The 68030 and 68040 on the Zorro III Bus

by Michael Sinz

The Zorro III bus presents several special design issues for systems with either a 68030 or 68040 CPU. This article discusses those design issues and offers solutions to the potential problems that they present to those developing Zorro III devices, in particular, Zorro III devices that are not exclusively memory expansion devices.

Background - 68030 and 68040 Caches

Both the 68030 and the 68040 have two caches; one for instructions and one for data. The 68030's caches are 256-bytes long. The 68040's caches are considerably larger: each is 4K long. Both CPU's caches store memory in 16-byte blocks which are referred to as a *cache line*. The CPU only keeps one address for each cache line. Each cache line is further broken up into long words called *cache entries*. On the 68030, each cache entry is marked as either valid or invalid, telling the CPU which long words in the cache still contain valid data. On the 68040, only an entire cache line can be marked as valid or invalid.

When the 68030 caches a memory address, it uses bits four through seven of the address as a hash value. This value is an index that tells the 68030 which cache line to use for a specific address. This means each memory address corresponds to only one cache line. For example, if the 68030 tried to read a long word from 0x0FFFFF9F, since the 68030 index extends from bit four through seven, the index is 0x9, which corresponds to the tenth cache line. This also means that many memory addresses correspond to the same cache line. For example, the addresses 0x01FFFF9F, 0x02FFFF9F, 0x03FFFF9F, and 0x04FFFF9F all correspond to the same cache line.

The 68040 cache uses a similar indexing scheme, but the 68040 is a little more dynamic. The 68040 has four cache lines for each index, so the 68040 cache can hold on to four different addresses that share the same index. The 68040 also has a larger index (six bits).

While the caches are active, when the CPU executes an instruction that reads memory, it will check if that memory is already in the cache. If the memory is in the cache and the cache entry (or, in the case of the 68040, the cache line) is marked as valid, a *cache hit* occurs, so the CPU reads from the cache instead of main memory. If the memory is not in the cache, or the cache entry (or cache line) is marked invalid, a *cache miss* occurs, so the CPU has to perform a main memory fetch.

The cache lines are used in conjunction with the 68030's and 68040's *burst mode*. Normally, the 68030 fills its caches one entry at a time. While in burst mode, the CPU fills its caches a whole cache line at a time. This helps to reduce the number of cache misses.

On the 68030, it is possible to turn burst mode on and off independently of its caches. If the 68030 cache is on and burst mode is off, the 68030 can fill its cache a single long word at a time, rather than the four words at a time it would do in burst mode. The 68040 is different. On the 68040, the only way to turn on burst mode is to turn on the cache, so there is no way to prevent a burst access when using the cache. The 68040 always fills a whole cache line at a time.

The instruction cache on both CPUs is fairly straightforward. As the CPU fetches instructions from memory, it copies them into the cache for quick access later. The only time the CPU changes the instruction cache is when it does a memory fetch.

The data cache is different. The data cache can change when the CPU fetches memory and when the CPU executes an instruction that writes to memory. The 68030 and 68040 deal with this differently.

The 68030 data cache is a *write-through* cache. This means whenever the 68030 executes an instruction that writes to memory, the 68030 always performs a write to main memory, *even if that address is in the data cache*.

When a data write causes a cache miss, the 68030 will act as if it has no data cache and write directly to memory (except in write-allocate mode--see the next paragraph). When a data write causes a cache hit, the 68030 will update the cache entry (or entries) as well as write to memory. Basically, on a memory write operation, the 68030 will only update cache entries that are currently cached. It will not allocate a new cache entry for a cache miss.

The 68030 data cache has a mode called *write-allocate*. In this mode, the 68030 not only updates the data in the cache, but, in the case of a cache miss, the 68030 can also allocate a new cache entry. While in this mode, if a data write causes a cache miss, the CPU first marks the corresponding cache entry as invalid (or, if in burst mode, the CPU marks the entire cache line as invalid). If the data write is a long word write and it is aligned on a long word boundary, the CPU updates that long word in the cache and marks it as valid.

The 68040 data cache is not always a write through cache. It has a mode called *copyback*. While in copyback mode, a write operation on the 68040 will not write through to memory. The data will remain in the data cache until the CPU flushes it out. For more information, see the article, ‘‘68040 Compatibility Warning’’ from the July/August 1991 *Amiga Mail*.

Zorro III and the 68030

When the 68030 reads data from a memory address, it will cache that address only if that memory address is marked as cachable. Certain areas of memory cannot be cachable, for example, hardware registers of a Zorro III card. On the Zorro III bus, when the CPU attempts to read an address that is not cachable, the device that exists in that address space asserts the Zorro III cache inhibit line (/CINH). The bus controller will turn this signal into the CPU’s cache inhibit signal, which tells the CPU not to cache the address.

The problem is with the 68030’s data cache in write-allocate mode (which the Amiga OS requires). When write-allocate mode is disabled, the 68030 will only allocate a cache entry for a data address if the address is cachable. The CPU knows if the address is cachable because the device told it using the cache inhibit line.

While in write-allocate mode, the 68030 will also allocate a cache entry during certain write operations. If, while in write-allocate mode, the 68030 writes a long word to a long word aligned address, the 68030 will write to that address and will allocate a cache entry for that address. This provides a loophole where the 68030 will allocate a cache entry for a non-cachable memory address. If the CPU does a long word write to a Zorro III hardware register that happens to be aligned on a long word address, the 68030 will put that address in the cache. If the CPU attempts to read from that address again and that address happens to still be in the data cache, it will see the value in the cache and will not attempt to read the hardware register.

So far, the conditions under which this loophole can occur have been rare. The loophole requires that a hardware register be both writable and readable, aligned on a long word address, and be four bytes long. This precludes the Amiga custom chip registers as they are not both readable and writable (in general they are not four bytes long either). Zorro II devices don’t apply as Zorro II devices only have a 16 bit wide data path. The small size of the 68030’s data cache also makes it tough for a register write and read to occur without a cache flush happening in between.

However, as Zorro III devices start to hit the market, the conditions under which the loophole can occur will become more commonplace. To avoid this problem, Zorro III card designers can utilize the following hardware trick.

The trick is to ‘‘mirror’’ all of the hardware registers. In this scheme, every register that is both readable and writable is accessible at two addresses. One address is exclusively for reading and the other address is exclusively for writing. Now, if the 68030 performs a write and allocates a cache entry, the 68030 caches the writing address, but not the reading address.

Another hardware trick that might seem to be a viable solution is to align 32-bit register ports so that they do not fall on a long word boundary. Using this method, the 68030 will never cache the register address on a data write because it is not aligned properly. The problem with this method is that reading (or writing) a long word from a non-long word aligned address is considerably slower than from a long word aligned address. This can almost double the amount of bus traffic, making the entire system slower.

The 68040 and Zorro III

The 68040 does not have the problem that the 68030 has with Zorro II space. The 68040 contains two registers to give data space a default mapping without the need of a Memory Management Unit (MMU). On an Amiga with a 68040, Exec uses one of these registers to map the low 24-bits of the Amiga's address space (the Zorro II range, $00000000-$00FFFFFFFF) as non-cachable and serialized[1] .

The Amiga uses the second register to map the remaining memory ($01000000-$FFFFFFFF) as cachable and non-serialized. Because of its mapping, any RAM in this region will yield considerably higher performance than RAM in Zorro II space. Unfortunately, this mapping can cause problems for a Zorro III device that is not RAM.

When the 68040 accesses a Zorro III device that is in cachable address space, the device can still tell the CPU that an address is not cachable by asserting the CPU's cache inhibit line. This overrides the default mapping the 68040 has placed on the address space. However, this does not stop the CPU from doing a full line burst. When accessing address space mapped as cachable, the 68040 will always attempt to read or write a block the size of an entire cache line (four long words).

This presents a problem when the 68040 attempts to read from a Zorro III device that is in cachable address space and the device asserts the CPU's cache inhibit line. The 68040 cannot notice that the Zorro III device asserted the CPU's cache inhibit line until the 68040 reads the first long word of the burst cycle. By the time the 68040 sees that the first long word is not cachable, it is already too late to stop the burst cycle, so the 68040 finishes the burst. When the burst is done, the 68040 throws out the extra three long words from the burst read. In this case, the 68040 performs four long word memory accesses instead of just one.

---

[1] The 68040 is capable of rearranging the byte order of memory accesses to make them faster. If the 68040 knows a memory range is serialized, it knows it must perform the memory access in order.

Writing to the device is even worse. When the 68040 writes data to an address that is not currently in the data cache, the 68040 will first try to fill a cache line. When the 68040 sees that the device asserts the CPU's cache inhibit line, it will finish the read and then write out one long word. Essentially, to perform a single memory write, the 68040 performs four memory reads and one memory write.

These excessive memory accesses can significantly hinder system performance. Certain Zorro III designs could make the 68040 as much as four to five times slower.

The full line bursts also cause a second potential problem for some possible Zorro III devices. Reading certain types of hardware registers will trigger the hardware to perform some extra function as well. It is not uncommon for a hardware device to supply a new data value for a register after the CPU reads that register. If a Zorro III device has such a register and the device is located in cachable address space, the device can experience problems with reads and writes of addresses surrounding the register. If the CPU reads a second hardware register at an address that is in the same quad-long word as the register (i.e. the first register's address would be in the same cache line as the second register's address), when the CPU performs its full line burst, it will read the first register in addition to the second register. Because the CPU reads the first register, the device will reload the first register with a new value, losing the previous value.

The Solution

There is a solution that will fix both potential problems for Zorro III cards on 68040-based Amigas. The MMU in the 68040 can map specific pages of memory as non-cachable. The 37.10 version of the *68040.library* creates MMU tables that map only Zorro III memory devices as cachable (actually it maps all RAM except Chip RAM as cachable). The library marks other Zorro III devices as non-cachable. The new library prevents the 68040 from doing full line bursts to non-cachable devices, so the CPU only reads or writes one long word at a time. As the *68040.library* uses the MMU to map all address space, invalid addresses can no longer cause bus errors (Guru #00000002), which may help a few ill-behaved products to work on 68040 systems.

The 37.10 *68040.library* is part of the V39 OS present on the current A4000. Developers who are working on or have released 68040-based expansion devices should contact CATS to obtain information on distributing the library with their product.

There is only one problem with this solution. Not all 68040s have MMUs. There are three kinds of 68040 chip: the MC68040, the MC68LC040, and the MC68EC040. The MC68040 has both an Floating Point Unit (FPU) and an MMU. The MC68LC040 is a regular MC68040 without an FPU. The MC68EC040 is a MC68LC040 without an MMU.

As the *68040.library* requires an MMU to map address space, the fix described above will not work on systems with an MC68EC040. Because burst mode on the 68040 is activated along with the cache, there is no way to prevent a 68EC040-equipped Amiga from doing full line bursts when accessing cachable address space. This means a 68EC040 cannot prevent the excessive reads and writes when reading non-cachable Zorro III devices that reside in cachable address space. *A 68EC040-equipped Amiga will experience a significant decrease in performance when accessing non-cachable Zorro III devices.* ***For this reason we cannot recommend that anyone use a 68EC040 (or any future 68000 series CPU that has no MMU) as the CPU on a Zorro III bus system.***

If someone decides not to heed this warning and create a 68EC040 CPU card for the Zorro III bus, there is nothing the 68EC040 card can do to prevent these problems, although there is still a way for a Zorro III card to prevent the second 68040 problem (the ‘‘register trigger’’ problem). A Zorro III card that needs to use ‘‘trigger’’ registers can arrange the trigger registers so that each register is in its own quad-long word. This way, when the 68EC040 reads one of these registers, the read operation won’t disturb other registers, as two registers do not reside in the same quad-long word. Note that this fix will not prevent the first problem (the performance decrease problem) and it does not address the possibility that future CPUs may have an eight or sixteen long word cache line.