AN1707

Dual Port Memory for Multiprocessor Applications

Prepared by: Jim Lewis and Lakshmi Mandyam

The most common implementation of a multiprocessor (MP) system is a bus–based multiprocessor, i.e., one where the processors in the system share a common system bus. In this implementation, the system bus is the sole avenue to access main memory and the system's I/O devices. Because of this, the system bus becomes a bottleneck causing performance degradation. The use of external cache memory can help alleviate this problem. There are various issues that should be considered when implementing external caches in MP systems. This applications note discusses the various implementations of external caches and the pros and cons of each implementation. In particular, this document will illustrate the advantages of using Motorola's dual port SRAM, the MCM69D618, for both the tag and data RAM of an inline cache. First we will discuss the issues involved with shared bus architectures, and then we will discuss the issues involved with inline cache implementation.

ISSUES WITH SHARED BUS ARCHITECTURES

A major problem in such a system is that the shared bus becomes a bottleneck. Each microprocessor has a large appetite for data which will require frequent accesses to memory. Today's microprocessors have large internal caches but these are not enough to satisfy all requests. Each processor has to compete with other devices for access to the shared system bus to satisfy its internal data requirements. With processor internal clock speeds reaching the hundreds of megahertz range, but system bus speeds remaining in the 66 to 133 MHz range, the gap between the two is widening rapidly. Therefore, requests from the processor can queue up more quickly than the main memory system can satisfy them, leading to system saturation and performance degradation.

A solution to these problems is larger external caches that can satisfy more of the processors' memory requests, thus taking some of the burden off of the main memory controller. However, if these external caches are resident on the system bus, in addition to everything else already present, the bus bottleneck problem is still not solved.

The ideal solution is to make the external caches "inline", i.e., a given cache sits logically between the system bus and the processor being supported by the cache. This allows the processor to have unimpeded access to its own external cache and also limits the traffic on the system bus to those transactions which cannot be satisfied by the external caches.

The system shown in Figure 1 is a generic MP system. There are four microprocessors connected to the system bus through an inline cache. The System bus also has I/O and memory controllers connected to it which are in turn connected to I/O devices and main memory respectively.

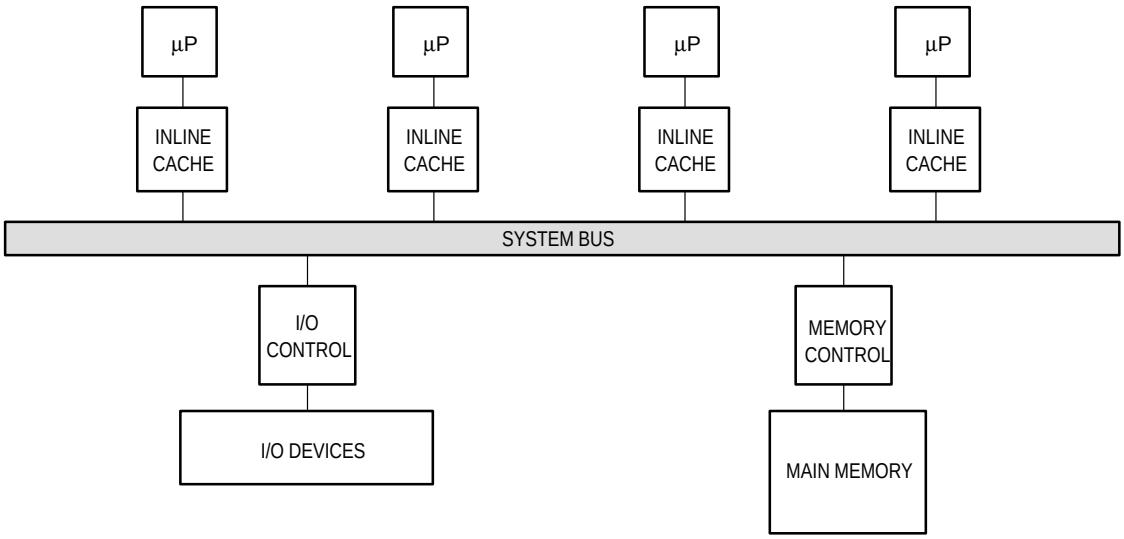


Figure 1. Generic MP System

NetRAM is a trademark of Motorola, Inc.

1/31/97

As with most designs, there is more than one way to solve the problem of implementing an inline cache. There are two basic assumptions to be made about the cache:

- It will require two logical forms of RAM: tag and data
- A controller will be needed to communicate with both the processor and system buses as well as control the tag and data RAMs.

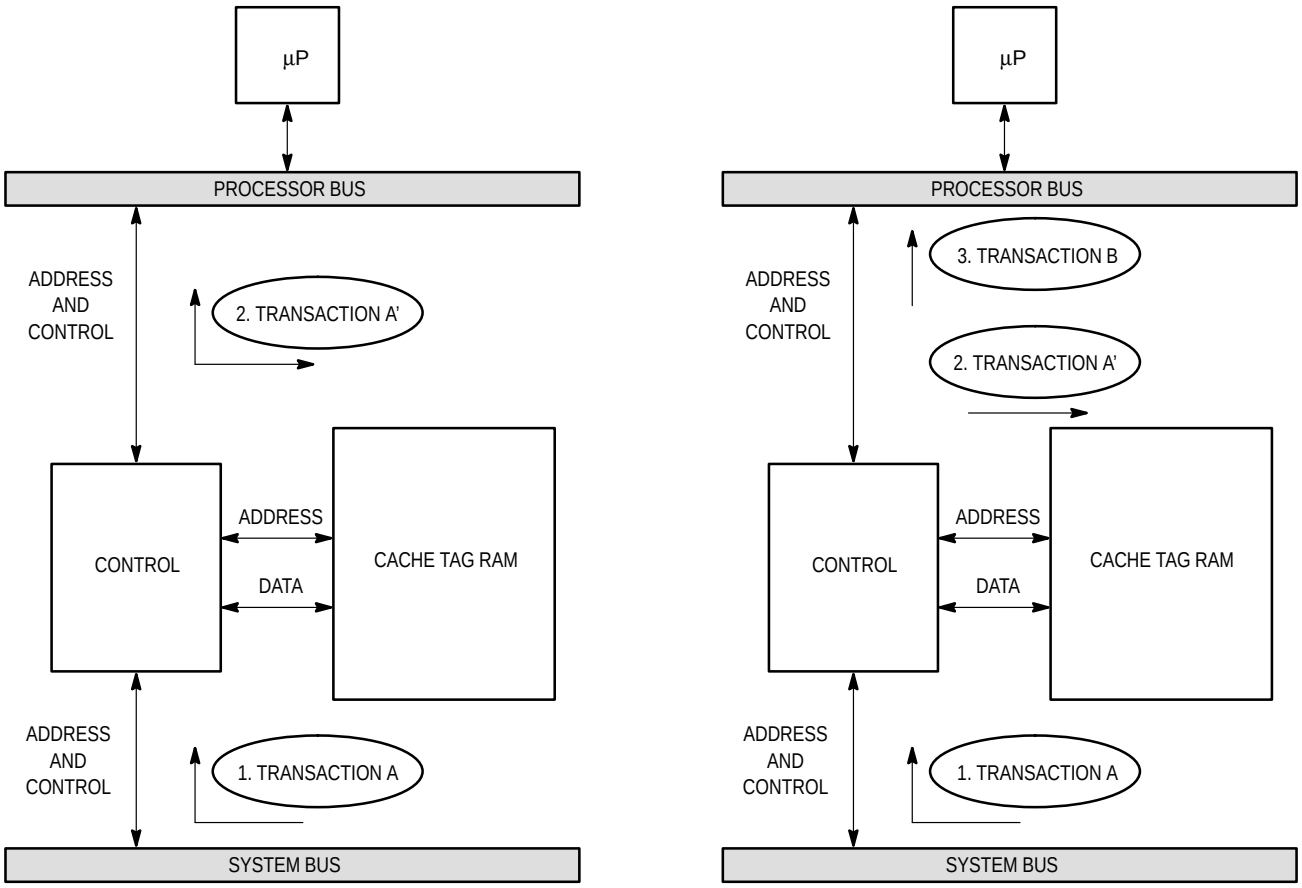
To better understand the advantages of each inline cache implementation, we will first examine cache coherency with inline caches.

CACHE COHERENCY WITH INLINE CACHES

In addition to limiting system bus traffic, an inline cache helps to maintain the coherence of system memory by monitoring all transactions that occur on the system bus. In particular, the inline cache "snoops" or watches for transactions which may require state changes to cached data due to the cache coherence protocol of the system. These transactions shall be referred to as snoop transactions. When a snoop transaction is detected, the inline cache is responsible for ensuring that the data in both the processor's on–chip cache and the inline's own data RAM transition to the proper state when the transaction completes.

A straightforward method of ensuring that coherence is maintained is for the inline cache to mimic all snoop transactions on the processor bus. For example, when the inline cache detects snoop transaction A on the system bus, it will initiate transaction A' on the processor bus. See Figure 2a. This, in effect, causes the processor to see all snoop transactions as if the inline cache were not there. While transaction A' is proceeding, the inline cache will also check its own tag to determine if any state changes are necessary. This implementation negates one of the advantages of having an inline cache, i.e., isolating the processor from the system so that it will have higher bandwidth access to its external cache.

Processor isolation can be achieved by the inline cache with a process called snoop filtering. Snoop filtering limits the snoop transactions propagated by the cache to the processor bus from the system bus. "Inclusion" is a necessary condition for snoop filtering. Inclusion means that the contents of the processor's cache are a subset of the contents of the external cache. By maintaining inclusion, the inline cache can now query its own tag RAM when it detects a snoop transaction. So when there is a snoop transaction A on the system bus, the cache generates a transaction A' to the cache tag. If the cache tag does not contain the memory locations affected by snoop transaction A then no further action needs to be taken and the processor will never be made aware that the transaction even occurred. However, if the inline cache determines that the snoop transaction affects its own contents, it will propagate the snoop transaction as transaction B to the processor bus. See Figure 2b.



a. Without Snoop Filtering **b. With Snoop Filtering**

Figure 2. Cache Coherency Implementations

The inline cache, as previously described, now has the task of handling requests from two different sources: processor–initiated transactions on the processor bus and snoop transactions on the system bus. Both of these transactions require access to the tag RAM of the inline cache.

One utilization of this shared resource is to have a single physical tag RAM as shown in Figure 3. In this design, the inline cache implements an arbitration scheme to resolve collisions between two simultaneous requests. While this implementation may be the least expensive, it imposes a performance penalty on the processor by forcing it to give up access to its external cache whenever there is a snoop transaction occurring on the system bus.

Another design uses two physical copies of the tags, one dedicated to servicing processor transactions and the other dedicated to queries from snoop transactions as shown in Figure 4. While this allows the processor the greatest access to the external cache it has the added cost of twice as much tag RAM and the overhead of maintaining two copies of the tag.

The third alternative is implemented using Motorola's MCM69D618 as the tag RAM as shown in Figure 5. The MCM69D618 is a dual port memory with two address ports (AX,AY) and two data ports (DQX, DQY) into the same SRAM array. The MCM69D618 can be clocked at 83 MHz and the user can perform reads and writes to different addresses simultaneously. Figure 5 illustrates how the MCM69D618 could be situated within the inline cache subsystem. The X port can be used to satisfy processor requests and the Y port can be used to query the tag when snoop transactions are detected. It is assumed that to prevent overloading the bus, the Y address port (AY) cannot be directly connected to the system bus. By using the MCM69D618 in this manner, an excellent cost–effective solution is obtained without compromising performance.

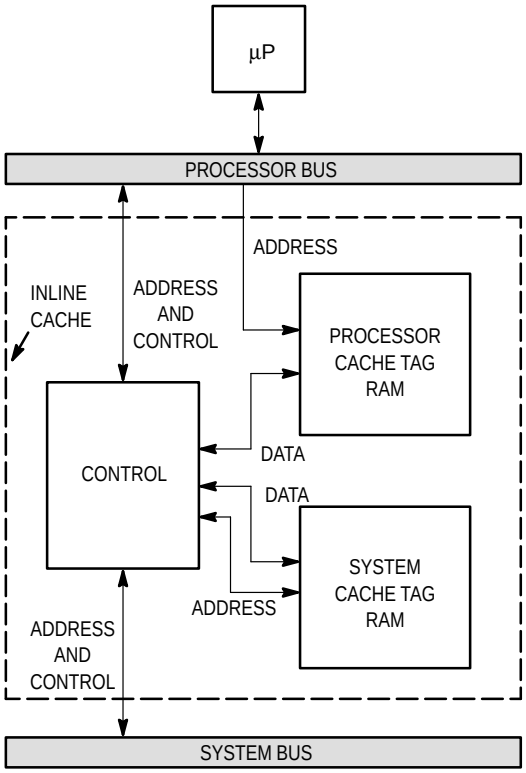


Figure 4. Inline Cache with Two Tag RAMs

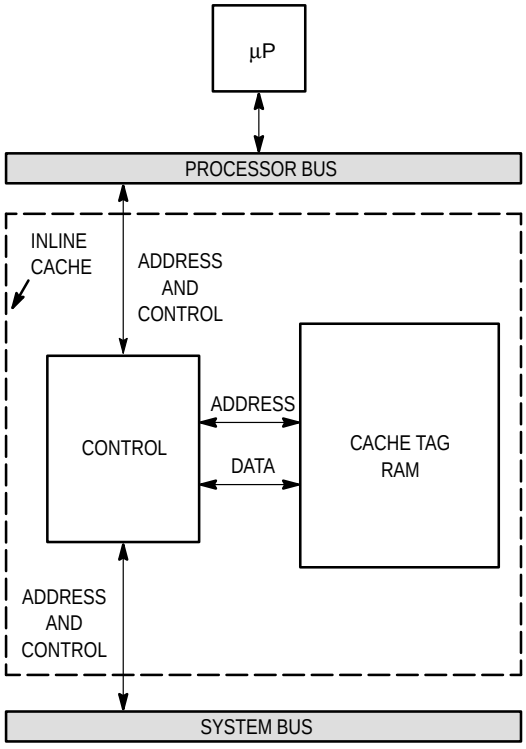


Figure 3. Inline Cache Implementation with a Single Tag RAM

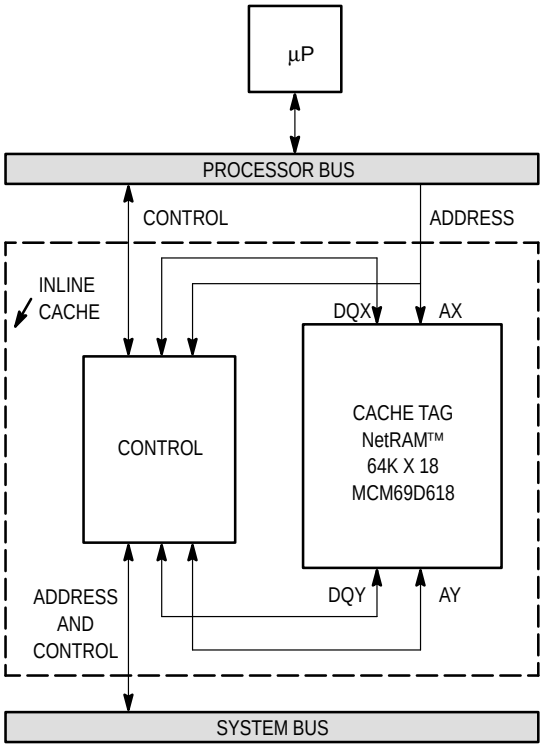


Figure 5. Inline Cache Tag Using Dual Port SRAM

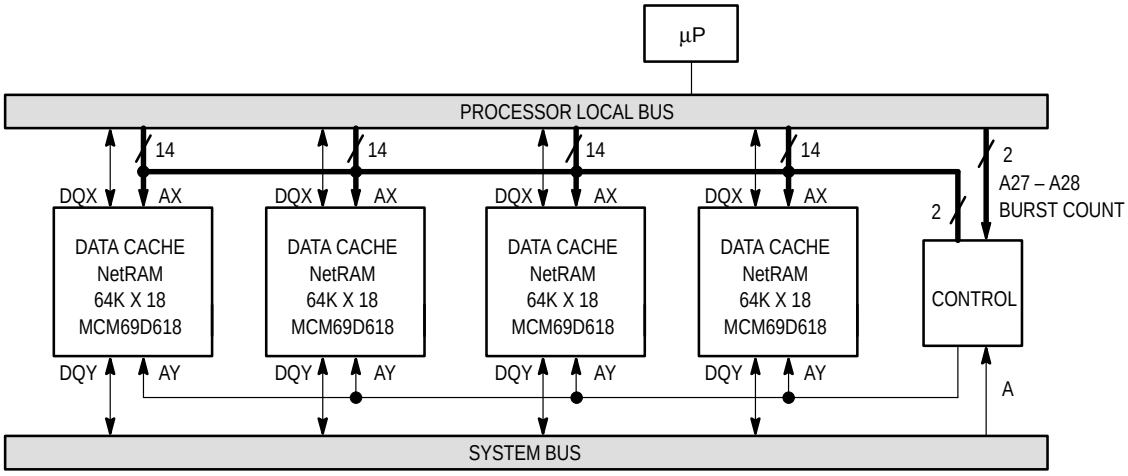


Figure 6.

DATA RAM IMPLEMENTATION IN AN INLINE CACHE

Another function that must be performed by an inline cache is the transfer of data between the processor and system buses. Assuming 64–bit buses plus parity, the inline cache must have 144 signal pins just for the data path between the two buses. The MCM69D618 has a pass through function which can write the data input from one data port to the other enabling the RAM to serve as a data path from the processor bus to the system bus. Therefore, by using MCM69D618 as the building block for the data RAM of the inline cache, the data signal pins can be removed from the inline cache control device.

Figure 6 shows four MCM69D618s being used to implement a 512KB cache. As with the tag RAM described earlier, the X port connects to the processor bus and the Y port connects to the system bus. Again, the Y port addresses are assumed to come from the inline cache controller rather than the system bus to reduce loading. Since the MCM69D618 is not a BurstRAM, the two least significant bits of the X port address will need to come from the inline cache controller to support burst responses to processor requests.

The configurable I/Os let the user read from or write to either of the data ports. Although the clock speed is 83 MHz because of the dual port feature, we can get the same performance as a 166 MHz single address RAM. The MCM69D618 provides a cost–effective method of implementing an inline cache for systems that have cost constraints placed on them.

CONCLUSION

This document has shown how the MCM69D618 can be used as the building block to implement an inline cache. The MCM69D618 provides an excellent solution for the tag RAM without compromising performance. It can also facilitate a cost–effective solution for the data RAM at the expense of a lower performance system.

The MCM69D618 system performance enhancing features are:

- The external clock runs at 83 Mhz.
- The fast access times of 6 ns help boost performance.
- The dual addressing scheme enables the user to perform simultaneous read/writes, reads/reads, writes/writes in the same clock cycle. This provides performance equivalent to 166 MHz of a conventional single address RAM.
- The price will be at a better competitive price point compared to other dual ports of this depth and performance.

Thus the MCM69D618 can greatly enhance system price/performance.

How to reach us:
USA / EUROPE / Locations Not Listed: Motorola Literature Distribution; P.O. Box 5405, Denver, Colorado 80217. 303–675–2140 or 1–800–441–2447

Mfax™: RMFAX0@email.sps.mot.com – TOUCHTONE 602–244–6609
– US & Canada ONLY 1–800–774–1848

INTERNET: http://motorola.com/sps

JAPAN: Nippon Motorola Ltd.: SPD, Strategic Planning Office, 4–32–1, Nishi–Gotanda, Shinagawa–ku, Tokyo 141, Japan. 81–3–5487–8488

ASIA / PACIFIC: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park, 51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852–26629298