



Intel[®] Technology Journal

Converged Communications

Using Intel[®] Technologies to Build Next-Generation Media Servers

Using Intel® Technologies to Build Next-Generation Media Servers

Joseph Grecco, Digital Enterprise Group, Intel Corporation
Mark Mize, Digital Enterprise Group, Intel Corporation
Ranjan Singh, Digital Enterprise Group, Intel Corporation

Index words: Media Servers, Video, VoIP, IMS, NGN

ABSTRACT

In this paper we describe the role of the media server in traditional Public Switch Telephone Network (PSTN), Internet Protocol (IP), and next-generation networks. We examine the Intel® building-block technologies and their use in developing powerful, cost-effective multimedia communication solutions. We begin with a description of the components that make up a media server and then outline three representative network configurations into which media servers are deployed. The following sections examine the Intel technologies used to build a next-generation media server from the top down. We start by taking a look at Intel NetStructure® Host Media Processing (HMP) Software, and how it makes developing interactive multimedia applications straightforward and cost effective; we walk through the steps required to build a simple voice and video application. The foundation of Intel NetStructure Host Media Processing Software is the Intel Architecture processors, chipsets, and Intel® Integrated Performance Primitives (Intel® IPP). We show how Intel Architecture processors and Intel IPP can offer world-class performance of media processing algorithms such as audio and video codecs. Finally, we look at how developer tools like the Intel C++ Compiler and VTune™ Performance Analyzer can be used to produce high-performance application code.

INTRODUCTION

Fixed-mobile convergence—the convergence of wire line and wireless devices into a single telecom network—promises to enable service providers to reach a greater

number of potential customers with a wider range of service offerings. Service providers in the telecommunications industry are looking to multimedia to increase revenues by allowing them to add video to their traditional voice services, thus enriching the end-user experience and making the services more attractive to their subscribers. For example, adding video content to a traditional voicemail application enables a new level of personalization. Users can be greeted with a subscriber's personal video, humorous clip, or cartoon animation. This level of personalization is particularly appealing to the teenage subscriber base.

Greater levels of personalization are being provided via new revenue-generating services such as video color ringback and video caller ID. For example, with video color ringback the subscriber can replace the standard ring tone that the caller hears with a personalized video message. Imagine seeing and hearing the person you are calling running toward the phone yelling, "I'm coming...." IP Multimedia Subsystem (IMS) from the Third Generation Partnership Project (3GPP) defines media servers as part of its next-generation network architecture for multimedia solutions. Similarly, in the enterprise, multimedia solutions are being investigated in conjunction with IP technology as a means of reducing operating expenses while improving worker productivity and customer satisfaction.

Intel NetStructure Host Media Processing Software simplifies the development of multimedia telecommunication applications: supporting new capabilities such as video is simple and straightforward.

The move toward IP-centric solutions can represent significant changes in the way media servers and applications are modeled. Intel is committed to protecting our customers' investments. Existing applications that run on Application Programming Interfaces (APIs) delivered with Intel NetStructure Host Media Processing Software

®™ Intel, Intel NetStructure, and VTune are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

can easily migrate from a traditional PSTN environment to an IP-based next-generation network. At the same time, Intel embraces emerging and evolving open industry standards in architectures, protocols, and interfaces.

The powerful combination of Intel Architecture processors and chipsets, innovative platform technologies such as dual core, Hyper-Threading Technology¹ (HT Technology) and Intel® EM64T²; and software technologies such as Intel IPP and Intel NetStructure Host Media Processing Software, will meet the demanding performance requirements of media signal-processing applications. By supporting these technologies with a wealth of available development, test, and performance-tuning tools, Intel offers an exceptional signal-processing platform for building next-generation media servers with the lowest total cost of ownership.

As new and innovative technologies such as multi-core and Intel Virtualization Technology become available, Intel platforms will provide more end-user value through new features and improved performance.

These Intel products and technologies are used to deploy services in a media service network. A media service network is a network through which media services are provided to an end user. We start our discussion with a description of the components that make up a media service network using three representative network configurations. We then take a top-down approach to discussing the Intel technologies used to build a next-generation media server. We start by taking a look at Intel NetStructure Host Media Processing Software, and how it makes developing interactive multimedia applications straightforward and cost effective. We walk through the steps involved in developing a simple voice application

using Intel NetStructure Host Media Processing Software APIs. We then examine what is involved in adding video capabilities to the application, demonstrating how developers can add features to existing applications while retaining their investment in existing code.

At the heart of an Intel-based media server is the Intel Architecture processors, chipsets, and Intel IPP. In the subsequent sections we show how these technologies can offer outstanding performance of media processing algorithms such as codecs. In the last sections we look at how developer tools like the Intel C++ Compiler and VTune Performance Analyzer can be used to produce high-performance application code.

TAXONOMY OF A MEDIA SERVICE NETWORK

A media service network is a network through which media services are provided to an end user. The services are implemented by applications controlling media resource functions on a media server. In this section we discuss the components that make up a media service network, illustrate three representative communication network configurations into which media servers are deployed, and describe the interfaces used by media service applications.

There are a number of architectures in use today that define the composition of a media service network. Some are specified by industry standards bodies and others are proprietary. Most can be generalized to a simple model where the media server interoperates with a number of other components to form the complete solution.

Components of a Media Service Network

We will first define the basic components of a media service network: end-user terminals, application server, media server, media store, and gateways.

End users access media services through the public network from a local device we will call a terminal. Examples of terminals are a plain old telephone (POT), a video-enabled cell phone, and a multimedia PC.

The application logic that realizes the media service is hosted by an application server. The application server provides the runtime execution environment for the application.

Media processing such as transcoding and video image processing (e.g., text overlay, resizing) are performed by media processing resources on the media server.

The media store is responsible for the storage and retrieval of the media (i.e., to/from disk).

¹ Hyper-Threading Technology requires a computer system with an Intel® Pentium® 4 processor supporting HT Technology and a HT Technology enabled chipset, BIOS and operating system. Performance will vary depending on the specific hardware and software you use. See www.intel.com/homepage/land/hyperthreading_more.htm for additional information.

² Intel® EM64T requires a computer system with a processor, chipset, BIOS, operating system, device drivers and applications enabled for Intel EM64T. Processor will not operate (including 32-bit operation) without an Intel EM64T-enabled BIOS. Performance will vary depending on your hardware and software configurations. See <http://developer.intel.com/technology/64bitextensions/> for more information including details on which processors support Intel EM64T or consult with your system vendor for more information.

protocols are in use), the media server will have to interoperate with gateways that are responsible for translating between the different network protocols. Gateways are typically classified as signaling gateways that translate one signaling protocol to another and media gateways that translate from one media format to another. We explain signaling in the following section.

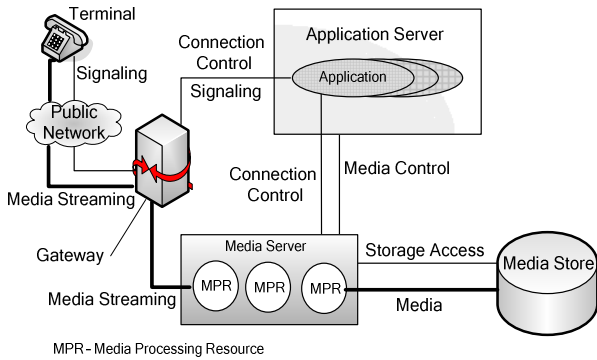


Figure 1: Basic media service network architecture

Control and Data Plane

Components in a media service network can be thought of as interoperating in two planes. The data plane, where the actual media traffic flows, is depicted as the thick lines in Figure 1. The control plane, shown as the thin lines, is where control and signaling messages flow. Signaling messages are used to establish and manage the media session between the end-user terminal and the media server. The session contains the state and any other context associated with the connection. The terms “session” and “call,” as in a telephone call, are used interchangeably. Signaling typically originates from the end-user terminal and the application, potentially being translated by a signaling gateway. As mentioned, the media traffic flows along the data plane. Routing and flow of the media are controlled using a connection control protocol in the control plane. Connection control typically originates from the application and is directed toward the terminal or gateway and media server.

The application logic controls the media server through a media control API, discussed in more detail later.

Composing a Media Server

The media server is the architectural element responsible for transmitting media content to an end user over a communications network. The content may be in the form of video, audio, text, or a combination of the three. The media server often contains digital signal processing resources to process or transform the media prior to transmitting it to the user (e.g., gain and speed control of an audio stream). In addition to transmitting media content

to the end user, a media server may also be capable of accepting control input from the end user’s terminal. This input may be used by the media service application, making it interactive.

Figure 1 illustrates the logical components of a media service network. In this diagram the application server, media server, media store, and gateway are shown as separate entities. This is a logical organization and does not necessarily imply a physical decomposition. In practice, depending on the individual system requirements, any or all of these elements can be combined within a single node. In the following sections we will often simplify the diagrams by combining the application server, media server, and media store elements into a single node.

CIRCUIT-SWITCHED NETWORK

The traditional circuit-switched telephone network, also known as the Public Switch Telephone Network (PSTN), is a connection-oriented communications system where dedicated “circuits” are used to transport the media between the end user and the media processing resources. The PSTN consists of a collection of switches, as well as connections between switches and end-user terminals (a.k.a. telephones).

The establishment and routing of circuits between the end-user devices and the central office switches are controlled via signaling. Traditional phone devices use in-band signaling, where the signaling information is carried over the same circuit as the media. An example familiar to everyone is the generation of DTMF tones from the touch-tone pad of a common telephone used to identify an end user by number.

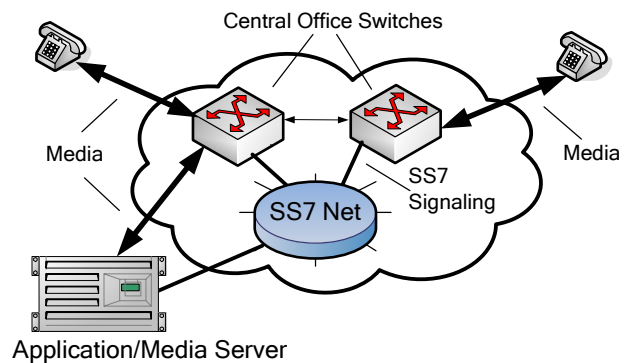


Figure 2: Traditional circuit-switched network

Signaling System 7 (SS7) [1] is an international standard signaling system that provides an efficient and protected method for routing circuits through the network. SS7 is an out-of-band signaling method that reserves dedicated

circuits called signaling links for transmitting the signaling information. The Integrated Services Digital Network (ISDN) [2] also defines an out-of-band signaling protocol. In either case the control, or signaling, information is separate from the media.

Media servers can use in-band signaling to control the connections between end-user devices and media resources. However, out-of-band methods such as SS7 have become essential for many media service applications and have become the method of choice for media servers deployed today (see Figure 2).

Intel has a number of technologies that allow media servers to connect to the PSTN circuit-switch network using in-band or out-of-band signaling techniques.

PACKET-SWITCHED NETWORK

Unlike circuit switching, which relies on dedicated point-to-point connections to transmit contiguous streams of media, a packet-switched network breaks up the media stream into small message packets. Each packet contains address information specifying the desired destination for the packet. Because packets are addressed, no pre-established communication path or reserved “circuit” is required for a packet network. In contrast to a circuit-switched network, bandwidth in a packet network is used as needed to send packets. A quiescent channel need not produce load on the network.

As packets traverse the packet-switched network, they do not necessarily follow the same path to the destination. Network traffic conditions or outages can result in packets being dynamically routed through different paths. As a result, packets may take different amounts of time to reach the destination, they may arrive in a different order from the one in which they were sent, or they may be lost altogether. It is the responsibility of the destination device to deal with packet order, latency, jitter, and loss. Once they are reassembled, the destination device can render the media in real time. Packet loss, network latencies, and packet ordering represent some of the biggest challenges to providing reliable, high-quality, real-time communications over a packet network.

Several protocols have been designed for packet-based networks, the most popular being the ubiquitous Internet Protocol (IP). IP, which specifies the addressing scheme and packet format, was originally designed for data communications between dissimilar computers. It is a connectionless protocol and not inherently reliable. Numerous additional protocols have been designed to run on top of IP specifically to address the needs of real-time voice and video communications.

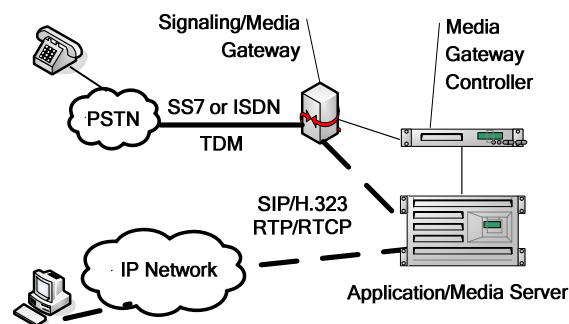


Figure 3: IP network with PSTN gateway

Two predominant standards have emerged in the IP signaling domain: H.323 [3] and SIP [4]. H.323 is an International Telecommunications Union (ITU) standard for the transmission of voice, video, and data over an IP network. An umbrella specification that consists of a suite of protocols and standards, H.323 defines a complete framework for multimedia communications. It specifies detailed protocols, messages, and state machines. H.323 has its roots in the traditional telephone network and attempts to address a wide range of problems including basic call states, supplementary services such as call forwarding and call waiting, Quality of Service (QoS), mobility (users moving from one address to another), and security.

SIP is a protocol defined by the Internet Engineering Task Force (IETF) that has begun to supplant H.323 in popularity. Unlike H.323, which attempts to address specific functionality such as basic and supplementary multimedia services, SIP defines a protocol that supports a generic session model upon which systems can be built. The SIP specification (IETF RFC 3261) defines a small set of messages that addresses location services, session creation and termination, and session parameter passing. It is designed to support a wide range of multimedia applications.

Although SIP and H.323 seem to have approached the problem of multimedia communications from different angles, they both define a number of architectural elements that enable location services, authentication, mobility, and interoperability with the circuit-switched PSTN. At a high level, SIP and H.323 are similar in terms of functional decomposition. In addition, both SIP and H.323 use RTP/RTCP [5] (IETF RFC 3550) as the media streaming protocol over IP.

For interoperability with the PSTN, SIP and H.323 define an architectural element called a gateway. The gateway is divided into two functional components. The signaling gateway (SG) component converts between PSTN control signals and the appropriate SIP/H.323 messages. The media gateway component (MG) converts between circuit

and packet media. A MG controller is the entity that controls the MG (see Figure 3).

In architecture diagrams, the MG, SG, MG controller, application server, and media server are often shown as individual devices. Depending on the system requirements, these devices may be implemented on separate nodes or be physically combined within a single server.

3GPP and IMS

The Third Generation Partnership Project (3GPP) developed the IP Multimedia Subsystem (IMS) [6]. IMS is an example of a next-generation communications network architecture. It enables service providers to deploy new IP-based, multimedia communication services over both the fixed wireline and mobile telecommunications networks.

With IMS, services can be provided over any IP network (GPRS, WLAN, etc.). The IMS infrastructure is IP based, using standard SIP/IP between the core network elements. Originally designed for the mobile network, IMS can provide IP-based services to external circuit-switched networks as well as external IP networks.

The IMS architecture defines functional entities falling into the six main categories listed in Table 1. The IMS entities collectively address interoperability with other networks (e.g., circuit-switched and radio access networks), security, roaming, policy control, billing, and service deployment.

Figure 4 shows a simplified view of the IMS architectural elements that cover session management and call routing, security and policy management, network interoperability, security, and services.

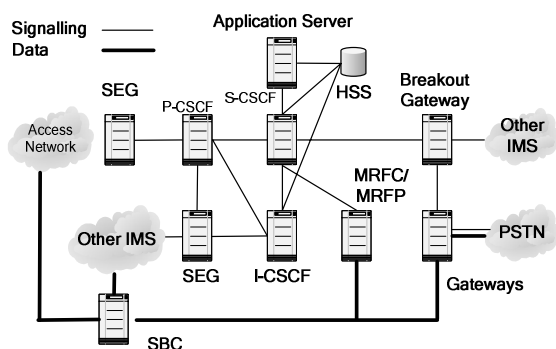


Figure 4: Simplified IMS architecture

Table 1: IMS functional categories

Session and routing	Call Session Control Functions (CSCF)
Databases	Home Subscriber Server (HSS), Subscription Locator Function (SLF)
Interoperation	Breakout Gateway Control Function (BGCF), Media Gateway Control Function (MGCF), IMS Media Gateway (IMS-MGW), Signaling Gateway Function (SGF)
Services	Application Server (AS), Multimedia Resource Function Control (MRFC), Multimedia Resource Function Processor (MRFP)
Support	Policy Decision Function (PDF), Security Gateway (SEG), Topology Hiding Inter-network Gateway (THIG)
Charging	Charging Collection Function (CCF)

The IMS elements most relevant to this paper are the Multimedia Resource Function Controller (MRFC), Multimedia Resource Function Processor (MRFP), and the Applications Server (AS). The MRFC is the element responsible for taking SIP requests from the AS and translating them to messages that control the media processing resources residing in the MRFP. The MRFP is where the actual media processing resources reside.

While the IMS specifications separate the AS, the MRFC, and the MRFP, implementations can combine one or more of these elements into a single node, as noted earlier. In fact, it is widely expected that the MRFC and MRFP elements will typically be deployed as a single unit.

In Figure 5 we show a combined AS, MRFC, and MRFP and collapse the other IMS elements. This diagram shows many of the same functional elements as shown in Figure 3: an IP-based Media Server interoperating with the circuit-switched network via a media and signaling gateway combination. In fact, many of the same Intel components may be used to build both systems.

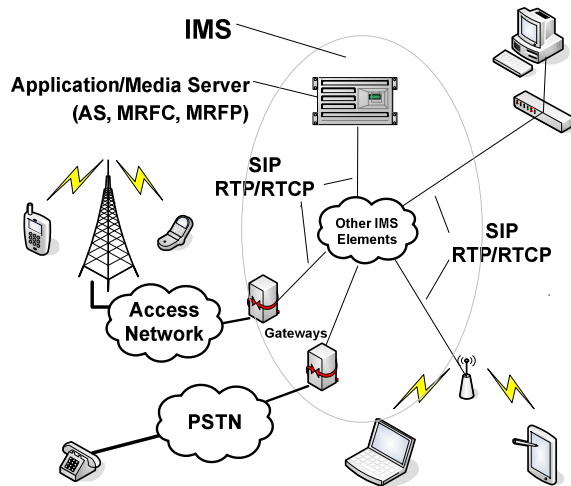


Figure 5: Media server view of the IMS architecture

APPLICATION PROGRAMMING INTERFACES

The components shown in Figure 1 interoperate along the control plane through a collection of interfaces. Of particular interest to the media service application developer are the interfaces between the application server and the media server. These APIs can be divided into three categories: signaling, connection control, and media control.

As was previously mentioned, signaling messages are used to establish and manage the media session between the end-user terminal and the media server. The primary signaling interfaces for next-generation applications are SIP for IP and IMS and Signaling System 7 (SS7). Intel NetStructure SS7 Boards provide applications with basic and advanced signaling capabilities in both wireline and wireless networks. For SIP signaling, applications can use either SIP protocol stacks from third parties or Intel's own APIs. Intel NetStructure Host Media Processing Software supports standard SIP protocols through the Global Call API. This is discussed in more detail below.

Connection control APIs allow the application to control the routing and flow of media along the data plane. There are numerous methods for controlling the media data flow. For the IP-centric next-generation media server, the most common method involves directing Real-Time Protocol (RTP) streams using SIP. The Media Gateway Control Protocol (MGCP) [7], as defined by the Internet Engineering Task Force (IETF), allows applications to control the routing and flow of media through a gateway.

The category of interface most relevant to this discussion, media control, covers a wide variety of media functions. It includes basic voice messaging such as the ability to play

announcements and record messages, as well as video functions. Media control also covers the ability to accept control input from the end user. This input may be in the form of in-band tones such as DTMF or speech, as well as out-of-band messages such as those defined by RFC-2833 [8]. Intel supports a variety of APIs for media control. In the following paragraphs we discuss a few notable ones.

Voice Extensible Markup Language (VoiceXML) [9] is a standard XML-based language that enables voice/speech interactions with Web pages. VoiceXML is defined by the World Wide Web Consortium (W3C). Application logic is expressed in the form of VoiceXML documents that are processed by a VoiceXML interpreter (a.k.a. browser) which, in turn, drives the media server. VoiceXML is popular for Interactive Voice Response (IVR) applications because it defines a simple yet powerful high-level programming language.

Two additional media control interfaces attracting interest for next-generation media servers are the Media Sessions Markup Language (MSML) [10] and Media Objects Markup Language (MOML) [11]. MSML and MOML are Internet drafts submitted to the SIPPING work group of the IETF. MSML/MOML are XML-based interfaces that, while capable of standing on their own, are designed to complement one another. In contrast to VoiceXML, which presents a high-level language, MSML/MOML defines a framework for describing a wide range of media functions in an extensible manner. Besides offering rich semantics, MSML/MOML is attractive for use by next-generation media servers because it is designed to be compatible with SIP. The downside of MSML and MOML is that they are not yet recognized as standards and are still relatively immature.

Another notable media control interface is Intel's R4 "C" language APIs. The R4 APIs have been a staple of computer telephony applications programming for over 10 years. They are supported on Intel media processing technology boards as well as Intel NetStructure Host Media Processing Software.

All of the interfaces described above can be used with Intel-based media servers. Some, such as VoiceXML, can be acquired through third parties, while others are directly available from Intel. In the following sections we discuss Intel NetStructure Host Media Processing Software and the R4 APIs. We demonstrate how to implement a simple media service application that will run on media servers built using HMP software.

INTEL NETSTRUCTURE[®] HOST MEDIA PROCESSING SOFTWARE

Intel NetStructure Host Media Processing Software is an Intel architecture-based technology for processing media.

It is an ideal platform for implementing a media server, whether in the rigorous environment of an IMS network or in less standardized applications.

About Intel NetStructure® Host Media Processing

Intel NetStructure Host Media Processing Software runs on a general-purpose computer running Windows* or Linux*. The HMP software processes media on the Intel CPU and interfaces with external entities using standard IP protocols through the computer's Network Interface Controller (NIC).

Intel NetStructure Host Media Processing Software supports all common media server functions, including T.38 fax, audio/video play, audio/video record, audio conferencing, audio transcoding, IP call control, and also provides hooks for speech recognition and text-to-speech. As of this writing, HMP software from Intel runs up to 400 concurrent channels (i.e., bi-directional audio sessions) on a single computer. Additionally the HMP software interfaces with PSTNs, PBXs, and digital stations through telephony interface cards [13,28]. Table 2 lists the features in more detail.

Table 2: Intel NetStructure Host Media Processing Software feature tables

Network Interface	
	IP over Ethernet, Traditional Telephony interface cards

IP Call Control	
Protocol	H.323, SIP
Integration with third-party call and connection control stacks	Provided via R4 IPML

IP Streaming	
Protocol	RTP (RFC 3550)
Audio formats	G.711 A-law, μ -law 8-bit 8K G.723.1 (5.3 KHz and 6.3 KHz) G.729a, G.729b
Video formats	H.263 profile 0, level 30 (RFC 2190)
QoS	Alarms Frames per packet control Packet loss reduction RTP/RTCP timeouts
Tone generation and detection	RFC 2833 H.245 User Input Indication
Media control over RTP	Programmatic control of inbound RTP stream gain and output RTP stream volume

Voice Processing Features	
Features supported	Play, record, and tone generation and detection
Play	Volume control and index play
Record	AGC
Audio file formats for play/record	OKI ADPCM 24 Kbps, 32 Kbps G.711 A-law, μ -law @ 48 Kbps, 64 Kbps All of the above in WAV format Linear PCM 8b @ 11 Khz (Wave format only) Linear PCM 64 Kbps Linear PCM 128 Kbps G.726

Conferencing Features	
Total parties/server	240
Advanced Features	N-way summing Coach/pupil mode DTMF detection DTMF clamping Active talker notification

User Input Features	
In-Band	In-Band DTMF generation and detection User-defined global tone generation and detection (GTG, GTD)
Out-of-Band	RFC 2833 RTP Messages H.245 User Input Indicator

* Other names and brands may be claimed as the property of others.

Video Processing Features	
Play	Playback of synchronized voice and video
Record	Stores synchronized voice and video to file
Video format	H.263 (profile 0 level 30)
Picture Sizes	CIF, QCIF, and sub-QCIF
Video file formats	Dual proprietary file formats Audio file (.pcm): Linear PCM 16b 8K Video file (.vid): H.263 bit-stream data
Offline conversion tool	Convert AVI Type-2 (DVSD or DV25) files (PAL or NTSC) to proprietary format Convert proprietary format to 3GP Release-4 file format (.3gp)

The value of Intel NetStructure Host Media Processing Software lies largely in its reduced Total Cost of Ownership (TCO), often 50% less than traditional DSP-based solutions [12]. Because HMP software does not require specialized hardware, it is able to take advantage of economies of scale, resulting in lower hardware costs and some free performance improvements. As Intel develops better CPUs, HMP software performs better. It rides the Intel technology wave of silicon and software advancements: dual-core processors, HT Technology, Intel IPP, NICs, and VTune Performance Analyzer among others.

Because Intel NetStructure Host Media Processing Software runs on a general-purpose computer, development, deployment, and maintenance costs are reduced as well. Table 3 and Table 4 illustrate these points. Table 3 describes a hardware-based solution and a solution based on Intel NetStructure Host Media Processing Software. Table 4 analyzes the costs of the two solutions [12]:

Table 3: Hardware and Intel NetStructure Host Media Processing Software Solutions

Expense	Hardware-Based Detail	HMP Software-Based Detail
Basic Building Blocks	Intel NetStructure IPT2400 + Intel NetStructure DMV2400AC PCI	Resources: 240 Voice, 240 RTP, 60 CSP + server with Intel® Xeon® processor (3.2 GHz -- \$3,500)
Development System	10% of deployed ports	5% due to NFR software pricing
Inventory	25% of volume	None
Shipping	Ship boards from distributor	Electronic distribution
Installation and Configuration	4 hrs @ \$100 per hour	1 hr @ \$100 per hour
Spares	30% of boards	Instant emergency license
Field Upgrade + 25% Capacity	Buy Intel NetStructure® DMIP301 boards	Add 60 Ports + additional server

Table 4: Analysis of both solutions

Expense	HW-Based Cost	HMP Software-Based Cost	Savings
Basic Building Blocks	\$21,260	\$11,300	\$9,960
Development System	\$2126	\$650	\$1,476
Inventory	\$1,875	\$0	\$1,875
Shipping	\$100	\$0	\$100
Installation and Configuration	\$300	\$100	\$200
Spares	\$2,250	\$0	\$2,250
Field Upgrade + 25% Capacity	\$6,378	\$5,456	\$922
Total	\$34,289	\$17,506	\$16,783 (49%)

® Intel and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Intel APIs

Intel NetStructure Host Media Processing Software supports the R4 and Global Call APIs, which have been staples of computer telephony for over a decade. Both are high-level “C” APIs that make implementing media servers easier for the application developer. The APIs are the same APIs that run on Intel media processing technology boards, making the transition from board-based solutions to HMP software-based solutions nearly seamless. Connection control is internal if it is internal to the HMP software, and external if it connects the HMP software to an external entity. Global Call is the signaling and external connection control API, and R4 is the media control and internal connection control API.

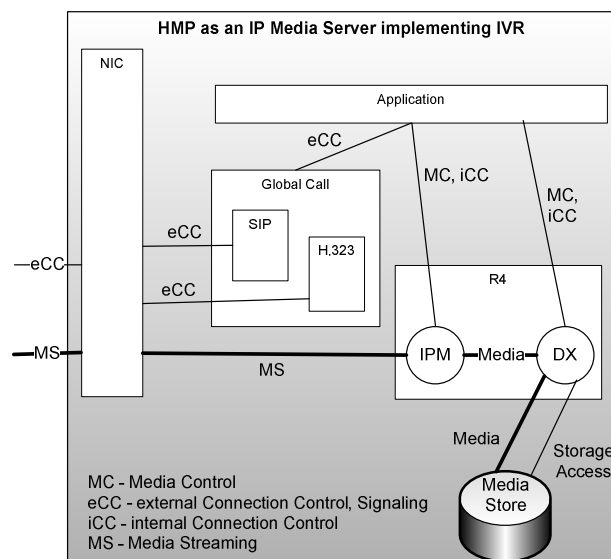


Figure 6: HMP software-based IP media server

Global Call abstracts the H.323 and SIP signaling stacks and can be configured to automatically establish IP calls or let the application manage the connections. After setting up calls with Global Call, an application performs the media control functionality through the R4 API. The R4 API is divided into several families [13-17]:

Table 5: R4 API families

API Support	
Voice processing	R4 voice (dx_)
Multimedia processing	R4 multimedia (mm_)
Connections	R4 routing (sc_ and dev_)
Conferencing	R4 conferencing (dcb_)
Fax	R4 fax (fx_)
Continuous speech processing	R4 EC (ec_)
IP media (QoS, etc.)	R4 IPML (ipm_)
Framework - Event reporting, device enumeration, and other related functionality	R4 SRL (sr_)

R4 abstracts media control functionalities into R4 devices. Example devices are the DX device, which performs audio play and record, the MM (multimedia) device, which performs multimedia play and record, and the IPM (IP media) device, which performs RTP functionality. Applications use these devices not only to perform actions, but also to receive events and make connections. All devices support duplex media streaming. Some of these functions are described in more detail below.

R4 includes two more abstractions that are worth an introduction: Run-Time Controls (RTCs) and termination conditions. RTCs allow applications to arm devices to respond to events without application interaction. For instance, the application can arm the voice device to increase volume on detection of a digit. In essence, RTCs are a means to meet real-time performance requirements.

A termination condition is a condition associated with an action, such as record, that should terminate upon occurrence of the condition. Setting up termination conditions is a common use of RTCs. A common termination condition is silence duration, which might trigger a voice device to terminate recording of a media stream.

R4 and Global Call both support a variety of programming models, broadly categorized as synchronous and asynchronous. Application developers are encouraged to use the asynchronous programming model for superior performance.

In the asynchronous programming model, the action initiated by a function call, such as playing, does not complete when the function returns. Instead, HMP software sends an asynchronous completion event to the application once the action has completed. By doing this, an application does not have to wait for one action to complete before starting another, and the application does

not have to spawn a thread for every channel in the system.

A robust technique for asynchronous programming is to program an application state model. In this state model, events, including completion events, incite the state transitions. Hence, an application event handler checks the state of the device on which the event is received in order to determine the next action.

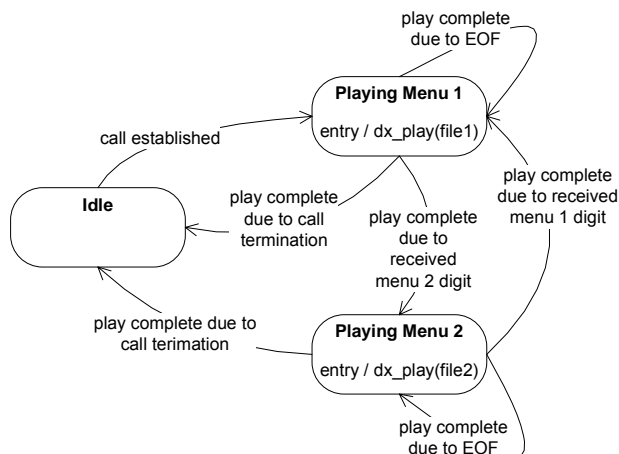


Figure 7: Simple IVR application state diagram

Figure 7 is an example of an Interactive Voice Response (IVR) application state diagram. The “entry/dx_play(…)” notation indicates that the application performs a dx_play() on entry into that state (more on dx_play() later). By using an application state model, the vast majority of the business rules for an application is built into the model, making the application not only robust but also flexible.

Building an HMP Software Application

This section applies our Intel NetStructure Host Media Processing Software and API knowledge towards building an IVR and Video Portal application. This section does not provide an exhaustive discussion or code listing, but does briefly address all of the key pieces. Finally, this section uses pseudo-code with function names identical to R4 and Global Call function names.

An IVR application has four main pieces: establishing a call, connecting an IPM device to a local voice device, connecting an IPM device to a remote endpoint, and controlling the media in the call through the voice device. We will tackle these pieces in that order. The application design is consistent with Figure 7.

Global Call provides the APIs for making and receiving calls [18].

```
gc_MakeCall() // to make a call
gc_WaitCall() // to wait for a call
```

After these APIs complete, via asynchronous completion events, a variety of other Global Call APIs are used to complete the external connection setup. Before we connect an IPM device to a remote endpoint, we need to connect it to a local voice device using an internal connection control.

```
dev_Connect(IPM_device, voice_device, duplex)
```

And eventually, we need to configure our IPM device:

```
ipm_StartMedia(IPM_device, audio_codec, UDP_port,
               IP_address, ...)
```

The codec, UDP port, and IP address information are available from Global Call. At this point we use Global Call to inform the far side that the connection is up and running.

Now we are ready to perform IVR functionality, for example, play a file from the voice device.

```
dx_play(DX_device, file);
```

That completes the critical steps for building an IVR application. Each step above is probably three to five steps for most applications. A typical IVR application will typically perform multiple audio plays, depending upon input received from the user. Furthermore, the application needs to be constructed asynchronously, as described above, but the concepts and steps are intuitive.

To change our IVR application into a video portal, we need to add video functionality. In HMP software from Intel, video is handled through the multimedia libraries (mm_), which differ from the voice-only libraries (dx_). Therefore, we need to change the dx_ calls to mm_ calls, but we retain the same state machine. For example:

```
dev_Connect(IPM_device, DX_device, duplex) →
dev_Connect(IPM_device, MM device, duplex)

dx_play(DX_device, file) →
mm_Play(MM_device, file)
```

Most dx_ calls have analogues in the mm_ domain. The parameters are arranged a little differently, but the style is the same, so switching from dx_ to mm_ is a natural and necessary process, as the dx_ calls could not support video requirements.

In addition, we must update our Global Call code and ipm_StartMedia to include video stream configuration:

ipm_StartMedia(IPM_device, IPM_device, audio codec, video codec, UDP_ports, IP_address, ...)

Again, Global Call provides the codec and port information.

That completes the transformation of our application from IVR to Video Portal. As we have seen, Intel NetStructure Host Media Processing Software is a powerful tool for implementing media applications, including media servers. The HMP software reduces application development time and TCO. Now we look at some of the Intel technologies on which Intel NetStructure Host Media Processing Software is built.

INTEL ARCHITECTURE FOR SIGNAL PROCESSING APPLICATIONS

Intel offers an exceptional signal-processing platform for building next-generation media servers with the lowest total cost of ownership.

Intel® Integrated Performance Primitives

Intel IPP are a highly optimized suite of libraries for audio, video, imaging, cryptography, speech recognition, and signal processing functions [22]. To maximize performance, Intel IPP use advanced performance-tuning techniques such as pre-fetching and cache-blocking, avoiding data and trace-cache misses as well as branch mis-predictions. Intel IPP exploit instruction set architectures like Intel Wireless MMX™ technology, SIMD Extensions (SSE), and HT Technology.

The Intel IPP libraries can be linked to the application as dynamically loadable modules, which make the applications platform independent. The libraries automatically detect the underlying processor platform at run-time and execute the function implemented for that particular platform. Table 6 lists some of the Intel IPP that are related to Media Processing. Intel NetStructure Host Media Processing Software uses the Intel IPP for high performance. Figure 8 demonstrates the performance gain offered by IPP over Compiled C code [22]. In Figure 9 we show IPP encoder performance for H.263 profile 0, QCIF at 15 frames per second.

Table 6: Intel IPP

Media Processing Function	Intel IPP Available
Audio/Video Play and Record, Audio and Video Transcoding for multi-media connections over IP.	Audio Codecs – G.711, G.723.1, G.729, G.722, G.722.1, G.726, G.728, GSM-AMR, MP3, AAC, AC3. Video Codecs – H.263, H.264, MPEG4
AGC, Tone Detection, Tone Generation, VAD, Conferencing Summer	Signal Processing and Vector Math Library – Digital filtering (Adaptive, FIR, IIR), Fourier Transforms, Signal Generation
Speech Recognition	Audio Processing – Acoustic Echo Cancellation, Noise Reduction, VAD, Feature Extraction. Speech Processing – Pitch Detection, Speech Resampling
Echo Cancellation	G.168-2000 compliant Echo Canceller
Secure RTP	Symmetric Cryptography (DES, 3DES) Hash Algorithm Data Authentication (DES, 3DES)

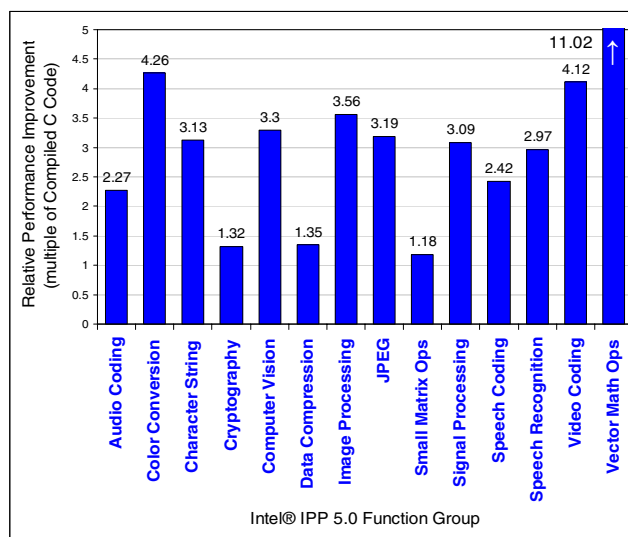


Figure 8: Intel IPP 5.0 average performance gain over compiled C code³[22]

™ Intel Wireless MMX is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

³ All code running on a PC with an Intel® Xeon® processor supporting Hyper-Threading Technology, 3.6 GHz, 1 MByte L2 cache and 2 GB RAM using Microsoft Windows* XP.

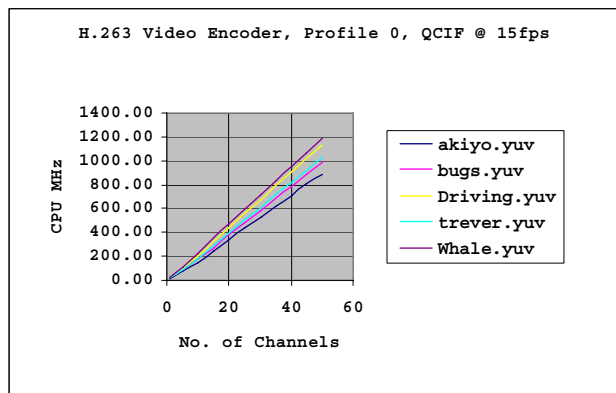


Figure 9: H.263 IPP encoder performance⁴

Figure 10 shows performance data for advanced video processing functions such as text overlay and tiling for video conferencing with video streams of QCIF @ 15fps.

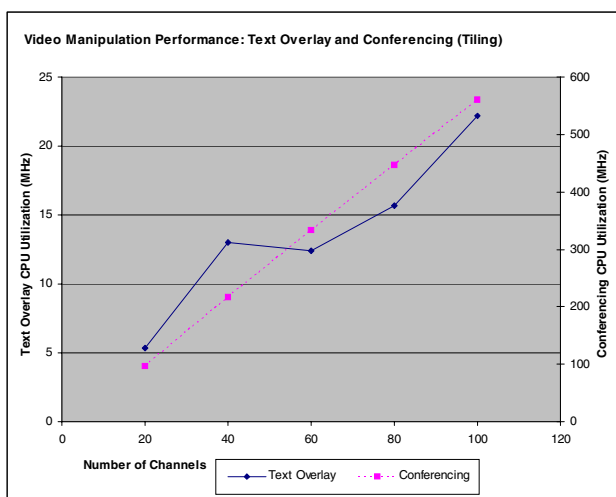


Figure 10: Video manipulation performance⁴

Processors and Chipsets

Intel offers a wide variety of processors and chipsets to address the desktop, server, and mobile market segments. Table 7 lists a sampling of the processors available today, addressing each of the market segments. These processors offer a combination of features and technologies such as HT Technology, dual-core processors, increased amounts of L2 and L3 cache, increased FSB speeds, Intel Extended Memory 64 Technology (Intel EM64T), Streaming SIMD Extensions 3 (SSE3), and multi-processor platforms (MP). The hardware platform combined with the software tools already mentioned makes Intel Architecture Processors a

⁴ All code running on a PC with an Intel® Pentium® M processor supporting Hyper-Threading Technology, 2.1 GHz, 2 MByte L2 cache and 1 GB RAM using Microsoft Windows* XP.

very compelling media processing hardware platform from both a functionality and a density standpoint. The impact of HT Technology, dual-core processors, multi-processors, cache sizes, Intel C++ Compiler, VTune Performance Analyzer, and Intel IPP on media processing is discussed in detail. All data presented in this paper have been generated on a specific platform configuration that may vary for each set of results. All numbers presented are representative of the platform configuration for only that given set of data.

Table 7: Sampling of Intel processors

	Pentium® Processor Extreme Edition 840 ⁵	Pentium® 4 Processor Extreme Edition	Intel® Xeon® Processor Dual-Core	Intel® Xeon® Processor MP
System Type	UP	UP	DP	MP
L2 Cache	2x1MB	512 KB	2x2MB	1MB
L3 Cache	N/A	2 MB	N/A	8 MB
Clock Speed	3.20 GHz	3.46 GHz	2.80 GHz	3.33 GHz
FSB (MHz)	800 MHz	1066 MHz	800 MHz	667 MHz
dual-core	✓		✓	
Intel® EM64T	✓		✓	✓
HT	✓	✓	✓	✓
Execute Disable Bit	✓		✓	✓
SSE3			✓	
EIST			✓	
DBS			✓	✓
Chipset	Intel® 955X Express	Intel® 925XE	Intel® E7520	Intel® E8500
Memory Type	Dual-Channel DDR2	Dual Channel DDR2 400/533 (CL3)	Dual Channel DDR, DDR2	Quad Channel DDR, DDR2

⁵ Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See http://www.intel.com/products/processor_number for details.

Table 8: Overview of Intel processor technologies

System Type	Number of processor sockets in a platform or computer system. Uni-processor means that only one processor can be present. Dual-processor systems allow for up to two processors, and multi-processor systems allow for more than two processors.
L2 Cache	Ultra-fast memory that buffers information being transferred between the processor and the slower RAM in an attempt to speed these types of transfers.
L3 Cache	Size of 3 rd -level cache, typically larger than L2. L3 Cache is ultra-fast memory that buffers information being transferred between the processor and the slower RAM in an attempt to speed these types of transfers. Integrated L3 cache provides a faster path to large data sets stored in cache on the processor.
FSB (Front Side Bus)	Bus connecting the processor to the main external memory. The frequency shown in the table represents the operating frequency of the bus.
Intel [®] EM64T	Intel Extended Memory 64-bit technology enables 64-bit computing.
Execute Disable Bit	Allows the processor to classify areas in memory where application code can execute and where it cannot.
SSE3	Internet Streaming SIMD (Single Instruction Multiple Data) Extensions are instructions that reduce the overall number of instructions required to execute a particular program task. 3 refers to the 3 rd iteration of these enhanced instructions.
EIST	Enhanced Intel SpeedStep [®] technology enables a system to dynamically adjust processor voltage and core frequency.
DBS	Demand-Based Switching uses EIST to dynamically lower the processor voltage and core frequency based on processor utilization.

Hyper-Threading Technology

HT Technology boosts computing performance by enabling a single processor to function as two “virtual” processors by executing two threads in parallel, allowing software to multi-task more effectively [23]. It provides

[®] Intel SpeedStep is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

thread-level parallelism on a single processor, resulting in more efficient use of processor resources, higher processing throughput, and improved performance of multi-threaded applications. Multi-threaded software divides its workloads into processes and threads that can be independently scheduled and dispatched by the operating system. In a multiprocessor system, those threads execute on different processors, whereas in a single processor that is HT Technology enabled, the threads execute in parallel on a single processor. HT Technology uses the idle cycles in a processor core such as stalls due to memory access, to enable parallel execution of another thread such as arithmetic computation that utilizes internal registers.

Prior to HT Technology, media-processing functions such as playing audio (disk or memory I/O intensive) and decoding a g.729a VoIP packet would complete in order of priority. If the thread executing audio play happened to be higher priority, the g.729a decode thread would not be able to take full advantage of the stalled processor cycles in the audio play thread due to memory or disk I/O. With HT Technology enabled, the g.729a decode thread can execute in parallel during the stalled cycles by doing any arithmetic operations that do not require memory I/O.

Figure 11 and Figure 12 show the performance due to HT when compared with single- and dual-processor systems for a specific media processing application: the SIP-to-SIP connection is configured for G.711 and G.729a RTP connection. For a G.711 RTP connection, HT Technology results in close to a 50% reduction in CPU utilization—equivalent to executing on a dual-processor configuration. For a G.729a RTP connection, HT Technology results in a 15-20% reduction in CPU utilization, whereas a dual-processor configuration without HT Technology results in a 30-40% reduction in CPU utilization. In both test application configurations, the platform comes close to achieving the theoretical maximum of four times the performance increase when compared to a dual-processor which is two times the performance speed, with HT Technology-enabled (x2) platform with a single-processor platform without HT Technology. The difference can be attributed to more frequent conflicts between the various channel threads using shared resources such as the execution engine, external memory access, and cache. If there is contention in usage of these resources by two different threads, the operations are serialized. A G.711 encode/decode algorithm has a small code and data footprint that will most likely not result in any cache misses. The G.729 algorithm has a much larger data footprint and data structure per channel and there is likely to be more cache thrashing and contention for external memory access.

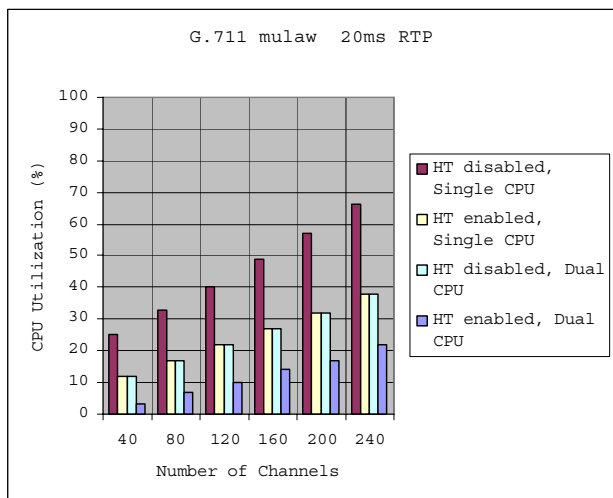


Figure 11: HT technology and dual processor performance G.711⁶

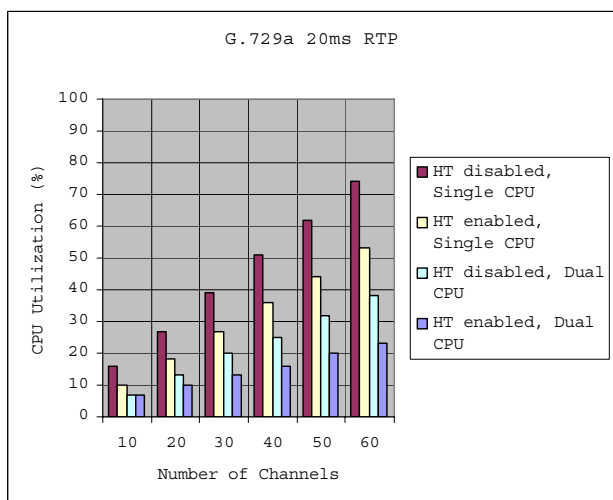


Figure 12: HT technology and dual processor performance G.729⁶

Dual-Core Processors

Dual-core processors provide increased computing performance by combining two full processing cores into a single processor. These processors are well suited to multi-tasking environments because there are two complete execution cores instead of one, each with an independent interface to the FSB. Unlike HT Technology, dual-core processors offer complete parallelism of threads through independent execution units with no contention between threads in accessing resources such as the execution engine, FSB, and cache. Dual-core processors

⁶ All code running on a PC with dual Intel® Xeon® processors supporting Hyper-Threading Technology, 3.2 GHz, 512 KByte L2 cache and 1 GB RAM using RedHat® Enterprise Linux.

also open up a multitude of opportunities for executing completely independent tasks on each processor such as gaming on one core while running a virus scan on the other. The dual-processor results in Figure 12 indicate the potential performance boost due to dual-core processors.

Cache-Intel Architecture Processors

Cache Intel Architecture processors are offering increasing amounts of L2 and in some cases L3 cache, with significant potential benefits for cache-friendly applications. Media-processing applications benefit from increasing levels of cache as CPU-intensive signal-processing algorithms execute more quickly, with fewer processor stalls associated with fetching instructions and data from memory. A typical signal-processing chain for an audio play application to a SIP endpoint involves fetching a block from memory or file, parsing the block for header and raw data, feeding the raw data into a decoder, and adjusting gain by a gain block. The output of the gain block is then fed into an encoder, followed by packetization into RTP packets. A multi-channel voice mail server would repetitively perform the signal processing functions associated with audio play with the instructions executing out of cache. Data for each channel are fetched only at the start of the processing with subsequent blocks accessing the data out of cache. Large amounts of L2 and L3 cache ensure that there are fewer cache misses during the entire signal-processing chain. Figure 13 and Figure 14 show the performance boost (reduction of 12-15% in CPU utilization) due to L3 cache for an audio play application.

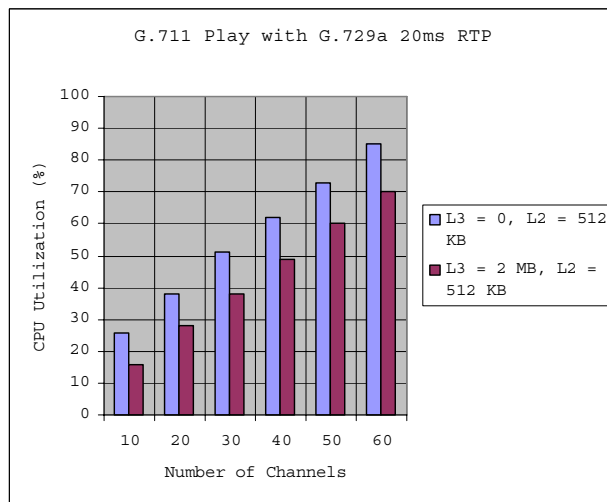


Figure 13: Cache performance G.729a⁷

⁷ All code running on a PC with an Intel® Pentium® 4 processor supporting Hyper-Threading Technology (disabled), 3.2 GHz, 512 KByte L2 cache, 2 MByte L3 cache and 1GB RAM using RedHat® Enterprise Linux.

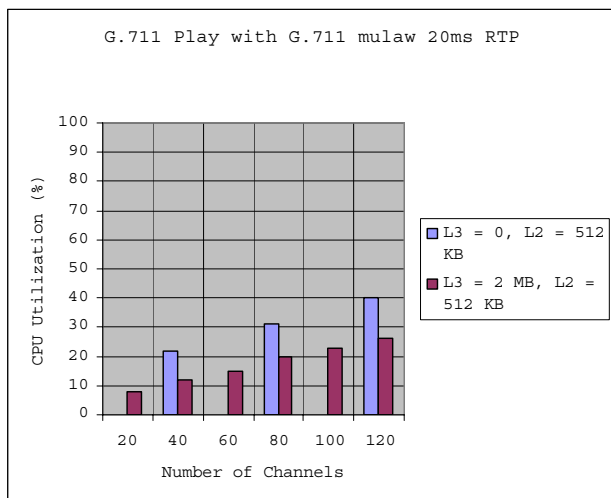


Figure 14: Cache performance G.711⁷

INTEL DEVELOPMENT ENVIRONMENT

Intel VTune Performance Analyzer

Intel VTune Performance Analyzer is a powerful tool for analyzing the performance of an application. VTune analyzer has two main modes, sampling and call graph mode [20].

Sampling mode is a non-intrusive way to profile the entire system. In sampling mode, statistics are rolled up to the application level. The user can then delve down to the function level. The user can double-click on a function and get instruction-level statistics as annotations in the source files [20].

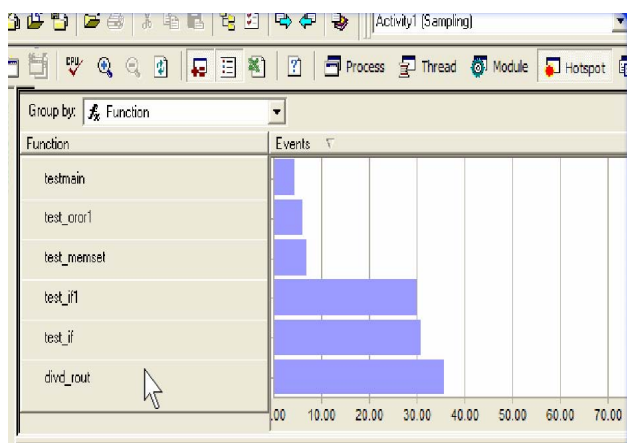


Figure 15: VTune Performance Analyzer sampling mode output [20]

Call graph mode is more invasive and slows the program under test. It does, however, enable a graphic view of calling sequences of the different procedures within the program. It further identifies critical paths in the program. Statistics are provided for each procedure, including “wait

time,” the amount of time the function spends waiting for an event to occur.

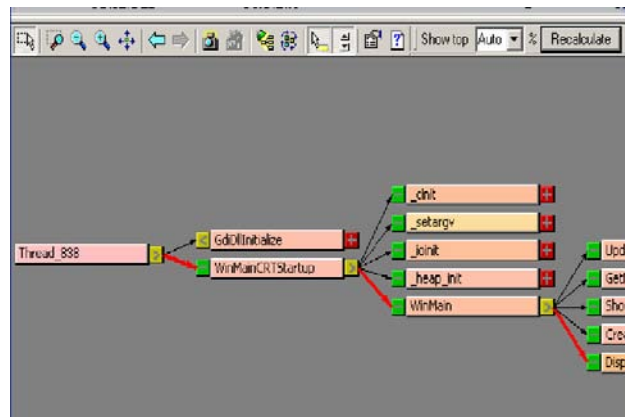


Figure 16: VTune Performance Analyzer call graph mode output [20]

Beyond CPU utilization and wait time, VTune Performance Analyzer collects important statistics such as cache misses, clock ticks per instruction, and thread utilization.

Under Windows, another tool, VTune Performance Analyzer Tuning Assistant, makes recommendations on improving the performance of certain functions.

Use of VTune Performance Analyzer can be thought of as a three-step process: set up, find an area to improve, and improve. Finding an area to improve, given VTune analyzer’s intuitive user interface, usually takes no more than an hour.

VTune analyzer’s output is function-centric. In some cases, it is nice to abstract function-level results into higher-level module results. A parsing utility can easily be used to aggregate function-level statistics into module-level statistics.

VTune Performance Analyzer should not be used as a primary tool for the measurement of performance. VTune analyzer’s role is in improving performance. However, VTune analyzer can be useful for understanding the performance characteristics of one release over another. For instance, suppose we upgrade a software release from A to B. If, as a matter of course, we measure A and B with VTune analyzer, we can compare their performance characteristics. Are we seeing the same relative performance of our code? Has one module significantly decreased in performance? Does this make sense? While use of this A-B sanity check is not strictly necessary, it can help rapidly identify probable design or coding inefficiencies.

VTune Performance Analyzer is an important tool for getting maximum performance out of the Intel

Architecture. Intel NetStructure Host Media Processing Software, in particular, benefits from regular inspection with VTune analyzer.

Intel C++ Compiler

The Intel C++ compiler is one of the software development tools available to accelerate software performance on Intel platforms. It is available for a number of different operating systems, including Windows and Linux. The Intel C++ Compiler 9.0 for Linux provides outstanding application performance for software running on Intel processors [25]. It includes advanced optimization features such as full support of multi-core processors with capabilities including Auto-Parallelization, Optimized floating point instruction throughput, Interprocedural Optimization (IPO), Profile-Guided Optimization (PGO), and Data prefetching. It includes a Compiler Code-Coverage tool and a Compiler Test-Prioritization tool. Optimizations specific to the IA-32 architecture are provided, such as full support for Streaming SIMD Extensions 3 (SSE3), Automatic vectorizer, and Processor Dispatch, as well as support for Intel Extended Memory 64 Technology. More details are available in “Intel C++ Compiler for Linux” [25]. The compiler also includes an enhanced debugger that allows debugging of optimized code, as well as support for stack frame runtime error checking to help reduce buffer overrun security exploits. Various case studies are presented in “Intel Software Tools Case Studies” [27], highlighting performance improvements due to Intel software tools. Table 9 highlights the performance improvement provided by Intel Compiler and IPP in one particular case study—H.263 Video Encoding [26].

Table 9: H.263 Encode Performance Improvement due to Intel Compiler and IPP [26]

ImageCom PC Encoder Configuration		Intel Pentium 4 processor-based system	
Intel IPP	Intel Compiler	Encode Time for 80-sec clip	Percentage Improvement
		123 sec	0%
Y		84 sec	32%
	Y	69 sec	44%
Y	Y	57 sec	54%

WHERE WE GO FROM HERE

While this paper has primarily focused on media servers from the point of view of traditional telecommunication applications, media servers will play an important role in the next-generation digital home as well. As broadband to

the home becomes commonplace, we will see the emergence of new and exciting applications that will have a big impact on our personal lives. We are already seeing the emergence of TV over IP (IPTV) promising to change the way we access public content such as broadcast and on-demand programming. Providers will be able to personalize what their customers receive; not only providing them with content that interests them but also content when they want it.

Going beyond public content, media services will be used in new and innovative ways to enrich our personal lives. For example, media servers may be used to share private content such as personal photos and video with friends and families, and to do so securely. Residential media servers may be used for home security and health monitoring as well as entertainment.

Intel technology will continue to advance to meet the demands of these future applications. With leadership technologies such as multi-core, I/O acceleration, virtualization, power management and security, to name just a few, Intel promises to continue to provide platforms that exploit the next generations of platform technologies to their fullest.

CONCLUSION

A media service network is a network through which a wide range of media services are provided by application programs controlling media resource functions on a media server. Intel offers a wide spectrum of standards-based technologies that facilitate the building of flexible, high-performance, low-cost media servers that can be deployed in the circuit-switched, pure-IP and next-generation converged networks.

We have seen that Intel NetStructure Host Media Processing Software is a feature-rich platform, requiring no special-purpose hardware, on which a wide variety of media applications can be developed for the next generation of IP-based networks. Utilizing industry standards and the popular Windows and Linux operating systems, HMP software-based solutions can be built with a significantly lower total cost of ownership and shorter time to market than proprietary hardware-based solutions. We walked through a simple example of how the high-level abstractions provided by the APIs in Intel NetStructure Host Media Processing Software make application development straightforward.

We demonstrated how low-level building blocks like Intel IPP make Intel Architecture processors capable of functionality previously reserved for special-purpose DSPs. The popular Intel Architecture processors provide a cost-effective infrastructure for media signal processing that is high in performance as well as economical.

Tools such as the VTune Performance Analyzer, the Intel C++ compiler, and Intel IPP enable software technologies, including Intel NetStructure Host Media Processing Software, to get the absolute highest performance from the hardware platform.

Leading the advances in technologies such as multi-core, I/O acceleration, virtualization, power management, and security, Intel promises to continue providing technologies and platforms that enable cutting-edge media services.

PERFORMANCE TESTING

All testing was performed internally by Intel.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit <http://www.intel.com/performance/resources/limits.htm>.

ACKNOWLEDGMENTS

We thank all of our colleagues who reviewed and commented on this work. We especially acknowledge Asha Abraham, Rangarajan R. Calyanakoti, Ashok Kondru, Steve Masson, Bhaskar Rao, Daniel Rhee, and Glen Shires, without whose insightful technical contributions we could not have produced this paper.

REFERENCES

- [1] "Introduction to CCITT Signaling System No. 7," *ITU-T Q.700* (03/93), March 1993.
- [2] "ISDN user-network interface layer 3 specification for basic call control," *ITU-T Q.931* (05/98), May 1998.
- [3] "Packet-based multimedia communications systems," *ITU-T H.323* (11/00), Nov. 2000.
- [4] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, "SIP: Session Initiation Protocol," *IETF RFC 3261*, June 2002.
- [5] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," *IETF RFC 1889*, Jan. 1996.
- [6] "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; IP Multimedia Subsystem (IMS); Stage 2 (Release 5)," *3GPP TS 23.228 V5.5.0*, 2002-06.
- [7] M. Arango, A. Dugan, I. Elliott, C. Huitema, S. Pickett, "Media Gateway Control Protocol (MGCP) Version 1.0," *IETF RFC 2705*, Oct. 1999.
- [8] H. Schulzrinne, S. Petrack, "RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals," *IETF RFC 2833*, May 2000.
- [9] "Voice Extensible Markup Language (VoiceXML) 2.1," *W3C Working Draft 28*, July 2004.
- [10] Melanchuk, T., Sharratt, G., "Media Sessions Markup Language (MSML)," *draft-melanchuk-sipping-msml-03*, Aug. 15, 2004.
- [11] Melanchuk, T., Sharratt, G., "Media Objects Markup Language (MOML)," *draft-melanchuk-sipping-moml-03*, Aug. 15, 2004.
- [12] Intel Corporation, "A Comparison of Intel NetStructure® Host Media Processing Software and Traditional Solutions," <http://www.intel.com/network/csp/applnots/9331an.pdf>, 2005.
- [13] Intel Corporation, "Intel NetStructure® Host Media Processing Software Release 2.0 for Windows," <http://www.intel.com/design/telecom/prodbref/8762.pdf>, 2005.
- [14] Intel Corporation, "Multimedia API Library Reference," http://www.intel.com/design/network/manuals/telecom/hmp15lin/pdf/files/multimedia_api_v1.pdf, Sep. 2005.
- [15] Intel Corporation, "Voice API for Host Media Processing Library Reference," http://www.intel.com/design/network/manuals/telecom/hmp15lin/pdf/files/voice_api_hmp_v2.pdf, Apr. 2005.
- [16] Intel Corporation, "IP Media Library API for Host Media Processing Library Reference," http://www.intel.com/design/network/manuals/telecom/hmp15lin/pdf/files/ip_media_api_hmp_v5.pdf, Aug. 2005.
- [17] Intel Corporation, "Device Management API for Windows* and Linux* Operating Systems Library Reference," http://www.intel.com/design/network/manuals/telecom/hmp15lin/pdf/files/device_mgmt_api_v3.pdf, Aug. 2005.

- [18] Intel Corporation, "Global Call API for Host Media Processing Library Reference," http://www.intel.com/design/network/manuals/telecom/hmp15lin/pdffiles/globalcall_api_hmp_v3.pdf, Aug. 2005.
- [19] Intel Corporation, "Intel NetStructure[®] Host Media Processing Software," <http://www.intel.com/network/csp/promo/9403web.htm>, 2005.
- [20] Intel Corporation, "Intel[®] VTune[™] Performance Analyzer Flash Demo," http://download.intel.com/software/products/vtune/downloads/vtune_v7_b42.exe, 2003.
- [21] Intel Corporation, "Intel[®] VTune[™] Performance Analyzers," <http://www.intel.com/cd/software/products/asm-na/eng/vtune/index.htm>, 2005.
- [22] Intel Corporation, "Intel[®] Integrated Performance Primitives 5.0," <http://www.intel.com/cd/software/products/asm-na/eng/perflib/ipp/index.htm>, 2005.
- [23] Intel Corporation, "Hyper Threading Overview," <http://www.intel.com/business/bss/products/hyperthreading/overview.htm>, 2005.
- [24] Intel Corporation, "Intel[®] Integrated Performance Primitives 5.0 performance data" <http://www.intel.com/cd/software/products/asm-na/eng/perflib/ipp/238657.htm>, 2005.
- [25] Intel Corporation, "Intel[®] C++ Compiler for Linux*" <http://www.intel.com/cd/software/products/asm-na/eng/compiler/clin/219622.htm>, 2005.
- [26] ImageCom Inc, "A short cut to success," http://cache-www.intel.com/cd/00/00/21/92/219258_imagecom.pdf, 2005.
- [27] Intel Corporation, "Intel[®] Software Tools Case Studies," <http://www.intel.com/cd/software/products/asm-na/eng/casestudies/index.htm>, 2005.
- [28] Intel Corporation, "Intel NetStructure[®] Host Media Processing Software Release 1.5 for Linux," <http://www.intel.com/design/telecom/prodbref/9323.pdf>, 2005.

AUTHORS' BIOGRAPHIES

Joseph Grecco is a system architect in DEG Architecture and Planning at Intel, Parsippany, NJ. He has 20 years of experience in the communications industry with over 10 years involved in the architecture and design of media servers, middleware, and APIs. He holds a B.S. degree in

Computer Information Science from the New Jersey Institute of Technology. Joe can be reached by e-mail at joe.grecco@intel.com.

Mark E. Mize is a system architect of media systems at Intel in Parsippany, NJ. He has a B.S. degree in Mathematics and a B.S. degree in Computer Science from the University of Texas at Austin. Mark has worked on IP protocols and media system design for the last eight years. Mark can be reached by e-mail at mark.mize@intel.com.

Ranjan Singh received his B.Eng. degree from Victoria University of Technology in Melbourne, Australia. Since joining Intel in 1999, he has worked on the digital signal processing components within Intel's NetStructure products. His current interests include architecture of next-generation media server products for enabling video and other emerging applications. Ranjan can be reached by e-mail at ranjan.singh@intel.com.

Copyright © Intel Corporation 2006. All rights reserved. This publication was downloaded from <http://developer.intel.com/>.

Legal notices at <http://www.intel.com/sites/corporate/tradmarx.htm>.

For further information visit:

developer.intel.com/technology/itj/index.htm