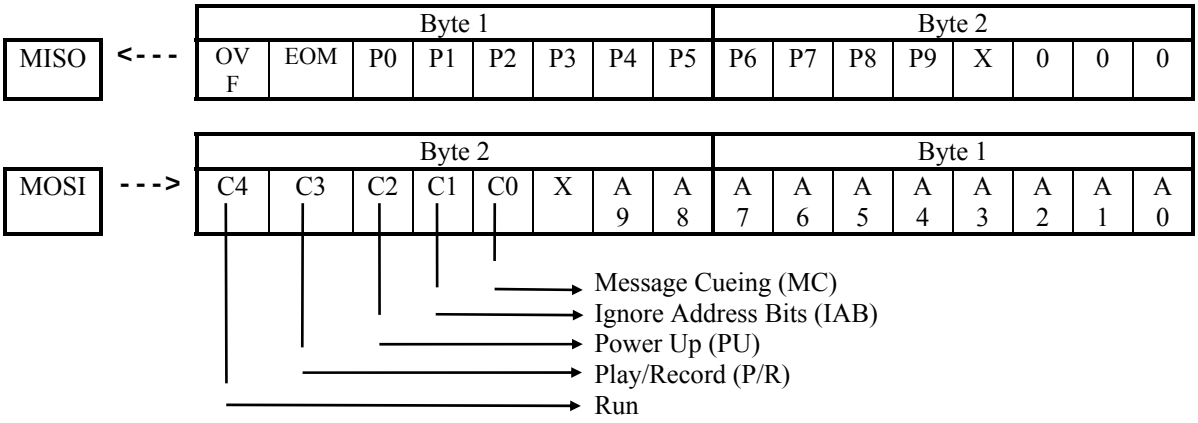
A Discussion of the ISD4000 SPI Control Port
Operations, Tricks and Techniques

A simple but powerful command structure is built into the ISD4000 SPI control port. It's inherent flexibility allows the software programmer to direct the operation of the ISD4000 device with the minimum number of control cycles while allowing full control of the device in it's several modes. The following paragraphs give some tips on the use of this port and suggest some methods of simplifying some operations.

The SPI command registers in the ISD4000 have the following composition:



The MOSI pin of the SPI port on the device is an input. MOSI stands for Master Out, Slave In. The MISO is an output. MISO stands for Master In, Slave Out. The ISD4000 is operating as a "slave" device to the microcontroller that is running the system. The SPI port also has a serial clock input called SCLK and a select input called SS. The SS stands for Slave Select and is an active LOW signal.

All input command cycles in the ISD4000 start with SS pin going LOW; and end with SS pin going HIGH. The bit position of the input data at the time the SS pin goes HIGH determines how the device reacts to the command. Any number of bits may be clocked into the SPI port on an input cycle. The last 5 bits input to the MOSI pin at the time the SS pin goes HIGH determine the command being input, even if many bytes were clocked into the port while the SS pin was LOW. Only the last 16 bits shifted into the SPI port are retained in the SPI port hardware.

The SCLK signal clocks the data into the ISD4000. The input data to the MOSI pin must be valid on the rising edge of the SCLK. The output data from the MISO pin changes following the falling edge of the SCLK pin. Consult the data sheet on the ISD4000 for the exact timing. The examples stated below assume the initial state of the SCLK pin is LOW when the SS pin goes LOW to start the command cycle but this is not required.

Example #1, Power up the device - The following is an example of a simple command to the ISD4000 SPI port. In this operation, the device is to be powered up so that a Record or Playback cycle can be started later. Consult the diagram at the beginning of this Application Note that shows the SPI command registers. This drawing shows the bit positions of the 5 bits that compose the heart of the controls, e.g. C0 through C4. This diagram should be used as a reference when reading the explanations below:

1. The device is powered up by setting the PU bit. That is, bit C2 must be HIGH and bits C0, C1, C3 and C4 must all be LOW at the end of the command cycle. The 5 control bits in the SPI control register should then be set as <00100> where the bit positions are as indicated by the diagram above.

2. In this example, an entire byte will be shifted in, since some hardware SPI ports can only operate on whole bytes. It is necessary therefore to shift in HEX address 20, <0010 0000> just to set the C2 bit. Note that the state of the A9, A8 and "X" bits during this operation do not matter. This will be explained in more detail later.

3. The command cycle starts by changing SS to a LOW. The input to the MOSI pin is also LOW. The input byte, <0010 0000>, is to be clocked in, right to left. The first 5 bits clocked in, therefore, are all LOW. A single HIGH bit is then clocked in followed by two more LOW bits. This sequence is shown graphically below. The command ends with the SS pin going back HIGH and the ISD4000 will then begin a power up cycle. In T_{PUD}[1] time, the device will be powered up.

The table below shows the power up bit data being shifted from the command byte in the microcontroller into the MOSI pin of the ISD4000 and into byte 2 of the input side of the SPI input port register. Note that the PU bit is shifted to the right out of the microcontroller and into the ISD4000 SPI port.

Shift #	command byte in microcontroller		SPI port in ISD4000 after "N" shifts	Shift #	command byte in microcontroller		SPI port in ISD4000 after "N" shifts
Initial condx.	<0010 0000>	--->	<0000 0000>				
1	<0001 0000>	--->	<0000 0000>	5	<0000 0001>	--->	<0000 0000>
2	<0000 1000>	--->	<0000 0000>	6	<0000 0000>	--->	<1000 0000>
3	<0000 0100>	--->	<0000 0000>	7	<0000 0000>	--->	<0100 0000>
4	<0000 0010>	--->	<0000 0000>	8	<0000 0000>	--->	<0010 0000>

In this first example, we did not care about the value of the address bits A0 through A9. In fact, since the Run bit, C4, was input as a LOW only the HIGH PU bit had any effect on the ISD4000 and the state of C0, C1 and C3 also did not matter. *ISD recommends that anytime data is shifted into a SPI port command register bit position that "does not matter" it should be a "0" for compatibility with possible future features that may be added to the device.*

Example #2, Record at an address - This second example assumes the ISD4000 is already powered up and that recording is to begin at a specific address. This example we will use address 92, or HEX address 5C, <00 0101 1100>. Each control bit in the input side of the command register of the device is set up as follows:

C0 - The MC bit must be LOW since a message cueing cycle is not desired.
C1 - The IAB bit must be LOW since the address is **not** being ignored.
C2 - The PU bit must remain HIGH since the device is already powered up and must remain powered up after this command cycle.
C3 - The P/R bit must be LOW since a Record cycle is to be started.
C4 - The RUN bit must be HIGH since an active cycle is to be started.
A0 - A9 - The address bits must be all be defined since the IAB bit is LOW.

1. The transfer of the two bytes of data is begun by changing the SS pin to a LOW. The bytes to be sent are:
HEX address A0 5C, < 1010 0000> <0101 1100>.

2. Again, we are clocking the bits in, right to left[2]. Therefore, the first bit clocked into the SPI port will be the A0 address bit which is a "0," followed by A1 which is a "0," followed by A2 which is a "1" etc.

[1] T_{PUD} is the power up delay time of the device. See the ISD4000 data sheet.

[2] Note that some microcontrollers with hardware SPI ports only have the ability to do a left to right shift. Consult the µP data sheet for details. If this is so, then the bit placement of the data shifted in would have to be inverted. In this

3. The last of the 16 bits clocked into the SPI port will be the RUN bit which, of course, is a "1."

4. The command cycle ends with the SS pin going back HIGH. At this time, the Record cycle starts at the address as specified by A0 through A9.

5. One additional command must be given to the device so that it will proceed through the chip's memory as desired. The IAB bit in the command register must be set to enable the address sequencer internal to the ISD4000. Otherwise the device will do a looping record at the address specified by A0 through A9. This second command cycle must be given before the chip reaches the end of the row where recording begins. This allows at least 150 milliseconds to send this command.

6. The command byte to be input must be one that does not interrupt the record operation in progress but sets the IAB bit. Additional address information is not needed. If it is assumed that the SPI port of the microcontroller can input any number of bits in a control cycle, this operation only requires that 5 bits be input to the SPI port. The proper control bits to send are the same as originally sent in the first control operation with the exception that the IAB bit is now set. It is therefore necessary to only send <1011 0> to the SPI port.

7. Once again an SPI cycle is begun by taking SS LOW. The first bit shifted into the ISD4000 SPI port is C0, which is a "0." The fifth and last bit shifted into the SPI port is C4, the RUN bit, which is a "1." The control sequence ends with the SS pin going back HIGH and results in recording proceeding through the memory in the ISD4000 row by row until we tell the ISD4000 to stop.

Note: *After each command cycle, the data shifted into the ISD4000 remains in the chip's SPI port hardware. As subsequent operations shift more data in, bits shifted to the right out of the 16 bit register space are lost. A record or play operation started with only a 5 or 8 bit input command, and with the IAB bit not set (i.e. at "0") will cause the A0 through A9 data resident at the end of the operation to be treated as an address pointer. This will, of course, be data input to the SPI port in the previous command cycle mostly from the C0 through C4 bit positions. This is probably not valid address data.*

Example #3, Playback at an address - This second example assumes the ISD4000 is already powered up and that playback is to begin at a specific address. This example we will use address 92, or HEX address 5C, <00 0101 1100>. Each control bit in the input side of the command register of the device is set up as follows:

C0 - The MC bit must be LOW since a message cueing cycle is not desired.
C1 - The IAB bit must be LOW since the address is **not** being ignored.
C2 - The PU bit must remain HIGH since the device is already powered up and must remain powered up after this command cycle.
C3 - The P/R bit must be HIGH since a Playback cycle is to be started.
C4 - The RUN bit must be HIGH since an active cycle is to be started.
A0 - A9 - The address bits must be all be defined since the IAB bit is LOW.

1. The transfer of the two bytes of data is begun by changing the SS pin to a LOW. The bytes to be sent are:
HEX address E0 5C, < 1110 0000> <0101 1100>.

2. The first bit clocked into the SPI port will be the A0 address bit which is a "0," followed by A1 which is a "0," followed by A2 which is a "1" etc.

3. The last of the 16 bits clocked into the SPI port will be the RUN bit which, of course, is a "1."

4. The command cycle ends with the SS pin going back HIGH. At this time, the Playback cycle starts at the address as specified by A0 through A9.

---

example, the A0 would become 05 and the 5C would become 3A. When doing the bit placement inversion, don't forget that two of the address bits fall in the "control byte" of the data sent into the device.

5. One additional command must be given to the device so that it will proceed through the chip's memory as desired. The IAB bit in the command register must be set to enable the address sequencer internal to the ISD4000. Otherwise the device will do a looping playback at the address specified by A0 through A9. This second command cycle must be given before the chip reaches the end of the row where playback begins. This allows at least 150 milliseconds to send this command.

6. The command byte to be input must be one that does not interrupt the playback operation in progress but sets the IAB bit. Additional address information is not needed. If it is assumed that the SPI port of the microcontroller can input any number of bits in a control cycle, this operation only requires that 5 bits be input to the SPI port. The proper control bits to send are the same as originally sent in the first control operation with the exception that the IAB bit is now set. It is therefore necessary to only send <1111 0> to the SPI port.

7. Once again an SPI cycle is begun by taking SS LOW. The first bit shifted into the ISD4000 SPI port is C0, which is a "0." The fifth and last bit shifted into the SPI port is C4, the RUN bit, which is a "1." The control sequence ends with the SS pin going back HIGH and results in playback proceeding through the memory in the ISD4000 row by row until we tell the ISD4000 to stop playback or a set EOM bit is encountered.

Example #4, Stop record or playback and power down the device - This example assumes that a Record cycle or a Playback cycle is underway and that the operation is to be terminated. The device is also to be powered down by this control cycle. The same command may be used to perform this function in both the Record and the Playback cycles. In this example a full byte transfer is assumed.

1. A record or playback operation is terminated by clocking a command cycle into the SPI port with the RUN bit LOW. The PU bit must also be low in this example so that the device powers down. The rest of the bits in the command byte do not matter but by convention they are to be left LOW. The command byte needed, therefore is HEX address 00, <0000 0000>.

2. Once again, the transfer of the command byte of data is begun by changing the SS pin to a LOW followed by the byte consisting of all zeros.

3. Record or playback will end in approximately 50 milliseconds (again, the data sheet will have exact information) followed by device power down.

Example #5, Address jump during record or playback - One of the ISD4000's key features is to seamlessly jump from one address to any other address during the course of a Record or Playback cycle. This function enables the Message Management operations described in Application Note #2, "Message Management in the ISD4000[3]." The following discussion describes how this feature works:

The ISD4000 memory array may be thought of as an address map of x rows by y columns. The device is currently sold in two array sizes. Both versions have 1200 columns. The smaller device has 400 rows and the larger has 800 rows. Only the rows may be addressed. So when you begin an operation on a row at address "n", it begins in column "0" and proceeds through the row to column 1199 if not stopped by a command cycle along the way.

The internal address sequencer in the ISD4000 may be thought of as operating in one of two possible modes. As a record or playback operation proceeds through the device, at the end of each row, a decision must be made as to where to go next. If the IAB bit is set ("1") when the end of the row is reached, then the address sequencer increments by 1 count and record or playback proceeds to the next row in sequence. If the IAB bit is cleared ("0") when the end of the row is reached, then the address sequencer pulls the address from the SPI command register and record or playback proceeds at the row defined by this address. This transition from the end of one row to the beginning of some other row not in sequence is seamless. No samples are lost as record or playback proceed across the boundary without problems.

[3] Application Note #2 may be found in the 1996 Second Edition and subsequent ISD Data books.

Programmer Caution: An address jump cannot be made from the last row in memory. Record or playback to end of the last row in the memory[4] results in a device overflow (OVF) interrupt. The software that controls the ISD4000 must trap for this address and make sure that this row is not used in a message management algorithm where an address jump might be necessary.

Interrupt Service

The interrupt service structure of the ISD4000 may be read with only 8 SPI clock cycles. The following example demonstrates how this is possible:

Example #5, Read Interrupt status and clear the device interrupt condition - The ISD4000 device has only two interrupting conditions: (1) End Of Message (EOM) which signifies that a set EOM bit has been found during a Playback cycle and (2) Overflow (OVF) which indicates that the end of device memory has been encountered during either a Record or a Playback cycle. Note that an Overflow interrupt can only occur at the end of the last row in the device's memory.

Both of these conditions result from the end of device operation, i.e. playback or record has ceased. Both conditions cause the INT pin[5] of the device to be pulled LOW. This pin will remain LOW until the next SPI cycle in the ISD4000 device.

As the interrupt condition is read and clocked out of the MISO pin, data is being simultaneously being clocked into the MOSI pin. This input data will be interpreted as a control input and the device will react accordingly. It is necessary, therefore, to make sure this input data leaves the ISD4000 in a desired state. Since the interrupt always results from the end off an operation, shifting in a power-up command, as illustrated in example #1 above, is usually a safe response. In this example below, we are assuming we are shifting in HEX address 20, <0010 0000> while the interrupt data is being shifted out.

The following description shows how interrupt status may be read from the ISD4000:

1. The interrupt status read cycle begins with the SS pin of the device being changed to a LOW. At that time, the state of the OVF bit will be presented to the MISO output. This level is static. That is, it will remain low as long as the SS pin stays LOW and the SCLK pin is not clocked.

2. The first clock cycle of the input to the SCLK pin will now cause the state of the EOM bit to be presented to the MISO output.

3. Seven more clocks must be presented to the SCLK pin to insure the interrupt condition is properly cleared.

4. The interrupt status read is terminated by changing the SS pin back to a HIGH. When the SS pin goes HIGH, the INT pin output will be allowed to go back HIGH, pulled up by the external pull-up resistor.

The above examples demonstrate the ease of use of the ISD4000 family device. A Record or Playback cycle may be started and controlled with a minimum number of SPI clock cycles. Some short cuts exist that enable the device to be controlled with less than a full 8 bit transfer to the SPI control port. This translates into a voice record and playback system that requires very little overhead from the controller in the system.

[4] Consult the data sheet for the device being used to determine the number of rows in the memory array.

[5] INT is an active LOW open drain output. If this output is to be used, a pull-up resistor must be installed to pull the pin up to the proper HIGH level.

How to use ISD4000 with a Microcontroller

This application note describes how to use ISD4000 with a microcontroller. The purpose of this application note is to illustrate the ease of use of the ISD4000 family with any SPI compatible microcontroller.

Unlike other families of Single chip record and playback from ISD, the ISD4000 is a microcontroller slave peripheral device. It is controlled with a microcontroller either with an SPI compatible hardware already on the microcontroller or with I/O ports that emulate SPI protocol. For the ease of use and illustration a microcontroller with an SPI compatible serial interface is used in this application note.

Serial Peripheral Interface (SPI):

ISD4000 operates from an SPI serial interface. The SPI protocol is a synchronous serial data transfer protocol. The SPI interface has 4 control I/O pins.

1. Slave Select ($\overline{SS}$) : This pin when low will select the ISD4000.
2. Master Out Slave In (MOSI): This is the serial input to the ISD4000.
3. Master In Slave Out (MISO): This is the serial input to the ISD4000.
4. Serial Clock (SCLK) : This is the clock input to the ISD4000. The master microcontroller provides this clock.

The data transfer protocol assumes that the Microcontroller's SPI shift registers are clocked on falling edge of the SCLK. On the ISD4000, data is clocked in on the MOSI input on the positive clock edge and data is clocked out on the MISO output on the negative clock edge.

All serial data transfers begin with the falling edge of Slave Select pin. Slave Select is held low during all serial interface and it is held high between instructions. Each operation that ends in an EOM or Overflow from the ISD4000 will generate an Interrupt. The interrupt will be cleared the next time an SPI cycle is initiated. As the interrupt data is shifted out of the ISD4000 MISO pin, control and address data are simultaneously being shifted into the ISD4000 MOSI pin. Care should be taken such that the data shifted in is compatible with current system operation. An operation begins with the RUN bit set and ends with the RUN bit reset. All operations begin with the rising edge of the SS\.

SPI Control Register :

The SPI control register provides control of individual device functions such as Play, Record, Stop/Pause, Message Cueing(Fast Forward), Power-up, Power-Down, Start and Stop operations and Ignore Address pointers.

There are five control bits associated with the ISD4000 that control the device. These bits are:

- **C0 = MC** When this bit is set to 1 during playback it starts a message cueing cycle (Fast forward to the next EOM). C0 operation is not defined during Record.
- **C1 = IAB** When this bit is set to 1, any address data shifted into the SPI MOSI shift register is ignored. A resulting Record, Playback or Message Cueing operation will begin at the next address in the device memory at the end of the proceeding operation.

When IAB (Ignore Address Bit) is set to 0, a playback or record operation starts from address (A9-A0) and ends at the end of that row. If no other control data is input to the SPI port, the device will continue that operation over again at the same address and therefore "loop" on that row. To continue playback or record consecutively through the memory, a second SPI cycle should immediately be input with the IAB bit changed to 1 before the device reaches the end of a row. To stop an operation it is important to note that if IAB bit is set to 0 the data shifted into address bit locations will be transferred to the internal address register of the device and the address internal to the device before the stop instruction will be lost.

It is recommended to set IAB = 1 when a Stop/ Pause command is issued if address read back is desired.

- **C2 = PU** When this bit is set to 1 the device powers up and will be ready for an operation after TPUD(25ms approx.). The user needs to wait TPUD before issuing an operational command. It is important to wait TPUD after the power up command is issued before sending another command. The device will not function properly if a record or playback command with power-up command is issued in the same SPI cycle. For example to REC from address 00 the following program cycle should be used.

1. Send POWER-UP command (00100<X-X)
2. wait TPUD(per device specification, approximately 25 mSec)
3. Start Recording (10100<A9-A0>)
4. Continue consecutive recording(10110<X-X>)

- **C3 = P/ R** When this bit is set to 1 the device goes to playback mode and when it is 0 the device goes to Record mode.
- **C4 = RUN** When this bit is set to 1 the device starts an operation and when it is set to 0 it will stop an operation.

SPI OPCODE Format:

The ISD4000 accepts either an 8-bit command or 16- bit command from a microcontroller through the SPI port. The opcode format is 5 control bits and 11 address bits for a 16-bit command. The opcode format is 5 control bits and 3 address bits for an 8 bit command with the 3 address bits being "don't care" bits.

Examples of Record/Playback Operation:

- **Record at an address**

Power Up the device

Send 00100<xxxx> to the SPI

Wait Tpud

Start Recording at <10 bit address >

Send 10100<x><10bit address> (note; IAB = 0)

Send 10110<xxxx> (note; IAB = 1)

Recording will continue until memory is full or a new SPI cycle with RUN=0 is input or until the end of memory is reached (Overflow Interrupt).

Stop Record and Power down

Send 00000<xxxx> to the SPI

OR

Stop/Pause Record and don't Power down

Send 0 0110 <xxxx>

- **Record a Message at the next available address**

If the device is not powered Up

Send 00100<xxxx>

Wait Tpud

Record the next message

Send 10110<xxx>

Recording continues until ready to Stop (or Device runs out of memory space)

- **Playback at an address**

Power Up the device

Send 01100<xxx> to the device

Wait Tpud

Start Playback at <10 bit Address>

Send 11100<x><10 bit address>

Send 11110<xxx> to continue playback.

Playback continues until memory overflows or ready to end (reaches EOM)

Stop Playback and Power down

Send 01010<xxx>

OR

Stop Playback don't power down

Send 01110 <xxx>

- **Playback at the next Message**

If the device is not powered Up

Send 01100<xxxx>

Wait Tpud

Playback the next Message

Send 11110<xxx>

Playback continues until an EOM is reached (or chip runs out of memory Space)

- **Playback the "3rd" Message (Fast Forward to message number 3 and play it)**

Power up the device

Send 01100<xxx>

Wait Tpud

Start Message Cueing Cycle at "0" Address

Send 11101<x><00 0000 0000>

Send 11111<xxx>

Device runs at 800x normal play speed, audio muted.

Device Stops at next EOM, Gives EOM Interrupt, Increments address counter 1 count, now the address pointers points to '2nd" Message. Execute 2nd Message cueing cycle at "next" message.

Send 11111<xxx>

Device runs at 800x normal play speed, audio muted.

Device Stops at next EOM, Gives EOM Interrupt, Incumbents address counter 1 count, now the address pointers points to '3rd" Message. Play "next" message.

Send 11110<xxx>

The 3rd message will play at normal speed. When message ends EOM interrupt occurs

- **Read Interrupt Status Bits and Current Address**

Clock in Instruction

Send <xxxxx><xxx>

Device will execute the operation as specified by the instruction.

If there is no desire to change the status of the device, care should be taken that the command is compatible with current operation.

Send SPI 8 clocks to read the Status Bits (OVF and EOM)

Send SPI 16 clocks to read Status and current Address.