ScanWorks®
Platform for Embedded Instruments

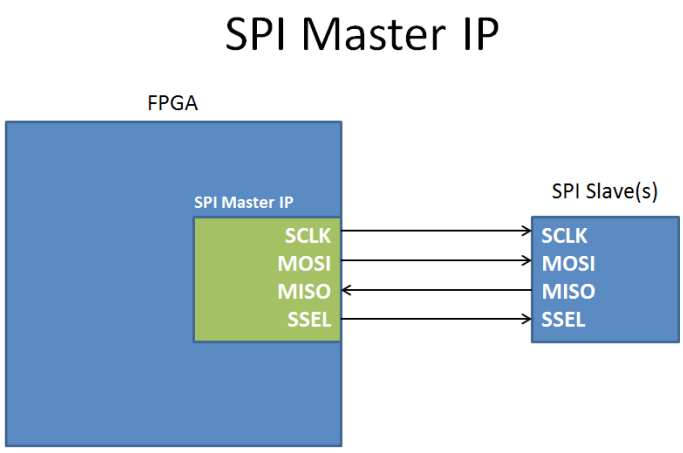
SPI Master IP -
FPGA-Controlled Test Instrument

Fact Sheet

Introduction

The SPI Master IP is an easy-to-use and powerful instrument providing at-speed functional test capabilities of a Serial Peripheral Interface (SPI) Bus in a board test FPGA environment. The SPI Master IP is used within the ScanWorks® FPGA-Controlled Test (FCT) Development software. When you select and configure the SPI Master IP instruments, ScanWorks automatically connects it up with other instruments of your choice and turns them into a cohesive, on-chip tester architecture.

SPI Master IP is included with the FCT Development software. For more information about FCT and other FPGA-Controlled Test Instrument IPs from ASSET visit our website at: https://www.asset-intertech.com/products/fpga-controlled-test.



Functional Description

The SPI Master IP uses MISO (master input, slave output) and MOSI (master output, slave input) serial registers to transfer data synchronized with the serial clock (SCLK) to and from addressable slave devices on the SPI bus, see Figure 1. The SPI Master IP operates in full duplex mode and communicates in master/slave mode where the master initiates the data frame. Multiple slave devices are allowed with individual slave select (SSEL) lines. It is easily adjusted to your application by setting programmable parameters, including the number of slave devices, serial clock frequency, and data width. The IJTAG IEEE P1687 supported Instrument Connectivity Language (ICL) and Procedural Description Language (PDL) describe the access to the IP, and provide a versatile debug and fault detection environment. High-level user-accessible functions in the IP are described as procedures in the automatically generated PDL, accessible from the IJTAG Instruments Action in ScanWorks.

Embedded Tester Generator Configurable Parameters

The Embedded Tester Generator (ETG) software will automatically wrap an IJTAG IEEE 1687 network around the IP and guide you in the parameter setting, pin mapping, and synthesis processes. The following parameters can be set when synthesizing the IP into an FPGA.

Key Benefits:

- Versatile IP can be used for a wide range of structural test and functional applications for serial hardware devices
- Verify and test serial hardware devices without the need for the system's functional software
- Reducing the prototype, debug, and validation phases of critical serial communication interfaces shortens product delivery schedules
- Can be combined with other structural and functional test as a part of the overall test sequence

Key Features:

- Four Wire SPI Bus Support
- Multiple Slave Communication
- Adjustable Data Width
- Adjustable SCLK

Configurable Parameter	
Name	**Description**
DATA_WIDTH	Controls the width of serial data. SPI device dependent. Minimum value of 8 (for internal IP registers). Set value to maximum register width for all SPI slaves on the SPI bus.
NUMBER_OF_SLAVES	Controls the width of SSEL output ports
SSEL_ACTIVE_POLARITY	Controls polarity (assert high or low) of SSEL output port
SPI_POL	Controls the polarity mode of SPI transfer
SPI_PHASE	Controls the phase mode of SPI transfer

The following configurable IOs are required to be pin-mapped when synthesizing the IP into an FPGA.

Configurable IO			
Name	**IO Type**	**Parameter Control**	**Description**
sysclk	Input	n/a	IP system clock input.
miso	Input	n/a	SPI data input. Master Input, Slave Output.
mosi	Output	n/a	SPI data output. Master Output, Slave Input.
sclk	Output	n/a	SPI clock output.
ssel	Output	n/a	SPI slave select output.

PDL Accessible IP Registers

The following IP registers can be accessed from the PDL (R=Read Only, W=Write Only, RW=Read & Write).

Register Map				
Register Name	**Register Type**	**Register Address**	**Format**	**Description**
IP_ID	R	0x00	Hex - Defaults to 0x00 (width=DATA_WIDTH)	IP ID Register for debug
TEST DATA REGISTER	RW	0x01	Hex - Defaults to 0x00 (width=DATA_WIDTH)	Test register for debug
CLOCK DIVIDE REGISTER	RW	0x02	Hex - Defaults to 0x00 (width=DATA_WIDTH)	Controls the SCLK frequency
SPI SLAVE ADDRESS	W	0x03	Hex - Defaults to 0x00 (width=DATA_WIDTH)	Controls which of the SSEL output asserts during transfer
MOSI DATA REGISTER	W	0x04	Hex - Defaults to 0x00 (width=DATA_WIDTH)	Holds the data that is going to be transferred to the SPI slave
MISO DATA REGISTER	R	0x05	Hex - Defaults to 0x00 (width=DATA_WIDTH)	Holds the data from the SPI slave device
TRANSFER LENGTH	RW	0x06	Hex - Defaults to 0x00 (width=DATA_WIDTH)	Holds the length of serial data that needs to be transferred. Less than or equal to DATA_WIDTH.
STATUS REGISTER	R	0x07	Hex - Defaults to 0x00 (width=DATA_WIDTH)	Status flag that indicates if transfer is done

PRIMITIVE PDL COMMAND PROCEDURES

These are the primitive commands available in the SPI Master PDL file, to be used when communication with the IP for writing application specific PDL procedures.

SPI Master IP Instrument

ASSET

PDL Command Procedures	
Procedure	**Description**
_SEND_STOP_COMMAND { }	Put SPI_Master into an idle state.
_SEND_RESET_COMMAND { }	Resets all of the SPI_Master internal registers to their default states.
_SET_CLKDIV { { clkdiv 0x00 } }	Sets the SPI_Master CLKDIV register which controls the SCLK frequency according to: sclk = sysclk / (2*(clkdiv + 1))
_SET_SSEL { { ssel_index 0x00 } }	Sets the SPISLAVEADDR register in the SPI_Master. Used in multi-slave STAR architectures.
_SET_MOSIDATA { { data 0x00 } }	Sets the MOSIDATA register in the SPI_Master.
_SET_TRANSFERLENGTH { { transferlength 0x00 } }	Sets the SHIFTCOUNT register in the SPI_Master. When set to a non-zero number, SPI data transfers are executed.
_READ_MISODATA { }	Reads the MISODATA register in the SPI_Master.
_CHECK_TRANSFER_STATUS { }	Reads the STATUS register in the SPI_Master. STATUS[0] = 1 means SPI transfer completed.

SPI_MASTER PDL UTILITY PROCEDURES

These are the utility commands available in the SPI Master PDL file, to be used when writing application specific PDL procedures.

PDL Utility Procedures	
Procedure	**Description**
_bitslice { msblsb bitrange }	Extracts subrange of binary bit values of a binary string. Example: set my_2bits [_bitslice $some_binary 3:2]
_hex2bin { hex }	Converts hex number (with or without '0x' or '0X') to a binary string without the 0b prefix.
_bin2hex { bin }	Converts binary number (with or without '0b' or '0B') to a hex string without the 0x prefix.

TEST PROCEDURES

Note: This PDL procedure is application specific in communicating with Analog Devices ADT7301 for temperature measurement. New user-defined PDL procedures will be required depending on the SPI device targeted.

PDL Utility Procedures		
Name	**Arguments***	**Description**
MeasureADT7301	Yes	Communicates with Analog Devices ADT7301 for temperature measurement.
* See Detialed Argument Description Table		

Detaled Argument Description			
Name	**Arguments**	**Format**	**Description**
MeasureADT7301	clkdiv	Two digit hex. Defaults to 0x01	The clkdiv register value adjusts the SCLK frequency according to this formula: sclk = sysclk / (2*(clkdiv + 1))
	spislaveaddr	Two digit hex. Defaults to 0x00	SPI Slave Address
	mosidata	Two digit hex. Defaults to 0x0000	Data to be transmitted to SPI Slave
	transferlength	Hex - Defaults to 0x0F (width=DATA_WIDTH)	Length of data to be transmitted

SPI Master IP Instrument

EXAMPLE TEST PROCEDURE

```
##############################################################################
# Example PDL Procedure definition for the SPI_Master that communicates to
# an Analog Devices ADT7301 for temperature measurement.
#
# Target Device = 13 bit SPI ADC
##############################################################################
proc MeasureADT7301 { { clkdiv 0x01 } { ssel_index 0x00 } { mosidata 0x0000 } { transferlength 0x0F } } {
          set done_bit 0;
          set watchdog 15;
          set count 0;
          set read_status 0x00;

          _RESET_SPI_Master_IP;                 # Resets the IP
          _SET_CLKDIV $clkdiv;                  # Sets the SCLK frequency based on clkdiv register value
                                                # sclk = sysclk / (2*(clkdiv + 1))
          _SET_SSEL $ssel_index;                # SPI SSEL number to assert.  Value can be 0-N where N = NUMBER_OF_SLAVES-1.
          _SET_MOSIDATA $mosidata;              # Data to be transmitted to SPI Slave
          _SET_TRANSFERLENGTH $transferlength;  # Length of data to be transmitted.  This starts the SPI comms.

          while { $done_bit != 1 && $count < $watchdog } {
                    set count       [expr {$count + 1}];
                    set read_status [_CHECK_TRANSFER_STATUS];
                    set done_bit    [_bitslice $read_status 0:0]; # 1 == transfer complete; 0 == transfer incomplete.

if {$done_bit == 1} {
                              puts "    SPI Transfer to SSEL\[$ssel_index\] Finished (done_bit == $done_bit).";
                    } else {
                              puts "    SPI Transfer to SSEL\[$ssel_index\] incomplete. (done_bit == $done_bit)";
                    }
          }

          if {$done_bit == 1} {
                    set temperature [_READ_MISODATA];               # Retrieve SPI slave data.

                    _SEND_STOP_COMMAND;           # Puts the IP in it's IDLE state. Change of IP command
                                                  # resets the done status.

                    set Temperature [expr $temperature/32 ];        # Operate on SPI Slave Data.
                    puts "Temperature Measured is: $Temperature Celsius";
          } else {
                    puts "FAIL: Watchdog limit of $watchdog reached.  The SPI_Master 'DONE' flag never asserts to a 1.";
                    iFail;
          }
}
```

ASSET Contacts:

Please contact your ScanWorks sales representative for more information.

ASSET InterTech, Inc.
2201 N. Central Expy., Ste 105
Richardson, TX 75080
+1 888 694-6250 or +1 972 437-2800
http://www.asset-intertech.com

SPI Master IP Instrument