AC459
Application Note
LX7730 Interface Reference Design Using RTG4

Microsemi.

Power Matters.™

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and must not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided “as is, where is” and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

Contents

1 Revision History 1

1.1 Revision 1.0 1

2 LX7730 Interface Reference Design Using RTG4 2

2.1 Introduction 2

2.2 Project Files and GUI Installation 2

2.3 References 3

2.4 LX7730 Design Implementation 3

2.4.1 UART Interface Module 3

2.4.2 LX7730 SPI Parallel Interface Module 6

2.4.3 Resource Utilization and Design/Performance Parameters 12

2.5 Conclusion 13

Figures

Figure 1 System Block Diagram of the LX7730 Interface Demo 2
Figure 2 Hierarchical Design 3
Figure 3 UART_IF Module 4
Figure 4 Content of the UART_IF Module 4
Figure 5 FSM for UART Data Transfer 5
Figure 6 LX7730 SPI Parallel Interface 7
Figure 7 Timing Diagram for Parallel Write 9
Figure 8 Timing Diagram for Parallel Read Data 10
Figure 9 Pin Multiplexer Block 11

Tables

Table 1 Byte Order for various UART Commands 5
Table 2 Input and Output Ports of uart_comp Module 5
Table 3 SPI Frame 7
Table 4 Input and Output Ports of spi_if_i Module 7
Table 5 Input and Output Ports of spi_lx7730_if Module 8
Table 6 Clock Division Inputs 8
Table 7 Input and Output Ports of lx7730_parallel_if Module 10
Table 8 Input and Output Ports of lx7730_pin_mux Module 11
Table 9 Input and Output Ports of lx7730_pin_mux Module 12
Table 10 Resource Utilization of the LX7730 Interface on the RTG4 Device 13

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 1.0

Revision 1.0 was the first publication of this document.

2 LX7730 Interface Reference Design Using RTG4

This application note provides information about the top-level field programmable gate array (FPGA) hardware design of the LX7730 interface with serial and parallel communication using the RTG4 device, which will allow customers to evaluate the key features of the LX7730, and the interface implemented in RTG4.

2.1 Introduction

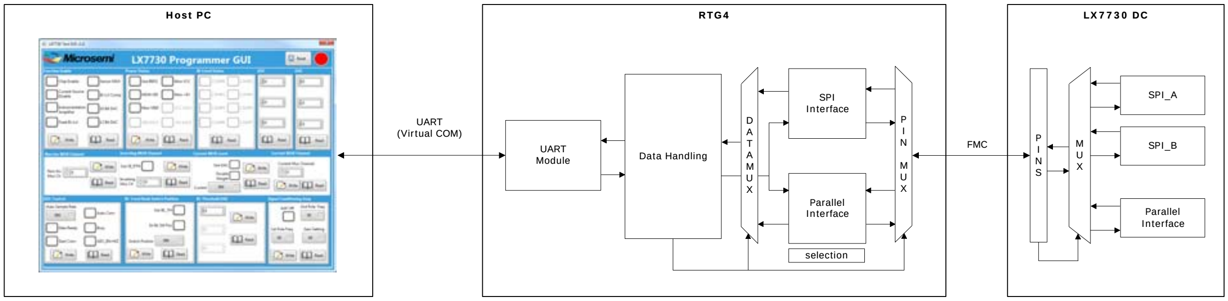
The LX7730 is a 64-analog channel telemetry device that is targeted for space applications. It provides the following features:

- 64-channel multiplexer
- Break-before-make switching
- 25kSPS 12-bit ADC
- 2% precision adjustable current source
- 1% precision 5.00V Source
- Threshold monitoring
- 8 bi-level logic
- 10-bit ADC
- Parallel or Dual SPI interface
- Radiation Tolerant: 100krad TID, 50krad ELDRS

The LX7730 interface design provides an interface to the LX7730 registers through the SPI and parallel interface. The LX7730 can be programmed through one of the two SPI ports (SPI-A, SPI-B) or the parallel interface. The LX7730 design can communicate with a graphical user interface (GUI) on the host machine over UART (virtual COM port on the kit). Figure 1 shows a system level block diagram of the LX7730 interface. The user interface is used to perform the following functions:

- Set the protocol or channel used to communicate with LX7730
- Write data into registers (using SPI or parallel)
- Read data from registers (using SPI or parallel)
- Read bi-level outputs from LX7730

Figure 1 • **System Block Diagram of the LX7730 Interface Demo**



2.2 Project Files and GUI Installation

The project files are available for downloading from the following path in the Microsemi website:

http://soc.microsemi.com/download/rsc/?f=rtg4_ac459_libero_project

The GUI installers are available for download from the following location in the Microsemi website:

http://soc.microsemi.com/download/rsc/?f=rtg4_ac459_gui

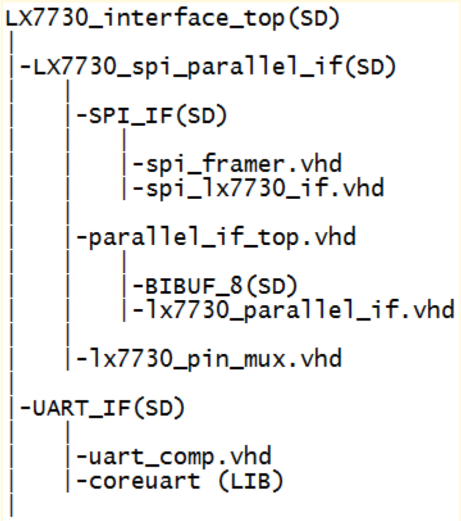
2.3 References

For more information about LX7730 device and features, refer to the *LX7730 datasheet*.

2.4 LX7730 Design Implementation

The design has been implemented in VHDL hardware description language. The design has been arranged hierarchically as shown in Figure 2.

Figure 2 • **Hierarchical Design**



```
LX7730_interface_top(SD)
|
|-LX7730_spi_parallel_if(SD)
|     |
|     |-SPI_IF(SD)
|     |     |
|     |     |-spi_framer.vhd
|     |     |-spi_lx7730_if.vhd
|     |
|     |-parallel_if_top.vhd
|     |     |
|     |     |-BIBUF_8(SD)
|     |     |-lx7730_parallel_if.vhd
|     |
|     |-lx7730_pin_mux.vhd
|
|-UART_IF(SD)
|     |
|     |-uart_comp.vhd
|     |-coreuart (LIB)
|
```

Some of the levels have been implemented as a SmartDesign (SD), a feature of the Libero IDE, which allows the user to instantiate and connect various modules visually. In turn, these SmartDesign modules are used to generate the VHDL files for synthesis.

2.4.1 UART Interface Module

The UART interface module uses the COREUART module (from the Libero IP catalog) and has custom logic (uart_comp.vhd) to interface this to the SPI and Parallel interface block.

Figure 3 shows the UART_IF module, as it appears in the LX7730_interface_top SmartDesign.

Figure 3 • **UART_IF Module**

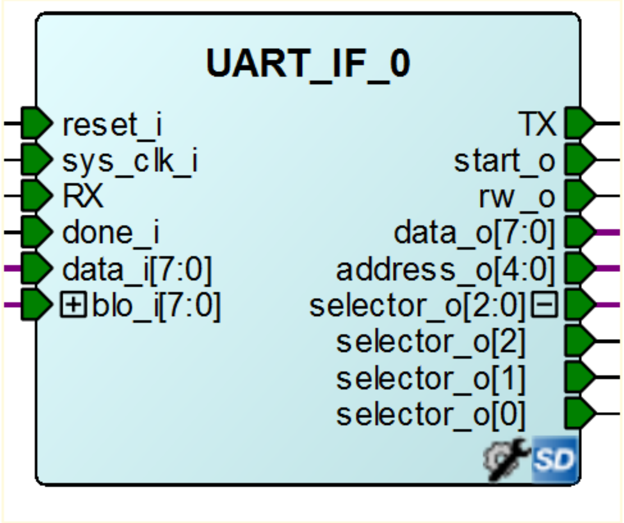
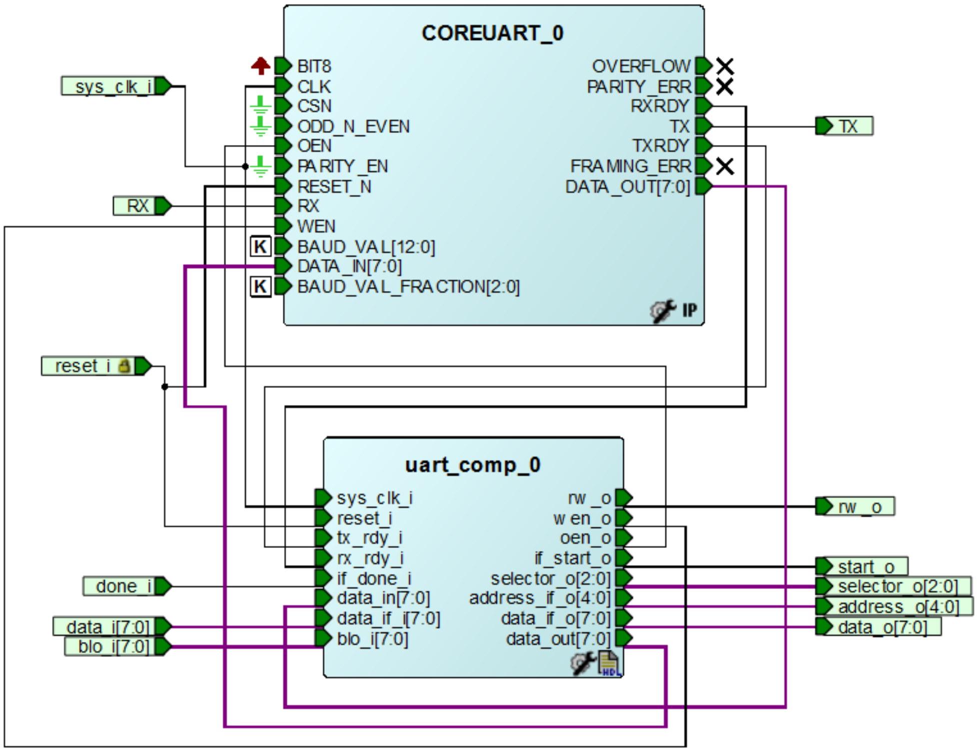


Figure 4 • **Content of the UART_IF Module**



The uart_comp module implements an FSM to send and receive data from the user interface. All the interactions start with a handshake byte (0x55) followed by a command identification byte. The command identification byte selects an action such as acknowledging a request to connect to the user interface, read register data, write register data, getting bi-level output data from LX7730, selecting SPIA, SPIB, or parallel protocol. The rest of the bytes are decided based on action that needs to be taken, which in turn is based on the command identification byte. Figure 5 shows the FSM implemented, and Table 1 shows the byte order for each command.

A special case of the READ register operation is the ADC read operation. When configured in the single conversion mode, this command waits for the DATA READY bit to be asserted before reading the ADC result register. In the Autoconvert mode, the FSM checks for two successive data ready bit assertions before reading the ADC result register, to ensure that the ADC data is valid.

Figure 5 • **FSM for UART Data Transfer**

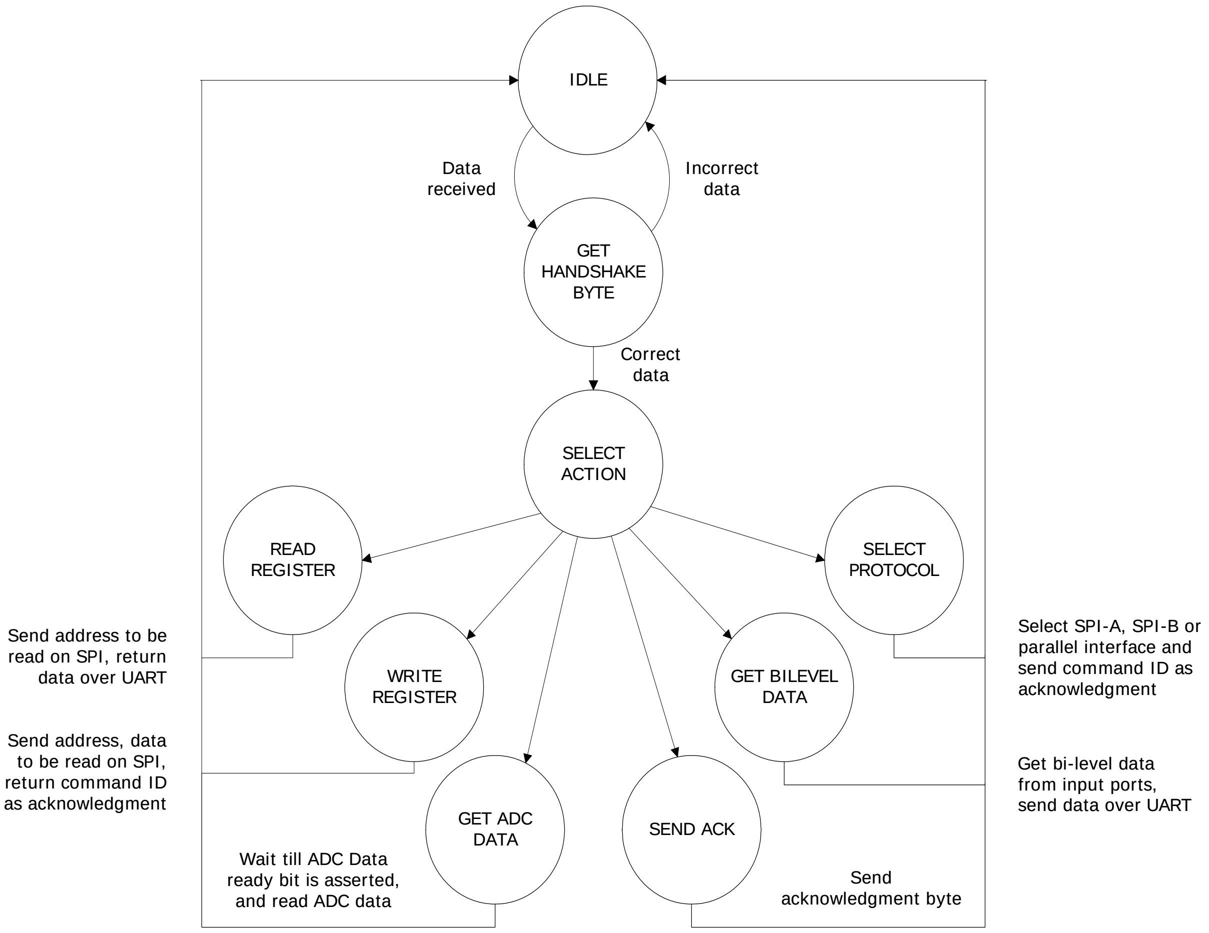

IDLE
Data received
Incorrect data
GET HANDSHAKE BYTE
Correct data
SELECT ACTION
READ REGISTER
SELECT PROTOCOL
WRITE REGISTER
GET BILEVEL DATA
GET ADC DATA
SEND ACK
Send address to be read on SPI, return data over UART
Send address, data to be read on SPI, return command ID as acknowledgment
Wait till ADC Data ready bit is asserted, and read ADC data
Send acknowledgment byte
Select SPI-A, SPI-B or parallel interface and send command ID as acknowledgment
Get bi-level data from input ports, send data over UART

Table 1 • **Byte Order for various UART Commands**

Command					
CONNECT	0x55	0x09			
Response from Hardware			0x09	0x0A	
SET	0x55	0x02	<ADDR>	<DATA>	
Response from Hardware					0x02
GET	0x55	0x01	<ADDR>		
Response from Hardware				0x01	<DATA>
GET ADC DATA	0x55	0x05			
Response from Hardware			0x05	<LSB>	<MSB>

Table 2 shows the port list of the uart_comp module.

Table 2 • **Input and Output Ports of uart_comp Module**

Signal Name	Direction	Description
sys_clk_i	Input	System clock

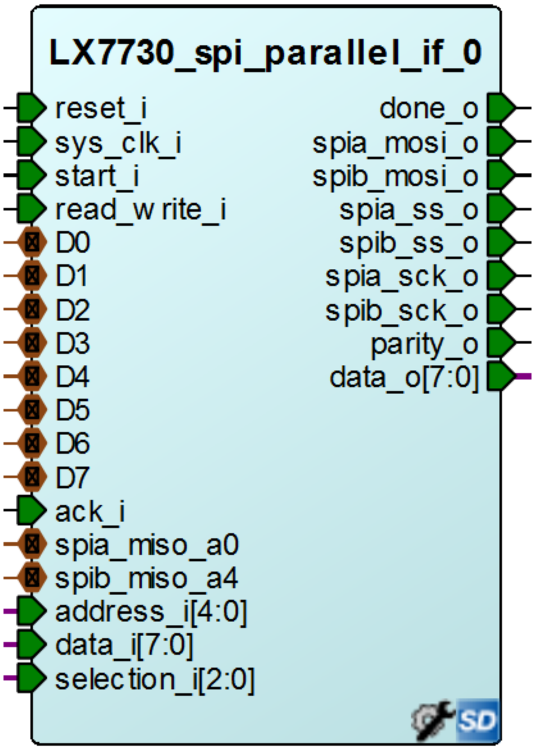
Table 2 • **Input and Output Ports of uart_comp Module**

Signal Name	Direction	Description
reset_i	Input	Reset
tx_rdy_i	Input	Ready to transmit signal from COREUART
rx_rdy_i	Input	Data ready signal from COREUART
if_done_i	Input	Done signal from SPI/parallel interface
data_in	Input	Data input from COREUART
data_if_i	Input	Data input from SPI/parallel interface
blo_i	Input	Bi-level output data from LX7730
rw_o	Output	Read/Write output signal to SPI/parallel interfaceblock
wen_o	Output	Write enable signal to COREUART
oen_o	Output	Output enable signal to COREUART
if_start_o	Output	Start signal to SPI/parallel interface
selector_o	Output	Protocol selector signal to select between SPI-A, SPI-B, and Parallel interface
address_if_o	Output	Address to SPI/parallel interface
data_if_o	Output	Data to SPI/Parallel interface
data_out	Output	Data output to COREUART

2.4.2 LX7730 SPI Parallel Interface Module

The lx7730_spi_parallel_if block contains the SPI interface module, the parallel interface module and a pin multiplexer module. The SPI and parallel communication is multiplexed on the same pins in the LX7730 device. The lx7730_pin_mux module multiplexes the SPI interface and the parallel interface between the SPI_A, SPI_B, and parallel configurations. Figure 6 on page 7 shows the SPI Parallel interface module, as it appears in the SmartDesign.

Figure 6 • **LX7730 SPI Parallel Interface**



2.4.2.1 SPI Interface Block

The SPI_IF module provides a custom 15-bit mode-0 SPI interface. A spi_framer block is used to generate parity and frame the read-write, address, data and parity bits for transmission to LX7730. The frame is input to the spi_lx7730_if module, and it outputs the frame as a SPI transaction. The read operation is similar to the operation mentioned above, but does not require data. The read operation triggers two SPI transactions because the data from the LX7730 is available to RTG4 after the second transaction. Table 3 shows the SPI frame. The input and output ports of the spi_if and spi_lx7730_if module are listed in Table 4 and Table 5 respectively.

Table 3 • **SPI Frame**

Read / Write (1-bit)	Address (5-bits)	Data (8-bits)	Parity (1-bit)

Table 4 • **Input and Output Ports of spi_if_i Module**

Signal Name	Direction	Description
sys_clk_i	Input	System clock
reset_i	Input	Reset
start_spi_i	Input	Start SPI transaction
read_write_i	Input	Read/Write bit input
address_i	Input	Address input
data_i	Input	SPI Data input
start_spi_o	Output	Start SPI
spi_frame_o	Output	Frame output for SPI

Table 5 • **Input and Output Ports of spi_lx7730_if Module**

Signal Name	Direction	Description
reset_i	Input	Reset
sys_clk_i	Input	System clock
start_i	Input	Start of ADC interface operation
sdi_i	Input	Start SPI transaction
control_reg_i	Input	Clock divider input
spi_frame_i	Input	Data frame to be transmitted
sdo_o	Output	Single bit data output to serial line
cs_o	Output	Chip/Slave select signal for serial interface
sck_o	Output	Clock signal for serial interface
data_o	Output	Data output
address_o	Output	Address output
full_frame_o	Output	Frame data from LX7730
results_rdy_o	Output	Signal indicates end of transaction

The spi_lx7730_if block provides clock division for generating SPI clock from the system clock. This is controlled using the control_reg_i input, which is 3-bits wide. Table 6 shows valid inputs to the control_reg_i port.

Table 6 • **Clock Division Inputs**

Signal Name	Clock Division
010	/8
011	/16
100	/32
101	/64
110	/128
111	/256
00X	Undefined

2.4.2.2 Parallel Interface

The lx7730_parallel_if block generates signals for parallel communication with the LX7730 device. The block provides control signals to send or receive parallel data. The parallel interface uses 8 dedicated data lines for bi-directional data transfer, five address lines and three control lines (/OE, /WE, /CE). Bi-directional buffers are used with each data line as the data lines allow bi-directional data transfer. The bidirectional buffers need to be enabled while writing, and disabled while reading data from the bus. Figure 7 shows the timing diagram for writing data and Figure 8 shows the timing diagram for reading data.

Figure 7 • **Timing Diagram for Parallel Write**

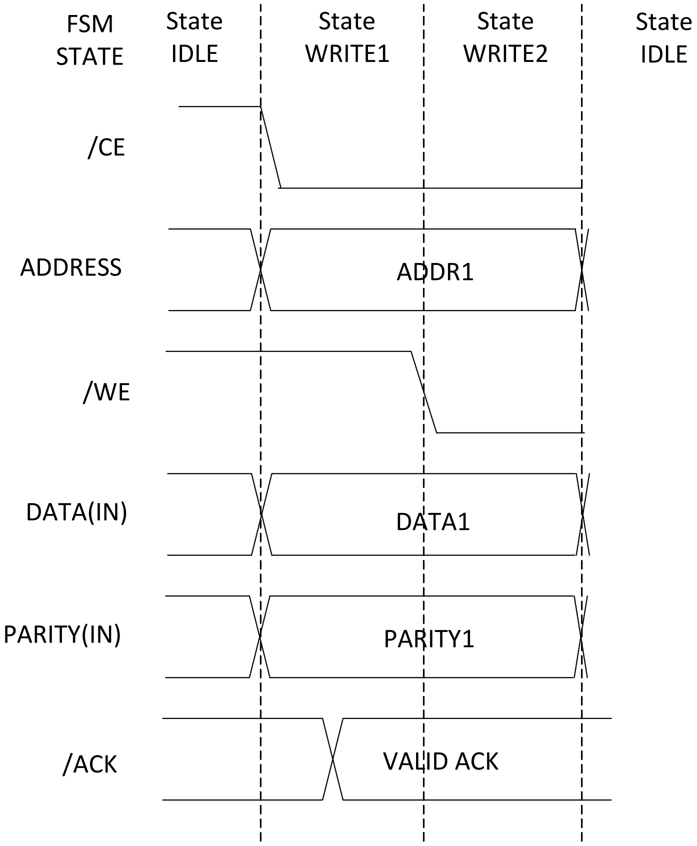


Figure 8 • **Timing Diagram for Parallel Read Data**

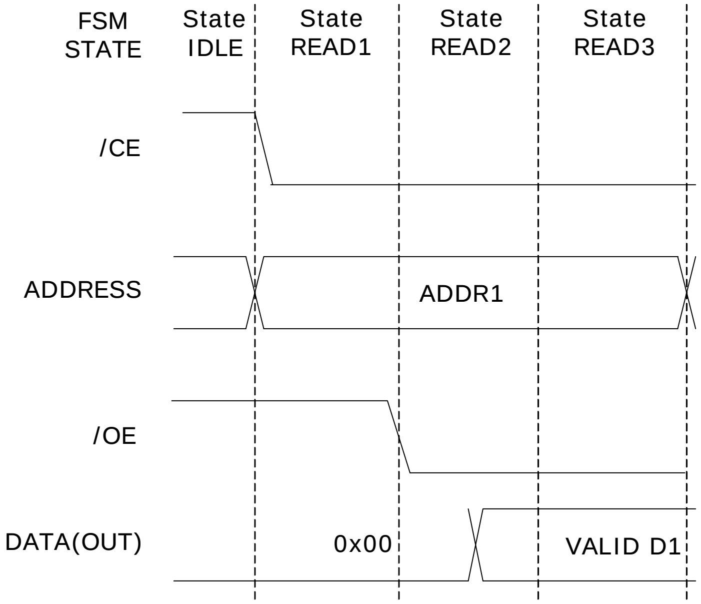


Table 7 shows the port list of the parallel interface.

Table 7 • **Input and Output Ports of lx7730_parallel_if Module**

Signal Name	Direction	Description
reset_i	Input	Reset
sys_clk_i	Input	System clock
rw_i	Input	Start of ADC interface operation
ack_i	Input	Start SPI transaction
start_i	Input	Clock divider input
d_i	Input	Data frame to be transmitted
data_input_i	Input	Single bit data output to serial line
addr_input_i	Input	Chip/Slave select signal for serial interface
bibuf_en_o	Output	Clock signal for serial interface
oe_n_o	Output	Data output
ce_n_o	Output	Address output
we_n_o	Output	Write enable (asserted when low) to LX7730
parity_o	Output	Parity output to LX7730
parallel_done_o	Output	Done signal indicating parallel read/write operation complete
data_out_o	Output	Data output to GUI
a_o	Output	Address output to LX7730
d_o	Output	Data output to LX7730

2.4.2.3 Pin Multiplexer

The lx7730_pin_mux block is a multiplexer used to select between SPI_A, SPI_B, and parallel interface because the LX7730 pins allow the use of one of the three channels. The SPI_A and SPI_B pins are used as the control pins and address pins in parallel transfer. The selection is based on SPI_A and SPI_B selection pins (which are currently connected to physical switches on the daughter card). The SPI input (SDI) signals act as outputs in parallel configuration. This is solved by using bi-directional buffers for these pins. Bi-directional buffers are also used with data lines because the same set of pins are used to send data to LX7730, as well as receive data from LX7730. Figure 9 shows the pin multiplexer block as it appears in the SmartDesign. Table 8 shows the input and output ports of the pin multiplexer block.

Figure 9 • Pin Multiplexer Block

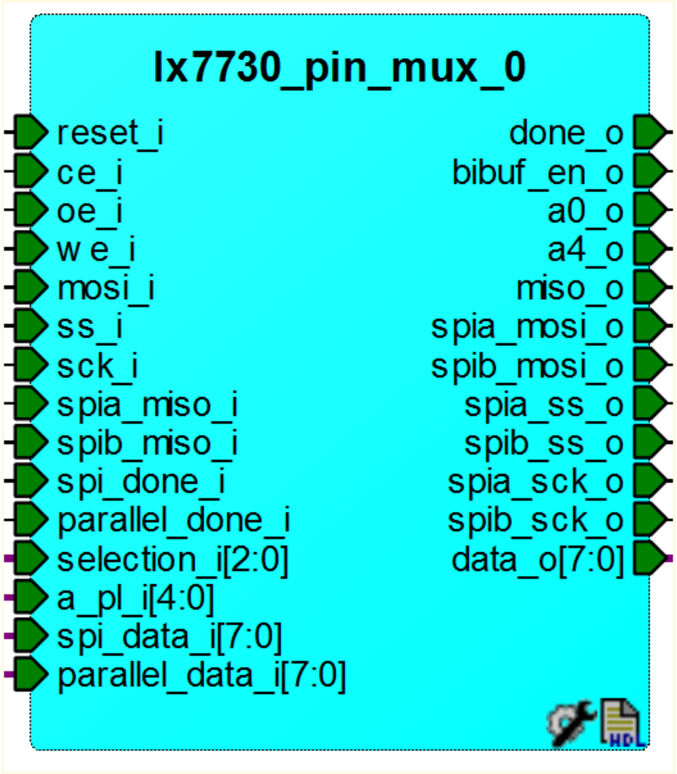


Table 8 • Input and Output Ports of lx7730_pin_mux Module

Signal Name	Direction	Description
reset_i	Input	Reset
ce_i	Input	Chip enable input from parallel interface module
oe_i	Input	Output enable input from parallel interface module
we_i	Input	Write enable input from parallel interface module
mosi_i	Input	MOSI (SPI) input from SPI interface module
ss_i	Input	Chip select input from SPI interface module
sck_i	Input	System clock input from SPI interface module
spia_miso_i	Input	MISO input from SPI A pin on LX7730
spib_miso_i	Input	MISO input from SPI B pin on LX7730
spi_done_i	Input	SPI done from SPI A pin on LX7730
parallel_done_i	Input	Parallel r/w operation done from parallel interface module
selection_i	Input	Multiplexer selection input from GUI
a_pl_i	Input	Address input from parallel interface module
spi_data_i	Input	SPI data input from SPI interface module

Table 8 • **Input and Output Ports of lx7730_pin_mux Module**

Signal Name	Direction	Description
parallel_data_i	Input	Parallel data input from parallel interface module
done_o	Output	Done output signal to UART module from SPI/Parallel module
bibuf_en_o	Output	Bidirectional buffer enable signal for parallel address lines/SPI MISO
a0_o	Output	a0 address line output to pin
a4_o	Output	a4 address line output to pin
miso_o	Output	MISO line to SPI module
spia_mosi_o	Output	SPI A MOSI line to LX7730
spib_mosi_o	Output	SPI B MOSI line to LX7730
spia_ss_o	Output	SPI A slave select line to LX7730
spib_ss_o	Output	SPI B slave select line to LX7730
spia_sck_o	Output	SPI A SCK line to LX7730
spib_ss_o	Output	SPI_B SCK line to LX7730
data_o	Output	Data output lines to LX7730

2.4.3 Resource Utilization and Design/Performance Parameters

The LX7730 interface has been designed for a system clock frequency of 40MHz. The minimum clock divider for SPI is divide-by-8, and hence the SPI is run at a clock of 5MHz. The parallel write operation uses two FSM states, and hence requires two clock cycles, while the parallel read operation uses three FSM states, and hence requires three clock cycles. The design has been tested at upto 120MHz system clock. For more information about the maximum permissible limits of the SPI and parallel interface, refer to the *LX7730 datasheet*.

Table 9 • **Input and Output Ports of lx7730_pin_mux Module**

Parameter	Data
System Clock	40MHz
SPI Clock	5MHz
Parallel data write	50µs
Parallel data read	75µs

Table 10 shows the resource utilization for the LX7730 design. The design utilizes less than 1% of the RTG4 device resources, which leaves the rest of the device available for other features desired by the user.

2.4.3.1 Resource Usage

Table 10 • **Resource Utilization of the LX7730 Interface on the RTG4 Device**

Type	Used	Total	Percentage
4LUT	353	151824	0.23
DFF	297	151824	0.20
I/O Register	0	2154	0.00
User I/O	37	718	5.15
Single-ended I/O	37	718	5.15
Differential I/O Pairs	0	359	0.00
RAM64x18	0	210	0.00
RAM1K18	0	209	0.00
MACC	0	462	0.00
H-Chip Globals	3	48	6.25
CCC	1	8	12.50
RCOSC_50MHZ	1	1	100.00
SERDESIF Blocks	0	6	0.00
FDDR	0	2	0.00
GRESET	1	1	100.00

2.5 Conclusion

The LX7730 interface design has been implemented on the RTG4 to evaluate the key features of the LX7730 device. The design can communicate with the LX7730 using an SPI or parallel interface. A user interface is used to communicate with the RTG4 over a UART interface to read/write LX7730 registers, select the communication channel and read bi-level outputs. The design consumes about 1% of the RTG4 device resources, leaving the rest of the device available for other user logic.