PCI 9054/PCI 9054 AN

July 31, 2000

Version 2.0

PCI 9054 to PCI 9054 Shared Local Bus Application Note

Features

- Two PCI 9054 sharing the same local bus.
- Local Bus Arbiter Code.
- Two PCI Bus.

General Description

PLX Technology PCI 9054 2.2 compliant 32 bit, 33Mhz PCI Bus Master I/O Accelerator with PCI Power Management features for adapters and embedded systems. The PCI 9054 has Direct Master, DMA and Direct Slave data transfer capabilities. The Direct Slave gives a master device on the PCI Bus the ability to access memory on the Local Bus. The PCI 9054 allows the Local Bus to run asynchronously to the PCI Bus through the use of bi-directional FIFOs. In this design example, two PCI 9054s share the same Local bus.

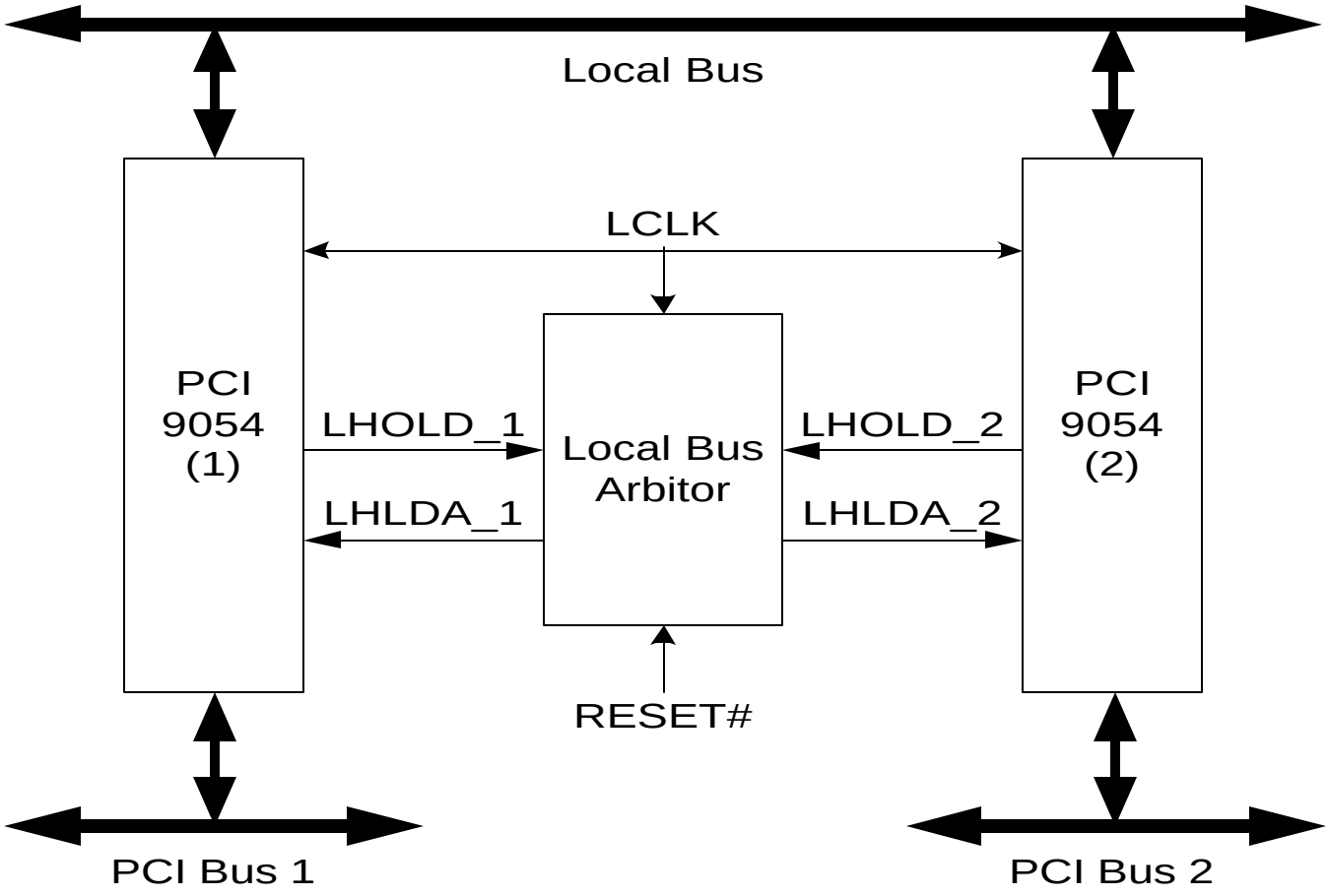

Figure 1. Two (2) PCI 9054 with Shared Local Bus

PLX Technology, Inc, 390 Potrero Avenue, Sunnyvale, CA 94085, Phone 408-774-9060, Fax 408-774-2169

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1. INTRODUCTION

This applications note contains two PCI 9054 sharing the same local bus. The PCI 9054s are attached to different PCI buses. Local bus arbitration is done by a CPLD. Sample verilog code for the CPLD is included.

2. Architecture

Shown in Figure 1-1 are two PCI 9054 sharing the same local bus, but on different PCI buses. Each PCI 9054 has its LHOLD/LHOLDA pair attached to a Local Bus Arbiter CPLD. The CPLD is clocked with the same local clock as both PCI 9054s. The CPLD is locally reset.

3. Local Bus Arbitration

A CPLD performs the Local bus arbitration for the two PCI 9054s. A PCI 9054 asserts LHOLD to request the Local Bus. It owns the Local Bus when LHOLD and LHOLDA are asserted. A PCI 9054 releases the Local Bus by negating LHOLD and floating its local bus outputs.

4. Arbitration State Machine

The state machine that implements the arbitration is discussed here. Figure 3-1 shows the state diagram for the arbiter.

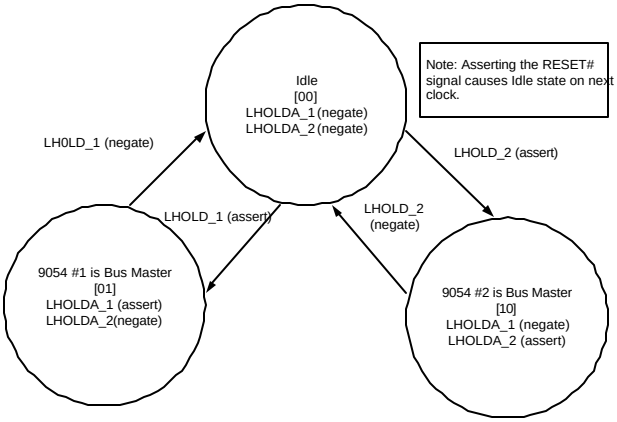

Figure 2. State Diagram of Local Bus Arbiter

If there is not requests for the Local bus it moves to the idle state. If PCI 9054 #1 requests the bus by asserting LHOLD_1 it is granted the bus on the next clock by the CPLD asserting LHOLDA_1. Once the PCI 9054 #1 is done with the bus it will negate its LHOLD_1 and the arbiter will negate LHOLDA_1 on the next clock. The arbitration for PCI 9054 #2 is the same. Note that there is no protection in case both PCI 9054 request the bus as the same time. The state table below describes the states and their causes.

Table 1. State Table

State Bits	State Description	Cause
00	Idle (no Local Bus Master)	RESET#, or PCI 9054 #1 or #2 negating their LHOLD.
01	9054 #1 is Local Bus Master	The CPLD asserts LHOLDA_1.
10	9054 #2 is Local Bus Master	The CPLD asserts LHOLDA_2.
11	Not Allowed	-

5. Assumptions

1. The state machine assumes that only one of the two PCI 9054 will assert its LHOLD at anyone time.
2. There is no other local bus master.

6. REFERENCES

The following is a list of additional documentation to provide the reader with further information about the PCI 9054.

- PCI 9054 Data Book
  PLX Technology, Inc.
  390 Potrero Avenue
  Sunnyvale, CA 94085 USA
  Tel: 408-774-9060, 800-759-3735
  Fax: 408-774-2169
  http://www.plxtech.com

7. CPLD Verilog Source Code

```
/*************************************************************************/
/* MODULE: arbiter.v                                                     */
/* AUTHOR: PLX Technology                                                */
/* DATE:   Dec 2-1999                                                    */
/*                                                                       */
/* REVISION HISTORY:                                                     */
/* Rev 1.0                                                               */
/*                                                                       */
/* DESCRIPTION:                                                          */
/*                                                                       */
/*  This is sample local bus aribiter for two PCI 9054s                  */
/*                                                                       */
/*************************************************************************/

`timescale 1ns/1ns

module      arbiter(
            clk_i,
          reset_L_i,

            lhold_1_i,
            lholda_1_o,

            lhold_2_i,
            lholda_2_o
           );
//
// inputs and outputs
//
input clk_i;
input reset_L_i;
input lhold_1_i, lhold_2_i;

output lholda_1_o, lholda_2_o;

reg lholda_1_o, lholda_2_o;

// lholda_1_o : always statement //
always @ (posedge clk_i)
  begin
    if (reset_L_i == 1'b0)
      begin
        #5 lholda_1_o = 1'b1;
      end
    else
      begin
        if (lhold_1_i == 1'b0)
          begin
            #5 lholda_1_o = 1'b0;
```

```
end
        else
          begin
            #5 lholda_1_o = 1'b1;
          end
      end
  end

// lholda_2_o : always statement //
always @ (posedge clk_i)
  begin
    if (reset_L_i == 1'b0)
      begin
        #5 lholda_2_o = 1'b1;
      end
    else
      begin
        if (lhold_2_i == 1'b0)
          begin
            #5 lholda_2_o = 1'b0;
          end
        else
          begin
            #5 lholda_2_o = 1'b1;
          end
      end
  end

endmodule
```