



Facilitating the Migration from 8-bit to 32-bit Microcontrollers

By Jacko Wilbrink, ARM Product Manager, Atmel Rousset

Summary

The continuous increase in functionality integrated in products is raising the performance requirements and the program complexity of the microcontrollers. Barriers to higher performance microcontrollers have been the unit cost and the cost of migration. Unit cost is not only related to advanced manufacturing technologies reducing the cost impact for the extra processor bandwidth, but also a function of system integration. The cost to port the existing code to a new generation of higher performance microcontrollers cannot be eliminated but the transition can be facilitated in order to reduce the total effort required. This paper describes how Atmel's Smart ARM 32-bit RISC Microcontrollers address these points.

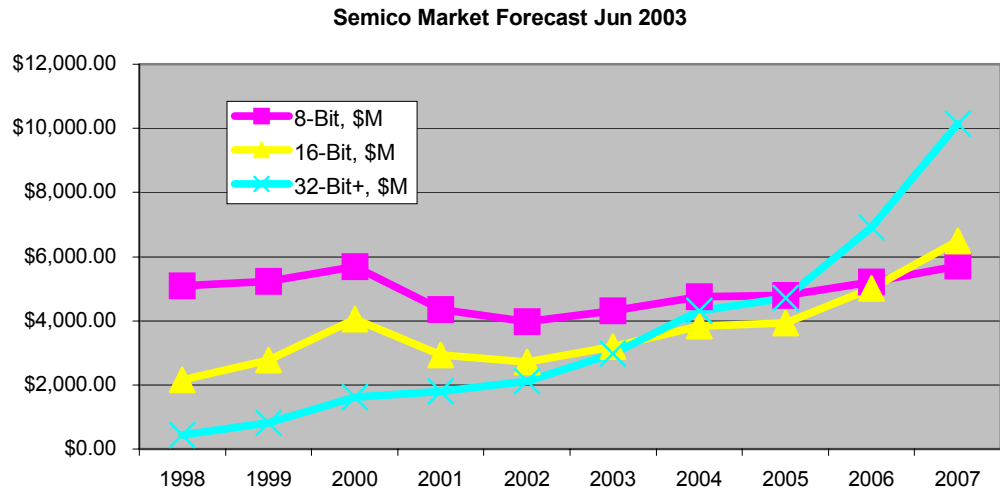
Table of Contents

<i>The move from 8-bit to 32-bit microcontrollers is in progress</i>	3
<i>Cost is a function of technology and integration levels</i>	3
<i>The meaning of performance</i>	5
<i>Reducing the complexity of the transition</i>	6
<i>Conclusion</i>	8

The move from 8-bit to 32-bit microcontrollers is in progress

The market research company, Semico, predicted in June 2003 that the total revenue of 32-bit microcontrollers will exceed the 8-bit by 2005/2006. (Figure 1)

Figure 1. Semico microcontroller market forecast, June 2003



Historically, when 8-bit microcontrollers reached their limit in performance and memory addressing capability, 16- or 32-bit microcontrollers came into play. Today, advanced manufacturing technologies (0.18µm and below) plus the reduced complexity of modern 32-bit RISC processors have lowered the cost of 32-bit microcontrollers at a point where they are competitive far down into the 8-bit market.

Cost is a function of technology and integration levels

A 32-bit RISC processor and a small Flash memory on a single chip is certainly not expensive using 0.18µm technology but it does not make it fit into an 8-bit application. Applications for 8-bit microcontrollers require a much higher level of system integration to meet the total system cost target. The device must operate from a single power supply, integrate high-current drive I/Os and transceivers, control the system power-up, generate the system clock from a low cost crystal and have programmable pull-ups on I/Os. Furthermore it is important to guarantee a predictable behaviour of the application at any time with a brown-out detector, a programmable watchdog running off an integrated RC oscillator and lock bits on the Flash.

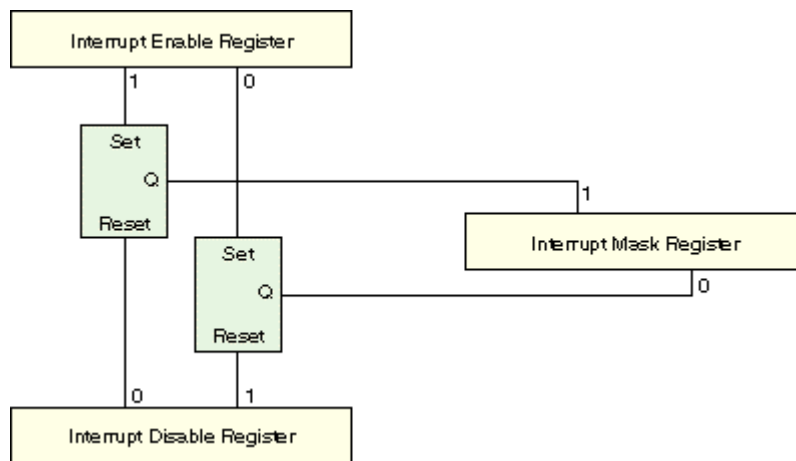
The meaning of performance

Applications using 8-bit microcontrollers are typically control oriented. This implies predictable response time to events. Cache memories used to accelerate the access to slow non-volatile storage media are not deterministic therefore not acceptable for time critical applications. Improving the access time of embedded Flash memories is the only way to improve real-time performance without penalizing cost. Market leading 0.18µm technology allows single cycle operation on an ARM microcontroller at up to 30MHz resulting in 27 MIPS of raw performance which is far more than an average 8-bit microcontroller today.

Beside deterministic processor performance, efficient interrupt handling and bit manipulation are important in 8-bit applications. The ARM processor has only two interrupt sources, FIQ and IRQ. Combining all interrupts on the low level IRQ and keeping the high priority FIQ for emergencies increases significantly the interrupt processing overhead. A vectored interrupt controller featuring an 8-level hardware priority scheme significantly reduces the software overhead and the response time to interrupts.

Bit set and reset operations cannot be done with a single ARM instruction. They require a read-modify-write sequence. In case of multiple tasks sharing the same peripheral, bit manipulations must be atomic (undividable). This implies masking interrupts during the read-modify-write operation. Two virtual registers, one to set (Enable) and the other to reset (Disable) bit(s) in a register allow bits to be modified by a single store instruction. (Figure 3) Since instructions are atomic on the ARM processor there is no need to mask interrupts anymore.

Figure 3. Hardware assisted bit manipulation



No H/W assisted bit manipulation

```
; Disable Interrupts (I & F) at processor level
```

```
MRS  r0, CPSR
```

```
ORR  r0, r0, #I_BIT:OR:F_BIT
```

```
MSR  CPSR_c, r0
```

```
; Read/Modify/Write the USART0 Interrupt Mask
```

```
LDR  r0, =US0_BASE
```

```
LDR  r1, [r0, #US_IMR]
```

```
ORR  r1, r1, #&80
```

```
STR  r1, [r0, #US_IMR]
```

```
; Re-enable Interrupts (I & F) at processor level
```

```
MRS  r0, CPSR
```

```
BIC  r0, r0, #I_BIT:OR:F_BIT
```

```
MSR  CPSR_c, r0
```

Hardware assisted bit manipulation

```
LDR  r0, =US0_BASE
```

```
MOV  r1, #&80
```

```
STR  r1, [r0, #US_IER]
```

Sustaining a constant data transfer rate on a peripheral requires a timely response from the processor. A peripheral DMA controller removes this constraint. Instead of the CPU transferring data one-by-one, the DMA transfers a block of data without CPU intervention. A single interrupt is generated at the end of the block transfer removing the need for peripheral polling. A dual pointer mechanism can automatically chain block transfers which avoids time critical pointer reconfiguration.

Reducing the complexity of the transition

In order to increase the acceptance of low cost 32-bit ARM microcontrollers it is mandatory to facilitate the porting of existing firmwares. Simplifying the device programming model optimised for C-language and offering powerful debug tools and support for 32-bit microcontroller on established 8-bit development tools are keys to shorten the learning curve.

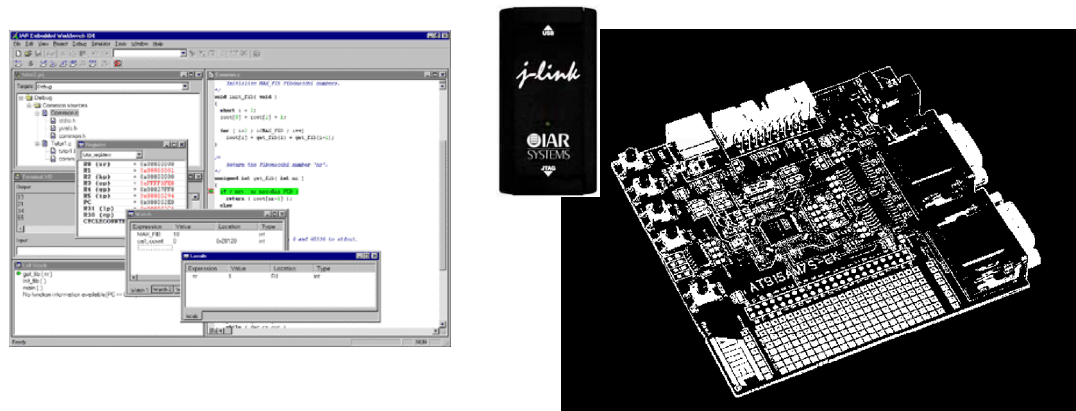
A comprehensive and consistent programming model for the integrated peripherals over the whole family of microcontrollers is the basis of simplification. A DMA controller dedicated to peripheral data transfers to and from memory tightly integrated in the peripheral memory space significantly reduces the software overhead. Providing project examples and peripheral drivers including source code will speed up the learning process and get the user quickly started. A large offering in operating systems and software IPs such as protocol stacks are available for the ARM7 processor. Instead of redeveloping industry standard protocols the customer should consider to acquire them. Finally a fast

growing network of ARM microcontroller consultants can help migrating customers to absorb the temporarily overload during the porting process.

The quality of the debug tools is critical to the software developer in order to speed up the porting process and validate the firmware. Full-blown emulators with trace capability have dropped significantly in market acceptance due to their cost and the delay between the microcontroller availability and its emulator. Embedded trace capability requires large number of pins to meet the required bandwidth, which is incompatible with low cost devices. The only alternative is to give up on device functionality during debug.

In-circuit-emulation interfaces with support for hardware breakpoints, provide complete access to the processor registers and internal memory space. They are also an interface to the debugger for software controlled trace. Today, they provide the best quality-to-cost debug solution. The use of instruction set simulators allow the programmer to raise the quality level of the firmware prior to debug on hardware, generally shortening the development time.

Figure 4. JTAG-ICE debug provides the best quality to cost debug solution today.



In addition to the microcontroller debug port, on-chip H/W can facilitate the development process. Misaligned data accesses are very difficult to trap unless an on-chip monitor identifies them and generates an abort to the processor. When the processor is stopped during debug mode, there is a high risk that a watchdog overflow occurs. This can be prevented if the watchdog is automatically stopped when the processor is in debug mode. Peripherals can generate invalid interrupt requests during debug too and should be filtered.

The best answer to the threat of the last minute bug is Flash memory for program storage. Flash memory offers the flexibility for software changes at the final stage of development without impacting the production ramp.

Conclusion

The migration from 8-bit to 32-bit microcontrollers is not just a question of device cost. The integration level reached on 8-bit microcontrollers during the last years must be matched, the real-time performance level increased and a range of memory size options provided. State-of-the-art manufacturing technologies allow 32-bit microcontrollers to meet 8-bit pricing for identical feature sets. Offering familiar development tools and advanced debug solutions facilitates migration to a 32-bit microcontroller. In addition, setting the preference to hardware instead of software implementation simplifies the device programming. Atmel's Smart ARM Microcontroller family is designed to meet the challenge.

Editor's Notes About Atmel Corporation

Founded in 1984, Atmel Corporation is headquartered in San Jose, California with manufacturing facilities in North America and Europe. Atmel designs, manufactures and markets worldwide, advanced logic, mixed-signal, nonvolatile memory and RF semiconductors. Atmel is also a leading provider of system-level integration semiconductor solutions using CMOS, BiCMOS, SiGe, and high-voltage BCDMOS process technologies.

Further information can be obtained from Atmel's Web site at www.atmel.com.

Contact: Jacko Wilbrink, ARM Product Manager, Atmel Rousset,
Tel: (+33) (0)4 42 53 60 24, e-mail: jacko.wilbrink@rfo.atmel.com