# FRDM-KL02Z

## Lab Guide

Rev. 0

## About FRDM-KL02Z

KL02 chip acts as an ultra low power, cost effective microcontroller to provide developers an appropriate entry-level 32-bit solution, and also acts as a bridge between 8-bit and 32-bit devices.  It's valuable for cost-sensitive, portable applications requiring long battery life.
There are 5 demos prepared for developers, all of them are based on CodeWarrior, IAR and KEIL, so they are very easy to reuse and develop.

## Software requirements

1. IDE

   The sample projects can support 3 kinds of IDE: CodeWarrior(v10.3), IAR(v6.40.5) and KEIL(v4.60).
2. TSSW

   KL02 has no TSI module on chip, however the sample code offers a GPIO method to emulate touch sensor function based on TSS 3.0.1.
3. Uart Terminal

## Easy-to–use FRDM-KL02Z

Use a mini USB cable to attach the FRDM-KL02Z to a USB port.

### Demo 1: KL02_Freedom_Demo

This is the default demo and shows the functionality of TSS library, PWM, Accelerometer Sensor, Timer and I2C module. KL02 has no TSI module on chip, so in this demo the Analog Slider pad is controlled with GPIO method by TSSW.
Down load the code and run. On the uart terminal, the following information will be displayed:

```
‶************************************************************″
“Running in KL02 demo project.”
 ‶--Unlock the LEDs: slide from 2 to 1 on E1 pad.”
‶--Lock the LEDs: slide from 1 to 2 on E1 pad.”
‶************************************************************″
“KL02 Freedom Demo. Slide to unlock first.‶
“Running the freedom_test project.”
```

And now 3 steps can be followed:

1. **Slide Down to Unlock.**

    unlock the LEDs and the uart  terminal will display :

   ```
   “LEDs are unlocked. Slide to lock.”
   “Tilt the board to check the changing of LEDS' colour and
   brightness.”
   ```

2. **Tilt to adjust the LED.**
   With the demo “Unlocked”, tilting the board will change the RGB colour and brightness .
3. **Slide Up to Lock.**
   Lock the LEDs, which will display the unchanged colour. The terminal will display:
   ```
   “LEDs are locked. Slide to unlock.”
   ```

## Demo 2: KL02_GPIO_TSS_Demo

This demo will show how to use TSSW and configure the GPIO method. Down load the code and run. The uart terminal will display:

```
“********************************************************************“
“Running in KL02_GPIO_TSS_Demo.”
“Slide on the electrode to change the LED's brightness.“
“KL02 has no TSI, but TSSW takes the GPIO method to realize touch
sensor. “
“********************************************************************”
```

And now slide on the E1 pad, the LEDs’ brightness will change.

## Demo 3: KL02_LowPower_Demo

This demo can show how to configure all the low power modes, besides the current consumption can be measured from J4(this jump is default on, open it when using). Download the code and run. The uart terminal will display:

```
“*********************************************************”
“ 1.Wait    mode”
“ 2.PStop2  mode”
“ 3.PStop1  mode”
“ 4.Stop    mode”
“ 5.VLPR    mode”
“ 6.VLPW    mode”
“ 7.VLPS    mode”
“ 8.VLLS3   mode”
“ 9.VLLS1 mode”
“ A(a).VLLS0-1 mode -- POR ON”
“ B(b).VLLS0-2 mode -- POR OFF”
“ Wake up source: LPTMR, Reset or NMI”
“ *******************************************************”
“ POR before changing into another power mode.”
“ Enter the key number to choose one mode, mode a and b need external
clock input from Extal pin.”
```

And then follow the information to choose one mode needed. The default wake up source is the LPTMR module and the wake up time is set to about 5s, if the developers want to get more time to measure the current, it is easy to modify in the code.

## Demo 4: LQRUG_bme_ex1

The Bit Manipulation Engine (BME) provides hardware support for atomic read-modify-write memory operations to the peripheral address space in Cortex-M0+ based MCU.
This demo will show the difference between the normal C code bit operation and BME bit operation, the uart terminal will display the Systick time used by both of them when realize the same function. Part of the displayed information is like the following:

```
“extracted data is 2”
“systick start value: 0xffffe9”
“systick end value: 0xffffda”
“actual execution cycle for UBFX operation with normal C: 0xd”
“extracted data is 2”
“systick start value: 0xffffef”
“systick end value: 0xffffe2”
“actual execution cycle for UBFX operation with bme macro: 0xb”
```

**Note:**
This demo needs the compiler has the highest level of optimization.

## Demo 5: bootloader

This demo prepares a flash bootloader, which takes AN2295- Developer’s Serial Bootloader for reference.

Note:

1. This demo needs the compiler has the highest level of optimization.
2. The user application should obey the protocol of AN2295’s, and in the user application the fei_fee() should not check whether it is in fei mode or not.
3. Baud rate: 19200