

A nine layer bus matrix and 18 DMA channels overcome data bottlenecks in graphics intensive applications.  
By **Graham Pitcher.**

# Buses avoid delays

In the last three decades or so, the microprocessor has changed almost unrecognisably. The first recognisable microprocessor was Intel's 8080. The 8bit part, which ran at 2MHz, established the x86 dynasty. Today, Intel is manufacturing quad core devices – effectively bringing supercomputer level performance to the desktop.

Intel, of course, is just one microprocessor manufacturer. All have taken advantage of Moore's Law to turn the clock rate up with each new process technology. But certain elements haven't always kept up with what seemed to be clock rate centric development.

Atmel points to graphics intensive applications as an example. It claims the microcontrollers that drive these applications must store, process and move 'massive amounts' of data at high rates between peripherals, processor and memory.

Dany Nativel, technical product marketing manager for Atmel, believes that clock rate isn't the only important

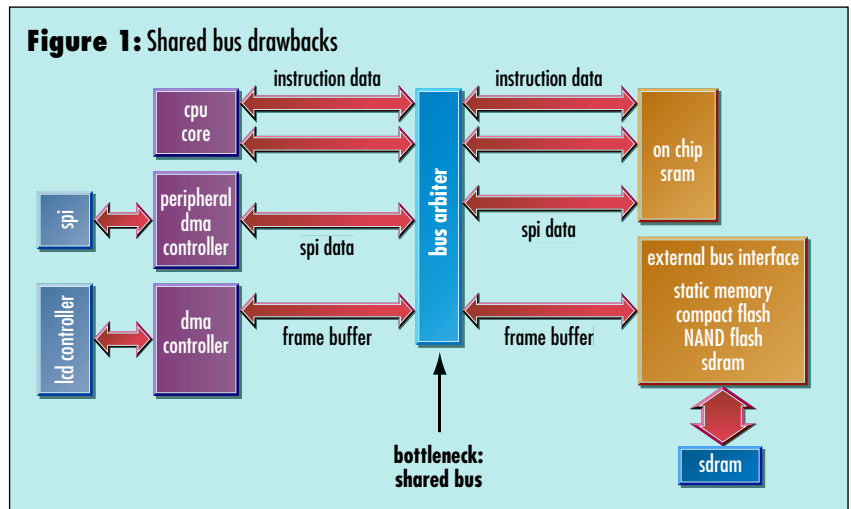
parameter involved in the microprocessor's evolution. "It's all about bandwidth," he claimed. "Communications with memory – both on and off chip – haven't kept up." The result, at least in Nativel's opinion, is a data bottleneck.

The problem becomes more complex when the microprocessor is used in graph-

ically intensive applications. He sees hand-held gps devices as a typical application for the modern microprocessor. "Here, the traditional approach is inefficient and we estimate that it could incur an 80 cycle penalty." The solution, says Nativel, is direct memory access, or DMA.

And that's the approach which Atmel has taken with one of its latest microcontroller products – the AT91SAM9263. And it has played its DMA hand in spades: the SAM9263 boasts 27 DMA channels, including an 18channel peripheral DMA controller (PDC), a nine layer bus matrix and two further buses for data and instruc-

**Figure 1: Shared bus drawbacks**





tion tightly coupled memories. The combination of these memory interfaces is said by Atmel to allow the SAM9263 to support on chip data communications rates in excess of 41Gbit/s. Meanwhile, the part's two external bus interfaces allows the use of multigigabit external memories.

Natvel outlined the conventional load/store architecture in a typical 32bit microcontroller. "The cpu manages all communication links. Data needs to be read from a peripheral to the cpu core, then written from the core to memory." And the same process works in reverse.

He claimed that a typical ARM9 based cpu running at 200MHz requires 80 cpu cycles to transfer 1byte of data between memory and peripheral. When the cpu is attempting to pass 20Mbit/s, even with the memory management unit (mmu) enabled, there is no processing power available for any other tasks. Worse, if the mmu is disabled, the controller is limited to handling around 4Mbit/s, which Atmel believes is insufficient to handle a high speed uart.

According to Natvel and his colleague Jacko Wilbrink, the use of DMA in such applications is a 'natural evolution' for embedded architectures where the number of on chip peripherals is growing rapidly, as is the amount of data to be handled. They say DMAs solve part of the problem by allowing direct peripheral to memory transfers. This approach uses only a fraction of the bandwidth required

by a processor centric architecture.

But the Atmel team points out a downside to DMA; it is intended primarily for memory to memory transfers. "This adds unnecessary software overhead and complexity to the system," they claimed. In their opinion, a better approach is to use an optimised peripheral between peripherals and memory.

And this is the PDC. This approach is said by Atmel to allow a data rate of 20Mbit/s to be achieved using the same 200MHz cpu, with 88% of the device's processing power still available.

The PDC integrates 18 single cycle peripheral DMA controllers, five DMA controllers with burst mode support to the USB host, and a range of other features. The DMA controllers offload completely the execution of data transfers.

At first sight, the use of these DMA controllers could be considered to require a large amount of silicon. But Atmel says peripheral DMAs need 90% less silicon than their memory to memory equivalents and this makes it a cost effective approach to implement dedicated DMA channels for each peripheral. "Moving the DMA channel configuration and control registers into the peripheral memory space," said Natvel and Wilbrink, "simplifies the peripheral drivers." The consequence is that application developers need only configure the destination buffer in memory and specify the number of transfers.

Another problem faced by those designing data intensive applications is on chip bandwidth. When multiple DMA controllers and the processor push large amounts of data across one bus, this can become overloaded and slow the system.

Natvel and Wilbrink say a 32bit bus running at 100MHz can handle 3.2Gbit/s. "Although this sounds a lot," they noted, "there may be so much data that the bus itself becomes a bottleneck."

The solution is multiple parallel on chip buses, along with a small amount of scratchpad memory – in this case 96kbyte provided in blocks of 80k and 16k.

The SAM9263 features 11 on chip buses; two providing tightly coupled memory running at 200Mbit/s and nine 32bit links capable of sustaining 100Mbit/s. Between these buses, the device can support on chip data transfers of 41.6Gbit/s.

## External bus

A further potential bottleneck can be encountered when an application shares external memory between the processor and peripherals. Here, the bus interface can limit data transfer.

Atmel has attempted to overcome this problem with the provision of two parallel external bus interfaces, both connected to the internal multilayer bus. One of the external interfaces is for system memory, the other supports either a high speed peripheral or a coprocessor.

Natvel and Wilbrink give an example. A 24bit colour vga panel requires a 900kbyte frame buffer, but an lcd controller with this amount of sram would be too expensive, so the buffer must be stored in external ram. A vga panel running in 24bit true colour mode would require 7.2Mbit per refresh, equating to 432Mbit/s. "A conventional 200MHz ARM9 processor cannot possible achieve this level of throughput," they contended.

A second external bus interface can be connected to an external memory which is used as a frame buffer. This second bus can eliminate the need for the lcd controller and cpu to share memory, increasing the amount of available processing power by up to 40%. 