# THE FREESCALE SEMICONDUCTOR

# i.MX21 PROCESSOR

Standardized microprocessors sold by solid, reliable vendors have a lot to offer system designers. As The Gilder Report has noted, "a custom chip can forgo all power-using circuitry not needed for its one function. But a general purpose microprocessor is assured of 10 to 100 times the unit volumes. Thus, it can justify 10 to 100 times the design effort of a single-purpose custom chip. Microprocessor designers can spend effort on clock-gating, circuit-tuning, and power control that is way beyond what engineers can afford for custom hardware."[1]

Moreover, general purpose microprocessor vendors are world leaders in offering complete solutions, especially a plethora of development tools to enable system designers to get to market as quickly as possible with their complete hardware/software platform. There's no question that having robust compilers (and a selection of them), effective debuggers, and a choice of operating systems is still a huge strength of the merchant chip vendor, especially if those are tested to prove that they can work together.

No longer is it sufficient to simply provide a microprocessor (of course, the best companies knew this all along). Technology does not include just a processor core, but rather an entire program consisting of core, peripheral processing blocks, software to use those peripherals, firmware, a workable and usable software tool chain, and one or more operating systems.

***Highlights***

- Introduction to Hardware-Accelerated General Purpose Processors
- Performance and Power Benchmarking of i.MX21 with Comparisons
- i.MX21's crossbar technology provides excellent data movement performance
- MPEG4 acceleration really works
- Low power consumption and good performance make i.MX21 a "best buy" for mobile computing and communications applications

---

[1] The Gilder Technology Report, September 2004, Vol. IX, No. 9, page 3

Consistent profitability and progress in the embedded processor market means identifying real niches and filling those with the appropriate technology at the right time at the right price. Given the enormous number of mobile phones sold each year (estimated to be 600 million, with the fastest growing markets being China and Africa) and the dominance of the **ARM** (Advanced RISC Machines) architecture in the mobile phone and convergence-personal digital assistant (PDA) market segments, it makes a great deal of sense for companies in this market segment to license ARM and then seek to add value.

*Why is ARM a world leader in microprocessors for handheld devices? Common opinion holds that they were the first to offer very low power 32-bit processors specifically targeted at small computing devices. Others believe that it is because, once established, there was a critical mass of developed software (otherwise known as an "installed base"), and that system developers were loathe to change processor architectures. Perhaps it was ARM's early licensing terms, which lead to the creation of the intellectual property microprocessor vendor industry. Whatever the reason, it is clear that ARM, and ARM-variants, form the core of the most popular integrated mobile processors, including those from Intel, Texas Instruments, and Freescale Semiconductor. It may be true that in the future configurable and reconfigurable processors will become pervasive (along with an open source OS such as Linux that might have a chance of taking advantage of this sort of hardware), but for now ARM is the worldwide market leader. When Motorola (now Freescale) gave up its own M*core processor in favor of ARM, that confirmed this reality.*

One very good way to add value is to identify important new functionality that should be included as part of standard hardware, and we see this in many different market segments. For example, **Via Technologies** has introduced cryptography functionality into their general purpose x86 processors[2], while **Transmeta** has introduced hardware-level virus protection for the same x86 market[3]. One of the earliest adopters of this strategy has been Advanced Micro Devices (**AMD**), in the form of their Geode line of x86 processors with integrated 2D graphics.

We decided to look into wireless mobile communications and computing processors, and see if the inclusion of hardware acceleration blocks were worthwhile. Our primary focus is both performance and power consumption.

**Freescale Semiconductor** has chosen to focus on multimedia acceleration in the i.MX family line of processors, an acknowledgement that the coming 3G revolution in wireless cellular/mobile phones will mean audio and video streaming to handheld

[2] Fall Processor Forum 2004 disclosure from their respective companies
[3] ibid

devices. As we have noted, this is a modern strategy for dealing with the need to provide general control functions "with something extra", one that is being used more and more in the processor industry.

**The Freescale i.MX21** is based on an ARM926 core, and includes MPEG4 decode and encode acceleration in hardware, accessible to programmers by a software library. Freescale, recognizing the need for efficient performance and excellent power consumption characteristics, claims that they tried to focus on choosing the best performance per milliwatt envelope, which they believe will open up new applications and enable other handheld and portable devices, such as energy management systems and mobile gaming platforms.

A lot of consumer-level products are going to be built in the next three years that require general purpose processors "with something extra." That "special sauce" will often take the form of hardware accelerators for specific application functions, and Freescale's MPEG4 hardware block (for both encoding and decoding) moves them to the head of the class in terms of leading edge technology within the comfort of a general purpose, merchant market vendor. "Comfort", in this case, means all of the items we discussed previously: supported mainstream operating systems, high quality, well-tested C and C++ compilers, debuggers, interface tools, and so on.

Four major operating systems dominate the mobile computing and communication devices (MCCD) world: Windows CE from **Microsoft** (and its variants, Smartphone and PocketPC); **PalmOS** from palmOne; **Symbian32,** from Symbian; and **Embedded Linux**. Being able to support these operating systems, and offering a complete package to original device manufacturers (ODM's), is simply vital to be a player in today's market (which explains why a number of companies with excellent technology, high performance, and thrifty energy utilization haven't broken in to the mobile computing and communication device world). In short, if Microsoft, palmOne, Symbian, and some form of embedded Linux aren't supported, a chip vendor's market becomes quickly truncated.

Synchromesh Computing, a leader in benchmarking, benchmark certification, performance analysis, and embedded software tools analysis, decided to study the relative performance of three leading contenders for the emerging MCCD market. We selected two popular offerings (the Texas Instruments OMAP 1611 and the Intel PXA 255), and compare and contrast that against the Freescale Semiconductor i.MX21. (In a future Report, we will compare the new Intel "Bulverde" and the Texas Instruments OMAP 17xx against the next generation Freescale i.MX processor). We also benchmarked the i.MX21's hardware acceleration for MPEG4, and devised a scenario to investigate how much this hardware block offloaded the main ARM processor.

Since this market is driven by both performance and low power consumption, it was natural to focus on both of these and, in the interests of time, we selected Windows CE 4.x as a common operating system.

Because it's incredibly difficult to open up a mobile phone, or a PocketPC, and take meaningful measurements, we decided to use what are called "reference platforms" that have the processor, memory subsystem, and interfaces laid out on a nice big board. We tried to get the same hardware configurations and OS version between all three as similar as possible. This is not always easy: often a vendor will port Windows CE to a particular reference platform or development system, put it out into the market, and then go on to the Next Big Thing. Sometimes vendors of these systems get out of the business. We did our best to eliminate bias, and to find systems that are representative of clock speeds and memory configurations that are actually used by customers who buy processors.

As you can see by the table below, we had problems obtaining an "apples to apples" hardware environments, but we came close. First, clock speed: the processors obtained were in the clock speeds that the vendors actually offer. Second, bus speeds differ (and it was nearly impossible to ascertain the bus speed of the OMAP platform).

It is obvious that the Intel platform has a lot more work to do in terms of drawing objects on the screen than the other two systems, because its video subsystem is both standard VGA and 32-bit at 640 x 480. To be fair to Intel, it was therefore important to select benchmarks that were not graphically-intensive, and highlight the processor performance more than graphics subsystem (and yet, we could not completely ignore graphics because all three of these processors are built to use graphics in a finished system).

The operating system version issue

| Manufacturer | Freescale | Intel | Texas Instruments |
|---|---|---|---|
| **Processor** | i.MX21 | PXA 255 | OMAP 1611 |
| **Processor Clock Speed / Bus Speed** | 266 MHz / 133 MHz | 398 MHz / 100 MHz | 180 MHz / ? MHz |
| **RAM** | 31 MB | 31 MB | 13 MB |
| **Internal Data Cache** | 16K Sync (4-way, 32 byte line size) | 32K Sync (32-way, 32 byte line size) | 8K Sync (4-way, 32 byte line size) |
| **Internal Instruction Cache** | 16K Sync Write-Thru (4-way, 32 byte line size) | 32K Sync (32-way, 32 byte line size) | 16K Sync Write-Thru (4-way, 32 byte line size) |
| **Video System** | 240x320 16-bit | 640x480 32-bit | 240x320 16-bit |
| **Development Board** | Freescale ADS | Accelent | TI Innovator |
| **OS Name** | WinCE | WinCE | WinCE |
| **OS Version** | 4.2 | 4.1 (Build 908) | 4.2 (Build 1088) |

concerned us: we know, from past experience, that different OS versions can have dramatic effects on performance, and power consumption. We did the best we could given the nature of this study, but we did learn that device driver performance can have an even more profound effect on performance than the OS. We compensated for this as best we could.



The memory size is interesting: both Freescale and Intel reported 31MB, but T.I.'s platform reported on 13MB. Still, none of the benchmarks (along with the operating system) were paging, so we were relatively comfortable with this arrangement.

To make sure that the benchmarks we built from source code were as identical as possible, we used the Microsoft Embedded Visual C/C++ Version 4, with Service Packs 1, 2, and 3 installed, and we built the binaries using the ARM4i target on all processors.

Selection of benchmarks, which are software programs and workloads designed to help measure performance and other factors important in a design or buying decision. Notice the last clause: "in a buying decision." While there are certainly uses for benchmarking to help design better (faster, smaller, more efficient, thriftier) processors and systems, it is inescapable that benchmarks are used in sales and marketing. Doubtlessly this Report will be used by some in that manner. Acknowledging this is simply acknowledging reality, and Engineers are nothing if not realistic.

The question is, "could we find benchmarks that were fair to all, and would 'resonate' with the fast-moving, quick-adopting MCCD market?" Could we find or build benchmarks that would highlight performance similarities and differences?

We decided to use a mixture of popular benchmarks available from the Internet that are widely used by consumers and companies evaluating technology, and include some benchmarks that have very good reputations in computing circles. In doing so, though, we ran right into the "binary wall." Some of the popular benchmarks are not available in source code form, but rather are binaries that you take and run "wholesale", as is, on the system.

**BENCHMARKS USED**

- HINT
- STREAM
- MPEG4 Encode
- MPEG4 Decode
- SiSoft SANDRA[tm]
- BMQ
- VObench
- ByteMark

We found that, despite the "standard core architecture" for all three processors, some binaries ostensibly built for "ARM" in fact failed to run on some of the processors. The long ARM of binary applications compatibility did not extend to, for example, SiSoft's **SANDRA** for Windows CE: it would not run on the Intel PXA 255 system. The same was true for the popular **VObench** benchmark, yet the equally popular **BMQ** benchmark tests ran fine on all three platforms.

Even our own **SynchroBench**, quickly becoming a mainstay in our arsenal of benchmarks for all sorts of internet-focused platforms, failed to run on all three platforms, but this time it was due to incomplete Media Players (in the case of Texas Instruments, the Media Player was simply not built into the system image – a common situation in the highly-customizable Windows CE world). We decided to build command line benchmarks instead for MPEG4.

**VObench**, from Virtual Office Systems, is an immensely popular benchmark suite for PocketPC type devices, so much so that even PocketPC Magazine chose it as its own benchmark suite in a roundup of PocketPC platforms in July of 2002[4]. Its important to pick at least one or two popular benchmark suites in a general survey of performance so that your study is credible, some of the scores can be compared against other, previously published platforms, and so that a database of such scores can be built up in the industry. But relying upon these alone, in our judgment, is inappropriate because they all have defects in performance analysis. It's better to have a few of these and augment them with others, (perhaps less known but more reliable).

**BMQ** benchmarks CPU, memory, and graphics using five tests: Integer, Floating point, Drawing, Window operations, and Memory are measured. It is very popular in Japan, and comes in source code.

We had good success in building **HINT**, written by Ames Research Lab, Department of Defense. HINT has gained a reputation for being the most scalable and accurate measure of CPU and memory subsystem performance.

*"Most benchmarks measure either the number of operations that can be performed in a given time period, or the time required to perform a given fixed calculation.*

---

[4] http://www.pocketpcmag.com/_archives/jul02/BenchmarkingPocketPC2002.asp

*HINT does neither; rather, it performs a particular calculation (estimating upper and lower bounds for the definite integral of a monotone-decreasing function) with ever-increasing accuracy. The accuracy of the result at any given time is called the "Quality"; we may measure the improvement in quality at any given time as "Quality Improvements per Second," or QUIPS. As the computation progresses and the quality of the result improves, more memory and more operations are required to improve the answer .... Higher is better. HINT curves are a function of raw CPU processing power, L1 and L2 cache size and speed, and main-memory bandwidth." - Nicholas Coult, Ph.D., Assistant Professor of Mathematics, Augsburg College*

**STREAM**, written by Dr. John McAlpin of Silicon Graphics, is another open source, industry-standard benchmark suite that does an excellent job of measuring sustainable memory bandwidth. Embedded processors are almost always connected to the Internet, and almost always must process large amounts of data typical of multimedia bit streams.

*"The STREAM benchmark is a simple synthetic benchmark program that measures sustainable memory bandwidth (in MB/s) and the corresponding computation rate for simple vector kernels. Computer cpus are getting faster much more quickly than computer memory systems. As this progresses, more and more programs will be limited in performance by the memory bandwidth of the system, rather than by the computational performance of the cpu.*

*As an extreme example, several current high-end machines run simple arithmetic kernels for out-of-cache operands at 4-5% of their rated peak speeds --- that means that they are spending 95-96% of their time idle and waiting for cache misses to be satisfied.*

*The STREAM benchmark is specifically designed to work with datasets much larger than the available cache on any given system, so that the results are (presumably) more indicative of the performance of very large, vector style applications." - John D. McAlpin, creator, STREAM*

Remember **ByteMark**? Remember BYTE Magazine? The seminal journal for popular computing (even before PC Magazine), BYTE produced a benchmark suite that can be built from source code, and covers basic algorithms:

Numeric Sort
String Sort
Bitfield
FP Emulation
Fourier
Assignment
Idea
Huffman
Neural Net
LU Decomposition

ByteMark has the interesting summary metric of comparing processors to a fixed reference (in this case, an original model Pentium at 90 MHz!). But the main reason we picked Byte was that:

*"… the benchmarks are designed with a ``small-code, large-data'' structure. The executable code portion of each test occupies less than 16 KB of machine code. It can be argued that this gives only a partial picture of, say, cache performance, because an entire benchmark's code (and, in some cases, the associated data) will fit inside at least the secondary cache of modern processors.*

*Once again, in our defense, this is really the way that a good program should work. Code should exhibit locality--that is, the program should spend the majority of its time executing instructions that are close together (preferably following one another)* *and relatively little time jumping across large address distances."*[5]

Many embedded applications exhibit this: relatively small code size, but doing real work algorithms.

Our own version of **MPEG4 Encode** and **MPEG4 Decode** was built from open source code, and was built to run in a common, command line environment.

**HIGHLIGHTS**

- Synchromesh Computing uses industry-standard, open source benchmarks well-respected by experts, combined with recognized commercial benchmarks.
- We always recommend that companies join and participate in EEMBC.

A note about industry-standard benchmarks: Freescale Semiconductor is a member of EEMBC, the Embedded Microprocessor Benchmark Consortium, and plans on submitting the i.MX21 for benchmark Certification by the EEMBC Certification Laboratory (ECL) in the future. Indeed, EEMBC is also working on standardizing power consumption benchmarking, a tree that has taken over 3 years to grow but looks to be close to bearing fruit.

Power consumption is one of the next frontiers in benchmarking (see Synchromesh Computing's White Paper, "**Less Power, Engineer Scott!**"). Not surprisingly, we found curiosities in trying to measure power consumption on these three platforms. Not all vendors are especially eager to have customers do so, it seems, as they don't include documentation on where one can take power measurements. Going to the various vendor websites didn't prove very helpful, either. To their credit, Freescale's i.MX21 includes documentation on how to measure core and bus power, a brilliant move (if you're a potential customer evaluating processors and systems).

Fortunately, our Lab's engineers are up to the task: unlike many benchmarking organizations, we have hardware engineers on staff and, with the judicious use of probes, scopes, and intelligent detective work, we were able to obtain some power measurements without shorting our any of our boards. We chose to measure power directly using very precise power resisters, rather the more crude method of measuring battery life. We believe that these processors will be deployed in mobile phones, PDA's, and gaming systems using different battery technologies, some of which are evolving rapidly.

One surprise: how one builds Windows CE can determine how much power is consumed, and if the software engineers are not very careful, they can build an image that results in much greater power consumption that expected.

Power Consumption is tricky business for many reasons, and this is why it has taken EEMBC, the industry-standard

[5] BYTE's New Benchmarks, March 1995

benchmark group, literally years to develop a standardized methodology. Once this methodology is approved and the tools are in place, Synchromesh Computing will work closely with the EEMBC Certification Laboratory, LLC (ECL) to generate industry-standard power consumption figures for its clients. Until then, we believe that these measurements are fairly accurate. To get as fair a measurement as possible given the lack of documentation for the Texas Instruments and Intel platforms, we will focus on the core power of the part.

In the end, what we will present is both performance and power consumption benchmarking on MCCD devices and will provide some analysis as to what is going on for these platforms.

## THE PERFORMANCE AND POWER CONSUMPTION SCORES

CPU clock speed, bus clock speed, memory access clock speed can all have an effect on performance, but if the benchmark is processor bound, it is CPU clock speed that matters most. On the other hand, if the benchmark blasts out to main memory, bus speed and memory access speed is important.

We used the exact same compiler, and the same switch settings (ARM4) (same binary) to eliminate the effects of the compiler as much as possible, but we believe that optimized libraries would quickly jump many of these benchmarks upwards. This is especially true for ByteMark (where string and memory functions dominate).

The amount of memory (such as T.I.'s smaller memory size) prohibited some benchmarks from running, but since these are mostly single-threaded benchmarks run to completion, this is sort of binary situation: did it work, or not?

### Stream

Stream, a synthetic benchmark, consists of four sub-benchmarks (kernels): Copy, Scale, Add, and a combination of all three called, naturally enough, Triad. Measuring sustainable bandwidth in MB/sec. and the computational load, and its primary focus is on how well a processor integrates with its memory subsystem. Pure computational loads such as ADD and SCALE show the higher clock speed advantage of Intel, and clearly outline T.I.'s clock speed disadvantage compared to both Freescale and Intel. As the chart shows on the following page, though, the Copy benchmark test shows the i.MX21 showed amazing abilities in data movement and in memory-to-memory transfers. This is hard evidence that processor CPU clock speed is not the only architectural factor of importance in performance.

| | i.MX21 | PXA 255 | TI 1611 |
|---|---|---|---|
| Copy: | 152.381 | 47.761 | 40.506 |
| Scale: | 22.069 | 26.446 | 14.222 |
| Add: | 30.968 | 34.043 | 17.455 |
| Triad: | 16.842 | 20.601 | 10.573 |
| GEOMEAN | 36.392 | 30.679 | 18.057 |

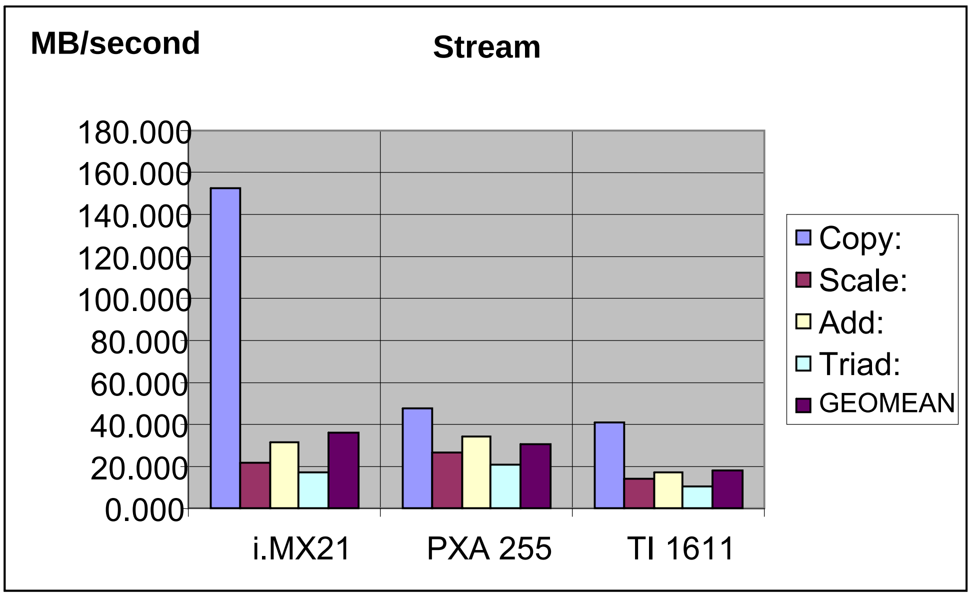
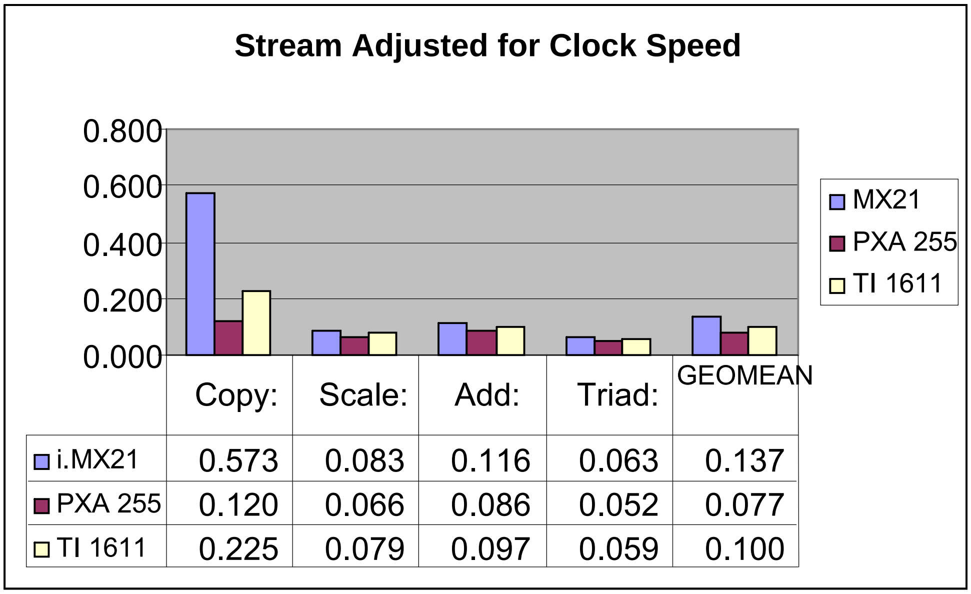



*Chart 1: Stream Benchmark Scores*

Adjusting the scores for the clock speeds of the various processors yields an interesting, though not surprising result:




| | Copy: | Scale: | Add: | Triad: | GEOMEAN |
|---|---|---|---|---|---|
| i.MX21 | 0.573 | 0.083 | 0.116 | 0.063 | 0.137 |
| PXA 255 | 0.120 | 0.066 | 0.086 | 0.052 | 0.077 |
| TI 1611 | 0.225 | 0.079 | 0.097 | 0.059 | 0.100 |

*Chart 2: Stream Factored by CPU Clock Speed*

**Factoring in power consumption for the processor core**, we find the higher clock speed of the Intel platforms tends to cause it to burn more milliwatts:

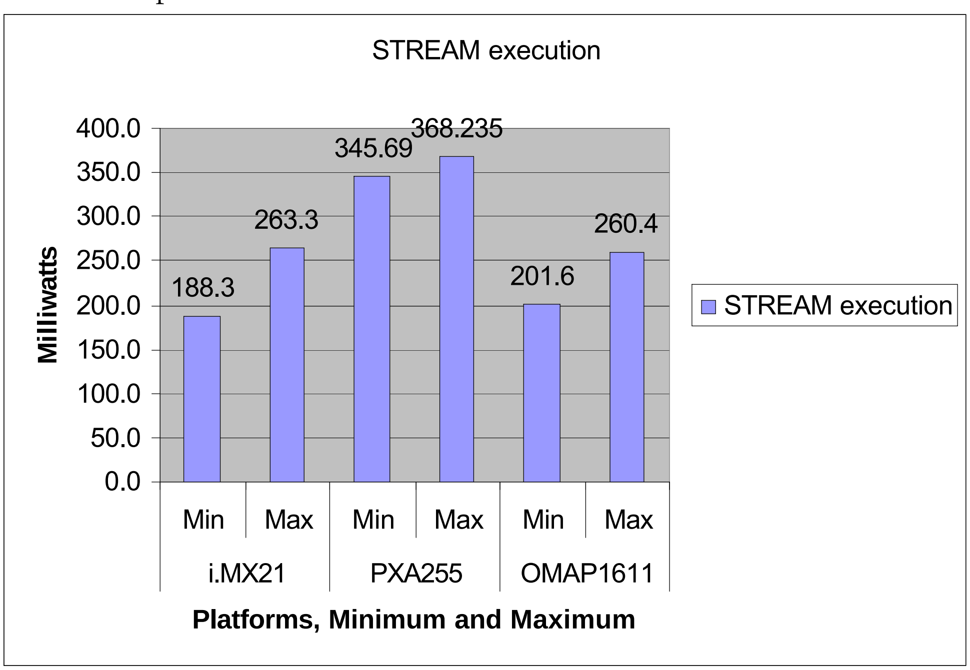



*Chart 3: Stream Power Consumption, Min and Max*

One interesting fact stands out right away: the i.MX21 has a higher difference between Minimum and Maximum Power utilization:

| | MX21 | PXA255 | OMAP1611 |
|---|---|---|---|
| Stream Max-Min Power | 75 | 22.545 | 58.8 |

But given that the execution of Stream for all of its tests, over a Geometric mean, yields this:

| | i.MX21 | PXA255 | OMAP1611 |
|---|---|---|---|
| Stream Average Milliwatts | 225.8 | 356.9625 | 231 |

It appears that the i.MX21 makes good use of its energy in generating the highest scores per milliwatt, at least on Stream:

| | i.MX21 | PXA255 | OMAP1611 |
|---|---|---|---|
| GeomeanPerformance/Milliwatts | 0.161169 | 0.085944 | 0.078169 |

This must be what Freescale apparently calls "Smart Speed" and it apparently shows that the i.MX21 is most efficient at using its energy on this benchmark. If your concern is absolute power consumption, clear the OMAP 1611 is best in this study, and Intel PXA 255 is the worst. But if efficiency at generating work throughput counts, i.MX21 wins. This is actually not surprising: the heritage of the Intel PXA 255 was the Intel (and before that, Digital Equipment) StrongARM, which was always known for much higher clock speeds than other, competing processors. Its design heritage was the same team that designed the DEC Alpha, another very high performance but very high clock speed processor.

| | i.MX21 | | PXA255 | | OMAP1611 | |
|---|---|---|---|---|---|---|
| **Idle at Windows CE Screen** | 105.3 | 108.1 | 106.713 | 109.719 | 110.88 | 114.24 |
| **Blank Screen** | 105.3 | 108.1 | 106.713 | 109.719 | 110.88 | 114.24 |
| **Screen Draws (fast peak)** | 105.3 | 231.3 | 106.713 | 396.792 | 201.6 | 228.48 |
| **Downloading files RS-232 (56K)** | 183.5 | 248.5 | 106.713 | 335.169 | 211.68 | 262.08 |
| **ByteMark execution** | 247.0 | 293.5 | 357.714 | 395.289 | 240.24 | 282.24 |
| **HINT execution** | 292.9 | 314.5 | 365.229 | 386.271 | 272.16 | 292.32 |
| **STREAM execution** | 188.3 | 263.3 | 345.69 | 368.235 | 201.6 | 260.4 |
| **BMQ execution** | 183.3 | 294.2 | 330.66 | 398.295 | 213.36 | 277.2 |

This table (above) shows all of the power consumption readings for three processors in the various benchmarks, except for the i.MX21 running its hardware accelerator on

MPEG4 decoding and encoding. We'll get to that in a little bit. Note that power does not scale in a linear fashion.

It turns out that this pattern is repeated rather consistently with HINT, ByteMark, Stream, and BMQ:

- The relative difference between MIN and MAX power consumption was greatest in the i.MX21, indicating its ability to throttle down when necessary
- However, on screen draws and downloading files over RS-232, the PXA 255 showed superior power management.
- The highest overall power utilization was Intel (also the highest relative performance)
- The best performance per milliwatt was i.MX21
- The lowest power consumption was by the OMAP 1611 (due to lower core frequency)

We really are interested in how the MPEG4 acceleration block affected power consumption, and we'll cover that in the section on MPEG4. For now, we again stress the importance of measuring power consumption using the real application (if possible, though it rarely is), or looking into the upcoming EEMBC Power Consumption Benchmarks. Until then, we believe this gives a very good assessment of how these three processors fare.

## MPEG4 DECODE and ENCODE

Synchromesh Computing experimented to see what the performance might be, "out of the box", without any acceleration at all on the i.MX21, and then what the impact was of MPEG4 acceleration on decoding and encoding.

MPEG4, at its heart, is similar to MPEG2 in using a discrete cosign transform (DCT) for encoding. This is also called "quantization", which is a fancy name for "shrinking something down while maintaining its essential characteristics, but in an encoded, compressed fashion." The reverse, reserved for MPEG decoding, is naturally the inverse discrete cosign transform, or iDCT. It turns out that encoding is a much more computationally intensive activity, involving significant amount of mathematics. Smaller caches will tend to hurt MPEG4 encoding worse than decoding, since once a frame is decoded it is essentially emitted (either to the graphics processor for display, or to memory for further processing). MPEG4 encodings must also deal with motion estimation, motion compensation, and other numerically-intensive math functions. But things get complicated: if you have a hardware acceleration block that can do some of the work, how you apportion out the workload between the main ARM processor core and the accelerator can determine total runtime. Superscalar architectures (which none of these three are) have distinct advantages, as do digital signal processors.

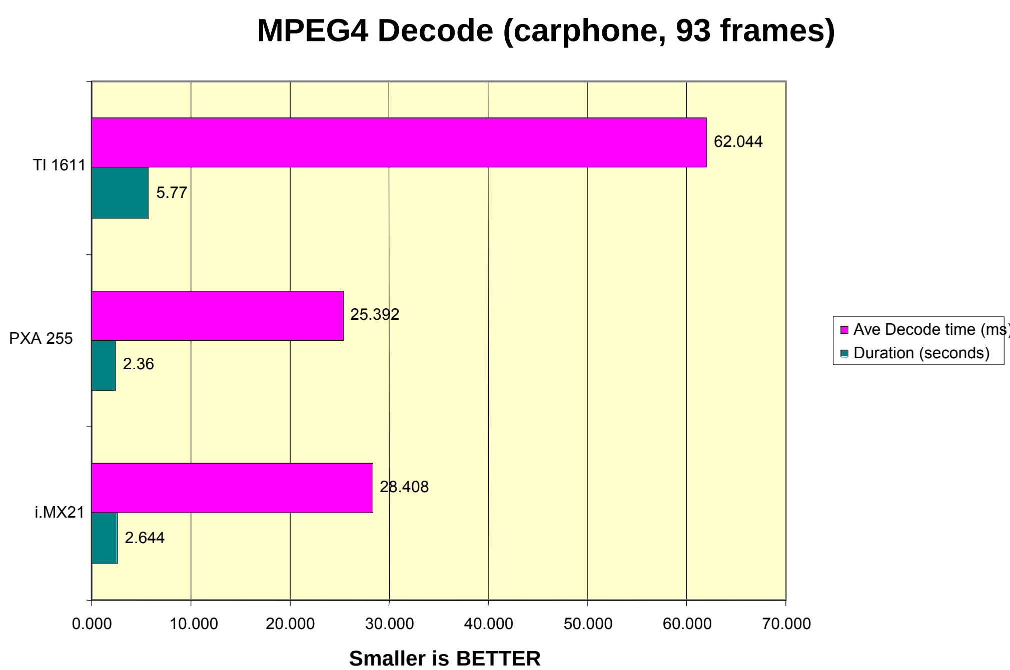



Chart 10: MPEG4 Decode. Small L1 cache and slow bus speed hurts TI

As we can see, the i.MX21's strong data movement abilities shine through, and despite its much lower clock speed (266 MHz to the PXA's 398 MHz), it performs nearly as well. The OMAP 1611 suffers from an 8K data cache, and combined with a probable slow memory bus, really overwhelms its ability to keep up.

On MPEG4 encode we find a similar situation:

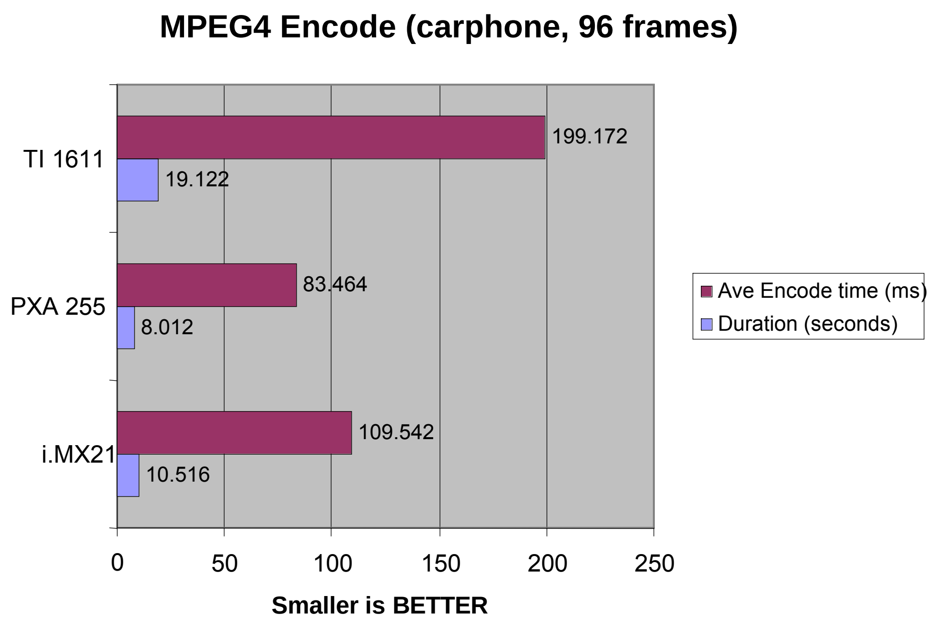



Chart 11: MPEG4 Encode. Considering its clock speed, i.MX21 performs well

## MPEG4 DECODE and ENCODE using i.MX21 MPEG4 Hardware Acceleration

What happens when we benchmark the i.MX21 using the hardware acceleration block? While we don't expect single cycle decode or encode, we do expect huge improvements in performance, justifying a price premium and/or increased attractiveness for the processor for handheld encoding and decoding applications. Rather than do a shootout between i.MX21, OMAP1611, and PXA 255, we decided to just highlight the performance improvements in using the hardware assisted MPEG4 block (to be fair to Texas Instruments, for example, we would have to code the MPEG4 application in DSP code, a task outside the scope of this report).

First, a review of the MPEG4 un-optimized scores on all three platforms:

| Un-optimized MP4 DECODE | | i.MX21 | PXA 255 | TI 1611 |
|---|---|---|---|---|
| 93 Frames | Duration (seconds) | 2.644 | 2.36 | 5.77 |
| | Ave Decode time (ms) | 28.408 | 25.392 | 62.044 |
| Unoptimized MP4 Encode | | i.MX21 | PXA 255 | TI 1611 |
| 96 Frames | Duration (seconds) | 10.516 | 8.012 | 19.122 |
| | Ave Encode time (ms) | 109.542 | 83.464 | 199.172 |

Now let's factor in hardware assist on the i.MX21 (lower is better):

| | | HW Assist | Un-optimized |
|---|---|---|---|
| Hardware Assist MP4 DECODE | Duration (seconds) | 0.5 | 2.644 |
| 94 frames | Ave Decode time (ms) | 5.3 | 28.408 |
| | | HW Assist | Un-optimized |
| Hardware Assist MP4 Encode | Duration (seconds) | 1.18 | 10.516 |
| | Ave Encode time (ms) | 12.6 | 109.542 |

As you can see, there is a huge performance advantage gained by having hardware assisted MPEG4 encoding and decoding. Our strong belief is that the i.MX21's hardware assist unit will indeed offload a substantial amount of processing load from the main CPU.

We also wanted to explore the nature of the power consumption of this hardware block:

| | Measured Voltage | | Computed Current (mA) | | Computed Core Power (mWatt) | |
|---|---|---|---|---|---|---|
| | Min | Max | Min | Max | Min | Max |
| Idle (for Comparison) | 15.1 | 15.6 | 68.6363636 | 70.9090909 | 113.25 | 117 |
| MPEG4 Encode | 45.8 | 46.8 | 208.181818 | 212.727273 | 343.5 | 351 |
| MPEG4 Decode | 41.2 | 43.2 | 187.272727 | 196.363636 | 309 | 324 |
| Software MPEG4 encode/decode | 44.4 | 46.5 | 201.818182 | 211.363636 | 333 | 348.75 |

The hardware-assist actually reduces to total power consumption for the processor, since the main core is not working as hard.

## HINT

HINT, a synthetic benchmark, shows pure computational processing ability. HINT is fairly balanced, doing a lot of computation and then memory movement, and despite i.MX21's better data movement, the PXA's cache (double i.MX21's) started to take its toll. We do not believe that the "ways" (32 ways to i.MX21's 4-way cache) made a difference. The Intel platforms both show huge dropoffs once you're out of cache - indicating wait states and poorer memory latency performance (to verify this, we also arranged to test a Compaq iPAQ with an Intel PXA processor). Both Freescale and Texas Instruments show lesser dropoffs, reflecting a smaller proportional ratio between processor CPU speed and bus speeds.

**HINT** (http://hint.byu.edu/) has a large database on the Internet of other benchmark scores from various systems going back to the early days of computing, and is a benchmark recognized for accuracy and fairness. A movie that explains how HINT works is here: http://hint.byu.edu/tutorials/hint.mpeg

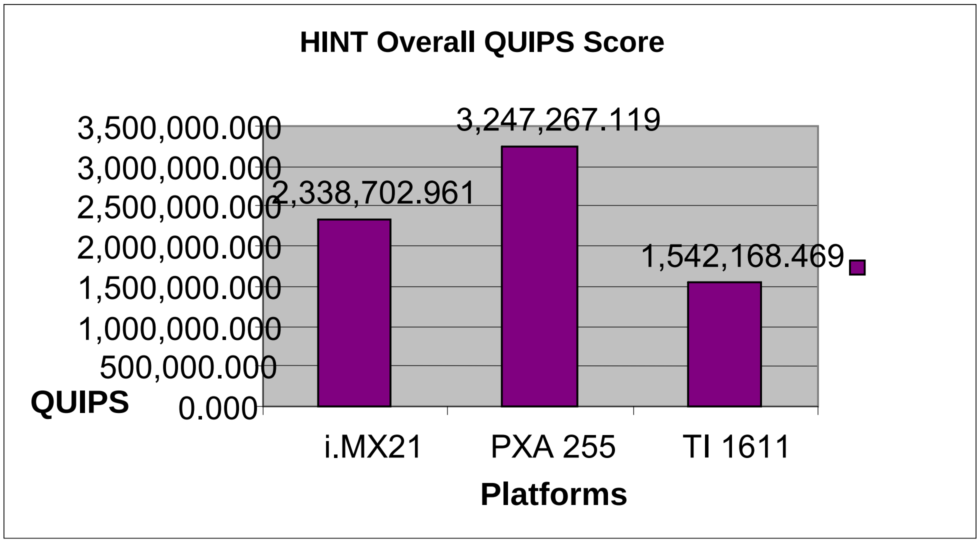



*Chart 4: HINT Overall Quality Improvements per Second*

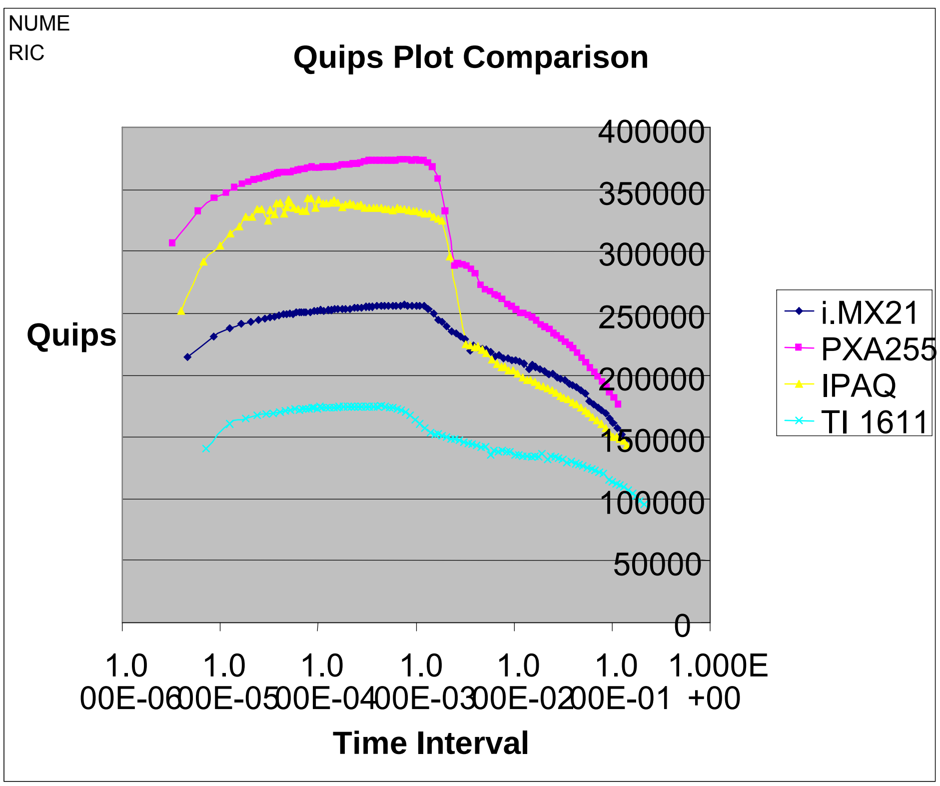



*Chart 5: HINT Quality Improvements per Second Plots Over Time*

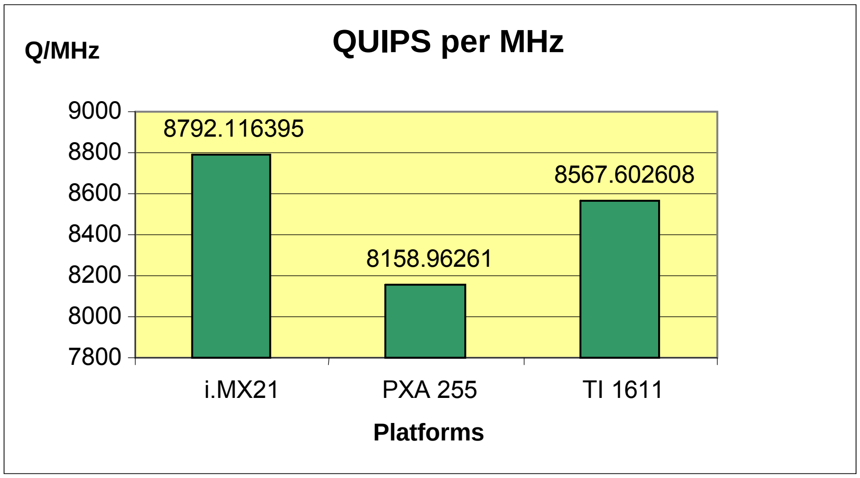


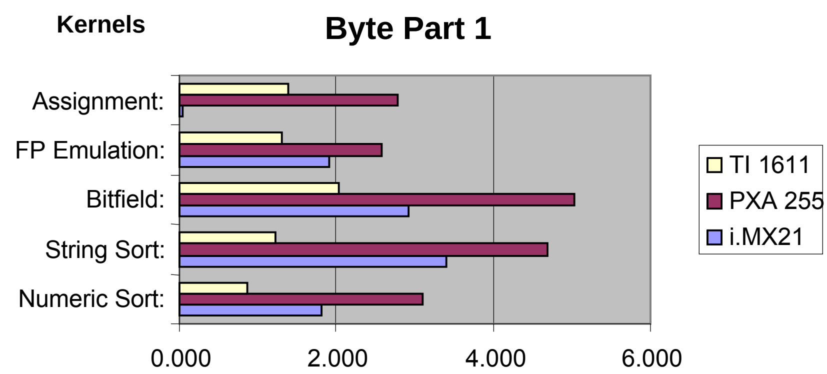

Chart 6: HINT QUIPs per MHz: the "Smart" in "Smart Speed"

The interesting thing here is that, adjusted for clock speed of the CPU, we find that i.MX21 actually compares quite well (although note that the differences are slight - this chart exaggerates the magnitude of difference a bit). Remember the TI 1611 COPY Stream score? Well, their memory subsystem apparently is better than the PXA 255's.

## ByteMark

Although certainly better than Dhrystone because it uses actual algorithm kernels, ByteMark unfortunately tracks two things: CPU clock speed (integer and floating point), and how well a C compiler has implemented its string and memory library functions. None of these companies has special, highly optimized libraries, however, that could be linked in to produce a more efficient binary.

So why do we like Byte? For one thing, as long as libraries are not optimized, it gives a fairly decent representation of a set of common algorithms (though EEMBC is much better), and it too has a vast history on the Internet of existing benchmark scores). As we said before, we like that it's written in ANSI C, that it exhibits "small code, real algorithms" characteristics, and that it's portable. It is not a replacement for EEMBC, but it does give a very good idea of simple workloads and is portable to Linux, Windows CE, and practically any other operating system environment.




| | Numeric Sort: | String Sort: | Bitfield: | FP Emul. | Assign ment: |
|---|---|---|---|---|---|
| TI 1611 | 0.867 | 1.228 | 2.024 | 1.318 | 1.384 |
| PXA 255 | 3.104 | 4.697 | 5.027 | 2.576 | 2.770 |
| i.MX21 | 1.805 | 3.399 | 2.927 | 1.903 | 0.040 |

**Index Scores**

Chart 7: ByteMark. Intel's Bitfield and String Sort Show Strongly

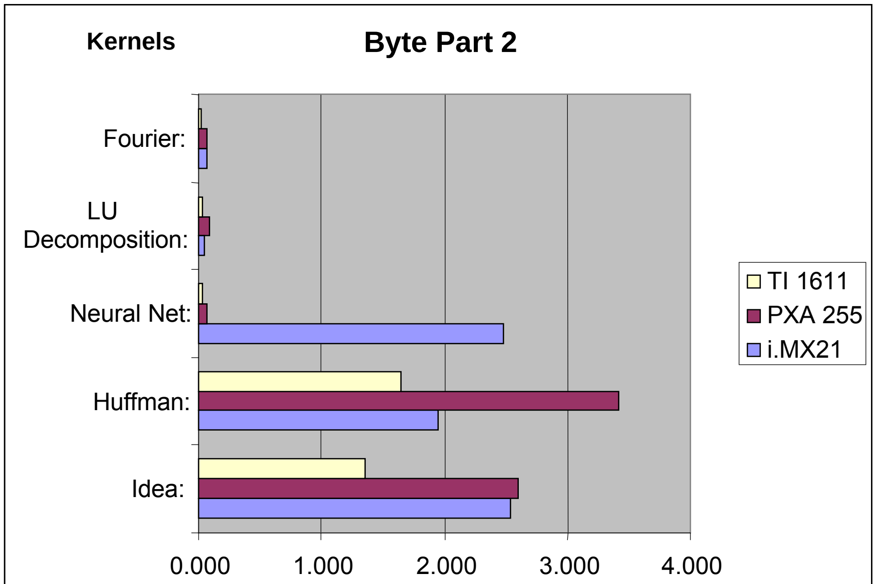



Chart 8: ByteMarks. i.MX21 Blows Away Idea and Neural Network Processing

| | i.MX21 | PXA 255 | TI 1611 |
|---|---|---|---|
| Idea: | 2.543 | 2.600 | 1.355 |
| Huffman: | 1.946 | 3.415 | 1.648 |
| Neural Net: | 2.482 | 0.073 | 0.030 |
| LU Decomposition: | 0.048 | 0.092 | 0.038 |
| Fourier: | 0.065 | 0.072 | 0.024 |
| | | | |

We're impressed by the second half of Byte for the i.MX21, especially its performance on Neural Net processing (a back-propagation network system), and its strong showing on Idea, a block cipher algorithm. Idea moves through data in 16-bit chunks, and we believe again shows the i.MX21's good data movement characteristics. Neural Net moves through very small arrays, and is heavily dependent on the exponential function and, to some extent, the floating point performance. To check on this, lets go ahead and summarize the Integer and Floating Point scores (remember, a 1.0 equals a Pentium original generation at 90 MHz):

| | i.MX21 | PXA 255 | TI 1611 |
|---|---|---|---|
| Integer Index | 2.370 | 3.341 | 1.364 |
| Floating Point Index | 0.050 | 0.079 | 0.030 |

To factor it by clock speeds to get some idea of performance per megahertz and to focus on just integer (since all three processors use software floating point emulation anyway):

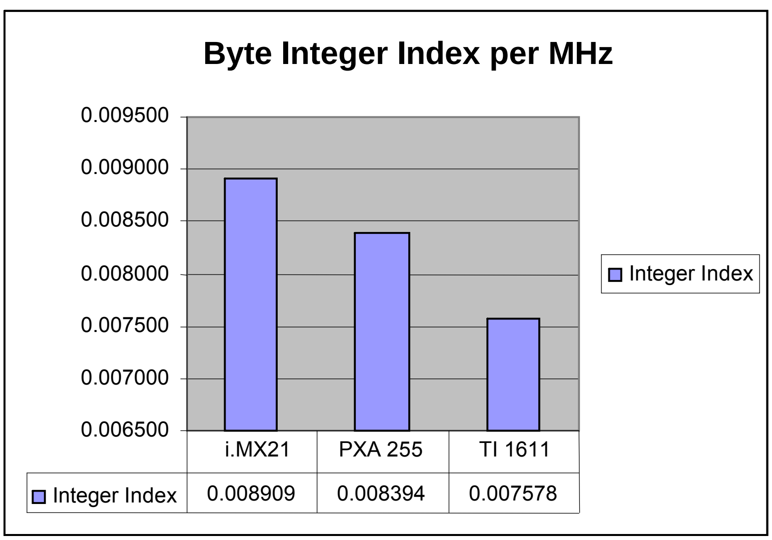



Chart 8: ByteMarks. Per MHz, Integer Summary Score

## BMQ

This series of tests benchmarks CPU, memory, and graphics using five tests: Integer, Floating point, Drawing, Window operations, and Memory are measured. It is very popular in Japan for Windows CE platforms, and comes in source code. While it does use extremely simple arithmetic, and is really no better than Dhrystone or other simple loop synthetic benchmarks, it does includes some Windows CE "create windows and display them" and "draw on the screen" tests.

What this showed is that the PXA 255 system is suffering from a terrible video device driver. A Window operation scores of "11" is 20 times slower than our research data shows (a score of about 220 is more normal for a 400 MHz Xscale type processor).

Given what we learned, it's important to always analyze benchmark scores with the underlying software in mind, and to not place more trust in them than warranted (nor less than should be applied, for that matter).

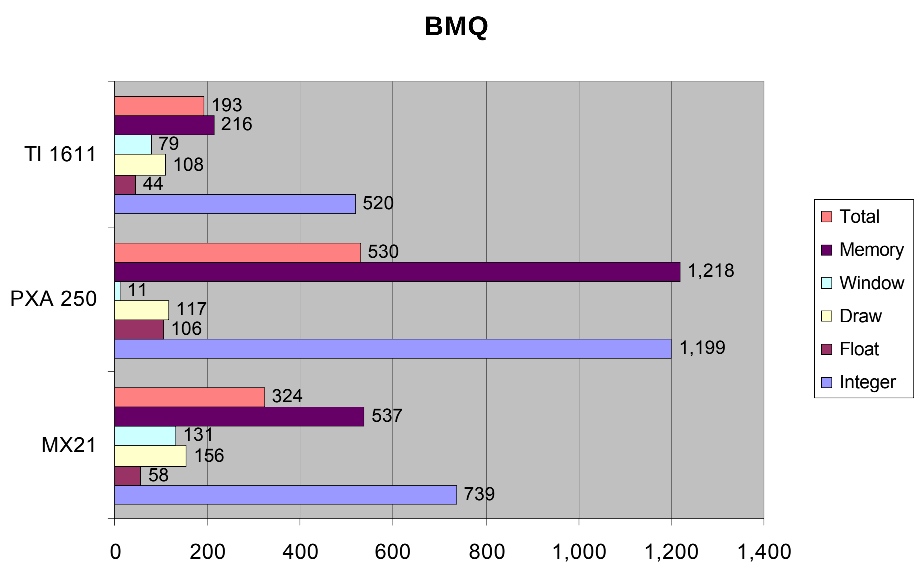



Chart 9: BMQ: Bad Device Drivers Stall Intel on Window operations

| | i.MX21 | PXA 255 | TI 1611 |
|---|---|---|---|
| Integer | 739 | 1,199 | 520 |
| Float | 58 | 106 | 44 |
| Draw | 156 | 117 | 108 |
| Window | 131 | 11 | 79 |
| Memory | 537 | 1,218 | 216 |
| Total | 324 | 530 | 193 |

Conclusions are fairly easy to reach given what we now know:

- Clock speed often hurts "efficiency", as measured by performance per milliwatt. It's important to make sure data movement is good in your processor, and the i.MX21 crossbar switch does its job to keep the processor core fed.
- Small L1 and L2 caches unquestionably hurt performance on larger applications (and benchmarks!).
- MPEG4 hardware-assist functionality on the i.MX21 proved its worth, and dramatically reduced the time it took to encode YUV files into MPEG files. The strategy of using hardware acceleration to offload the CPU is sound, delivering measurable benefits to customers.
- Optimizing the entire system (processor core/CPU, memory subsystem, bus speed, data movement, software such as device drivers, operating system, and libraries, and the board itself all contributes to performance. As we've seen, ignoring or failing to achieve on any of these can seriously penalize a company's efforts.
- The i.MX21, per milliwatt and per clock speed, lives up to the Freescale "Smart Speed" marketing campaign. In fact, we believe can say it performed better than TI and, "pound for pound", better than the Intel processor.
- The Intel PXA 255 has good raw performance based on its higher CPU and bus clock speeds and big caches. We look forward to testing "Bulverde", the code name for the follow-on processor (and the follow-on to the i.MX21, the next generation i.MX processor).
- Windows CE is sensitive to good device drivers. This is not a reflection of the processor per se, but it is a reflection of the expected performance of a CE-based system if a design house doesn't also invest in software personnel.
- Beware of binary-only benchmarks, as you never really know how they were built – or even if they will run properly.
- The EEMBC scores, especially for Consumer Version 2 (DENmark), should be very interesting as they will have MPEG2, MPEG4, and MP3.

# Synchromesh Computing
Voice: 512-219-0302
Fax: 512-219-0402
Email: inquiry@synchromeshcomputing.com