Programmers Handbook for the MAX5290-MAX5295, MAX5580-MAX5585, MAX5590-MAX5595 User-Programmable DACs

The Programmers Handbook provides detailed timing patterns and advanced programming features for the MAX5290-MAX5295, MAX5580-MAX5585, MAX5590-MAX5595 user-programmable D/A converters (DACs).

This handbook provides detailed timing patterns and advanced programming features for the MAX5290-MAX5295, MAX5580-MAX5585, and MAX5590-MAX5595 user-programmable D/A converters (DACs). Each device in this family shares a common, highly flexible, 3-, 4-, or 5- wire serial interface. The inputs and outputs which make up the interface are:

- **SCLK** - serial clock input. Data may be clocked into, and out of, the serial interface on either rising or falling clock edges, depending on configuration.
- **DIN** - serial data input.
- **CS** - active-low chip select. For all modes except the DSP Frame Sync mode, the falling edge of CS corresponds to the start of the serial interface command, and the rising edge of CS corresponds to the end of the command.
- **DSP** - DSP is sampled at the end of the power-on reset sequence, and its state determines the active edge of the SCLK signal for clocking in data at DIN. Connect DSP to DVDD to clock in data on the rising edge of SCLK, or connect to DGND to clock in data on the falling edge of SCLK. DSP may also be actively driven, in which case the first rising edge of DSP after the power-on reset will enable the DSP Frame Sync mode.
- **UPIO1/UPIO2** - Each device in the family has two user-programmable I/O ports (UPIO1 and UPIO2) which can be configured in a variety of modes including serial output data ports, either for read-back (DOUTRB) or daisy-chaining (DOUTDC0 or DOUTDC1).

Please refer to the MAX5290-MAX5295, MAX5580-MAX5585, or MAX5590-MAX5595 datasheets for details on timing specifications and device configurations.

Any device within the family is capable of supporting a wide range of configurations, including (but not limited to) the following:

- Write operation to a single device
- Read operation from a single device

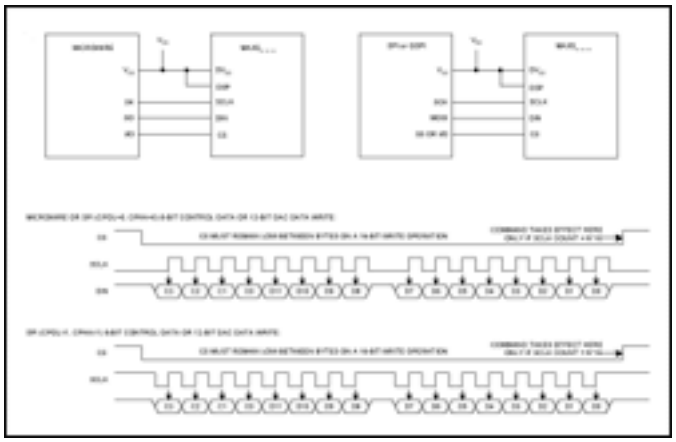
- Write and read access of multiple devices in a daisy-chain
- Direct write and read access of multiple devices (no DSP Frame Sync mode)
- Direct write and read access of multiple devices in DSP Frame Sync Mode

In the following sections, a sample of the possible configurations are shown as examples to expand on the capabilities of the serial interface.

Write Operation to a Single Device

The serial interface supports write-only operations using three inputs: CS, SCLK, and DIN. To clock data in on the rising edge of SCLK, connect DSP to DV_{DD}. To clock data in on the falling edge of SCLK, connect DSP to DGND. Once DSP has been sampled at the end of the power-on-reset cycle, the active clock edge for SCLK remains selected until either a rising edge on DSP is detected or power is removed from the device.

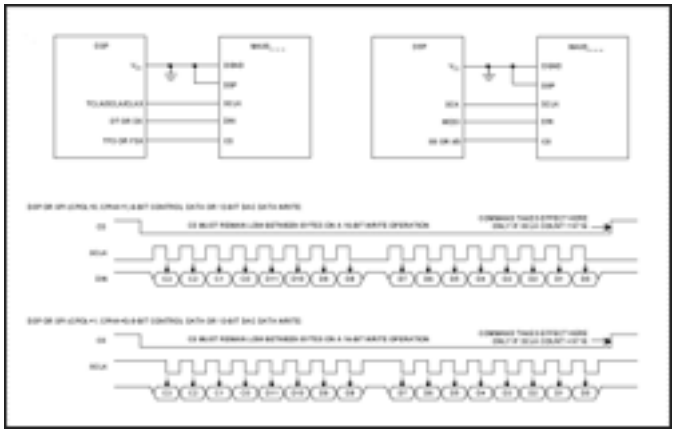
Figure 1 illustrates the CS, SCLK, and DIN patterns required to communicate with the device when data is being clocked in on the rising edge. Drive CS low and clock serial data at DIN into the input shift register on the rising edge of SCLK. Data or command writes take effect when CS is returned high after an integer multiple of 16 (ie. N*16) SCLK pulses have been received. If CS is brought high before N*16 SCLK cycles, the write is ignored.



For Larger Image

Figure 1. Single Device Write- Data Clocked In On Rising Edge of SCLK

Figure 2 illustrates the CS, SCLK, and DIN patterns required to communicate with the device when data is being clocked in on the falling edge of SCLK. Drive CS low and clock serial data at DIN into the input shift register on the falling edge of SCLK. Data or command writes take effect when CS returns high after an integer multiple of 16 (ie. N*16) SCLK pulses have been received. If CS is brought high before N*16 SCLK cycles, the write is ignored.



For Larger Image

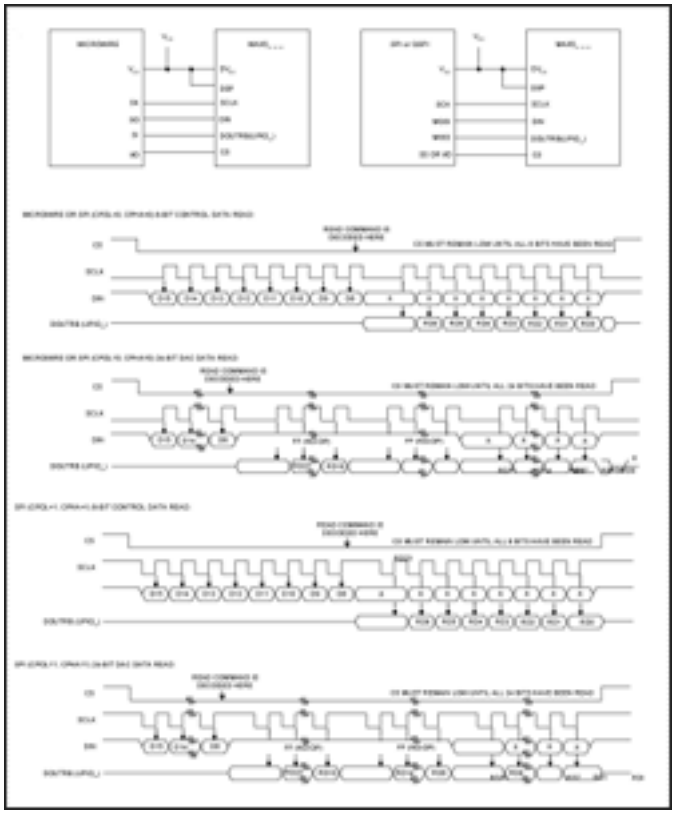
Figure 2. Single Device Write- Data Clocked In On Falling Edge of SCLK

Read Operation from a Single Device

The serial interface for this family of devices is capable of supporting several options for read commands. In most cases the output data from a read command is 8 bits, resulting in a total command sequence of 16 bits (8 bits for the command and 8 bits for the output data). For a read operation of the DAC data, the interface outputs the data from both the input (12-bit) and DAC (12-bit) registers for the selected channel. The result is 24 bits of output data and a total command sequence of 32 bits.

To read data from the device, configure UPIO1 or UPIO2 as DOUTRB (DOUT for Read Back). The interface samples the DSP input at power-up to determine which clock edge will be used to transmit serial data from DOUTRB. If DSP is connected to DVDD at power-up, data is clocked in on the rising edge of SCLK and clocked out on the falling edge of SCLK. If DSP is connected to DGND at power-up, data is clocked in on the falling edge of SCLK and clocked out on the rising edge of SCLK.

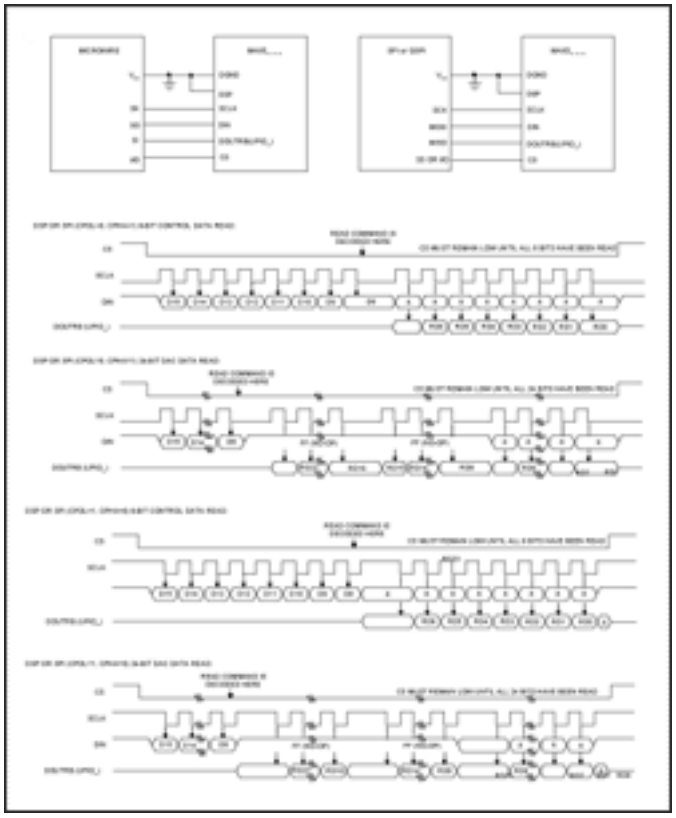
Figure 3 illustrates the CS, SCLK, and DIN patterns required to communicate with the device when data is clocked in on the rising edge of SCLK. Drive CS low and clock the 8-bit read command at DIN into the input shift register on the rising edge of SCLK. Data appears on DOUTRB during the next 8 or 24 SCLK cycles, depending on the read length. A 24-bit read requires two extra NO-OP command (0xFF) writes following the 8-bit read command in order to keep the last two bytes of data flowing from DOUTRB. Output data on DOUTRB changes on the falling edge of SCLK and is valid on the rising edge of SCLK. Keep CS low until the entire read operation is complete.



For Larger Image

Figure 3. Single Device Read- Data Clocked In On Rising Edge of SCLK

Figure 4 illustrates the CS, SCLK, and DIN patterns required to communicate with the device when data is clocked in on the falling edge of SCLK. Drive CS low and clock the 8-bit read command at DIN into the input shift register on the falling edge of SCLK. Data appears on DOUTRB during the next 8 or 24 SCLK cycles, depending on the read length. A 24-bit read requires two extra NO-OP command (0xFF) writes following the 8-bit read command in order to keep the last two bytes of data flowing from DOUTRB. Output data on DOUTRB changes on the rising edge of SCLK and in valid on the falling edge of SCLK. Keep CS low until the entire read operation is complete.



For Larger Image

Figure 4. Single Device Read- Data Clocked In On Falling Edge of SCLK

Write Operation to Multiple Devices in a Daisy-Chain

The serial interface for this family of devices is capable of supporting a wide variety of daisy-chain configurations. Data is clocked into each device in the chain on either rising or falling clock edge. As a result, the serial data from the previous device in the chain may be provided on either clock edge. The serial interface provides flexibility for applications requiring the same clock phase or an alternation of clock phase through the daisy-chain. A number of board level considerations may affect the choice of clocking scheme including the following:

- Clock Skew - If there is a chance of significant clock skew between device A and device B, consider applying opposite edge data transfers (e.g. out of device A on the rising edge of SCLK and into device B on falling edge of SCLK). This is robust to clock skew, but may be difficult to implement at the maximum clock rate supported by the serial interface without careful board level design.
- Clock Rate - If it is essential to daisy-chain at the maximum clock rate, consider running the daisy-chain with a common clock edge throughout the chain. However, the board environment must be designed carefully before selecting this option. Avoid long data paths, significant device-to-device clock skew, or wide variations in board conditions (temperature, supply voltage, etc.) in order to ensure proper operation of the daisy-chain.
- Clock Duty Cycle - An alternating clock scheme through the daisy-chain relies on reasonable duty cycles for the clock signal since the half clock period is used when transferring data from one device to the next. A daisy-chain with a slow clock but a very low or high duty cycle could still fail if not designed carefully.

- Slow Clock or Data Edges - Inadequate drive of the clock signal used in the daisy-chain will result in slow rise and fall times. This can be treated as a special case of clock skew and may cause different devices in the daisy-chain to see the clock event to occur at different times.

Configure UPIO1 or UPIO2 of the device as a DOUTDC0 (DOUT for Daisy-Chaining- Mode 0) or DOUTDC1 (DOUT for Daisy-Chaining- Mode 1) output. In Mode 0, data at DOUTDC0 changes on the falling edge of SCLK and is valid on the rising edge of SCLK. In Mode 1, data at DOUTDC1 changes on the rising edge of SCLK and is valid on the falling edge of SCLK.

In a daisy-chain, the first device in the chain gets its DIN from the bus master. Subsequent devices in the chain get their DIN from the DOUTDC_ output of the preceding device. In order for a device A to pass daisy-chain data through its serial interface to device B, CS signal for device A must remain low (no rising edge) after receiving the 16-bit command sequence. The data clocked out of DOUTDC_ is the same as the data clocked into DIN delayed by 16 clock cycles.

All devices in the daisy-chain will perform the write command stored in the serial register at the rising edge of CS (see *Interaction with Daisy-Chain and Read Back* section for read commands). If there is no rising edge of CS (CS remains low) before the next write command is clocked into a device, then the new write command will overwrite the old write command in the input shift register. This allows data to be passed through a device to other devices in the chain without causing any change of state. Apply a NO-OP (0xFF) command to those devices which should be unaffected.

It is possible to control both the input clock edge and the output clock edge which means that for successful daisy-chaining, the data may need to be delayed by 15.5, 16, or 16.5 clock periods. The CPOL and CPHA control bits set the DIN to DOUTDC_ delay through a given device to ensure correct operation through the chain regardless of clock configuration from one device to the next.

CPOL and CPHA have similar definitions to those used in descriptions of SPI interfaces:

- CPOL describes the clock polarity
- CPHA describes whether there is a leading clock edge before the first active clock edge in any given sequence

The programmable CPOL and CPHA bits combined with the DSP input cause serial daisy-chain output data on DOUTDC1 or DOUTDC0 to be delayed by the appropriate amount. See Table 1.

Table 1. DSP, CPOL, and CPHA Settings

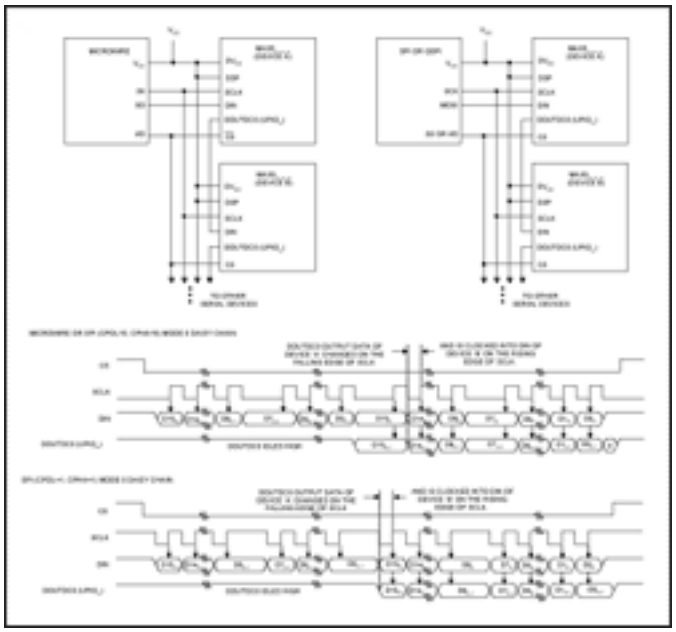
DSP	CPOL	CPHA	DOUTDC1 (ALWAYS CLOCKED OUT ON RISING EDGE OF SCLK)*	DOUTDC0 (ALWAYS CLOCKED OUT ON FALLING EDGE OF SCLK)*	COMMENT
DGND	0	1			DIN clocked in on falling edge of SCLK. Power-up state of CPOL/CPHA for this DSP connection.
DV_{DD}	0	0			DIN clocked in on rising edge of SLCK. Power-up state of CPOL/CPHA for this DSP connection.
DGND	0	0	Invalid	Invalid	This combination is unused.
DGND	0	1	Delay of 15.5 clocks from SCLK falling edge (active edge for DIN) to SCLK rising edge (active edge for DOUTDC1)	Delay of 15 clocks from SCLK falling edge (active edge for DIN) to SCLK falling edge (active edge for DOUTDC0)	
DGND	1	0	Delay of 15.5 clocks from SCLK falling edge (active edge for DIN) to SCLK rising (active edge for DOUTDC1)	Delay of 16 clocks from SCLK falling edge (active edge for DIN) to SCLK falling edge (active edge for DOUTDC0)	
DGND	1	1	Invalid	Invalid	This combination is unused.
DV_{DD}	0	0	Delay of 16 clocks from SCLK rising (active edge for DIN) to SCLK rising (active edge for DOUTDC1)	Delay of 15.5 clocks from SCLK rising (active edge for DIN) to SCLK falling (active edge for DOUTDC0)	
DV_{DD}	0	1	Invalid	Invalid	This combination is unused.
DV_{DD}	1	0	Invalid	Invalid	This combination is unused.

DV_{DD}	1	1	Delay of 15 clocks from SCLK rising (active edge for DIN) to SCLK rising (active edge for DOUTDC1)	Delay of 15.5 clocks from SCLK rising (active edge for DIN) to SCLK falling (active edge for DOUTDC0)	

**Data is always clocked out of DOUTDC_ on the valid (rising for DOUTDC0 and falling for DOUTDC1) edge of SCLK.*

Figure 5 provides the CS, SCLK, DIN, and DOUTDC0 patterns for writing to multiple devices in one configuration of daisy-chaining, where DIN is clocked in on the rising edge of SCLK in each device in the chain and DOUTDC0 is used to pass data from one device to the next.

To perform the daisy-chain write operation, drive CS low to select all devices in the daisy-chain. Data from the bus master or DOUTDC0 of the previous device is clocked into DIN of each cascaded device. The devices are unaffected while data passes through the input shift register as long as CS remains low. When the data has propagated through the daisy-chain, drive CS high to load the DAC commands stored in the input shift registers of all the daisy-chained devices simultaneously, once again assuming an integer multiple of 16 (i.e. N*16) SCLK pulses have occurred. If CS goes high before N*16 SCLK cycles, every device in the daisy-chain ignores the write.



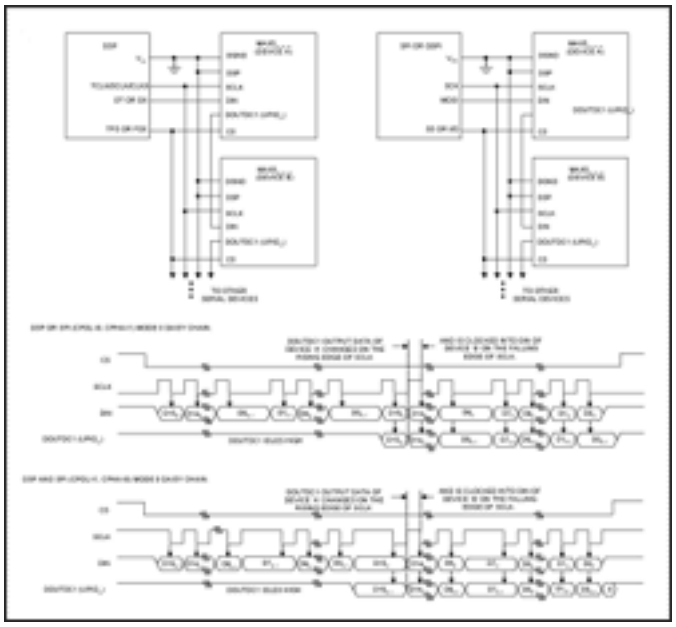
For Larger Image

Figure 5. Daisy-Chaining- Data Clocked In On Rising Edge of SCLK

Figure 6 provides the CS, SCLK, DIN, and DOUTDC1 patterns for writing to multiple devices in one particular configuration of daisy-chaining, where DIN is clocked in on the falling edge of SCLK in each device in the chain and DOUTDC1 is used to pass data from one device to the

next.

To perform the daisy-chain write operation, drive CS low to select all devices in the daisy-chain. Data from the bus master or DOUTDC1 of the previous device is clocked into DIN of each cascaded device. The devices are unaffected while data passes through the input shift register as long as CS remains low. When the data has propagated through the daisy-chain, drive CS high to load the DAC commands of all the daisy-chained devices simultaneously, once again assuming an integer multiple of 16 (i.e. N*16) SCLK pulses have occurred. If CS goes high before N*16 SCLK cycles, every device in the daisy-chain ignores the write.



For Larger Image

Figure 6. Daisy-Chaining- Data Clocked In On Falling Edge of SCLK

Daisy-Chaining Example

The following example sets up a daisy-chain consisting of three devices (A, B, and C). Device A is the closest to the bus master. Each device will be configured differently from the other two.

1. Bring CS low to configure device A. Send a command from the bus master to device A to configure its UPIO1 as DOUTDC0 (COMMAND 1). This command is stored in device A's input shift register. Return CS high to perform the command stored in the input shift register of device A. Device A is now configured.
2. Bring CS low to configure device B. Send a command from the bus master that will pass through device A to configure device B's UPIO2 as DOUTDC1 (COMMAND 2). Device B's input shift register will begin updating on the clock cycle immediately following this 16-bit write.
3. Send a command from the bus master to device A to overwrite its input shift register data so that the state of device A remains unchanged (COMMAND 3). Any command on device A may be used in place of the NO-OP if a specific operation is desired (i.e. UPIO1 configuration, changing speed bits, etc). Device B's input shift register updates with the

previous command during these 16 clock cycles. Return CS high to perform the commands stored in the input shift registers of devices A and B. Devices A and B are now configured.

4. Bring CS low to configure device C. Send a command from the bus master that will pass through devices A and B to configure device C's UPIO1 as DOUTDC1 (COMMAND 4).
5. Send a command from the bus master through device A to overwrite the input shift register data of device B so that the state of device B remains unchanged (COMMAND 5). Device B's input shift register updates with command from step 5.
6. Send a command from the bus master to device A to overwrite its input shift register data of device A so that the state of device A remains unchanged (COMMAND 6). Device B's input shift register updates with command from step 6. Device C's input shift register updates with command from step 5. Return CS high to perform the commands stored in the input shift register of devices A, B, and C.

Table 2. Device A, B, C Input Shift Register Contents- Daisy-Chaining Example

STEP	DEVICE A (FROM BUS MASTER)	DEVICE B (FROM DOUTDC_ OF DEVICE A)	DEVICE C (FROM DOUTDC_ OF DEVICE B)	RISING EDGE OF CS?
1	COMMAND 1	-	-	YES
2	COMMAND 2	-	-	NO
3	COMMAND 3	COMMAND 2	-	YES
4	COMMAND 4	-	-	NO
5	COMMAND 5	COMMAND 4	-	NO
6	COMMAND 6	COMMAND 5	COMMAND 4	YES

Interaction Between Daisy-Chain and Read-Back

The serial interface for each of the devices in this family has 2 UPIO ports. This means that one may be configured for read-back (DOUTRB) and the other could be used for daisy-chaining (DOUTDC1 or DOUTDC0). This is a perfectly valid combination and in most cases the read-back and daisy-chaining will function normally. In two instances, using this configuration has some consequences.

Daisy-Chain with Data Read-Back Example

This example uses two devices: A and B, with A being the closest to the bus-master. If device A has DOUTRB and DOUTDC_ and device B has DOUTRB, then the following would be a valid sequence:

1. Bring CS low. Send an 8-bit read command (COMMAND 1) from the bus master to device A. Keep CS low. Do not return CS high.
2. Send some other read or write command (COMMAND 2) from the bus master to device A. Device B's input shift register updates with COMMAND 1. Valid output data resulting

from COMMAND 1 appears on DOUTRB of device A. Return CS high to complete the command sequence.
3. Valid output data resulting from COMMAND 1 appears on the DOUTRB of device B. Device A executes COMMAND 2.

In this case the 8-bit read command intended for device B passes through device A. While the command passes through device A, the DOUTRB port on device A responds to the read command (giving valid read data). This does not prevent the read command from being passed to device B, and device B also responds to the read command with valid data.

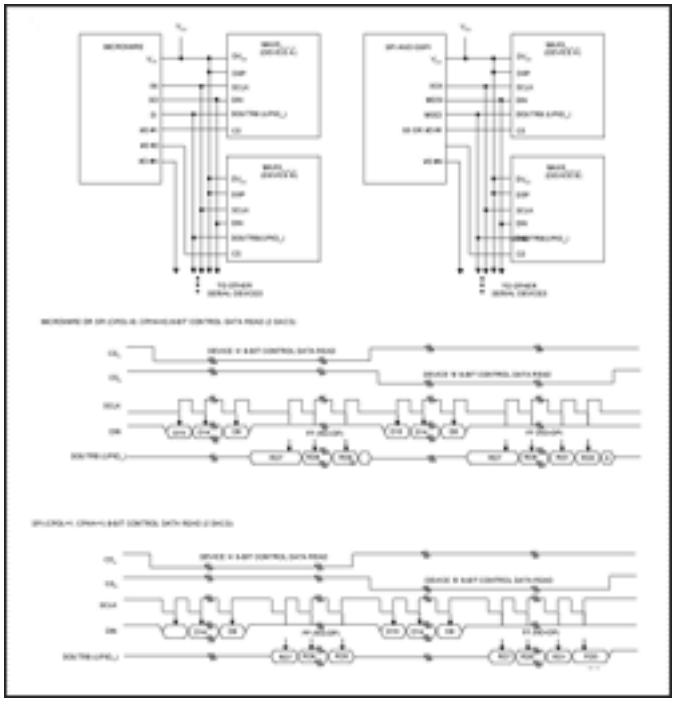
The second example of interaction is with a 24-bit DAC read from only device A (the one closest to the bus master), this command would work correctly. With the exception of the first device in the daisy-chain, 24-bit reads are not compatible with daisy-chaining. This is because the first device in the daisy-chain converts a 24-bit read into a NO-OP when passing the data out on DOUTDC_. Also, a 24-bit read is a 32-bit total command sequence which is incompatible with the 16-bit command sequence required by the daisy-chain.

Direct Write and Read Access to Multiple Devices

Writing to or reading from multiple devices is possible over a shared 3+N wire serial interface as illustrated in Figure 7 and Figure 8. N is the number of devices sharing the serial interface as well as the number of distinct chip select lines since each device has its own CS. Write and read operations follow the protocols described in earlier sections. The SCLK, DIN, and DOUTRB signals are common to all devices. The two main differences between this method and daisy-chaining are 1) each device has its own CS and 2) DIN is common to all devices.

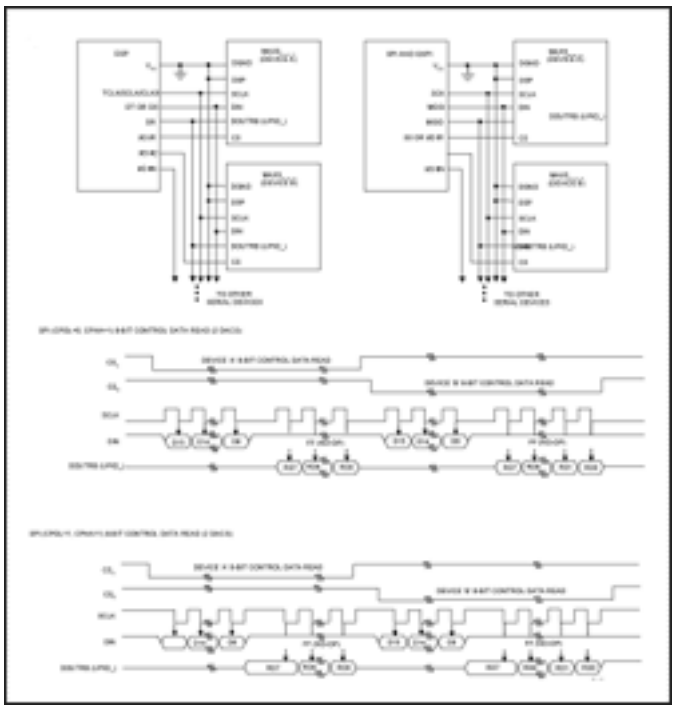
Consider the following when connecting multiple devices to a shared serial interface:

- The same data can be broadcast to several devices with a single write operation by asserting multiple chip selects at the same time.
- Only one device's DOUTRB can be active at one time during data read back to avoid contention. On an unselected device, DOUTRB is high impedance, allowing other devices to drive the bus.
- Loading caused by multiple devices connected to individual signals (SCLK, DIN, DOUTRB) slows down the serial interface. The speed reduction depends on external factors such as the PCB layout and the number of devices that are on the shared bus.
- There is typically software overhead involved with accessing multiple devices since the CS lines are usually driven from microcontroller or DSP ports. This is in contrast to single DAC applications that may have CS permanently connected low or controlled by a DSP's hardware-based frame sync signal.



For Larger Image

Figure 7. Multiple Device Reads- Data Clocked In On Rising Edge of SCLK



For Larger Image

Figure 8. Multiple Device Reads- Data Clocked In On Falling Edge of SCLK

DSP Frame Sync Mode

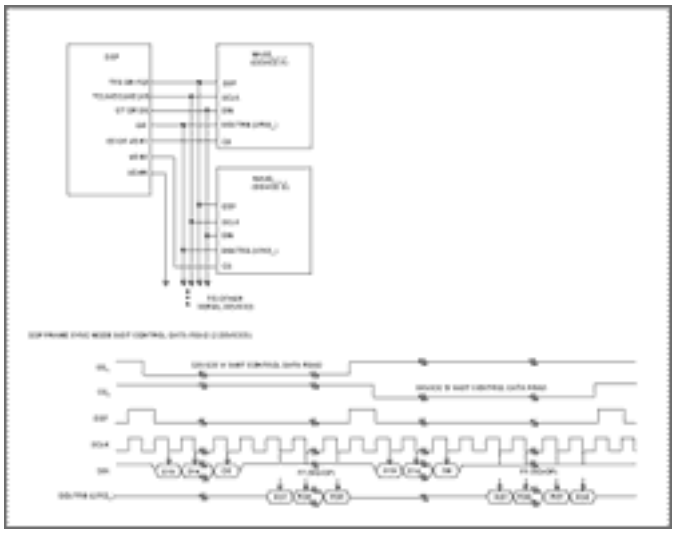
Read and write operations in DSP Frame Sync mode are similar to the protocols described in previous sections, but there are two major differences. First, the beginning of a write operation is referenced to a falling edge of DSP rather than CS (which must still be low in DSP Frame Sync

mode in order to select the device). Second, a 16-bit DAC data or command write takes effect 16 SCLK periods after the falling edge of DSP. The CS rising edge condition required to execute a command in the other modes is not required in the DSP Frame Sync mode.

Note that DSP Frame Sync mode is not compatible with daisy-chaining since the CS rising edge is no longer available to tell a device to act on a certain 16-bits of data instead of passing it through to the next device.

Read and write operations to multiple devices using DSP Frame Sync mode can be performed using a shared 4+N wire serial interface. N is the number of devices sharing the serial interface as well as the number of distinct chip select lines since each device has its own. The device enters DSP Frame Sync mode after power-up on the first rising edge of DSP. To avoid this mode, connect DSP to DVDD or DGND at power-up, as illustrated in the previous sections.

The main advantage of the DSP Frame Sync mode is that a single hardware signal (frame sync) controls the timing of DSP accesses of multiple devices on the serial bus. This provides a performance advantage since software control of the chip select is no longer the critical factor determining the timing of DAC operations. One example application that benefits from this feature is a multi-chip CODEC where a common frame sync signal drives a (constantly enabled) DAC and ADC at the same rate. The DSP software fills a buffer with DAC data and retrieves a buffer full of ADC data in a single shot, and it allows the serial interface hardware to handle the full-duplex data transfer.



For Larger Image

Figure 9. DSP Frame Sync Mode- Multiple Reads

More Information

MAX5290: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5291: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5292: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5293: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5294: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5295: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5580: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5581: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5582: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5583: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5584: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5585: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5590: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5591: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5592: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5593: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5594: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX5595: QuickView -- Full (PDF) Data Sheet -- Free Samples