Keywords: daisy-chain, daisy-chaining, daisy-chains, diasy-chain, diasy-chaining

APPLICATION NOTE 3947

Daisy-Chaining SPI Devices

Dec 15, 2006

Abstract: In typical SPI systems with one master and multiple slaves, a dedicated chip-select signal is used to address an individual slave. As the number of slaves increase, so do the number of chip-select lines. In this situation, the board layout of the system can become quite a challenge.

One layout alternative is daisy-chaining. This article explains the details of a daisy-chained SPI system and shows how software can be used to propagate commands through a string of slaves.

A standard SPI™/QSPI™/MICROWIRE™-compatible microcontroller communicates with its slave devices through a 3- or 4-wire serial interface. The typical interface includes a chip-select signal (active-low CS), a serial clock (SCLK), a data input signal (DIN), and occasionally, a data output signal (DOUT). Devices that are individually addressable, as is usual in I²C systems, communicate easily with a single device on a bus.

A Basic Serial Communication Interface

Many SPI devices are not individually addressable. Consequently, communication between those devices and a single device on a bus requires additional hardware or software organization. **Figure 1** shows a system in which one microcontroller communicates with multiple slave devices.

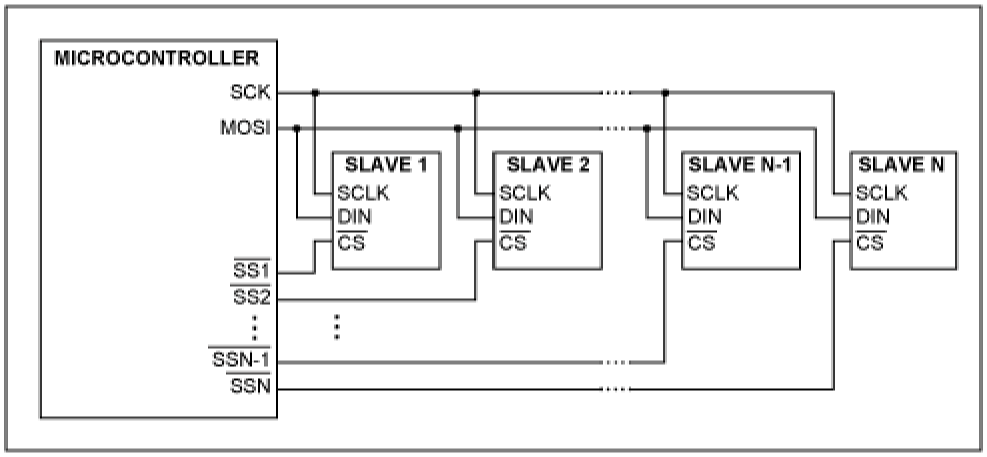

Figure 1. Microcontroller with independent chip selects for multiple slave devices.

In the above system, the microcontroller uses one serial clock output (SCK) and one master-out/slave-in line (MOSI) to command all the slaves. The microcontroller allots an independent slave-select signal (active-low SS_) to each slave device so they can be addressed individually. Because all slaves share the single clock and data lines, only the slaves with their active-low CS inputs asserted low will acknowledge and respond to the activity on the serial clock and data lines. This system is simple to implement when there are very few slave devices in the system. In systems with many slave devices, the microcontroller will need as many active-low SS_ outputs as the number of slaves. This architecture increases hardware and layout complexity.

The Daisy-Chain Alternative

Hardware constraints can make the method in Figure 1 impractical and difficult to implement. An alternative method for serial-interface applications is daisy-chaining, which propagates commands through devices connected in series. **Figure 2** shows an N-device system in a daisy-chain configuration.

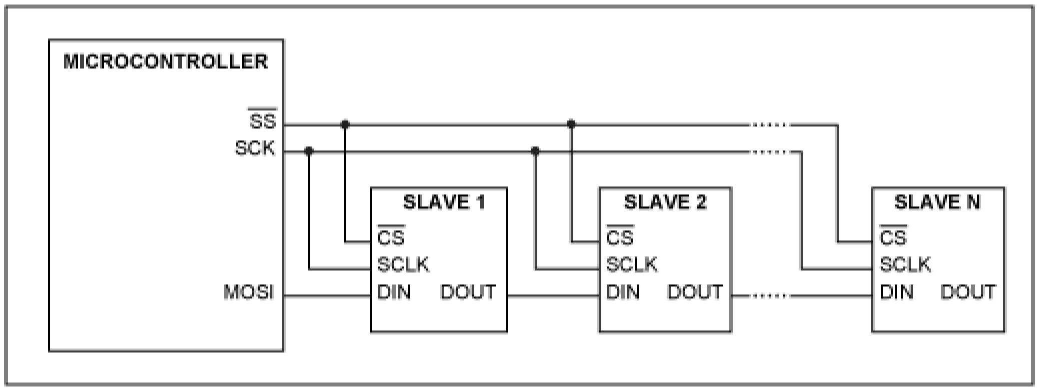

Figure 2. Microcontroller with multiple daisy-chained slave devices.

A single active-low SS (or active-low CS) signal controls all the slaves' active-low CS inputs; all slaves receive the same clock signal. Only the first slave in the chain (SLAVE 1) receives the command data directly from the microcontroller. Every other slave in the network receives its DIN data from the DOUT output of the preceding slave in the chain.

For daisy-chaining to work successfully, the slave must be able to input a command at DIN during a given command-cycle (defined by the number of clock pulses required to clock in one command), and output the same command at DOUT during the subsequent command-cycle. Stated simply, there is a DIN-to-DOUT delay of one command-cycle. The slave must, moreover, only execute the command written to it on the rising edge of active-low CS. This means that as long as active-low CS remains low, the slave ignores the command and outputs it at DOUT on the following command-cycle. If active-low CS goes high after a given command-cycle, all slaves execute the commands just written to their respective DIN inputs. If active-low CS goes high, data is not output at DOUT. This process makes it possible for every slave in the chain to execute a different command. As long as these daisy-chain requirements are satisfied, the microcontroller only needs three signals (active-low SS, SCK, and MOSI) to control all the slaves in the network.

How Daisy-Chaining Is Accomplished

In a daisy-chained system (Figure 2), SLAVE 1 receives data directly from the microcontroller. This data is clocked into SLAVE 1's internal shift register. As long as active-low CS (or active-low SS) remains low, this data propagates through to SLAVE 1's DOUT output. DOUT of SLAVE 1 goes into DIN of SLAVE 2, so the data is clocked into SLAVE 2's internal shift register as the data appears on SLAVE 1's DOUT output. Just as SLAVE 2 receives its data from SLAVE 1, the microcontroller can simultaneously send another command to SLAVE 1. This new command overwrites the previous data in SLAVE 1's shift register. As long as active-low CS remains low, the data propagates through the entire daisy-chain until each of the slave devices has received its appropriate command. The command loaded into each slave's shift register executes on the rising edge of active-low CS. The following examples use the MAX5233 and MAX5290 to demonstrate daisy-chaining.

Example Circuit #1

Figure 3 shows three MAX5233 ICs connected in a daisy-chain configuration. The MAX5233 is a dual, 10-bit DAC (contains two DAC channels, A and B). With RSTV connected to V_{DD}, the analog outputs power up to midscale.

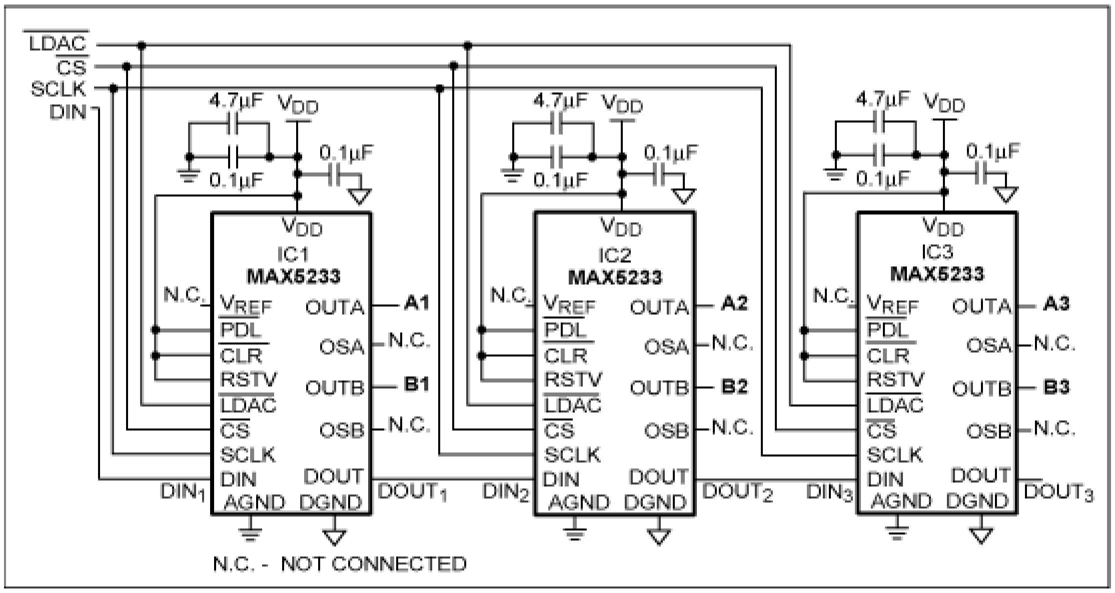

Figure 3. Daisy-chain circuit #1.

Figure 4 shows the command sequence to set IC1's (A1 and B1), IC2's (A2 and B2), and IC3's (A3 and B3) outputs to zero-, mid-, and full-scale, respectively. For this example, the following commands are used:

- 0x7FF8—loads IC3 DAC registers with full-scale data and sets both outputs (A3, B3) to full scale
- 0x7000—loads IC2 DAC registers with midscale data and sets both outputs (A2, B2) to midscale
- 0x6000—loads IC1 DAC registers with zero-scale data and sets both outputs (A1, B1) to zero scale

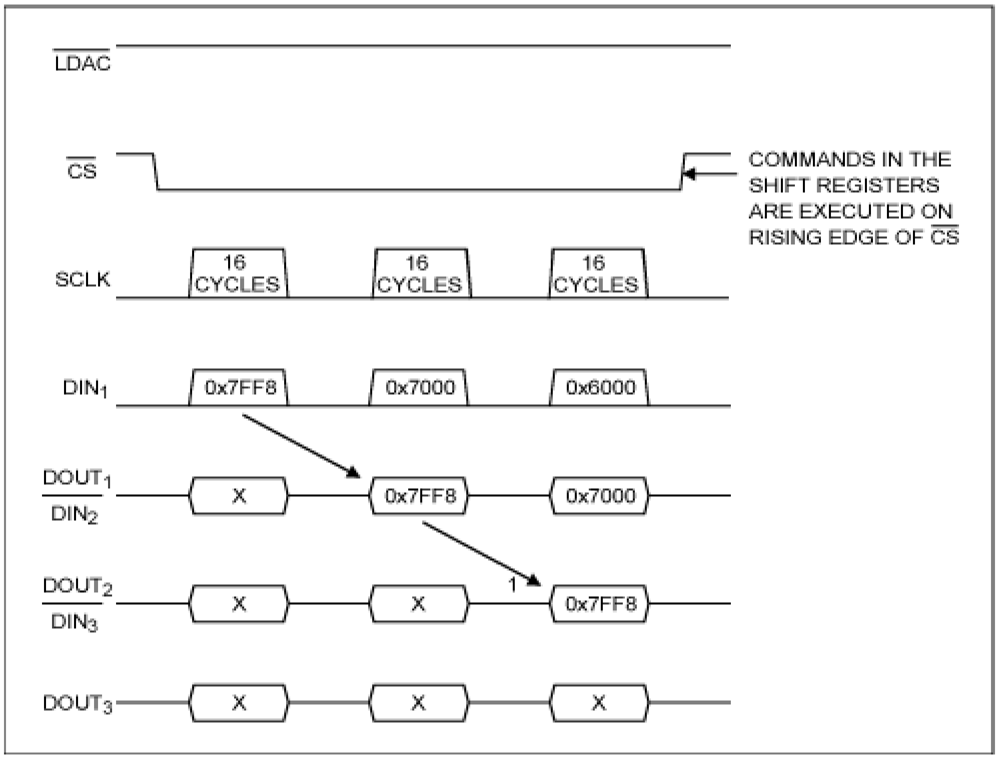

Figure 4. Circuit #1—Command Sequence A.

During the first command cycle (set of 16 SCLK pulses), 0x7FF8 gets loaded in the shift register of IC1. With active-low CS remaining low, this data propagates through IC1 and is output at $DOUT_1$ during the next command cycle. During this second command cycle, the data from $DOUT_1$ moves directly into DIN_2, and 0x7FF8 is loaded into IC2's shift register. Simultaneously, a new command, 0x7000, gets loaded into IC1's shift register, thus overwriting its previous command.

In the third command cycle, the first command, 0x7FF8, is loaded into IC3's shift register. The second command, 0x7000, gets loaded into IC2, and IC1 receives a new command, 0x6000. All three ICs now have a command, which they received through the daisy-chain in their shift registers. When active-low CS goes high, the loaded commands execute; A1 and B1 are set to zero scale, A2 and B2 set to midscale, and A3 and B3 set to full scale.

Figure 5 illustrates a more complicated command sequence. The following commands are used (See the MAX5233 data sheet for more information.):

- 0x3FF8—loads Input Register A with full-scale data, DAC registers and output are unchanged
- 0x3000—loads Input Register A with midscale data, DAC registers and output are unchanged
- 0x2000—loads Input Register A with zero-scale data, DAC registers and output are unchanged
- 0xBFF8—loads Input Register B with full-scale data, DAC registers and output are unchanged
- 0xB000—loads Input Register B with midscale data, DAC registers and output are unchanged
- 0xA000—loads Input Register B with zero-scale data, DAC registers and output are unchanged
- 0x0000—NO-OP

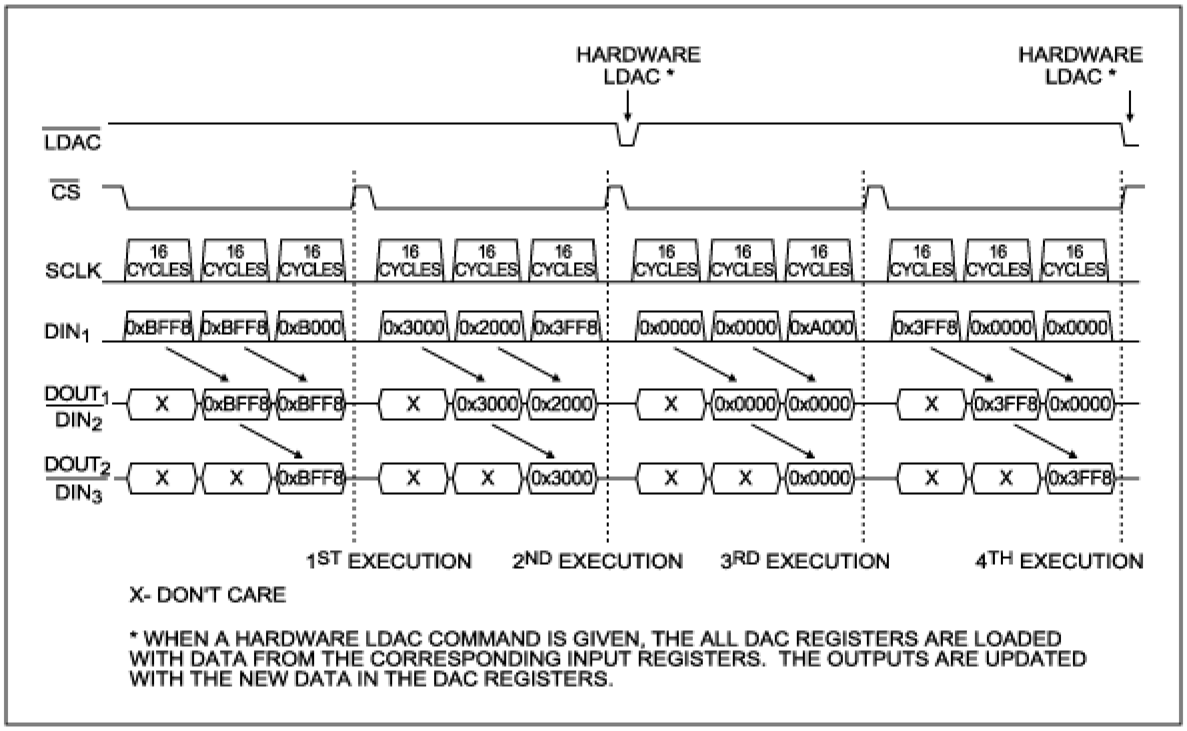
HARDWARE LDAC *
HARDWARE LDAC *
LDAC
CS
SCLK
16 CYCLES
DIN1
0xBFF8 0xBFF8 0xB000 0x3000 0x2000 0x3FF8 0x0000 0x0000 0xA000 0x3FF8 0x0000 0x0000
DOUT1 / DIN2
X 0xBFF8 0xBFF8 X 0x3000 0x2000 X 0x0000 0x0000 X 0x3FF8 0x0000
DOUT2 / DIN3
X X 0xBFF8 X X 0x3000 X X 0x0000 X X 0x3FF8
1ST EXECUTION
2ND EXECUTION
3RD EXECUTION
4TH EXECUTION
X- DON'T CARE
* WHEN A HARDWARE LDAC COMMAND IS GIVEN, THE ALL DAC REGISTERS ARE LOADED WITH DATA FROM THE CORRESPONDING INPUT REGISTERS. THE OUTPUTS ARE UPDATED WITH THE NEW DATA IN THE DAC REGISTERS.

Figure 5. Circuit #1—Command Sequence B.

During the first three command cycles, each of the three ICs in the daisy-chain receives a command in its shift register. The commands for IC1, IC2, and IC3 are 0xB000, 0xBFF8, and 0xBFF8, respectively. These commands execute on the rising edge of active-low CS (the first execution). After the first execution, input register B of IC1, IC2, and IC3 is loaded with data for midscale, full scale, and full scale, respectively. At this point, B1, B2, B3 remain unchanged because the DAC register B of each IC remains unchanged.

In the following three command cycles, the commands for loading the input register A only are written to each IC's shift register. The DAC register A and its outputs remain unchanged. At the rising edge of active-low CS, the input register A of IC1, IC2, and IC3 is loaded with data for full scale, zero scale, and midscale, respectively. At this point, A1, A2, and A3 remain unchanged as only the input register A, and not the DAC register A, was updated.

The hardware active-low LDAC command (drive active-low LDAC low) following the second execution loads all the DAC registers with data from their respective input registers. The DAC outputs are updated with the data from their corresponding DAC registers. A1, B2, and B3 go to full scale. A2 drops to zero scale and A3 stays at midscale.

During the third series of command cycles, IC2 and IC3 are given NO-OP commands (0x0000) while IC1 receives the 0xA000 command to load its input register B with zero-scale data. After the third execution, all outputs remain unchanged.

In the fourth series of command cycles, IC1 and IC2 receive NO-OP commands while IC3 receives a 0x3FF8. After the fourth execution, the input register A of IC3 is loaded with full-scale data. Another hardware active-low LDAC command loads the DAC registers with the data from the inputs registers. This causes a change in B1 from midscale to zero scale, and A3 from midscale to full scale. All other outputs remain unchanged.

Table 1. Output states of IC1, IC2, and IC3 After Power-Up and the Hardware Active-Low LDAC Commands in Sequence B

	Analog Output Name	State of Output		
		After Power-Up (RSTV = V_{DD})	First Hardware Active-Low LDAC	Second Hardware Active-Low LDAC
IC1	A1	Midscale	Full scale	Full scale
	B1	Midscale	Midscale	Zero scale
IC2	A2	Midscale	Zero scale	Zero scale
	B2	Midscale	Full scale	Full scale
IC3	A3	Midscale	Midscale	Full scale
	B3	Midscale	Full scale	Full scale

Example Circuit #2

Figure 6 shows three MAX5290 dual, 12-bit DACs in a daisy-chain configuration. With PU connected to DV_{DD}, the analog outputs power up to full scale. The MAX5290 does not have a dedicated digital output for daisy-chaining. Instead one of the two UPIO (user-programmable input/output) pins must be programmed by the serial interface for DOUTDC_ mode. See the MAX5290 data sheet for details.

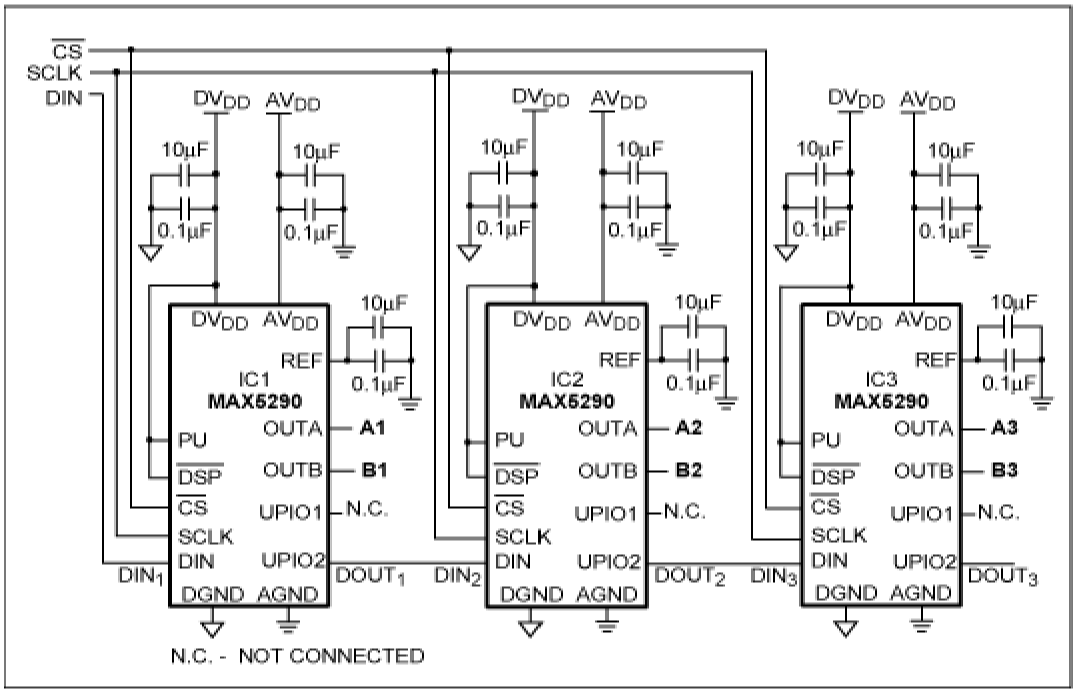

Figure 6. Daisy-chain circuit #2.

Figure 7 provides a command sequence example and the following commands are used (See the MAX5290 data sheet for more information.):

- 0xDFFF—loads all input and DAC registers with full-scale data, DAC A and B outputs are updated
- 0xD800—loads all input and DAC registers with midscale data, DAC A and B outputs are updated
- 0xD000—loads all input and DAC registers with zero-scale data, DAC A and B outputs are updated
- 0xE400—places DAC A and DAC B in shutdown mode

- 0xE40F—takes DAC A and DAC B out of shutdown mode
- 0xFFFF—no-op

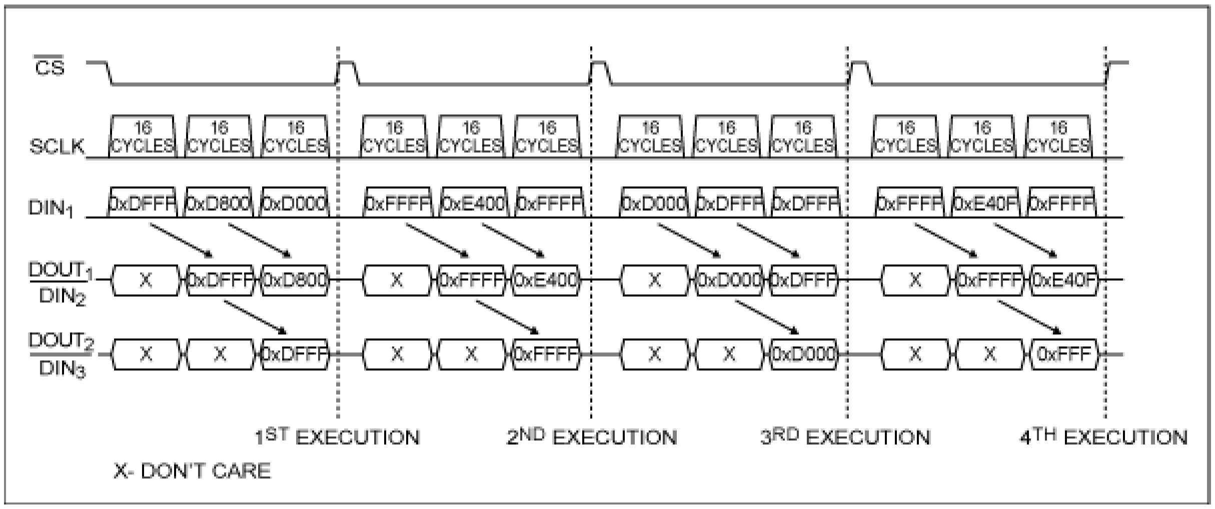

Figure 7. Circuit #2 example command sequence.

The commands loaded into each device's shift register are executed on the rising edge of active-low CS. At the first execution, all the DAC outputs are updated. IC1's DAC outputs go to zero scale, IC2's DAC outputs go to midscale, and IC3's DAC outputs go to full scale.

During the second command cycle, both DACs A and B of IC2 are placed in shutdown mode with the command 0xE400. The no-op commands leave IC1 and IC3 unaffected. After the third command cycle, IC1's outputs change to full scale and IC3's outputs change to zero scale. While IC2's outputs remain in shutdown, the data is updated in the internal input and DAC registers. IC2 is reinstated to normal operating mode during the final command cycle, and the outputs go to full scale.

Table 2. Output states of IC1, IC2, and IC3 After Power-Up and Command Executions in Example Command Sequence.

	Analog Output Name	State of Output				
		After Power-Up (PU = V_{DD})	After First Excution	After Second Execution	After Third Execution	After Fourth Execution
IC1	A1	Full scale	Zero scale	Zero scale	Full scale	Full scale
	B1	Full scale	Zero scale	Zero scale	Full scale	Full scale
IC2	A2	Full scale	Midscale	Shutdown	Shutdown	Full scale
	B2	Full scale	Midscale	Shutdown	Shutdown	Full scale
IC3	A3	Full scale	Full scale	Full scale	Zero scale	Zero scale
	B3	Full scale	Full scale	Full scale	Zero scale	Zero scale

A similar article appeared in the September 2006 issue of *EE Times*.

Related Parts		
MAX5233	3V/5V, 10-Bit, Serial Voltage-Output Dual DACs with Internal Reference	Free Samples
MAX5290	Buffered, Fast-Settling, Dual, 12-/10-/8-Bit, Voltage-Output DACs	Free Samples