

## How does the serial interface operate?

The serial interface on this family of converters is implemented as a state machine. The interface works by counting clocks on each data transfer. Therefore, if you are performing a write operation to one of the ADC's 8-bit registers, 8 SCLK cycles must be applied to the ADC and, the converter will transfer in the 8 bits of DATA on the DIN line on each of the eight SCLK cycles. When this operation is complete, the part returns to a state in which it expects the next operation to be a write operation to the communications register. This happens regardless of which register has been accessed and whether a read or write operation has taken place. The ADC knows how many clock cycles should be in a transfer to/from a particular register and in this way knows when the transfer is complete.

The first bit of the communications register is a gating bit that must be set to 0 to access the communications register. If there is a 1 on the DIN line when the part is expecting a write to this bit, the part effectively wraps itself around to monitoring that bit. If there is a 0 on the DIN line when the part is expecting a write to this bit, the part continues to load the next 7 bits of the communications register on the next 7 SCLK cycles.

The serial interface on these converters consists of five signals, CS, SCLK, DIN, DOUT and DRDY. The DIN line is used for transferring data into the on-chip registers while the DOUT line is used for accessing data from the on-chip registers. SCLK is the serial clock input for the device and all data transfers (either on DIN or DOUT) take place with respect to this SCLK signal. The DRDY line is used as a status signal to indicate when data is ready to be read from the converter's data register. DRDY goes low when a new data word is available in the output register. It is reset high when a read operation from the data register is complete. It also goes high prior to the updating of the output register to indicate when not to read from the device. This is to ensure that a data read is not attempted while the register is being updated. CS is used to select the device. It can be used to decode the individual converters in systems where a number of parts are connected to the serial bus.

The serial interface can operate in three-wire mode by tying the CS input low. In this case, the SCLK, DIN and DOUT lines are used to communicate with the converter and the status of DRDY can be obtained by interrogating the MSB of the Communications Register. This scheme is suitable for interfacing to microcontrollers. If CS is required as a decoding signal, it can be generated from a port bit. For microcontroller interfaces, it is recommended that the SCLK idle high between data transfers.

These converters can also be operated with CS used as a frame synchronization signal. This scheme is suitable for DSP interfaces. In this case, the first bit (MSB) is effectively clocked out by CS since CS would normally occur after the falling edge of SCLK in DSPs. The SCLK can continue to run between data transfers provided the timing numbers are obeyed.

## Are there any precautions that should be taken to make the interface more robust?

The interface implementation on these converters is basically a state machine that counts clock pulses and defaults to waiting for a write to the communications register when an operation is complete. When the communications register is written to, specifying a write to the Setup register for example, the ADC knows that 8 clocks are required to clock in the data. Spurious clocks on the interface cause the interface to lose synchronisation, leading to wrong registers being addressed and thereby corrupting the interface. Tying DIN high between write operations prevents invalid data being written to the ADC.

The first bit in the write operation to the communications register is a gating bit that must be '0' to allow the remaining 7 bits to be clocked into the register to specify the next operation. To avoid the possibility of interface issues due to spurious clocks, it is advised to take the DIN line of the converter to a logic high as soon as each write to the converter is complete. Since these converters default to waiting for a write to the communications register, taking DIN high when it has completed a sequence prevents invalid data being written to the communications register if spurious clocks occur. At this time, the ADC is waiting for a zero on DIN before it enables further data into the register. If the part is set up for a write to the communications register with the DIN high, it is effectively immune to spurious serial clocks. This will not prevent spurious clocks received during a write operation from corrupting the interface.

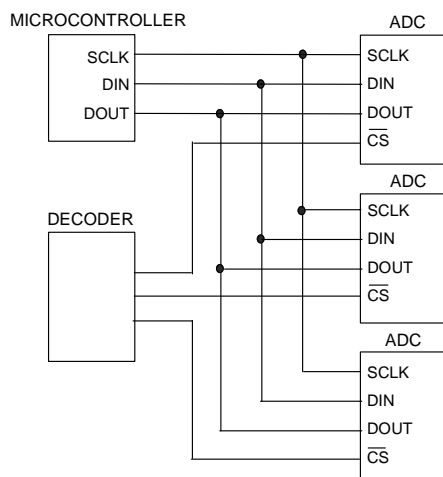
## If the interface does get corrupt what are the options to regain control of the ADC?

There are two options that can be used to regain control of the interface on these ADCs. Writing 32 ones to the interface will reset the serial interface into a known state where the ADC is waiting for a write to the communications register. This does not reset the contents of any of the internal registers to their default conditions so, therefore, all registers need to be reconfigured following the interface reset. This is a software method of regaining control of the interface.

The second option is to pulse the RESET line to the ADC. This resets the interface and all of its internal registers to their power-on-reset values. This is a hardware method of resetting the complete ADC to its default status.

## How do I interface to multiple ADCs over the same serial interface?

A single microcontroller/DSP can be used to communicate with several AD7714/15 devices. The  $\overline{CS}$  input of the ADC can be used to enable or disable the serial interface of the ADC. By controlling the  $\overline{CS}$  inputs to the ADCs using a decoder, the microcontroller/DSP can communicate with each ADC individually or simultaneously. The figure below shows the interface between a microprocessor and several ADCs. The  $\overline{CS}$  input of each ADC is connected to the decoder. Using the decoder, the microprocessor can select the ADC with which it wants to transfer data/instructions. When  $\overline{CS}$  is high, the serial interface of the ADC is disabled and it ignores any activity on the data bus. To communicate with the ADC, its  $\overline{CS}$  line can be taken low. The ADC will then have access to the data bus between itself and the microprocessor. The datasheet should be consulted for timing specifications.



### Interfacing several ADCs to a Single Microcontroller

**When reading from the AD7714/15, only the first read after power up is successful, all subsequent read operations give invalid results. What could be happening?**

It is probable that the digital interface is entering an unexpected state following the read operation. When using the digital interface of the AD7714/15, it is critical that the correct number of clock cycles is provided when reading from or writing to the control registers. If one extra clock cycle is provided and a write operation is then attempted, the part will enter an unexpected state. Similarly, if less than 16 clocks are provided when reading from a 16-bit register, the interface will lose synchronization.

Another probable cause is spurious clocks on the SCLK line. The first bit of any write to the communications register is a zero. The ADC will ignore bus activity until it receives a zero. If DIN is left low after performing a read, any spurious clocks on the SCLK line will clock a bit into the communications register. The next time that eight bits are written to the communications register, only the first seven bits will be loaded and the eighth bit will be interpreted as the first bit of a subsequent write. This problem can be avoided by pulling DIN high after each write.

If the  $\overline{CS}$  line is being used as a Frame Synchronization signal, ensure that this line is brought high at the correct time. Also ensure that the correct number of clock pulses have occurred before  $\overline{CS}$  is brought high.

If the interface does lose synchronization, writing 32 1s into the device will reset the interface. The registers on board the ADC should also be reconfigured as they may have been overwritten with invalid data while the interface was asynchronous.

**What is the function of the CLK bit in the Clock register and how does it influence the performance of the ADC?**

Refers to the AD7715 only, the AD7714 does not have a CLK bit. The CLK bit should be set to '1' if the Master clock frequency is greater than 2MHz and should be cleared to '0' if the master clock frequency is equal to or less than 2MHz. If this bit is set as recommended in the datasheet, the part will meet all the datasheet specifications in terms of power consumption, noise, accuracy and output update rate. If the CLK bit is set incorrectly, the part will not meet the datasheet specifications.

The CLK bit is used to boost the current within the ADC when it is operating with the higher master clock so that the device will meet the datasheet specifications. If the part is used with a master clock less than 2 MHz and the CLK bit is set to 1, the part will still operate but it will draw more power than necessary. Also, the filter notches may deviate from the locations described in the datasheet. If the AD7714/15 is operated with a master clock in excess of 2MHz and the CLK bit equals 0, the modulator will not be supplied with enough current and the settling time for the modulator will increase. At frequencies slightly over 2MHz (with CLK = 0), gain errors will occur and the noise will increase. As the frequency is increased further, the part will eventually stop working as the modulator will refuse to operate with the reduced current.

## What is the sequence of events in switching between channels for the AD7714?

Every sequence of events with this converter commences with a write to the communications register to specify the next operation. The sequence of events for a channel switching operation using the AD7714 is as follows (channels AIN1-AIN2 and AIN3-AIN4 being used). Every sequence of events with this converter commences with a write to the communications register to specify the next operation to be performed. When the specified operation is complete, the interface defaults to waiting for

**Write 5C Hex to Communications Register.** This specifies the next operation as a read of the DATA Register for channel AIN1-AIN2.

**Poll DRDY.** Poll the DRDY pin to determine if valid data is available in the data register. When DRDY goes low, apply 24 serial clocks to clock the data from the ADC. A read should not be initiated when DRDY is high.

**Write 5D Hex to Communications Register.** When the previous read is complete, the ADC reverts to waiting for a write to the communications register. This write specifies that the next operation is a read from the data register of channel AIN3-AIN4. The channel bits specify the new channel input and switch the input multiplexer. This will cause the DRDY output to be set high and it will remain high until valid data is available in the data register.

**Poll DRDY.** Poll the DRDY pin to determine if valid data is available in the data register and, when it goes low, apply 24 serial clocks to clock the data from the ADC. A read should not be initiated when DRDY is high.

Continually running through this loop will allow data to be read from each channel in sequence. The throughput time will be 1/3 of the programmed update rate due to the settling time associated with the filter.

## The evaluation board fails to interface via the printer port what needs to be checked?

All evaluation boards are fully tested as part of our evaluation board production process and should interface directly via the PC printer port without any issues. After power up, ensure that communication with the evaluation board has been established by attempting to read the default values from one of the registers. When evaluating the AD7714, for example, read from the calibration registers after power up and reset. The word 1F4000H should be output from the zero-scale register and 5761ABH from the full-scale register.

If the interface does not operate, there may be a problem running the evaluation board software with Windows NT because that operating system tends to restrict the software access to hardware such as the printer port. The software definitely runs on windows95 and Windows98 but does not operate on Windows 2000. It is also worth configuring the PC printer port as an output only port. The input lines used on the evaluation boards use the dedicated input lines that cannot be re-configured. An issue may arise with the other lines if they have been re-configured by other pieces of software used in configuring scanners etc.

The following is a quick debug that will also help to source the issue.

Use an oscilloscope probe to check the SCLK pin of the device. This comes into the board on pin 5 of the printer port connector. Load the software and click on the "Program Calibration Coefficients" button. This will cause the software to write to the ADC and then attempt to read the calibration data back. Going in and out of this section will repeat the write/read sequence. A burst of clock pulses should appear on the SCLK pin.

If the SCLK pulses do not occur, it is possible that the printer port has been set as an input port and any attempt to write to it is being ignored. To overcome this, set the port to AT/Unidirectional mode. This must be done in the BIOS software. The method used to edit the BIOS depends on the PC but it is normally some combination of keystrokes when the PC is booting up (typically DEL or CTRL + ALT + ENTER).

If the SCLK pulses are present but the data is read back as all 1's or all 0's, there may be a fault on the board or in the printer port. This can be checked by tracing the SCLK signal from the edge connector to the part and tracing the SDATA signal from the part to the edge connector.