**NXP Semiconductors**
Application Notes

Document Number: AN5325
Rev. 0, 08/2016

# Emulating I2S bus on KE06

## Contents



## 1. Introduction

This application note shows how to use a typical SPI interface and proper timer to emulate the I2S peripheral to transmit digital audio data to external audio codec IC. This example takes a low cost and low end MCU KE06 as an example to do the demo. Meanwhile, it should be able to apply to all of the Kinetis MCU which has SPI and FTM (or TPM). As KE06 does not have DMA, this example uses the polling mode of SPI for I2S transmission.

This document only introduces the emulating of the I2S transmission by SPI and timer, but the way of emulating the I2S receiver should be similar.

## 2. Overview of Emulating I2S Bus

The I2S is an audio bus using a three-wire connection for synchronous serial data communication. Data are transmitted on the TXD line (MSB first). Typical data length is 16/24/32 bits. The transmitter data are synchronized on the rising edge of BCLK and the receiver data on the falling edge of BCLK. A two-channel audio signal is represented by two data words, the right and the left channel sample, transmitted and multiplexed on the same wire. The frame sync (TXFS) control signal determines if the word is for the right or

left channel. TXFS can be synchronized to either the rising or the falling SCK edge and precedes the MSB by one BCLK period in order to have enough time to store the data by the receiver. In the Kinetis KE06 MCU, we use SPI to generate the bit clock (BCLK) on SPIx_SCK pin and shift out the stream of audio data on SPIx_MOSI pin. A missing frame sync (TXFS) signal is generated by the FlexTimer PWM output pin (FTMxCHx), while the timer counter input clock is from the BCLK edges. This means that the FlexTimer counter clock input must be externally connected to the SPI clock output. When the timer counter counts to 16 BCLK edges, the counter is reloaded and the counter output is toggled, generating the TXFS output signal. The I2S Master Clock (MCLK) is directly from on board 24.576 Mhz crystal of the TWR-AUDIO-SGTL for easy demo. In the best case, the MCLK should be generated by MCU with the same clock source of SPI to reduce the jitter.

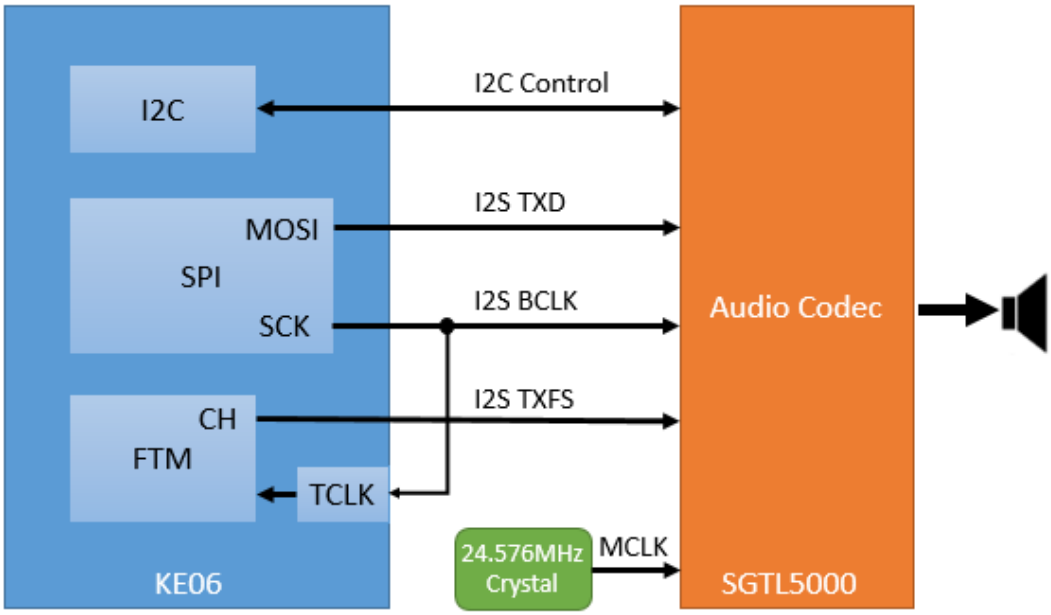


**Figure 1. System Diagram**

The audio data is received by a SGTL5000 audio codec IC on the TWR-AUDIO-SGTL card. The SGTL5000 works in the I2S Slave mode, and playback the audio data to a headphone or speaker. The initialization and control to the SGTL5000 is done by the I2C bus. Any other I2S audio codec IC can be used for reconstruction of the audio signal on the customized hardware.

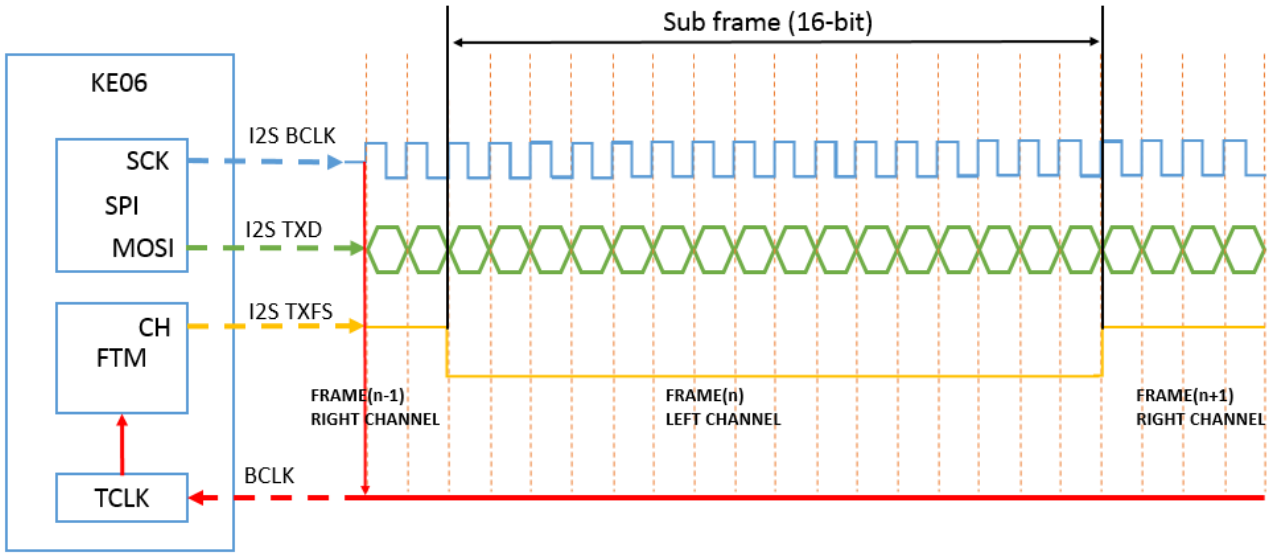


**Figure 2. I2S Timing**

*Figure 2* shows the exact signal timing. The data is valid on the rising edge of the BCLK. One sub frame contains 16 data and the frame sync is toggled on the rising edge of the BCLK after one sub frame transmission is completed. When the frame sync is low, the data of LEFT channel is sent on the TXD. When it switches to high, the data of RIGH channel is sent on the TXD. The BCLK is looped back externally from SPI SCK pin to TCLK1 input pin, acting as the FTM counter clock source.

# 3. Environment Setup

## 3.1. Requirements

To implement and run this example, there are some requirements for both software and hardware:

### 3.1.1. Software requirements

- IAR Embedded Workbench 7.40
- Example source code package (include KE06 driver lib)

### 3.1.2. Hardware requirements

This example uses the following hardware boards:

The technical documentation editors will remove the RED Preliminary, FCP, and NDA information, before publication.

- FRDM-KE06Z
- TWR-AUDIO-SGTL (RevF)
- TWR-ELEV (primary and secondary)

The FRDM-KE06Z is official released FRDM development platform for Kinetis KE06 MCUs. It enables easy access to MCU I/O.

The TWR-AUDIO-SGTL is a peripheral module compatible with the NXP® Tower® System and features the SGTL5000 audio codec. This peripheral module provides an audio interface for the Tower System and can be used with a wide range of Tower System MCU/MPU, peripheral, sensor and communication modules.

The TWR-ELEV Elevator modules are the basic building blocks of the Tower® System. Designed to connect MCU and peripheral modules, the Elevator modules provide the power regulation circuitry and structural integrity needed for all configurations of an assembled Tower System.

## 3.2. Hardware setup

### 3.2.1. Connection between boards

To do this example, the TWR-AUDIO-SGTL board is set up with the TWR-ELEV, and the FRDM-KE06Z is connected with TWR-ELEV by wire. *Table 1* shows the connection between boards.

**Table 1. Boards Connection**

| Pin Function | FRDM-KE06Z | TWR-AUDIO-SGTI (TWR-ELEV) |
|---|---|---|
| I2C SDA | PTA2 (J2-18) | ELEV_I2C0_SDA (PRIMARY A8) |
| I2C SCL | PTA3 (J2-20) | ELEV_I2C0_SCL (PRIMARY A7) |
| I2S BCLK | PTB2 (J2-12) | ELEV_I2S0_SCLK (PRIMARY A22) |
| I2S TX FS | PTC5 (J1-11) | ELEV_I2S0_LRCLK (PRIMARY A23) |
| I2S TXD | PTB3 (J2-8) | ELEV_I2S0_DIN (PRIMARY A25) |
| GND | GND (J2-14) | GND (PRIMARY A26) |

*Figure 3* shows how to connect the three boards:

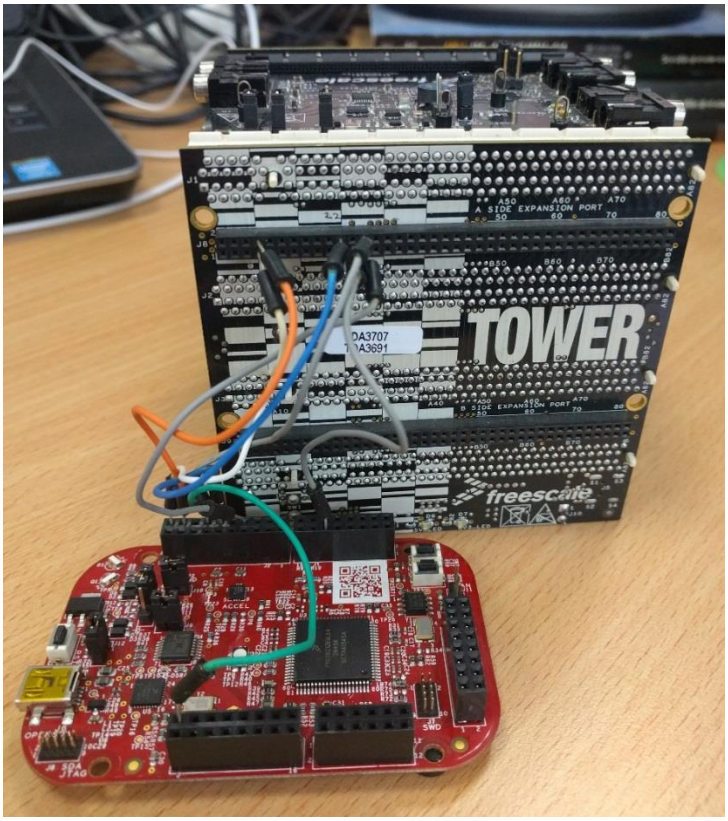


**Figure 3. Board Connection**

## 3.2.2. Board configurations

### FRDM-KE06Z

To generate Frame Sync signal for I2S by using FTM, the SPI SCK (BCLK) must be used as FTM clock source, connected to TCLK1:

- PTB2 (J2-12) <-> PTE0 (J4-1)

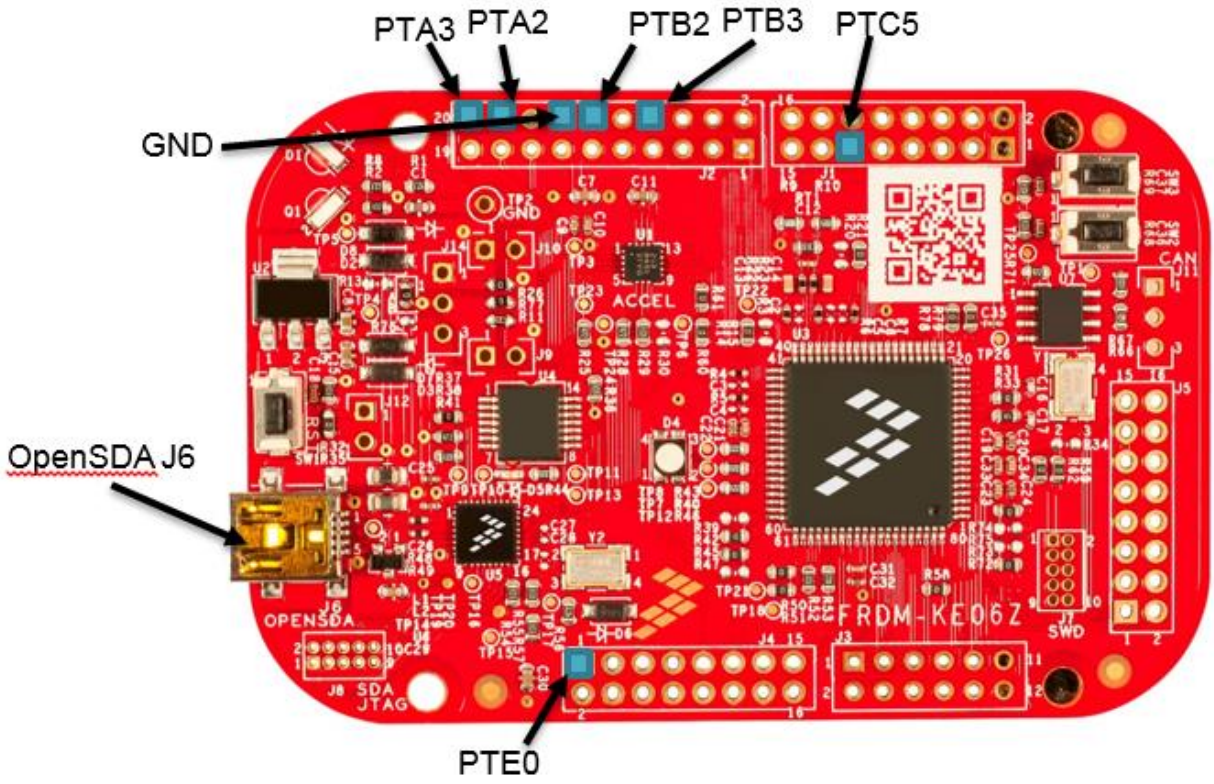


**Figure 4. FRDM-KE06Z and Used Pins**

**TWR-AUDIO-SGTL**

- Power from the ELEVATOR 5V: J5 is set.
- MCLK is in from onboard 24.576MHz crystal: J6 is unset.
- Headphone is inserted to J7.

# 4. Software Description

This section describes how the software is configured for each peripheral to emulate the I2S bus.

The emulated I2S bus timing in this example is defined as: in the examples.

- Sample rate – 12KHz
- Channel – two channels
- Format – 16bit per channel, left justified, little endian
- Bit Rate – 32x12KHz = 384KHz

## 4.1. Software configurations

**System Clock**

- External 8Mhz OSC as clock source (FEE)

- Core clock – 40Mhz
- Bus clock – 20Mhz

**SPI**

- SPI0 SCK
  - Baud Rate – 384kbps
  - Clock Polarity – Active high
  - Clock Phase - First edge on SPSCK occurs at the start of the first cycle of a data transfer. This can make sure the Audio Codec fetch the data on the clock falling edge correctly.
- SPI0 MOSI
  - MSB

**FTM**

- FTM1 work in EPWM mode
- Clock source is from TCLK1 input pin
- FTM1 C1V=16, MOD=31
- FTM1 CH1 output as I2S TX_FS

**I2C**

- I2C0 SDA/SCL access the SGTL5000 audio codec

**SGTL5000**

- Enable DAC with I2S IN
- Unmute DAC
- I2S data valid on falling edge
- Data length 16

To make sure the audio codec received the I2S data correctly, the SPI must be always working to transmit data, and keep the clock and frame sync signal continuously. In this example, we use the polling mode to make demo easy. In real use case, SPI's interrupt mode should be used to free CPU. For other MCU which has DMA capability, the DMA is recommended to transmit audio data from memory to SPI data register. Otherwise, the CPU cannot be free to do other tasks.

The audio data is stored in the music[] array in a type of char in the flash for demo. It contains a ring tone in a format that:

1. the first byte is high 8 bit of the left channel
2. the second byte is low 8 bit
3. the third byte is high 8 bit of the right channel
4. the fourth byte is low 8 bit

Therefore, we just need to fill the audio data into the SPI data register in sequence of the array.

## 4.2. Software workaround

To use the SPI SCK as external clock source for FTM EPWM counter, there is a limitation: the FTM counter is increased on the rising edge of external clock, so the channel match and overflow event (FTM1CH1 output toggle) would happen on the rising edge of the external clock. Due to this, the data on the SPI MOSI must be valid on the SCK falling edge to make sure the data would not be fetched on the edge of frame sync. This causes the first left channel frame sync (TX_FS) would be one BCLK ahead of I2S data. The first left channel only contains 15 bits valid data, as shown in *Figure 5*.

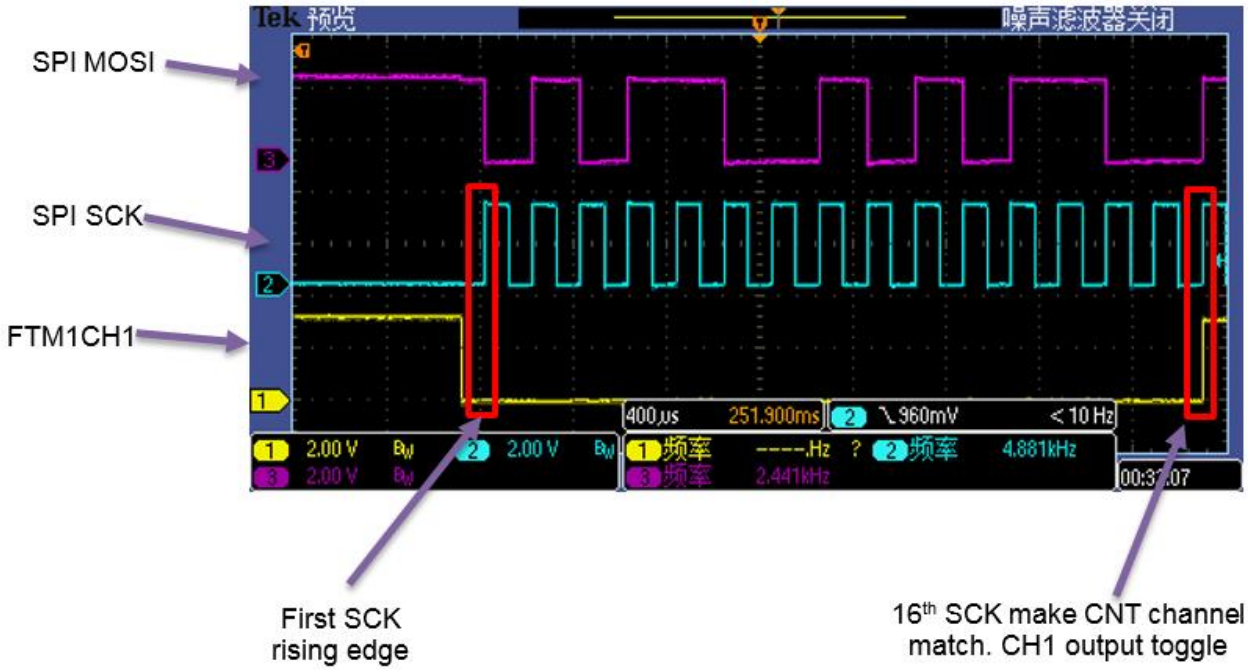


**Figure 5. First I2S frame timing**

A software workaround must be made to drop the 8th bit in the first data, to make sure all of the other data left and TX_FS is synced. Below is a simple code snippet for dropping the 8th bit in the first data:

```
SPI_WriteDataReg(SPI0, (music[0] << 1) | ((music[1] >> 7) & 0x01));
/* start to transfer to send and receive date in non-block mode */
For (i = 1; i < MUSIC_LEN - 1; i++)
{
    while(!SPI_IsSPTEF(SPI0));
    SPI_WriteDataReg(SPI0, (music[i] << 1) | (music[i+1] >> 7));
}
```

# 5. Running the Demo

The user can download a program image to the MCU through OpenSDA. The PC host obtains a serial port after a USB cable is connected between the PC host and the on-board debugger USB socket (J6).

The project and workspace files of the demo are located in:

*build/iar/ke06/SPI_MasterInt_demo/iar*

The source file is located in:

*src/projects/KE06/SPI_MasterInt_demo*

Open the workspace file, e.g. IAR *.eww*, and then build the demo project. Download the demo.

Before running the demo, please power up the TWR-AUDIO-SGTL board by connecting a USB cable between TWR-ELEV J5 socket and PC, and plug a headphone on the TWR-AUDIO-SGTL J7 connector. After running the demo, you can hear an audio ring is playback in a loop.

# 6. Other Considerations

The general I2S frame rate is 12K/32K/44.1K/48K/96K, bit length is 16/24/32. So the BCLK frequency is frame_rate x 16, 24 or 32. For I2S clocking, a 32.768 KHz, 12.288 MHz or 24.576 MHz clock is recommend to generate the BCLK/FSCLK.

You cannot get a very accuracy BCLK from SPI SCK by using the internal IRC or connected a normal external crystal like 8MHz, 12MHz. The SPI clock source is from BUS_CLK, and no fractional divider supported. So to get an accuracy sample rate, a 12.288MHz, 24.576MHz is recommend to connect to EXTAL/XTAL for the BCLK/FSCLK generation. The FLL bypassed external (FBE) clock mode is selected with FLL disabled.

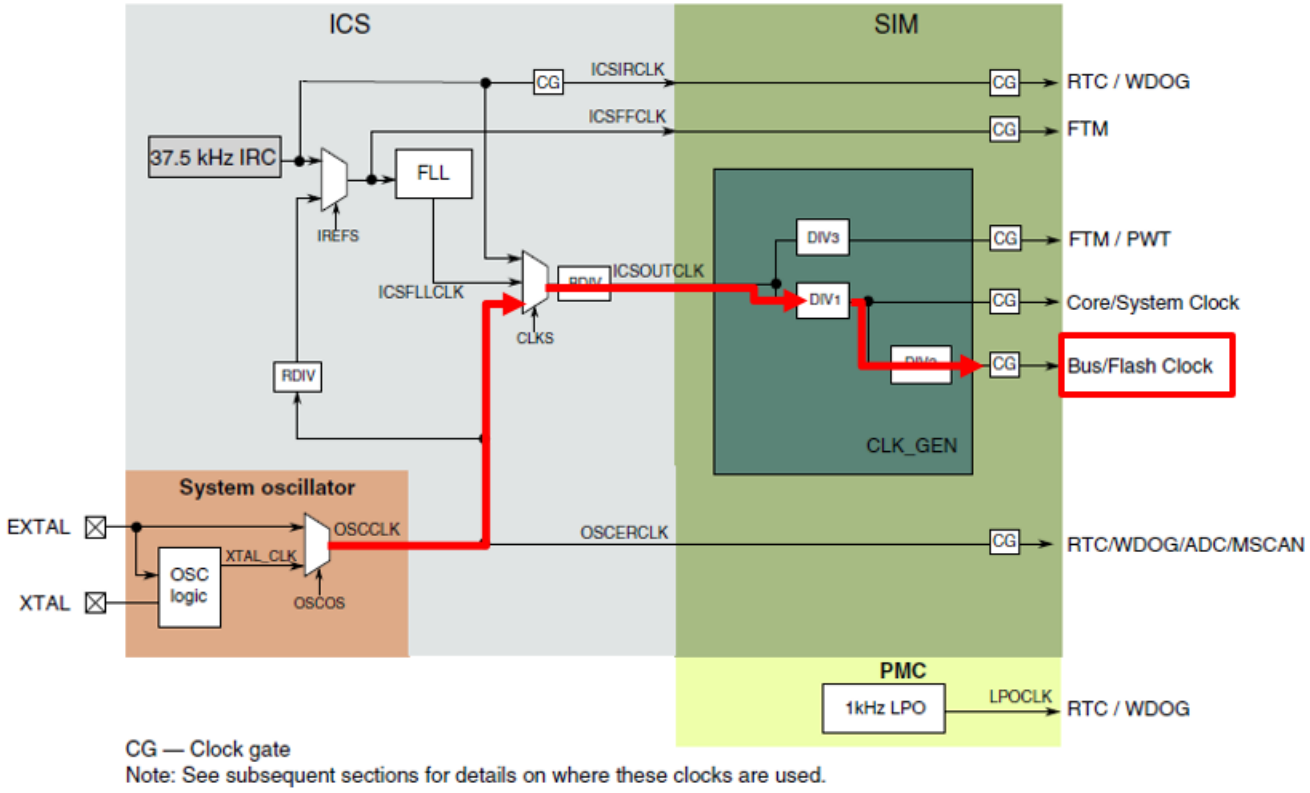


**Figure 6. KE06 clock distribution**

# 7. Conclusions

This application shows the Kinetis KE06 MCU acting as the digital audio data source, and the I2S audio bus functionality is successfully emulated by SPI and timer/counter modules.

# 8. References


1. Kinetis KE06 SoC Reference Manual and Data Sheet

   http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/kinetis-cortex-m-mcus/e-series-5v-robust-m0-plus-m4/kinetis-ke06-48-mhz-mainstream-with-can-microcontrollers-mcus-based-on-arm-cortex-m0-plus-core:KE06?fpsp=1&tab=Documentation_Tab

2. FRDM-KE06Z: Freedom Development Platform for Kinetis® KE06 MCUs

   http://www.nxp.com/products/software-and-tools/hardware-development-tools/freedom-development-boards/freedom-development-platform-for-kinetis-ke06-mcus:FRDM-KE06Z

3. Emulating I2S bus on Kinetis-M
   http://cache.nxp.com/files/microcontrollers/doc/app_note/AN4944.pdf?fsrch=1&sr=1&pageNum=1


# 9. Revision history

**Table 2. Revision history**

| Revision number | Date | Substantive changes |
|---|---|---|
| 0 | 08/2016 | Initial release |

***How to Reach Us:***

**Home Page:**
nxp.com

**Web Support:**
nxp.com/support



Document Number: AN5325
Rev. 0
08/2016