PSIM

TUTORIAL
Using SPI in F2833X/F2803X Target

October 2016

With the SimCoder Module and the F2833x/F2803x Hardware Target, PSIM can generate ready-to-run codes for DSP boards that use TI F2833x/F2803x series DSP. By using the Serial Peripheral Interface (SPI) blocks in the F2833x/F2803x Target library, one can implement the functions to communicate with external SPI devices (such as external A/D and D/A converters) easily and conveniently. Writing code manually for SPI devices is often a time-consuming and non-trivial task. With the capability to support SPI, PSIM greatly simplifies and speeds up the coding and hardware implementation process.

This tutorial describes how SPI blocks and defined and used in PSIM. To illustrate the process, several examples with SPI D/A converter and A/D converters are provided.

1. SPI in F2833x/F2803x Target

There are one set of SPI module in TI F2833x DSP and 2 sets of SPI modules (SPIA and SPIB) in F2803x DSP. They have different sets of GPIO ports. PSIM supports the use of these GPIO ports to communicate with SPI devices. The GPIO ports used by SPI modules are listed below.

For the F2833x Target:

To use Ports GPIO16-GPIO19:

- GPIO16 as SPI data output pin
- GPIO17 as SPI data input pin
- GPIO18 as SPI clock SPICLK
- GPIO19 as SPI slave transmit-enable pin SPISTE

To use Ports GPIO54-GPIO57:

- GPIO54 as SPI data output pin
- GPIO55 as SPI data input pin
- GPIO56 as SPI clock SPICLK
- GPIO57 as SPI slave transmit-enable pin SPISTE

For the F2803x Target:

To use Ports GPIO16-GPIO19 of SPIA:

- GPIO16 as SPI data output pin
- GPIO17 as SPI data input pin
- GPIO18 as SPI clock SPICLK
- GPIO19 as SPI slave transmit-enable pin SPISTE

To use Ports GPIO3, 5, 18 and GPIO19 of SPIA:

- GPIO3 as SPI data input pin
- GPIO5 as SPI data output pin
- GPIO18 as SPI clock SPICLK
- GPIO19 as SPI slave transmit-enable pin SPISTE

To use Ports GPIO12-GPIO15 of SPIB:

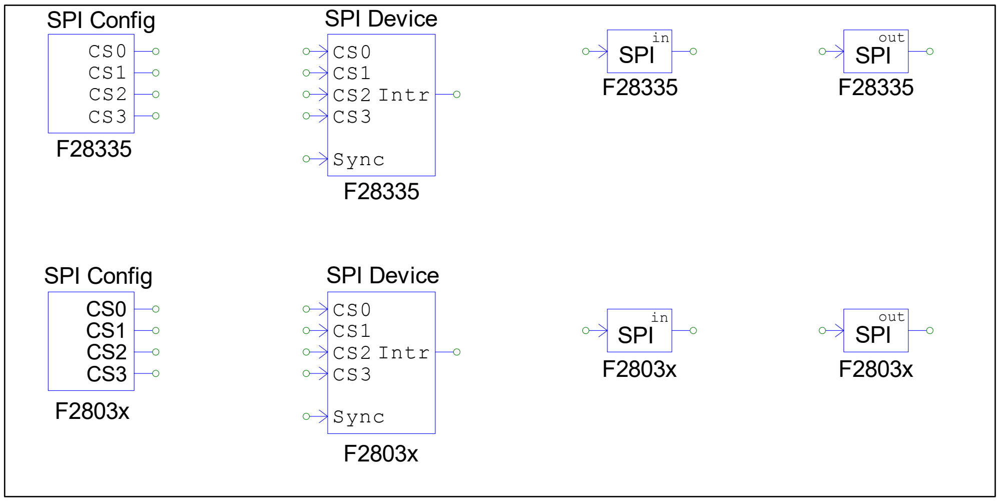
- GPIO12 as SPI data output pin
- GPIO13 as SPI data input pin
- GPIO14 as SPI clock SPICLK

- GPIO15 as SPI slave transmit-enable pin SPISTE

To use Ports GPIO24-GPIO27 of SPIB:

- GPIO24 as SPI data output pin
- GPIO25 as SPI data input pin
- GPIO26 as SPI clock SPICLK
- GPIO27 as SPI slave transmit-enable pin SPISTE

There are four types of SPI library elements in PSIM's F2833x/F2803x Target library: *SPI Configuration*, *SPI Device*, *SPI Input*, and *SPI Output*. Their images are shown below. They can be accessed by going to **Elements** >> **SimCoder** >> **F2833x Target** or **F2803x Target** in PSIM.



The functions and definitions of these elements are described in the following sections.

SPI Configuration

The *SPI Configuration* block defines chip selection pins and the buffer size for the SPI commands. The SPI Configuration block must be present in a schematic if SPI is used, and this element must be in the main schematic.

The parameters of the SPI Configuration block are:

- *SPI Port*: Define the SPI port as either GPIO16-19 or GPIO54-57 for F2833x; or GPIO16-19, GPIO3, 5, 18, 19, GPIO12-15 or GPIO24-27 for F2803x.
- *Chip Select Pin0* to *Pin3*: Both F2833x and F2803x Targets support up to 16 SPI devices, which requires four GPIO pins for chip select as defined by Chip Select Pin0 to Pin3. These GPIO ports and the SPI slave transmit-enable pin SPISTE are used to generate the chip select signal for SPI devices.

  If there is only one SPI device, one can use just the SPI slave transmit-enable pin SPISTE as the chip select signal. If a chip selection pin is not used, set it to *Not Used*.

  In the examples in this tutorial, there are up to three SPI devices. Thus two chip select pins, Pin0 and Pin1, will be sufficient. The other pins, Pin2 and Pin3, are not used.

- *SPI Buffer Size*: Define the buffer size of the SPI commands. Each memory cell of the buffer saves the index of a SPI command. Normally, one can specify the buffer size as 1 plus the number of SPI commands (i.e. Start Conversion Command, Receiving Data Command, Sending Data Command, and Sync. Command) in all SPI Input/Output elements.

SPI Device

A *SPI Device* block defines the information of the corresponding SPI hardware device. The number of SPI Device blocks in the schematic must be same as the number of SPI hardware devices.

The parameters of the SPI Device block are:

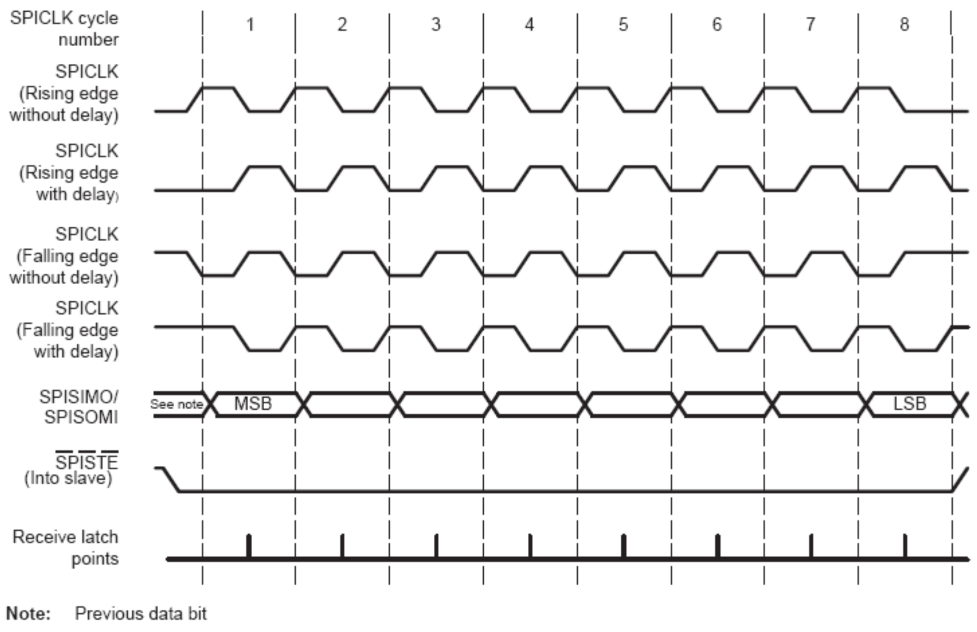
- *Chip Select Pins*: Specify the state of the chip select pins corresponding to the particular SPI device. When the chip select pins are at this state, this SPI device is selected.

  In a schematic, the chip select pins of all the SPI devices are connected to the chip select pins of the SPI Configuration block, without defining how the chip select logic is implemented. In the actual hardware, however, one would need to implement the corresponding chip select logic accordingly.

  In the examples in this tutorial, Pins CS0 and CS1 of external A/D and D/A converters are connected to Pins CS0 and CS1 of the SPI Configuration element.

- *Communication Speed* (*MHz*): Specify the SPI communication speed, in MHz. Note that different SPI devices can have different communication speeds.

- *Clock Type*: F2833x DSP supports four SPI clock types: rising edge without delay, rising edge with delay, falling edge without delay, and falling edge with delay, as shown in the picture below.


SPICLK cycle number
1
2
3
4
5
6
7
8
SPICLK (Rising edge without delay)
SPICLK (Rising edge with delay)
SPICLK (Falling edge without delay)
SPICLK (Falling edge with delay)
SPISIMO/ SPISOMI
See note
MSB
LSB
SPISTE (Into slave)
Receive latch points
Note: Previous data bit

Note that PSIM only supports the situation where the DSP and SPI device latch data at the same rising edge or falling edge of the SPI clock signal SPICLK.

- *Command Word Length*: Define the word length, or the length of the significant bits, of SPI communication commands. It can be from 1 to 16 bits.
- *Sync. Active Mode*: Specify the triggering mode of the synchronization signal of the SPI device. It can be either *Rising edge* or *Falling edge*.

  A SPI device can be either an input device or an output device. For example, an external A/D converter is an input device. Usually DSP will send one or multiple A/D conversion commands to the device, and then set the synchronization signal to start the conversion. The synchronization signal is reset at the next command of the same device.

  A SPI input device using the synchronization signal usually needs an interrupt pin to trigger DSP to enter the interrupt service routine.

  On the other hand, an external D/A converter are an output device. Usually DSP sends one or multiple D/A conversion commands to the device, and then sets the synchronization signal to start the conversion. The synchronization signal is reset at the next command of the same device.
- *SPI Initial Command*: Define the SPI command that initializes the SPI device.
- *Hardware Interrupt Mode*: Specify the type of the interrupt signal that the SPI device generates. This is valid only when the SPI device's interrupt output node is connected to the input of a digital output element. It can be one of the followings: *No hardware interrupt*, *Rising edge*, or *Falling edge*.
- *Interrupt Timing*: Specify how a SPI input device generates interrupt when it completes conversion. It can be one of the following:
  - *No interrupt*: No interrupt is generated. In this case, DSP sends the command to a SPI input device. This device starts the conversion and returns the result in the same command.
  - *Multiple interrupt in series*: Multiple interrupts are generated in series after each conversion. This is for a SPI device that has one A/D conversion unit and multiple input channels. In this case, DSP send the first conversion command, and the SPI device starts the conversion. When the conversion is complete, the SPI device will generate an interrupt. In the interrupt service routine, DSP will send a command to fetch the conversion result, and start a new conversion of another channel of the same SPI input device.
  - *One-time interrupt*: Only one interrupt is generated at the end of the conversion. This is for a SPI device that can perform multiple channel conversions in one request. In this case, DSP sends the command to the SPI input device, and the SPI device completes the conversion of multiple input channels. When all the conversions are complete, the SPI device will generate an interrupt.

- *Command Gaps* (*ns*): Define the gap between two SPI commands, in nsec.
- *Conversion Sequence*: Define the names of the SPI input elements, separated by comma that determine the conversion sequence. Note that this parameter is valid only when the SPI device generates multiple interrupts in series.

A SPI command consists of a series of 16-bit numbers separated by comma. In the 16-bit number, only the lower bits are the significant bits used by the command. For example, if the Command Word Length is 8, Bits 0 to 7 are the command, and Bits 8 to 15 are not used. As an example, "0x12, 7, 0" is a command with 3 command words.

SPI Input

A SPI input device may have multiple input channels. The *SPI Input* block is used to define the properties of an input channel for SPI communication, and one SPI Input block corresponds to one input channel.

The parameters of the SPI Input block are explained below:

- *Device Name*: Define the name of the SPI input device.
- *Start Conversion Command*: Define the start conversion command, in hex numbers, separated by comma (for example, 0x23, 0x43, 0x00).
- *Receiving Data Command*: Define the receiving data command, in hex numbers, separated by comma (for example, 0x23, 0x43, 0x00).
- *Data Bit Position*: Define where the data bits are in the receiving data string. The format is:

  ElementName={Xn[MSB..LSB]}

  where
  - *ElementName* is the name of the SPI input element. If it is the current SPI input element, use *y* instead.
  - {} means that the item in the bracket repeats multiple times.
  - *Xn* is the n_{th} word received from the SPI input device, and n starts from 0.
  - *MSB..LSB* defines the position of the significant bits in the word.

  This formula defines the data length of a SPI input device. For example, y=x1[3..0]x2[7..0], means that the data length is 12, and the result is the lower 4 bits of the 2nd word and the lower 8 bits of the 3rd word. If the received data string is 0x12, 0x78, 0xAF, then the result will be 0x8AF.

- *Input Range*: Specify the parameter V_{max} that defines the input range. This parameter is valid only when the SPI device is an A/D converter. If the device conversion mode is DC, the input ranges from 0 to V_{max}; if the device conversion mode is AC, the input ranges from $-V_{max}/2$ to $V_{max}/2$.
- *Scale Factor*: Specify the output scale factor K_{scale}. If the scale factor is 0, the SPI device is not an A/D converter, and the result will be exactly the same as what DSP receives from SPI communication. Otherwise, the SPI device is an A/D converter, and the result is scaled based on the following:

In the DC conversion mode:

- In simulation: Output = Input * K_{scale}
- In hardware: Output = Result * Vmax * K_{scale} / 2^{Data_Length}

In the AC conversion mode:

- In simulation: Output = Input * K_{scale}
- In hardware: Output = (Result - $2^{Data\_Length-1}$) * Vmax * K_{scale} / $2^{Data\_Length-1}$

The parameter *Data_Length* is calculated from the *Data Bit Position* formula.

- *ADC Mode*: Define the A/D conversion mode of the device. It can be either *DC* or *AC*. Note that this parameter is valid only when the device is an A/D converter.
- *Initial Value*: Define the initial value of the input.

SPI Output

A SPI output device may have multiple output channels. The *SPI Output* block is used to define the properties of an output channel for SPI communication, and one SPI Output block corresponds to one output channel.

The parameters of the SPI Output block are explained below:

- *Device Name*: Define the name of the SPI output device.
- *Scale Factor*: Specify the output scale factor K_{scale}. The output is calculated as follows:
  In the DC conversion mode:
  - In simulation: Output = Input * K_{scale}
  - In hardware: Output = Result * K_{scale} * 2^{Data_Length} / Vmax

  In the AC conversion mode:
  - In simulation: Output = Input * K_{scale}
  - In hardware: Output = $2^{Data\ Length-1}$ + Result * K_{scale} * $2^{Data\_Length-1}$ / Vmax

  The parameter *Data_Length* is calculated from the *Data Bit Position* formula, and Vmax is the Output Range.
- *Output Range*: Specify the parameter Vmax that defines the output range. This parameter is valid only when the SPI device is a D/A converter. If the device conversion mode is DC, the output ranges from 0 to Vmax; if the device conversion mode is AC, the output ranges from –Vmax/2 to Vmax/2.
- *DAC Mode*: Define the D/A conversion mode. It can be either *DC* or *AC*. Note that this parameter is valid only when the device is a D/A converter.
- *Sending Data Command*: Define the command to send the output data. The command is a series of hex numbers separated by comma (for example, 0x23,0x43,0x00).
- *Data Bit Position*: Define where the data bits are in the sending data string. The format is:

ElementName={Xn[MSB..LSB]}

where

- *ElementName* is the name of SPI output element. If it is the current SPI output element, use *y* instead.
- {} means that the item in the bracket repeats multiple times.
- *Xn* is the n_{th} word sent to the SPI output device, and n start from 0.
- *MSB..LSB* defines the position of the significant bits in the word.

For example, if y=x1[3..0]x2[7..0] and if the result is 0x8AF, the lower 4 bits of the 2nd word will be 0x8, and the lower 8 bits of the 3rd word will be 0xAF.

- *Sync. Command*: Define the command to synchronize output channels of the SPI output device. It consists of a series of hex numbers separated by comma (for example, 0x23,0x43,0). This command is used when the SPI output device does not have the synchronization signal.

2. SPI Command Sequences

A typical sequence for a SPI input device is shown below:

- Send the device initial command (optional).
- Send the start command and wait for interrupt (optional).
- Set the synchronization signal and wait for interrupt (optional).
- Send the receiving data command.
- Receive the result and call a function to continue the next action.

A typical sequence for a SPI output device is shown below:

- Send the device initial command (optional).
- Send the sending data command.
- Set the synchronization signal to enable the new value (optional).

3. Limitations

There are several limitations in the SPI functionality in PSIM.

a. SPI Operation Mode:

PSIM only supports SPI in the master operation mode. It assumes that only one DSP links to one or more SPI devices, and all SPI devices are in the slave mode.

b. Data Latch Timing:

TI F2833x/F2803x DSP assumes that input/output data are active at the same time. The latch time can be either the falling edge or rising edge of the SPI clock. SPI devices use the same timing to latch data. A SPI device will be not supported if it latches the input data at a different timing than the DSP. Note that this limitation is from the TI F2833x/F2803x.

c. Other Limitations:

- SPI command combined with GPIO port actions

PSIM does not support the type of SPI device if its command is combined with GPIO actions. For example, a SPI LCD display may need a GPIO port to define if the data is a command or a display character I/O.

- Checking the SPI device busy status

PSIM does not support the type of SPI device that needs to check if the device is ready to accept the next command. The device can be supported if it returns the result in the same command or it triggers an interrupt when conversion is complete.

- Daisy chain

PSIM does not support daisy chain.

4. Examples

To illustrate how SPI elements are used, three examples are provided in PSIM:

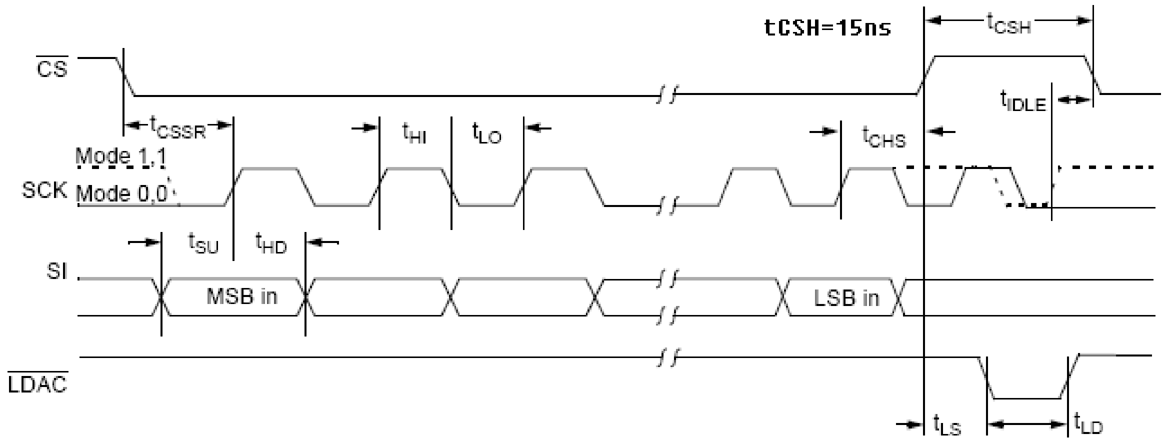
- External D/A converter MCP4922
- External A/D converter MCP3204 (without interrupt)
- External A/D converter TLV1548 (with interrupt)

These examples are located in the folder "examples\SimCoder\F2833x Target" or "examples\SimCoder\F2803x Target\" in the PSIM directory.

External D/A Converter Using MCP4922

MCP4922 is a D/A converter from Microchip Technology Inc. It has two D/A channels. The information that PSIM needs from the manufacturer datasheet is listed below:

- The SPI clock frequency is up to 20MHz.
- The serial interface timing diagram of MCP4922, as shown below, indicates that the DSP SPI clock type is *rising edge with delay*, and the time interval between two conversion commands is 15ns.

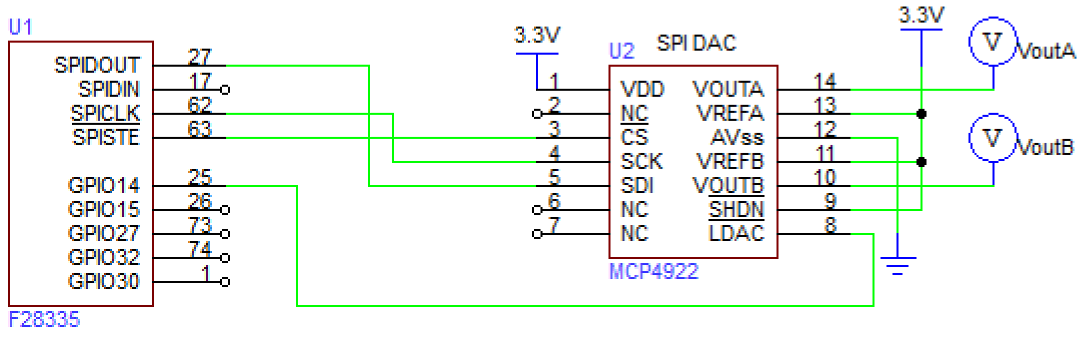


- There is no device initial command nor start conversion command.
- There is no interrupt port.
- There is a synchronization port to synchronize output timing.
- The sending data command of Channel A is 0x7000, and the command of Channel B is 0xF000. The result will be placed in the last 12 bits of the command.

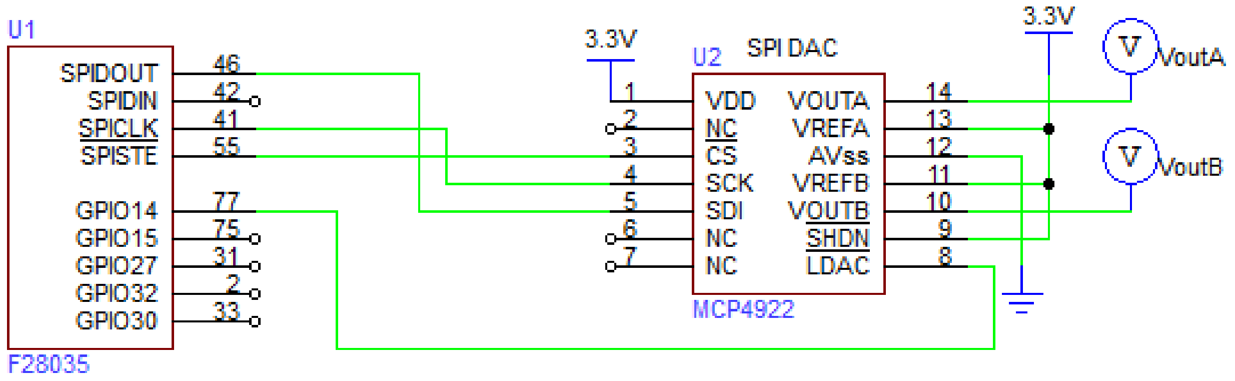
Circuit Schematic

This example is implemented in both F2833x and F2803x Targets, and the implementation in each target is identical except the DSP is different. In this example, two sine wave signals are generated in DSP. They are then sent to the external D/A converter MCP4922 via SPI as two analog outputs. The files of this example are in the folder "examples\F2833x Target\DAC with SPI" for F2833x and "examples\F2803x Target\DAC with SPI" for F2803x.

The hardware circuit diagram for the F2833x Target is shown below.

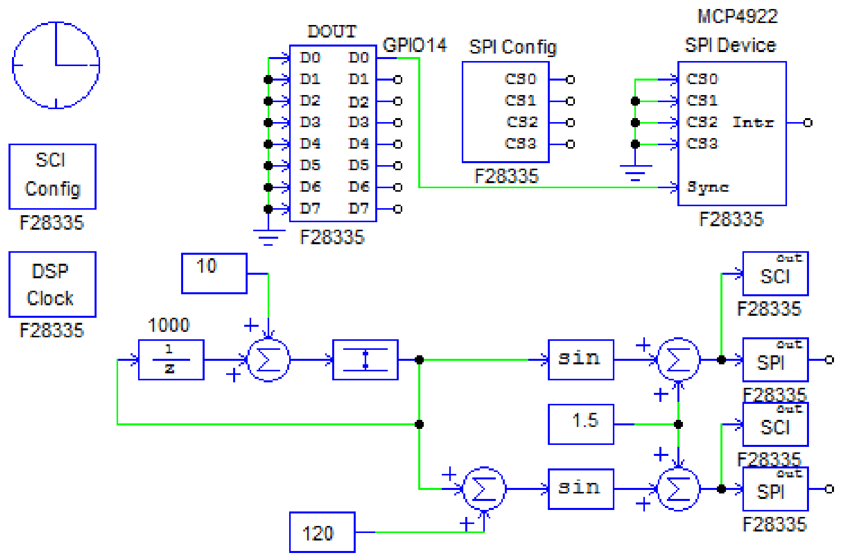


Similarly, the hardware circuit diagram for the F2803x Target is shown below.

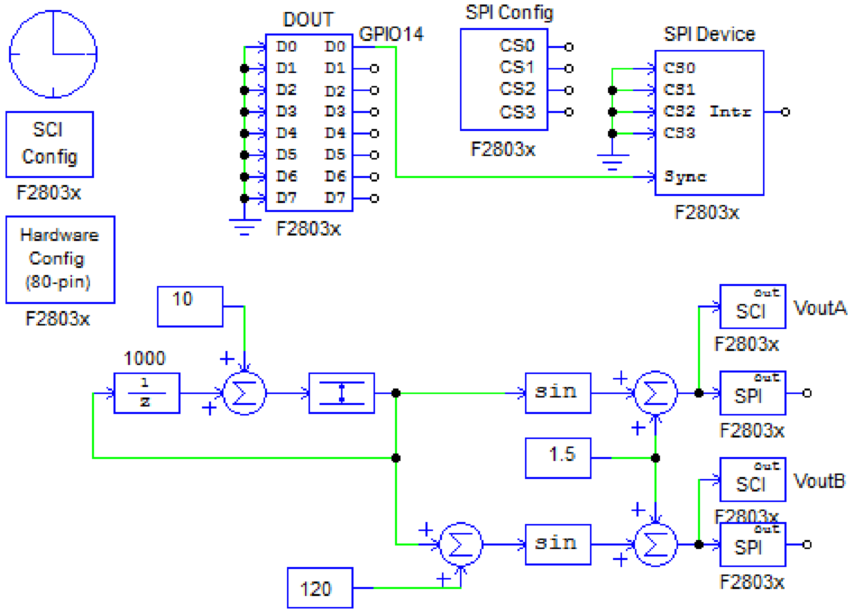


In both schematic, DSP's SPIDOUT pin (GPIO16) is connected to MCP4922's SDI pin for data transmission; SPICLK pin (GPIO18) is connected to MCP4922's SCLK pin as SPI clock; the SPI slave transmit-enable pin SPISTE (GPIO19) is connected to MCP4922’s chip select pin CS; and Port GPIO14 is connected to MCP4922’s LDAC. This connection is to synchronize two D/A outputs.

The corresponding PSIM schematic for the F2833x Target is shown below.



The corresponding PSIM schematic for the F2803x Target is shown below.



In both schematic, the output D0 of the Digital Output block is set to GPIO14 and is connected to the sync input of MCP4922 for synchronization.

With the use of the SPISTE signal, this circuit does not need any other GPIO ports to generate the chip select signal. That is why in the circuit, the chip select pins CS0 to CS3 are not used.

Note that in PSIM, it is implied that SPISTE, SPICLK, SPIDIN, and SPIDOUT pins are connected between the SPI Configuration block and the SPI Device block. Thus no external connection is needed.

Defining SPI Element Parameters

Three types of SPI elements are used in the PSIM schematic in this example: *SPI Configuration*, *SPI Device*, and *SPI Output*. Their parameters are set as follows:

- SPI Configuration: There should be only one SPI Configuration element in a schematic. Since there is only one SPI device in this case, SPISTE is used as the chip select signal in this case. The parameters of the SPI Configuration block are defined as follows:
    - *SPI Port*: GPIO16-19. Note that this is the only group allowed in the TI Experiment's Kit.
    - Chip Select Pin0 to Pin3: "Not used"
    - *SPI Buffer Size*: 32. In this example, the minimum length is 4.
- SPI Device MCP4922: According to the datasheet information of MCP4922, the parameters are defined as below:
    - *Chip Select Pins*: 0000. No chip select pin is used.
    - Communication Speed (MHz): 20
    - *Clock Type*: "Rising edge without delay"
    - Command Word Length: 8 bits

- Sync. Active Mode: "High to low"
- *SPI Initial Command*: No initial command
- *Hardware Interrupt Mode*: "No hardware interrupt"
- *Interrupt Timing*: Set to “No Interrupt”
- Command Gaps (ns): 15
- Conversion Sequence: None
- SPI Output MCP4922 Channel A

According to the datasheet information of MCP4922, the parameters are defined as below:

- *Device Name*: “SP_DAC”
- Scale Factor: 1
- Output Range: 3.3
- DAC Mode: DC
- Sending Data Command: 0x7000
- *Data Bit Position*: y = x0[3..0]x1[7..0]. The result will be placed in the last 12 bits of two 8-bit words.
- Sync. Command: None
- SPI Output MCP4922 Channel B

The definitions are the same as for Channel A above, except that the Sending Data Command is "0xF000".

Generating and Running Code on DSP

User can simulate and generate code in the following steps:

- Select **File** >> **Open** to load the example from "examples\F2833x Target\DAC with SPI" or "examples\F2803x Target\DAC with SPI".
- Select **Simulate** >> **Run Simulation** to run the simulation. Note that the value of the SCI Input will not change during the simulation.
- Select **Simulate** >> **Generate Code** to generate the code.
- Connect the DSP board to the computer physically through a USB cable. If a RS-232 cable is used to connect the DSP board with the computer for SCI data monitoring, be sure to disconnect the RS-232 cable from the computer. Otherwise the DSP board may not be able to connect to the computer properly.

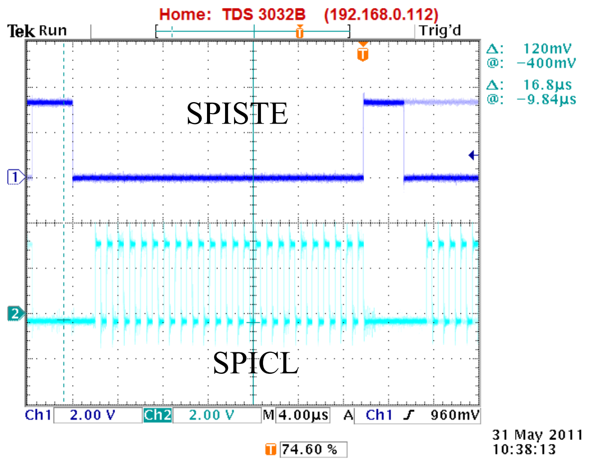
For CCS v5.5, upload the generated code and run this example following the steps below.

- Start CCS v5.5. In CCS, select **Project >> Import Legacy CCSv3.3 project** and load the generated project from the subfolder "DAC with SPI (C code)" of the schematic folder. In the dialog window, click on **Next** and then **Finish** to transfer the CCS v3.3 project to CCS v5.1. The project name will be displayed in the Project Explorer panel.
- Right click on the project name in the Project Explorer panel, and select **Build Project** from the pop-up menu to build this example.
- Select **View >> Target Configurations** to open the target configuration dialog window and link the corresponding user defined configuration (.ccxml file) to this project.

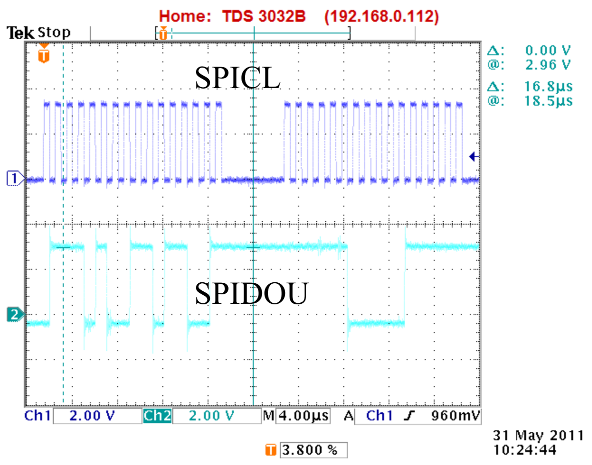
- Click the project name in the Project Explorer panel to set it as the current project, and select **Run >> Debug** to upload program to the target. After the program is uploaded, CCS will stop at the main() function. Select **Run >> Free Run** to run the program.

With the code running, connect the RS-232 cable between the DSP board and the computer, and use PSIM's DSP Oscilloscope feature to display the variables inside the DSP. At the same time, use a lab digital oscilloscope to observe the SPI signals and the D/A outputs.

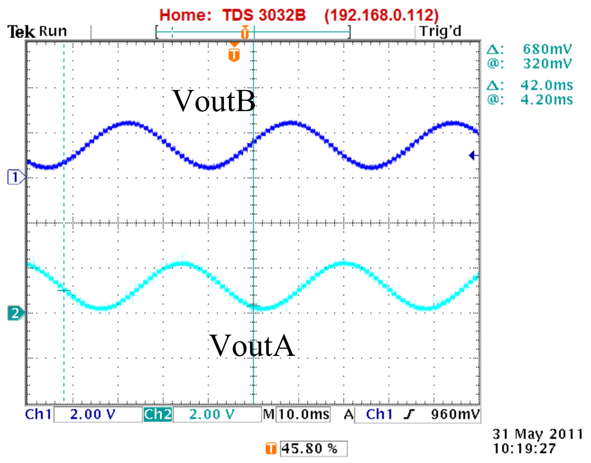
The figures below show the oscilloscope waveforms of the SPI chip select signal SPISTE, the SPI clock signal SPICLK, and D/A outputs VoutA and VoutB. The figure on the lower right shows the DSP Oscilloscope screen with the two sine signals from inside the DSP, which validate the analog output waveforms.



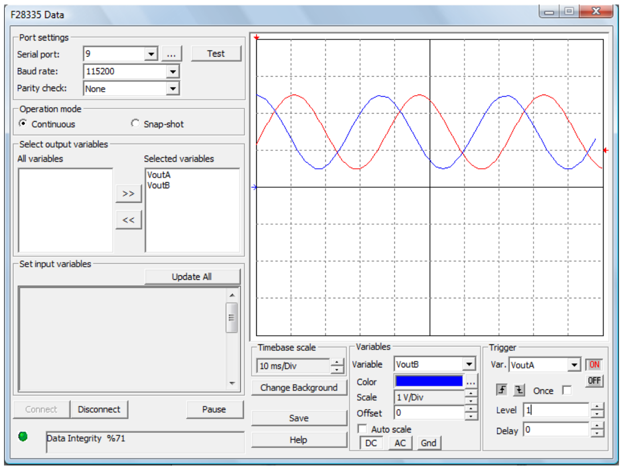
(a) Chip select signal SPISTE vs. SPI clock SPICLK



(b) SPI clock SPICLK vs. SPI output



(c) D/A outputs VoutA and VoutB



F28335 Data

Port settings
Serial port: 9
Baud rate: 115200
Parity check: None

Operation mode: (● Continuous) ○ Snap-shot

Select output variables
Selected variables: VoutA, VoutB

Timebase scale: 10 ms/Div

Variables
Variable: VoutB
Scale: 1 V/Div
Offset: 0
☐ Auto scale

Trigger
Var.: VoutA
Once ☐
Level: 1
Delay: 0

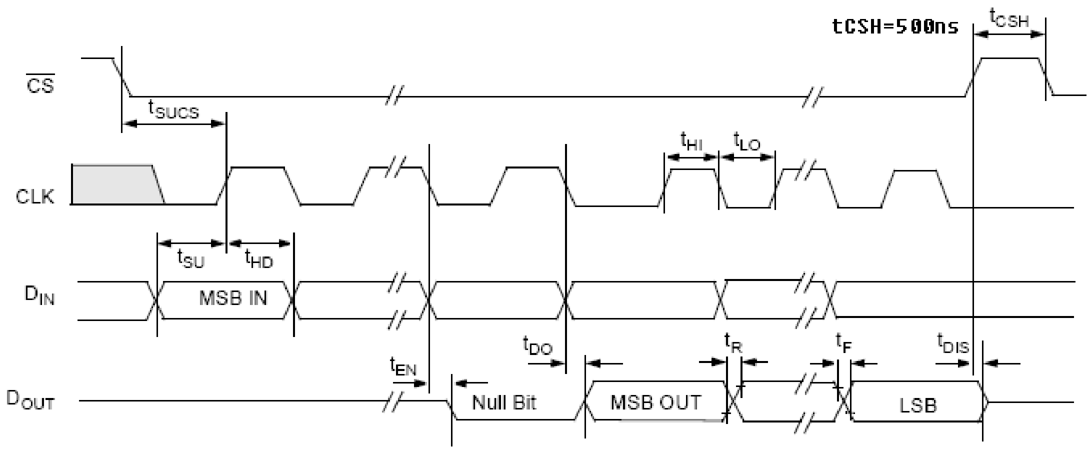
Data Integrity %71

(d) Two sine waveforms inside DSP

External A/D Converter Using MCP3204

MCP3204 is an A/D converter from Microchip Technology Inc. It has four input channels, and it receives a conversion command and returns the result in the same command. The information that PSIM needs from the manufacturer datasheet is listed below:

- The SPI clock frequency is up to 1MHz
- The serial interface timing diagram of MCP3204, as shown below, indicates that the SPI clock type is *rising edge with delay*, and the time interval between two conversion commands is 500ns.

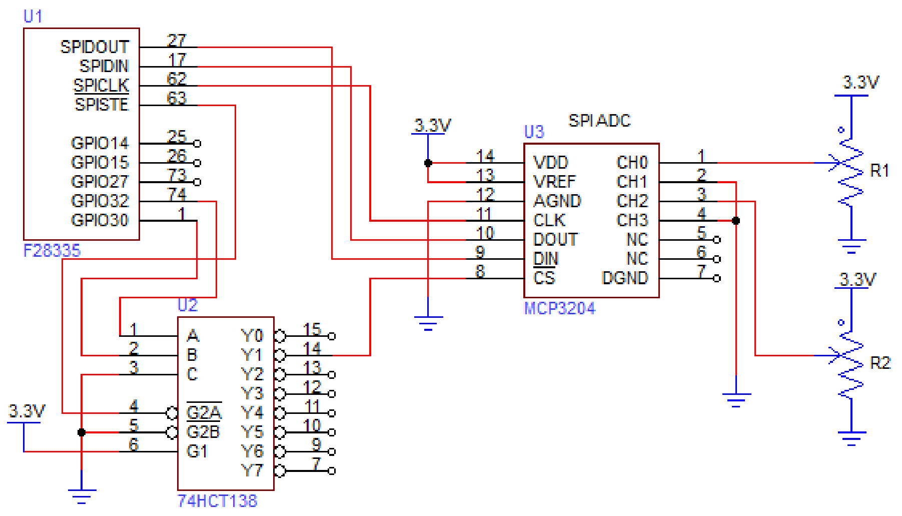


- There is no device initial command nor start conversion command.
- There are no interrupt port and synchronization port.
- The same receiving data command is used to start conversion and get the result.
- The receiving data command of Channel 0 is 0x06,0x00,0x00, and the command of Channel 2 is 0x06,0x80,0x00. The result will be placed in the last 12 bits of the command.

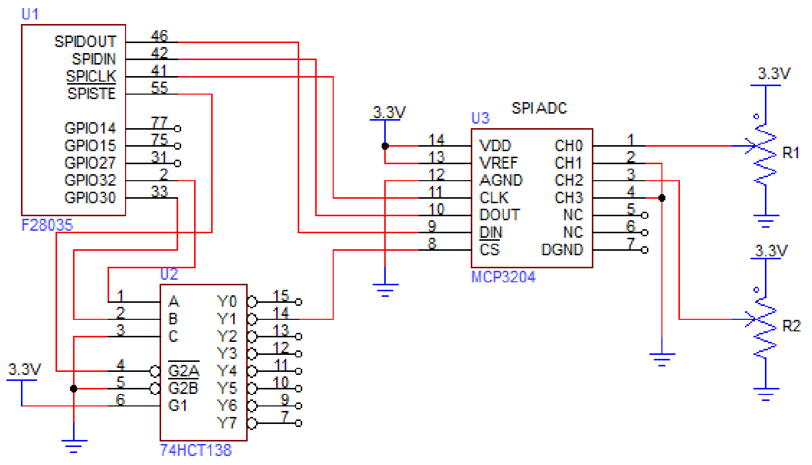
Circuit Schematic

In this example, two analog voltage signals are converted by MCP3204, and are sent to DSP via SPI. The files of this example are in the folder "examples\F2833x Target\ADC with SPI" for F2833x and "examples\F2803x Target\ADC with SPI" for F2803x.

The hardware circuit diagram for the F2833x Target is shown below.

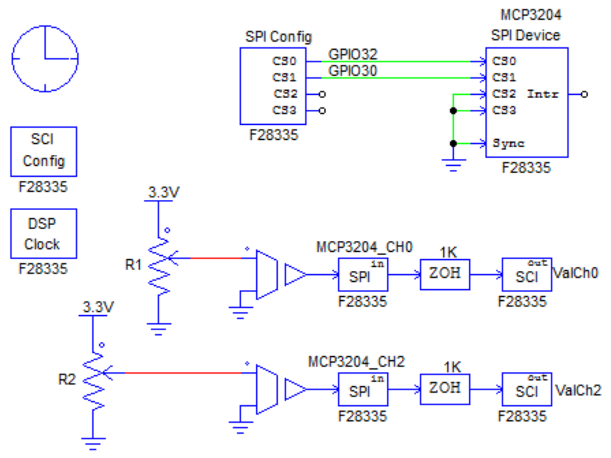


The hardware circuit diagram for the F2803x Target is shown below.

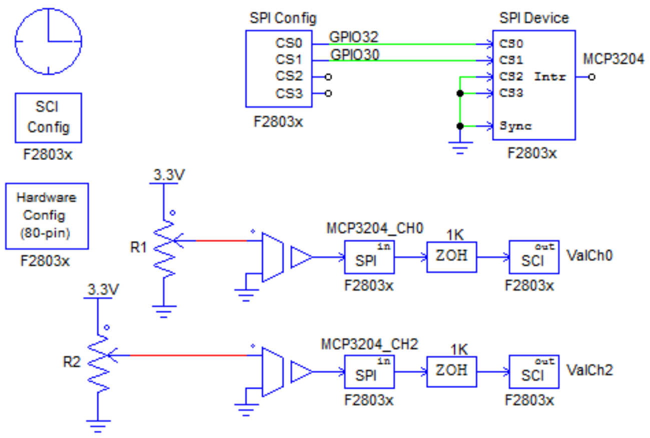


In this example, instead of using only the SPISTE signal, it will use SPISTE as well as GPIO32 and GPIO30, together with the decoder chip 74HCT138, to generate the chip select signal for the A/D converter.

The corresponding PSIM schematic for F28335 target is shown below.



The corresponding PSIM schematic for the F2803x Target is shown below.



Note that since two GPIO pins are used to generate the chip select signal, nodes CS0 and CS1 of the SPI Configuration block and the SPI Device are connected.

Defining SPI Element Parameters

Parameters of the SPI blocks are set as follows:

- SPI Configuration

Two chip select pins are used in this example.

- *SPI Port*: GPIO16-19
- Chip Select Pin0: GPIO32
- Chip Select Pin1: GPIO30
- Chip Select Pin2 and Pin3: "Not used"
- *SPI Buffer Size*: 32. In this example, the minimum is 4.
- SPI Device MCP3204

According to the datasheet information of MCP2304, the parameters are defined as below:

- *Chip Select Pins*: 0001. It means that GPIO32 = 1 and GPIO30 = 0.
- Communication Speed (MHz): 2.8
- *Clock Type*: "Rising edge with delay"
- Command Word Length: 8 bits
- Sync. Active Mode: Do not care
- *SPI Initial Command*: No initial command.
- Hardware Interrupt Mode: Do not care
- *Interrupt Timing*: "No interrupt"
- Command Gaps (ns): 0
- Conversion Sequence: None
- SPI Input MCP3204 Channel 0

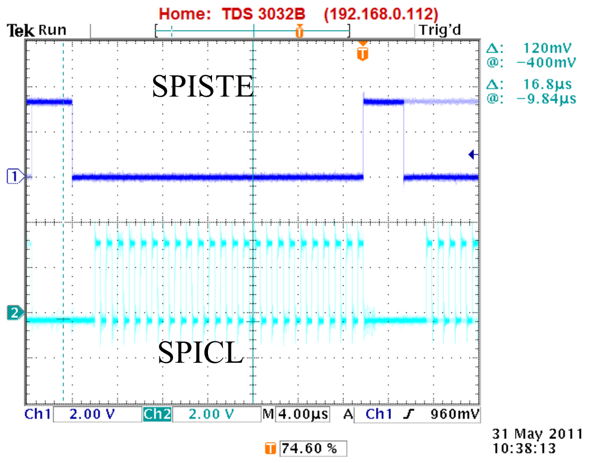
According to the datasheet information of MCP3204, the parameters are defined as below:

- *Device Name*: "MCP3204"
- Start Conversion Command: None
- Receiving Data Command: 0x06, 0x00, 0x00
- *Data Bit Position*: y=x1[3..0]x2[7..0]
- Input Range: 3.3V
- Scale Factor: 1
- ADC Mode: DC
- Initial Value: 0
- SPI Input MCP3204 Channel 2

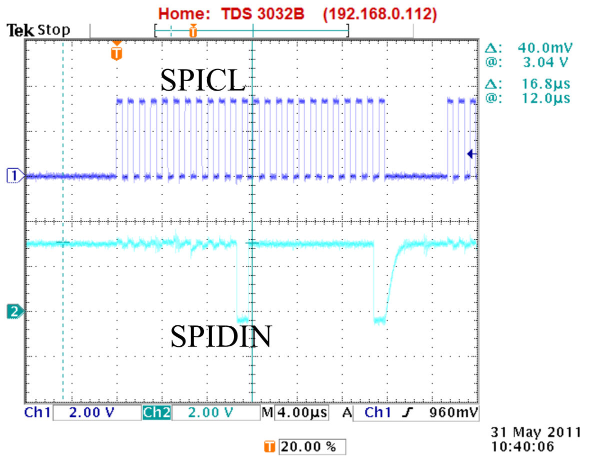
The definitions are the same as for Channel 0, except that the Receiving Data Command is 0x06,0x80,0x00.

Running the Code

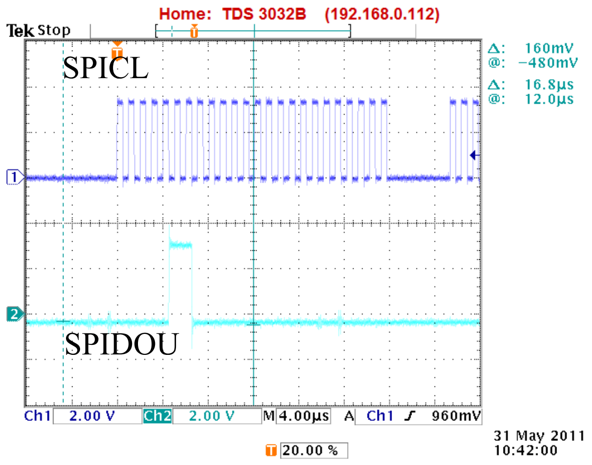
The process to generate, compile, and run the code is similar to what is described in Section 4.1.3. Below are some oscilloscope waveforms from lab experiments.



(a) Signal SPISTE vs. SPI clock SPICLK



(b) SPI clock SPICLK vs. A/D output



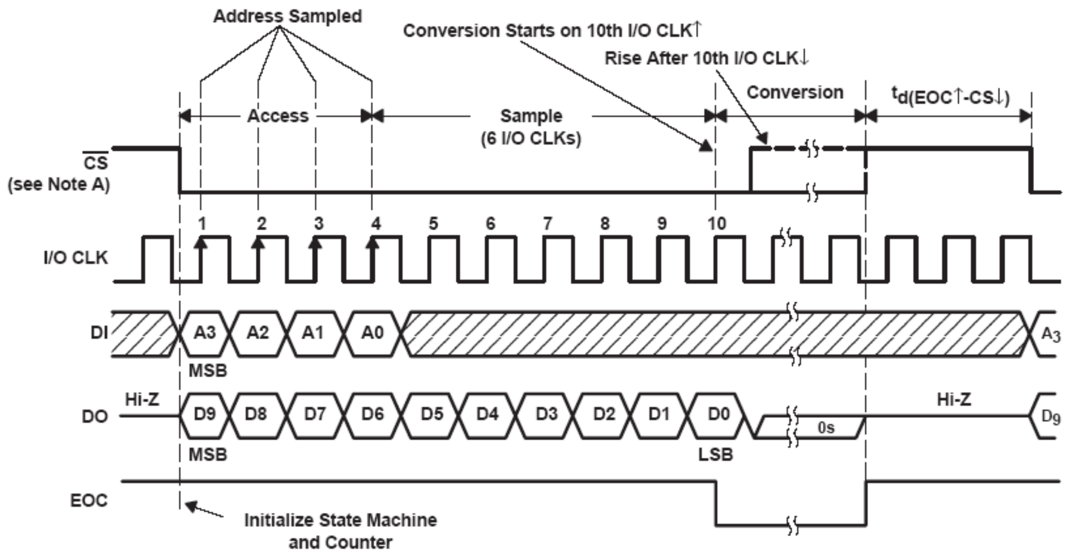
(c) SPI clock SPICLK vs. A/D conversion command

External A/D Converter with Interrupt Using TLV1548

TLV1548 is an 8-channel A/D converter from Texas Instruments. It receives a conversion command to start conversion, and it generates an interrupt when conversion completes. Then it receives the next channel conversion command and sends the result from the previous conversion in the same command sequence.

The information that PSIM needs from the manufacturer datasheet is listed below:

- The SPI clock frequency is up to 2MHz
- The command word length is found to be 14 bits (either 10 or 12 bits would not work).
- The serial interface timing diagram of TLV1548, as shown below, indicates that the SPI clock type is *rising edge with delay*, and the time interval between two conversion commands is 0ns.

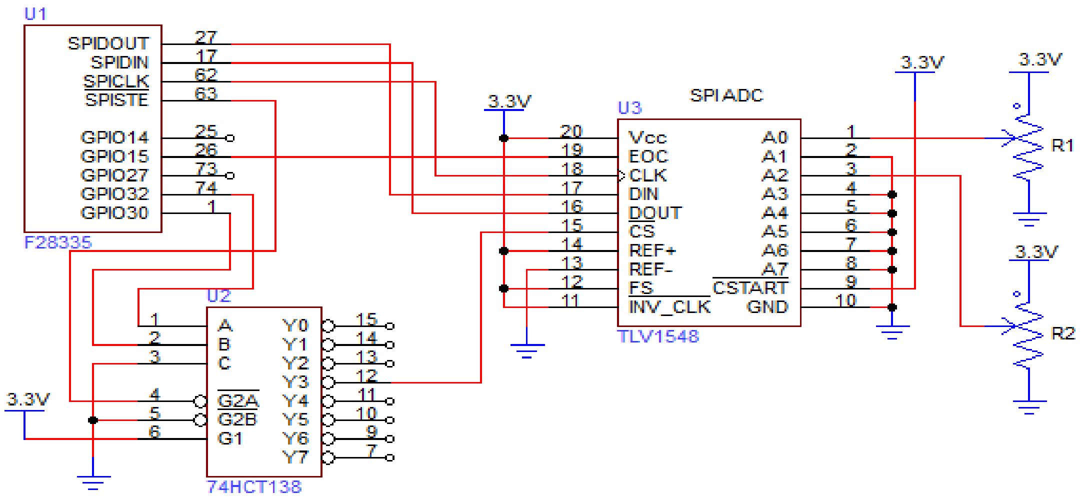


- There is no synchronization signal.
- For the fast conversion mode, the command is 0x2400.
- The hardware interrupt is triggered at the rising edge of EOC (end of conversion).
- Since there is only one 10-bit ADC unit in TLV1548, if there are multiple input channels, conversion needs to be done one channel at a time.
- This example only uses Channel 0 and Channel 2, and the conversion order is TLV1548_CH0, TLV1548_CH2.
- The conversion command is 0x0000 for Channel 0; and is 0x1000 for Channel 2.

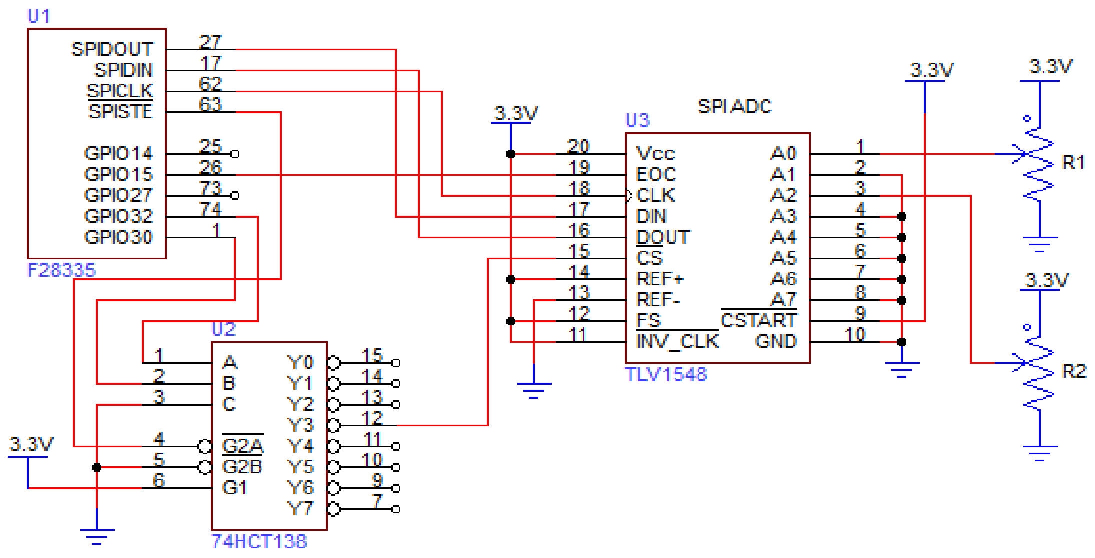
Circuit Schematic

In this example, two analog voltage signals are converted by TLV1548, and are sent to DSP via SPI. The files of this example are in the folder "examples\F2833x Target\ADC (interrupt) with SPI" for F2833x and "examples\F2803x Target\ADC (interrupt) with SPI" for F2803x.

The hardware circuit diagram for the F2833x Target is shown below.

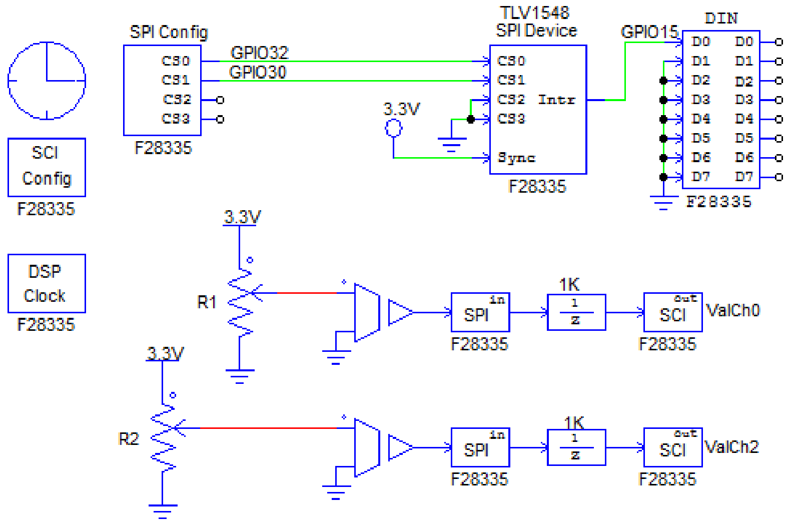


The hardware circuit diagram for the F2803x Target is shown below.

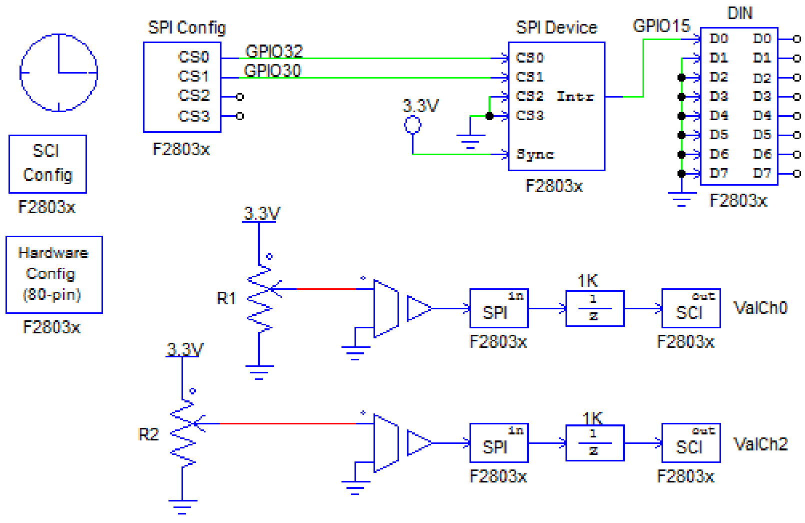


This example uses the SPISTE pin as well as GPIO30 and GPIO32 to generate the chip select signal. TLV1548 generates interrupt through Port GPIO15.

The corresponding PSIM schematic for the F2833x target is shown below.



The corresponding PSIM schematic for the F2803x Target is shown below.



In the schematic, nodes CS0 and CS1 of the SPI Configuration block and the SPI Device block are connected. Again, it is implied that SPISTE, SPICLK, SPIDIN, and SPIDOUT pins are connected between the SPI Configuration block and the SPI Device, and no external connection is needed.

The interrupt output of the SPI device TLV1548 is connected to the D0 pin of the digital input block, and D0 is defined as GPIO15.

When TLV1548 receives the start conversion command, it will start the conversion on Channel 0. Once the conversion is complete, TLV1548 will generate an interrupt. DSP will respond to this interrupt, and send another command. TLV1548 will start the conversion on Channel 2, and will send back the conversion result of Channel 0.

Defining SPI Element Parameters

Parameters of the SPI blocks are set as follows:

- SPI Configuration

Two chip select pins are used in this example.

- *SPI Port*: GPIO16-19
- Chip Select Pin0: GPIO32
- Chip Select Pin1: GPIO30
- Chip Select Pin2 and 3: "Not used"
- SPI Buffer Size: 32
- SPI Device TLV1548

According to the datasheet information of TLV1548, the parameters are defined as below:

- *Chip Select Pins*: 0011. That is, GPIO32 = 1 and GPIO30 = 1.
- Communication Speed (MHz): 2
- *Clock Type*: "Rising edge with delay"
- Command Word Length: 14 bits
- Sync. Active Mode: Do not care

- SPI Initial Command: 0x2400
- Hardware Interrupt Mode: "Rising edge"
- *Interrupt Timing*: "Multiple interrupt in series"
- Command Gap (ns): 0
- *Conversion Sequence*: "TLV1548_CH0,TLV1548_CH2"
- SPI Input TLV1548 Channel 0

According to the datasheet information of TLV1548, the parameters are defined as below:

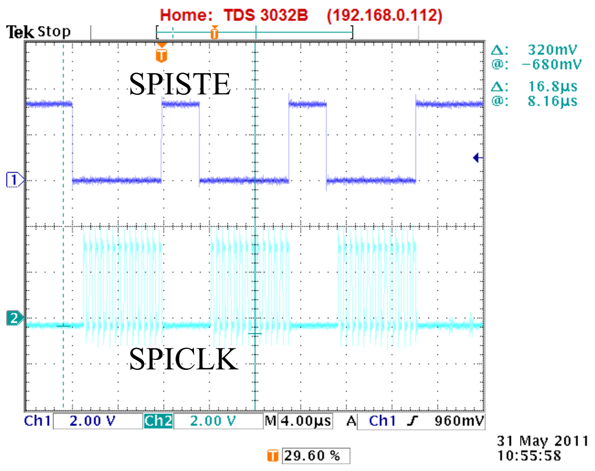
- *Device Name*: "TLV1548"
- Start Conversion Command: 0x0000
- *Receiving Data Command*: 0x0800. This is also the conversion command of Channel 2.
- Data Bit Position: y=x0[13..4]
- Input Range: 3.3
- Scale Factor: 1
- ADC Mode: DC
- Initial Value: 0
- SPI Input TLV1548 Channel 2

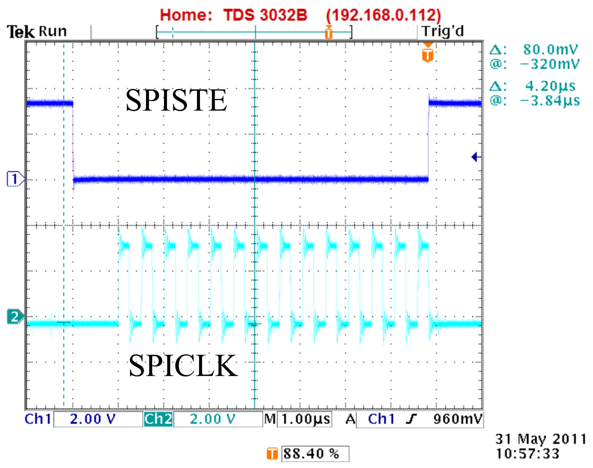
The parameters are the same as for Channel 0, except that there is no Start Conversion Command. The Receive Data Command of Channel 0 is also the conversion command of Channel 2. Also, the Receive Data Command is 0x2000, and this is a dummy command.

Generating and Running Code on DSP

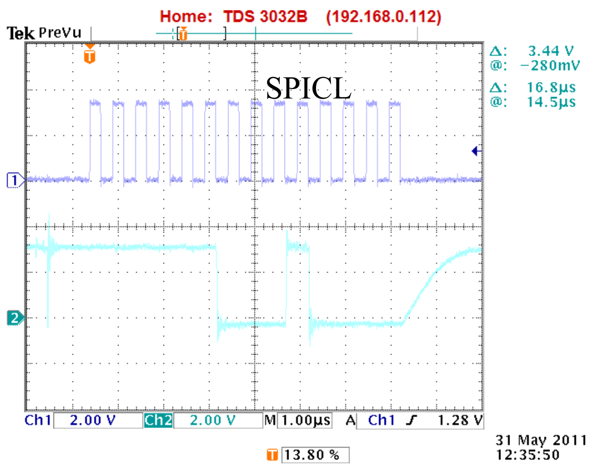
The process to generate, compile, and run the code is similar to what is described in Section 4.1.3.

Below are some oscilloscope waveforms from lab experiments.

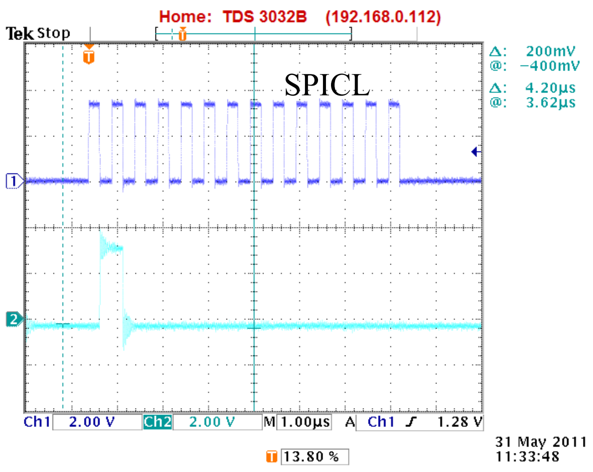




(a) Signal SPISTE vs. SPI clock SPICLK (b) Detailed view of SPISTE vs. SPICLK



(c) SPI clock SPICLK vs. A/D result



(d) SPI clock SPICLK vs. Conversion command