GreenArrays, Incorporated

WHITE PAPER

Comparison of GreenArrays™ chips with Texas Instruments® MSP430F5xx as Micropower Controllers

Written by: Greg Bailey

GreenArray chips enable unprecedented control over energy consumption in both quantity (many small cores, each consuming power only when necessary) and in time (ability to stop and start consuming power in picoseconds at the level of a single core). The basic mechanism for communication within and outside the GreenArray chips naturally takes advantage of these capabilities to minimize not only instantaneous power, but its integral over time, which is energy.

The purpose of this paper is to compare the energy consumption of our chips with that of a chip family that is considered to be a micropowered controller suitable for use in energy harvesting systems. We conclude that our chips are superior, particularly in the simpler applications such as the creation of smart sensors.

Scope of Comparison

Texas Instruments' MSP430 family has achieved great success and wide acceptance since its introduction, and this excellent, versatile family enjoys a reputation as a "gold standard" for micropower applications. It is useful, then, to see how GreenArrays energy demands stack up in comparison.

All information about the MSP430 comes from TI publication SLAS655A – January 2010–Revised March 2010, in which the initial description states *"The Texas Instruments MSP430 family of ultralow-power microcontrollers consists of several devices featuring different sets of peripherals targeted for various applications. The architecture, combined with extensive low-power modes, is optimized to achieve extended battery life in portable measurement applications"* and *"Typical applications for this device include analog and digital sensor systems, digital motor control, remote controls, thermostats, digital timers, hand-held meters, etc."*

The MSP430 family certainly includes peripheral options that current Green Array chips do not, such as flash memories and timers. In this paper, we do not consider peripherals; we are instead concerned with the energy consumed in the simple process of behaving as an intelligent, responsive device.

GreenArray chips contain multiple, architecturally identical, independent computers or *nodes,* currently F18A or F18B types. The computers are self contained, each with its own memory.

Data shown for GreenArray chips are preliminary, based on testing of small numbers of parts from engineering prototype runs, and are subject to change when production quantities are available for more statistically complete characterization.

Static Power while Running

The referenced MSP430 data sheet only specifies "core" current at Vdd of 3.0v. The chip can run on 1.8v but we don't have any basis for current estimation at that voltage, so we shall use 3.0v for this chip.

The fairest system clock frequency to use appears to be 8 MHz, with PMMCOREVx=0, because the chip is less energy efficient at higher or lower frequencies (ranging from 1 to 25 MHz).

Therefore, the value we shall quote is 880 microamps core current at 8 MHz, or 2.64 milliwatts with 3.0v supply when running typical code from RAM. (We understand that most programs actually execute code from flash, more than doubling the power required.)

In contrast, a single F18 computer in one of our chips uses 2.5 milliamps at 1.8v supply, or 4.5 milliwatts, when running typical code from RAM at full speed (approaching 700 MIPS.)

Summary	Static pwr	MIPS
MSP430	2.64 mW	8
One F18	4.5 mW	700
F18/MSP	1.7x	87x

Energy per work done, while Running

The most efficient instructions of the MSP430 are register to register, and they execute in one clock cycle. The multiply peripheral can perform a 16x16 bit multiply in 3 clock cycles, with presumably two clock cycles used to store the arguments into registers and two clock cycles to retrieve the result, for a total of 7 clock cycles using arbitrary data.

For the MSP430, then, at 8 MHz a basic 16-bit register-to-register operation takes 125 nanoseconds and consumes 330 picojoules (pJ), calculated as 125 ns multiplied by 2.64 mW. A 16x16 multiply takes seven times as long (875 nanoseconds) and even assuming that the multiply unit does not use any extra power, should take at least seven times the energy, or 2310 pJ.

In an F18 computer, a basic 18-bit ALU instruction takes nominally 1.5 ns and consumes on the order of 7 pJ ($1.5x10^{-9}$ * $4.5x10^{-3}$). Multiply times vary, because multiplication is programmed and timing depends on coding, but a typical 17x17 bit fractional multiply routine takes on the order of 87 nanoseconds. Calling that 100 nanoseconds to encompass as many techniques as possible, we can say that a multiply in the F18 should consume less than 450 pJ.

There is always the danger of comparing apples with oranges, and coding techniques differ widely between the MSP430 and the F18. However, for similar, primitive activities, the above data indicate that the MSP430 consumes on the order of 47 times the energy used by an F18, while for complex operations like multiplication, the MSP430 uses at least five times the energy, and probably more if the energy consumed by the MPY32 module is known.

Summary	Simple op	Multiply
MSP430	330 pJ	2310 pJ
One F18	7 pJ	450 pJ
F18/MSP	1/47	1/5

Why does speed matter?

It is not unusual for a person to ask "why do I need all that speed? Wouldn't a slower computer use less power?"

This is not surprising given that so many manufacturers these days specify their devices' power requirements in milliwatts per megahertz. But that is *power*, not *energy*. If the computer is running at maximum speed all the time, then its energy requirements are a direct function of static power. If the computer is *not* running flat out all the time, then what matters is *how long it needs to run* to accomplish the work set before it, multiplied by the power used during that period of time.

This is why it is crucial to discuss the requirements of chips in units of energy, such as picojoules, required to accomplish a given task.

Taking a simple example that will be carried forward into the next section, let us assume that every so often a device we are considering building needs to respond to a stimulus by performing 100 simple operations.

On an MSP430, this would take 33,000 picojoules; on an F18, it would take 700 pJ. The elapsed time required for these operations would be 12,500 ns for the MSP430 and 150 ns for the F18; the F18 would take about 1/80 the time and use about 1/50 the energy.

Let us now suppose that this has to be done once per millisecond, or a thousand times per second, by either of these chips. What would happen if both chips had no choice but to run flat out the rest of the time?

If that were the case, the MSP430, being done with its work in 12.5 microseconds, would then be burning energy at the rate of 2.64 milliwatts for the remaining 987.50 microseconds until there was once again useful work to do. This would waste 2.61 microjoules, or 80 times as much energy as was expended in doing useful work.

The F18 would fare much worse, because it is capable of doing much more work during a millisecond owing to its higher speed. In its case, having spent 150 nanoseconds on useful work, the F18 would be running continuously and accomplishing nothing for 999.850 microseconds burning energy at the rate of 4.5 milliwatts, wasting 4.5 microjoules and thus about 6,500 times as much energy as was spent in doing useful work.

In this unfortunate case, both computers would be extremely inefficient in terms of wasted energy. Fortunately, however, both computers have hardware that allows them to reduce power consumption during the inactive parts of their duty cycles, thus reducing their wasted energy to a minimum.

Note that if this were done perfectly, then at the limit each computer would be incurring an energy cost exactly equivalent to the amount of energy used to do the useful work, or 33,000 pJ for the MSP430 versus 700 pJ for the F18, a factor of 47.

Summary	Duty cycle	Ideal energy
MSP430	1.25%	33000 pJ
One F18	0.015%	700 pJ
F18/MSP	1/83	1/47

Dynamics of average power, or continuous energy consumption

Although doing it perfectly is not feasible, we will see that the F18 comes much closer than does the MSP430.

In the case of the F18, going to “sleep”, which we refer to as *waiting*, happens automatically when a port or pin is read, or a port is written, and is not in the desired state. “Awakening” from this condition occurs when the state changes. How much energy do we tally against this cycle of waiting/running? It can be justifiably described as *zero* if the port or pin operation is necessary anyway to transfer data, but to be conservative let’s assume that the purpose of the read or write was simply to wait for an event and not to accomplish anything else useful. In that case the time spent getting into and out of the waiting state would be on the order of 4 ns and the energy consumed less than 18 picojoules. The F18 would spend the next 999,846 ns waiting for the 1 millisecond event, during which the F18 leakage is 55 nanoamps or about 100 nanowatts, so during the waiting period the F18 would consume 100 picojoules due to leakage.

Adding it all up, the F18 would consume 700 pJ running, 18 pJ changing state, and 100 pJ waiting for 818 pJ per

millisecond, an average power (rate of energy consumption) of 818 nanowatts.

Considering that the MSP430 consumes energy in its sleeping low power mode 4 at the rate of 3600 nanowatts, 36 times what an F18 consumes while waiting and 4.4 times what it takes for the F18 to be performing this whole application, it's easy to see where this is going, but let's finish the calculation to establish a correct way to make this comparison.

When the MSP430 is in its LPM4 sleeping state, it consumes energy at the rate of 3.6 microwatts. The data sheet tells us that it takes a minimum of 5 microseconds to wake up from this state and begin running again. Certainly energy is consumed in this process but the data sheet does not tell us how much. Another thing the data sheet does not tell us is how long it takes for the computer to go back to sleep nor what energy might be consumed in that process. As a conservative but reasonable guess, let us assume that power increases linearly from 3.6 microwatts to 2.64 milliwatts during the 5 microsecond wakeup process, and let us further assume that the process of going back to sleep takes no time and no energy.

On this basis, average power during the 5 microsecond wakeup would be $(2.64 \times 10^{-3} – 3.6 \times 10^{-6})/2$, or 1.32 milliwatts, and energy consumed during the wakeup process would be estimated at 6,590 picojoules.

Finally, having spent 12.5 microseconds working, 5 microseconds waking up and zero going back to sleep, the MSP430 would spend 982.5 microseconds sleeping at 3.6 microwatts for sleeping energy of 3,540 picojoules.

Adding it up as we did for the F18, during each millisecond the MSP430 would consume 33,000 pJ running, 6,590 pJ changing state, and 3,540 pJ sleeping, giving 43,130 pJ per millisecond or an average power of 43.130 microwatts.

Compared with the 818 nanowatt average of the F18, the core energy consumption of the MSP430 in this application would be more than 52 times that of the F18.

Summary	Duty cycle	Practical energy
MSP430	1.25%	43130 pJ
One F18	0.015%	818 pJ
F18/MSP	1/83	1/52.7

But there are more F18s in a GreenArray chip, aren't there?

Indeed there are. Each is burning energy at a rate of 100 nanowatts if it has nothing to do. So, for a GA4 chip with one node running the application above, average power would increase from 818 nanowatts to 1.118 microwatts. For a GA144 chip, the 143 idle computers consume energy at a rate of 14.3 microwatts, and with one node running the above application the average power would be 15.118 microwatts, just a little over 1/3 of what the MSP430 would be consuming.

Summary	Duty cycle	Practical energy
MSP430	1.25%	43130 pJ
GA4	0.00375%	1818 pJ
GA4/MSP	1/333	1/24
GA144	0.000104%	15118 pJ
GA144/MSP	1/12019	1/3

What happens at the other end of the scale?

An MSP430, at its maximum speed of 25 MHz, executes a basic register to register instruction from RAM in 40 ns while consuming energy at a rate of 13.5 milliwatts. A single F18 executes a simple instruction in about 1.5 ns, about 26 times as fast, while consuming 4.5 milliwatts, about 1/3 the power. The ratio of work done per unit energy, on the order of 77 to one in favor of the F18, is larger than the factor of 47 at 8

MHz because to access 25 MHz it is necessary to run the MSP430 in a less efficient power control mode.

Of course, this is not the true far end of the scale in terms of raw performance. A GA4 chip with four F18B computers can execute up to 112 times the instruction rate of the 25 MHz MSP430 while consuming only 1/3 more power. A GA144 can execute on the order of 3,800 times the instruction rate on less than 50 times the power.

Max speed 100% duty	MIPS	Power mW	Energy per op pJ
MSP430	25	13.5	540
F18A	666	4.5	7
F18A/MSP	26x	1/3	1/77
GA4	2,800	18	6.5
GA4/MSP	112x	1.3x	1/83
GA144	96,000	650	7
GA144/MSP	3800x	48x	1/77

In Summary

As stated at the beginning, there are apples and oranges in this comparison. However, it should be clear that if the MSP430 is characterized as a *micropower* controller, it is not unreasonable to apply the term *nanopower* controller to the GreenArrays chips The following table illustrates some key comparisons between the characteristics of the MSP430 and the GreenArrays GA4.

	TI MSP430F5xx	GA4
Low operating voltage	1.8V	1.8V
Standby Power	3.6 µW	0.4 µW
Energy per primitive operation	330 pJ	7 pJ
Energy per multiply	>2310 pJ*	<450 pJ
Maximum operations/second	8 MIPS in most efficient mode	2800 MIPS all four computers active
Maximum Active Power	2.64 mW running from RAM not flash	18 mW all four computers active
Minimum active period	>5000 nS	>5 nS
Energy to sleep/wake	6,590 pJ*	<18 pJ
Low input capacitance	5 pF	3 pF
Low input leakage	± 50 nA	<10 nA

Note: * indicates a value that was estimated based on available data.

Greg Bailey is President of GreenArrays, maker of multicore processors with integrated peripherals. For more information, please visit our website: http://www.GreenArrayChips.com