Interfacing the ADS8361 to the TMS320F2812 DSP

Tom Hendrick, Miroslav Oljaca *Data Acquisition Products*

ABSTRACT

This application note presents several methods for interfacing the ADS8361 16-bit SAR analog-to-digital converter to the TMS320F2812 DSP. The software developed for this application note uses both the McBSP and SPI port to highlight the flexible digital interface of the ADS8361. In an effort to reduce development time, the source code for this application note can be found on the Texas Instruments web site at http://www.ti.com/lit/zip/SLAA167. Search for device number ADS8361 from the main page and follow the links to this application note.

Contents

1 Introduction 2
2 Hardware 2
2.1 eZdsp™ F2812 2
2.2 ADS8361EVM 2
3 Hardware Interfaces 2
3.1 Communication via McBSP 2
3.1.1 Mode II Using McBSP1 3
3.1.2 Controlling M0 and M1 5
3.1.3 Mode IV Using McBSP 5
3.2 Communication via McBSP and SPI 6
3.2.1 Mode I Using McBSP and SPI 6
3.2.2 Mode III Using McBSP and SPI 8
3.3 Communication via SPI 8
3.3.1 Mode II and Mode IV Using SPI1 in Master Mode 8
4 Software Interface 9
4.1 McBSP Setting for Mode II and Mode IV 9
4.2 McBSP and SPI Settings for Mode I and Mode III 9
4.3 SPI Settings for Mode II and Mode IV 9
4.4 Software Flow 10
5 References 10

Figures

Figure 1. Hardware Connections for Mode II and Mode IV via McBSP 3
Figure 2. Mode II Waveforms Using McBSP 4
Figure 3. Expanded Mode II Waveform 4
Figure 4. Synchronizing Data in Mode IV Output Using M0/M1 pins 5
Figure 5. Hardware Connections for Mode I and Mode III via McBSP and SPI (Slave) 6
Figure 6. Mode I Waveforms Using McBSP and SPI 7
Figure 7. Mode III Waveforms Using McBSP and SPI 7
Figure 8. Hardware Connections for Mode II and Mode IV via SPI (Master) 8
Figure 9. Software Flow Chart 10

1 Introduction

The ADS8361 is a 2+2 channel, 16-bit, pin-compatible upgrade to the ADS7861 (12-bit) 2+2 channel analog-to-digital converter. The ADS8361 can gluelessly interface to the TMS320F2812 digital signal processor (F2812 DSP). Hardware used in this example includes the eZdsp™ F2812 for TMS320F2812 from Spectrum Digital Incorporated and the ADS8361 EVM. More information on the eZdsp™ F2812, including ordering information, can be found on the Spectrum Digital website.

Software code written for this application note, as well as additional information on the ADS8361EVM, is available for download from the Texas Instruments website.

2 Hardware

The combination of the eZdsp™ F2812 and the ADS78_8361EVM is a convenient way to experiment with interfacing the TMS320F2812 DSP to the ADS8361.

2.1 eZdsp™ F2812

The eZdsp™ F2812 not only provides an introduction to F2000 DSP platform technology, but also is powerful enough to use for fast development of optical networking, communications, digital motor control, and other applications like data acquisition. See Spectrum Digital's website (http://www.spectrumdigital.com) for more information on the TMS320F2812 development tools.

2.2 ADS8361EVM

The ADS8361 is a member of the Motor Control Products family of serial ADCs available from Texas Instruments. The EVM provides a platform to demonstrate the functionality of the ADS8361 ADC with various Texas Instruments DSPs and microcontrollers, while allowing easy access to all analog and digital signals for customized end-user applications. For more information on the EVM, search for document number SLAU094 from the main page of the Texas Instruments website at http://www.ti.com.

3 Hardware Interfaces

The TMS320F2812 features a multichannel buffered serial port (McBSP), and a serial peripheral interface (SPI) for communication with serial devices such as the ADS8361. Basic methodologies for connecting the ADS8361 to both serial interfaces are presented in the following sections. The hardware interfaces presented in this application note show resistors connected between the DSP and the ADS8361. The resistors are shown as a reminder that high-speed signals can cause unwanted ringing in digital systems. This ringing ultimately degrades system performance. Factors such as clock speed, trace length, and physical PCB layout determine if these resistors are required in a particular design.

3.1 Communication via McBSP

Since the F2812 device has only one McBSP port, communication with both A and B serial outputs of the ADS8361 can be difficult to accomplish without additional glue logic. For this application note, high-speed communication with the ADS8361 via the McBSP only, is limited to modes II and IV.

3.1.1 Mode II Using McBSP1

The 3.3 V digital interface of the ADS8361 allows for a seamless connection between the eZdsp™ F2812 and the ADS78_8361 EVM. The hardware connections to McBSP1 are shown in Figure 1. The CLOCK, (RD+ CONVST), and SDA pins from the ADS8361 are connected to CLKX1, FSX1, and DRR1 pins of the McBSP respectively. The chip select (CS) pin is grounded because only one A/D converter is communicating with the serial port. If more than one device is on the bus, then chip select should be controlled by any available GPIO on the F2812 DSP.

For Mode II operation, transmitting an appropriate data stream via the DX pin of the McBSP to the ADS8361 A0 pin can control channel 0/1 selection. The sample code associated with this application note provides an example of channel selection, additional information can also be found later in this document.

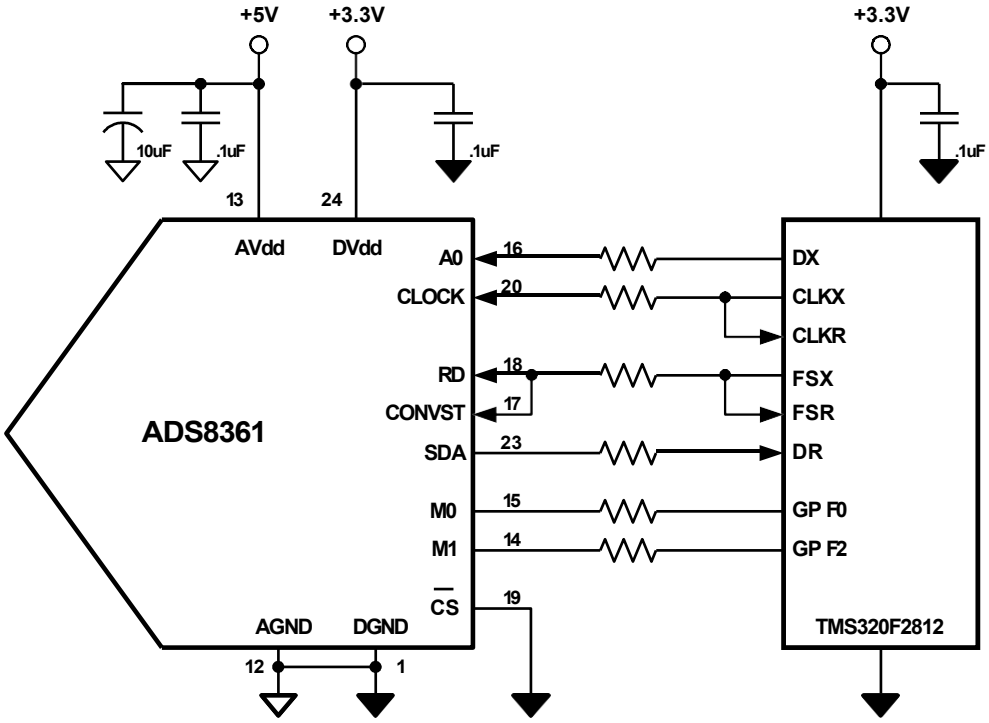

Figure 1. Hardware Connections for Mode II and Mode IV via McBSP

As shown in Figure 1, the frame sync transmit (FSX) pulse from the McBSP port is used to control the start of the conversion cycle. FSX high is synchronous with the rising edge of the conversion clock. FSX is programmed as an active high pulse, 3 clock cycles wide, to ensure the conversion begins on the first rising edge of the conversion clock.

Figures 2 and 3 provide example waveforms using the hardware and code examples mentioned previously. The received data is delayed by 1 bit to accommodate the CONVST validation timing. The entire cycle repeats every 21 clock periods.

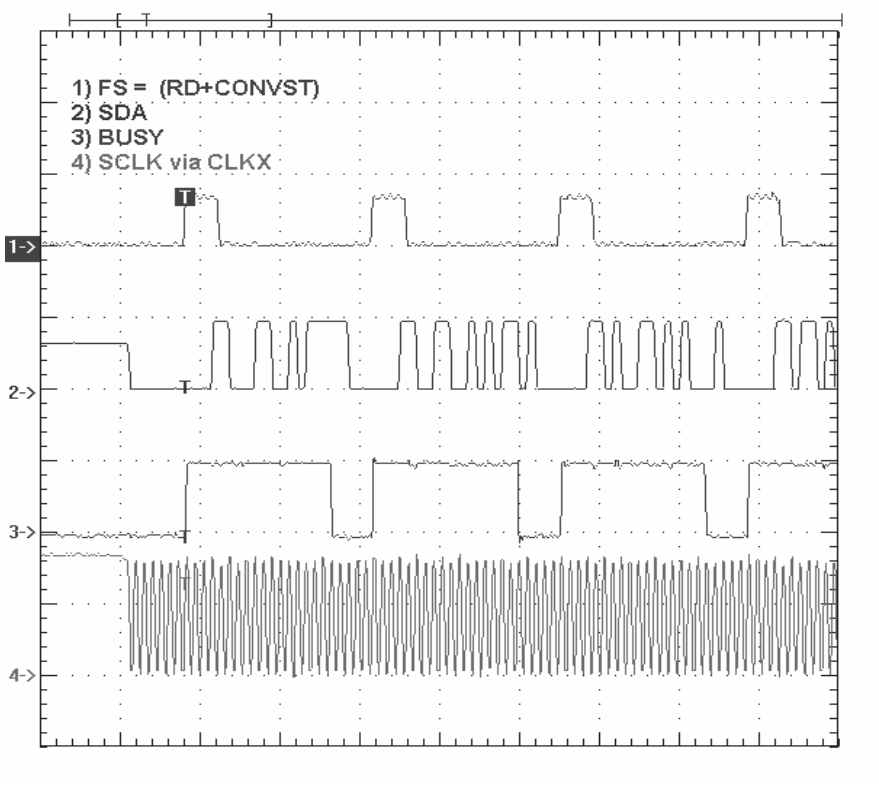

Figure 2. Mode II Waveforms Using McBSP

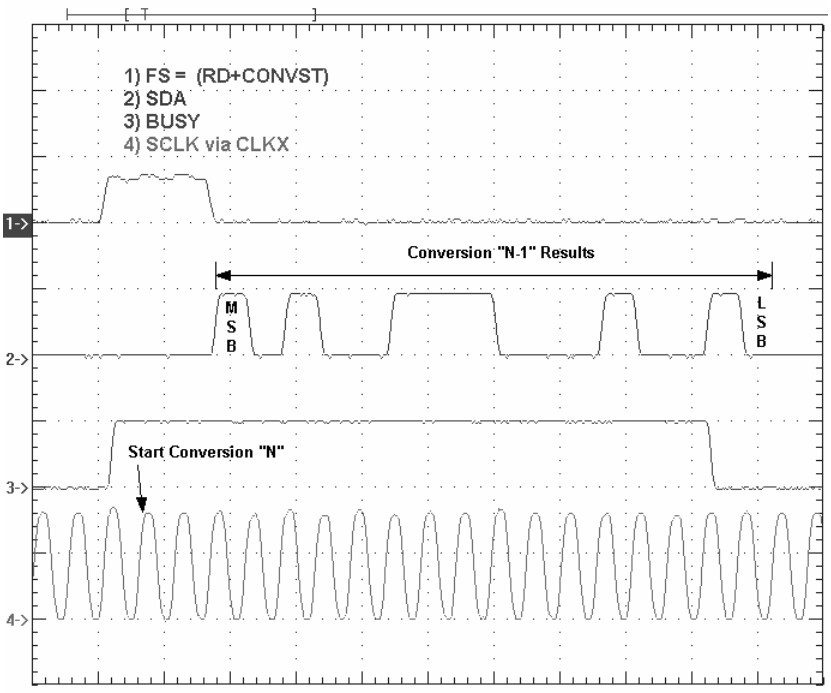

Figure 3. Expanded Mode II Waveform

3.1.2 Controlling M0 and M1

Since the channel information is presented in the output serial stream of the ADS8361, the M0 and M1 lines could be set to fixed states. While this keeps the interface simple, the routine to decode the channel information can cause unwanted software overhead.

As an alternative, the M0 and M1 lines can be controlled through GPIO functions. By controlling M0 and M1 through GPIO, it is possible to initialize the ADS8361 in a simple software loop so that data is presented to the serial output sequentially, starting with channel A0.

3.1.3 Mode IV Using McBSP

The code example written for this application note controls the M0 and M1 pins through GPIO. The conversion results are then stored in a simple two-dimension array. Data storage begins with the second CONVST cycle after entering Mode IV, as shown in Figure 4.

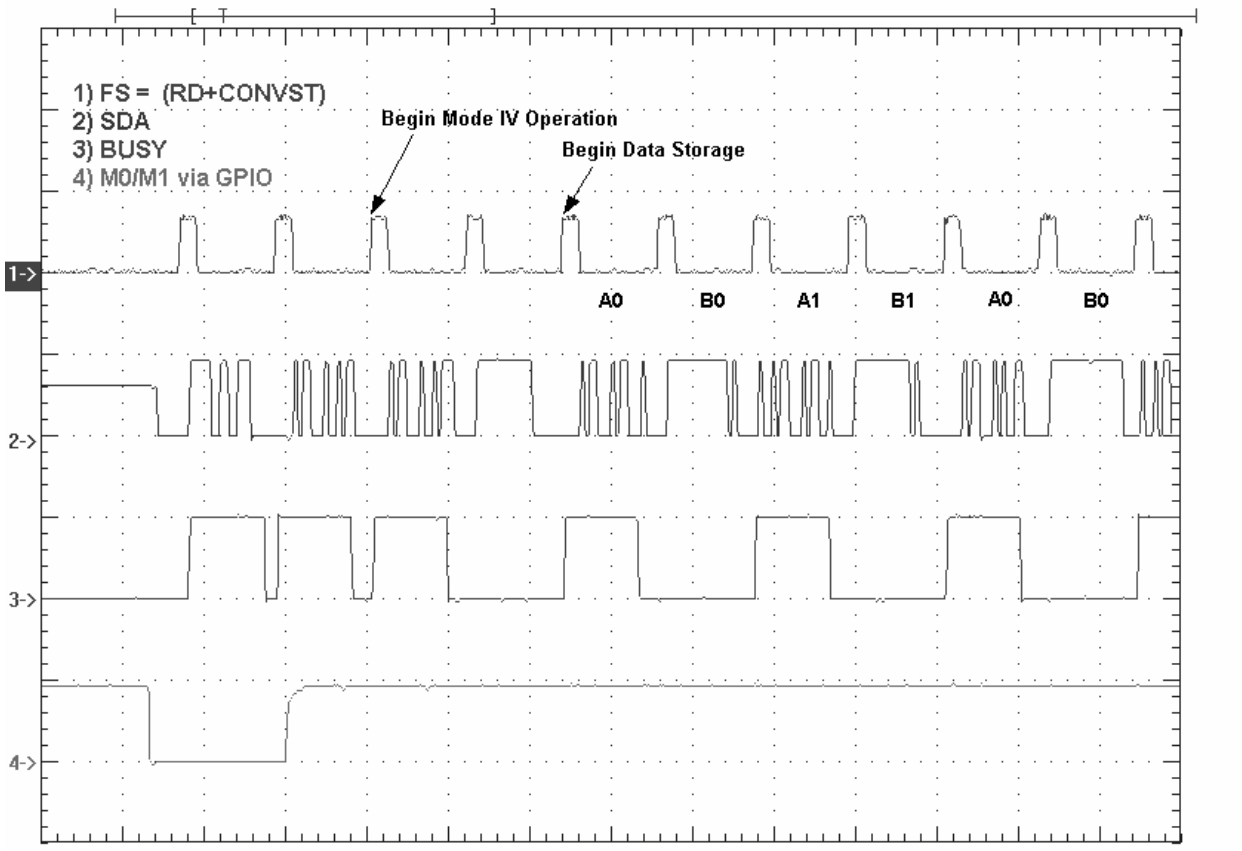

Figure 4. Synchronizing Data in Mode IV Output Using M0/M1 pins

3.2 Communication via McBSP and SPI

Communication with both A and B serial outputs of the ADS8361 can be accomplished by setting the McBSP port for SPI or *clock stop* mode, and configuring the SPI port as a slave device. The McBSP is set to provide the conversion clock, CONVST (via FSX) and to receive data from channel A. The SPI port is then configured as a slave to the McBSP controller, with the SIMO pin accepting data from channel B.

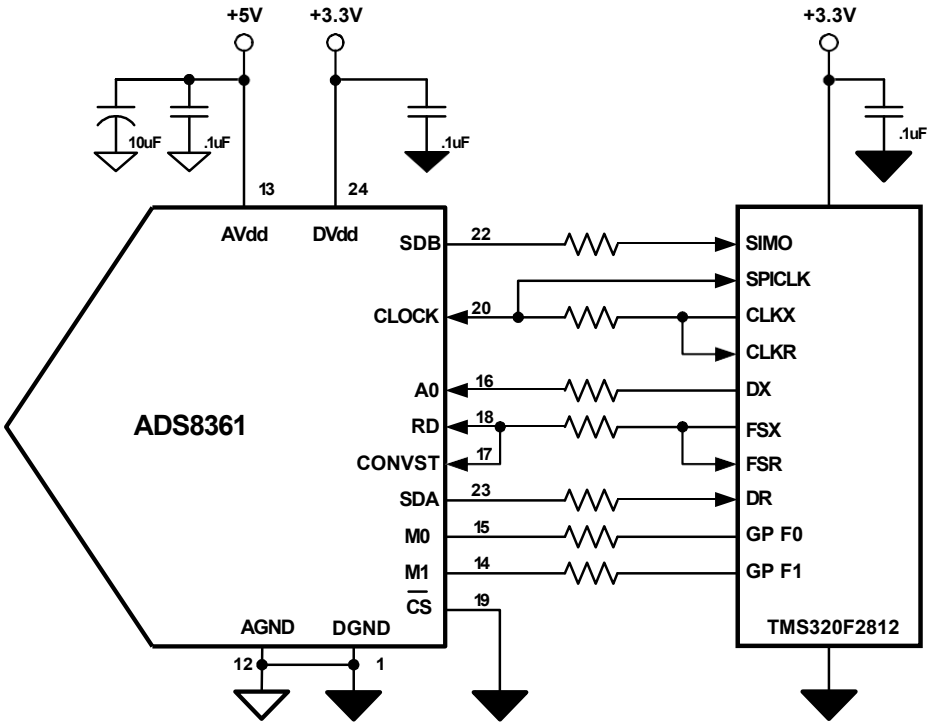

Figure 5. Hardware Connections for Mode I and Mode III via McBSP and SPI (Slave)

3.2.1 Mode I Using McBSP and SPI

When using the ADS8361 with an SPI based communication protocol, it is important to remember that the actual conversion starts on the first rising clock edge *after* CONVST is toggled high. When using the McBSP port in SPI mode, Frame Sync Transmit (FSX) should be programmed as an active high signal (see Figures 6 and 7).

In Mode I, a single channel is selected via the address pin A0. As shown in Figure 5, the A0 line is connected to the data transmit (DX) pin of the DSP. Selecting channel 1 is accomplished by loading the DXR2 register with the hex value 0x8. Channel 0 is selected with the hex value 0x0.

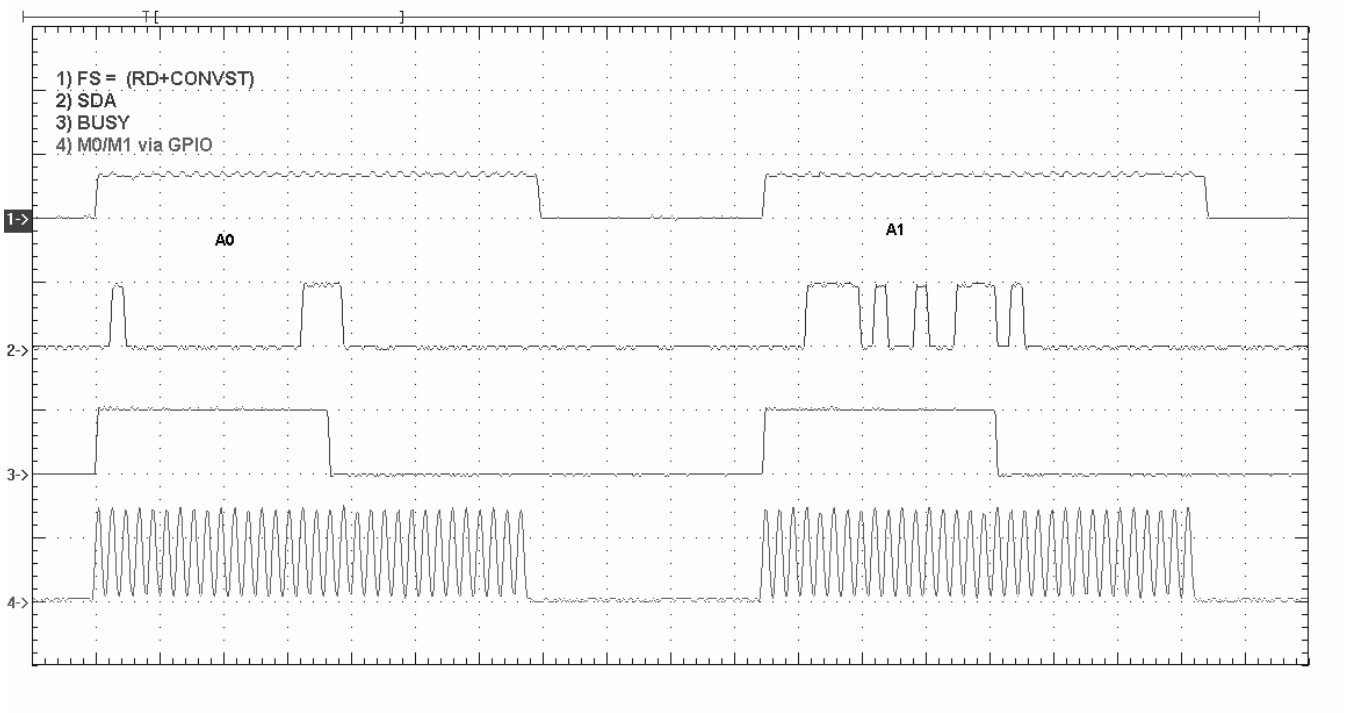

Figure 6. Mode I Waveforms Using McBSP and SPI

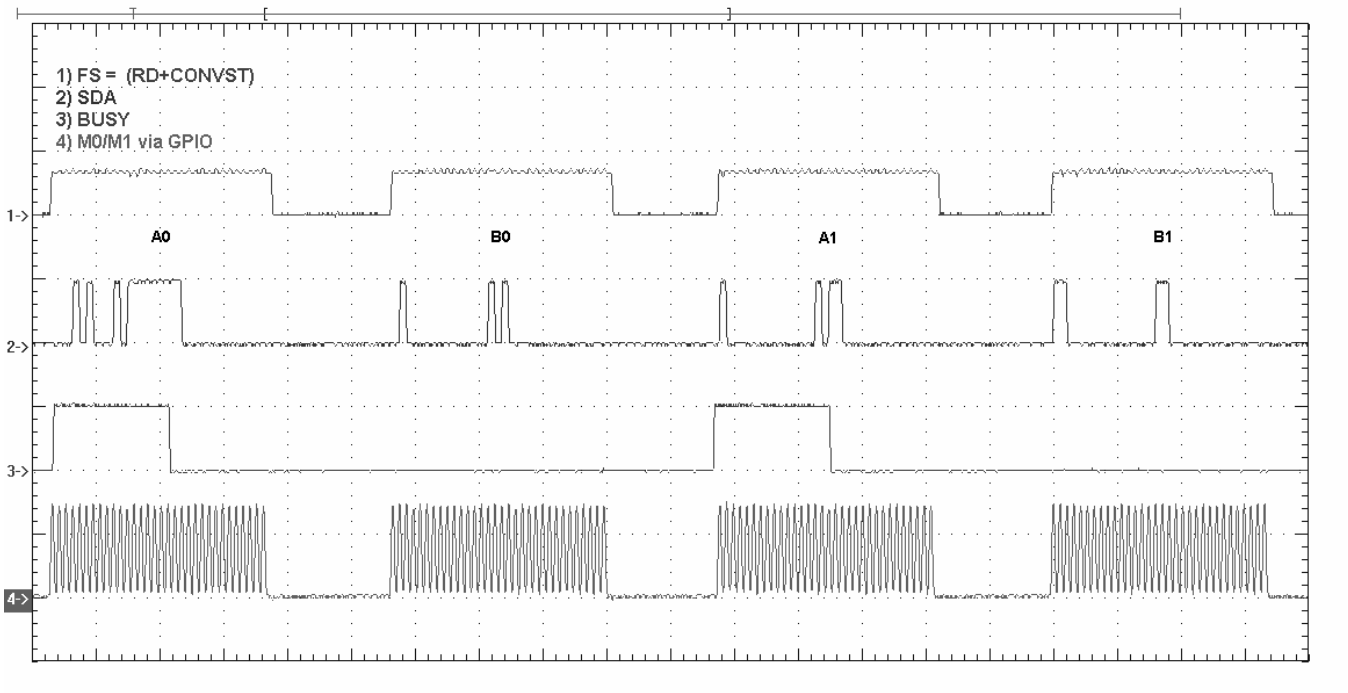

Figure 7. Mode III Waveforms Using McBSP and SPI

3.2.2 Mode III using McBSP and SPI

In Mode III, the A and B outputs are providing both channel 0 and channel 1 data. Channel information can be decoded from the serial data, or by running a simple synchronizing loop. In the code associated with this application note, the process begins by initializing the McBSP clock, running through a synchronizing loop, then enabling the SPI receiver. The return clock to the McBSP port (CLKR) is fed back to the SPI port along with the data from the OUTB pin of the ADS8361. Since this method uses the SPI port in slave mode, the OUTB data is fed to the SIMO pin of the F2812. Please refer to Figures 5 and 7.

3.3 Communication via SPI

For applications where only the SPI port is used, any available GPIO can be used to control the CONVST pin. As mentioned previously, when using the ADS8361 with an SPI based communication protocol, it is important to remember that the actual conversion starts on the first rising clock edge *after* CONVST is toggled high. Through GPIO, CONVST can be toggled high before the start of the SPI transfer, and returned to a low state when the transfer is complete.

3.3.1 Mode II and Mode IV Using SPI1 in Master Mode

The hardware connections to SPI1 are shown in Figure 8. The CLOCK and SDA pins from the ADS8361 are connected to the SPICLK and SOMI pins of SPI1 respectively. As before, the chip select (CS) pin is grounded, because only one A/D converter is placed on the port. With the SPI port, channel selection can be made in a similar fashion as described for the McBSP by connecting A0 of the ADS8361 to the to SIMO pin.

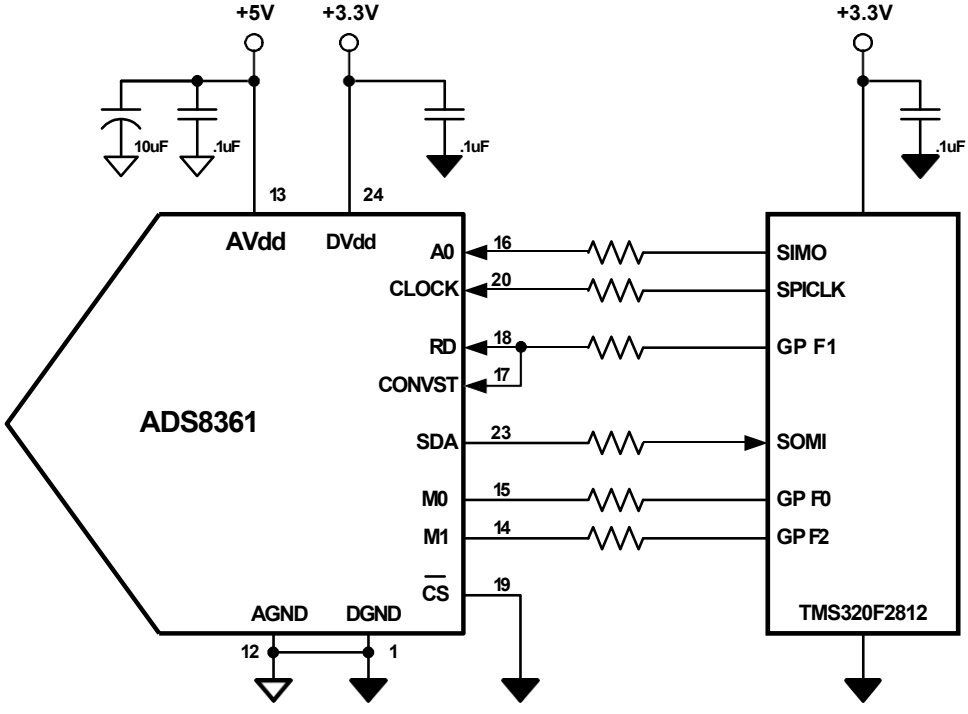

Figure 8. Hardware Connections for Mode II and Mode IV via SPI (Master)

4 Software Interface

All of the software was written and compiled using Code Composer Studio™ version 2.12 for the C2000 family of processors. The following sections describe the general setup of the McBSP port for Mode II and Mode IV operation, as well as the setup of *clock stop* mode for use with both serial outputs from the ADS8361 in Modes I and III. Setup of the SPI port in both master and slave modes, is also discussed.

4.1 McBSP Setting for Mode II and Mode IV

The McBSP is programmed as a serial port, in non-stop clock mode (or DSP mode). Frame sync and serial clock signals are output pins. The transmitter and receiver are set for 20-bit transfers with one-bit delay on data receive and transmit. The frame sync (FSX1) is generated by the sample rate generator and is used for both the RD and CONVST signal on the ADS8361 by jumper W2 on the EVM. Frame Sync is held high through 3 clock cycles.

In the sample code, the ADS8361 is running with a serial clock of 9.4MHz. The 9.4 MHz clock on CLKX was achieved by setting CLKGDV bit field in the sample rate generator register to 3.

The data received in the McBSP receive registers must be modified slightly in order to strip out the address and channel bits that are transferred as the two MSB's in the output serial stream from the ADS8361. The following lines of code perform this function:

```
{
mcbsp_xmit(var1);
    while(McbspaRegs.MFFRX.bit.ST==0) { }   // Check for McBSP FIFO receive ready
    ADC_Data[idx] = McbspaRegs.DRR1.all<<2; // Strip 1st 2 MSB's from Serial Stream
 }
```

4.2 McBSP and SPI Settings for Mode I and Mode III

The McBSP is programmed as a serial port, in clock stop mode (or SPI mode). Frame sync and serial clock signals are output pins. The transmitter and receiver are set for 32-bit transfers with one-bit delay on data transmit and receive. The frame sync (FSX1) is generated by the sample rate generator and is used for both the RD and CONVST signal on the ADS8361 by jumper W2 on the EVM.

The SPI port is set as a slave device with serial data from OUTB clocked on the rising edge of the McBSP receiver clock. CLKR and OUTB are tied back to the SPI port as SPICLK and SIMO respectively. The data received in the McBSP and SPI receive registers needs to be modified as mentioned previously.

4.3 SPI Settings for Mode II and Mode IV

The SPI port is set as a master device with serial data from OUTB clocked on the rising edge, read on the falling edge, of the SPI clock. The ADS8361's OUTA and A0 pins are connected SOMI and SIMO respectively. GPIO functions are used to trigger CONVST. CONVST and RD are connected via jumper W2 on the EVM. The data received in the SPI receive registers needs to be modified as mentioned previously.

4.4 Software Flow

The software presented in this application report reads 512 samples from one, two or four channels. The code is set up in six project files, with six workspaces. The workspaces bring up various waveform viewing windows showing the converted data from the ADS8361.

The McBSP, SPI and other DSP registers are configured appropriately before arriving in the main function. As a result, the main function simply enters a *forever* loop, which samples an array of 512 data elements. When the data array is full, it posts the conversion results to the graph in the project manager window and resets the sample index. The program then reenters the sample loop. Figure 9 shows an example flowchart for Mode I operation using the McBSP port.

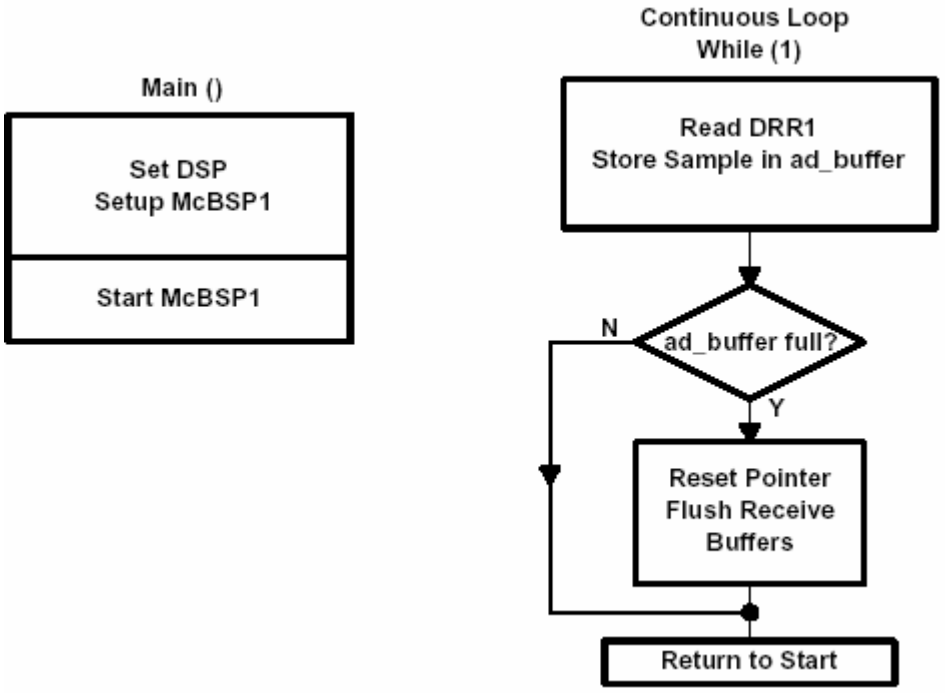

Figure 9. Software Flow Chart

5 References

1. ADS8361 data sheet (SBAS230)
2. TMS320F2812 data sheet (SPRS088)
3. TMS320C28x DSP CPU and Instruction Set Reference Guide (SPRU430)
4. TMS320F28x DSP Peripherals Reference Guide (SPRU566)
5. TMS320F28x Multichannel Buffered Serial Port (McBSP) Peripheral Reference Guide (SPRU061)
6. C28x Peripheral Examples in C (SPRC097)

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