Digital 21272/21274 Emulator

Design Specification

Author: Jonathan D. Belanger  
Creation Date: March 24, 2018  
Modify Date: March 24, 2018

Table of Contents

[1 Design Overview 3](#_Toc509665545)

[1.1 Cchip 3](#_Toc509665546)

[Table 1‑1PCI and 21274 Lexicon 4](#_Toc509665583)

# Overview

The Digital Alpha AXP 21264 Emulator (DECaxp) is loosely based on the DS40 and ES40 systems, and the es40 emulator. The es40 emulator had several design optimizations that made its CPU and System Board emulation not behave like a real ES40. Some of these design choices made it difficult to emulate an Symmetric Multi-Processor (SMP) System. The goal of the DECaxp emulator is:

* To be as close to the actual hardware as is reasonable.
* Be designed as a multi-threaded emulator.
* Be able to execute 4-integer and 2-floarting point operations simultaneously.
* Have a separate System Board (System) emulation in its own set of threads.

With the above in mind, the CPUs interface to the System like that of the D/ES40.

One of the things that the emulation does not need to worry about is the physical limitations of system design. As such, multiple chips required in the System, can generally be implemented as a just one (e.g., the physical implementation of the D/ES40 has one of 2, 4, or 8 Dchips).

The System emulation for the DECaxp emulator, is based on the 21272 (Tsunami) and the 21274 (Typhoon) chip sets. The only significant difference between the two is the number of CPUs supported, 2 versus 4, and the amount of memory that can be address (8GB versus 64GB). From this point forward, references to 21274 will also be valid for 21272, with the above constraints.

The System is comprised of 3 basic building blocks. These building blocks are the:

* Cchip – Interfaces with the CPUs, Pchip, and Dchip, and TIGbus
* Pchip – interfaces with the Cchip, Dchip, and PCI
* Dchip – interfaces with the CPUs, Cchip, Pchip, and Memory

## System Interface

The system interface Is utilized by the Cchip and Dchips. There are 2 buses involved, one if for command information and the other is for data. The Cchip deals with the command information and the Dchip deals with the data. So, we can the following communications from the CPU to 21274:

Table ‑ CPU 🡪 21274 Communications

| **Command** | **Data Movement** | **Description** |
| --- | --- | --- |
| Read Block | No | Read from Memory. |
| Read I/O | No | Read from I/O. |
| Write Victim | Yes | Write to Memory, data is sent to system. |
| Write I/O | Yes | Write to I/O, data is sent to system. |
| Change to Dirty | Yes or No | Change the state of a cache block from Clean, Shared, STx\_C, or Invalid to Dirty. The Invalid to Dirty can also include a Victim. Data may be sent to system. |
| MB | No | Memory Barrier. No data is sent to system. |
| Flush Request | No | Request the system to flush its VDB of all Write Victim and Probe entries. No data is sent to system. |
| Probe Response | Yes or No | This is sent as the result of a probe from the system. If the probe indicated data movement (read), then the requested data is sent along with the response. |

Table ‑ 21274 --> CPU Communications

|  |  |  |  |
| --- | --- | --- | --- |
| **Command** | **Data Movement** | | **Description** |
| Probe | | No | Probes request things like reading data and changing the state of a cache block. |
| Read Error | | Yes | System returns data for read commands. |
| Read Data | | Yes | System returns data for read commands. |
| Change to Dirty Response | | No | This response can be a success or a failure. No data is returned by the system. |
| MB Done | | No | This is the response sent by the system to a MB previously sent by the CPU. |
| Release Buffer | | No | No data is returned with this command, but this will release certain buffers within the CPU. |
| Write Data | | Yes | TBD. |

# Design Specifications

## Cchip

In the real implementation, the Cchip sends memory commands and addresses to the SDRAM arrays, and the Dchip moves data to and from the CPUs or Pchips into and out of memory. In the emulation, the Cchip will send the memory control and address, along with the data, to the Dchip.

### Overview

The Cchip takes requests from the CPUs, and Pchip to be processed by other components of the System. It contains one request queue.[[1]](#footnote-1) The request queue has provisions for 24 entries.[[2]](#footnote-2) Additionally, there is a wait queue, which is an ordered list used to maintain ordering rules of the Alpha architecture. The Cchip and Pchip interact to ensure the ordering and deadlock avoidance rules of the PCI Specification Revision 2.1. In this regard, the 21274 performs as a host bridge for CPU PIO and DMA memory access. The 21274 also performs as a PCI-to-PCI bridge for PTP operations from one Pchip to the other.

Table 1‑1 provides an interpretation of the PCI Specification, Rev 2.1 ordering rules in the context of the 21274 chipset.

Table ‑PCI and 21274 Lexicon

|  |  |
| --- | --- |
| **PCI Lexicon** | **21274 Lexicon** |
| PMW – Posted Memory Write | PIO Write Request DMA Write Request PTP Write Request |
| DRR – Delayed Read Request | PIO Read Request DMA Read Request PTP Read Request |
| DWR – Delayed Write Request | N/A (all writes are posted) |
| DRC – Delayed Read Completion | PIO Read Return DMA Read Return PTP Read Return |
| DWC – Delayed Write Completion | N/A (all writes are posted) |

### Design

#### CPU and Pchip Requests

The Cchip receives requests from each of the CPUs and each of the Pchips. The kinds of requests receive are somewhat different.

* The CPUs call the AXP\_System\_CommandSend and AXP\_System\_ProbeResponse functions.
* The Pchips use the AXP\_CAPbusMsg.

These requests are packaged into a Request Entry and queued up to the appropriate (CPU or Pchip) skid buffers. The Cchip will remove these entries from the skid buffers for dispatching (processing).

#### Cchip Request Processing

Each new request that arrives from a CPU or Pchip is dispatched for processing.

A request to the Cchip is for one of the following items:

1. Memory read or write
2. CSR read or write
3. IO read or write
4. Response to IO read or Write
5. Probe request
6. Probe response

TBD: We need to determine where memory and CSR access is going to be processed. Do we need a Dchip to manage this or can the Cchip handle it all. The one caveat is that not only the Cchip, but also the Pchip will want to read and write from memory.

1. The real hardware has four request queues. One for each memory array. [↑](#footnote-ref-1)
2. The real hardware has six entries in each of the 4 queues. [↑](#footnote-ref-2)