NXP Semiconductors

Application Note

Document Number: AN5397

Rev. 0, 02/2017

Emulating SSI Using FlexIO

1. Introduction

An SSI (Serial Synchronous Interface) is a widely used serial interface between an absolute position sensor and a controller. While most Kinetis parts do not directly support an SSI peripheral, one good solution is to use FlexIO to emulate SSI to get the similar performance as that of a dedicated SSI peripheral. This application note shows how to configure FlexIO and to ensure that the SSI function works well.

Contents

2. Overview of SSI

A Synchronous Serial Interface (SSI) is serial interface standard for industrial applications between a master (such as a controller) and a slave (such as a sensor). In this case, the SSI uses Texas Instruments synchronous serial frame format. It takes one serial clock period starting at its rising edge, prior to the transmission of each frame. For this frame format, both the SSI and the off-chip slave device drive their output data on the rising edge of SSInClk and latch data from the other device on the falling edge.

The typical SSI timing is shown in the following figure:

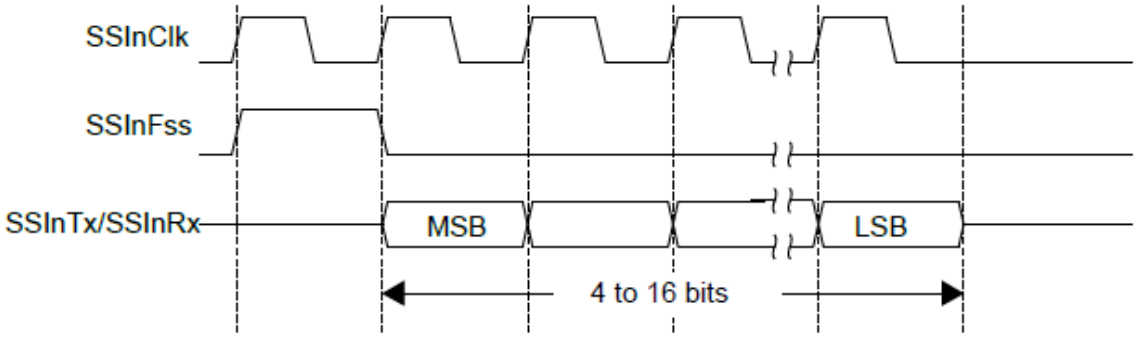

Figure 1. SSI timing (single transfer)

FlexIO emulates the SSI timing very effectively, as there is a high quality shifter and timer resource in the FlexIO module which supports flexible configuration to implement different timing. The next section details how to emulate the SSI master and slave function.

3. Emulating SSI

To emulate the SSI timing, as shown in Figure 1, requires an extra clock period for FSS signals (high pulse). The data is valid when FSS is asserted. Transmit data on the clock rising edge, and latch the data on the clock falling edge. The following two sub-sections detail the configuration of the master and slave devices.

3.1. Master configuration

There are two configuration solutions to implement the SSI master function. The first is to take one timer and three shifters, and the second is to use two timers and two shifters. The difference between the two solutions is in FSS signals generation. The first solution uses one shifter output and one start signal as FSS. The second solution is to use one timer to generate FSS signals. Due to synchronization delays the setup time for the serial input data is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

The detailed configuration is as follows:

- Use the timers(n) to generate the clock of SSI (SSInClk), and the shifter(n) to enable start bit transfer and transmit data on each rising edge of SSInClk. Use the shifter(n+1) to transfer the start bit as the high pulse of SSInFss, and the shifter (n+2) to receive data on each falling edge of SSInClk.

The master configuration diagram is shown in the following figure:

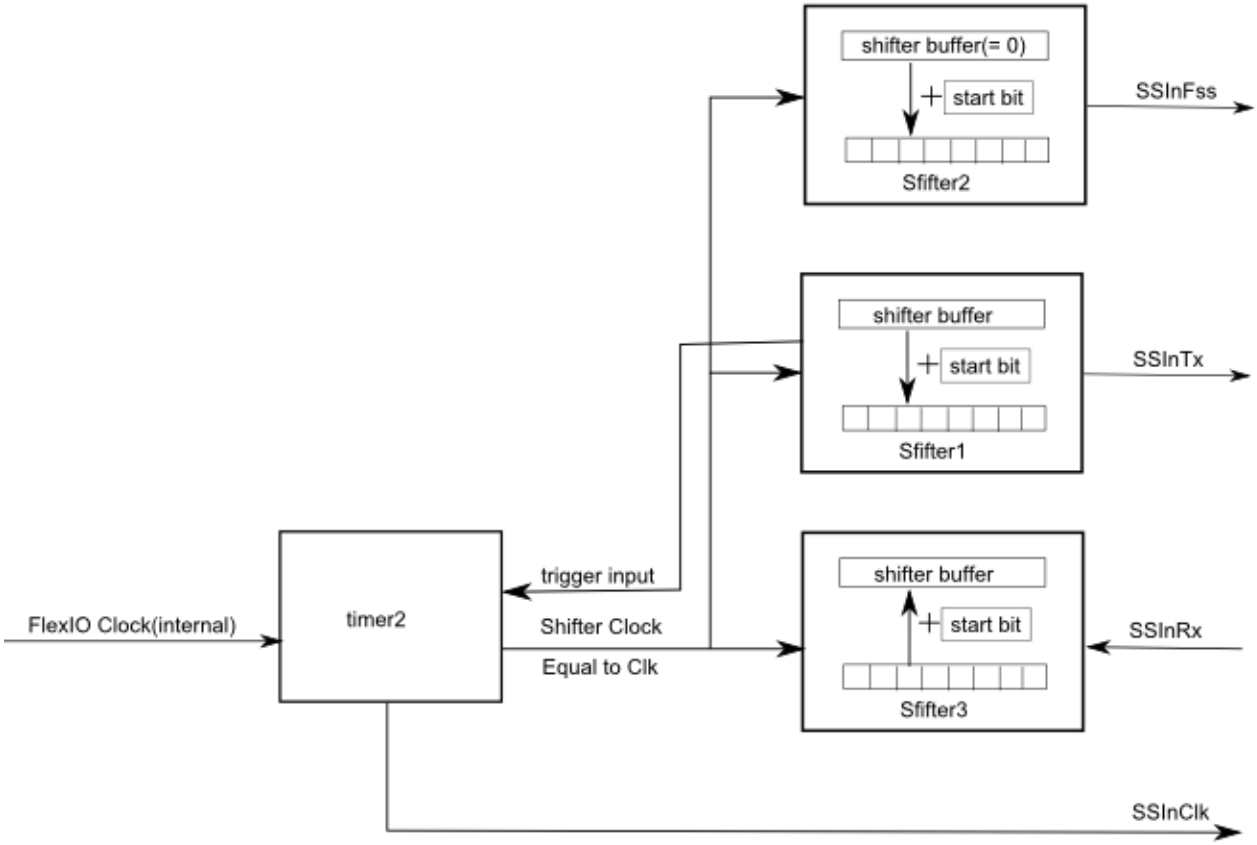

Figure 2. Master configuration (a)

The detailed configuration is as follows:

- Configure Timer (n) triggered by shifter (n) status flag, and configure as dual 8-bit counters baud/bit mode.
- Configure Timer (n) enable on trigger high and disable on compare, enable start bit, and decrement counter on FlexIO clock, shift clock equals timer (n) output.
- Configure shifter (n) triggered by timer (n), shift on the rising edge of shifter clock, enable transmit mode.
- Configure shifter (n) transmitter outputs start bit with value 0.
- Configure shifter (n+1) triggered by timer (n), shift on the rising edge of shifter clock, enable transmit mode.
- Configure shifter (n+1) transmitter outputs start bit with value 1 (to generate one high pulse).
- Configure shifter (n+2) triggered by timer(n), shift on the falling edge of shifter clock, enable receive mode.
- Configure shifter (n+2) receiver/match store sets error flag if start bit is not 0.

The detailed register configuration is as follows:

- FlexIO_TIMCTL[2] = 0x5C30701 // select pin 7 as clock output
- FlexIO_TIMCFG[2] = 0x2202
- FlexIO_TIMCMP[2] = 0x0f04 // set baud rate equal to master
- FlexIO_SHIFTCTL[1] = 0x2030502 // select pin5 as Tx
- FlexIO_SHIFTCFG[1] = 2
- FlexIO_SHIFTCTL[2] = 0x2030602 // select pin6 as Fss
- FlexIO_SHIFTCFG[2] = 3
- FlexIO_SHIFTCTL[3] = 0x2800401 //select pin4 as Rx
- FlexIO_SHIFTCFG[3] = 2

The output waveform is shown in the following figure:

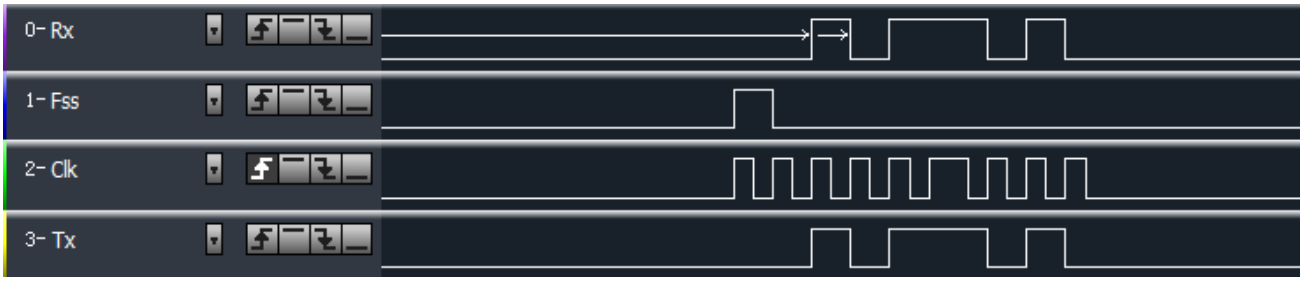

Figure 3. Single frame format of master

Use the timers (n) to generate the SSI (SSInClk) clock, and timer (n+1) to count one clock period to get one high pulse as SSInFss. Use the shifter(n) to transmit data on each rising edge of SSInClk, and the shifter (n+1) to receive data on each falling edge of SSInClk.

The master configuration diagram is shown in the following figure:

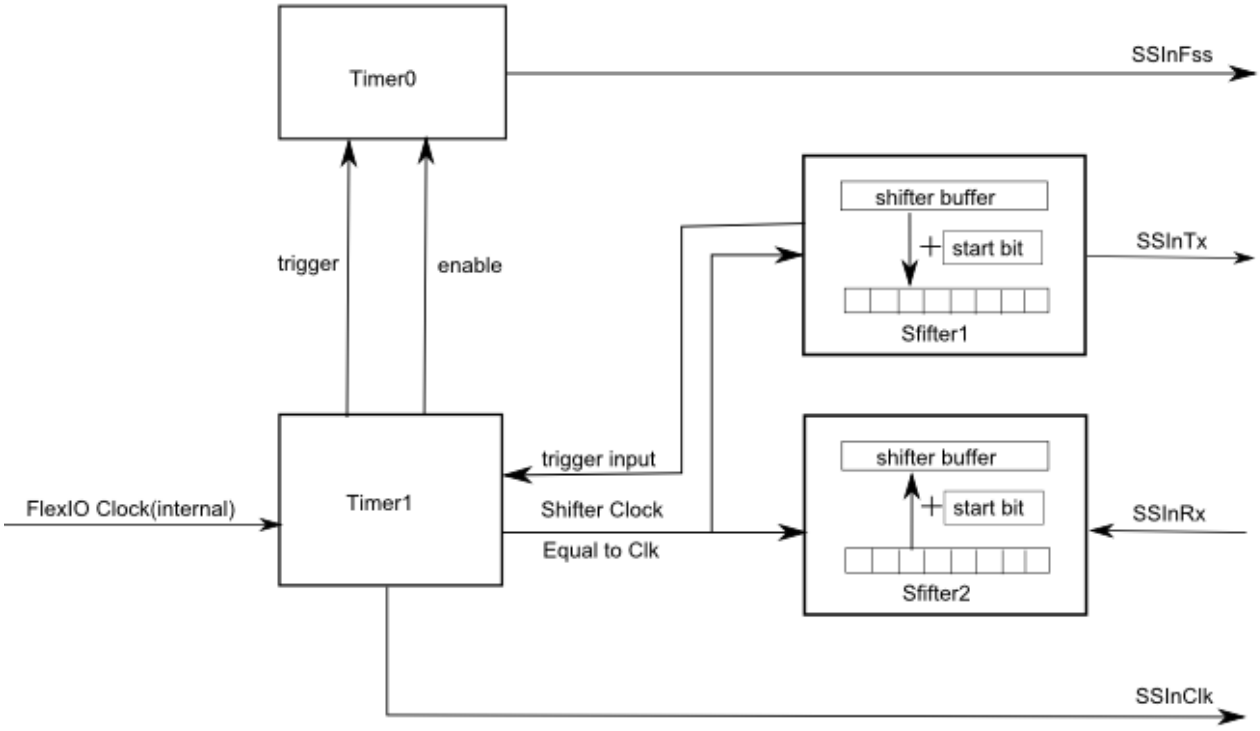

Figure 4. Master configuration (b)

The detailed configuration is as follows:

- Configure Timer (n) triggered by shifter (n) status flag, and configure as dual 8-bit counters baud/bit mode.
- Configure Timer (n) enable on trigger high and disable on compare, enable start bit and decrement counter on FlexIO clock, Shift clock equals Timer output.
- Configure shifter (n) triggered by timer (n), shift on the rising edge of shifter clock, enable transmit mode.
- Configure shifter (n) transmitter outputs start bit with value 0.
- Configure timer (n+1) triggered by timer (n), configure as single 16-bit counter mode.
- Configure timer (n+1) enable on timer (n) enable, and disable on compare, decrement counter on trigger input (both edges).
- Configure timer (n+1) output is logic one when enabled and is not affected by timer reset.
- Configure shifter (n+1) triggered by timer (n), shift on the falling of shifter clock, enable receive mode.
- Configure shifter (n+1) receiver/match store sets error flag if start bit is not 0

The detailed register configuration is as follows:

- FlexIO_TIMCTL[1] = 0x5C30701 //configure pin7 as clk
- FlexIO_TIMCFG[1] = 0x2202
- FlexIO_TIMCMP[1] = 0x0f04
- FlexIO_TIMCTL[2] = 0xb430603 //configure pin6 as Fss
- FlexIO_TIMCFG[2] = 0x102100
- FlexIO_TIMCMP[2] = 0x2
- FlexIO_SHIFTCTL[1] = 0x2030502 //configure pin5 as Tx
- FlexIO_SHIFTCFG[1] = 2
- FlexIO_SHIFTCTL[2] = 0x280401 //configure pin4 as Rx
- FlexIO_SHIFTCFG[2] = 2

The output waveform is the same as that in Figure 3.

3.2. Slave configuration

The slave device transmits data on the rising edge of the clock and latches data on the falling edge of the clock after FSS assert. It uses one timer triggered by FSS, and the second timer to count the expected value of the bit clock and generate the disable signal. The disable signal stores data for the shifter buffer, and takes two shifters for transmitting and receiving data. Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The following diagram show how it is configured:

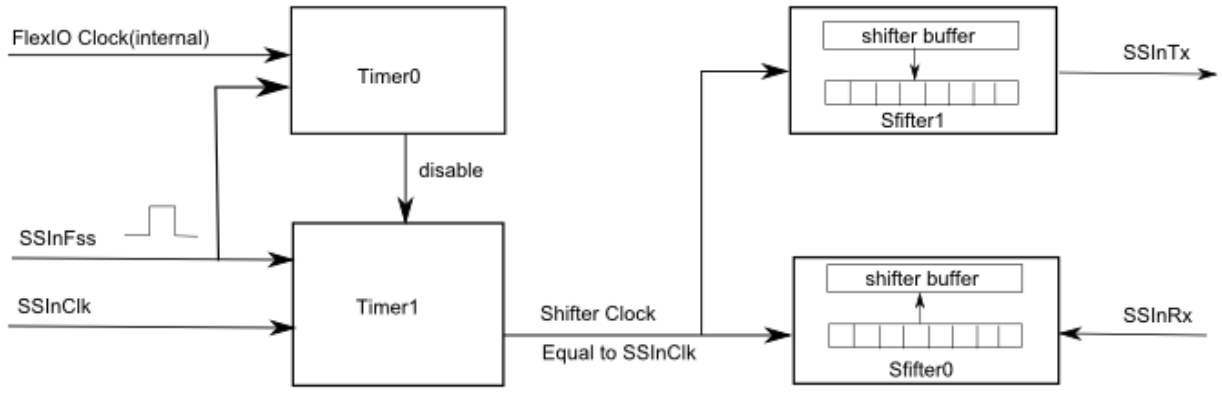

Figure 5. Slave configuration diagram

NOTE

One limitation for this solution is that it asks the timer0 to configure the same baud rate as the master, and does not support any clock stretch for SSnClk, otherwise it may not achieve the correct result.

The detailed register configuration is as follows:

- Configure Timer (n) triggered by the falling edge of SSInFss (invert SSInFss, and enable timer (n) on rising edge of trigger, disable timer(n) on compared).
- Configure TIMOD of Timer (n) is “Dual 8-bit counters baud/bit mode.” and set baud rate equal to master’s speed and number bits is 8-bit or others, which is up to master bit format.
- Configure Timer (n) decrement counter on FlexIO clock,
- Configure Timer (n+1) triggered by the falling edge of SSInFss (invert SSInFss, and enable timer (n+1) on rising edge of trigger, disable timer (n+1) on timer (n) disable.
- Configure TIMOD of Timer (n+1) is “single 16-bit counter mode” and set TIMCMP register to 0xF, or other value, which equal to expected bit number multiple 2 and decrease 1.
- Configure Timer (n+1) decrement counter on Pin input (both edges), Shift clock equals Pin input.
- Configure shifter (n) triggered by timer (n+1) and enable shifter on negedge of shifter clock, and enable receive mode.
- Configure shifter (n+1) triggered by timer (n+1) and enable shifter on rising edge of shifter clock, and enable transmit mode.

The detailed Timer and shifter configuration is as follows:

- FlexIO_TIMCTL[0] = 0xCC00081 // select pin6 as Fss
- FlexIO_TIMCFG[0] = 0x1002600
- FlexIO_TIMCMP[0] = 0x0f04
- FlexIO_TIMCTL[1] = 0xCC00703 //select pin7 as clk input, select pin6 as Fss
- FlexIO_TIMCFG[1] = 0x201600
- FlexIO_TIMCMP[1] = 0x0f
- FlexIO_SHIFTCTL[0] = 0x1800501 // select pin5 as Rx
- FlexIO_SHIFTCFG[0] = 0
- FlexIO_SHIFTCTL[1] = 0x1030402 // select pin4 as Tx
- FlexIO_SHIFTCFG[1] = 0

4. Performance

DMA support is provided to reduce the CPU loading and improve the throughput performance, and it is available to trigger a DMA request by shifter status flag. One example of enabling a DMA request of shifter0 and shifter1 is as follows:

- FlexIO->SHIFTSDEN |= 0x03 // enable shifter0 and shifter1 status trigger DMA request.

If the FlexIO clock is approximately 48 MHz, then the SSI clock can be up to 12 MHz in the master device, and 8 MHz in the slave device.

5. References

1. KL27P64M48SF2RM Reference Manual (document KL27P64M48SF2RM)
2. The Introduction of a Fallible FlexIO Configuration (document AN5396)

6. Revision History

Table 1. Revision history

Revision number	Date	Substantive changes
0	02/2017	Initial release

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: AN5397
Rev. 0
02/2017