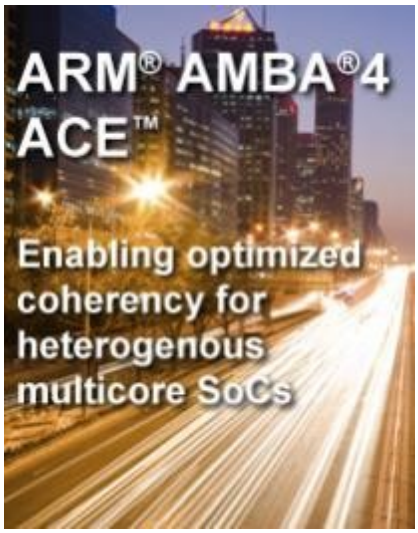


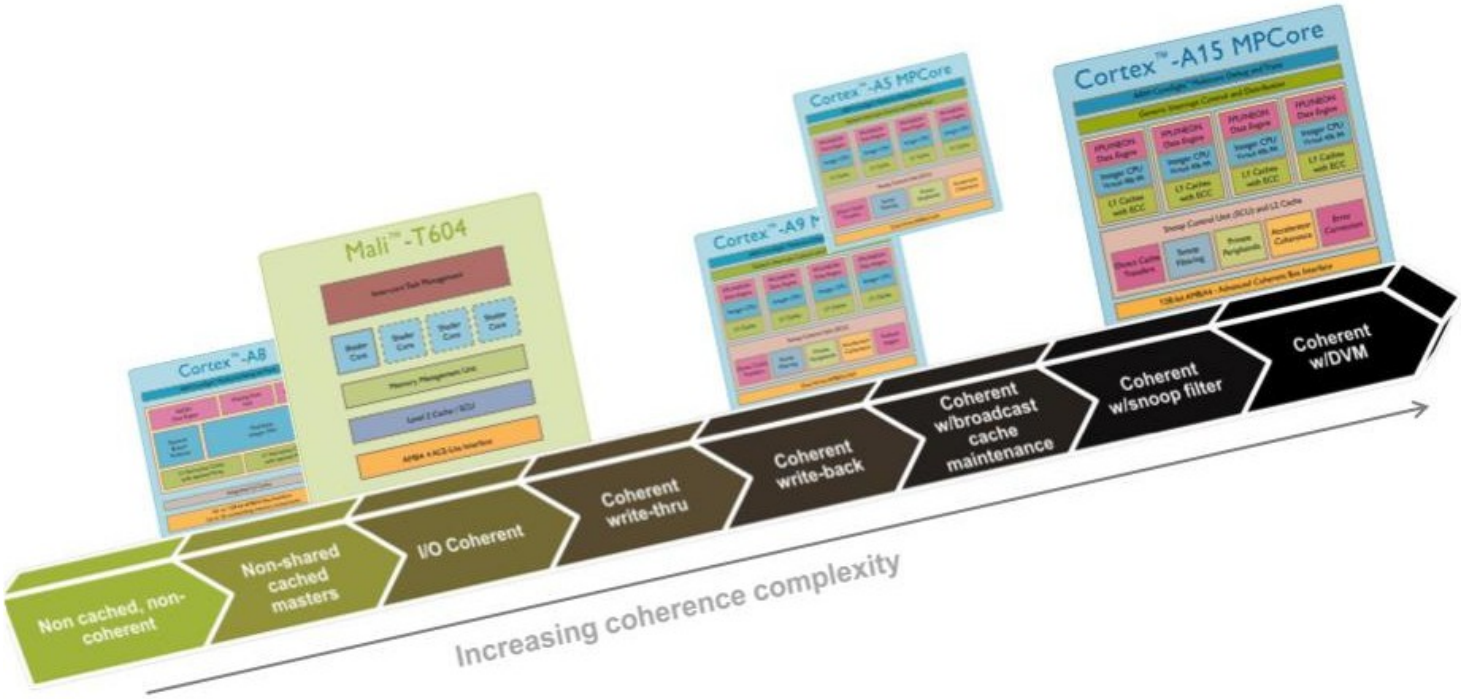
On-chip supercomputers, AMBA 4, Coore's law

by **Paul McLellan** Published on 07-11-2011 10:45 AM

At DAC I talked with Mike Dimelow of ARM about the latest upcoming revision to the AMBA bus standards, AMBA 4. The standard gets an upgrade about every 5 years. The original ARM in 1992 ran at 10MIPS with a 20MHz clock. The first AMBA bus was a standard way to link the processor to memories (through the ARM system bus ASB) and to peripherals (through the ARM peripheral bus APB). Next year ARM-bsed chips will run at 2.5 Ghz and deliver 7000 MIPS.

Eric's story of Thomson-CSF's attempt to build a processor of this type of performance in 1987 points out that in those days that would have qualified as a supercomputer.

The latest AMBA standard proposal actually steals a lot of ideas from the supercomputer world. One of the biggest problems with multi-core computing once you get a lot of cores is the fact that each core has its own cache and when the same memory line is cached in more than one place they need to be kept coherent. The simplest way to do this, which works fine for a small number of cores, is to keep the line in only one cache and invalidate it in all the others. Each cache monitors the address lines for any writes and invalidates its own copy, known as snooping. As the number of cores creeps up this become unwieldy and is a major performance hit as more and more memory accesses turn out to be to invalidated lines that therefore require an off-chip memory access (or perhaps another level cache, but much slower either way). The problem is further compounded by peripherals, such as graphics processors, that access memory too.



The more complex solution is to make sure that the caches are always coherent. When a cache line is written, if it is also in other caches then these are updated too, a procedure known as snarfing. The overall goal is to do everything possible to avoid needing to make an off-chip memory reference, which is extremely slow in comparison to a cache-hit and consumes a lot more power.

The news AMBA 4 supports this. It actually supports the whole continuum of possible architectures, from non-coherent caches (better make sure that no cores are writing to the same memory another is reading from) through the fully coherent snooped and snarfed caches described above.

I'll ignore the elephant in the room of how you program these beasts when you have large number of cores. Remember, what I call Coore's law: Moore's law means the number of cores on a chip is doubling every couple of years, it's just not obvious yet since we're still on the flat part of the curve.

The other big hardware issue is power. On a modern SoC with heterogeneous cores and specialized bits of hardware then power can often be reduced by having a special mini-core. For example, it is much more power-efficient to use a little Tensilica core for MP3 playback than to use the main ARM processor, even though it is "just software" and so the ARM can perfectly well do it. Only one of the cores is used at a time: if no MP3 is playing the Tensilica core is powered down, if MP3 is playing then the ARM is (mostly) idle.

However, when you get to symmetrical multiprocessing, there is no point in powering down one core in order to use another: they are the same core so if you didn't need the core then don't put it on the chip. If you have 64 cores on a chip then the only point of doing it is because at times you want to run all 64 cores at once. And the big question, to which I've not seen entirely convincing answers, is whether you can afford power-wise to light up all those cores simultaneously. Or is there a power limitation to how many cores we can have (unless we lower their performance, which is almost the same thing as reducing the number of cores).

The AMBA 4 specification can be downloaded here.