ADAPTIVE COMPUTING IC TECHNOLOGY ENABLES SDR AND MULTIFUNCTIONALITY IN NEXT-GENERATION WIRELESS DEVICES

Author: John Watson, Co-founder and VP Marketing
(QuickSilver Technology, San Jose, California, USA, john.watson@qstech.com)

ABSTRACT

Third generation (3G) and fourth generation (4G) wireless terminals will be required to provide exceptionally higher levels of service than their second generation (2G) counterparts. Ever-increasing demand for increased mobility introduces several physical requirements (e.g., longer battery life, reduced size, lighter weight, etc.), along with the requirements for massive processing power gains. Additionally, there is an increasing consumer expectation for software-defined radio (SDR), as well as multifunctionality, whereby multiple upgradeable standards, protocols, and applications can occur on a single platform, thus enabling a worldphone.

Wireless devices are fast approaching a point in the product development roadmap where, without a paradigm shift in the basic design architecture that moves away from fixed-function silicon technology, they will no longer be able to meet both the service and the mobility demands, simultaneously.

The ideal solution would be to take advantage of the processing power of the ASIC while retaining the flexibility of the DSP. This is the very essence of the Adaptive Computing Machine (ACM), described in this paper. The performance of an ACM for 3G devices, including SDR, is validated in hardware through the implementation of a series of baseband algorithms. Benchmarks of the ACM performance are presented, showing significant improvements to be feasible relative to conventional IC technologies.

1. INTRODUCTION

QuickSilver Technology's Adaptive Computing Machine, a new class of digital integrated circuit, is an outcome of its pioneering efforts in adaptive computing for commercial use. The ACM is the only software-programmable integrated circuit (IC) that combines high performance, low power consumption, low cost, and architecture flexibility in a single chip.

The inherent adaptability of the ACM's architecture allows algorithmic elements to be directly converted into dynamic hardware resources during run time. Simply put, software becomes hardware. The ACM changes on the fly, adapting tens or hundreds of thousands of times per second to create the exact hardware needed for that moment in time. This results in the most efficient use of hardware in terms of cost, size (silicon real estate), performance, and power consumption. The flexibility of the ACM enables not only SDR, but also longer battery life and multifunctionality – ideal attributes for bringing the PC experience consumers expect to next-generation mobile, wireless, and convergent devices.

Like many good ideas, the ACM concept is relatively simple, although its development requires a new approach to how we think about computing technology.

2. ALGORITHMIC EVALUATION

During the initial development of adaptive computing (AC) in the late 1990s, it became clear through mainstream research that FPGA-based reconfigurable computing (RC) has considerable limitations. Conventional RC technology approaches the problem at too macro a level. That is, RC tends to work at the level of entire applications or algorithms. In reality, it is critical to consider the problem at the micro level of algorithmic elements.

Consider just how many core algorithmic elements there are and for what purposes they are used. For example, consider the number of elements used in word-oriented algorithms, such as the compute-intensive Time Division Multiple Access (TDMA) algorithm employed in digital wireless transmission (see Figure 1 on the following page). Any variants, such as Sirius, XM Radio, EDGE, and so forth, form a subset of this algorithmic class. Therefore, a single adaptable architecture that can handle high-end TDMA will also be able to handle its less sophisticated cousins.

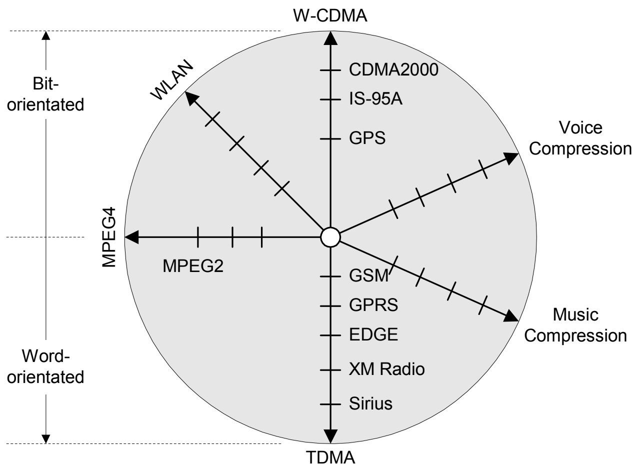

Figure 1. Algorithm Space

Once word-oriented algorithms have been evaluated, consider their bit-orientated counterparts, such as Wideband Code Division Multiple Access (W-CDMA) – used for wideband digital radio communications of Internet, multimedia, video, and other capacity-demanding applications – and sub-variants such as CDMA2000, IS-95A, and so forth. Other algorithms to consider comprise various mixes of word-oriented and bit-oriented components, such as MPEG, and voice and music compression. The ACM architecture is able to cover this very large problem space and all the points in between.

3. A HETEROGENEOUS AND FRACTAL ARCHITECTURE

Our evaluations revealed that algorithms are heterogeneous in nature, which means that, within a group of complex algorithms, their constituent elements are substantially different. In turn, this indicates that the homogeneous architectures associated with traditional FPGA-based RC approaches – which have the same lookup table replicated tens of thousands of times – are not appropriate for most algorithmic tasks. Even newly advanced FPGAs that have numbers of more complex elements like 18 x 18 multipliers don't satisfy the requirements of adaptive computing. The solution also had to incorporate the need to achieve the ASIC "gold standard" of high performance and low power consumption within the adaptable architecture even if it required rapid, real-time hardware adaptations from unexpected algorithmic inputs.

The solution is to create a fractal architecture that fully addresses the heterogeneous nature of the algorithms (see Figure 2). Start with five types of nodes: arithmetic, bit-manipulation, finite state machine, scalar, and configurable input/output used to connect to the outside world.

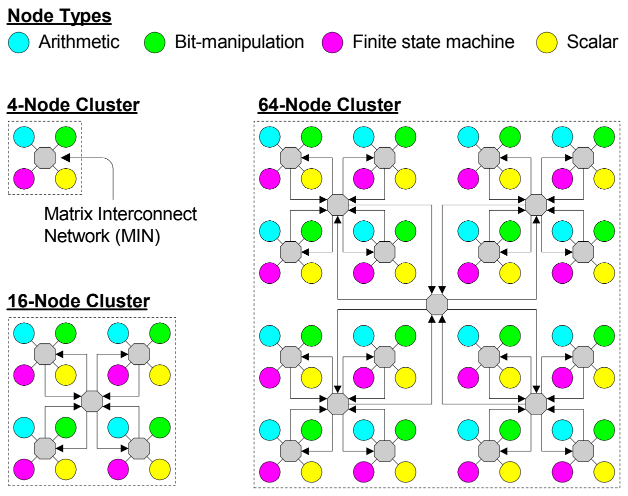

Figure 2. A Fractal Architecture

Each node consists of computational gates and its own local memory cache (approximately 75% of a node is in the form of memory). Additionally, each node includes configuration memory, but unlike FPGAs with their serial configuration bit-stream, an ACM has from a 32 to 128-bit bus to carry the data used to adapt the device.

It's important to realize that each node performs tasks at the level of complete algorithmic elements. For example, a single arithmetic node can be used to implement different variable-width linear arithmetic functions such as a FIR filter, a Discrete Cosine Transform (DCT), a Fast Fourier Transform (FFT), and so forth. Such a node can also be used to implement variable width non-linear arithmetic functions such as ((1/sine A) x (1/x)) to the 13th power.

Similarly, a bit-manipulation node can be used to implement different variable-width bit-manipulation functions, such as a Linear Feedback Shift Register (LRSR), Walsh code generator, GOLD code generator, TCP/IP packet discriminator, and other complex functions.

A finite state machine node can be used to implement any class of Finite State Machine (FSM). In the case of a really large or complex FSM, the machine can be spread across multiple FSM nodes, or different portions of the state machine can be time-sliced across a single node. This means that the node can be adapted to execute different portions of the state machine on-the-fly.

A scalar node can be used to execute legacy code, while a configurable input/output node (not shown in the figure) can be used to implement I/O in the form of a UART or bus interfaces such as PCI, USB, Firewire, and other I/O-intensive actions.

A key advantage of the ACM's architecture is that any node can be adapted to perform a new function, clock cycle-by-clock cycle. This means that any portion of the ACM – from just a few nodes and interconnects up to the entire chip – can be rapidly adapted, or changed. This results in a radical change in the way algorithms are implemented today. Rather than passing data from function to function, the data can remain resident in a node while the function of the node changes on a clock cycle-by-clock cycle basis. It also means that, unlike an ASIC implementation, the ACM can be adapted tens or hundreds of thousands of times a second, so that only those portions of an algorithm that are actually being executed need to be resident in the chip at any one time. This reuse of gates enables tremendous reductions in silicon area and power consumption as adaptations can happen in a micro-state rather than at the larger algorithm level (changing a FIR filter hardware element rather than completely rewiring the whole TDMA application). In the case of the ACM, this micro-adaptability versus a larger more modal (macro) adaptability allows for much more flexibility.

Algorithms have a locality of reference nature, meaning they always move from one mathematical step to the next immediate mathematical step and data is only passed to that next step (this can involve several steps in parallel but they are still the next logical data movement). This knowledge means that the wiring needed between next logical steps should be very dense, but the converse is not true. Unlike an FPGA that has very dense wiring along its whole XY plane, the ACM can reduce its wiring structure by understanding that algorithms do not communicate, or spread out, to the far ends of the silicon area. The algorithm always moves data to the next logical step. Therefore the ACM's wiring is fractal in nature; the further away any two nodes are, the less wiring there is between them. The closer two nodes physically reside, the more wiring there is between them.

Since the ACM's architecture is fractal in nature, it is totally scalable. A 4-node cluster is formed from arithmetic, bit-manipulation, FSM, and scalar nodes, which are connected via a Matrix Interconnection Network (MIN). A 16-node cluster is formed from four 4-node clusters linked by their own MIN, while a 64-node cluster is formed from four 16-node clusters linked by their own MIN, and so forth. An ACM can contain from one to thousands of node clusters, as required.

Because the architecture of the ACM is designed to efficiently compute and manipulate information at the algorithmic element level, it has a fractal wiring plane and heterogeneous compute array that can adapt on a single clock cycle to hold data in one physical area, while moving the logic around the data. It is very different from FPGA architectures that were originally built for TTL absorption, and ASIC prototype bug-fixes with an XY wiring plane that tie together all configurable logic blocks (CLB) with the same wiring structure as a homogeneous, fine-grained CLB array with very slow reconfiguration rates aimed only at complete algorithm model reuse.

These architectural differences lead to faster and easier ways to map applications into the chip's circuitry. While FPGAs and ASICs use high-level Hardware Description Languages (HDLs) and hardware synthesis, the ACM's tools are able to abstract the hardware specifics from the application, which can then become more like an embedded C design tool flow.

4. SPATIAL AND TEMPORAL SEGMENTATION

The spatial and temporal segmentation (SATS) process of the ACM enables SDR to occur. Unlike conventional IC technologies, the ACM architecture adapts to the problem at hand, enabling timesharing, or spatial and temporal segmentation. SATS is the process of mapping algorithms for a given task to dynamic hardware resources, then rapidly performing various portions of an algorithm in different segments of time (temporal) and in different locations in the adaptive fabric (spatial) of the ACM.

With SATS, the ACM's gates are rapidly reused, bringing into existence for the exact amount of time needed -- clock cycle by clock cycle -- the hardware an algorithm requires, and then efficiently running any number of different algorithms on the hardware engine.

Figure 3. Temporal Sharing

The temporal sharing aspect of the ACM fabric is illustrated in the Vocoder example, Figure 3. As each new task of the vocoder is needed, its binary file is loaded onto the ACM fabric from the cache. As the task is completed, it is removed from the fabric and the resources are freed for the next task. The ACM is adapting itself 400 times per second. In this example, the size of the ACM fabric is determined by the largest single element in the eight tasks. The other seven routines are able to run in the same space as the largest task. This reuse of the ACM fabric results in significantly more efficient use of the fabric and reduced costs.

5. DESIGN ADVANTAGES

ACM designs are represented in the SilverC™ language, which is C augmented with temporal and spatial extensions. Applications are developed in SilverC, and then compiled, to be expressed as downloadable applications in SilverWare™ binary modules. This means that, unlike an ASIC-based implementation in which algorithms are effectively frozen in silicon, the ACM can be quickly and easily adapted to accommodate the numerous evolving standards and protocols used in today's designs. In addition to accelerating time-to-market, this approach eases design reuse and reduces the risk of failure.

ACM technology also eliminates the very difficult problems associated with hardware/software co-design because the entire system is initially represented as software. Having said this, it's important to understand that SilverWare™ is not executed by the ACM in the same way that machine code is processed by a DSP, i.e. executing a long stream of instructions. Instead, SilverWare™ is used to dynamically adapt the ACM on-the-fly to create the exact hardware needed to perform whichever algorithmic tasks are required at any particular time. Complex algorithmic elements can be thought of as the smallest operators, and many of these complex algorithmic elements are temporally or spatially combined to form an application. In essence, software becomes hardware.

Because software is easier and faster to develop than the hardware of ASICs – *hours or days vs. several months or years, based on the number of functions* – a developer can rapidly move from design concept to silicon implementation for a product. Working in software also enables developers to make changes at any time during the design cycle, as well as after product shipment. For example, if updates or bug fixes are needed, turn around can quickly occur in software rather than going through the long lead-time and costly re-spin cycle of an ASIC.

6. THE BENCHMARKS

The first ACM test chip was compared to best-in-class ASIC implementations for a number of compute-intensive wireless functions. To date, these tasks have *always* been implemented in ASICs because DSP and FPGA implementations are much too slow, too power hungry, and use too much silicon area.

For example, as demonstrated in February at the Consumer Electronics Show (CES) in Las Vegas, the best-in-class ASIC was compared to an ACM for a CDMA2000 searcher with 2x sampling, using 512-chip complex correlations, with captured data processed at an 8x chip rate (equivalent to 16 parallel correlators running in real time). The ASIC took 3.4 seconds, while the ACM took only 1.0 seconds (3.4 x faster). In the case of a CDMA2000 pilot search, with the same parameters as above, the ASIC took 184 ms, while the ACM took only 55ms (3.3 x faster). For a W-CDMA searcher with 1x sampling using 256-chip correlations with streaming data, the ASIC took 533 µs while the ACM took only 232 µs (2.3 x faster). Furthermore, in addition to out-performing the three ASIC implementations, the single ACM counterpart performed all three tasks, used significantly less silicon real estate, and consumed only a fraction of the power.

7. THE MARKETPLACE

The software-to-hardware capability of the ACM technology enables OEMs of mobile, wireless, and convergent devices to achieve faster time to market, lower cost of development, higher margins, as well as the ability to extend the life of products, increase revenues, immediately react to market trends, create product differentiation, and build brand loyalty by offering a wide range of add-on features and functionality after the initial product sale. Service providers can go beyond their limited price-per-minute business model, with faster time to market, and new revenue streams for added features and services.

7.1 End-user advantages

By accessing any number of protocols, consumers will experience seamless roaming throughout the world, staying "connected" via the same single mobile device. Additionally, the ACM enables a single mobile/wireless product to perform a variety of functions, rapidly changing from a digital camera, to streaming video, to data retrieval, email, Internet and Intranet access, a global positioning system, or an MP3 player. The applications are limited only by the imagination. Today's handsets will essentially become mobile communicators with media rich (data/voice/image/video) applications and the needed features to call, page, email, and stay connected – at any time, and anywhere in the world.

For more information about QuickSilver Technology and adaptive computing, visit www.quicksilvertech.com.