EVALUATING THE LATEST DSPS FOR COMMUNICATIONS INFRASTRUCTURE APPLICATIONS

Kenton Williston (Berkeley Design Technology, Inc., Berkeley, CA, USA; williston@bdti.com)
and Jeff Bier (Berkeley Design Technology, Inc., Berkeley, CA, USA; bier@bdti.com)

ABSTRACT

Modern communications infrastructure applications place heavy signal processing demands on their processors. Choosing the right processor or core is critical, and can make the difference between the product's success and failure. Unfortunately, it can be difficult to assess which processor provides the best solution in terms of speed, ease of use, cost, energy efficiency, and other considerations. The decision process is complicated by the expanding array of architecture choices. Sorting through the options can be a full-time job.

In this paper, we employ analysis developed using our well-known DSP benchmark methodology to compare the performance of several processors targeting communications infrastructure products: Texas Instruments' TMS320C64xx; StarCore's SC140 core; Analog Devices' TigerSHARC; and LSI Logic's ZSP family of cores. We assess the aptitude of each architecture for communications infrastructure products, and identify key strengths and weaknesses. We also discuss the quality of each processor's development tools, the availability of off-the-shelf software, and other factors that can affect system development costs and time-to-market.

1. INTRODUCTION: APPLICATION NEEDS

1.1. Quantitative Considerations

Some digital communications systems, such as fax machines, use a point-to-point topology. The most commercially important systems, however, use two classes of transceivers: terminals (e.g., mobile phones) and infrastructure (e.g., cellular base stations). Terminal and infrastructure equipment typically perform many of the same tasks—and may even use the same basic algorithms—but these two classes of equipment have vastly different requirements. Terminals typically handle one communication channel and are severely constrained by cost, power, and/or size. Hence, designers of terminal equipment typically use the smallest, least expensive, and most energy-efficient chip that meets the application's processing requirements. Minimum memory use is another key criteria, as this leads to lower cost and power consumption. Terminal equipment designers also seek to minimize chip counts (and hence cost and power consumption) by using system-on-chip (SoC) designs to maximize system integration.

In contrast, infrastructure equipment typically handles many channels. For infrastructure equipment, density is at a premium; designers strive to minimize cost per channel, power per channel, and board area per channel. Infrastructure equipment designers are also often concerned with flexibility; infrastructure equipment often must comply with multiple standards and/or with standards that evolve over time. Unlike terminal equipment, infrastructure equipment is often designed with significant processing headroom. This is done to allow for the additional processing requirements of new features that will be added after the equipment is in the field.

1.2. Qualitative Considerations

DSP applications are complex and resource-hungry, and software often requires optimization to meet hard real-time constraints. Numeric fidelity is also an important consideration, especially in applications that involve bit-exact standards. These factors make good debugging and profiling tools essential to DSP software development.

DSP software is becoming larger and more complex, making the issues of maintainability, productivity, and portability increasingly important. Thus, system designers must consider the quality of compilers and supporting software such as libraries and operating systems.

In many applications, system developers must consider the quality of tools both for DSP-oriented tasks, such as audio processing, and for general embedded computing tasks, such as operating systems, network stacks, and device drivers. General-purpose applications are often written in C code that assumes the processor has a native data size of 32 bits and a large address space (due to their large size). However, many DSP processors, particularly low-cost designs, have a 16-bit native data size and a small address space. In addition, compilers for DSP processors are often poorly suited to general-purpose applications. Even DSPs that feature 32-bit data paths and large address spaces usually offer only a limited selection of off-the-shelf general embedded computing software components.

Terminal equipment designers, who face intense time-to-market pressures, are often concerned with chip packaging and chip-product roadmaps. Infrastructure equipment designers, who face long development cycles and service lifetimes, are often more concerned with the vendor's architecture roadmap.

2. BENCHMARK EVALUATION

DSP processor performance can be measured in many ways, but making fair and meaningful comparisons is difficult. In an effort to provide meaningful performance comparisons, BDTI, an independent DSP analysis and software development company, developed its own suite of DSP benchmarks. BDTI's benchmarks are based on DSP algorithm kernels, which are the most computationally intensive portions of DSP applications. Because typical DSP applications spend the vast majority of their processing time in these kernels, application-relevant algorithm kernels are strong predictors of overall performance. Example algorithm kernels relevant to communications applications include FFTs, FIR filters, and Viterbi decoders.

Throughout this paper, we present processor performance using a "radar graph." This graph shows four performance metrics, with bigger scores representing better performance in all cases:

- **Speed**, represented by the cheetah icon, is a processor's BDTImark2000™ score. The BDTImark2000 is a summary measure of DSP speed distilled from the BDTI Benchmarks™.
- **Memory efficiency**, represented by the diskette icon, is also distilled from the BDTI Benchmarks. This metric is the inverse of memory use; larger values represent lower memory use.
- **Affordability**, represented by the money icon, is the inverse of the price for a 10,000-unit order.
- **Energy efficiency**, represented by the battery icon, is calculated by dividing a processor's power consumption by its BDTImark2000 score.

Each axis is normalized relative to the processor with the best performance on that axis. For example, the speeds are normalized such that the fastest processor has a relative speed of 1.0. All four metrics are shown using a linear scale.

Figure 1 shows the performance of Texas Instruments' 120 MHz TMS320C5409, a mid-range member of the TMS320C54xx family. All later radar charts show this processor as a reference case. The TMS320C5409 is used as a reference because the TMS320C54xx family is well known and is by far the most popular family of DSPs.

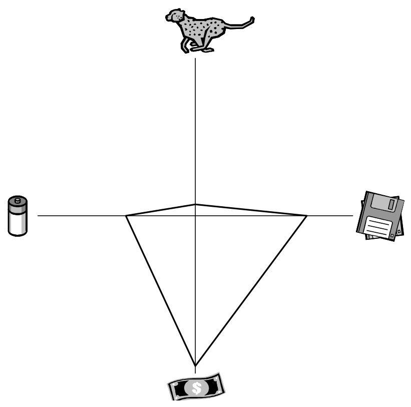

Figure 2. Performance of the Texas Instruments TMS320C5409

3. APPROACHES TO PARALLELISM

The processors surveyed in this paper achieve much of their performance by executing multiple operations in parallel. Although the details of each processor's parallel-execution techniques are beyond the scope of this paper, two techniques bear mention here. We refer to these two techniques as data parallelism and instruction parallelism.

3.1. Data Parallelism

All of the processors discussed in this paper support single-instruction, multiple-data (SIMD) operations. There are two common approaches to implementing SIMD operations: SIMD via multiple data paths and—more commonly—SIMD via data packing. As an example of the latter case, a processor with 32-bit resources (registers, data path, etc.) might perform two 16-bit operations per cycle by splitting each 32-bit word into 16-bit halves.

3.2. Instruction Parallelism

Except for the TMS320C54xx, all of the processors in this paper are multiple-issue DSP architectures. A multiple-issue architecture can execute two or more instructions per instruction cycle. Multiple-issue architectures fall into two broad categories: very long instruction word (VLIW) architectures and superscalar architectures. These categories are not rigid; there is a continuum of architectural approaches. It should be noted that SIMD can be—and often is—combined with multiple-issue techniques.

VLIW-based DSPs use long instructions composed of multiple sub-instructions. This approach is also known as compile-time scheduling because the sub-instructions are combined into a VLIW instruction by the complier or the assembly programmer. In contrast, superscalar architectures identify instructions that can execute in parallel as the code is executed. In this approach, known as run-time scheduling, the parallel execution of instructions may vary depending on factors such as the alignment of the code.

4. TEXAS INSTRUMENTS TMS320C64XX

Texas Instruments announced the first mainstream commercial VLIW-based DSP processor, the TMS320C62xx, in 1997. This 16-bit fixed-point architecture includes eight execution units (including two multipliers and four ALUs) and can use all of these execution units in parallel by issuing and executing up to eight 32-bit-wide instructions per clock cycle. In February 2000, TI announced the next generation of this architecture, the TMS320C64xx. The TMS320C64xx adds support for a range of 16-bit and 8-bit SIMD operations, including support for up to four 16-bit multiplications per cycle. Other new features of the TMS320C64xx include a 600 MHz clock speed and an expanded data bandwidth of 128 bits/cycle.

TMS320C64xx uses simple instructions, which helps simplify programming; however, the compiler or programmer must schedule instructions for parallel execution. The TMS320C64xx also has a longer pipeline than the other processors described in this paper (11 stages vs. 5 stages) and many operations have long latencies. Mitigating the effects of this long pipeline is a major challenge for the compiler or assembly-level programmer (see J. Bier et al. [1] for a detailed discussion of this problem).

The TMS320C64xx memory system is also fairly complex. The memory system includes two memory levels. The first-level memory (that is, the memory closest to the processor) is always configured as cache, and part of the second-level memory may also be configured as cache. These caches complicate the programming model and reduce the programmer's ability to predict program execution times.

Like most high-performance DSPs, the TMS320C64xx features an impressive list of on-chip peripherals. Various family members include peripherals such as PCI interfaces, high-bandwidth external memory interfaces, and coprocessors for Viterbi and turbo decoding.

The solid lines in Figure 2 shows the performance range of the TMS320C64xx family; the outer quadrilateral shows the best performance, and the inner quadrilateral shows the worst performance. These quadrilaterals represent the boundaries of available performance, not two specific products. For example, the outer quadrilateral shows speed for the 600 MHz TMS320C6414, but show affordability for the 500 MHz TMS320C6414. The dashed lines show the performance of the TMS320C5409 for reference. The following charts use the same convention to show the performance range of each processor family.

Thanks to its high level of parallelism and high clock speed, the TMS320C64xx is by far the fastest processor discussed in this paper. Except for the TMS320C54xx, the TMS320C64xx is the least energy-efficient processor surveyed here.

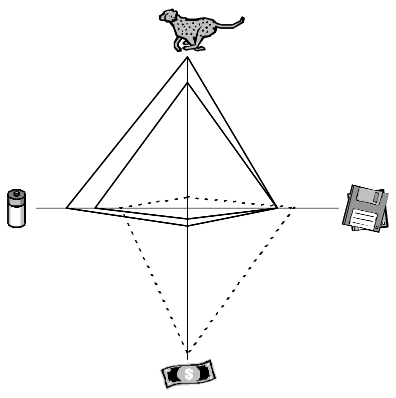
Figure 2. Performance of the Texas Instruments TMS320C64xx

The TMS320C64xx is fast, but it is also expensive. For example, the least expensive TMS320C64xx ($90) is about 2.5 times more expensive than the most expensive LSI40x ($36) from LSI Logic (which is discussed further below). Because of its large instruction words and long, complex pipeline, the TMS320C64xx also has far worse memory efficiency than any of the other processors discussed in this paper. However, the TMS320C64xx has a better speed-per-dollar ratio than most processors in this paper; only a few LSI40x family members offer better speed-per-dollar ratios. This is especially notable when one considers the ample peripherals and memory included in the TMS320C64xx.

The development tools and third-party support (that is, the supporting software and hardware provided by companies other than TI) for the TMS320C64xx build on the mature TMS320C62xx tools and support. However, the TMS320C64xx is a challenging target for programmers and compilers; the deep, complex pipeline is a particularly difficult challenge. Code optimization is also hampered by the dynamic caches, which reduce execution-time predictability.

A key strength of the TMS320C64xx is its assembly-level compatibility with other TMS320C6xxx family members, especially the TMS320C67xx floating-point DSPs. DSP application development often starts with floating-point algorithms, and it may be easier to migrate an application from the TMS320C67xx to the TMS320C64xx than, for example, from a PC to a conventional DSP. Due to its good support for 32-bit operations and its large address spaces, the TMS320C64xx is also a good target for general embedded computing tasks.

5. STARCORE SC140

In late 1998, Lucent Technologies (now Agere) and Motorola formed a joint technology development center called StarCore to develop DSP cores that would be used by the two companies in their own chip-level products. (Infineon has since joined with Agere and Motorola in forming a new core licensing entity, StarCore LLC.) The first core to emerge from StarCore was the SC140, announced in April of 1999 [2]. The SC140, like the TMS320C64xx, is a high-performance VLIW-based architecture. It includes four combined MAC/ALU/bit-field units and can issue and execute up to six instructions per clock cycle. The ALUs support SIMD operations such as dual-add. Like the TMS320C64xx, the SC140 has a data bandwidth of 128 bits/cycle. Chips based on the SC140 currently operate at up to 300 MHz.

The SC140 uses a simple, highly orthogonal instruction set. In other words, dissimilar SC140 instructions use a similar structure, and the instructions have fairly unrestricted access to processor resources such as registers. Almost all SC140 instructions support conditional execution, which allows the SC140 to either execute or ignore instructions based on the values of certain status registers. The SC140 uses a five-stage pipeline, which is much shorter than the eleven-stage pipeline of the TMS320C64xx. The SC140 also differs from the TMS320C64xx in that few SC140 instructions have multi-cycle latencies. Hence, the SC140 is a more benign target than the TMS320C64xx for a complier or assembly-language programmer.

To improve code density, the SC140 uses 16-bit instructions (half the width of the TMS320C64xx instructions) and allows the assembly programmer or compiler to add 16-bit “prefixes” where needed to extend the functionality of these instructions. This approach is similar to using a mixed-width instruction set (as found on the TMS320C55xx, for example), in which shorter instructions are used for control-oriented software, while wider, more powerful instructions are mainly used in performance-critical inner loops.

The three announced SC140-based chips all target communications infrastructure. These chips include the Agere StarPro2000, the Motorola MSC8101, and the Motorola MSC8102. The MSC8101 contains a single SC140 core and a simple memory system; both the MSC8102 and the StarPro2000 contain multiple SC140 cores and complex, multi-level memory systems. All three chips contain numerous on-chip peripherals that reflect their intended use in communications infrastructure, such as PowerPC bus interfaces and T1/E1 interfaces.

The solid lines in Figure 3 show the performance range of the single-core Motorola MSC8101 device. These results may not apply to the other SC140-based chips. As before, these two quadrilaterals represent the boundaries of available performance, not two specific products. The dashed lines show the performance of the TMS320C5409 for reference.

The MSC8101 is the most efficient architecture discussed here in terms of memory use, energy consumption, and performance per MHz. The MSC8101 is particularly efficient in comparison to the TMS320C64xx. However, the maximum clock speed of the MSC8101 is only half that of the TMS320C64xx, and the MSC8101 is not sufficiently efficient to compensate for this clock speed disadvantage. Hence, the MSC8101 is considerably slower than the TMS320C64xx.

Although the MSC8101 is efficient in most respects, it has the worst speed-per-dollar ratio of the processors discussed in this paper. For example, the prices for the 250 MHz MSC8101 and the 500 MHz TMS320C6414 are about the same, but the 500 MHz TMS320C6414 is 50%

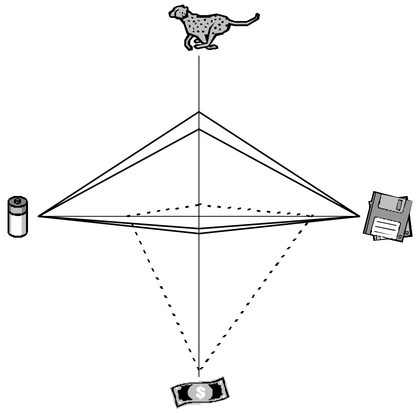

Figure 3. Performance of the StarCore SC140

faster than the 250 MHz MSC8101.

The initial development tools that BDTI used for its early evaluation of the SC140 were solid but unsophisticated, particularly compared to the tools for the

TMS320C64xx. The SC140 also has far less third-party support than the TMS320C64xx. However, the SC140 is a better target for compilers and assembly-language programmers than the TMS320C64xx.

A key weakness of the SC140 is its uncertain roadmap. StarCore announced a scaled-down version of the core, the SC110, in September 2000, but neither Agere nor Motorola have announced any products based on this core. Indeed, only three chips based on the SC140 have been announced so far, and these chips target a narrow range of applications. The future of the StarCore architecture is even less clear since Infineon joined Agere and Motorola in forming a new entity, StarCore LLC, to license the architecture. The details of StarCore's roadmap, licensing terms, and other business plans have not been released. On the other hand, the StarCore architecture is now backed by three of the largest semiconductor vendors and will also be available for license.

6. ANALOG DEVICES TS-101S (TIGERSHARC)

The ADSP-TS101S is a family of DSP processors from Analog Devices based on the "TigerSHARC" core, introduced in October 1998. The ADSP-TS101S, which operates at 250 MHz, is a VLIW architecture that can issue up to four instructions per cycle. The ADSP-TS101S operates on a variety of data types, including 8-, 16-, and 32-bit fixed-point and 32-bit floating-point. Although the ADSP-TS101S has some similarities with other Analog Devices processors—particularly the ADSP-2116x—it is incompatible with these processors [3].

The ADSP-TS101S provides two identical data paths, each of which contains an ALU, a MAC unit, and a shifter. The ADSP-TS101S supports the typical form of SIMD in which a single instruction operates on packed operands. In addition, the ADSP-TS101S can perform the identical operations in both data paths with a single instruction. These two types of SIMD operations can be combined so that a single instruction causes the two data paths to perform identical SIMD operations. The ADSP-TS101S can, for example, perform up to eight 16-bit fixed-point multiplications per cycle. The ADSP-TS101S can also perform up to two 32-bit floating-point multiplications per cycle.

The ADSP-TS101S is like the TMS320C64xx in that it uses 32-bit instructions. In other respects, the ADSP-TS101S is more like the SC140: the ADSP-TS101S uses a five-stage pipeline, and nearly all instructions have single-cycle latencies. As with the SC140, nearly all ADSP-TS101S instructions can be executed conditionally. Hence, the ADSP-TS101S is a fairly good compiler target, although its two-level SIMD architecture complicates the programming model.

The memory system of the ADSP-TS101S has three notable attributes. First, the ADSP-TS101S has a remarkably high internal memory bandwidth of 256 bits/cycle (twice as high as that of the TMS320C64xx and the SC140). Second, the ADSP-TS101S contains 768 Kbytes of on-chip SRAM; of the processors in this report, only the TMS320C64xx contains more memory (1056 Kbytes). Finally, the ADSP-TS101S contains "link ports" that allow up to eight ADSP-TS101S chips to access each other's memory without any external bus controllers.

Although the BDTI Benchmark results for the ADSP-TS101S are not yet complete, we can make some broad comparisons to other high-performance architectures. In terms of speed, the ADSP-TS101S is competitive with the high-performance processors surveyed here. For example, a 250 MHz ADSP-TS101S can perform 2.0 billion 16-bit fixed-point MACs per second. This is more than any processor surveyed here except for the TMS320C64xx, which can perform 2.4 billion 16-bit fixed-point MACs per second at 600 MHz. The ADSP-TS101S can also perform up to 500 million floating-point MACs per second. In comparison, Texas Instruments' fastest floating-point processor, the 225 MHz TMS320C6713, can perform 450 million floating-point MACs per second.

At $175, the ADSP-TS101S is the most expensive processor covered in this paper. The next most expensive processor, the 600 MHz TMS320C6416, costs $149. As another point of reference, the 225 MHz TMS320C6713 costs only $27 [4]. Hence, the ADSP-TS101S has a relatively poor price-performance ratio.

The tools for the ADSP-TS101S are very strong. However, its third-party support is limited. This is particularly problematic because the ADSP-TS101S is incompatible with other Analog Devices architectures and therefore cannot take advantage of code written for earlier processors.

There is also some uncertainty about the roadmap for the ADSP-TS101S. The ADSP-TS101S only began shipping in February 2002—over three years after the architecture was announced—and Analog Devices has not yet announced any other chips based on the TigerSHARC architecture.

7. LSI LOGIC ZSP400 AND LSI40X

The basic architecture of the ZSP400 was originally developed by ZSP Corporation in 1998. LSI Logic further developed the ZSP400 architecture after acquiring ZSP Corporation in 1999. At its introduction in 1998, the ZSP400 core was the first mainstream DSP core to make use of superscalar execution, branch prediction, and internal forwarding of results between execution units [5].

The ZSP400 is a 16-bit, fixed-point DSP that features two MAC units and two ALU/shifter units. Although it is inherently a 16-bit architecture, the ZSP400 has extensive support for 32-bit operations. The ZSP400 uses a 16-bit RISC-like instruction set and can execute up to four instructions per cycle. The ZSP400 also contains small instruction and data buffers that attempt to prefetch the data needed for upcoming instructions.

The ZSP400 execution units are fairly general-purpose. Like the other processors surveyed here, the ZSP400 supports SIMD operations that allows it to perform two 16-bit operations with a single instruction. Unlike the other processors in this paper, the ZSP400 does not contain dedicated address generation units. Instead, the ALUs double as address generation units.

The ZSP400 is available in three forms: as a licensable core, as part of LSI Logic's and IBM's ASIC libraries, and in LSI Logic's LSI40x family of packaged processors. In contrast, most DSPs are only available as packaged processors.

The solid lines in Figure 4 show the performance range of the ZSP400 family, including both currently available chips and the licensable core. As before, these two quadrilaterals represent the boundaries of available performance, not two specific products. The dashed lines show the performance of the TMS320C5409 for reference.

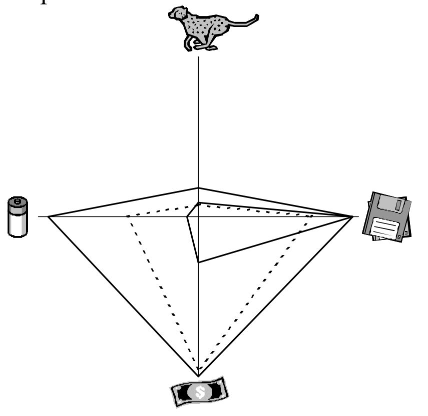
Figure 4. Performance of the LSI Logic LSI40x and ZSP400

Although the ZSP400 is faster than the TMS320C54xx, it is far slower than the high-performance processors discussed in this paper. This is to be expected: in terms of execution units, the ZSP400 bears more resemblance to the TMS320C54xx than to the high-performance DSPs surveyed here. However, the prices for the LSI40x are far lower than those of the other processors in this paper. As a result, the LSI40x offers the most speed per dollar of the processors in this paper. In addition, some versions of the LSI40x are remarkably energy efficient. (According to LSI Logic, early LSI40x chips were based on an older, less energy-efficient version of the core, while newer chips use a much more efficient version of the ZSP400 core.)

Because the ZSP400 is fairly new, its tools and third-party support are unproven. However, the ZSP400 has attracted an impressive list of licensees and third-party support providers. In addition, the ZSP400 is assembly-level upwards-compatible with the recently announced, higher-performance ZSP G2 architecture.

VLIW processors require the programmer or code-generation tools to explicitly identify instructions that should execute in parallel. In contrast, superscalar processors like the ZSP400 automatically identify instructions that can execute in parallel. Thus, the ZSP400 can achieve some parallelism without any effort on the part of the programmer or code generation tools. However, the ZSP400 has only limited abilities to identify instructions that can execute in parallel, and the programmer or tool must carefully arrange instructions in order to achieve maximum parallelism. This optimization may be harder than optimizing code for a VLIW processor.

8. CONCLUSIONS

Today's communications infrastructure system designers must evaluate and compare processors with few architectural similarities. Selecting a processor requires consideration of factors such as development cost and risk, system cost, energy efficiency, and speed. Many of these factors are difficult to gauge, and vendors do not always provide enough information to evaluate the suitability of a processor for an application. The situation is growing more complex as vendors continue to introduce new solutions, and as the requirements of DSP applications change.

9. REFERENCES

[1] J. Bier et al, *Buyer's Guide to DSP Processors*, BDTI, Berkeley, California, 2001.

[2] J. Eyre, G. Avkarogullari, et al, *Inside the StarCore SC140*, BDTI, Berkeley, California, 2000.

[3] J. Bier et al, *Processor Overviews*, www.BDTI.com/procsum/index.htm, BDTI, May, 2002.

[4] J. Giddings and K. Williston, *Pocket Guide to Processors for DSP*, www.BDTI.com/pocket/pocket.htm, BDTI, September, 2002.

[5] K. Williston et al, *Inside the LSI Logic ZSP G2*, BDTI, Berkeley, California, 2002 (forthcoming).