Introduction

The Xilinx 32-Bit Device Control Register Bus (DCR), a soft IP core designed for Xilinx FPGAs, provides the DCR bus structure as described in the *IBM 32-Bit Device Control Register Bus (DCR) Architecture Specification* to allow easy connection of the DCR Master to the DCR slaves. It provides the daisy-chain for the DCR data bus and the OR gate for the DCR acknowledge signals from the DCR slaves.

Features

- DCR connections for one DCR master and a variable number of DCR slaves, which are configurable via design parameter
- Daisy-chain connections for the DCR data bus
- Required OR function of the DCR slaves' acknowledge signal

LogiCORE™ IP Facts	
Core Specifics	
Supported Device Family [1]	Virtex®-5/-5FX, Virtex-4/-4Q/-4QV, Automotive Spartan®-3/-3E/-3A, -3A DSP
Resources Used	See Table 4 & Table 5.
Special Features	None
Provided with Core	
Documentation	Product Specification
Design File Formats	VHDL
Constraints File	N/A
Verification	N/A
Instantiation Template	N/A
Additional Items	None
Design Tool Requirements	
Xilinx Implementation Tools	ISE® 12.1
Verification	Mentor Graphics ModelSim v6.5c and above
Simulation	Mentor Graphics ModelSim v6.5c and above
Synthesis	XST
Support	
Provided by Xilinx, Inc.	

1. For a complete list of supported devices, see the 12.1 release notes for this core.

Functional Description

The relationships between the DCR Master/Slaves and the DCR Bus Module is shown in Figure 1.

32-Bit Device Control Register (DCR) Bus Interconnect

The Xilinx DCR bus module provides the necessary bus OR structure for the slaves' acknowledge signals and the correct daisy-chain connections for the DCR data bus. It allows for direct connection for 1 master and a variable number of slaves.

The Xilinx DCR Bus module does not provide a means for reducing the fan-out on DCR signals if the number of slaves is large, consequently you must analyze the timing of the DCR bus to determine if the number of slaves is too large. Figure 1 shows an example of the DCR connections when:

C_DCR_NUM_SLAVES=3, C_DCR_AWIDTH=10, and C_DCR_DWIDTH=32
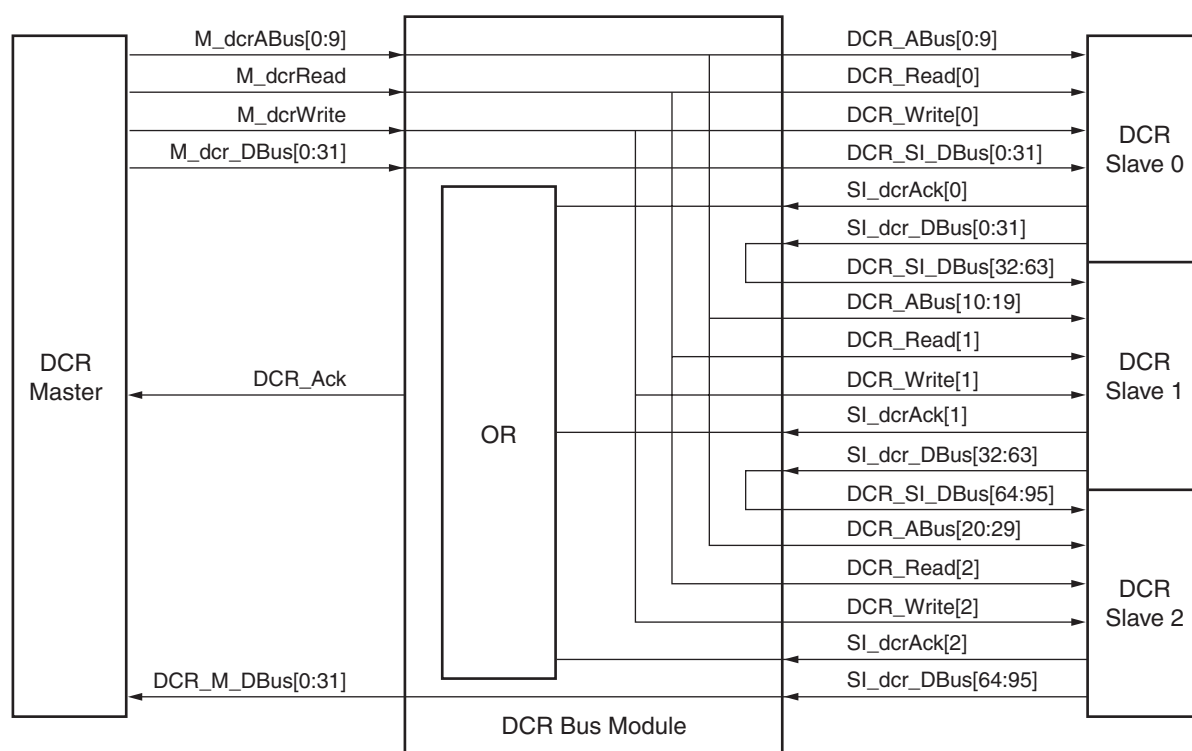
Figure 1: **DCR Interconnect Diagram**

DCR Design Parameters

To allow the user to obtain a DCR that is uniquely tailored the user's system, certain features can be parameterized in the Xilinx DCR design. This allows the user to have a design that utilizes only the

resources required by your system and runs at the best possible performance. The features that can be parameterized in the Xilinx DCR Bus are shown in Table 1.

Table 1: **DCR Bus Design Parameters**

Generic	Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
DCR Features					
G1	Number of DCR Slaves [1]	C_DCR_NUM_SLAVES	1 - 16	4	integer
G2	DCR Address Bus Width	C_DCR_AWIDTH	10	10	integer
G3	DCR Data Bus Width	C_DCR_DWIDTH	32	32	integer
Implementation					
G4	Implement the OR gate for the ACK signals using LUTs or MUXCYs [2]	C_USE_LUT_OR	1 = Implement the ACK OR gate using LUTs 0=Implement the ACK OR gate using MUXCYs	1	integer

Notes:

1. The DCR Bridge module does not provide any fan-out control on the DCR address bus or the DCR control signals. Therefore, care should be taken when using a large number of DCR slaves.
2. This parameter allows the choice of implementing the OR gate for the DCR acknowledge signal using LUTs or carry-chain muxes.

Allowable Parameter Combinations

All combinations of parameters are allowed. Care should be taken if the number of slaves is large due to the large fan-out of the DCR control signals.

DCR I/O Signals

Table 2 provides a summary of all Xilinx DCR Bus input/output (I/O) signals, the interfaces under which they are grouped, and a brief description of the signal.

Table 2: **DCR Pin Descriptions**

Port	Signal Name	Interface	I/O	Initial State	Description
Master DCR Signals					
P1	M_dcrABus[0:C_DCR_AWIDTH-1]	Master	I		Master DCR address bus
P2	M_dcrRead	Master	I		Master read from DCR indicator
P3	M_dcrWrite	Master	I		Master write to DCR indicator
P4	M_dcrDBus[0:C_DCR_DWIDTH-1]	Master	I		Master write data bus
P5	DCR_Ack	Master	O	0	Master DCR data transfer acknowledge
P6	DCR_M_DBus[0:C_DCR_DWIDTH-1]	Master	O	0	Master DCR read data bus
Slave DCR Signals					

Table 2: **DCR Pin Descriptions** *(Cont'd)*

Port	Signal Name	Interface	I/O	Initial State	Description
P7	DCR_ABus[0:C_DCR_NUM_SLAVES*C_DCR_AWIDTH-1]	Slave	O	0	DCR address bus to slaves
P8	DCR_Read[0:C_DCR_NUM_SLAVES-1]	Slave	O	0	DCR read request to slaves
P9	DCR_Write[0:C_DCR_NUM_SLAVES-1]	Slave	O	0	DCR write request to slaves
P10	DCR_Sl_DBus[0:C_DCR_NUM_SLAVES*C_DCR_DWIDTH-1]	Slave	O	0	DCR write data bus to slaves
P11	Sl_dcrAck[0:C_DCR_NUM_SLAVES-1]	Slave	I		DCR slaves' acknowledge
P12	Sl_dcrDBus[0:C_DCR_NUM_SLAVES*C_DCR_DWIDTH-1]	Slave	I		DCR slaves read data bus from slaves

Parameter/Port Dependencies

The width of many of the DCR signals depends on the number of DCR slaves. The dependencies between the DCR design parameters and I/O signals are shown in Table 3.

Table 3: **Parameter-Port Dependencies**

Generic	Name	Affects	Depends	Relationship Description
Design Parameters				
G1	C_DCR_NUM_SLAVES	P7, P8, P9, P10, P11, P12		The width of many buses is set by the number of DCR slaves in the design.
G2	C_DCR_AWIDTH	P1, P7		The width of these buses is set by the width of the DCR address bus.
G3	C_DCR_DWIDTH	P4, P6, P10, P12		The width of these buses is set by the width of the DCR data bus.
G4	C_USE_LUT_OR			
I/O Signals				
P1	M_dcrABus[0:C_DCR_AWIDTH-1]		G2	Width varies with the size of the DCR address bus.
P2	M_dcrRead			
P3	M_dcrWrite			
P4	M_dcrDBus[0:C_DCR_DWIDTH-1]		G3	Width varies with the size of the DCR data bus.
P5	DCR_Ack			
P6	DCR_M_DBus[0:C_DCR_DWIDTH-1]		G3	Width varies with the size of the DCR data bus.
P7	DCR_ABus[0:C_DCR_NUM_SLAVES*C_DCR_AWIDTH-1]		G1, G2	Width varies with the number of slaves and the size of the DCR address bus.

Table 3: **Parameter-Port Dependencies *(Cont'd)***

Generic	Name	Affects	Depends	Relationship Description
P8	DCR_Read[0:C_DCR_NUM_SLAVES-1]		G1	Width varies with the number of slaves.
P9	DCR_Write[0:C_DCR_NUM_SLAVES-1]		G1	Width varies with the number of slaves.
P10	DCR_SI_DBus[0:C_DCR_NUM_SLAVES*C_DCR_DWIDTH-1]		G1, G3	Width varies with the number of slaves and the size of the DCR data bus.
P11	SI_dcrAck[0:C_DCR_NUM_ SLAVES-1]		G1	Width varies with the number of slaves.
P12	SI_dcrDBus[0:C_DCR_NUM_SLAVES*C_DCR_DWIDTH-1]		G1, G3	Width varies with the number of slaves and the size of the DCR data bus.

DCR Bus Module Interfaces

DCR Master Interface

The interface of the DCR Master to the DCR Bus Module is shown in Figure 2.

DS402_02_020210
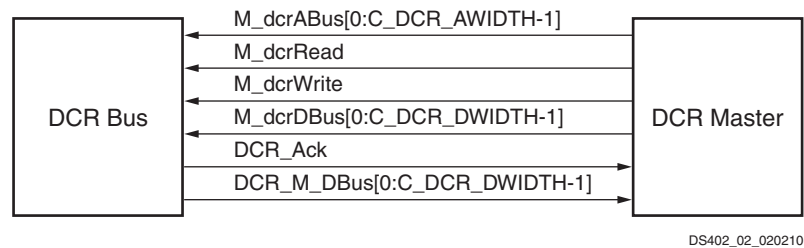
Figure 2: **DCR Master Interface**

DCR Slave Interface

The interface of a DCR slave to the DCR Bus Module is shown in Figure 3 where C_DCR_AWIDTH=10, C_DCR_DWIDTH=32 and the DCR slave number is n.
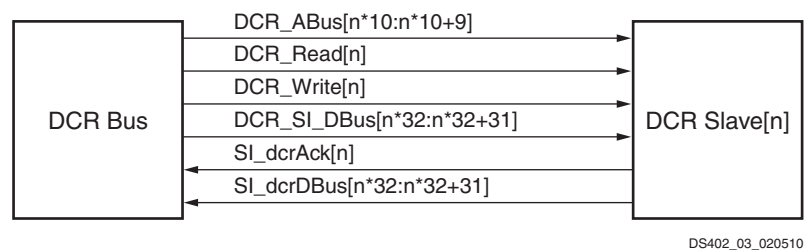

DS402_03_020510

Figure 3: **DCR Slave[n] Interface**

Timing Diagrams

Figure 4 and Figure 5 show DCR_V29 write and read transactions.
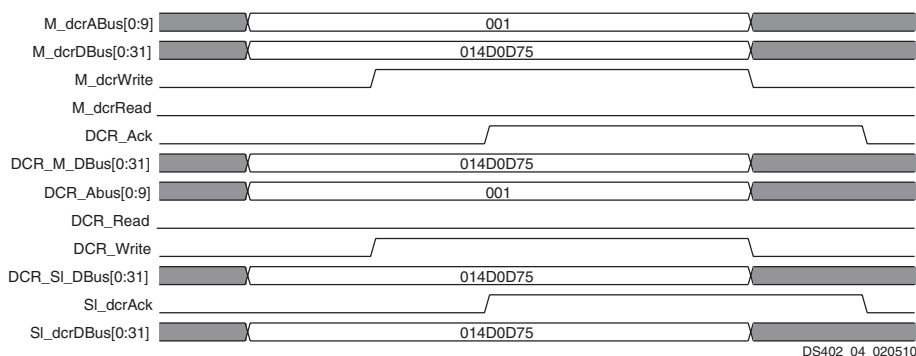

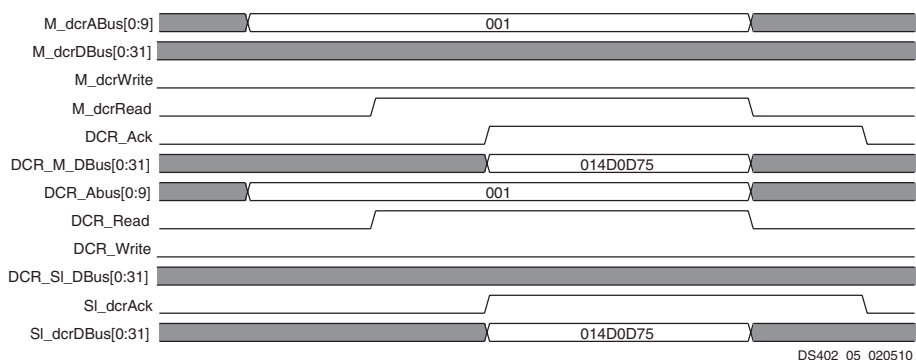*Figure 4:* **DCR v29 Write Cycle Timing Diagram**

Figure 5: **DCR v29 Read Cycle Timing Diagram**

Design Implementation

Target Technology

The target technology is an FPGA listed in the Supported Device Family (1) field of the LogiCORE IP Facts table.

Device Utilization and Performance Benchmarks

Since the DCR is a module that will be used with other design pieces in the FPGA, the utilization and timing numbers reported in this section are just estimates. As the DCR is combined with other pieces of the FPGA design, the utilization of FPGA resources and timing of the DCR design will vary from the results reported here.

To analyze the timing of the DCR within the FPGA, a design was created that instantiated the DCR with registers on all of the DCR inputs and outputs. This allowed a constraint to be placed on the clock net for the DCR to yield more realistic timing results. The f_{MAX} parameter shown in Table 5 was calculated with registers on the DCR inputs and outputs. Note however, that the resource utilizations reported in Table 5 do not include the registers on the DCR inputs and outputs.

The DCR benchmarks are shown in Table 4 are for a Virtex®-4 FPGA using multi-pass place and route.

Table 4: **Performance and Resource Utilization Benchmarks on Virtex-4 FPGA (xc4vlx200ff1513-10)**

Parameter Values				Device Resources			f_{MAX} (MHz)
C_DCR_NUM_SLAVES	C_DCR_AWIDTH	C_DCR_DWIDTH	C_USE_LUT_OR	Slices	Slice Flip-Flops	LUTs	f_{MAX}
1	10	32	1	0	0	0	568
2	10	32	1	0	0	1	444
3	10	32	0	0	0	1	201
4	10	32	1	1	0	1	407
4	10	32	0	1	0	1	407
5	10	32	1	1	0	1	279
6	10	32	0	1	0	2	322
7	10	32	1	1	0	2	166
8	10	32	1	2	0	3	143
8	10	32	0	1	0	2	366

Notes:

1. These benchmark designs contain only the DCR with registered inputs/outputs without any additional logic. Benchmark numbers approach the performance ceiling rather than representing performance under typical user conditions.
2. Device resource numbers do not include the registers for the DCR I/O.
3. Max frequency calculated with registers on the DCR I/O.

The DCR benchmarks are shown in Table 5 are for a Virtex-5 FPGA using multi-pass place and route.

Table 5: **Performance and Resource Utilization Benchmarks on Virtex-5 FPGA (xc5vlx220ff1760-2)**

Parameter Values				Device Resources			f_{MAX} (MHz)
C_DCR_NUM_SLAVES	C_DCR_AWIDTH	C_DCR_DWIDTH	C_USE_LUT_OR	Slices	Slice Flip-Flops	LUTs	f_{MAX}
1	10	32	1	0	0	0	839
2	10	32	1	0	0	1	226
3	10	32	0	0	0	1	248
4	10	32	1	0	0	1	262
4	10	32	0	0	0	1	286
5	10	32	1	0	0	1	358
6	10	32	0	0	0	2	304
7	10	32	1	0	0	2	227
8	10	32	1	0	0	2	192
8	10	32	0	0	0	2	258

Notes:

1. These benchmark designs contain only the DCR with registered inputs/outputs without any additional logic. Benchmark numbers approach the performance ceiling rather than representing performance under typical user conditions.
2. Device resource numbers do not include the registers for the DCR I/O.
3. Max frequency calculated with registers on the DCR I/O.

Reference Documents

The following documents contain reference information important to understanding the Xilinx DCR design:

1. *IBM 32-Bit Device Control Register Bus Architectural Specification (v2.9)*

Revision History

Date	Version	Revision
04/23/02	1.0	Initial Xilinx release.
05/20/02	1.1	Update for EDK 1.0
07/24/02	1.2	Add XCO parameters for System Generator
01/07/03	1.3	Update for EDK SP3
07/08/03	1.4	Update to new template
09/11/03	1.4.1	Update graphics to GSC standards
07/18/03	1.4.2	Correct trademarks
06/24/04	1.5	Remove references to any device family except V2p, per CR 190407
8/9/04	1.6	Updated trademarks and supported device families listing; inserted cross-references for tables and figures; made minor format and content edits
7/18/05	1.7	Converted to new data sheet template
05/18/06	1.8	Updated for Virtex-5 FPGA support
4/24/09	1.9	Replaced references to supported device families and tool name(s) with hyperlink to PDF file
12/2/09	2.0	Updated to v1.00b for EDK_L 11.4 release.
4/19/10	2.1	Converted to current data sheet template, updated images to graphic standard.

Notice of Disclaimer