ANALOG DEVICES Engineer To Engineer Note EE-154

Technical Notes on using Analog Devices' DSP components and development tools
Phone: (800) ANALOG-D, FAX: (781) 461-3010, EMAIL: dsp.support@analog.com, FTP: ftp.analog.com, WEB: www.analog.com/dsp

ADSP-2191 Host Port Interface

LH/GO: Revised 2002-10-09

Introduction

This application note introduces the reader to the ADSP-2191 Host Port Interface (HPI). This port is an 8- or 16-Bit parallel, address & data multiplexed, asynchronous slave that gives a host processor read/write access to all of the ADSP-2191 internal and external memory and IO (except IO page 0) space. In addition to memory access, the ADSP-2191 can be 'booted' by the host processor.

An application will be described showing how to set up the bus for use with a host such as a second ADSP-2191 EMI port.

Overview of HPI

The ADSP-2191 has 24 pins dedicated to the HPI.

Pin Name	I/O	Function
HAD[15..0]	I/O	MUX'ed Address & Data
HA16	I	MSB Address
HACKP	I	Acknowledge Polarity
HALE	I	Address Latch Enable
HRD	I	Read Strobe
HWR	I	Write Strobe
HACK	I/O	Access Ready Acknowledge
HCMS	I	Memory Space Chip Select
HCIOS	I	IO Space Chip Select

The HPI 16-bit address bus, HAD[15..0], is multiplexed with the data bus. The data bus width defaults to 8-bits using HAD[7..0] and is configurable by the DSP after $\overline{\text{RESET}}$, or by the Host, to 16-bits.

The HPI offers two hardware address modes called Address Cycle Control (ACC) and Address Latch Enable (ALE). These two address modes are shown in the timing diagrams in Appendix B. This offers flexibility with a number of different host processors.

Figure 1 shows the HPI connections to a host processor.

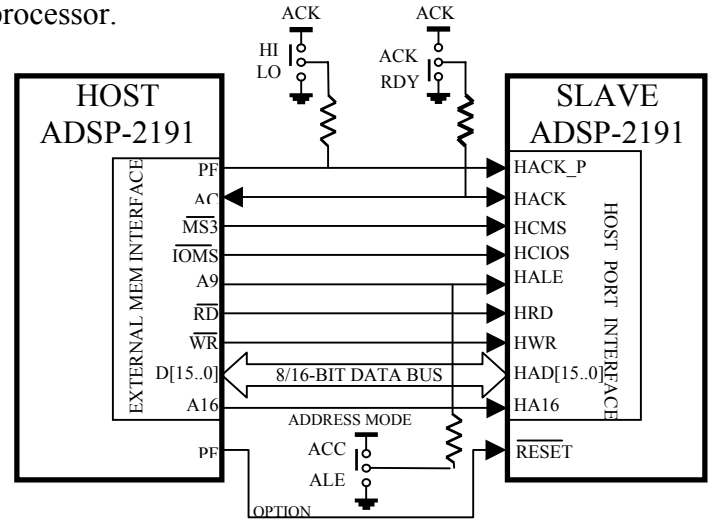


Figure 1: HPI Bus Connections

The bus communication protocol is asynchronous and has two handshake modes; ACKnowledge mode (ACK) and ReaDY mode (RDY). This is also intended to facilitate interconnection to alternative hosts.

The Host may write to, or read from, ADSP-2191 memory and IO locations either by directly specifying an address and then the data (Direct Mode) or by Direct Memory Access (DMA). In Direct Mode, a single HPI transaction will consist of an 'Address Phase' followed by a 'Data Phase'. Depending upon the size of the data word, the data phase will consist of one, two, three or four words being sent over the HPI bus.

In the case of DMA transfers, the internal HPI DMA configuration registers will specify the destination address and number of words so only the data will need to be passed over the HPI bus.

HPI Configuration

1. Read/Write Strobe Pin Polarity

Several of the interface pins may be programmed, at reset time, to be active high, or low. This gives the user more flexibility when interfacing to different host processors.

The active state of the $\overline{\text{HWR}}$ and $\overline{\text{HRD}}$ strobes is set to active low by holding the respective pin high, when the $\overline{\text{RESET}}$ pin goes high, and holding the state high for a minimum of 10 clock periods following the leading edge of Reset. This is the most common mode for most host processors.

2. Acknowledge Signal Mode & Polarity

Because the internal buses of the ADSP-2191 are shared between internal memories, the core and other peripherals, it is necessary for the HPI to access these components using an asynchronous protocol to signal, to the host, when the selected location is available for a transaction to proceed. There are two asynchronous handshake protocols provided, a direct acknowledge (ACK) and a continuous ready acknowledge (RDY).

The following table records the different modes.

Ext Pin State at RESET		Acknowledge Handshake	
HACK	HACK_P	Mode	Active State
0	0	Ready	Low
0	1	Ack	High
1	0	Ack	Low
1	1	Ready	High

Selecting the mode of the host port acknowledge (HACK) signal is done by externally programming the logic level on the HACK pin and the HACK_P pin when the $\overline{\text{RESET}}$ pin goes inactive.

Note that the active state of the acknowledge handshake signal (HACK) is set by the state of the HACK_P pin at $\overline{\text{RESET}}$. If the level on the HACK_P is high during $\overline{\text{RESET}}$, the HACK pin is active high and if the level is low, HACK is active low. It is important that the correct 'active' state for this pin is established otherwise the host processor could 'hang' waiting for an ACK signal that is never returned.

A typical method of programming these pins is shown in figure 1. Notice that if the host processor has control of the $\overline{\text{RESET}}$ pin, it can program the state of the HPI protocol under software control.

3. HPI Address Modes (ACC vs. ALE)

Selecting Address Cycle Control (ACC) or Address Latch Enable (ALE), address modes, is set by the logic level on the HALE pin, at $\overline{\text{RESET}}$. Please refer to the timing diagrams in Appendix B to study the differences in the two modes.

If the HALE pin is held low, by an off-chip pull-down resistor, or the host, during $\overline{\text{RESET}}$, the HALE pin will function in the ALE mode. In this mode, the HPI latches the address from the HAD[16..0] bus on the falling edge of the HALE signal.

If the HALE pin is held high during reset, it will function in the ACC. In this mode, a logic zero on the HALE pin will cause a trailing edge transition of the HWR write strobe to latch an address into the HPI.

Figure 1 shows a jumper selectable method of selecting the desired mode at $\overline{\text{RESET}}$.

4. Configuration registers:

The HPI contains 15 registers for configuration and controlling operation. Nine registers are reserved for DMA applications and can be ignored for Direct HPI read/write applications. Also, four semaphore registers contain a single programmable bit and are used for applications that require a communications semaphore, typically during boot operation.

For Direct mode there are two critical configuration registers, the Host Port Configuration Register (HPCR) and the Host Port Direct Page Register (HPPR). Please refer to Appendix A for a description of these registers. These two registers define the operating mode for the HPI and determine the memory page for data transfers.

Example Application: ADSP-2191 Host

The following example is an assembly code walk-thru of a simple application running on an ADSP-2191 master in order to communicate from the EMI port to the HPI port on a second slave ADSP-2191. This is similar to the system block diagram shown in figure 1.

The mode of operation is direct read/writes to the slave HPI, showing how the master processor sets up the slave HPI for 16-bit ACC/ACK operation. The object of the application is to establish communication with the slave HPI and change the default data width to 16-bits. The host then writes a block of data to the slave's data memory, computing a running check sum as it updates and transmits each byte. The host then reads the same block of data back, again recomputing the checksum. When the block has been read back, the check sums are compared to confirm data transmission integrity.

The following code section describes the masters reconfiguration of the slave's host port for 16-bit data transfers.

```
/******* Writes to HPI Slave - Change config to init 16-bit bus ********

The master address line A9 is used to drive the slave HALE pin. Use
master IO address 0x1FF to pull HALE low indicating a Slave Address
and IO address 0x3FF to pull HALE high, indicating Slave Data.

The slave HACK (acknowledge) signal drives the master ACK input.
The master /IOMS drives the slave HCIOS.  The master /MS3 drives the
slave HCMS - writes to the slave memory space should be made using the
master DMPG1, page 0xC0, to drive /MS3 active.

************************************************************
/
SlvHPIConfig:
  IOPG = 0x08;       // Setup Master IO Page to be EXT IO space
  AR = 0x3802;              // Slave HPCR address 0x07:001 = 0x3802
  IO(0x1FF) = AR;           // write Master external IO MSB address

  AR = 0x0F01;              // HPCR Data - ACK Mode, 16-bits
  IO(0x3FF) = AR;           // Send MS Data Byte
  SR = LSHIFT AR BY -8 (LO);
  IO(0x3FF) = SR0;          // Send LS Data Byte

  /*
  The slave HPI is now set up for 16-bit data transfers.
  */

/************************************************************
/
```

The following code section covers the writing of 16-bit data to the slave's internal data memory via the slave's host port

```
/***************  Write to All Slave Data Memory  ***************
**
Write to the slave 2191's data memory using 16-bit transfers.  Start at
internal memory dm location 0x8000 and increment data by 1 at each
address.
**
************************************************************
/
WrSlave:
        /*  Configure for 24-bit access                         */
Config_16:
   IOPG = 0x08;       // Setup Master IO Page to be EXT IO space
   AR = 0x3804;       // Slave HPPR address 0x07:002 = 0x3804
   IO(0x1FF) = AR;    // write Master external IO MSB address
   AR = 0x0000;       // HPPR Data = 16-bits
   IO(0x3FF) = AR;    // Send MS Data Byte
   // HPI is now set up for 16-bit data
   I2 = 0x8000;       // Get 16-bits of address
  // Initialize Loop for the same size of buffer
  CNTR = 0x7FFF;
  DMPG1 = 0xC1;       // Setup master Page to be EXT MS3 space
  Do Write_Slv_Dm Until CE;
        AX0 = I2;                       // Load DM address into AX0
        SR = LSHIFT AX0 BY 1 (LO);  //Shift by 1
        //****Address cycle****
        DM(0x01FF) = SR0;               // Write DM Address to HPI
        NOP;
```

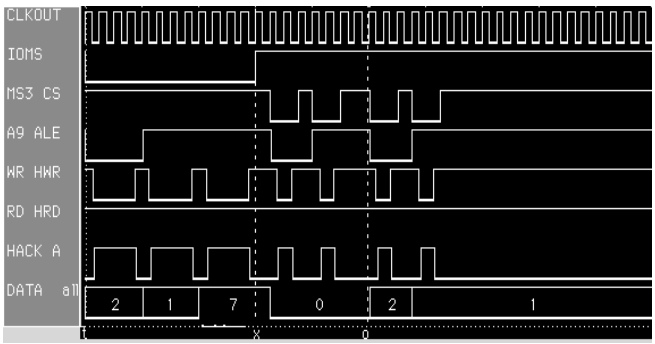
```
            //****Data cycle****
            AX0 = I2;                   // get data from Buffer
            DM(0x03FF) = AX0;           // Write DM Data
            NOP;
            modify(I2,m0);              // Modify I2
Write_Slv_Dm:    NOP;
```

The following code section covers reading of 16-bit data from the slave's internal data memory via the slave's host port

```
/*************** Read from Slave Data Memory ****************
**
Read back the slave data, written in the last function, and track its check
sum.

Compare checksums to ensure write/read integrity.
**
************************************************************
/
RdSlave:
   I0 = Buffer;
   M0 = 1;                          // modify of 1
   AX0 = I0;
   reg(B0) = AX0;                   // Init Base address of circular buffer
   L0 = length(Buffer);             // Init length of buffer
   CNTR = 0x7FFF;   // Initialize Loop for the same size of buffer
   Do Read_Slv_Dm Until CE;
            DMPG1 = 0xC1;       // Setup DM Page to be EXT MS3 space
            AX0 = I0;           // Load DM address into AX0
            SR = LSHIFT AX0 BY 1 (LO);   //Shift by 1
            //Address cycle
            DM(0x01FF) = SR0;    // Write DM Address to HPI
            NOP;
            //Data cycle
            X0 = DM(0x03FF);    // Read DM address as Data
            NOP;
            DMPG1 = 0x0;         // Setup master DM Page to be Block 1
            DM(I0,M0) = AX0;
Read_Slv_Dm:    NOP;
```

The following logic analyzer screen shot captures the initialization of the slave HPI (MS3 LO). The first write (A9 LO) is writing the slave's IO register and the next two writes (A9 HI) write the two 8-bit data bytes.



Following the re-configuration for 16-bit data transfers, are two sequential data write transactions. Note the address phase (A9 LO) and data phase (A9 HI) in each transaction.

Appendices

Appendix A: Configuration Registers

Host Port Configuration Register : (HPCR, IO:0x1C01)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	0	0	0	0	0	0	0	0	0	0	0	0

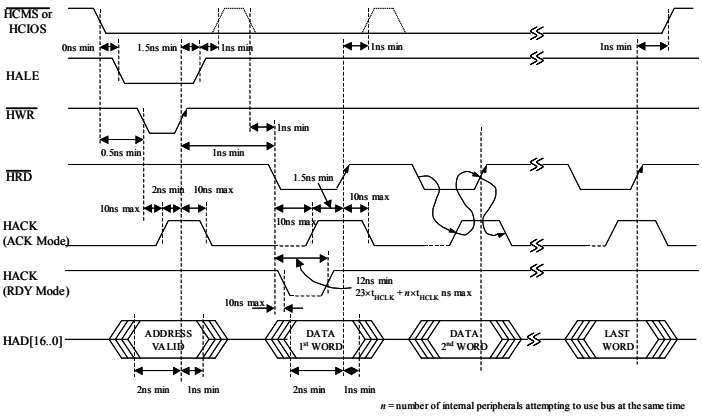
Default: 0x0000
Bit-0: HPI Data Width, 0=8-bits, 1=16-bits
Bit-1: Byte Endian; 0=Little, 1=Big
Bit-2: Data Order; 0=LSW First, 1=MSW First
Bit-3: Number of bytes for 24-bit Word, 8-bit bus
0=4-bytes
1=3-bytes
Bit-4: Pipelined Reads; 0=Normal, 1=Pipelined Reads
Bit-5: Prefetch reads on address phase
Bit-6: Acknowledge Mode; 0=ACK, 1=RDY
Bit-7: Reserved; Set to 0
Bit-8: Read Only, Read Strobe Active State
Bit-9: Read Only, Write Strobe Active State
Bit-10: Read Only, HACK Signal Active State
Bit-11: Read Only, HALE Signal Active State
Bits-12-15: Reserved, Set to 0

Host Port Direct Page Register : (HPPR, IO:0x1C02)

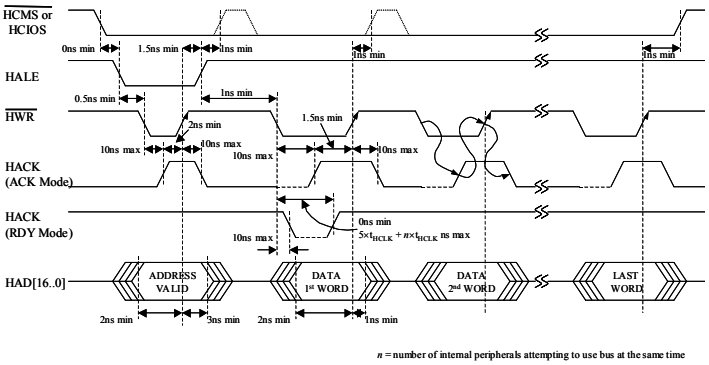
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	X	X	X	X	X	0	0

Default: 0x0000
Bit-0: HPI Memory Access Space; 0=Memory, 1=Boot
Bit-1: Memory/Boot Data Type; 0=16-Bits, 1=24-Bits
Bits-2..6: Reserved, Set to 0
Bits-7..15: MPAGE[8..0], Memory Page Address

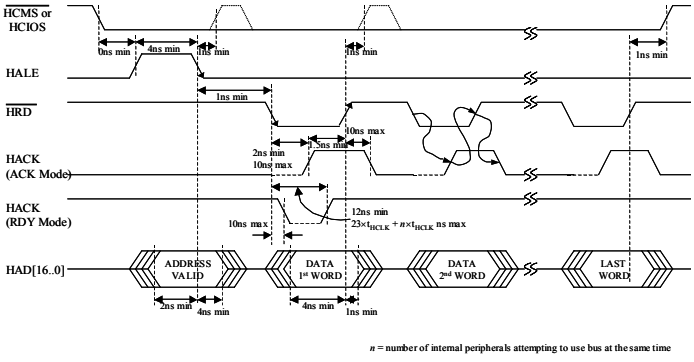
Appendix B: Timing Diagrams



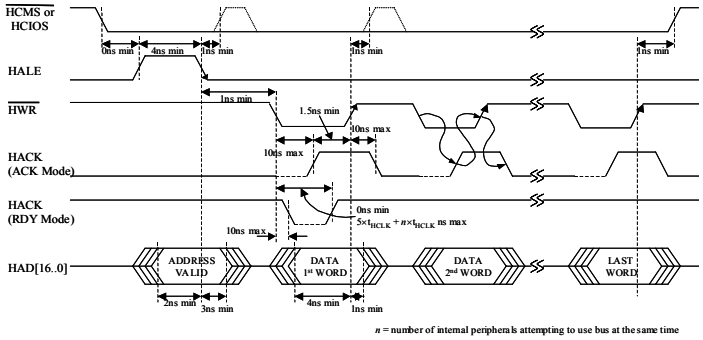
DIRECT READ - ADDRESS CYCLE CONTROL (ACC) MODE



DIRECT WRITE - ADDRESS CYCLE CONTROL (ACC) MODE



DIRECT READ - ADDRESS LATCH ENABLE (ALE) MODE



DIRECT WRITE - ADDRESS LATCH ENABLE (ALE) MODE

References

1. "*ADSP-219x/2191 DSP Hardware Reference*", Analog Devices., Inc, Part Number 82-000390-06
2. "*ADSP-2191M DSP Microcomputer Data Sheet*," Analog Devices., Inc.

Document History

Version	Description
February 2, 2002 by L. Hurst	- Created.
June 17, 2002	- Added Acknowledge Handshake on page 2.
	- Clarified mode selection paragraph on page 2.
October 9, 2002	- Clarified code snippets
	- Clarified example mode of operation as being ACC.