
Kinetis SDK v.1.1 Demo Applications User's Guide

Freescal Semiconductor, Inc.

Document Number: KSDK11DEMOUG

Rev. 0

Dec 2014





Contents

Chapter 1 Introduction

Chapter 2 ADC Hardware Trigger Demo

2.1	Overview	3
2.1.1	Trigger by PIT	3
2.1.2	Trigger by PDB	3
2.1.3	Trigger by LPTMR	3
2.1.4	Input signal for ADC	4
2.2	Supported Platforms	4
2.3	Getting Started	5
2.3.1	Prepare the Demo	5
2.4	Run the demo	5
2.5	Customization Options	5
2.5.1	Default configurations	6
2.5.1.1	ADC configurations	6
2.5.1.2	Sample frequency	6
2.5.2	Configure the number of samples	6
2.5.3	Configure the signal frequency	6
2.5.4	Configure the ADC instance	6
2.5.5	Configure the ADC input pin	7

Chapter 3 ADC Low Power Demo

3.1	Overview	9
3.2	Supported Platforms	9

Section number	Title	Page
3.3	Getting Started	9
3.3.1	Prepare the Demo	9
3.3.2	Run the demo	10

Chapter 4 BLDC Sensorless Demo

4.1	Overview	11
4.2	Supported Platforms	11
4.3	Getting Started	11
4.3.1	Prepare the Demo	12
4.3.2	Run the demo	12

Chapter 5 DAC ADC Demo

5.1	Overview	15
5.2	Supported Platforms	15
5.3	Getting Started	15
5.3.1	Hardware Settings	15
5.3.2	Prepare the Demo	16
5.4	Run the demo	16
5.5	Key Functions	17

Chapter 6 Flash Demo

6.1	Overview	21
6.2	Supported Platforms	21
6.3	Getting Started	22
6.3.1	Prepare the Demo	22
6.4	Commands/Directions	22

Section number	Title	Page
Chapter 7		
FlexCAN and UART Communication Demo		
7.1	Overview	25
7.2	Supported Hardware	25
7.3	Getting Started	25
7.3.1	Hardware configuration	25
7.3.2	Prepare the Demo	25
7.4	Run the demo	26
Chapter 8		
FTM PDB ADC Demo		
8.1	Overview	27
8.2	Supported Platforms	27
8.2.1	Prepare the Demo	27
8.2.2	Run the demo	27
Chapter 9		
FlexTimer PWM Demo		
9.1	Overview	29
9.2	Supported Platforms	29
9.3	Getting Started	29
9.3.1	Prepare the Demo	30
9.3.2	Run the demo	30
Chapter 10		
Hello World Demo		
10.1	Overview	31
10.2	Supported Platforms	31
10.3	Getting Started	31

Section number	Title	Page
10.3.1	Hardware Settings	31
10.3.2	Prepare the Demo	31
10.4	Run the demo	32
10.5	Communication Interface Settings:	32
10.5.1	Customization Options	33

Chapter 11 Hardware Timer Demo

11.1	Overview	35
11.2	Supported Platforms	35
11.3	Getting Started	35
11.3.1	Prepare the Demo	35
11.3.2	Run the demo	36
11.4	Customization Options	36
11.4.1	Configure the Hardware Timer Used	36
11.4.2	Configure which clock is used by the hardware timer	36
11.4.3	Configure which instance of the module is used	36
11.4.4	Hardware Timer Period	37

Chapter 12 I2C Communication Demo

12.1	Overview	39
12.2	Supported Platforms	39
12.3	Getting Started	40
12.3.1	Hardware configuration	40
12.3.2	Terminal configuration	42
12.3.3	Run the demo	42

Chapter 13 I2C Demo with RTOS

13.1	Overview	45
-------------	---------------------------	-----------

Section number	Title	Page
13.2	Supported RTOS	45
13.3	Supported Platforms	45
13.4	Getting Started	46
13.4.1	Build with different RTOS support	46
13.4.2	Hardware configuration	46
13.4.3	Prepare the Demo	48
13.5	Run the demo	49

Chapter 14 LPTMR Demo

14.1	Overview	51
14.2	Supported Platforms	51
14.3	Getting Started	51
14.3.1	Prepare the Demo	51
14.4	Run the demo	52

Chapter 15 HTTP Server Demo on lwIP TCP/IP Stack

15.1	Overview	53
15.2	Supported RTOS	53
15.3	Supported Hardware	53
15.4	Getting Started	53
15.4.1	Prepare the Demo	53
15.4.2	Network Configuration	54
15.4.3	Run the demo	54

Chapter 16 Ping Demo on lwIP TCP/IP Stack

16.1	Overview	55
16.2	Supported RTOS	55

Section number	Title	Page
16.3	Supported Hardware	55
16.4	Getting Started	55
16.4.1	Prepare the Demo	55
16.4.2	Network Configuration	56
16.5	Run the demo	56

Chapter 17 TCP Echo Demo on lwIP TCP/IP Stack

17.1	Overview	57
17.2	Supported RTOS	57
17.3	Supported Hardware	57
17.4	Getting Started	57
17.4.1	Prepare the Demo	57
17.4.2	Network Configuration	58
17.5	Run the demo	58

Chapter 18 UDP Echo Demo on lwIP TCP/IP Stack

18.1	Overview	59
18.2	Supported RTOS	59
18.3	Supported Hardware	59
18.4	Getting Started	59
18.4.1	Prepare the Demo	59
18.4.2	Network Configuration	60
18.5	Run the demo	60

Chapter 19 MMDVSQ Demo

19.1	Overview	61
-------------	---------------------------	-----------

Section number	Title	Page
19.2	Supported Platforms	61
19.3	Getting Started	61
19.3.1	Prepare the Demo	61
19.4	Run the demo	61

Chapter 20 Power Manager Demo

20.1	Overview	63
20.2	Supported Hardware	63
20.3	Getting Started	63
20.3.1	Hardware Settings	63
20.3.2	Prepare the Demo	63
20.3.3	Run the demo	64
20.3.4	Supported Low Power Modes By Platform	64

Chapter 21 RTC Function Demo

21.1	Overview	67
21.2	Supported Hardware	67
21.3	Getting Started	67
21.3.1	Prepare the Demo	67
21.4	Run the demo	68

Chapter 22 SAI Demo

22.1	Overview	69
22.2	Supported Hardware	69
22.3	Getting Started	69
22.3.1	GCC Compiler notes	69
22.3.2	Hardware Settings	69

Section number	Title	Page
22.3.3	Prepare the Demo	69
22.4	Run the demo	70
22.5	Key Functions	72

Chapter 23 SD Card Demo

23.1	Overview	77
23.2	Supported Hardware	77
23.3	Getting Started	77
23.3.1	Prepare the Demo	77
23.4	Run the demo	78

Chapter 24 Thermistor Lab Demo

24.1	Overview	79
24.2	Supported Hardware	79
24.3	Getting Started	79
24.3.1	Prepare the Demo	79
24.3.2	Demo Code Overview	79
24.3.2.1	ADC Differential Mode of Operation	80

Chapter 25 Watchdog Timer Reset Demo

25.1	Overview	81
25.2	Supported Hardware	81
25.3	Getting Started	81
25.3.1	Hardware configuration	81
25.3.2	Prepare the Demo	82
25.4	Run the demo	82



Chapter 1

Introduction

Kinetis SDK (KSDK) includes applications which provide examples that show how to use KSDK drivers. This document describes the applications and provides instructions to configure each application (if available). The document also describes the required hardware setup and steps to run the applications.



Chapter 2

ADC Hardware Trigger Demo

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

2.1 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)
- TPM trigger

2.1.1 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.

2.1.2 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.

2.1.3 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.

Supported Platforms

2.1.4 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 connected 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.

2.2 Supported Platforms

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

The `adc_lptmr_trigger` demo Supported Platforms:

- FRDM-K22F
- FRDM-K64F
- FRDM-KL03Z48M
- FRDM-KL46Z
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- TWR-K22F120MK02
- TWR-KV31F120MKV30
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z32
- TWR-KV31F

The `adc_pdb_trigger` demo Supported Platforms:

- FRDM-K22F
- FRDM-K64F
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- TWR-K22F120MK02
- TWR-KV31F120MKV30
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z32
- TWR-KV31F

The `adc_pit_trigger` demo Supported Platforms:

- FRDM-K22F
- FRDM-K64F
- FRDM-KL46Z
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- TWR-K22F120MK02
- TWR-KV31F120MKV30
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV31F

The `adc_tpm_trigger` demo Supported Platforms:

- FRDM-KL03Z48M

2.3 Getting Started

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.

2.4 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.

2.5 Customization Options

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

Customization Options

2.5.1 Default configurations

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

2.5.1.1 ADC configurations

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

2.5.1.2 Sample frequency

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

2.5.2 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 $[0, \text{row} - 1]$ 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
```

2.5.3 Configure the signal frequency

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

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

2.5.4 Configure the ADC instance

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

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


2.5.5 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 3

ADC Low Power Demo

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

3.1 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.

3.2 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
- FRDM-KL46Z
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- TWR-K22F120MK02
- TWR-KV31F120MKV30
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z75M
- TWR-KV31F120M

3.3 Getting Started

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

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

Getting Started

- 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.

3.3.2 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 4

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.

4.1 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.

4.2 Supported Platforms

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

- TWR-KV10Z75M

4.3 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

Getting Started

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.

4.3.1 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.

4.3.2 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 5

DAC ADC Demo

This demo application demonstrates the DAC and ADC demo.

5.1 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.

5.2 Supported Platforms

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

- FRDM-K22F
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- FRDM-K64F
- FRDM-KL46Z
- TWR-K22F120MK02
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z75M
- TWR-KV31F120M
- TWR-KV31F120MKV30

5.3 Getting Started

5.3.1 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-KL46Z	PTE30/DAC_OUT	J4-11	PTE20/DIFF_AD-C0_DP	J4-1

Run the demo

TWR-K22F120M	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	DAC0_OUT	Primary Elevator - A32	PTE2/ADC1_SE6a	Primary Elevator - B27

5.3.2 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.

5.4 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.

5.5 Key Functions

uint8_t demo_start(demo_state_t *prevState)

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

Parameters

<i>*prevState</i>	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'.

Key Functions

Parameters

<i>*prevState</i>	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

<i>*prevState</i>	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

<i>*prevState</i>	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

<i>*prevState</i>	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.

Key Functions

Parameters

<i>*prevState</i>	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

<i>*prevState</i>	Pointer to previous state for the state machine.
-------------------	--

Returns

msg Returns 0.

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. On platforms where the SWAP feature is available, this demo operates with the last two sectors of the lower-half and the whole upper-half of Flash memory. On platforms without the SWAP feature, this demo operates with the last six sectors of Flash memory. The user should not store any program or data code in these locations.
2. The flash swap demo fails if the tested board has already 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-K64F
- TWR-K64F120M
- TWR-K60D100M

Platforms without SWAP feature:

- FRDM-K22F
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- FRDM-KL03Z48M
- FRDM-KL46Z
- TWR-K22F120M
- TWR-K22F120MK02

Commands/Directions

- TWR-K24F120M
- TWR-KV10Z75M
- TWR-KV31F120M
- 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
- TWR-K60D100M

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.



Run the demo

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.

Chapter 8

FTM PDB ADC Demo

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

8.1 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 μ s. 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.

8.2 Supported Platforms

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

- TWR-KV10Z75M

8.2.1 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.

8.2.2 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."



Supported Platforms

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 9

FlexTimer PWM Demo

This demo application demonstrates the FlexTimer PWM demo.

9.1 Overview

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

9.2 Supported Platforms

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

- FRDM-K22F
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- FRDM-K64F
- TWR-K22F120M
- TWR-K22F120MK02
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV31F120M
- TWR-KV10Z75M
- TWR-KV31F120MKV30

9.3 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	N/A	J1/P5
TWR-K22F120M	FTM0 - CH4	PTD4	D7 - YEL/GRN	J16/P1-P2

Getting Started

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	FTM0 - CH7	PTD7	D7 - YEL/GRN	J17/P7-P8
TWR-KV31F120-MKV30	FTM2 - CH1	PTB19	D6 - ORANGE	J17/P5 - P6

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

9.3.1 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.

9.3.2 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 10

Hello World Demo

This demo application demonstrates the Hello World demo.

10.1 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.

10.2 Supported Platforms

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

- FRDM-K22F
- FRDM-K22F120MK02
- FRDM-K22F120MK0264
- FRDM-K64F
- FRDM-KL03Z48M
- FRDM-KL46Z
- TWR-K22F120M
- TWR-K22F120MK02
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z75M
- TWR-KV31F120M
- TWR-KV31F120MKV30

10.3 Getting Started

10.3.1 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.

10.3.2 Prepare the Demo

1. Connect a USB cable between the PC host and the OpenSDA USB port on the board.

Communication Interface Settings:

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.

10.4 Run the demo

This is an example how to run the demo.

```
Hello World!
```

10.5 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-K22F120MK02	PTE0 (N/A)	PTE1 (N/A)	UART1
FRDM-K22F120M-K0264	PTE0 (N/A)	PTE1 (N/A)	UART1
FRDM-K64F	PTB17 (N/A)	PTB16 (N/A)	UART0
FRDM-KL03Z48M	PTB1 (J1-2)	PTB2 (J1-1)	LPUART0
FRDM-KL46Z	PTA2 (J1-4)	PTA1 (J1-2)	UART0
TWR-K22F120M	PTE0 (J30)	PTE1 (J29)	UART1
TWR-K22F120MK02	PTE0 (J30)	PTE1 (J29)	UART1
TWR-K24F120M	PTE0 (N/A)	PTE1 (N/A)	UART1
TWR-K60D100M	PTE8 (N/A)	PTE9 (N/A)	UART5
TWR-K64F120M	PTC4 (J15)	PTC3 (J10)	UART1
TWR-KV10Z75M	PTB17 (J22-2)	PTB16 (J21-2)	UART0
TWR-KV31F120M	PTB17 (J23)	PTB16 (J22)	UART0
TWR-KV31F120MK-V30	PTB17 (J23)	PTB16 (J22)	UART0

10.5.1 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.



Communication Interface Settings:

Chapter 11

Hardware Timer Demo

This demo application demonstrates using the hardware timer driver.

11.1 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.

11.2 Supported Platforms

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

- FRDM-K22F
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- TWR-K22F120MK02
- FRDM-K64F
- FRDM-KL03Z48M
- FRDM-KL46Z
- TWR-KV31F120MKV30
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV31F120M
- TWR-KV10Z75M

11.3 Getting Started

11.3.1 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.

Customization Options

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.

11.3.2 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
```

11.4 Customization Options

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

11.4.1 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
```

11.4.2 Configure which clock is used by the hardware timer

Determine which clock source is used by the hardware timer.

```
#define HWTIMER_LL_SRCCLK    kCoreClock
```

11.4.3 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
```

11.4.4 Hardware Timer Period

Determine the timer period (in microseconds).

```
#define HWTIMER_PERIOD      100000
```


Chapter 12

I2C Communication Demo

This demo application demonstrates the I2C demo.

12.1 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.

12.2 Supported Platforms

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

- FRDM-K22F
- FRDM-K22F120MK02
- FRDM-K22F120MK0264
- FRDM-K64F
- FRDM-KL03Z48M
- FRDM-KL46Z
- TWR-K22F120M
- TWR-K22F120MK02
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z75M
- TWR-KV31F120M
- TWR-KV31F120MKV30

Getting Started

12.3 Getting Started

12.3.1 Hardware configuration

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

FRDM-K22F, FRDM-K22F120MK02 & FRDM-K22F120MK0264:

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

FRDM-KL03Z48M:

Master Board		Connects To	Slave Board	
Pin Name	Board Location		Pin Name	Board Location
PTB3/I2C0_SCL	J2 Pin 10	->	PTB3/I2C0_SCL	J2 Pin 10
PTB4/I2C0_SDA	J2 Pin 9	->	PTB4/I2C0_SDA	J2 Pin 9
GND	J2 Pin 6	->	GND	J2 Pin 6

FRDM-KL46Z:

Master Board		Connects To	Slave Board	
Pin Name	Board Location		Pin Name	Board Location
PTE1/I2C0_SCL	J2 Pin 20	->	PTE1/I2C0_SCL	J2 Pin 20
PTE0/I2C0_SDA	J2 Pin 18	->	PTE0/I2C0_SDA	J2 Pin 18

GND	J2 Pin 14	->	GND	J2 Pin 14
-----	-----------	----	-----	-----------

TWR-K22F120M, TWR-K22F120MK02, TWR-K24F120M & 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-K60F120M & 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

TWR-KV31F120MKV30:

Getting Started

Master Board		Connects To	Slave Board	
Pin Name	Board Location		Pin Name	Board Location
PTD2/I2C0_SCL	Primary Elevator A7	->	PTD2/I2C0_SCL	Primary Elevator A7
PTD3/I2C0_SDA	Primary Elevator A8	->	PTD3/I2C0_SDA	Primary Elevator A8
GND	Primary Elevator A81	->	GND	Primary Elevator A81

12.3.2 Terminal configuration

Configure the PC host serial console as shown:

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

12.3.3 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 13

I2C Demo with RTOS

This demo application demonstrates the I2C demo on different RTOS.

13.1 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.

13.2 Supported RTOS

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

13.3 Supported Platforms

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

- FRDM-K22F
- FRDM-K64F

Getting Started

- FRDM-KL46Z
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M

13.4 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.

13.4.1 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 μ C/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.

13.4.2 Hardware configuration

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

Freescal 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

Freescal 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

Freescal Freedom FRDM-KL46Z

FRDM-KL46Z Single Board				
Master		Connects To	Slave	
Pin Name	Board Location		Pin Name	Board Location
PTB2/I2C0_SCL	J4 - Pin 6	->	PTE1/I2C1_SCL	J2 - Pin 20
PTB3/I2C0_SDA	J4 - Pin 8	->	PTE0/I2C1_SDA	J2 - Pin 18

FRDM-K64F Two Boards				
Master (Board #1)		Connects To	Slave (Board #2)	
Pin Name	Board Location		Pin Name	Board Location
PTB2/I2C0_SCL	J4 - Pin 6	->	PTE1/I2C1_SCL	J2 - Pin 20
PTB3/I2C0_SDA	J4 - Pin 8	->	PTE0/I2C1_SDA	J2 - Pin 18
GND	J2 - Pin 14	->	GND	J2 - Pin 14

TWR-K22F120M & TWR-K24F120M 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

Getting Started

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-K60F120M & 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

13.4.3 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.

13.5 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.



Run the demo

Chapter 14

LPTMR Demo

This demo application demonstrates the LPTMR demo.

14.1 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.

14.2 Supported Platforms

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

- FRDM-K22F
- FRDM-K22F120MK02
- FRDM-K22120MK0264
- FRDM-K64F
- FRDM-KL03Z48M
- FRDM-KL46Z
- TWR-K22F120M
- TWR-K22F120MK02
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV31F120M
- TWR-KV10Z75M
- TWR-KV31F120MKV30

14.3 Getting Started

14.3.1 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.

Run the demo

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.

14.4 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.

The output on the serial terminal is as shown:

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

Chapter 15

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.

15.1 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.

15.2 Supported RTOS

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

15.3 Supported Hardware

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

- FRDM-K64F
- TWR-K60D100M
- TWR-K64F120M

15.4 Getting Started

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

15.4.1 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

Getting Started

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 steps, see a Kinetis SDK User's Guide for your board.

15.4.2 Network Configuration

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

15.4.3 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 which the
board returns opens in the browser.

Chapter 16

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.

16.1 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.

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 KSDK Ping demo.

- FRDM-K64F
- TWR-K60D100M
- TWR-K64F120M

16.4 Getting Started

See the *lwIP TCP/IP Stack and Kinetis SDK Integration User's Guide* (document KSDKLWIPUG) for instructions 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 these settings:
 - 115200 baud rate
 - 8 data bits
 - No parity
 - One stop bit

Run the demo

- 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 a Kinetis SDK User's Guide for your board.

16.4.2 Network Configuration

Configure the IP address of PC network adapters as shown:

- 192.168.2.100

16.5 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 17

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.

17.1 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.

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 TCP Echo demo.

- FRDM-K64F
- TWR-K60D100M
- 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

Run the demo

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 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 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 18

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.

18.1 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.

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 UDP Echo demo.

- FRDM-K64F
- TWR-K60D100M
- 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

Run the demo

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 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 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 19

MMDVSQ Demo

This demo application demonstrates how to use MMDVSQ driver.

19.1 Overview

The MMDVSQ Demo project is a simple demonstration program to show how to use the MMDVSQ driver. This demo demonstrates the efficiency of division and square root operations and typical C functions.

19.2 Supported Platforms

This demo supports the following Tower System module:

- TWR-KV10Z75M

19.3 Getting Started

19.3.1 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 a Kinetis SDK User's Guide for your board.

19.4 Run the demo

This is an example serial terminal output:

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

The tickcycles are used as a reference.



Run the demo

Chapter 20

Power Manager Demo

This demo application demonstrates how to use the Power Manager.

20.1 Overview

The Power Manager demo application demonstrates different Power Manager modes supported by the Kinetis SoCs. The set of supported low power modes and their transition possibility differ platform to platform. See section: "System Mode Controller" in a Reference Manual for each Kinetis Sub-family microcontroller.

20.2 Supported Hardware

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

- FRDM-K22F
- FRDM-K64F
- FRDM-KL03Z
- FRDM-KL46Z
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M

20.3 Getting Started

20.3.1 Hardware Settings

The demo does not require 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.

20.3.2 Prepare the Demo

Follow the instructions in *Getting Started with Kinetis Software Development Kit (KSDK)* to:

- Setup hardware connections
- Configure a Terminal session
- Build and download application to targeted devices

Note: The demo is configured to work with the Terminal baudrate at 9600 bps. When running the demo, unplug all debugger devices.

Getting Started

20.3.3 Run the demo

1. Press the reset button on the hardware.
2. A control menu was displayed on the Terminal window. Note that the list on the menu is not the same for all platforms. It depends on the list of supported low power modes. For example, on the TWR-K64F120M, this menu displays:

```
##### Power Manager Demo #####
Core Clock = 120000000Hz
SMC mode = kPowerModeRun
Select the desired operation
Press A for enter: Wait - Wait mode
Press B for enter: Stop - Stop mode
Press C for enter: VLPR - Very Low Power Run mode
Press D for enter: VLPW - Very Low Power Wait mode
Press E for enter: VLPS - Very Low Power Stop mode
Press F for enter: LLS - Low Leakage Stop mode
Press G for enter: VLLS0 - Very Low Leakage Stop 0 mode
Press H for enter: VLLS1 - Very Low Leakage Stop 1 mode
Press I for enter: VLLS2 - Very Low Leakage Stop 2 mode
Press J for enter: VLLS3 - Very Low Leakage Stop 3 mode
Press K for enter: RUN - Normal RUN mode
Waiting for key press..
```

3. Enter a command by pressing the corresponding input key. If the user enters an invalid mode transition, the demo displays this message in the Terminal window:

```
Can not go from RUN to VLPW directly
Next loop
```

In most valid mode transitions, the SoC wakes up after receiving the RTC alarm or the GPIO switch trigger. However, in some modes, the SoC only accepts either RTC alarm or the GPIO switch trigger. In that case, the demo prints the following message on the terminal:

```
The board does not support wake up from this mode by RTC due to disabled External
Entering Very Low Leakage Stop 0 mode, press the SW1 button to wake up.
Wake up goes through Reset sequence.
```

20.3.4 Supported Low Power Modes By Platform

This table shows the supported modes on different platforms:

Platform	Supported Power Modes	Wakeup Sources
----------	-----------------------	----------------

FRDM-K22F120M, TWR-K22-F120M	WAIT, STOP, VLPR(4MHz), VL-PW, VLPS, LLS3, VLLS0, VLLS1, VLLS2, VLLS3, RUN(80MHz), H-SRUN(80MHz)	RTC, SW1
FRDM-K64F120M, TWR-K64-F120M	WAIT, STOP, VLPR(4MHz), VL-PW, VLPS, LLS, VLLS0, VLLS1, VLLS2, VLLS3, RUN(120MHz)	RTC, SW1
FRDM-KL03Z48M	WAIT, STOP, VLPR(1MHz), VL-PW, VLPS, LLS, VLLS0, VLLS1, VLLS2, VLLS3, RUN(48MHz)	RTC, SW2
FRDM-KL46Z48M	WAIT, STOP, VLPR(4MHz), VL-PW, VLPS, LLS, VLLS0, VLLS1, VLLS3, RUN(48MHz)	RTC, SW1
TWR-K24F120M	WAIT, STOP, VLPR(4MHz), VL-PW, VLPS, LLS, VLLS0, VLLS1, VLLS2, VLLS3, RUN(48MHz)	RTC, SW2
TWR-K60D100M	WAIT, STOP, VLPR(4MHz), VL-PW, VLPS, LLS, VLLS1, VLLS2, VLLS3, RUN(100MHz)	RTC, SW1

@

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 Peripheral 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
- FRDM-KL03
- FRDM-KL46Z
- TWR-K22F120M
- TWR-K24F120M
- TWR-K60D100M
- 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 these settings:
 - 115200 baud rate

Run the demo

- 8 data bits
 - No parity
 - One stop bit
 - No flow control Note that for the FRDM-KL03 platform, the terminal baud rate should be 9600.
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 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-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.

Run the demo

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 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.

This table 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	GN- U-G- CC	KDS- GCC	IAR	ARM	GN- U-G- CC	KDS- GCC	IAR	ARM	GN- U-G- CC	KDS- GCC
TW- R-- K22- F120- M	44.1	44.1	11.- 025	11.- 025	16	16	16	16	2	2	2	2

Key Functions

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

<i>newWav</i>	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

<i>handler</i>	pointer to codec handler structure.
<i>module</i>	name of module on codec to set the volume for.
<i>volumeCtrl</i>	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

<i>dspType</i>	Used to select one DSP function to perform on the data.
<i>volumeCtrl</i>	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

Key Functions

<i>waveFile</i>	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

<i>pcmBuffer</i>	Pointer to data array containing WAV formatted audio data.
<i>volumeCtrl</i>	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

<i>pdataBuffer</i>	Pointer to data array containing WAV formatted audio data.
<i>length</i>	length of WAV file to send.
<i>dataFormat</i>	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

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

Returns

float32_t Returns fundamental frequency in Hz.



Key Functions

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 the 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-K60D100M
- 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 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 a Kinetis SDK User's Guide for your board.

Run the demo

23.4 Run the demo

1. Insert an SD or a micro-SD card depending on the board connector. 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 and measure on-board thermistor.

24.1 Overview

This lab shows how to configure and use the ADC module to sample the differential voltage across on-board thermistors RT1-RT4. If the user touches any on-board thermistor with a finger, the lab application detects a change in the thermistor temperature and starts 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

This Tower System module is supported by the Thermistor Lab demo.

-TWR-KV10Z75M

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 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 the FlexTimer0 via the PDB. The FlexTimer is configured to generate the 16 KHZ PWM and the channel1 trigger is used to trigger both ADCs via the PDB. The PDB is configured to generate four delayed trigger signals to both ADCs per FlexTimer0 Channel 1 trigger and, as a result, 4 ADCs samples are converted per each FlexTimer channel trigger. The ADC is configured to be in a 16-bit differential and ping-pong mode.

When an ADC conversion is complete, an interrupt is generated by the ADC module and an interrupt service routine is executed. The interrupt service routine ADCn_ISR() calls the ADCn_Task which executes these tasks: reads ADC results registers. filters ADC results with low-pass FIR filter. differentiates filtered

Getting Started

results to detect a change in a voltage across the thermistor. detects 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, configure the ADC for a differential mode of operation. In a differential mode, the ADC measures a difference between two analogous inputs. The ADC enables selecting input pairs which are treated as differential inputs.

Detection of a Change of Thermistor Voltage

If a user places a finger on a thermistor, its temperature increases. The temperature rise results in a voltage decrease across the resistor. If the user removes the finger, the temperature decreases and the voltage goes up.

A simple differentiators are used to detect a voltage change. The filtered thermistor voltage is stored in a buffer. The buffer size is defined by the `BUFF_SIZE`. The differentiator calculates a difference between an actual voltage sample and a sample delayed by `i_delay` pointer, which points to 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 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), an action is taken and a corresponding LED starts to flash.

Chapter 25

Watchdog Timer Reset Demo

This demo application demonstrates the Watchdog Timer Reset demo.

25.1 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.

25.2 Supported Hardware

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

- FRDM-K22F
- FRDM-K22FK02
- FRDM-K22FK0264
- FRDM-K64F
- TWR-K22F120M
- TWR-K22F120MK02
- TWR-K24F120M
- TWR-K60D100M
- TWR-K64F120M
- TWR-KV10Z75M
- TWR-KV31F120M
- TWR-KV31F120MKV30

25.3 Getting Started

25.3.1 Hardware configuration

These switch buttons are used by this demo:

Platform	Switch	Notes
FRDM-K22F120M(K02)	SW2	
FRDM-K64F120M	SW1	
TWR-K22F120M	SW1	
TWR-K24F120M	SW1	
TWR-K60F100M	SW1	
TWR-K64F120M	SW1	

Run the demo

TWR-KV10Z32M	SW1	
TWR-KV31F120M	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" outputs to the terminal.

25.3.2 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. Press either the reset button on your board or launch the debugger in your IDE to begin running the demo.

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

25.4 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	Yellow
TWR-K24F120M	Red
TWR-K60F100M	Yellow
TWR-K64F120M	Yellow
TWR-KV10Z32M	Yellow
TWR-KV31F120M	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 outputs to the terminal.

/*!

26 Revision history

This table summarizes the revisions made to this document.

Revision History		
Revision number	Date	Substantive changes
0	12/2014	Kinetis SDK 1.1.0 release
1.0.0	07/2014	Initial release

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.

Freescale, the Freescale logo, Kinetis, Processor Expert are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. Tower is a trademark of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. ARM, ARM Powered logo, Keil and Cortex are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

© 2014 Freescale Semiconductor, Inc.

