The Acceleration Company

HARDWARE ACCELERATOR METHODOLOGY AND EXPERIENCE—THE TARARI CONTENT PROCESSOR™

A Tarari White Paper

This page intentionally left blank

Table of Contents

Executive Summary

Hardware acceleration technology relieves system bottlenecks by offloading compute-intensive algorithms from software running on host processors to dedicated hardware. For example, a file-scanning application includes algorithms designed to match patterns against the contents of files in a datastream. Without acceleration, the pattern-matching takes place in software on the host CPU and results in delays to other parts of the application as the algorithms consume precious cycles. With acceleration, the pattern-matching algorithms reside in specialized hardware in the same computer, and the datastream flows there for processing in a fraction of the time required on the CPU. Hardware acceleration, therefore, not only greatly reduces the compute-time required, but also frees the CPU for other tasks in the application.

Through analysis of opportunities for hardware acceleration, description of the relative merits of different approaches to acceleration, and case studies, this paper highlights **three key messages** in hardware acceleration:

1) Short-term success requires an *extensible and robust* platform.
2) Long-term success requires the ability to provide *incremental updates*.
3) Emerging technologies require *reconfigurable* accelerators.

The Tarari Content Processor is an implementation of hardware acceleration designed to address these key messages and to meet the changing demands of evolving technologies. The Tarari Content Processor, based on several generations of platform and infrastructure, is mature enough for easy deployment to applications in areas where standards are still emerging, such as XML processing and network security. Solution providers, therefore, can focus on perfecting and speeding up their applications rather than on laboriously putting acceleration machinery in place themselves.

Acceleration Opportunity

Properly evaluating the opportunity for acceleration transcends the bytes-and-cycles level of pairing a performance bottleneck with an acceleration solution. At the very least, there are three areas to explore: the market, the application and the algorithm.

Furthermore, proper evaluation starts with the wide focus of the market and moves to the narrow focus of the algorithm. This helps avoid the trap of developing a solution eminently suited to the technology of acceleration yet poorly matched to willingness of the market to adopt or pay for the solution.

The Market

Is there a market that can support the opportunity? Are prospects asking for it? Is there revenue to be had? At what cost? Do we know enough about the market to penetrate it credibly, even with a short-term solution? These questions form the point of departure for considering the opportunity for acceleration.

Emerging vs. Established Markets

While both emerging and established markets have performance demands that can be successfully addressed by acceleration, different degrees of flux in the market result in different requirements for flexibility in an accelerator. Emerging markets are marked by quickly evolving standards and multiple “small slice of the pie” approaches, and they embrace acceleration solutions that can be implemented quickly and updated later. As these markets shake out over time and a few “large slice of the pie” approaches come to dominate, flexibility and time-to-market in an accelerator lose importance, and price-sensitivity increases. This affects how the market will accept different accelerators.

Bulk encryption, a component of digital cryptography used in SSL transactions, offers examples of both established and evolving markets. From the 1970’s to the 1990’s, bulk encryption represented an established market dominated by the Data Encryption Standard (DES) algorithm, acknowledged as the large slice of the pie (for some applications, the entire and only pie). In the late 1990’s, bulk encryption came to represent an evolving market when it turned to a more secure successor to DES, the Advanced Encryption Standard (AES), and competition arose among candidate AES algorithms. At that point, the successful accelerator would have featured a robust solution for the established DES market as well as the flexibility to accommodate upgrades for AES.

In addition, in an emerging market it is often easier to enter quickly with a simple, "low-hanging fruit" solution that addresses urgent needs in the short term. As the market matures, two benefits accrue: the acceleration solution comes to be associated with the market as the low-hanging fruit is harvested; and the accelerator, if built on an extensible platform, allows for addressing higher-hanging fruit as well.

XML provides an example in the algorithm widely used in parsing. The low-hanging fruit in XML—the need most acutely felt by the greatest number of customers—is acceleration of the parsing function, so an extensible XML accelerator that addresses short-run needs in that area can be augmented later to perform higher-hanging functions such as Random Access XML (RAX) and XPath processing.

Severity of the Bottleneck

Beyond the issue of emerging and established markets is the nature of the bottleneck itself. How severe is the bottleneck? Are customers willing to pay to relieve it? How much are they willing to pay? Will they pay for relief today, or do they think that the problem will go away by itself in the tolerably near future? The answers to these questions inform the business plan for developing the accelerator.

Leverage into Other Markets

If the bottleneck meets these criteria, then it is possible that it will address bottlenecks in other, possibly unrelated markets. For example, a successful acceleration solution for regular expression (RegEx) matching, designed from the beginning for flexibility and extensibility, may also offer leverage into areas as different as network security, intrusion detection and spam filtering. The more versatile the solution, the lower the subsequent investment in development time in addressing the new markets.

The Application

The next area of examination is the application containing the bottleneck, because while both the market and the acceleration opportunity may exist, there may yet be difficulties in accelerating the application *as a whole*.

Algorithm Integration Point

For acceleration to be effective, it must integrate smoothly into the larger application, either through a standard, plug-in type of interface or else by modifying the application’s code to accommodate the acceleration technology.

For applications such as cryptography, Microsoft Windows offers both an API and the infrastructure to support plug-ins, and Linux offers the standard OpenSSL Library. An accelerator developed for these published integration points will appeal to more markets than will a proprietary accelerator.

In applications such as XML processing, it is possible to accelerate at published integration points, but the benefits from using a completely different model and new integration points (e.g., Random Access XML, or RAX) are vastly greater. In other words, an automobile’s 2-liter engine may perform better by introducing higher-octane fuel to a very simple integration point, but it may require the paradigm shift of a 3.4-liter engine and new thinking about the integration point to deliver the required performance increase.

Because moving a datastream from point A to point B with no intermediate copying is so critical, another role of the integration point is that of input/output (I/O), or moving data to and from the application. If the application or its environment does not support raw I/O or is somehow inherently I/O-restricted, acceleration will suffer. For instance, there are performance penalties to streaming data between an accelerator and the Java Virtual Machine. While there are Java interfaces that provide for raw I/O and direct manipulation of memory, not all applications support these interfaces.

Performance Profile

Not all algorithms require the same number of cycles per byte; therefore, not all applications are good candidates for acceleration. Accelerators work at the level of the algorithm, so the application with one or two very compute-intensive algorithms is a far better candidate for acceleration than is an application with multiple algorithms each consuming small fractions of CPU-time. Even though multiple algorithms can run on the same accelerator simultaneously, the application benefits more from acceleration of the most demanding processes.

Figure 1 below shows the performance profile of an application in which the functions Fn0 and Fn1 together consume over 80% of CPU cycles. Accelerating these two functions alone would not only deliver the greatest overall performance boost but also free up most of the system's main processor cycles for other tasks.

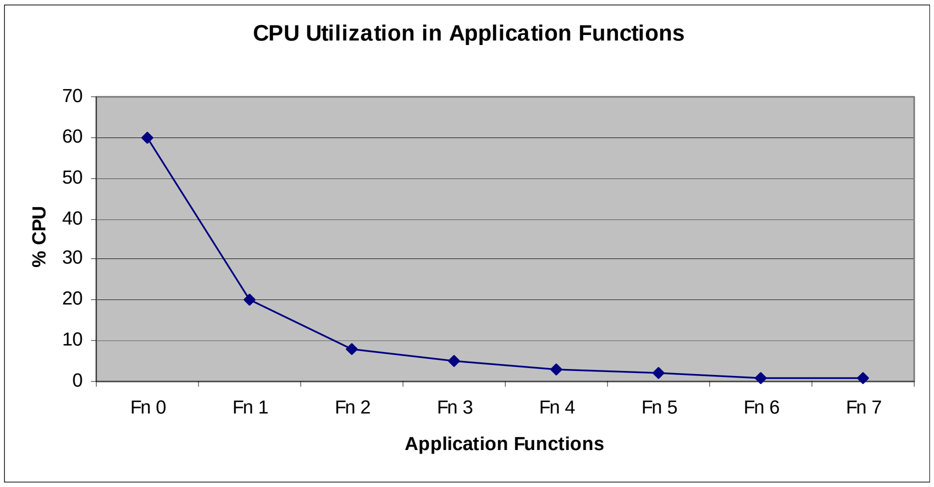


Figure 1—Good Candidate for Acceleration

Figure 2 shows the performance profile of an application which also comprises multiple algorithms, none of which, however, consumes any more than 20% of CPU resources. While it may be possible to accelerate all of the algorithms in this application for some overall benefit to the application, even the low-hanging fruit in this application is not especially tempting.

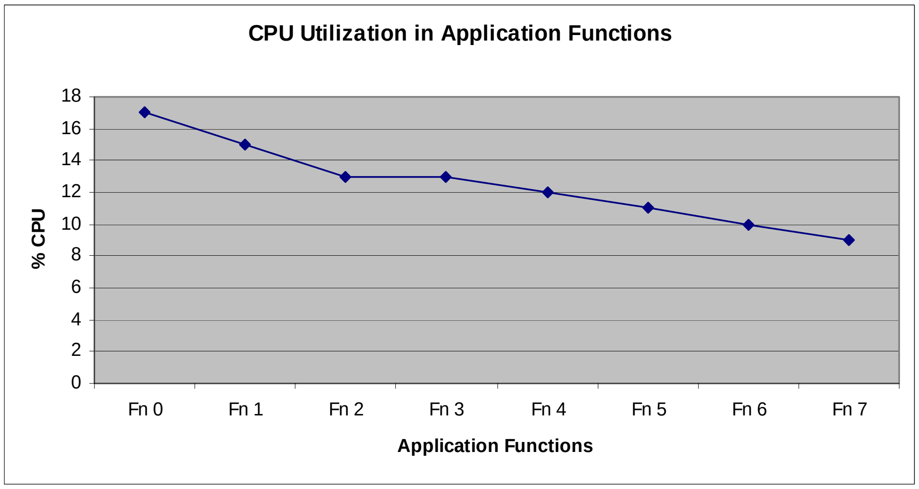


Figure 2—Poor Candidate for Acceleration

It is important to profile at the system level as well. This allows detection of bottlenecks outside of the algorithm and the application which may also require acceleration or greater bandwidth.

The Algorithm

Once the acceleration solution has passed the tests for market- and application-suitability, it must prove suitable at the level of individual algorithms within the application, since it is at this level that hardware acceleration delivers the most conspicuous benefits.

Algorithm Speedup vs. Application Speedup

The greater the desired factor of overall acceleration, the greater the fraction of work which must be accelerated. Speeding up Fn0 in Figure 2 by a factor of 50x will result in some overall acceleration, but speeding up Fn0 in Figure 1 will result in much greater overall acceleration, simply because so much more of the app's work is performed so much more quickly.

Amdahl's Law is a formula used to estimate the degree of systemwide acceleration due to a given performance increase in a given part of the system. In essence, attaining a large acceleration ratio requires accelerating a large fraction of the work:

$$Speedup = \frac{1}{(1 - FractionEnhanced) + \frac{FractionEnhanced}{SpeedupEnhanced}}$$

For example, 50x acceleration (or *SpeedupEnhanced* = 50) in an algorithm that consumes only 10% of system resources (or *FractionEnhanced* = .10) results in overall speedup of 1.11 times:

$$Speedup = \frac{1}{(1 - 0.10) + \frac{0.10}{50}} = 1.11$$

However, the same 50x acceleration in an algorithm which consumes 90% of system resources results in a much more conspicuous overall speedup of 8.47 times:

$$Speedup = \frac{1}{(1-0.90)+\frac{0.90}{50}} = 8.47$$

Applying Amdahl's Law to the principal functions or algorithms in an application, then, helps set expectations on the amount of systemwide benefit from acceleration. These calculations also demonstrate that, to attain systemwide acceleration on the order of 10x, it is necessary to offload and accelerate well over 90% of the application's work ($FractionEnhanced$).

Evaluating the Algorithm

Other, often overlooked factors affect the *SpeedupEnhanced* variable of the hardware-accelerated system. There are other aspects to acceleration than, for example, assuming that 10 individual functions performed simultaneously at the equivalent of 100MHz will make the hardware represent a 1GHz processor. Also of importance are:

- **CPU Offload**

  Hardware acceleration frees up the CPU to handle other demands of the application. Once the accelerator is addressing the most critical bottlenecks, the CPU can address lower-level bottlenecks, or simply conduct other tasks in the application.

- **Increased Throughput**

  In most applications, the newly freed-up CPU resources are devoted to feeding a greater volume of data to the algorithm running on the accelerator. As long as the accelerator can keep up with the datastream, the application continues to benefit from increased throughput.

- **Cycles per Byte Ratio**

  The Cycles per Byte Ratio describes the average number of CPU cycles spent in processing each byte of input data in a given application; the higher the ratio, the better the candidate for acceleration. Cycles per byte, measured at the outside of the system and encompassing all internal variables (CPU speed, bus speed, hardware acceleration, data flow in the application) accounts for the CPU resources to be freed up by offloading work to accelerator. So, the RSA algorithm is an excellent candidate for acceleration because its complex calculations require simultaneous multiplication of very large numbers—over 90% of the CPU's total compute cycles—and the CPU devotes many more resources to computation than to moving data.

Tarari Design

Following the evaluation of market, application and algorithm comes that of the design of the hardware accelerator itself. Which design meets the criteria set out on page 1: an extensible and robust platform with the capacity for receiving incremental updates and reconfigurability for emerging markets and technologies?

State of the Industry

Hardware Accelerator Categories

There are two primary categories of hardware accelerator:

- **Co-processor**

The co-processor is a plug-in card which offloads the work of acceleration from the host processor over a peripheral bus, such as PCI, to the card in the same system.

- **Appliance**

An acceleration appliance offloads the work of acceleration across network interfaces for input and output. It resides on the network as a separate system.

Each category has its relative merits, depending on factors such as latency, bandwidth, the movement of data to and from the accelerator, and how close to the bottleneck the accelerator can be integrated.

Options—Market Requirements and Processing Elements

Within the categories of hardware accelerators, there are several types of processing element used. (In the accompanying graphics, the shaded zones depict the extent to which each type of processing element satisfies the divergent market needs of Decreasing Time to Market, Increasing Volume, Increasing Standard Maturity and Increasing Performance.)

- **General Purpose CPU**

  It is possible to introduce to the system a PCI card with an X86-class processor running easy-to-use acceleration software that can be modified for

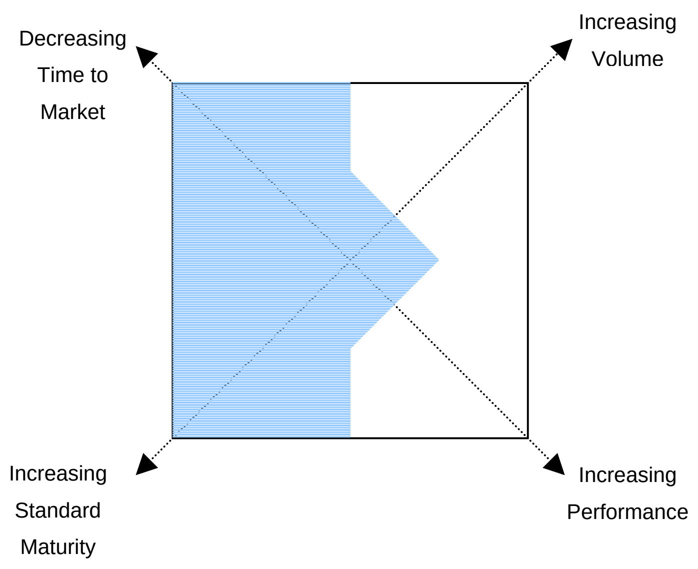


Figure 3—Market Coverage, General Purpose CPU

emerging standards. This offloads some of the work from the host processor; however, the price per chip is relatively high, and the acceleration processor is not specifically designed for the algorithm, so the price/performance ratio is not optimal.

- **Application-Specific Integrated Circuit (ASIC)**

  An ASIC designed for a particular algorithm offers the maximum potential for acceleration at relatively low production cost in large volumes. An ASIC can do one thing extremely well, but it can do only that one thing, and it cannot be reprogrammed when standards or requirements change. Design and prototype costs are relatively high, and time to market is long, so it is of greatest benefit when standards are mature, as in established markets.

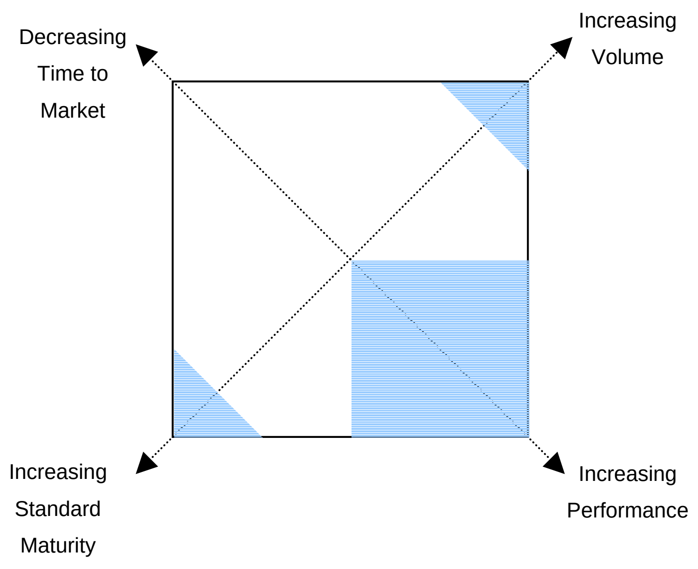


Figure 4—Market Requirements, ASIC

- **Reconfigurable Logic**

  More a platform than dedicated hardware, reconfigurable logic in silicon combines the benefits of relatively low design costs, short time to market and high performance. Reconfigurable logic goes beyond the flexibility to accommodate changing standards; the silicon can be programmed to execute completely different algorithms from one second to the next, outstripping the adaptability of ASICs and the performance of general-purpose CPU’s.

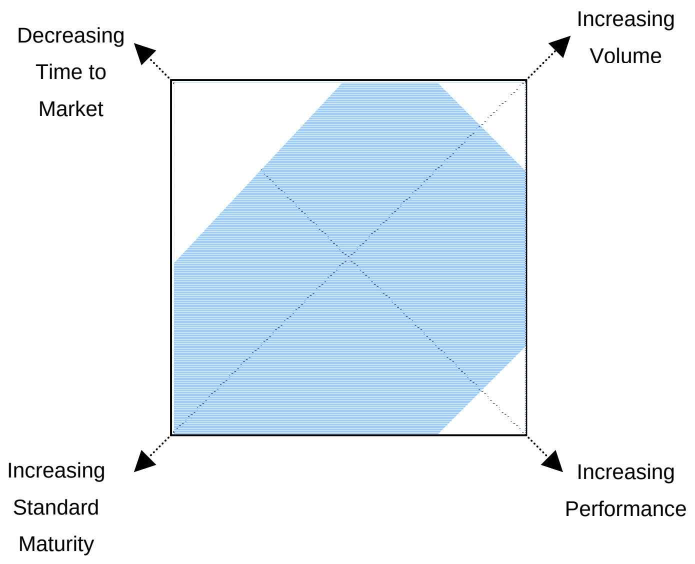


Figure 5—Market Requirements, Reconfigurable Logic

The Tarari Solution

Processing Elements

As the graphics above show, reconfigurable logic in silicon most nearly satisfies the market requirements of flexibility, performance, time to market and volume pricing. Tarari extends reconfigurable logic even further, to satisfy all market requirements of hardware acceleration.

- **Tarari Content Processor**

  The Tarari Content Processor supplements the time-to-market advantage of standard reconfigurable logic. It offers a solid hardware platform and flexible software framework for accelerating algorithms and applications. The Tarari Content Processor rounds out the high performance of reconfigurable logic with the infrastructure needed to create and deploy solutions in the shortest amount of time and the capacity for receiving updates.

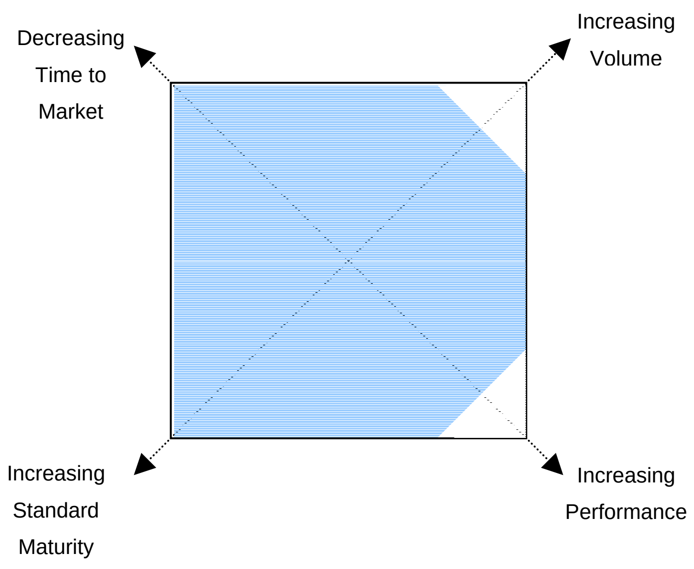


Figure 6—Market Requirements, Tarari Content Processor

- **Tarari ASIC**

  The Tarari ASIC completely satisfies market requirements for acceleration by blending the highest volume and highest performance advantages of the ASIC with the highest flexibility and shortest time to market of the Tarari Content Processor. The same acceleration solution can migrate from the Tarari Content Processor to a Tarari ASIC.

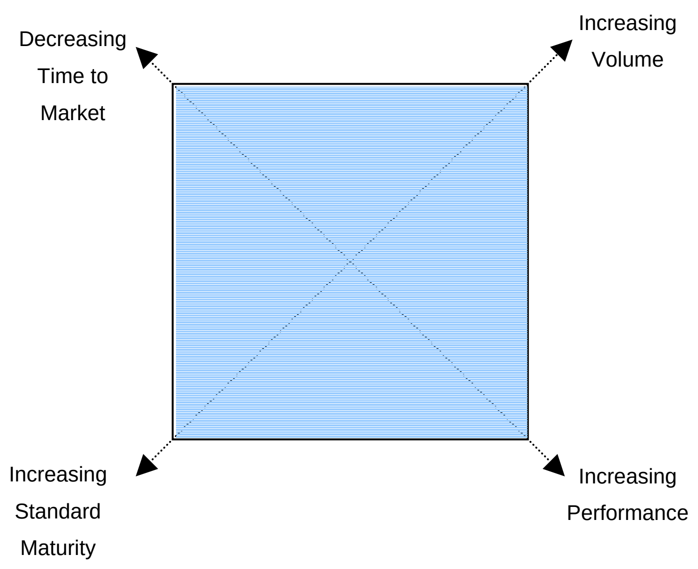


Figure 7—Market Requirements, Tarari ASIC

Design Goals

Tarari's approach to design has evolved through multiple generations of acceleration technology and rests on multiple goals (key points in **bold**):

- **Address multiple emerging markets** with minimal incremental effort

  The flexible platform and building blocks of the Tarari Content Processor make it easy for small engineering teams to gain leverage from one solution to another, especially when the focus is on bottlenecks that are common to multiple markets. The goal of minimal incremental effort requires a robust, stable platform upon which to build and run an accelerator.

- **Establish a market beachhead**

  To get the most out of the platform, it is necessary to establish a beachhead early, preferably in an emerging market.

- **Expand the market beachhead**

  Periodically updating functionality (new features, increased performance) expands the market beachhead and keeps the solution in the market for a long time. The increased longevity of the design lowers the overall cost of the solution and extends the lifespan of the product.

- **Accelerate application by 10x**

  As described above, algorithm speed-up and application speed-up are very different and their relationship is not always readily discernible. While Tarari’s platform seeks to accelerate algorithms by a factor of 50-200x and applications by a factor of 10x (e.g., 10Mbps to 100Mbps throughput), it is still worthwhile to create a beachhead in a market with the initial Tarari Solution delivering a modest performance boost, then to stair-step acceleration and functionality dramatically in updates.

Content Processing Platform—Design Elements and Implementation

The Tarari Content Processor offers a platform that combines hardware components and extensible software framework to achieve these design goals. Its mature acceleration machinery allows solution providers to focus on speeding up their applications without having to build such infrastructure themselves. Accelerating new algorithms is very easy with this platform because the most difficult building blocks are already in place.

Figure 8 illustrates the relationships among the Tarari

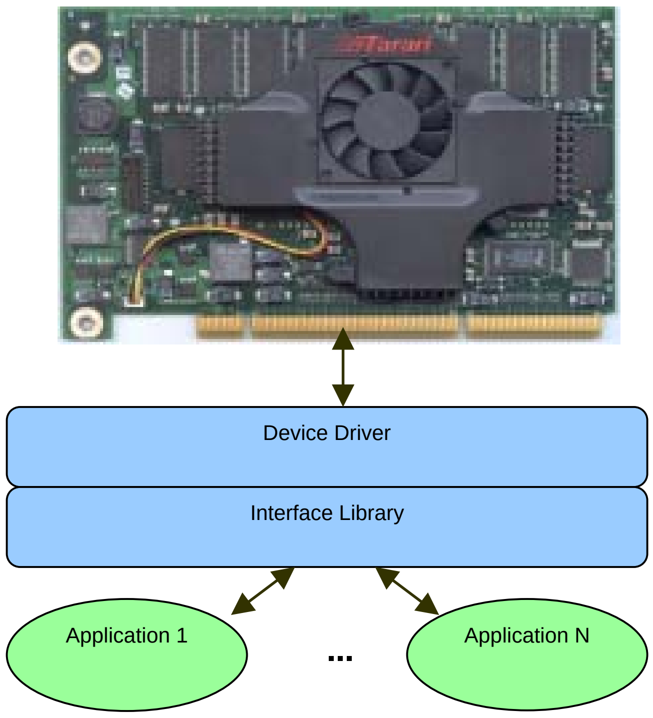


Figure 8—Content Processing Platform

Content Processor, its device driver, its fully documented interface library and the applications—1 through N—which it accelerates.

Tarari's platforms started as a memory card and have evolved to provide the on-board resource management, software framework and algorithm management needed for hardware acceleration, thereby minimizing the additional technical effort required. The main components on the platform are:

- **Content Processing Controller (CPC)**

  The CPC simplifies algorithm management. It configures the Content Processing Engines (see below) by initializing them with software images of the acceleration work to be performed. It absorbs the burden of managing memory allocation and arbitration (accessing SRAM and DRAM, communicating across the PCI bus to software).

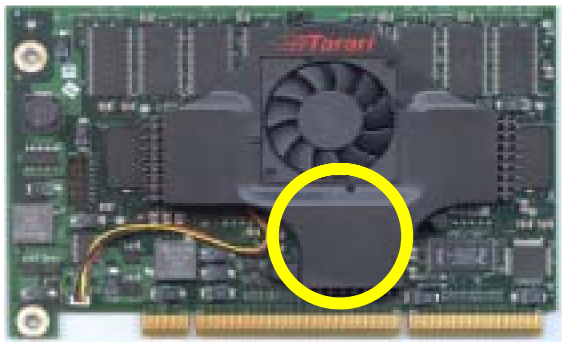


Figure 9—Content Processing Controller

- **Content Processing Engines (CPE's)**

  The CPE's form the foundation of reconfigurable logic on the platform and simplify the implementation of algorithms in hardware through a standard interface for writing them. The infrastructure for execution, therefore, is in place, ready to accommodate algorithms.

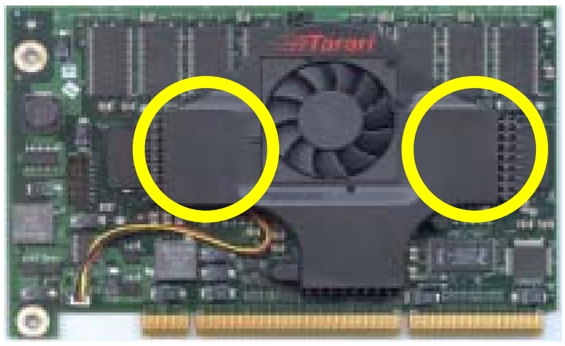


Figure 10—Content Processing Engines

- **On-board Resources**

  The Tarari Content Processor is endowed with resources adequate for handling the most complex algorithms at 4Gbps PCI bus throughput, including 256-2048MB of high-bandwidth DDR (Double Data Rate) Memory and dedicated low-latency SRAM (Standard Random Access Memory).

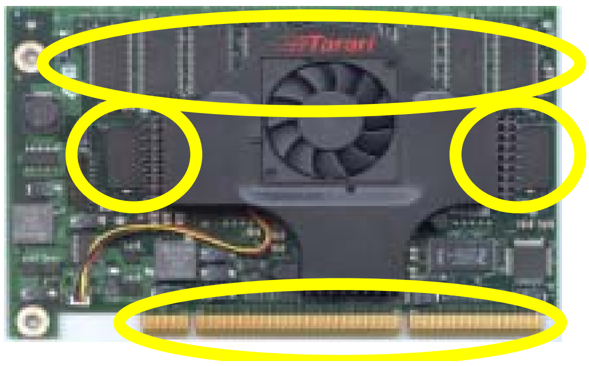


Figure 11—On-board Resources

- **Building Blocks of the Algorithm Suite**

  The software infrastructure of the Tarari Content Processor allows for different algorithms—or different instantiations of the same algorithm—to run as discrete, connected software building blocks in a suite. This versatility is particularly important in emerging markets because shifting technologies may call for the addition, deletion, substitution or modification of individual algorithms in a suite, which ASICs cannot accommodate. In an anti-virus application, for example, a datastream running across the PCI bus and onto the Tarari Content Processor may trigger a decoding algorithm depending on MIME type, then a decompression algorithm on an archive file, then a regular expression-matching (RegEx) algorithm to scan the file for virus signatures. In Tarari's hardware-accelerated implementation of Random Access XML (RAX), the Tarari Content Processor executes algorithms as diverse yet function-critical as DES/3DES, parsing, XPath, RSA, character conversion, SHA-1 hashing and random-number generation.

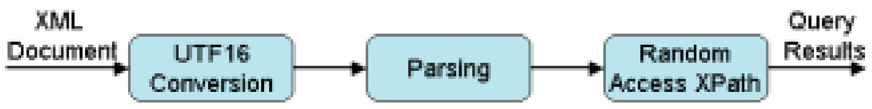
Tarari Design Evaluation

Tarari Design Goal	**Tarari Feature**
Address multiple markets	Building-Block Algorithm Suite
Establish market beachhead	Reusable Tarari Content Processor Infrastructure
Expand market beachhead	Incremental Updates
Extend lifespan of acceleration solution	Incremental Updates
Accelerate algorithm by 50-200x	Reconfigurable Logic in Content Processing Engines
Accelerate application by 10x	Reconfigurable Logic in Content Processing Engines

Tarari Case Studies

Three case studies will illustrate the successful deployment of the Tarari Content Processor in acceleration applications.

Case Study: XML Processing



The Problem

XML messages and documents, with their versatility and structure, are gaining wider acceptance as the linchpin of electronic transactions. However, this acceptance comes at the cost of verbose messages and the demand for processor cycles for parsing.

XML processing also represents an example of accelerating multiple functions in sequence on a datastream. There are high costs in I/O and programming overhead associated with repeatedly accelerating a function in hardware, then returning results to the application, then handing off to hardware again for the next function.

In this example, an XML document undergoes three functions: conversion to UTF16 file format; parsing to break the XML document into constituent components according to its structure; and Random Access XML, which runs XPath queries on the tokens and puts out the query results.

The Opportunity for Acceleration

- The Market
  There are several competing standards in the market of XML processing—Document Object Model (DOM) Simple API for XML (SAX), and now a new proposal based on Random Access XML (RAX) is under

discussion. Furthermore, implementations of the standards vary widely as enterprises attempt to deploy XML applications without suffering increasingly acute performance bottlenecks.

- The Application
  XPath processing, while exceptionally powerful, is also exceptionally compute-intensive, and can consume upwards of 90% of CPU resources. Parsing, depending on its use in XML processing, can also consume a significant share of CPU resources.

- The Algorithm
  Profiles show that software approaches to XML processing require tens of thousands of cycles per byte. Algorithms performing the same work in hardware take advantage of parallelism to drive this down considerably.

Algorithm Suite

There are individual approaches to accelerating the algorithms required in XML processing, from the fundamental step of parsing to the simultaneous execution of XPath queries in RAX. In the Tarari solution, however, the algorithms function as a suite in the software framework of the Tarari Content Processor, with consequent benefits to overall performance.

- Parser
  2-5x algorithm speedup

- UTF16 Conversion
  CPU offload

- RAX
  40-200x application speedup (inclusive of parsing)

Design Goals Met

Referring to the goals in *Design Summary* above, Tarari's XML processing solution achieves the following:

- Goal 1: Addresses multiple markets
  Since the building blocks of the algorithm suite enable functions as raw as parsing XML and as refined as performing queries on the XML messages, they are well suited to XML as implemented in applications as diverse as databases and content-based routing.

- Goal 2: Establishes market beachhead
  Accelerating the parsing function alone offers significant speedup at the application-level. This provided not only quick entry to market, but also valuable, early feedback on what customers wanted next.

- Goal 3: Expands market beachhead
  RAX acceleration, the next building block in the suite, proved easier to implement once the beachhead had been established with parsing. Customers deployed RAX as a simple firmware update, rather than as an invasive change to system hardware.

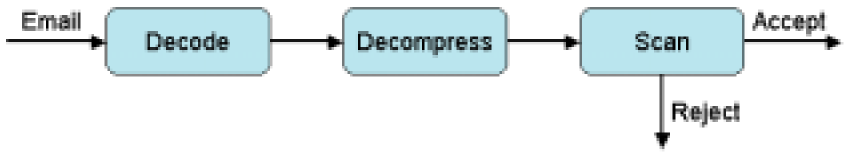
- Goal 4: Accelerates application by 10x
  XML acceleration at the application-level boosts performance well beyond the goal of 10x to between 40x and 200x.

Lessons Learned

Tarari distills its experience in accelerating XML processing to these main lessons:

- Profile the application early and often.
  Profiling different types of traffic (e.g., different file sizes and types of XML documents) helps to identify other points with high potential for acceleration.
- Consider acceleration opportunities holistically.
  The Tarari Content Processor can be configured to parse raw XML successfully at a throughput rate of 2Gbps. While this is an interesting milestone, it loses practicality in a world of applications that suffer from multiple bottlenecks. Rather than over-allocate hardware resources to a single bottleneck, Tarari took advantage of reconfigurable logic to redistribute resources on the Tarari Content Processor and create a holistic, balanced system that addressed multiple bottlenecks (e.g., cryptography as well as parsing).

Network Security



The Problem

Protecting the network from malware is at the top of every system administrator's priority list. Competing with that, of course, is the priority of keeping network traffic moving smoothly, a task made harder as more and larger files require scanning for more types of infection.

Inbound files undergo a typical sequence of decoding, decompression (in the case of archive files) and scanning. On exceptionally large files, or in exceptionally high traffic, any one of these operations can result in a bottleneck; the combination of all three can hamper network throughput unacceptably.

The Opportunity for Acceleration

- The Market
  While the standards for decoding and decompression (gzip, pkzip, uuencode, etc.) are established, there is no established standard for the scanning done by various anti-virus products (or, similarly, spam filters or intrusion detection). Overall, then, this is an emerging market, calling for quick market entry with a flexible product. The initial solution starts the customer interaction needed for subsequent updates based on the standards actually in use.

- The Application
  The decode/decompress/scan combination can, at its most severe, devour 90% or more of CPU cycles, slowing network traffic to a crawl. Even if such compute-intensiveness is only a function of random events (widespread downloading of patches, periodic huge attachments), its effects are still undesirable.

- The Algorithm
  In software approaches, decompression alone takes hundreds of cycles per byte, and the additional tasks of decoding and scanning can easily boost this to thousands of cycles per byte.

Algorithm Suite

To round out its network security offering and meet the market's needs, Tarari's algorithm suite comprises these tools:

- Decoder
  Supports Base64 and UUEncode

- Decompressor
  Handles PKZIP (WinZip), gzip, and LZW algorithms

- Fixed Pattern Matcher
  Scans for set of known byte sequences (here, static virus signatures)

- Regular Expression (RegEx) Matcher
  Scans for set of regular expressions (here, wildcards and characteristics of virus signatures)

Design Goals Met

Tarari's network security solution achieves the following, in light of the *Design Summary* above:

- Goal 1: Addresses multiple markets
  Decoding and decompression represent fundamental steps in analyzing any kind of network traffic, so Tarari's existing building blocks for these steps allow it to enter, in this case, the network security market. Similarly, RegEx compares any regular expression to a window in a datastream, so whether the function is benign (e.g., text search) or defensive (e.g., anti-virus, anti-spam, intrusion detection, content filtering), the same building block underlies it.

- Goal 2: Establishes market beachhead
  In its initial release, the solution comprised the decoder, decompressor and fixed pattern matcher, a combination adequate to accelerate network security against sets of known virus signatures.

- Goal 3: Expands market beachhead
  As a result of customer feedback, the scanning function evolved to include a full RegEx matcher, allowing more-robust identification of malware in the datastream.

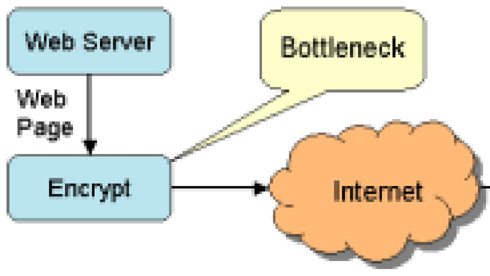
- Goal 4: Accelerates application by 10x
  At the application-level, the network security solution boosts performance far beyond 10x to as much as 40x.

Lesson Learned

- Eliminate multiple data transfers to the accelerator.
  Tarari's solution effectively "amortizes" the costs of transferring data to the accelerator by processing those data as much as possible while they are on the accelerator. One of the greatest performance benefits to having the Content Processing Controller reside on

the accelerator is that it supports the chaining of one algorithm’s output to another algorithm’s input (e.g., decoding to decompression, decompression to scanning) without the return trip across the bus to the software and CPU.

Cryptography



The Problem

The compute-intensiveness of SSL in the encryption step of serving secure Web pages dramatically reduces the connection rate with the server and introduces a severe performance bottleneck.

The Opportunity for Acceleration

- The Market
  The year 2000 saw an emerging cryptography market as the AES standard, the successor-apparent to DES/3DES, was still in flux. Although the standards are more stable now, the severe bottleneck in encrypting online transactions continues to plague e-commerce, so the market opportunity persists.

- The Application
  Standard interfaces for cryptography are manifold and mature (OpenSSL, Java Cryptography Extension, Microsoft Crypto API), giving Tarari the potential to plug cryptography acceleration in easily, with the main application unaffected by the change in the underlying software framework.

  Furthermore, crytographic functions consume over 90% of a typical Web server's CPU resources, suggesting an enormous boost in application performance once these functions are offloaded to acceleration hardware.

- The Algorithm
  Profiles of the software libraries show that tens of thousands of cycles go into processing each byte of data in cryptography, leaving ample opportunity for better performance through parallel processing in hardware.

Algorithm Suite

Accelerating any single cryptographic function results in a modest overall increase in performance, but gradually accelerating multiple functions with the Tarari Solution results in a dramatic increase in performance, rounds out the suite of cryptography-related algorithms and makes a more profound impact on customers and the market.

- RSA Public/Private Key Operations
  10x increase in new connections to Web server per second
- DES/3DES Symmetric Encryption
  3x throughput increase for large files
- SHA-1 Hashing Algorithm
  2x throughput increase for message digests
- True Random Number Generator
  Higher quality than software-generated pseudo-random numbers

Design Goals Met

Referring to the goals in *Design Summary* above, Tarari's cryptography solution achieves the following:

- Goal 1: Addresses multiple markets
  Cryptography is nearly ubiquitous in networking—a rich source of emerging markets—so creating building-block solutions in an area such as RSA has conduced to solutions in similar areas, such as RNG.
- Goal 2: Establishes market beachhead
  Tarari's initial, quick-to-market release in cryptography accelerated RSA for a systemwide speedup.
- Goal 3: Expands market beachhead
  Tarari then capitalized on the software framework in place and fleshed out its cryptography suite with short, one- to two-month development efforts (DES/3DES, SHA-1, RNG) as the market required. Furthermore, once in place, the Tarari Content Processor and its infrastructure will accommodate customers' eventual migration from DES/3DES, as AES becomes the bulk encryption algorithm of choice, thereby extending the lifespan of the cryptography acceleration solution.
- Goal 4: Accelerates application by 10x
  The cumulative performance boost to the application from the suite of cryptography algorithms exceeds the goal of 10x, providing up to 20x improvement.

Lessons Learned

From Tarari's experience in accelerating cryptography, two prominent lessons have emerged:

- Design and build a solution with commoditization in mind.
  The main factors driving commoditization in this category are price and customer preference for fewer devices. Hardware accelerators for cryptography have nosedived in pricepoint from over $10,000 to less than $500 as the form factor has become smaller and the technology has become smarter. Tarari's building-block architecture has not only allowed it to meet these market requirements by offering cryptography with other products required by emerging markets—rather than trying to force itself into the established market by offering ordinary cryptography components—but it has also extended the revenue stream from first- and second-generation intellectual property (e.g., RSA acceleration).
- Apply resources to the development effort as needed, not up front.
  In technology, it is easy to predict that standards and market needs will evolve; however, it is far more difficult to predict how and when they will evolve. Anticipating the potential shift from DES/3DES to AES, and knowing that the flexible platform of the Tarari Content Processor would accommodate the migration smoothly, Tarari did not need to scramble for a new, ground-up solution. Rather than dedicate engineering resources to what-if projects and reserve precious resources on the Tarari Content Processor for contingencies, Tarari was able to focus on near-term market needs, evaluating them as they evolved and getting to market quickly with the right solution at the

right time—moving fast when it made sense to move fast.

Hardware Acceleration—Key Messages

Hardware acceleration grows in importance as technology demands more of venerable, indispensable functions such as compression, expression matching, encryption and high-performance computing.

As customer needs and market requirements evolve, three truths about hardware accelerators stand out:

1. **Short-term success requires an extensible and robust platform.**
   Long development cycles are not compatible with getting an accelerator to market quickly. Short time-to-market demands reuse of an existing platform, if that platform delivers high performance while also being extensible to new applications.

2. **Long-term success requires the ability to provide incremental updates.**
   As the accelerator becomes accepted in the market, customers seek to increase short-term functionality for the long term. The capacity for incremental updates (added features, improved performance) throughout the life of the product is what allows the accelerator to grow from a short-term, stopgap fix to a long-term, market-satisfying solution.

3. **Emerging technologies require reconfigurable accelerators.**

   While established, slow-changing markets may be more forgiving, emerging markets and technologies absolutely require that accelerators be reconfigurable to keep pace with evolving standards and frequent changes in the needs they address. This requirement of an accelerator goes beyond the advantage of accommodating a faster chip or more memory; the core functionality of the accelerator must be reconfigurable.

Conclusion

Solutions dependent on high profit margins with little room for evolution are becoming extinct. The successful model for hardware acceleration assumes commoditization within five years and capitalizes on a platform that has been following the falling pricepoint and reducing the number of devices required.

Tarari’s hardware accelerator methodology is the result of many years of experience and three generations of evolution. Since it is risky to anticipate the needs of an emerging market and build a solution based on them, Tarari has developed a platform flexible enough for short-term success in a variety of markets, and extensible enough for long-term success through updates.

For More Information

To begin exploring how Tarari's hardware acceleration can address your organization's needs, please visit our Web site: www.tarari.com or contact us at info@tarari.com.

This page intentionally left blank

HARDWARE ACCELERATOR METHODOLOGY AND EXPERIENCE—THE TARARI CONTENT PROCESSOR™

A Tarari White Paper

Additional information: info@tarari.com

Internet: www.tarari.com

Telephone: (858) 385-5131

Fax: (858) 385-5129

Tarari, Inc.

10908 Technology Place

San Diego, CA 92127-1874

USA