Interfacing the HI7190 to a Microcontroller

Application Note | September 1995 | AN9527

Authors: Stephen LaJeunesse and John Kornblum

Introduction

The Intersil HI7190 is a 24-bit monolithic instrumentation sigma delta A/D converter designed for applications such as Process Control and Measurement, Industrial Weigh Scales, Motion Control and Medical Equipment. The HI7190 serial I/O port is compatible with most synchronous transfer formats including the Motorola 6805/11 series SPI and Intel 8051 series SSR protocols. This application note discusses the HI7190 Serial Interface Port and details two application circuits useful in demonstrating how to interface the HI7190 to a microcontroller. For further information on this topic, see the HI7190 product datasheet and Technical Brief 331 “Using The HI7190 Serial Interface”.

HI7190 Serial Port Signals

The HI7190 Sigma-Delta A/D converter communicates to a controlling device over either a 2 or 3-wire serial interface. Data is transmitted or received via a synchronous clock. The Serial Clock (SCLK) line is the synchronous data clock used to strobe the serial stream in or out of the HI7190 A/D converter. The data clock can be generated by the converter or can be supplied to the converter. When the HI7190 is the clock master, that mode is referred to as the Self Clocking mode. When the HI7190 is a clock slave, that mode is referred to as the External Clocking mode. The serial port also contains a status flag (Data Ready, $\overline{DRDY}$) that signals a controller that the HI7190 has completed a conversion and the digital result is now available for reading from the device. The Data Ready flag is cleared by reading the HI7190’s Data Register.

The HI7190 is selected, enabling an I/O operation, whenever the Chip Select ($\overline{CS}$) line is asserted low.

The HI7190 has 2 data lines that can be used with 2-wire or 3-wire serial bus interfaces. The Serial Data I/O (SDIO) line is a bidirectional data line that can be used as a dedicated input or a bidirectional data path. The Serial Data Out (SDO) line is a dedicated output pin for use in 3-wire interfaces where there must be a separate path for data in and data out. In a 2-wire interface, such as that used with Intel microcontrollers, the SDIO line is used exclusively for bidirectional data transfers. Figure 1 shows the pinout of the HI7190.

HI7190 Serial Protocol

When communicating with the HI7190 a set protocol must be followed. During the first phase of a transfer an instruction byte must be written to the device. The instruction byte contains the internal register address that will be accessed in the rest of the communication cycle. A typical communication cycle would involve an Instruction Cycle and a Data Cycle as shown in Figure 2. The data accessed during the Data Cycle is determined by the instruction byte contents.

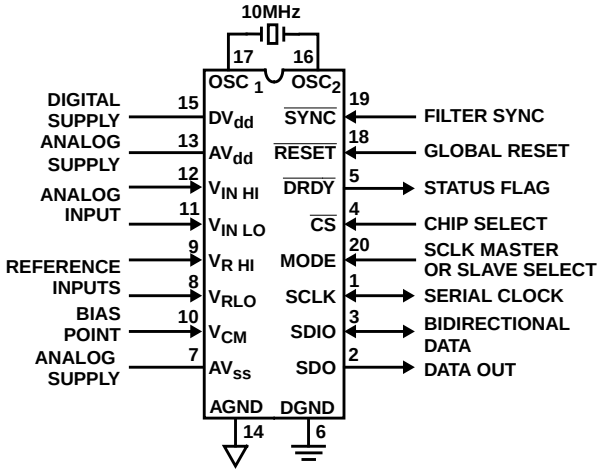

FIGURE 1. HI7190 SIGNAL DESCRIPTIONS

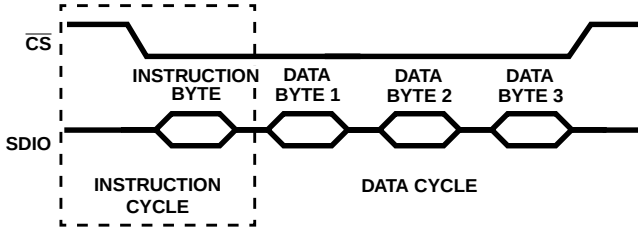

FIGURE 2. HI7190 COMMUNICATION CYCLE

The instruction byte allows access to the following registers internal to the HI7190.

- Control Register
- Data Register
- Zero Scale Calibration Register
- Positive Full Scale Calibration Register
- Negative Full Scale Calibration Register

The instruction byte is organized as follows.

MSB	6	5	4	3	2	1	LSB
$\overline{R}$/W	MB1	MB0	FSC	A3	A2	A1	A0

$\overline{R}$/W - Bit 7 of the Instruction Register determines whether a read or write operation will be done following the instruction byte load. 0 = READ, 1 = WRITE.

MB1, MB0 - Bits 6 & 5 of the Instruction Register determine the number of bytes that will be accessed following the instruction byte load. See Table 1 for the number of bytes to transfer in the Data Cycle.

TABLE 1. MULTIPLE BYTE ACCESS BITS

MB1	MB0	DESCRIPTION
0	0	Transfer 1 Byte
0	1	Transfer 2 Bytes
1	0	Transfer 3 Bytes
1	1	Transfer 4 Bytes

FSC - Bit 4 is used to determine whether a Positive Full Scale Calibration Register I/O transfer (FSC = 0) or a Negative Full Scale Calibration Register I/O transfer (FSC = 1) is being performed (see Table 2).

A3, A2, A1, A0 - Bits 3 and 2 (A3 and A2) of the Instruction Register determine which internal register will be accessed while bits 1 and 0 (A1 and A0) determine which byte of that register will be accessed first. See Table 2 for the address decode.

TABLE 2. INTERNAL DATA ACCESS DECODE STARTING BYTE

FSC	A3	A2	A1	A0	DESCRIPTION
X	0	0	0	0	Data Output Register Byte 0
X	0	0	0	1	Data Output Register Byte 1
X	0	0	1	0	Data Output Register Byte 2
X	0	1	0	0	Control Register Byte 0
X	0	1	0	1	Control Register Byte 1
X	0	1	1	0	Control Register Byte 2
X	1	0	0	0	Offset Cal Register Byte 0
X	1	0	0	1	Offset Cal Register Byte 1
X	1	0	1	0	Offset Cal Register Byte 2
0	1	1	0	0	Positive Full Scale Cal Register Byte 0
0	1	1	0	1	Positive Full Scale Cal Register Byte 1
0	1	1	1	0	Positive Full Scale Cal Register Byte 2
1	1	1	0	0	Negative Full Scale Cal Register Byte 0
1	1	1	0	1	Negative Full Scale Cal Register Byte 1
1	1	1	1	0	Negative Full Scale Cal Register Byte 2

Interfacing to the 8X51 SSR Protocol

The HI7190 can interface to microcontrollers that use a 2 or 3-wire serial hardware interface. A 2-wire interface involves a tightly coupled system where a single converter is connected to a single microcontroller. In this mode only the serial clock line (SCLK) and the bidirectional data line (SDIO) are used to communicate between the A/D and the microcontroller. Figure 3 shows a 2-wire interface to an 8X51 style microcontroller.

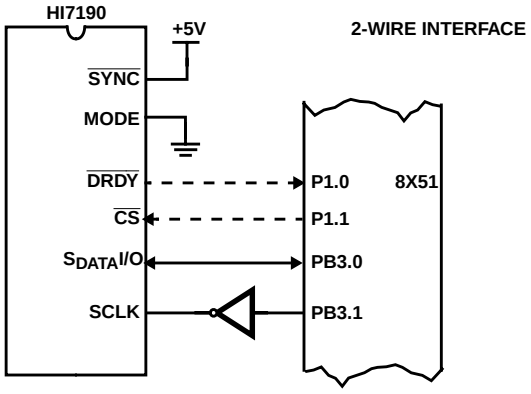

FIGURE 3. HI7190 INTERFACE TO 8X51

8051 Setup

Mode 0 of the 8X51 uses RXD (Port 3, Line 0) as the data port and TXD (Port 3, Line 1) as the shift clock. Data is shifted with LSB being the first bit in the sequence. The baud rate is fixed to 1/12 the microcontroller oscillator frequency. The 8X51 is the serial shift clock master therefor the HI7190 is placed in external clocking mode by grounding the MODE pin. The HI7190 can be set-up in polled mode where the status of the $\overline{DRDY}$ line is read into the 8X51. When $\overline{DRDY}$ is low, the HI7190 is ready to be accessed. In a multi-converter application the $\overline{CS}$ line can be used to address each individual A/D in the system. In a single converter application the $\overline{CS}$ may be grounded and an access is started by initiating the Instruction byte. The HI7190 should be reset to ensure proper power-up state. On power-up the HI7190 is configured for MSB first transfers and descending byte mode.

Since the 8X51 Intel microcontroller is a little endian designed machine, the HI7190 should be programmed for LSB first and ascending byte mode. Ascending byte mode will sequence through multiple bytes from least significant byte to most significant byte. The HI7190 expects data to be valid for the rising edge of the shift clock and shifts data out on falling edges. The 8X51 microcontroller is just the opposite. An inverter is needed on the serial clock line if the user wants to maintain approximately 1/2 clock cycle setup and hold times at the HI7190. Eliminating the inverter would give approximately a full clock cycle of setup time and zero hold time. The HI7190 will work in either design. Figure 4 shows the HI7190 port timing.

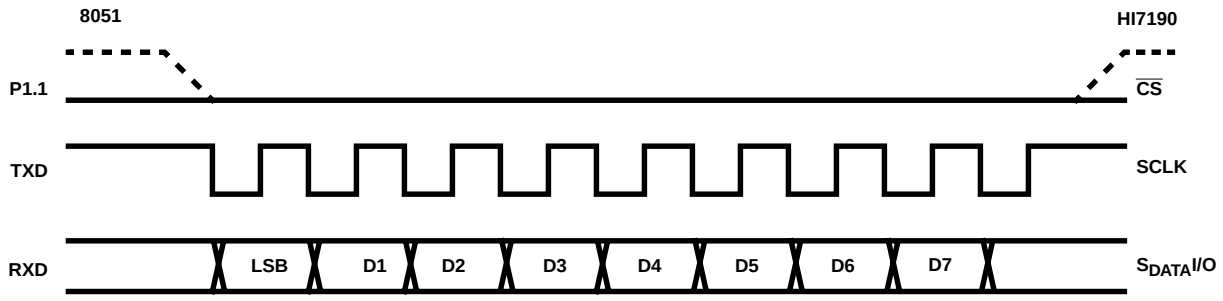

FIGURE 4A. DATA SEND/WRITE

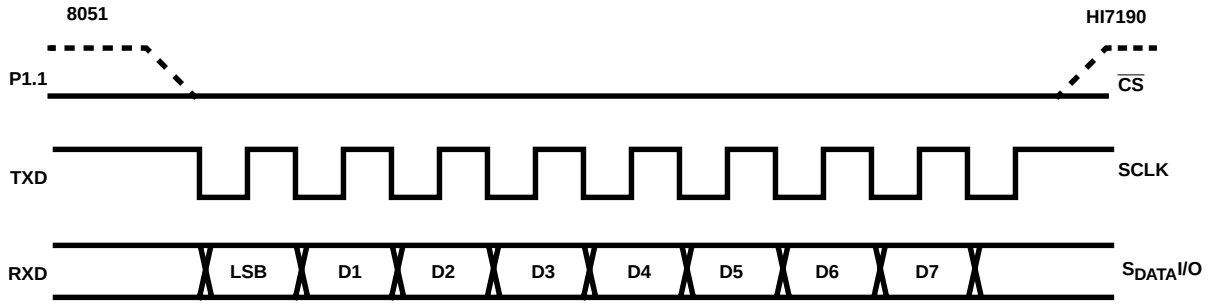

FIGURE 4B. DATA RECEIVE/READ

FIGURE 4. H17190 SERIAL PORT TIMING

Programming the HI7190 with the 8X51

The serial port of the 8X51 and the HI7190 need to be configured after power-up or a hardware reset. The HI7190 Control Register must be set to comply with the 8X51 data format and a conversion rate must be set. The following program initializes the 8X51 serial port and HI7190. Data is read in a polled fashion instead of interrupt driven.

8X51Microcode Example

```
;
;           Power-up/Reset, Port initialization
;           Set-Up Port 1 for reading status bits
;           Routine, Set Mode 0, Baud Rate=
;           Polled Data, (No interrupts)
;
SSRINIT:    MOV     SCON, #0000 0000B;
            SETB    91H; Set P1.1 for CS (Chip Select)
;
;           Reconfigure the HI7190 for
;           Ascending Byte direction, LSB First
;           Note: HI7190 expects MSB first format until
            after this write is complete.

ADINIT:     MOV     SBUF, # 0010 0001B;
            MOV     SBUF, # 0110 0000B;
;
;           Poll DRDY Signal for Data Ready
;           when ready assert Chip Select (CS),
;           write instruction byte and read 24 bits
;           of information
;
ADRUN:      MOV     R1, #003H;
            MOV     R0, START_ADDRESS;
            MOV     R2, #001H;
            MOV     R3, DATA_STREAM_SIZE;
POLL_DRDY   MOV     A, P1;
            ANL     A,R2;
            JZ      READ_DATA;
            SJMP    POLL_DRDY;
READ_DATA   CLR     P1.1;
            MOV     SBUF, #0100 0000B;
DATA_LOOP   MOV     A,SBUF;
            MOV     @R0, A;
            INC     R0;
            DEC     R1;
            JZ      DATA_LOOP;
            SETB    P1.1;
            DEC     R3;
            JZ      FINISHED;
            SJMP    POLL_DRDY;
            RET;
```

The initialization (SPINIT) configures the Serial Port in Mode 0 operation where the shift clock is generated by the 8X51 and the baud rate is set at $f_{OSC}/12$. The baud rate should not exceed the HI7190's specification of 5Mbps. Port 1 bit 1 is the control bit Chip Select that enables the HI7190's serial port. The ADINIT module configures the HI7190 operating mode. After a power-up the HI7190's control register is initialized for offset binary data coding, and a conversion rate of 30Hz. The gain is set to 1, the byte sequencing on port accesses is descending (2..1..0) the MSB is the first bit shifted in serial transfer, the serial data out line is disabled, the burn-out current source is disabled. Also, after power-up, a self calibration is completed before the HI7190 begins actively converting.

The ADINIT module will change the byte sequencing to ascending where the least significant byte is sent first (0.. 1.. 2) to match the Intel little endian data structure. The shift order is also changed from the MSB first to the LSB first in the serial transfer. Gain = 1 and SDO disabled are maintained.

The ADRUN module initializes the byte count for data transfers into the R1 register while the starting address for the incoming data storage is set as well as the data buffer size. R2 is set with the mask value for the $\overline{DRDY}$ flag which can be read or Port 1 bit 0.

The POLL_DRDY module checks the status of the $\overline{DRDY}$ flag from the HI7190 A/D converter, upon detecting $\overline{DRDY}$ being low the READ_DATA module will be called.

The READ_DATA module will assert the $\overline{CS}$ signal for the HI7190 serial port low and write the instruction byte to the A/D. Three bytes of data will be read from the A/D comprising the entire 24 bits from the conversion and the if the data buffer is full the routine will return to the main calling routine

Interfacing to the SPI Bus Protocol

The Serial Peripheral Interface (SPI Bus) is a serial bus using a 3-wire hardware interface. The three lines used to transfer data from one device to the other are the Serial Clock (SCK) line, the Master In Slave Out (MISO) data line and the Master Out Slave In (MOSI) data line. Data is shifted MSB first, and byte sequencing is in descending order (2.. 1 ..0). The clock is typically inactive low. Port D line 4 is the SCK. The shift clock is generated by the bus master which can be either a microcontroller or a peripheral. Data is routed either to PD2 (Master In Slave Out) or PD3 (Master Out Slave In) depending or software initialization. The Slave Select ($\overline{SS}$) line determines if the 68HC11 microcontroller is a Master or Slave on the SPI Bus.

The Serial Peripheral Data I/O register in the microcontroller initiates transmission/reception of a byte. The SPI port on the microcontroller is configured using the Serial Peripheral Control Register. Many devices contain SPI ports such as the 6805, and 6802 but this discussion will center on the 68HC11. When connecting an HI7190 Sigma-Delta A/D converter to the SPI Port of the 68HC11 the user has many configuration options available. The serial clock generation can be generated by the HI7190 using Self Clocking mode or by the 68HC11. In Figure 5 the HI7190 is configured as the clock master for the SPI port. This is accomplished by pulling the MODE line on the HI7190 high ('1') and grounding the Slave Select ($\overline{SS}$) pin of the microcontroller. Conversely, if the microcontroller was to be the clock master then the $\overline{SS}$ line would be tied high and the MODE pin grounded. The $\overline{DRDY}$ line of the A/D converter can be monitored via an interrupt scheme or by using simple polling. The programming example uses a polled status scheme.

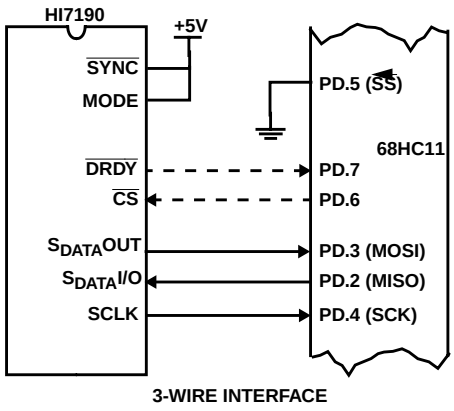

FIGURE 5. HI7190 INTERFACE TO 68HC1

Programming the HI7190 with the 68HC11

The serial ports of the HI7190 A/D converter and the 68HC11 must be configured after power-up or a hardware reset.

68HC11 Microcode Example

```
*This subroutine will configure the SPI port and the
*HI7190 A/D converter. It will read a stream of data
*and store it in memory
*
*Configuration of the SPI Control Register in no
*interrupt, system enable, normal CMOS outputs,
*slave mode, SCK idle hi, clock phase hi, clock divder=2
*
SPINIT  CLRA
        LDAA  #%x1xx xxxx  Bit 6 Port D drives CS
        STAA  PORTD        CS inactive
        LDAA  #$4C         Init SP
        STAA  SPCR         Load SPI Control Reg
        LDAA  SPDR         Read to clear port
*
*Initialize the HI7190 Control Register, Two's
*Complement, Conversion Rate = 30Hz Conversion
*Mode Operation, Bipolar, Gain = 1, Descending
*Byte direction, MSB First, Serial Data Out Enabled.
*
ADINIT  LDAA  #%x0xx xxxx  Bit 6 Port D drives CS
        STAA  PORTD        CS active
        LDAA  #$C6         Instruction Byte
        STAA  SPDR         Load SPI Data Reg
WAIT    LDAA  SPSR         Check Port Status
        BPL   WAIT         Wait for port to Empty
        LDAA  #$A8         Control Reg Byte 2
        STAA  SPDR         Load SPI Data Reg
WAIT1   LDAA  SPSR         Check Port Status
        BPL   WAIT1        Wait for port to Empty
        LDAA  #$B1         Control Reg Byte 1
        STAA  SPDR         Load SPI Data Reg
WAIT2   LDAA  SPSR         Check Port Status
        BPL   WAIT2        Wait for port to Empty
        LDAA  #$01         Control Reg Byte 0
        STAA  SPDR         Load SPI Data Reg
```

```
WAIT3   LDAA    SPSR        Check Port Status
        BPL     WAIT3       Wait for port to Empty
        LDAA    #%x1xx xxxx Bit 6 Port D drives CS
        STAA    PORTD       CS inactive

*
*This subroutine will collect data from the HI7190
*Sigma-Delta Converter, Poll DRDY Signal for Data
*Ready when ready assert Chip Select (CS), write
*instruction byte and read 24 bits of information
*
ADRUN   PSHX
        PSHY
        PSHA
        PSHB
        LDY     STRT_ADD    Data Buffer Pointer
        LDX     BUFF_SIZE   Data Buffer Size
DRDY    LDAA    PORTA       Poll Data Ready
        LDAB    #$03        Byte Counter

        ANDA    DRDYMASK    80H for Port D MSB
        BNE     DRDY        DRDY Cleared?
RD_DATA LDAA    #%x0xx xxxx Bit 6 Port D drives CS
        STAA    PORTD       CS active
        LDAA    #$42        Instruction Byte
        STAA    SPDR        Load into HI7190
WAIT4   LDAA    SPSR        Check Port Status
        BPL     WAIT4       Wait for port to Empty
D_LOOP  LDAA    SPSR        Check Port Status
        BPL     D_LOOP      Wait for new input data
        LDAA    SPDR        Read Data Byte
        STAA    STRT_ADD    Store in Memory
        INY                 Bump Address Pointer
        DECB                Decrement Byte Counter
        CMPB    #$00        Test Byte Counter
        BNE     RD_DATA     Read another byte
        CPY     X           Compare Buffer pointer
                            to buffer size
        BNE     DRDY        Poll for more data
        RTS                 Done Return from
                            subroutine
```

Conclusion

This application note has described two typical application circuits with microcode segment examples.

The first circuit is designed with the 8X51 microcontroller(s) in a configuration such that the 8X51 is the clock master in a two line interface and data transfers are LSB to MSB format.

The second circuit is designed with the 68HCxx microcontroller(s) in a configuration such that the Hi7190 is the clock master in a three line interface and data transfers are MSB to LSB format.

Sales Office Headquarters

NORTH AMERICA
Intersil Corporation
P. O. Box 883, Mail Stop 53-204
Melbourne, FL 32902
TEL: (321) 724-7000
FAX: (321) 724-7240

EUROPE
Intersil SA
Mercure Center
100, Rue de la Fusee
1130 Brussels, Belgium
TEL: (32) 2.724.2111
FAX: (32) 2.724.22.05

ASIA
Intersil (Taiwan) Ltd.
7F-6, No. 101 Fu Hsing North Road
Taipei, Taiwan
Republic of China
TEL: (886) 2 2716 9310
FAX: (886) 2 2715 3029