Memory SHIP Outline

Greg Gibeling
Wednesday October 11th, 2006
GDG03 – Memory SHIP Outline

1.0 Introduction

This memo attempts to describe, in text, the behavior of the memory SHIP outlined in class on October 4th, and diagrammed in NI01. I will keep this brief, as it is only one of many such descriptions of this SHIP.

2.0 Read

2.1 Single Word Access

A single word read is accomplished by sending the address to be read to the `ReadAddress` port. The data will later be delivered on the `ReadData` port. For this simple case, no tokens or counts are needed.

2.2 Strided, Counted Access

A strided, counted read is initiated upon reception of a `ReadAddress`, indicating the starting address, as well as a `ReadCount` and a `ReadStride`. Data is produced at `ReadData` upon reception of each `ReadNext` token, and the final data word is accompanied by the production of a `ReadDone` token.

Because the use of `ReadDone` cannot be synchronized with the use of the final word of `ReadData`, `ReadDone` is truly only useful for consistency and coherence management, whereas the actual end of the data stream is easy enough to count. `ReadDone` therefore indicates that the last word has been read from memory, but not necessarily delivered to its point of use, or even out of the memory SHIP.

3.0 Write

3.1 Single Word Access

A single word write is accomplished by sending the address to be written to the `WriteAddress` port, and the data to be written to the `WriteData` port. Again, no tokens are needed.

3.2 Strided, Counted Access

A strided, counted write is initiated upon reception of a `WriteAddress`, `WriteCount` and `WriteStride`. Upon reception of each word of data at `WriteData`, it is sent to the memory and a `WriteNext` token is produced. Once the final word of data has been fully committed to memory, a `WriteDone` token will be produced as well.

4.0 Multiple Ports

In order to keep single word, and strided, counted accesses separate the final bits of the SHIP port address are used to decide whether an access will be single or multi-word. In one sense this multiplexes the existing SHIP ports, in another it simply provides separate ports for single and multi-word access. I recommend the later view, as it could be highly desireable to replicate the ports, and even to have more of one kind or another.

5.0 Conclusion

In general the memory SHIP seems well designed. In particular I imagine the multi-word access methods could be optimized to e.g. bypass cache, and even set up optimized SDRAM transactions, taking advantage of recent advances in DRAM interfaces in a new and very powerful way.

The only flaw of this design is over the confusion between whether the lowest bits of SHIP port address (those that are “decoded by the SHIP”) actually represent different physical ports, with different buffers, or simply different meanings for the same port.