

Re: New ARM Cortex Microcontroller Product Family from STMicroelectronics

Source: <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2007-06/msg01128.html>

- *From:* wilco.dijkstra@xxxxxxxxxxxxx
 - *Date:* Sun, 24 Jun 2007 08:55:49 -0700
-

On 23 Jun, 03:10, rickman <gnu...@xxxxxxxxxx> wrote:

On Jun 22, 7:34 pm, "Wilco Dijkstra" <Wilco_dot_Dijks...@xxxxxxxxxxxxx> wrote:

"rickman" <gnu...@xxxxxxxxxx> wrote in
messagenews:1182540547.184518.91830@xx

The data sheet says it requires one wait state from 24 to 48 MHz and 2 wait states above 48 MHz. So compared to the Luminary parts running at 50 MHz with *NO* wait states, I say the ST M3 parts are dogs.

It's not that bad. Cortex-M3 has a prefetch buffer and branch prediction. This means that the cost of a single waitstate can be hidden for conditional branches, ie. only indirect branches have a penalty. With 2 wait states the branch prediction only works on unconditional branches, so you'll get a slowdown. However you can change loops to use an unconditional branch at the end so they run at the speed of zero-wait state memory.

I don't follow what you are saying at all. Branch prediction relates to pipelining. I don't see how it relates to wait states.

Adding a wait state is the same as increasing the pipeline depth, and branch prediction coupled with prefetching can hide some of that latency.

Re: New ARM Cortex Microcontroller Product Family from STMicroelectronics

The required wait states are added because of a fundamental limitation in the bandwidth of the Flash memory. You can look-ahead all you want, but you can still only return one word from Flash per 3 clock cycles when running at full speed. Unless the Flash word width is increased (as in the NXP designs) or the instruction size is reduced (many Cortex M3 instructions are 16 bits, but they would need to be 10 bits with two wait states and 32 bit memory) this will limit performance in the Cortex M3.

Am I completely missing something? I always leave that possibility open...

You have to increase the fetch width if you add waitstates, that's a given. The M3 TRM recommends 64-bit flash fetch. While this allows straightline code to run at full speed, branches are still slow. What I meant is that M3 has branch optimizations that reduce this slowdown.

Yes, that is all pretty obvious. But it does not address the point of the Y intercept being a hefty 9 mA. This is not as high as the Analog Devices ARM parts, but it is significant. It means you need to use modes and hardware features to get better power savings compared to just slowing the clock which is much simpler to do.

From what I remember, I think it is the PLL and flash that cause high power consumption at lower speeds. The specs showed various settings that use significantly less than 9mA below 8MHz. So I don't think it really burns 9mA at 0MHz (which is what I think you mean with Y intercept, right?).

The power for the STM32 is from the data sheet and includes basic power to the peripherals, although since they are not performing work the power they draw is less than typical. So the comparison is not perfect.

The ST specs list some numbers with peripherals off, and that is less than half the normal current, 21mA from flash at 72MHz IIRC – pretty good.

Now consider that an M3 runs twice as fast as a SAM7 at the same frequency,
so the MIPS/Watt is 4 times as good!

Re: New ARM Cortex Microcontroller Product Family from STMicroelectronics

How do you support the claim that the M3 runs twice as fast as the SAM7 at the same frequency??? Maybe I don't want to know...

Because I've benchmarked it myself?

I have not seen anyone claim that the M3 runs twice as fast as an ARM7 clock for clock. I don't even think ARM claims that. I seem to

ARM claims 70% improvement, but that is an understatement due to the Dhrystone benchmark. EEMBC is more accurate here.

recall that after all the hoopla is removed, you might see from 10% to 25% speedup from the ARM7 to the M3 depending on your application. If you disagree on this basic point, then I think we should not discuss it further. I have seen it discussed before ad nauseum with no hard information to support any given number.

You seem to forget that the M3 was designed from the ground up to be an efficient MCU, while ARM7 wasn't:

1. Thumb-2 is more efficient than Thumb-1 (just like ARM is faster than Thumb-1)
2. Cortex-M3 has a more efficient micro architecture than ARM7 (it beats ARM9)
3. Cortex-M3 slows down less with wait states than ARM7
4. New features like unaligned access, hardware divide, better interrupts

Both 1 and 2 provide about 40% improvement, so about 2 times speedup together. 3 can give somewhere between 10 and 20% extra performance, but only at higher frequencies when waitstates are used. Depending on the application used, 4 can make a huge difference (I've seen benchmarks go twice as fast just because of hardware divide).

I've personally benchmarked the effects of 1, 2 and 4 on a large amount of benchmarks. Now where did you get that 10-25% number from?

Actually, I guess a power factor would be required for the SAM7 parts as well since they run with one wait state at their top speed. So maybe the STM32 part do better on power than I realized!

Re: New ARM Cortex Microcontroller Product Family from STMicroelectronics

If you're trying to compare MIPS/Watt don't forget that different cores running at the same frequency do not run at the same speed.

Yes, but that is a small delta compared to adding waitstates with a 2x or 3x reduction in performance and therefore the same effect on power efficiency.

Not at all. Adding waitstates doesn't slow you down by that much as you make the memory wider (not doing that makes no sense at all, so I do not consider that a valid option). But instruction set and microarchitecture differences can easily make a factor of 2 difference, as shown above.

Wilco

.