Interfacing the TLC4541 to TMS320C6711 DSP

Lijoy Philipose *Data Acquisition Applications*

ABSTRACT

This application note presents a method for interfacing the TLC4541 16-bit SAR analog-to-digital converter to the TMS320C6711™ DSP McBSP1. The software developed reads 1024 samples continuously from the TLC4541. In an effort to reduce development time, the source code is made available on http://www.ti.com/lit/zip/SLAA156.

Contents

List of Figures

1 Introduction

The TLC4541 is a single-channel, 16–bit upgrade for the TLV2541 (12-bit) and TLC3541 (14–bit) analog-to-digital converters. The converter is able to gluelessly interface to the TMS320C6711 digital signal processor (C6711™ DSP). For system development the TMS320C6711 DSP starter kit (DSK) and TLC4541EVM was used. Software code, written for this note is available for download from TI's website.

2 Hardware

The TMS320C6711 DSK is an easy medium for interfacing the TMS320C6711 DSP to the TLC4541. The TLC4541EVM plugs onto the C6711 DSK through two SAMTEC connectors.

2.1 TMS320C6711 DSK

The TMS320C6711 DSP starter kit (DSK) not only provides an introduction to C6000™ DSP platform technology, but is powerful enough to use for fast development of networking, communications, imaging and other applications like data acquisition. See TI's website (www.ti.com) for more information on the C6711 DSK.

3 TLC4541EVM

The TLC4541EVM is a member of the multipurpose (MP) family of serial EVMs available from Texas Instruments. It provides a platform to demonstrate the functionality of the TLC4541 ADC and the TLV5636 DAC. For more information on the EVM, search for SLAU081 on TI's website.

3.1 Hardware Interface

The hardware interface is seamless between the C6711 DSK and the TLC4541. The C6711 DSP provides two McBSPs—this report uses McBSP1. The hardware connections are shown in Figure 1. The SCLK, FS, SDO pins from the TLC4541 are connected to CLKR1, FSR1, and DRR1 pins of McBSP1 respectively. The chip select ($\overline{CS}$) pin is grounded because only one A/D converter is placed on the port. If more than one device is on the bus, then chip select can not be grounded.

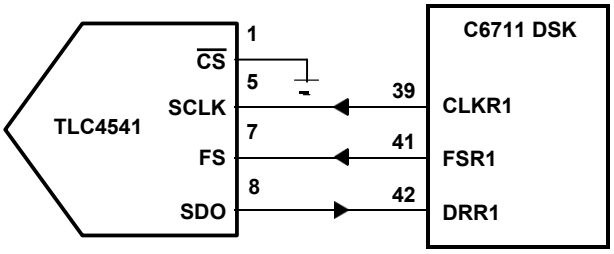

Figure 1. Hardware Interface Block Diagram

4 Software Interface

All of the software was written and compiled using Code Composer Studio. The most involved portion of writing the code for this simple interface is programming the multichannel buffered serial port (McBSP).

4.1 McBSP Setting

The GUI interface of the configuration tool (i.e., CDB file); DSP/BIOS and CSL has made it easier than ever to write programs and set up the multichannel buffered serial port. To see how easy it is to set up the McBSP registers, double click on the *.cdb* file (for our case TLC4541.CDB) in the project window, and find the McBSP configuration manger under CSL. Right click on mcbspCfg1 and select *properties*. There we see all the registers as tabs, and individual bit field settings as pulldown options. Once all options have been selected, click OK. You can confirm the registers are set correctly by opening the file tlc4541cfc_c.c. The GUI-generated code for this application is shown below.

```
MCBSP_Config mcbspCfg1 = {
    0x01000000,         /*  Serial Port Control Reg. (SPCR)    */
    0x00010040,         /*  Receiver Control Reg. (RCR)    */
    0x00010000,         /*  Transmitter Control Reg. (XCR)    */
    0x204A0004,         /*  Sample-Rate Generator Reg. (SRGR)    */
    0x00000000,         /*  Multichannel Control Reg. (MCR)    */
    0x00000000,         /*  Receiver Channel Enable(RCER)    */
    0x00000000,         /*  Transmitter Channel Enable(XCER)    */
    0x00000500          /*  Pin Control Reg. (PCR)    */
};
```

The McBSP is programmed as a serial port, in nonclock stop mode (or DSP mode). Frame sync and serial clock signals are output pins. The receiver is set for 16-bit transfers with one bit delay on data. Frame sync (FSR1) is generated by the sample rate generator. The TLC4541 is running at full speed, which means the serial clock is at 15 MHz and frame sync frequency is at 200 kHz.

The C6711 DSK clocks the C6711 DSP at 150 MHz. The sample rate generator clock source is half the CPU clock, or 75 MHz. The 15 MHz clock on CLKR was achieved by setting CLKGDV bit field in the sample rate generator register to 4. The formula for calculating the serial clock is given below as equation 1.

$$CLK(X/R) = \frac{\left(\frac{CPU\ clock}{2}\right)}{(CLKGDV + 1)} \quad (1)$$

We know that each clock is 67 ns, so we can calculate that triggering a frame-sync pulse every 75 serial clock cycles gives a sample rate of 200 kHz. The frame period (FPER) field, in the sample rate generator register, is where we wrote 74.

4.2 Software Flow

The software presented in this application report reads 1024 samples at 200 kHz continuously. As selected in the configuration tool, all the register and peripheral programming is done during initialization. DSP/BIOS preinitializes all the McBSP registers and other DSP registers before arriving in the main function. As a result, the main function simply enables the interrupt service routine and McBSP1—from then on the DSP/BIOS and McBSP receive ISR do all the work. When a McBSP1 receive interrupt occurs, *McBSP1Rcv_ISR* reads the port and stores the data in ad_buffer. When the buffer is full, it resets the index, i, to the beginning and flushes the receive buffer.

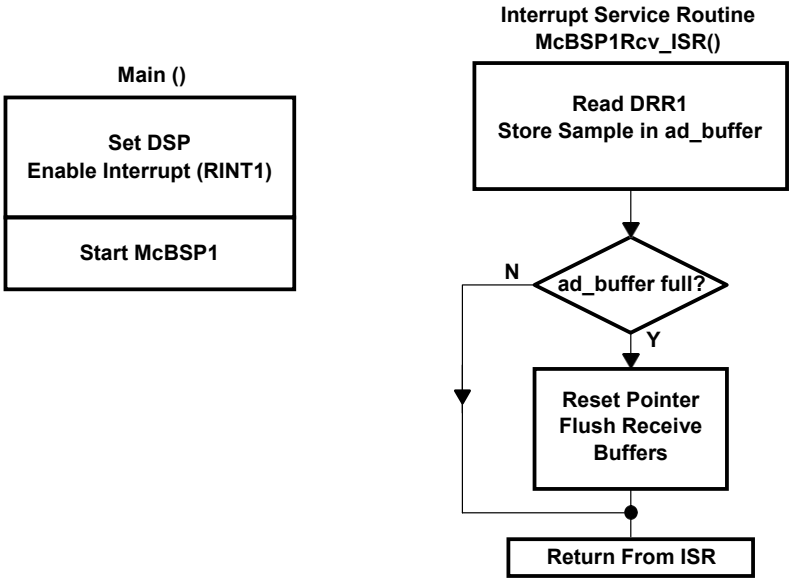

Figure 2. Software Flowchart

5 References

1. TLC4541 data sheet (SLAS293)
2. TMS320C6711 data sheet (SPRS088)
3. TMS320C6000 DSP/BIOS User's Guide (SPRU303)
4. TMS320C6000 Code Composer Studio Tutorial (SPRU301)

Appendix A MAIN.C

```
/*********************************************************/
/* FILENAME: main.c                                      */
/* DESCRIPTION: This program uses McBSP1 to read 1024    */
/* samples continuously from the TLV4541 16-bit 200KSPS  */
/* Analog-to-Digital Converter.  The samples are stored in  */
/*    the buffer called ad_buffer.                       */
/* AUTHOR   : DAP Application Group, L. Philipose, Dallas   */
/*    CREATED 2002(C) BY TEXAS INSTRUMENTS INCORPORATED.    */
/* VERSION: 1.0                                          */
/*********************************************************/

/* Header file */
#include "csl.h"
#include "csl_irq.h"
#include "csl_mcbsp.h"

/*Declarations*/
#define BLOCK_SZ 1024 /* size of data buffer */

/*DSP/BIOS variables*/
extern far MCBSP_Handle hMcbsp1;

/*Global variables*/
unsigned short ad_buffer[BLOCK_SZ], i=0;

/*************************************************************************\
* Function: main()
* Description: Enables McBSP1 receive interrupt and McBSP1
\*************************************************************************/

void main() {

   IRQ_enable(IRQ_EVT_RINT1);
   MCBSP_start(hMcbsp1, MCBSP_RCV_START | MCBSP_SRGR_START| MCBSP_SRGR_FRAMESYNC,
0);

}

/*************************************************************************\
* Function: McBSP1Rcv_ISR()
* Description: McBSP1 Receive Interrupt Service Routine.  Reads sample out of
*  receive register and stores it in array call ad_buffer.  When the buffer
*  table is full, it resets the pointer to the beginning and flushing receive
*  buffers.
\*************************************************************************/

void McBSP1Rcv_ISR(void)
{
    /* wait until a value is received then read it */
```

```
    while (!MCBSP_rrdy(hMcbsp1));
    ad_buffer[i++] = MCBSP_read(hMcbsp1);

   if (i >= BLOCK_SZ )
   {
      i=0;
      MCBSP_read(hMcbsp1);   /*flush receive register*/
      MCBSP_read(hMcbsp1);   /*flush receive register*/
   }

}

/*****************************************************************************\
* End of main.c
\*****************************************************************************/
```

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265