator - B27
TWR-K24F120M	DAC0_OUT	Primary Elevator - A32	ADC0_DP3	Primary Elevator - A29

TWR-K60D100M	DAC0_OUT	Primary Elevator - A32	PTB4/ADC1_SE10	Primary Elevator - B30
TWR-K64F120M	DAC0_OUT	Primary Elevator - A32	PTB4/ADC1_SE10	Primary Elevator - B27
TWR-KV10Z75M	DAC0_OUT	J16-11	PTE17/ADC0_SE5	J16-6
TWR-KV31F120-M(128R\256R)	DAC0_OUT	Primary Elevator - A32	PTE2/ADC1_SE6a	Primary Elevator - B27

4.3.2 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

4.4 Run the demo

This example shows how to run the demo on TWR-K22F120M:

```
DAC ADC Demo!
Please refer to Kinetis SDK Demo Applications User's Guide document,
Chapter DAC ADC demo, for pins configuration information.
Press space bar to start demo.
```

The user is prompted to enter a voltage to output on the DAC:

```
Select DAC output level:
        1. 1.0 V
        2. 1.5 V
        3. 2.0 V
        4. 2.5 V
        5. 3.0 V
->
```

After entering a valid input, the ADC captures the voltage set by the DAC and displays the result in the terminal:

```
Select DAC output level:
        1. 1.0 V
        2. 1.5 V
        3. 2.0 V
        4. 2.5 V
        5. 3.0 V
->3


ADC Value: 2471


ADC Voltage: 1.99


What next?:
        1. Test another DAC output value.
        2. Terminate demo.
->
```

At this point, the user can test another DAC output value or terminate the demo.

This configuration exhibits up to 2% error when reading back voltage.

4.5 Key Functions

uint8_t demo_start(demo_state_t *prevState)

Prints out a welcome message and pins required by the demo.

Parameters

*prevState	Pointer to previous state for state machine.

Returns

msg Returns the character entered into the terminal by the user.

uint8_t device_config(demo_state_t *prevState)

Configures the DAC and the ADC. The DAC is configured for software updates. The ADC is set in 'Blocking Mode'.

Parameters

∗prevState	Pointer to previous state for state machine.

Returns

msg Returns 0.

uint8_t dac_set(demo_state_t ∗prevState)

Sets output level on the DAC.

Parameters

∗prevState	Pointer to previous state for state machine.

Returns

msg Returns the character entered into the terminal by user.

uint8_t wait_state(demo_state_t ∗prevState)

Performs a wait and possible state change based on the ∗prevState.

Parameters

∗prevState	Pointer to previous state for state machine.

Returns

msg Returns 0.

uint8_t adc_get(demo_state_t ∗prevState)

Gets ADC values from a channel connected to the DAC output.

Parameters

∗prevState	Pointer to previous state for state machine.

Returns

msg Returns the character entered into the terminal by the user.

uint8_t device_deinit(demo_state_t *prevState)

Deinitializes the DAC and the ADC module following a user command to terminate the demo. Also frees allocated memory.

Parameters

∗prevState	Pointer to previous state for the state machine.

Returns

msg Returns 0.

uint8_t demo_end(demo_state_t ∗prevState)

Indicates to the user that the demo has been terminated.

Parameters

∗prevState	Pointer to previous state for the state machine.

Returns

msg Returns 0.

Chapter 5
DSPI EDMA HAL Demo

This demo application demonstrates how to use the DSPI and EDMA Hal layers to send and receive data.

5.1 Overview

The DSPI EDMA HAL demo is a demonstration program that uses the KSDK software. For low density devices, which do not have enough resources to use the KSDK driver functions, use the KSDK HAL layer functions. This demo needs two boards, a board for master and a board for slave. The master sends data from the SPI port with the DMA. The slave receives data with the DMA and displays the results on the terminal.

5.2 Supported Platforms

This Tower System module is supported by the Kinetis software development kit DSPI EDMA HAL demo.

- TWR-KV10Z32

5.3 Getting Started

This project is designed to work with the Tower System or in a stand alone setting.

5.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For detailed instructions, see the appropriate board User's Guide.

5.3.2 Run the demo

1. Connect the SPI port for the master board and slave board. The SPI pin mux is in pinmux.c.
2. Press the reset button on the slave board.

3. Press the reset button on the master board.
4. The master board prints the message and data it received on the slave board.

Chapter 6
Flash Demo

This demo application demonstrates how to use the Flash drivers.

6.1 Overview

The Flash demo project shows how to erase, program, and performs swap (if available) on the Flash module. Note:

1. Target exists for Flash memory space. Since the demo will operate with two last sectors of lower half and the whole upper half of Program flash memory of the platforms with SWAP feature; six last sector of Program flash of the platforms without SWAP feature, the user should NOT store any program code or data in the above locations.
2. The flash swap demo will be failed if the tested board has aready run swap command with swap indicator address different from the value defined in demo. To overcome the issue, erase all chip to un-initialize the swap system and re-run the demo. The features include:

1. Security status report
2. Read to non-volatile information memory region
3. Flash Erase by block or sector, including margin read options
4. Programming region defined by user
5. Flash verify support
6. Flash Swap (if supported on device)

6.2 Supported Platforms

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK Flash demo.

Platforms with SWAP feature:

- FRDM-K64F120M
- TWR-K64F120M
- TWR-K60D100M

Platforms without SWAP feature:

- FRDM-K22F12M
- FRDM-K22F120MK02
- FRDM-K22F120MK0264
- FRDM-KL03Z48M
- FRDM-KL46Z48M
- TWR-K22F120M
- TWR-K22F120M128
- TWR-K22F120M256

- TWR-K22F120MK02
- TWR-K24F120M
- TWR-KV10Z75M
- TWR-KV30F100M
- TWR-KV30F100MK02
- TWR-KV30F100MK0264
- TWR-KV31F120M
- TWR-KV31F120M128
- TWR-KV31F120M256
- TWR-KV31F120MKV30

6.3 Getting Started

The Flash Demo example code shows how to erase and program the Flash content and use the swap feature if it is supported on the device.

6.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 (9600 for FRDM-KL03Z48M) for baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

6.4 Commands/Directions

1. Select the Debug target from within the IDE and build the project selected for the target hardware. The default Debug target runs from flash and demonstrates the Swap feature for devices that support Swap (e.g. TWR-K64F120M).
2. Connect one end of the USB cable to a PC host and the other end to the OpenSDA connector on the board.
3. Open Terminal program such as TeraTerm, Putty, or Hyperterminal.
4. Configure the Terminal program to select the OpenSDA COMx port for the board using
   - 115200 8N1: 115200 baud, 8 data bits, No parity, 1 Stop bit.
   - Or FRDM-KL03Z48M 9600 8N1: 9600 baud, 8 data bits, No parity, 1 Stop bit.
5. Connect to the board with the debugger (download & debug), run the program, and view the Terminal messages for Flash operations being performed.

6. For devices that support Swap, the Flash_Debug target copies (programs) the application that is running from the lower half to the upper half and then issues swap commands.
7. Flash memory blocks are swapped at the next reset. Disconnect debug session and hit the reset button on the board. Note: During swap, some memory locations depending on program flash size (e.g. for TWR-K64F120M: 0x7F100 & 0xFF100) are swapped and displayed on the terminal showing how the memory map changes.
8. For devices that do not support swap, view the terminal messages for Flash operations that are occurring for the demo.
9. Terminal displays the message "Flash Demo Complete!" when finished.
   Note: Callback functions are not currently supported during flash erase or program operations
   Note: For K22F and KV31, Flash erase and program operations are not allowed in High-Speed RUN modes. Therefore, the core clock speed is restricted to 80 MHz or less.

Chapter 7
FlexCAN and UART communication Demo

This demo application demonstrates how to use the FlexCAN and UART drivers.

7.1 Overview

This is a FlexCAN and UART communication demo which demonstrates the communication between two boards which is handled by FlexCAN and the UART input/output. On one board, the user inputs characters by using the UART debug terminal and sends the data with the FlexCAN interface. On the other board, the FlexCAN receives the data and prints them to the UART terminal.

7.2 Supported Hardware

This Tower System module is supported by the KSDK FlexCAN and UART demo.

- TWR-K64F120M

7.3 Getting Started

7.3.1 Hardware configuration

TWR-SER Tower System module configuration

1. Short J5(1-2), J5(3-4), J5(5-6), J5(7-8), and J5(9-10) to enable CAN connection.
2. Connect the two TWR-SER modules through the CAN port (J7).

7.3.2 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

7.4 Run the demo

1. Select the node ID for each board by typing A/a or B/b followed by Enter.
2. You may now transfer characters by using serial terminals. For more details, see the video in the //video folder.

Chapter 8
FreeMASTER Demo

This demo application demonstrates usage of the FreeMASTER graphical visualization tool.

8.1 Overview

Modules

- FreeMASTER Demo Introduction
  This part provides an introduction to the FreeMASTER Demo application.
- FreeMASTER Demo User's Guide
  This part provides a user's guide to the FreeMASTER Demo application.

8.2 FreeMASTER Demo Introduction

This chapter provides an introduction to the FreeMASTER Demo application.

8.2.0.1 FreeMASTER Demo Introduction

Overview

The FreeMASTER Demo application project is a simple demonstration showing how to integrate the Freescale FreeMASTER communication driver into a KSDK-based application. You should be able to use the FreeMASTER tool running on Host PC and connect it over a serial line, CAN interface or BDM cable to this demo application. The demo.pmp file can be opened in FreeMASTER tool project for a quick access to selected variables. The variables can be monitored in a text grid or in a graph; you can also try to modify the values directly in the variable watch grid.

In addition to the variable monitoring, the FreeMASTER tool can be also used to build rich graphical control panels for any embedded application. See more information on the FreeMASTER home page at www.freescale.com/freemaster.

Supported Platforms

The FreeMASTER tool and the driver are two software packages distributed separately, outside of KSDK. This release of KSDK contains a copy of FreeMASTER driver version 1.8.1 which supports the whole Kinetis K and Kinetis L families and others platforms.

The KSDK freemaster_demo application has been tested with the following boards: frdmk22f120m frdmk64f120m frdmkl03z48m(notice: can not support the clock configuration of CLOCK_SETUP=2 in kl03); twrk22f120m twrk22f120m128r twrk22f120m256r twrk60d100m twrk64f120m twrkv10z75m twrkv31f120m twrkv31f120m128r twrkv31f120m256r

However the driver included in the src folder supports other Kinetis platforms too.

Please get the FreeMASTER PC Host tool installer from the www.freescale.com/freemaster web site.

8.3 FreeMASTER Demo User's Guide

This chapter provides a user's guide to the FreeMASTER Demo application.

8.3.0.2 FreeMASTER Demo User's Guide

Getting Started

Open the demo application project in your favorite development environment. In case the application project is not ready out of the box for your particular board, you can create an empty KSDK stationery project or reuse any other demo which works with your hardware. Adding FreeMASTER to an existing project is very easy just by adding relevant source files available in the demos/freemaster_demo/src/driver-_v1_8 folder.

Connecting with UART (default)

Inspect the freemaster_cfg.h file to see the FreeMASTER configuration. The serial line is enabled by default. The default console port is used for communication with FreeMASTER PC Host tool. Do not use the demo with a console terminal tools like with most of the other KSDK examples. There are also other communications options available, like BDM or CAN interface. Consult the user documentation for more information.

Running the FreeMASTER PC Host Tool

Compile and download the program to a target board. Connect serial cable to the PC host computer and start the FreeMASTER application (installer available at www.freescale.com/freemaster). Use the Start menu in the MS Windows environment to lookup the FreeMASTER launcher item. It is recommended that you also read the FreeMASTER user documentation available also in the Start menu group.

Using the FreeMASTER Demo project

In the FreeMASTER tool main window, press the Ctrl+K key first to select the serial COM port which connects your host computer with the target board. Select the 115200 as the default communication speed and press OK. Now use the Ctrl+O to open the demo.pmp project file. The file is available in the same folder as the freemaster_demo application in the KSDK package. If everything goes well, you should see values of several variables in the variable watch grid in the bottom area of the main FreeMASTER window. Click the Oscilloscope item in the project tree on the left side and see the graphical representation of the selected variables.

Trobleshooting

In case of any problems with the freemaster_demo application, try to start with the simplest hello_world example and a console terminal application to verify whether the serial line communication works well. The most typical problem why FreeMASTER can not communicate with the board is a wrong COM port selected on the host PC, wrong UART port selected on the target processor side or a bad communication speed.

Getting more information

The FreeMASTER is a versatile and powerful tool in which you can create your own interactive graphical monitors and control panels for any embedded application. To understand how to use all FreeMASTER features, scripting and communication options, refer to User Guide document included with the FreeMASTER installation package. The target-side communication driver is further described in a User Manual included in the official distribution of the FreeMASTER Serial Driver, which is a separately downloadable package available at www.freescale.com/freemaster.

Chapter 9
FTM PDB ADC Demo

This demo application demonstrates how to use FTM external trigger to start ADC conversion via PDB.

9.1 Overview

This application demonstrates how to use the FTM external trigger to start the ADC conversion using the PDB. The FTM0 is configured as a complementary combined mode. Each channel output frequency is 16 KHz. The complementary channel dead time is 1 µs. The PDB pre-trigger works in back-to-back mode. The ADC0 and ADC1 work in single-end mode. The ADC0 uses channel 1 and channel 5. ADC1 uses channel 1 and channel 7.

9.2 Supported Platforms

This Tower System module is supported by the KSDK FTM PDB ADC demo.

- TWR-KV10Z32(75M)

9.2.1 Prepare the Demo

Use default jumper settings on TWR-KV10Z32. Ensure that the J21(2∼3 is short), J22(2∼3 is short), J11(1∼2 is short, 3∼4 is short),J12(1∼2 is short, 3∼4 is short).

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on the board or launch the debugger in the IDE to begin running the demo.

For detailed instructions, see the appropriate board User's Guide.

9.2.2 Run the demo

1. Download and run the ftm_pdb_adc code on the board.
2. Terminal prints this message: "Run PDB trig ADC with FlexTimer demo." and "Input any character to start demo."

3. Input a character to the serial terminal, which has 256 lines of information for the ADC conversion result.
4. Input any character to the serial terminal. The process repeats again.

Chapter 10
FlexTimer PWM Demo

This demo application demonstrates the FlexTimer PWM demo.

10.1 Overview

This application demonstrates the FTM edge-aligned PWM function. It outputs the PWM to control the intensity of the LED.

10.2 Supported Platforms

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK FlexTimer demo.

- FRDM-K22F120M(K02/K0264)
- FRDM-K64F120M
- TWR-K22F120M(128R/256R/K02)
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z32(75M)
- TWR-KV31F120M(128R/256R/KV30)

10.3 Getting Started

Hardware configuration

This table lists the FTM channels and MCU pins and the corresponding LEDs for this demo application. This table also lists which connections should be made (if any) to ensure proper demo operation.

Platform	FTM Instance/-Chnl	MCU Pin	LED/Color	Jumper Config
FRDM-K22F120-M(K02/K0264)	FTM0 - CH5	PTD5	D14 - Blue	N/A
FRDM-K64F120-M	FTM0 - CH0	PTC1	D12 - Green	J1/P5 - J2/P1
TWR-K22F120-M(128R/256R/-K02)	FTM0 - CH4	PTD4	D7 - YEL/GRN	J16/P1-P2
TWR-K24F120M	FTM0 - CH4	PTD4	D7 - Yellow	J3/P1-P2
TWR-K60F100M	FTM2 - CH1	PTA11	D7 - Orange	N/A

TWR-K64F120M	FTM3 - CH1	PTE6	D5 - YEL/GRN	J30/P1-P2
TWR-KV10-Z32(75M)	FTM0 - CH1	PTE25	D2 - YEL	NA
TWR-KV31F120-M(128R/256R/K-V30)	FTM0 - CH7	PTD7	D7 - YEL/GRN	J17/P7-P8

NOTE: The FRDM-K64F120M requires an external jumper wire to make the appropriate connections.

10.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with the following settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on the board or launch the debugger in the IDE to begin running the demo.

For detailed instructions, see the appropriate board User's Guide.

10.3.2 Run the demo

1. Download and run the ftm_pwm code on the board.
2. Terminal prints the message "Welcome to FTM PWM demo!"
3. The LED on the board increases/decreases intensity according to PWM pulse width changes. The LED affected is indicated in the table in the Hardware Configurations part.

Chapter 11
Hello World Demo

This demo application demonstrates the Hello World demo.

11.1 Overview

The Hello World project is a simple demonstration program that uses the KSDK software. It prints the "Hello World" message to the terminal using the KSDK UART drivers. The purpose of this demo is to show how to use the UART and to provide a simple project for debugging and further development.

11.2 Supported Platforms

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK Hello World demo.

- FRDM-K22F
- FRDM-K64F
- TWR-K22F120M
- TWR-K64F120M
- TWR-KV31F120M

11.3 Getting Started

11.3.1 Hardware Settings

The Hello World project does not call for any special hardware configurations. Although not required, the recommendation is to leave the development board jumper settings and configurations in default state when running this demo.

11.3.2 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

Communication Interface Settings:

For detailed instructions, see the appropriate board User's Guide.

11.4 Run the demo

This is an example how to run the demo.

```
Hello World!
```

11.5 Communication Interface Settings:

This part provides the information to customize the Hello World demo. The Hello World demo is configured to use these port pins for the platforms by default. If applicable for the board, jumpers are specified to select between serial output via OpenSDA and serial output via TWR-SER.

Platform	TX MCU Pin (Board Pin)	RX MCU Pin (Board Pin)	Module Instance
FRDM-K22F	PTE0 (N/A)	PTE1 (N/A)	UART1
FRDM-K64F	PTB17 (N/A)	PTB16 (N/A)	UART0
TWR-K22F120M	PTE0 (J30)	PTE1 (J29)	UART1
TWR-K64F120M	PTC4 (J15)	PTC3 (J10)	UART1
TWR-KV31F120M	PTB17 (J23)	PTB16 (J22)	UART0

11.5.1 Customization Options

The USE_STDIO_FUNCTIONS definition determines whether the demo uses standard I/O functions, such as printf. If it is not defined, then the demo accesses the UART driver directly.

Chapter 12
Hardware Timer Demo

This demo application demonstrates using the hardware timer driver.

12.1 Overview

The Hardware Timer project is a demonstration program to show how to use the Hardware Timer driver. A Hardware Timer interrupt is created and fires multiple times until it reaches the requested number.

12.2 Supported Platforms

These Freescale Freedom development platforms and Tower System modules are supported by the Kinetis SDK Hardware Timer demo.

- FRDM-K22F
- FRDM-K64F
- TWR-K22F120M
- TWR-K64F120M
- TWR-KV31F120M
- TWR-KV10Z32

12.3 Getting Started

12.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with the following settings:
    - 115200 baud rate
    - 8 data bits
    - No parity
    - One stop bit
    - No flow control
3. Download the program to the target board.
4. Either press the reset button on the board or launch the debugger in the IDE to begin running the demo.

For detailed instructions, see the appropriate board User's Guide.

12.3.2 Run the demo

1. Press the reset button on your board.

2. "Hwtimer Example" message is displayed on the terminal.
3. A dot is printed when an Hwtimer interrupt occurs until the HWTIMER_DOTS_PER_LINE ∗ HWTIMER_LINES_COUNT (defined in hwtimer_demo.c) interrupts occur.
4. Finally, the "End" message is displayed.

```
Hwtimer Example
....................................
....................................
End
```

12.4 Customization Options

This demo application is customizable to show different types of hardware timers.

12.4.1 Configure the Hardware Timer Used

Determine which timer the hardware timer driver uses. The ARM core Systick timer is used by default.

```
#define HWTIMER_LL_DEVIF    kSystickDevif
```

12.4.2 Configure which clock is used by the hardware timer

Determine which clock source is used by the hardware timer.

```
#define HWTIMER_LL_SRCCLK   kCoreClock
```

12.4.3 Configure which instance of the module is used

Determine which instance of the selected hardware module to use. For the Systick timer only ’0’ is valid. If the PIT is used, use this to select the PIT channel.

```
#define HWTIMER_LL_ID       0
```

12.4.4 Hardware Timer Period

Determine the timer period (in microseconds).

```
#define HWTIMER_PERIOD          100000
```

Chapter 13
I2C Communication Demo

This demo application demonstrates the I2C demo.

13.1 Overview

The I2C communication application demonstrates I2C data communication between two boards. It also features low power wakeup of the slave board by using I2C address matching.

First, the I2C slave board enters the low power wait mode. An LED on the I2C slave board is on to indicate that the MCU is in sleep mode and no code is running.

Then, the I2C slave board is woken up by the I2C address matching interrupt when the I2C master boards sends the proper address. The LED on the I2C slave board is toggled during the data communication.

After power on, the I2C master starts reading data from the I2C slave data buffer. The I2C slave has "sub" addresses to access a specific byte of data on the slave board. The master prints this data out via the serial terminal. The master can then modify the data at a specific "sub" address on the slave board. When the data is received, the I2C slave changes the content at that requested "sub" address. This change is reflected when the master reads the slave data buffer again.

13.2 Supported Platforms

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK I2C Communication demo.

- FRDM-K22F
- FRDM-K64F
- TWR-K22F120M
- TWR-K64F120M
- TWR-KV31F120M
- TWR-KV10Z32

13.3 Getting Started

13.3.1 Hardware configuration

This demo requires two separate boards. Make these connections between the two boards by using external wires:

FRDM-K22F:

Master Board		Connects To	Slave Board	
Pin Name	Board Location		Pin Name	Board Location
PTB2/I2C0_SCL	J24 Pin 12	->	PTB2/I2C0_SCL	J24 Pin 12
PTB3/I2C0_SDA	J24 Pin 10	->	PTB3/I2C0_SDA	J24 Pin 10
GND	J2 Pin 14	->	GND	J2 Pin 14

FRDM-K64F:

Master Board		Connects To	Slave Board	
Pin Name	Board Location		Pin Name	Board Location
PTE24/I2C0_SCL	J2 Pin 20	->	PTE24/I2C0_SCL	J2 Pin 20
PTE25/I2C0_SDA	J2 Pin 18	->	PTE25/I2C0_SDA	J2 Pin 18
GND	J2 Pin 14	->	GND	J2 Pin 14

TWR-K22F120M & TWR-KV31F120M:

Master Board		Connects To	Slave Board	
Pin Name	Board Location		Pin Name	Board Location
PTE24/I2C0_SCL	Primary Elevator A7	->	PTE24/I2C0_SCL	Primary Elevator A7
PTE25/I2C0_SDA	Primary Elevator A8	->	PTE25/I2C0_SDA	Primary Elevator A8
GND	Primary Elevator A6	->	GND	Primary Elevator A6

TWR-K64F120M:

Master Board		Connects To	Slave Board	
Pin Name	Board Location		Pin Name	Board Location
PTC10/I2C1_SCL	Primary Elevator A75	->	PTC10/I2C1_SCL	Primary Elevator A75
PTC11/I2C1_SDA	Primary Elevator A60	->	PTC11/I2C1_SDA	Primary Elevator A60
GND	Primary Elevator A65	->	GND	Primary Elevator A65

TWR-KV10Z32:

Master Board		Connects To	Slave Board	
Pin Name	Board Location		Pin Name	Board Location
PTC6/I2C0_SCL	Primary Elevator A7	->	PTC6/I2C0_SCL	Primary Elevator A7
PTC7/I2C0_SDA	Primary Elevator A8	->	PTC7/I2C0_SDA	Primary Elevator A8
GND	Primary Elevator A81	->	GND	Primary Elevator A81

13.3.2 Terminal configuration

Configure the PC host serial console as shown:

- 115200 baud rate
- 8 data bits
- No parity
- One stop bit
- No flow control

13.3.3 Run the demo

1. Connect the I2C slave board to the master board using the connections listed above.
2. Power on the I2C slave board.
3. Download and run the i2c_comm_slave project to the I2C slave board.
4. The terminal of the I2C slave board prints out a "====== I2C Slave ======" message.
5. Power on the I2C master board.
6. Download and run the i2c_comm_mstr project to the I2C master board.
7. The terminal of the I2C master board prints out a "====== I2C Master ======" message and the data received from the I2C slave.
8. The I2C slave project creates some "sub" addresses to access a specific byte of data on the slave board. The master reads all these "sub" addresses and prints out the data.

Slave Sub Address	**Character**
[0]	I
[1]	2
[2]	C
[3]	-
[4]	C
[5]	O

[6]	M
[7]	M

9. To change the I2C slave sub address content, input a new character in the I2C master command line:

```
Input slave sub address and the new character.
Slave Sub Address: 5
Input New Character: F
```

10. The master then displays the updated content on the terminal output.

Slave Sub Address	Character
[0]	I
[1]	2
[2]	C
[3]	-
[4]	C
[5]	F
[6]	M
[7]	M

Chapter 14
I2C Demo with RTOS

This demo application demonstrates the I2C demo on different RTOS.

14.1 Overview

This I2C application demonstrates the SDK Peripheral drivers working on different RTOSes. The application acts as both the I2C master and the slave device on different I2C buses, such as the I2C Master on the I2C0 bus and the I2C Slave on the I2C1 bus. It can run on a single board or on two different boards. When connecting the two I2C buses on one board, the master sends the command using the I2C0 bus to the slave using the I2C1 bus. When connecting the I2C0 bus to the I2C1 bus on the other board, the application running on the first board is a master and sends a command to the other board which acts as a slave. This means that the first board can send a command and get a response from the other board by using the I2C bus. The basic purpose of this demo is:

1. Read the Kinetis chip UID (low 32bits) from the slave board
2. Read the Kinetis chip internal temperature from the slave board
3. Control the RED/GREEN/BLUE color LEDs on the slave board

The application creates three different tasks to handle events concurrently:

1. Master task: responds to the user interface interaction, runs as a I2C master, and acts as a simple UI. It accepts user's commands to read the basic chip UID, chip temperature and control the on board LED, and power mode on the slave.
2. Slave task: responds to the command received from the I2C master and returns the result to the master.
3. ADC sample task: responds to getting the chip temperature in a period.
4. For the bare metal version, the master and slave tasks are separated into two separate projects.

14.2 Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

14.3 Supported Platforms

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK I2C demo with RTOS.

- FRDM-K22F
- FRDM-K64F

- TWR-K22F120M
- TWR-K64F120M

14.4 Getting Started

The I2C RTOS application is designed to work on one single board or two different boards. Note that the bare-metal version only supports two boards.

14.4.1 Build with different RTOS support

Before running this application, build it with the RTOS you want to use. The projects for different RTOSes are differentiated by the workspace file name in the format of i2c_rtos_<rtos>.eww For example, in IAR, the i2c_rtos_ucosii.eww workspace file is the µC/OS-II version of this application. After opening the appropriate workspace, build the ksdk_<rtos>_lib project and build the application project. A binary named i2c_rtos_<rtos>.out is generated.

14.4.2 Hardware configuration

Make the connections between the listed signals by using the external wires.

Freescale Freedom FRDM-K22F

FRDM-K22F Single Board				
Master		Connects To	Slave	
Pin Name	Board Location		Pin Name	Board Location
PTB2/I2C0_SCL	J24 - Pin 12	->	PTC10/I2C1_SCL	J1 - Pin 13
PTB3/I2C0_SDA	J24 - Pin 10	->	PTC11/I2C1_SDA	J2 - Pin 7

FRDM-K22F Two Boards				
Master (Board #1)		Connects To	Slave (Board #2)	
Pin Name	Board Location		Pin Name	Board Location
PTB2/I2C0_SCL	J24 - Pin 12	->	PTC10/I2C1_SCL	J1 - Pin 13
PTB3/I2C0_SDA	J24 - Pin 10	->	PTC11/I2C1_SDA	J2 - Pin 7
GND	TP21	->	GND	TP21

Freescale Freedom FRDM-K64F

FRDM-K64F Single Board

Master		Connects To	Slave	
Pin Name	Board Location		Pin Name	Board Location
PTE24/I2C0_SCL	J2 - Pin 20	->	PTC10/I2C1_SCL	J4 - Pin 12
PTE25/I2C0_SDA	J2 - Pin 18	->	PTC11/I2C1_SDA	J4 - Pin 10

FRDM-K64F Two Boards				
Master (Board #1)		Connects To	Slave (Board #2)	
Pin Name	Board Location		Pin Name	Board Location
PTE24/I2C0_SCL	J2 - Pin 20	->	PTC10/I2C1_SCL	J4 - Pin 12
PTE25/I2C0_SDA	J2 - Pin 18	->	PTC11/I2C1_SDA	J4 - Pin 10
GND	J2 - Pin 14	->	GND	J2 - Pin 14

TWR-K22F120M Tower System module

TWR-K22F120M Single Board				
Master		Connects To	Slave	
Pin Name	Board Location		Pin Name	Board Location
PTE24/I2C0_SCL	Primary Elevator - Pin A7	->	PTC10/I2C1_SCL	Primary Elevator - Pin B50
PTE25/I2C0_SDA	Primary Elevator - Pin A8	->	PTC11/I2C1_SDA	Primary Elevator - Pin B51

TWR-K22F120M Two Boards				
Master (Board #1)		Connects To	Slave (Board #2)	
Pin Name	Board Location		Pin Name	Board Location
PTE24/I2C0_SCL	Primary Elevator - Pin A7	->	PTC10/I2C1_SCL	Primary Elevator - Pin B50
PTE25/I2C0_SDA	Primary Elevator - Pin A8	->	PTC11/I2C1_SDA	Primary Elevator - Pin B51
GND	Primary Elevator - Pin A65	->	GND	Primary Elevator - Pin A65

TWR-K64F120M Tower System module

TWR-K64F120M Single Board				
Master		Connects To	Slave	
Pin Name	Board Location		Pin Name	Board Location

PTD8/I2C0_SCL	Primary Elevator - Pin A7	->	PTC10/I2C1_SCL	Primary Elevator - Pin A75
PTD9/I2C0_SDA	Primary Elevator - Pin A8	->	PTC11/I2C1_SDA	Primary Elevator - Pin B71

TWR-K64F120M Two Boards				
Master (Board #1)		Connects To	Slave (Board #2)	
Pin Name	Board Location		Pin Name	Board Location
PTD8/I2C0_SCL	Primary Elevator - Pin A7	->	PTC10/I2C1_SCL	Primary Elevator - Pin A75
PTD9/I2C0_SDA	Primary Elevator - Pin A8	->	PTC11/I2C1_SDA	Primary Elevator - Pin B71
GND	Primary Elevator - Pin A65	->	GND	Primary Elevator - Pin A65

14.4.3 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

14.5 Run the demo

This menu displays in the terminal window:

```
Available Commands:
LED Red Toggle (1)    - Red Light toggles on/off
LED Green Toggle (2)  - Green Light toggles on/off
LED Blue Toggle (3)   - Blue Light toggles on/off
Read Temperature (4)   - Get temperature of client
Read Id     (5)    - Read client unique ID
```

Enter your choice (1 - 5):

You can select to toggle the RGB LED, read the temperature of the client board, and read the client unique ID.

Note that a different colored LED may turn on if the selected color is not available on that board.

Chapter 15
LPTMR Demo

This demo application demonstrates the LPTMR demo.

15.1 Overview

The LPTMR (Low Power Timer) project is a simple demonstration program to show how to use the LPTMR driver. It triggers an LPTMR interrupt once every second, and prints out the number of interrupts that have occurred since the program started running.

15.2 Supported Platforms

This demo supports the following Freescale Freedom development platforms and Tower System modules:

- FRDM-K22F
- FRDM-K64F
- TWR-K22F120M
- TWR-K64F120M
- TWR-KV31F120M
- FRDM-KL03Z
- TWR-KV10Z32

15.3 Getting Started

15.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with the following settings:
   - 115200 baud rate (9600 for FRDM-KL03Z)
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

15.4 Run the demo

An LPTMR interrupt occurs every second and prints out to the serial terminal the number of interrupts that have occurred since the program started running. The LPTMR module uses the internal 1 kHz Low Power Oscillator (LPO) as its clock source for this demo.

Run the demo

The output on the serial terminal is as shown:

```
Low Power Timer Example
Started LPTMR
1 2 3 4 5 6 7
```

Chapter 16
HTTP Server Demo on lwIP TCP/IP Stack

This demo application demonstrates the HTTPServer demo on lwIP TCP/IP stack with bare metal SDK or different RTOSes.

16.1 Overview

This is an HTTPServer set up on lwIP TCP/IP stack with bare metal SDK or different RTOSes. The user uses an Internet browser to send a request for connection. The board acts as an HTTP server and sends a Web page back to the PC.

16.2 Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

16.3 Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the Kinetis software development kit HTTPServer demo.

- FRDM-K64F
- TWR-K64F120M

16.4 Getting Started

See the *lwIP TCPIP Stack and Kinetis SDK Integration User's Guide* (document KSDKLWIPUG) for more information about the setup and requirements.

16.4.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with the following settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.

4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions steps, see a Kinetis SDK User's Guide for your board.

16.4.2 Network Configuration

Configure the IP address of PC network adapters as shown: IP address - 192.168.2.100 Subnet Mask - 255.255.255.0

16.4.3 Run the demo

1. Download the program to target board, which should be installed in TWR or FRDM.
2. Connect the Ethernet cable between the PC and the board.
3. When successfully connected, reset the board to run the demo.
4. Open the PC command window, type in "ping 192.168.2.102" to test whether lwIP stack is running. If successful,
   four echo request packets are successfully replied.
5. Input "192.168.2.102" in the URL of an Internet browser on a PC. If successful, the web page the board returns opens in the browser.

Chapter 17
Ping Demo on lwIP TCP/IP Stack

This demo application demonstrates the Ping demo on lwIP TCP/IP stack with bare metal SDK or different RTOSes.

17.1 Overview

This is a Ping Demo on the lwIP TCP/IP stack which uses the ICMP protocol. The application on board periodically sends the ICMP echo request to a PC and processes the PC reply. Type the "ping $board_-address" in the PC command window to send an ICMP echo request to the board. The lwIP stack sends the ICMP echo reply back to the PC.

17.2 Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

17.3 Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK Ping demo.

- FRDM-K64F
- TWR-K64F120M

17.4 Getting Started

See the *lwIP TCPIP Stack and Kinetis SDK Integration User's Guide* (document KSDKLWIPUG) for instructions and requirements.

17.4.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control

3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

17.4.2 Network Configuration

Configure the IP address of PC network adapters as shown:

- 192.168.2.100

17.5 Run the demo

1. Download the program to the target board.
2. Connect the Ethernet cable between the PC and the board.
3. When successfully connected, reset the board to run the demo.
4. Open the terminal. Ping send and ping receive are successful.
5. Type in "ping 192.168.2.102" in PC command window. If the operation is successful, four packets are
successful replied.

Chapter 18
TCP Echo Demo on lwIP TCP/IP Stack

This demo application demonstrates the TCP Echo demo on lwIP TCP/IP stack with bare metal SDK or different RTOSes.

18.1 Overview

This is a TCP echo demo on the lwIP TCP/IP stack with bare metal SDK or different RTOSes, which uses the TCP protocol and acts as an echo server. The application on board sends back the TCP packets from the PC, which can be used to test whether the TCP connection is available.

18.2 Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

18.3 Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK TCP Echo demo.

- FRDM-K64F
- TWR-K64F120M

18.4 Getting Started

See the *lwIP TCPIP Stack and Kinetis SDK Integration User's Guide* (document KSDKLWIPUG) for instructions and requirements.

18.4.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.

4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

18.4.2 Network Configuration

Configure the IP address of PC network adapters as shown:

- 192.168.2.100

18.5 Run the demo

1. Download the program to the target board.
2. Connect the Ethernet cable between the PC and the board.
3. When successfully connected, reset the board to run the demo.
4. Open the command window on PC, type in "ping 192.168.2.102" to test whether the LwIP is running.
5. If it is running, use an external echo tool to perform the echo request. This tool sends TCP packets to the board and checks whether the content sent back from board is the same. A similar tool named "echotool" can be downloaded from the: http://bansky.net/echotool/ [example: echotool 192.168.2.102 /p tcp /r 7 /d hello]
6. If the operation is successful, all packets sent back are same as the packets sent to the board.

Chapter 19
UDP Echo Demo on lwIP TCP/IP Stack

This demo application demonstrates the UDP Echo demo on lwIP TCP/IP stack with bare metal SDK or different RTOSes.

19.1 Overview

This is a UDP echo demo on the lwIP TCP/IP stack with bare metal SDK or different RTOSes, which uses the UDP protocol and acts as an echo server. The application on board sends back the UDP packets from the PC, which can be used to test whether the UDP connection is available.

19.2 Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

19.3 Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK UDP Echo demo.

- FRDM-K64F
- TWR-K64F120M

19.4 Getting Started

See the *lwIP TCPIP Stack and Kinetis SDK Integration User's Guide* (document KSDKLWIPUG) for instructions and requirements.

19.4.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.

4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

19.4.2 Network Configuration

Configure the IP address of PC network adapters as shown:

- 192.168.2.100

19.5 Run the demo

1. Download the program to the target board.
2. Connect the Ethernet cable between the PC and the board.
3. When successfully connected, reset the board to run the demo.
4. Open the command window on PC, type in "ping 192.168.2.102" to test whether the lwIP is running.
5. If it is running, use an external echo tool to perform the echo request. This tool sends UDP packets to the board and checks whether the content sent back from board is the same. A similar tool named "echotool" can be downloaded from the: http://bansky.net/echotool/ [example: echotool 192.168.2.102 /p udp /r 7 /d hello]
6. If the operation is successful, all packets sent back are the same as the packets sent to the board.

Chapter 20
MMDVSQ Demo

This demo application demonstrates how to use MMDVSQ driver.

20.1 Overview

The MMDVSQ Demo project is a simple demonstration program to show how to use the MMDVSQ driver. This demo shows the efficiency of MMDVSQ divide and squre root operation and normal C function .

20.2 Supported Platforms

This demo supports the following Freescale Freedom development platforms and Tower System modules:

- TWR-KV10Z75M

20.3 Getting Started

20.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with the following settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

20.4 Run the demo

The output on the serial terminal is as shown:

```
MMDVSQ Demo start!
C library calculation takes 622 tickcycles
MMDVSQ t calculation takes 521 tickcycles
MMDVSQ Demo end
```

The tickcycles just as reference.

Chapter 21
RTC Function Demo

This demo application demonstrates how to use the RTC driver.

21.1 Overview

This RTC demo application demonstrates the important features of the RTC Module by using the RTC Periperhal Driver. It supports these features:

- Calendar
  - Get the current date time with Year, Month, Day, Hour, Minute and Second.
  - Set the current date time with Year, Month, Day, Hour, Minute and Second.
- Alarm
  - Set the alarm based on the current time.
  - Application prints a notification when the alarm expires.
- Seconds interrupt
  - Use second interrupt function to display a digital time blink every second.
- Compensation
  - Configure the compensation with cycles.
  - The 1 Hz RTC clock with compensation configured is output to a pin. Use an oscilloscope to check the compensation result.

21.2 Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK RTC Function demo.

- FRDM-K22F
- FRDM-K64F
- TWR-K22F120M
- TWR-K64F120M

21.3 Getting Started

21.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with the following settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control

3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

21.4 Run the demo

This menu is displayed on the serial terminal:

```
Please choose the sub demo to run:
1) Get current date time.
2) Set current date time.
3) Alarm trigger show.
4) Second interrupt show (demo for 20s).
5) Set RTC compensation.

Select:
```

Chapter 22
SAI Demo

This demo application demonstrates how to use the SAI drivers.

22.1 Overview

The SAI Demo project is a digital audio demonstration program that uses the KSDK software. It performs audio playback from either a .wav file, stored in Flash, or from the line-in on a TWR-AUDIO-SGTL Tower System module using the KSDK I2S and I2C drivers. On the TWR-K22F120M, TWR-K24F120M and the TWR-K64F120M Tower System modules, the project also uses the CMSIS-DSP library to perform a Fast Fourier Transform, and return the fundamental frequency of the line-in audio.

22.2 Supported Hardware

This demo supports the following Freescale Freedom development platforms and Tower System modules:

- TWR-K22F120M
- TWR-K22F120M128
- TWR-K22F120M256
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M

22.3 Getting Started

22.3.1 GCC Compiler notes

When building the demo with GCC, ensure that the demo and platform library are built with this option:

```
CHOOSE_FLOAT=HARD_FP
```

Otherwise, the project does not use the Kinetis device's hardware floating point when using the CMSIS--DSP library.

22.3.2 Hardware Settings

These Tower System modules are required to run the sai_demo:

- TWR-ELEV
- TWR-AUDIO-SGTL (except TWR-K24F120M which has a built-in one)

22.3.3 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with these settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

22.4 Run the demo

To hear the audio playback, connect a set of headphones to the headphone output on the TWR-AUDIO--SGTL card. For input to the codec, connect an audio source to the Line-In on the TWR-AUDIO-SGTL.

When the demo starts, this message is displayed in the terminal output window:

```
Audio Demo!

Press spacebar to start demo.

Demo begin...
```

The user can either play back audio from the line-in source, or play a .wav file stored in the Flash.

The line-in option plays the audio gathered from the codec line-in for approximately 15 seconds.

```
Select player:
        1. Line-In Playback
        2. Wav File Playback
->1
```

If selecting playback from the line-in source, decide whether to perform an FFT analysis to find the fundamental frequency of the audio input. Finding the fundamental frequency is best suited for pure tones played into the line-in of the TWR-AUDIO-SGTL card.

```
Select filter:
        1. FFT - Find Fundamental Frequency
        2. None
->1
```

The user is prompted to select from a list of headphone output levels:

```
Choose headphone dB level:
        1. +3.0 dB
        2. 0.0 dB
        3. -3.0 dB
        4. -6.0 dB
        5. -12.0 dB
        6. -24.0 dB
        7. -48.0 dB
->5
Frequency is 93 Hz
```

The table shows the terminal display after playback has completed and the FFT option was selected.

These are the options for the .wav file option:

```
Select player:
        1. Line-In Playback
        2. Wav File Playback
->2
Select Wav file:
        1. Audio Demo
->1
Choose headphone dB level:
        1. +3.0 dB
        2. 0.0 dB
        3. -3.0 dB
        4. -6.0 dB
        5. -12.0 dB
        6. -24.0 dB
        7. -48.0 dB
->5
```

The quality of the .wav file PCM data depends on the demo system and the compiler.

The table below shows the audio sample rate, channels and bit depth of the .wav file for the various platforms and compilers.

Hardware System	Sample Rate (kHz)				Bit Depth				Channels			
	IAR	ARM	GNU-GCC	KDS-GCC	IAR	ARM	GNU-GCC	KDS-GCC	IAR	ARM	GNU-GCC	KDS-GCC
TWR--K22-F120M	44.1	44.1	11.-025	11.-025	16	16	16	16	2	2	2	2

TWR--K22-F120-M128	11.-025	11.-025	11.-025	11.-025	16	16	16	16	2	2	1	1
TWR--K22-F120-M256	11.-025	11.-025	11.-025	11.-025	16	16	16	16	2	2	1	1
TWR--K24-F120-M	44.1	44.1	44.1	44.1	32	32	32	32	2	2	2	2
TWR--K60-F100-M	44.1	44.1	44.1	44.1	32	32	32	32	2	2	2	2
TWR--K64-F120-M	44.1	44.1	44.1	44.1	32	32	32	32	2	2	2	2

Quality differences of the .wav playback depend on the size constraints of the target device, the Flash size, and the density of the code generated by the compiler.

Note that all supported platforms play audio from the line-in option with the same quality: 16-bit, 44.1 kHz, 2 channels.

22.5 Key Functions

void audio_stream_init(void)

Initializes the I2S, I2C, and TWR-AUDIO-SGTL Tower System module for streaming audio from Line-In.

void audio_wav_init(wave_file_t *newWav)

Initializes the I2S, I2C, and TWR-AUDIO-SGTL Tower System module for playing back WAV file in Flash.

Parameters

newWav	Pointer to wave file data structure.

uint32_t config_volume(sgtl_handler_t *handler, sgtl_module_t module, uint32_t volumeCtrl)

Sets volume from the user input.

Parameters

handler	pointer to codec handler structure.
module	name of module on codec to set the volume for.
volumeCtrl	user input data from terminal menu.

Returns

status_t Return kStatus_Success if function completed successfully, return kStatusFail if function failed.

snd_status_t stream_audio(dsp_types_t dspType, uint8_t volumeCtrl)

Plays a stream of audio.

Parameters

dspType	Used to select one DSP function to perform on the data.
volumeCtrl	Value used to set decibel level on codec.

Returns

Returns soundcard status

snd_status_t get_wav_data(wave_file_t *waveFile)

Collects data from WAV file header.

Parameters

waveFile	Data structure of pcm data array.

Returns

status_t Return kStatus_Success if function completed successfully, return kStatusFail if function failed.

snd_status_t play_wav(uint32_t *pcmBuffer, uint8_t volumeCtrl)

Plays the PCM audio data from the WAV format array.

Parameters

pcmBuffer	Pointer to data array containing WAV formatted audio data.
volumeCtrl	Value used to set decibel level on codec.

Returns

status_t Return kStatus_Success if function completed successfully, return kStatusFail if function failed.

void send_wav(uint8_t *dataBuffer, uint32_t length, sai_data_format_t *dataFormat)

Sends audio data to the sound card.

Parameters

pdataBuffer	Pointer to data array containing WAV formatted audio data.
length	length of WAV file to send.
dataFormat	Point to audio_data_format_t for sound card.

float32_t do_fft(sai_data_format_t *dataFormat, uint8_t *buffer, float32_t *fftData, float32_t *fftResult)

Performs frequency analysis and finds fundamental frequency of the PCM data.

Parameters

dataFormat	Pointer to audio data format structure.
buffer	Pointer to data array to store modulated PCM data.
fftData	Pointer to data array for storing Fast Fourier Transform data.
fftResult	Point to data array for storing real frequency bins from FFT.

Returns

float32_t Returns fundamental frequency in Hz.

Chapter 23
SD Card Demo

This demo application demonstrates the SD Card demo.

23.1 Overview

The SD Card demo application demonstrates the use of SD card driver. It displays the card information followed by a write-read compare test and the erase operation.

23.2 Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK SD Card demo.

- FRDM-K64F
- TWR-K64F120M

23.3 Getting Started

Hardware configuration

There is no specific hardware requirement. The default configuration of the supported targets is sufficient for this demo. The demo uses the on-board connector support for the card detect signal which is connected to a GPIO line for card detection.

23.3.1 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with the following settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Either press the reset button on your board or launch the debugger in your IDE to begin running the demo

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

23.4 Run the demo

1. Insert an SD or a micro-SD card depending on the connector on board. Ensure that the card doesn't contain any important content because the demo will erase and overwrite some sectors.
2. After the card detection, the card-specific information, such as capacity, is shown. Then, the user is encouraged to back up data as needed. A write-read-compare access is performed to demonstrate the use case.
3. If the card was not inserted as mentioned in step 1, the demo waits for the card insertion. Once a card is inserted, it auto-detects and proceeds as shown in step 3.

Chapter 24
Thermistor Lab Demo

This demo application demonstrates how to use PDB to trigger ADC to measure on-board thermistor.

24.1 Overview

This lab will help you to understand how to configure and use ADC module to sample differential voltage across on-board thermistors RT1-RT4. If a user touch any on-board thermistor by a finger the lab application detects a change in the thermistor temperature and start flashing the corresponding LED pair.

- The lab tutorial demonstrates:
  - how to configure ADC module to read differential inputs
  - how to filter and process ADC results
  - how to use FreeMASTER visualization tool to display sampled results.

24.2 Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK RTC Function demo.

-TWR-KV10Z32

24.3 Getting Started

24.3.1 Prepare the Demo

1. Short pin 1 & 2 on J11 to J14.
2. Short pin2 and 3 on J8.
3. Download the program to the target board.
4. Touch 4 on-board thermistor to see LED change.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

24.3.2 Demo Code Overview

The lab application configures both ADCs to be triggered by FlexTimer0 vis PDB. The FlexTimer is configured to generate 16KHZ PWM and the channel1 trigger is used to trigger both ADCs via PDB. The PDB is configured to generate four delayed trigger signals to both ADCs per FlexTimer0 Channel 1 trigger and as a results there will be 4 ADCs sample converted per each Flextimer channel trigger. The ADC is configure to be 16-bit differential mode, and in ping-pong mode.

When an ADC conversion is complete an interrupt is generated by ADC module and an interrupt service routine is executed. The interrupt service routine ADCn_ISR() call ADCn_Task which executes following

tasks: read ADC results registers. filter ADC results with low-pass FIR filter. differentiate filtered results to detect a change in a voltage across the thermistor. detect a negative/positive slope of a voltage change to determine which LED will be turned on/off. executes a software timer, whose time out period is 100ms and it resets every 400ms. the software timer is used to generate a time base for LEDs flashing.

24.3.2.1 ADC Differential Mode of Operation

To measure a voltage across the thermistor it is beneficial to configure ADC for a differential mode of operation. In differential mode the ADC measures a difference between two analogy inputs. The ADC enables to select an inputs pairs, which will be treated as differential inputs. The ADC enable to select an input pairs, which will be treated as differential inputs.

Detection of a Change of Thermistor Voltage

If a person places a finger on a thermistor its temperature will rise. The temperature rise will result in a decrease of a voltage across the resistor. If the finger is removed then the temperature will decrease and voltage goes up.

To Detect a change in a voltage a simple differentiators are used. The filtered thermistor voltage are stored in a buffer. Buffer size is defined by BUFF_SIZE. The differentiator calculates a difference between an actual voltage sample and a sample delayed by i_delay pointer, which points into the buffer.

```
delta_rt1 = rt1_filt -rt1_filt_buff[i_delay];
```

If the voltage across the thermistor decreases the differentiator returns a negative value. If the voltage increases the differentiator returns a positive value. If there is no change in a voltage the deviator output returns zero. The bigger the slope of voltage increase/decrease the more positive/negative value the differentiator returns. The lab application uses this information to detect if the finger is placed on the particular thermistor or if the finger was removed.

Placing/removing a finger on the thermistor is characterized by a certain slope (rate) of voltage decrease/increase. The application defines positive and negative thresholds for each thermistor.

If a difference output exceeds threshold limits(for at least three consequent samples) then an action is taken and a corresponding LED starts to flash.

Chapter 25
Hello World Demo on the uIP TCP/IP stack

This demo application demonstrates how to write uIP application using protosockets.

25.1 Overview

Modules

- Hello World Demo on the uIP TCP/IP stack Introduction
  This part provides an introduction to how to write uIP application using protosockets.
- Hello World Demo on the uIP TCP/IP stack User's Guide
  This part provides a user's guide to how to write uIP application using protosockets.

25.2 Hello World Demo on the uIP TCP/IP stack Introduction

This chapter provides an introduction to how to write uIP application using protosockets.

25.2.0.2 Hello World Demo on uIP TCP/IP Stack Introduction

Overview

The Hello World project is a simple demonstration program that uses the uIP protosockets handling the application communication. The application is waiting for any TCP connection and responses to the connection by sending out the message "Hello. What is your name?". the application will then send out the message"Hello XXXXX" after receiving the response "XXXXX" from the connection. The purpose of this demo is to show how to write uIP application using protosockets.

Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the Kinetis software development kit uIP Hello World demo.

- TWR-K64F120M
- FRDM-K64F

25.3 Hello World Demo on the uIP TCP/IP stack User's Guide

This chapter provides a user's guide to how to write uIP application using protosockets.

25.3.0.3 Hello World Demo on uIP TCP/IP Stack User's Guide

Getting started

See the < uIP TCPIP Stack and Kinetis SDK Integration User's Guide > for more information about the setup and requirements.

Terminal Configuration

Configure the IP address of PC network adapters as:

- 192.168.0.x
- 255.255.255.0

The debug serial terminal is configured for these settings:

- 115200
- 8 data bits
- No parity
- 1 stop bit

Run the demo

1. Download the program to target board, which should be installed in TWR or FRDM.
2. Run the demo.
3. Connect the Ethernet cable between PC and board.
4. Open the telnet terminal Command line with IP 192.168.0.10 on PC. when connected, you will see the message "Hello. What is your name?" on the terminal command line. please enter the message you want to reply to the message and press the enter button. for example, enter the "world" message and press the enter button. you will then see the message "Hello world" from the target board.

Chapter 26
HttpClient Demo on the uIP TCP/IP stack

This demo application demonstrates the HttpClient Demo on the uIP TCP/IP stack.

26.1 Overview

Modules

- HttpClient Demo on the uIP TCP/IP stack Introduction
  This part provides an introduction to HttpClient Demo on the uIP TCP/IP stack Introduction.
- HttpClient Demo on the uIP TCP/IP stack User's Guide
  This part provides a user's guide to HttpClient Demo on the uIP TCP/IP stack Introduction.

26.2 HttpClient Demo on the uIP TCP/IP stack Introduction

This chapter provides an introduction to HttpClient Demo on the uIP TCP/IP stack Introduction.

26.2.0.4 HttpClient Demo on uIP TCP/IP Stack Introduction

Overview

The HttpClient project is a simplistic implementation of an HTTP Client. This demo shows a HTTP Client that is able to download web pages and files from web servers.

Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the Kinetis software development kit HttpClient Demo.

- TWR-K64F120M
- FRDM-K64F

26.3 HttpClient Demo on the uIP TCP/IP stack User's Guide

This chapter provides a user's guide to HttpClient Demo on the uIP TCP/IP stack Introduction.

26.3.0.5 HttpClient Demo on uIP TCP/IP Stack User's Guide

Getting started

See the < uIP TCPIP Stack and Kinetis SDK Integration User's Guide > for more information about the setup and requirements.

Terminal Configuration

The debug serial terminal is configured for these settings:

- 115200
- 8 data bits
- No parity
- 1 stop bit

Run the demo

1. Download the program to target board, which should be installed in TWR or FRDM.
2. Connect the board to the LAN and open the serial terminal.
3. Run the demo.
4. you will see the similar log on the serial terminal as below: first, successfully get the DHCP IP and do the DNS resolve:
   enet device init success! uIP log message: udp: no matching connection found uIP log message: udp: no matching connection found uIP log message: udp: no matching connection found Got IP address 10.192.242.47 Got netmask 255.255.255.0 Got DNS server 10.192.130.201 Got default router 10.192.242.254 Lease expires in 172800 seconds Found name 'www.freescale.com' = 121.205.161.96

```
second, sucessfully get the files from the target website(in normal, the file download can b
finished in less than two seconds, but sometimes, due to busy network and the uIP TCP connec

Webclient: connected, waiting for data...
Webclient: got 217 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
```

```
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 732 bytes of data.
Webclient: got 166 bytes of data.
Webclient: got 0 bytes of data.
Webclient: got 0 bytes of data.
```

Chapter 27
HttpServer Demo on the uIP TCP/IP stack

This demo application demonstrates the HttpServer Demo on the uIP TCP/IP stack.

27.1 Overview

Modules

- HttpServer Demo on the uIP TCP/IP stack Introduction
  This part provides an introduction to HttpServer Demo on the uIP TCP/IP stack Introduction.
- HttpServer Demo on the uIP TCP/IP stack User's Guide
  This part provides a user's guide to HttpServer Demo on the uIP TCP/IP stack Introduction.

27.2 HttpServer Demo on the uIP TCP/IP stack Introduction

This chapter provides an introduction to HttpServer Demo on the uIP TCP/IP stack Introduction.

27.2.0.6 HttpServer Demo on uIP TCP/IP Stack Introduction

Overview

The HttpServer project is a simplistic implementation of an HTTP Server. It can server web pages and files from a read-only ROM file system and provides a very small scripting language.

Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the Kinetis software development kit HttpServer Demo.

- TWR-K64F120M
- FRDM-K64F

27.3 HttpServer Demo on the uIP TCP/IP stack User's Guide

This chapter provides a user's guide to HttpServer Demo on the uIP TCP/IP stack Introduction.

27.3.0.7 HttpServer Demo on uIP TCP/IP Stack User's Guide

Getting started

See the < uIP TCPIP Stack and Kinetis SDK Integration User's Guide > for more information about the setup and requirements.

Terminal Configuration

Configure the IP address of PC network adapters as:

- 192.168.0.x
- 255.255.255.0

The debug serial terminal is configured for these settings:

- 115200
- 8 data bits
- No parity
- 1 stop bit

Run the demo

1. Download the program to target board, which should be installed in TWR or FRDM.
2. Run the demo.
3. Connect the Ethernet cable between PC and board.
4. Make sure not choose the proxy server in LAN setting on the MS Internet Explorer browser.
5. Type the IP address 192.168.0.10 to the MS Internet Explorer browser on PC. you will see the front page with header "Welcome to the uIP web server!" and with the content "These web pages are served by a small web server running on top of the uIP embedded TCP/IP stack." There are three sub-pages below for more information: "File statistics" shows the details of the file statistics. "Network statistics" shows the statistics of IP/TCP/UDP/ICMP packets. "Network connections" shows the current connections.

Chapter 28
SMTP Demo on the uIP TCP/IP stack

This demo application demonstrates the SMTP Demo on the uIP TCP/IP stack.

28.1 Overview

Modules

- SMTP Demo on the uIP TCP/IP stack Introduction
  This part provides an introduction to SMTP Demo on the uIP TCP/IP stack Introduction.
- SMTP Demo on the uIP TCP/IP stack User's Guide
  This part provides a user's guide to SMTP Demo on the uIP TCP/IP stack Introduction.

28.2 SMTP Demo on the uIP TCP/IP stack Introduction

This chapter provides an introduction to SMTP Demo on the uIP TCP/IP stack Introduction.

28.2.0.8 SMTP Demo on uIP TCP/IP Stack Introduction

Overview

The SMTP project is a simple example implementation to show how to implement protocols in uIP to send out e-mail on Internet. This demo only tests with the Freescale internal e-mail address xxx@freescale.com.

Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the Kinetis software development kit SMTP Demo.

- TWR-K64F120M
- FRDM-K64F

28.3 SMTP Demo on the uIP TCP/IP stack User's Guide

This chapter provides a user's guide to SMTP Demo on the uIP TCP/IP stack Introduction.

28.3.0.9 SMTP Demo on uIP TCP/IP Stack User's Guide

Getting started

See the < uIP TCPIP Stack and Kinetis SDK Integration User's Guide > for more information about the setup and requirements.

Terminal Configuration

The debug serial terminal is configured for these settings:

- 115200
- 8 data bits
- No parity
- 1 stop bit

Run the demo

1. Change the SMTP server address and e-mail address in below lines in smtp_app in smtp_main.c for your demo:
   uip_ipaddr(ipaddr, 10,81,153,31); -> change the 10,81,153,31 to your test SMTP server IP address smtp_configure("remotesmtp.freescale.net", ipaddr); -> change remotesmtp.freescale.net to your test SMTP server name SMTP_SEND("b43761@freescale.com", NULL, "b43761@freescale.-com", ->change the b43761@freescale.com to your test e-mail address "Testing SMTP from uIP", "Test message sent by uIP\r\n");
2. Rebuild the demo project and download the program to target board, which should be installed in TWR or FRDM.
3. Connect the board to the LAN and open the serial terminal.
4. Run the demo.
5. you will see the similar log on serial terminal as below:
   enet device init success! uIP log message: udp: no matching connection found uIP log message: udp: no matching connection found Got IP address 10.192.242.47 Got netmask 255.255.255.0 Got DNS server 10.192.130.201 Got default router 10.192.242.254 Lease expires in 172800 seconds SMTP done with code 0
   (a) Check your test e-mail and see if a new mail with content"Test message sent by uIP" and with subject "Testing SMTP from uIP" is received.

Chapter 29
Telnet Server Demo on the uIP TCP/IP stack

This demo application demonstrates the Telnet Server Demo on the uIP TCP/IP stack.

29.1 Overview

Modules

- Telnet Server Demo on the uIP TCP/IP stack Introduction
  This part provides an introduction to Telnet Server Demo on the uIP TCP/IP stack Introduction.
- Telnet Server Demo on the uIP TCP/IP stack User's Guide
  This part provides a user's guide to Telnet Server Demo on the uIP TCP/IP stack Introduction.

29.2 Telnet Server Demo on the uIP TCP/IP stack Introduction

This chapter provides an introduction to Telnet Server Demo on the uIP TCP/IP stack Introduction.

29.2.0.10 Telnet Server Demo on uIP TCP/IP Stack Introduction

Overview

The Telnet Server project is a simple demonstration program that uses the uIP telnet Server. The application provides simple shell command to get the information from the target board through telnet.

Supported RTOS

- Freescale MQX™ RTOS
- FreeRTOS
- µC/OS-II
- µC/OS-III
- Bare Metal (no RTOS)

Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the Kinetis software development kit Telnet Server Demo.

- TWR-K64F120M
- FRDM-K64F

29.3 Telnet Server Demo on the uIP TCP/IP stack User's Guide

This chapter provides a user's guide to Telnet Server Demo on the uIP TCP/IP stack Introduction.

29.3.0.11 Telnet Server Demo on uIP TCP/IP Stack User's Guide

Getting started

See the < uIP TCPIP Stack and Kinetis SDK Integration User's Guide > for more information about the setup and requirements.

Terminal Configuration

Configure the IP address of PC network adapters as:

- 192.168.0.x
- 255.255.255.0

The debug serial terminal is configured for these settings:

- 115200
- 8 data bits
- No parity
- 1 stop bit

Run the demo

1. Download the program to target board, which should be installed in TWR or FRDM.
2. Run the demo.
3. Connect the Ethernet cable between PC and board.
4. Open the telnet terminal Command line with IP 192.168.0.10 on PC. when connected, you will see the shell log:

uIP command shell Type '?' and return for help uIP 1.0>

please enter "?" for detail shell commands. if you enter "?" and press enter button, you will see the shell log:

Available commands: stats - show network statistics conn - show TCP connections help, ? - show help exit - exit shell uIP 1.0>

"conn" is used to get the established TCP connections. "stats" is used to get the network statistics. "exit" is used to exit the telnet and shell.

Chapter 30
Watchdog Timer Reset Demo

This demo application demonstrates the Watchdog Timer Reset demo.

30.1 Overview

The Watchdog Timer Reset demo application demonstrates how the Watchdog module can be used to reset a device. The overflow time for a Watchdog timer is approximately 2 seconds.

30.2 Supported Hardware

These Freescale Freedom development platforms and Tower System modules are supported by the KSDK Watchdog Timer Reset demo.

- FRDM-K22F120M
- FRDM-K22F120MK02
- FRDM-K22F120MK0264
- FRDM-K64F120M
- TWR-K22F120M
- TWR-K22F120M128R
- TWR-K22F120M256R
- TWR-K22F120MK02
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z32M
- TWR-KV31F120M
- TWR-KV31F120M128R
- TWR-KV31F120M256R
- TWR-KV31F120MKV30

30.3 Getting Started

30.3.1 Hardware configuration

These switch buttons are used by this demo:

Platform	Switch	Notes
FRDM-K22F120M(K02)	SW2	
FRDM-K64F120M	SW1	
TWR-K22F120M(128R\256R)	SW1	

TWR-K24F120M	SW1	
TWR-K60F100M	SW1	
TWR-K64F120M	SW1	
TWR-KV10Z32M	SW1	
TWR-KV31F120M(128R\256R)	SW1	Jumper J26 should be removed

Note that on the some boards, the reset source information printed out to the terminal may be incorrect and may sometimes display an "External Pin Reset" message. This occurs when the Watchdog resets the system, and the OpenSDA circuit also sends a pin reset to the target MCU through the voltage level translator. Sometimes this translator can't pull up the pin in time and the duration of the low level of this pin is long. When this occurs, the Watchdog reset status is overridden by the external pin reset and the message "External Pin Reset" is output to the terminal.

30.3.2 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.
2. Open a serial terminal with the following settings:
   - 115200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
3. Download the program to the target board.
4. Press either the reset button on your board or launch the debugger in your IDE to begin running the demo.

For more detailed instructions, see a Kinetis SDK User's Guide for your board.

30.4 Run the demo

1. When the program is running, the Watchdog is enabled. The program continuously refreshes the Watchdog to prevent the CPU reset.
2. The message: "Watchdog example running, Loop #: xx, press <SW> to start watchdog timeout..." displays on the terminal.
3. An LED also blinks. The color of the LED depends on the board:

Platform	LED Color
FRDM-K22F120M(K02)	Red
FRDM-K64F120M	Red
TWR-K22F120M(128R\256R)	Yellow
TWR-K24F120M	Red
TWR-K60F100M	Yellow

TWR-K64F120M	Yellow
TWR-KV10Z32M	Yellow
TWR-KV31F120M(128R\256R)	Red

1. When the SW button is pressed, the LED begins to blink rapidly, signifying that the Watchdog is about to expire.
2. When the Watchdog signals a reset, the "Watchdog (COP) Reset" message and "Watchdog (COP) reset count: xx" message is output to the terminal.

/*!

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Document Number: KSDKDEMOUG
Rev. 1.1.0
10/2014