Embedded Software in Real-Time Signal Processing Systems: Application and Architecture Trends

PIERRE G. PAULIN, MEMBER, IEEE, CLIFFORD LIEM, MARCO CORNERO,
FRANÇOIS NAÇABAL, AND GERT GOOSSENS, MEMBER, IEEE

Invited Paper

This paper presents an extensive survey of trends in embedded processor use with an emphasis on emerging applications in wireless communication, multimedia, and general telecommunications. We demonstrate the importance of application-specific instruction-set processors (ASIP's) in high-volume, low cost applications.

We also examine some of the underlying trends of the applications in which embedded processors are used. This is followed by a description of embedded software development tool requirements. High-performance software compilation emerges as a key requirement.

Finally, specific industrial case studies of products in MPEG, videophone, and low-cost digital signal processor (DSP) applications are used to illustrate the architecture design tradeoffs, and highlight specific tool requirements. A companion paper in this issue [1] presents a comprehensive survey of embedded software development tools, focusing mostly on retargetable software compilation.

I. Introduction

The telecommunications, multimedia, and consumer electronics industries are witnessing a rapid evolution toward integrating complete systems on a single chip. System-level integration, combined with extremely short product design cycles, is only possible by implementing large parts of the system functionality in software running on integrated processor cores. Solutions range from general-purpose processor cores available in foundry catalogues to cost- and power-effective application-specific instruction-set processor cores (ASIP's).

While there is a clear trend in processor use for personal computing, with the domination of X86-based architectures and the prevailing use of a single operating system, the consumer electronics, multimedia, and telecommunication applications cannot be characterized so easily. Three important factors need to be considered when considering the future role of embedded processors in these applications.

1) The convergence of computing, communication, and consumer electronics. It is likely that the market characteristics of the latter will dominate: extremely short time-to-market combined with very low costs.
2) The stabilization of the personal computer (PC) market growth.
3) The increasing growth of wireless and multimedia.

This means the applications that will most influence technology evolution in the late 1990's and in the early 21st century will likely be consumer-oriented, with wireless communication and multimedia the main contributors. These trends have a major consequence on the underlying architectures and the use of embedded processors. This paper will analyze some of the trends in the following important new areas.

- A review of recent developments in embedded processor use, with particular emphasis on multimedia and wireless applications, two major areas slated for increasing growth.
- A more detailed analysis of embedded processor use based on an extensive survey at a telecom system house. This survey covered over 25 design groups using 8–24 bit digital signal processors (DSP's) and microcontrollers (MCU's) [2].
- Specific case studies of products in MPEG, videophone, and low-cost DSP applications.

We will then examine what will be required of embedded software development tools as a result of these trends. A review of recent development tools, focusing mainly on retargetable compilation, is presented in a companion paper [1].

Manuscript received February 1, 1996; revised December 2, 1996.

P. G. Paulin, M. Cornero, and F. Naçabal are with SGS-Thomson Microelectronics, F-38921 Crolles Cedex, France (e-mail: pierre.paulin@st.com; marco.cornero@st.com; francois.nacabal@st.com).

C. Liem is with the TIMA Laboratory/INPG and SGS-Thomson Microelectronics, F-38031 Grenoble, France (e-mail: liem@verdon.imag.fr).

G. Goossens is with Target Compiler Technologies and IMEC, B-3001 Leuven, Belgium (e-mail: goossens@imec.be).

Publisher Item Identifier S 0018-9219(97)02015-X.

II. Market and Processor Trends

A. Overall Semiconductor Market Trends

Two key trends are essential to consider when discussing the current and future characteristics of embedded processors:

- the continued growth of the market share of processors and memories in the overall semiconductor business;
- the emergence of multimedia and wireless applications as growth leaders.

In 1994, processors and memories represented 54% of the semiconductor market revenue. According to the World Semiconductor Trade Statistics [3], [4] this number is expected to grow to 61% by 1999. A key question therefore is to determine *which* processors will dominate this important part of the market. While a complete answer to this question is beyond the scope of this paper, we will examine the trends in some key emerging application areas.

1) New Technology Drivers: During the 1980's and early 1990's, it was widely acknowledged that general-purpose computing chips and memories were the main contributors to the evolution of VLSI technology and design methods. It appears that this situation is changing and that new applications are assuming this important role [5], [6]:

- wireless communication, e.g., GSM digital cellular, DECT cordless telephone, and IS-54B digital cellular;
- multimedia, e.g., MPEG2 decoders for set-top boxes and digital video disks (DVD), high-definition television (HDTV), videophone, and three-dimensional (3-D) video;
- video games.

From a market perspective, the following predictions by Dataquest and Forward Concepts for GSM, DSP, and PC market growth support this new role.

- The growth rate of GSM in the next five years will be over 30%, from 28 million parts in 1994 to an expected 100 million parts by 1999.
- The global DSP market is expected to grow by 40% per year, from its 1995 value of $1.7 billion to $9.1 billion in 2000 [7].
- The PC market registered sales of 60 million units in 1995, representing a healthy revenue growth rate of 25.6%. Recent numbers by Dataquest and IDC [8] show that the sales growth rate in North America and Europe for the first quarter of 1996 is starting to level off at 14% and 12.8%, respectively (relative to the first-quarter sales of 1995).

This puts the role of the PC as a growth leader in question for the midterm, and increases the relative importance of other emerging embedded applications. In the next section, we will examine the general trends in worldwide processor use, followed by a more detailed look at embedded processors used in emerging large volume applications.

B. Embedded Processor Trends

The use of programmable processors is typically divided into two main application classes: computing and embedded systems. Computing applications include desktop computers, notebooks, workstations, and server systems. They are characterized by the fact that the end-user can program them, and by the broad range of applications which they perform.

Embedded systems, which are the focus of this paper, are much more specific in nature. Camposano and Wilberg [9] define an embedded system by describing its main properties.

- It performs a dedicated function. Some examples are antilock brakes, control of electrical appliances and electronic consumer equipment, signal processing for personal audio or wireless terminals, etc. This is in contrast with computing-oriented products where it is more typical to run a large variety of applications.
- Real-time behavior must conform to very strict requirements.
- Correctness of the design is essential due to the potential impact on the surrounding environment or the person using the equipment.

We refer to the instruction-set programmable processors used in embedded systems as *embedded processors*. These include MCU's, DSP's, and microprocessor units (MPU). The latter are commonly divided into two categories: complex instruction set computers (CISC) and reduced-instruction set computers (RISC).

Another class of processors encountered in embedded systems is the ASIP. This is a programmable processor that is designed for a specific, well-defined class of applications. It can be seen as a further specialization of the MCU, DSP, and MPU classes above. An ASIP is usually characterized by a small, well-defined instruction set that is tuned to the critical inner loops of the application code. Alternatively, it can be a stripped down version of a standard processor (MCU, DSP or, more rarely, MPU) in order to meet cost constraints. ASIP's are most often encountered in real-time signal processing, image processing, and MCU applications. Numerous examples of ASIP's are presented in Section III, and their characteristics are reviewed more formally in the companion paper [1].

1) Worldwide Processor Volume Distribution: In spite of the high media visibility of the 32-bit RISC and CISC processors commonly used in PC's and workstations, from a pure parts volume perspective, the overall processor market is largely dominated by 4-bit and 8-bit MCU's. Fig. 1 shows the relative proportions of the volume of worldwide shipment of processors in 1994 [3], including MCU's, DSP's, and MPU's. Volume here refers to the *number* of parts sold, not the value of the parts sold. The total processor volume in 1994 exceeds 2.8 billion parts shipped.

According to Dataquest (see also the De Micheli paper [10]), MPU's (8, 16, and 32 bit) currently account for 60% of the total processor sales *revenue*, in spite of much lower volumes than those of MCU's and DSP's. This is due to the price of 32-bit MPU's which is typically an order of magnitude higher than that of 8 bit machines. Nevertheless,

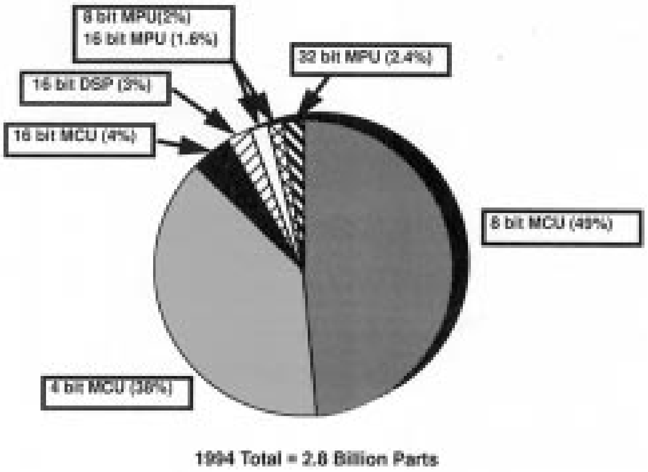

Fig. 1. Relative volumes of embedded processors in 1994.

8-bit MCU's alone still account for close to one quarter of total processor revenues.

In the future, it will become more and more difficult to separate out the revenues of processors from those of ASIC's, since the latter will often include an embedded core. The main message of Fig. 1 is that for high-volume products, the majority of embedded cores will likely be low-end MCU's and DSP's.

2) Evolution of 8, 16, and 32 Bit MCU Market Share: Given the importance of MCU in terms of volume and revenue, it is useful to take a closer look at the evolution of this market over recent years. Fig. 2 illustrates the relative proportion of the *revenue* from the sales of 8, 16, and 32 bit embedded MCU's. The percentages given are the proportions of revenues for each category. This shows that the transition from 8-bit MCU's to 16 and 32 bit is happening very slowly [10], [11]. We can therefore expect an important market presence of low-cost 8- and 16-bit MCU's for the midterm. As we will see later, this has a large impact on the associated embedded software development tool requirements.

3) Characteristics of 32-Bit Processor Market: While it might be natural to assume that the higher performance 32-bit processors are used predominantly for computing applications, in fact, the opposite is true. According to Dataquest, processors used in computing applications account for only 43% of processors sold in 1995, the remaining 57% are for embedded systems.

Furthermore, while x86-based processor architectures account for nearly 90% of all computing applications shipped in 1995, they only account for 30% of revenues in the embedded processor market [12], where a far greater diversity of architectures coexist. This diversity of architectures is a key characteristic of the embedded systems area.

C. Embedded Processor Trends Summary

- Low-end MCU's and DSP's largely dominate the overall processor market from a part volume perspective. They also account for approximately 40% of the total processor revenues.

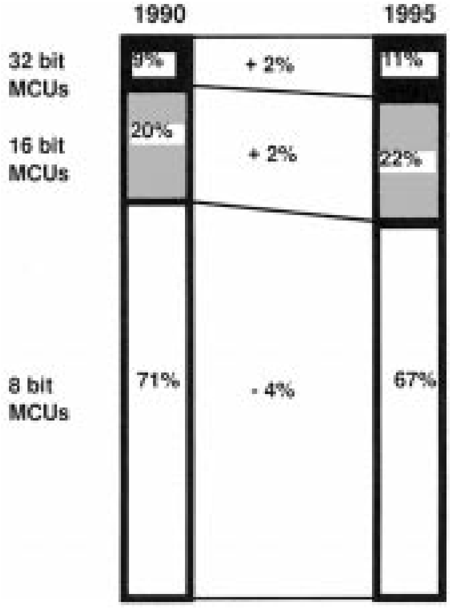

Fig. 2. Evolution of relative sales revenue of 8-, 16-, and 32-bit embedded MCU's.

- An extremely diverse collection of low-cost 8- and 16-bit processors currently dominate the overall revenues of the *embedded* processor market, with a very slow transition of volume and revenues toward 32-bit processors.
- In the 32-bit embedded processor market, there is much more architectural variety than in computing applications.

III. Embedded Processors in Multimedia, Wireless, and Telecom

In this section, we take a more focused look at embedded processors used in new emerging application areas. Specifically, our emphasis will be on the following three product classes:

- *multimedia*, including set-top boxes, HDTV, digital video broadcast (DVB), videophones, 3-D video, and video games;
- *wireless communication*, with European digital wireless standards like GSM and DECT, and North American digital cellular standards such as IS-54B;
- *telecommunications*, with a broad range of high-volume telecom products which make use of embedded processors.

We have selected what we hope is a representative sample of products in these three areas.

A. Multimedia

1) Multimedia Processors: Table 1 gives a list of products in the areas of MPEG, videophone, and high-end audio multimedia applications. See [13] for an overview of many of these products. This category is of particular interest, since many common blocks like MPEG1 and MPEG2 video/audio decoders, will be shared over a variety of emerging applications like set-top boxes for satellite

Table 1 Multimedia Processors (Set-top Boxes, Digital TV, MPEG, Videophone, Dolby)

Company	Processor	Application(s)
Philips	TriMedia$^{\text{TM}}$ VLIW ASIP [15]	MPEG1 encode/decode + MPEG2 decode + H.261 videophone
TCEC	68 bit VLIW DSP ASIP [16], 64 bit VLIW DSP ASIP	MPEG1 audio decoder, MPEG2/Dolby-AC3 audio decoder
	MCU ASIP, Hard-wired ASIC	MPEG2 video encounter control, MPEG2 video decoder
Chromatic	Mpact$^{\text{TM}}$ media ASIP [17]	MPEG1 and 2 video decoder + H.261 videophone
NTT	Two in-house RISC [18] + hard-wired logic	MPEG2 video encoder
Texas Instruments	MCU/DSP ASIP [19]	MPEG2 video decoder
Zoran [20]	20 bit DSP ASIP ZR38500	Dolby six channel AC-3
	20 bit DSP ASIP ZR38521	Two Channel AC-3 + MPEG1 audio
	20 bit DSP ASIP ZR38501	Two Channel AC-3 only
SGS-Thomson	8 bit MCU ASIP [21], [22]	Videophone: master control
	8 bit MCU ASIP	Videophone: bit stream processing
	VLIW DSP ASIP	Videophone: motion vector predictor

Table 2 3-D Accelerator Processors

Company	Processor	Architecture Style
Philips	Trimedia$^{\text{TM}}$ TM-1	ASIP VLIW
Chromatic	Mpact$^{\text{TM}}$ media processor	ASIP (2000 MOPS)
IBM	Mfast$^{\text{TM}}$	Array of 20 VLIW ASIP's
Rendition	Verite$^{\text{TM}}$	ASIP: RISC-style processor + pixel H/W engine
3Dfx	Voodoo$^{\text{TM}}$	Hard-wired graphics engine
3Dlabs	Permedia$^{\text{TM}}$	Hard-wired graphics engine
Nvidia/SGS-Thomson	NV1 (STG2000), NV3	Hard-wired graphics engine

and cable digital TV, DVB, DVD, PC-based multimedia accelerators, digital audio broadcast (DAB), and high-end audio systems for home and cinema applications.

Table 1 shows that virtually all of the MPEG2 video decoders are based on ASIP's, or on custom hard-wired logic. This is especially interesting in light of the fact that most vendors supplying MPEG decoder chips hold licenses for embedded microprocessor cores [14]. ASIP's are also very present in videophone applications as well as high-end audio like Dolby AC-3 and MPEG2 multichannel audio decoders.

The widespread decision to develop special ASIP cores highlights many important aspects of the multimedia market and other embedded applications [14]. First, cost is a critical factor. If a tailored architecture can deliver the same function at a lower cost, then it will become a viable engineering choice. This is particularly true for the cost-reduced version of a successful product. Typically, much of the revenue comes from the second-generation cost-reduced version of a chip or chip set.

Second, none of the current 32-bit microprocessor architectures is well-suited to the task of performing routines necessary for video decompression, e.g., inverse discrete cosine transformation (IDCT). Furthermore, video-output tasks gain very little benefit from the conventional data cache found on general-purpose microprocessors. Unlike computing applications, there is little locality of reference for video data; it is only processed once so there is no need to keep it in cache.

Finally, software compatibility is not an issue for a chip that will execute only a carefully optimized set of specific tasks. This is probably the most important issue, and one that clearly differentiates this market from the general purpose processor market.

2) 3-D Video Acceleration: Table 2 shows that four out of seven vendors of 3-D video processors use a dedicated ASIP approach, the three others use hard-wired graphics engines [23]. Once again, standard RISC or CISC-type MPU's are absent.

To illustrate the performance discrepancy of these ASIP's with respect to standard processors, a few numbers are worth examining:

- The Trimedia$^{\text{TM}}$ [15] is claimed to have a peak performance of 3.8 billion operations per second.
- The Mpact$^{\text{TM}}$ media processor [17] is claimed to deliver two billion operations per second.
- Finally, the Mfast$^{\text{TM}}$ [24], which is made up of an array of 20 VLIW ASIP's, is designed to deliver 20 billion operations per second.

It is clear that these levels of performance require dedicated architectures. Although the recently announced multimedia extensions (MMX) to the X86 class processors [25] will allow a part of 3-D processing to be realized in software, there is still an estimated factor of ten performance speedup needed beyond MMX to handle high quality game programs or professional-level 3-D graphics [26].

3) Video Games: As shown in Table 3, the manufacturers of video games often combine a high-end standard or customized RISC with one or more dedicated ASIP's [27]. The graphics performance of the product is more dependent on the ASIP than the embedded RISC. In most cases, the graphics power of these "toys" exceeds that of most PC's and some workstations at a fraction of the price. This area also illustrates another design tradeoff, namely, the use of an ASIP as an alternative to a hardware co-processor.

B. Wireless Communications

Representative examples of wireless designs in European, North American, and Asia-Pacific companies are shown in

Table 3 Game-Oriented Processors

Company	Product Model	Architecture Style
Nintendo	Ultra64$^{\text{TM}}$ ("Magic Carpet" spec.)	MIPS R4300 RISC + Media Signal Proc. ASIP + Media Display Proc. ASIP
Sega	Saturn$^{\text{TM}}$	SH7604 RISC (2x)
Sony	Playstation$^{\text{TM}}$	Custom R3000 + hard-wired audio/graphics logic
Atari	Jaguar$^{\text{TM}}$	64 bit ASIP
3D0	MultiPlayer$^{\text{TM}}$ (M2)	PowerPC 602 processor + ARM60 processor + hard-wired graphics logic

Table 4 Processors for GSM (Europe)

Company	Processor	Application(s)
France Telecom (CNET)	LIW DSP ASIP [28]	Hands-free ISDN terminal
	LIW (55 bit) DSP ASIP	GSM terminal
	LIW (46 bit) DSP ASIP	G.721 codec (DECT, CT2)
Alcatel	16 bit DSP ASIP [29]	GSM handset
Italtel	IEQU DSP ASIP [30]	GSM base station: equalization
	IEDM DSP ASIP	GSM base station: demodulation
Philips	"KISS" 16 bit DSP [31]	GSM handset
	Epics configurable DSP [32], [33]	DECT handset
SGS-Thomson	16 bit D950 DSP [34], [35] + user-defined co-processor	GSM, cellular, modern
Siemens	Pine$^{\text{TM}}$ DSP core (under license)	GSM handset

Table 5 Processors for GSM (North America and Asia–Pacific)

Company	Processor	Application(s)
AT&T	DSP16 ASIP family [36]	GSM, Modem Cellular handset Cellular base station
Northern Telecom	8 bit MCU ASIP + Motorola 56K	Cellular base station
TI	16 bit DSP C540 ASIP [37]	GSM handset
Toshiba	16 bit DSP ASIP (TC-8005)	GSM
NEC	16 bit DSP ASIP (uPD7701X)	IS-54 cellular phone
Fujitsu	Hard-wired ASIC	GSM

Tables 4 and 5. Nearly all the major system houses have designed ASIP's for this key product class. To the best of our knowledge, all of the second- and third-generation DSP's used in today's GSM handsets are based on dedicated ASIP's.

1) Europe: As shown in Table 4, the CNET R&D group of France Telecom claims that their DSP ASIP used for an ISDN hands-free phone allows a 50% power reduction over commercial DSP solutions [28]. This power reduction came mostly from the use of a large instruction word and higher levels of parallelism. This resulted in lower clock frequencies (16 MHz versus ∼40 MHz for a commercial DSP).

The development of a GSM ASIP at Alcatel [29] also led to a 50% power reduction, due to the dedicated architecture and an optimized clocking strategy.

Italtel claims that two in-house ASIP's replaced eight commercial DSP chips in a GSM base station [30]. For example, the Italtel "IEQU" DSP ASIP which performs GSM equalization, provides a throughput of 58 MIPS at 20 MHz. An implementation using a commercial DSP would require a clock cycle of 116 MHz to achieve the same performance as the ASIP. The second ASIP DSP, the "IEDM" chip, performs GSM demodulation, and provides a throughput of 90 MIPS with a 20 MHz clock rate. Here, a commercial DSP solution would require a clock rate of 180 MHz.

Philips designed the KISS16 architecture as a dedicated architecture for GSM applications only [31]. Subsequently, they have developed the EPICS family, based on a reconfigurable DSP core architecture. This approach was developed in order to reduce the development time of ASIP's [32], [33].

Another approach to ASIP development is the one developed by SGS-Thomson for the D950 16-bit DSP core [34], [35]. This core includes a coprocessor interface that allows the user to implement custom instructions on the coprocessor. Up to 16 user-defined instructions can be called directly from program memory. This combines many of the advantages of a standard core and a dedicated ASIP.

2) North America: Table 5 includes sample DSP and MCU designs at AT&T, Northern Telecom, and Texas Instruments (TI). Of these, the system houses have been particularly active on the ASIP front. AT&T has stated publicly that it considers proprietary ASIP design as a key advantage [36].

At Northern Telecom, ASIP's are used in many strategic high-volume products, including line cards and telephone sets [2]. As shown in the table, the Northern Telecom base station design combines standard DSP's and an ASIP MCU.

As a rule, most semiconductor companies emphasize the use of their existing standard DSP cores. Nevertheless, TI designed the "Lead" chip (TMS320C540), a DSP optimized for GSM [37], [38].

3) Asia-Pacific: The final three rows of Table 5 highlight DSP's designed for wireless communication by Toshiba, NEC, and Fujitsu. Although Fujitsu uses a hard-wired solution, they have stated their intention to move to a programmable one due to customers' request for highly flexible solutions [37]. The other two projects used programmable, but heavily optimized, processor cores. Toshiba stripped down an existing core, while NEC designed a dedicated ASIP.

C. General Telecom

The previous sections concentrated on a well-defined class of applications. Here, we will give a broader perspective for a wide class of telecommunications applications.

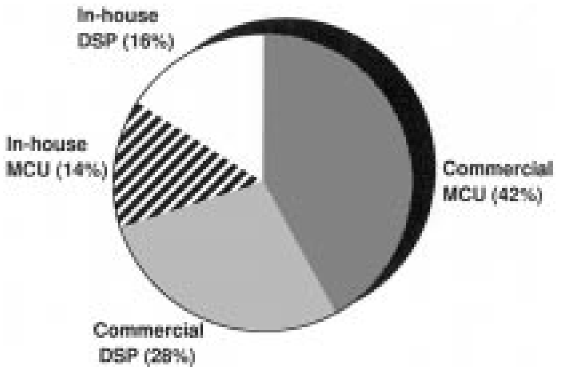

Fig. 3. Relative usage of commercial and in-house processors by design groups at Northern Telecom. Relative area is proportional to the number of design groups using a processor type.

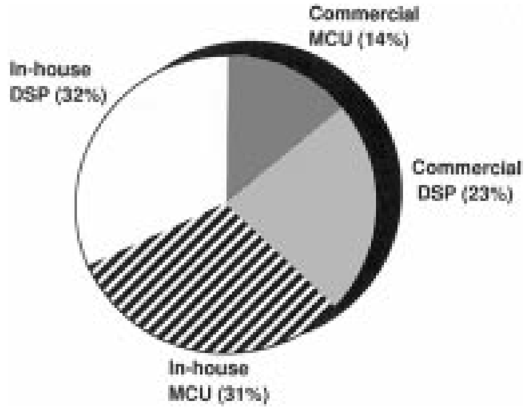

Fig. 4. Relative volume of commercial and in-house processors at Northern Telecom area is proportional to the number of parts shipped.

For this, we will rely on an extensive survey [2] of design groups at Northern Telecom Ltd., a major telecommunications system house. The survey was targeted exclusively at design groups which made use of embedded processors. The survey was limited to low-end processors (8–16-bit MCU and 16–24-bit DSP's), and specifically excluded 32-bit RISC and CISC processors. The survey consisted of a questionnaire to be completed by design group managers. In many cases, a telephone interview was conducted after reception of the completed questionnaire. In this paper, we will be highlighting the results for four areas covered by the survey:

1) list of processor(s) used in the design;
2) number of processors shipped for the previous and current years (or an estimate);
3) programming languages used (high-level language and assembly language);
4) embedded software development tool requirements for the future.

The latter two areas will be covered in Section V-C.

1) Application Areas: The types of products developed by the design teams covered a wide range of applications. A nonexhaustive, representative list follows:

1) automatic call distribution and operator headset voice processing;
2) asynchronous transfer mode (ATM) and data networks;
3) low power wireless base stations for cordless telephone standards CT2 and CT2+;
4) line interfaces for large telephone switches;
5) synchronous optical network (SONET) interface units;
6) integrated sound and data network (ISDN) phone terminals;
7) call display phones;
8) message delivery devices;
9) modems for private branch exchange (PBX);
10) telephone sets for PBX;
11) digital cellular base station radio;
12) reconfigurable line cards;
13) cryptography for secure data communications;
14) video codecs;
15) next-generation switches.

2) Processors Used: Fig. 3 depicts the relative usage by design groups of in-house ASIP's in comparison with commercial DSP's and MCU's. In Fig. 3, the area is proportional to the number of design teams using a category of processors (not the number of actual processors actually sold). This shows that roughly two thirds of the design teams made use of commercial chips, split equally between DSP and MCU chips.

3) Volume of Processors Shipped: The following chart, Fig. 4, shows a different story in terms of processor volume. Here, the area is proportional to the number of processors shipped. In other words, roughly two-thirds of the chip volume is for in-house ASIP's rather than commercial processors.

These two charts summarize the main difference between ASIP and general-purpose processor use: the latter are used when time-to-market constraints dominate, while ASIP's are predominant in large volume, low-cost applications. They are also a transition path for applications that were performed on hard-wired logic but require more flexibility while minimizing any performance or cost compromise.

D. Embedded Processor Trends Conclusions

This presentation of embedded processors for wireless and multimedia applications clearly highlights some important facts.

- The increasing diversity of processor architectures, driven by low-cost consumer-oriented markets.
- The wide range of architecture partitioning strategies and the diversity of building blocks. Many products performing the same function are designed using very different combinations of RISC, ASIP, and/or hard-wired co-processors.
- The dominance of ASIP's for the high-volume, low-cost segments of the market.

The decision to develop special ASIP cores highlights many important aspects of the multimedia market and other high-volume embedded applications [14]. First, cost is a critical factor. Second, most of the standard microprocessor architectures are not well suited to the task of performing routines necessary for specific problems in image and audio processing for example. Finally, software compatibility is not an issue for a chip that will execute only a carefully optimized set of specific tasks. This is probably the most important issue, and one that clearly differentiates this market from the general purpose processor market.

1) Today's General-Purpose Processors Solve Yesterday's Problems: A common argument against the continued existence of ASIP's is that many recent applications which required an ASIP solution could be replaced by the latest general purpose processor. Unfortunately, the applications themselves do not stand still, and by the time MPEG1 audio decoding can be performed by a standard DSP, CPU, or RISC, the market demand is for low-cost DSP's which perform MPEG2 and Dolby AC3.

This may explain why the native signal processing (NSP) initiative—whereby part of the signal processing tasks are performed on a X86 class processor—has had few big commercial applications so far [25]. In other words, the increasing demands for multimedia continually require a higher end X86 processor, and the same function can be offered more cost-effectively by a dedicated multimedia processor.

This phenomenon should be compounded by the presence of standard application programming interfaces (API's) that isolate the hardware from the operating system. The same API call can be mapped to an X86 class processor, or to a low-cost ASIP.

Manufacturers of X86-class processors have responded recently with specific MMX for its new processors [25]. Nevertheless, there is still an estimated factor of ten performance speedup needed beyond MMX to handle high-quality game programs or professional-level 3-D graphics [26].

2) Outlook: If past history gives any indication of future trends, then the most important conclusion from this part of the survey is that emerging and future embedded applications like MPEG4, multimode wireless, HDTV, virtual reality, and interactive 3-D games will be expected to be available at competitive prices. This will continue to justify the development of innovative dedicated ASIP architectures. This is the main message of this section.

Of course, this is not to say that general-purpose processors will not have an important role. They should continue to dominate the number of design starts, particularly in lower volume applications. Furthermore, when an ASIP is present, it is often coupled with a standard RISC or MCU. In the latter case, however, the role of the general-purpose processor or MCU is that of a commodity part. The ASIP and/or a hard-wired coprocessor, provide most of the added competitive value.

IV. Embedded Systems Application Trends

In the previous sections, we have focused on the processor architecture trends. It is also important to examine some of the underlying trends of the application requirements for multimedia and wireless products. Here, we will briefly cover two fundamental ones: namely, the growth of application complexity and the emergence of new standards. Both will have significant impact on design tradeoffs.

A. Complexity Growth

The single most important characteristic of wireless and multimedia applications is the rapid growth of the complexity of the design.

- New wireless handsets and base stations will need to support multiple modes, e.g., one or more of GSM, IS-54B digital cellular, the emerging CDMA-based digital cellular, DECT, pager, etc.
- The integration of communication and fax functions in wireless terminals.
- Merge of cellular phone and PDA (personal digital assistant) functions.
- Videophone standard evolution: the new H.263 standard is many times more complex than the previous H.261 standard.
- The continued evolution of video coding standards from JPEG, to MPEG1, MPEG2, and eventually to MPEG4. Each standard evolution is accompanied by significant complexity increase.
- In multimedia audio, the simple stereo systems of the 1980's are now replaced by complex audio coding standards, e.g., Dolby AC3, MPEG2 that support multichannel "surround" audio.

As a result, many functions currently in hardware will be performed in software. In many cases, an ASIP will be required for performance or cost reasons.

B. Emergence of New Standards

In the PC area, MSDOS$^{\text{TM}}$, Windows$^{\text{TM}}$, and X86 object code have become *de facto* standards. While the situation is not as simple in multimedia applications generally, there are some definite trends.

1) Multiple subsystem standards are emerging: MPEG2 audio and video, Dolby audio (Prologic$^{\text{TM}}$, AC-3, etc.), H.263 videophone, digital wireless (GSM, DECT, IS-54B).
2) These standards are invariably described as ANSI C executable specifications.
3) Furthermore, the chips executing the applications are increasingly being invoked from a standardized applications program interface (API). For example, the new "DirectX" API designed for PC-based audio and video subsystems [39], or the common applications programming interface (CAPI) being defined for GSM [40].

As a result, this frees up the designers in the choice of processor architectures [41], since they are constrained to the API standard only, and have more freedom on the architecture implementing the functions called via the API. Furthermore, they are not bound by legacy code, and only need to develop a well-defined S/W function.

In our opinion, tools that will allow the designer to *efficiently* transform, refine, and map the executable C

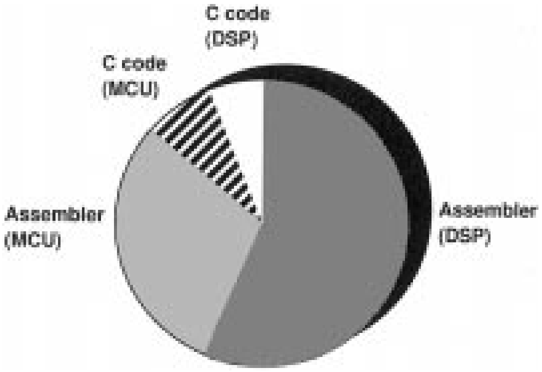

Fig. 5. Use of C code versus assembly code by design teams at Northern Telecom area is proportional to the number of lines of original codewritten by designers.

descriptions onto a cost- and power-efficient architecture will provide significant competitive advantage. This will be explained further in the next section.

V. Embedded Software Development Needs

The design teams developing embedded software for the wide variety of processor architectures cited above will require increasingly sophisticated tools. We first take a brief look at commercial tool support trends, followed by the tool needs derived from the Northern Telecom embedded processor survey. We conclude this section with a view of the emerging "ideal" hardware-software codesign tool environment.

A. Trends in Processor Tool Support

1) Increase of CAD Vendor Support for Cores: In the last few years, there has been an increasing level of support for various processor cores by computer-aided design (CAD) vendors [42]. Also, semiconductor companies are either working more closely with companies specializing in compiler development [43], [44], or simply acquiring them [45], [46]. These alliances and acquisitions can only improve the design flow for optimized hardware/software designs.

2) C Compiler Status: Nevertheless, the current situation regarding the quality of the code produced by commercial C compilers is not promising. Two aspects need to be considered: code size and code execution speed. Low code size is typically the priority in MCU applications, while execution speed is usually the most important criteria for DSP applications [2]. One company specializing in disk drives, claims that the compiled code for fixed point DSP's is as much as ten times slower than hand-coded assembly [47]. Another, specializing in DSP-based modems, claims that the execution speed is, at best, about four times slower; at worst, nine times slower [47]. The "DSPstone" benchmark of the compilers for the main commercial fixed point DSP's (Motorola 56001, TI TMS320C5x, ADI ADSP2101, NEC uPD77016, and AT&T DSP1610), confirms these execution speed figures [48], [49]. This is true for DSP *and* control applications (running on a DSP), although control-oriented code fared slightly better (execution speed degradation of a factor of three is typical).

Although low-cost, fixed-point, register-poor DSP's are notoriously difficult to develop efficient compilers for, there is apparently still much progress to be made in high-end, floating-point DSP compilers as well. Some companies claim a two-to-one execution speed degradation for existing floating-point DSP compilers [47].

These numbers are in startling contrast with embedded software designer requirements, who claim that an execution speed degradation of more than 10–25% is rarely tolerable. For this reason, embedded systems designers continue to program in assembly code, as confirmed in the Northern Telecom embedded processor survey presented next. In the long term, this causes a major problem with 10–100 K lines of legacy assembly code which locks designers to old architectures.

B. Telecom Survey

The results of the Northern Telecom survey give some very clear messages on the industrial needs for embedded software development. They also quantitatively confirm many of the observations made above.

1) C Versus Assembler: An important area covered by the survey was in the use of programming languages for embedded software development. Only one high-level language was used, namely ANSI C. In all other cases, designers programmed directly in assembly language. Fig. 5 shows the relative proportions of the different types of programming languages used throughout the design groups. The area in the chart is proportional to the number of lines of code originally written by the designer. We do not include here the number of lines of assembly code compiled from the C code. Four categories of code are presented: lines of C code written for MCU's, C code written for DSP's, assembly code written for MCU's, and assembly code for DSP's.

The most important fact is the dominance of assembly code as the means of algorithm capture.This is true for both DSP's and MCU's. For the latter, assembly code represents over 75% of code written. For DSP's, assembly code accounts for over 90% of the lines of code!

In many cases, the groups we talked to complained of the poor quality of assembly code generated by available compilers. In some cases, the code produced was incorrect. Essentially, the performance of applications was too important to sacrifice, especially for DSP chips, as reflected in the comparatively greater proportion of DSP code written in assembly.

2) Embedded Software Development Effort Trend: Fig. 6 depicts the manpower associated with the development of embedded software in the products designed by the groups surveyed. The height of each of the three columns is proportional to the number of person-years spent on this activity over six years (three periods of two years). These figures are based on the combined numbers collected from the Northern Telecom embedded processor survey [2], as well as an earlier 1992 survey of DSP design teams, also at Northern Telecom [50]. This clearly illustrates a strong trend: namely the rapidly increasing effort associated with

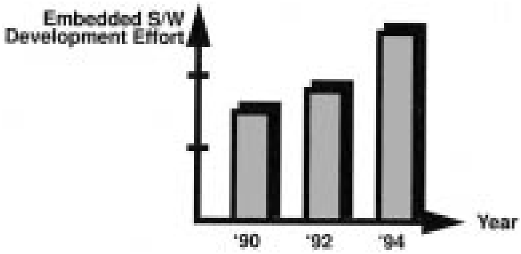

Fig. 6. Embedded software development effort trends. Column height is proportional to the number of person-years associated with embedded software development in the design groups surveyed.

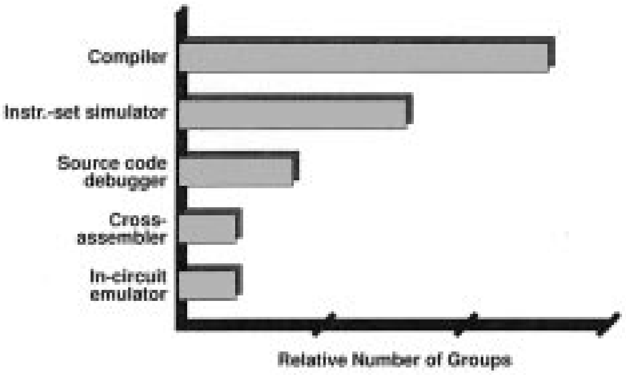

Fig. 7. Relative importance of embedded software development tools.

embedded software development. By the end of 1994, for the entire sample of design groups surveyed, the effort associated with embedded software development exceeded significantly that of hardware-oriented development.

3) Future Requirements: Although the graphs of Figs. 5 and 6 indicate the current situation for embedded processor users, here the focus is on future tool needs. In order to identify these, each design group was asked to identify the most important tool for the future. Fig. 7 illustrates these needs, with the x-axis width proportional to the number of groups which requested a specific tool type. By far the most pressing need is for *improved compiler technology*, in order to allow the designers to express their algorithms in a high-level language rather than assembly-level language. The shift has already occurred for developers using general-purpose architectures in computing systems, but it is clear that compiler technology has obviously lagged in the field of embedded processors.

The second most desired tool is a high-performance instruction-set simulator. This must run application code at 10 000–1 million instructions per second. The ability to run cycle-accurate simulations is an important option.

The third most cited tool is a source-level debugger. This type of tool links the compiled application object code, running on the instruction-set simulator or the actual processor, back to the C source code. This is essential since the user is not familiar with the compiled code and should be able to debug directly from the original C source code.

An important common characteristic we have discovered in practice is that while a high-performance compiler is the highest need, there are very few groups that will accept to use it without a source-level debugger. Furthermore, in order to validate a compiler for an ASIP in the early design stages, it is essential to have a fast functional model of the ASIP to run functional simulations and compare a Unix-based compilation route with the actual ASIP assembly code running on a model of the ASIP. In other words, the top three requirements need to be considered as a whole, and are inseparable in practice.

The requirement for a cross assembler is due to the enormous amount of legacy assembly code that needs to be remapped onto a new processor. Some of the design teams surveyed stated that the legacy assembly code amounted to nearly 100 person-years of development. The requirement for cross assemblers will therefore not disappear soon, and will continue to exist until a majority of code is developed in C.

C. Embedded Software Needs: Bottom Line

The increased complexity of applications highlighted in the previous section, coupled with the predominance of low-cost 8- and 16-bit processors, and the continued importance of ASIP's, leads to some very clear, yet sometimes opposing, needs.

1) High-performance compilers for low-cost, irregular architectures, with heterogeneous register structures—such as those found in many MCU's, most DSP's, and nearly all ASIP's.
2) In turn, this implies a compiler development technology that offers:

- rich data structures to support the complex instruction sets and algorithmic transformations required to exploit these effectively;
- extensive search to explore the numerous register allocation, scheduling, and code selection permutations;
- methods for capturing architecture specific compiler optimizations easily.

3) An environment that supports the rapid development of these compilers, due to the variety of processors to support.
4) The compilers need to be associated with tools like performance profilers, source-level debuggers and in-circuit emulators. These tools need to be retargetable to specific processors with a minimum of effort.
5) For ASIP-based designs, the ability to quickly provide feedback on instruction set selection decisions.
6) Rapid deployment of cycle-accurate instruction set models. Ideally, these models need to run at speeds of 10 K–1 M instructions per second.
7) Synthesis of lightweight real-time operating systems (RTOS). The purpose of an RTOS is to handle the run-time scheduling of tasks, taking into account the interaction with the system's real-time environment. Existing general-purpose operating systems are rarely used as they are expensive in terms of execution

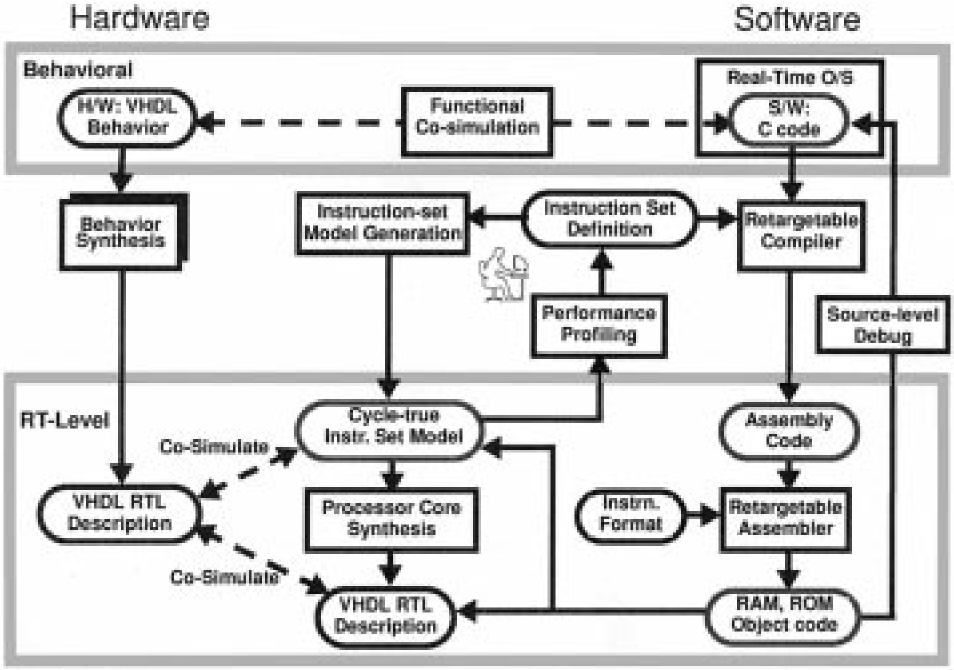

Fig. 8. The "ideal" hardware–software co-design environment.

speed and code size. Ideally, an application-specific RTOS with only the desired functionality would be generated automatically.

A block diagram of an "ideal" co-design environment that supports many of these requirements [51] is shown in Fig. 8.

Two main levels of abstraction are shown: behavioral and register-transfer level (RTL). It is assumed that commercially available logic synthesis and physical design tools are used for the gate-level and physical implementation.

At the behavioral level, HDL processes are used to describe hardware. This description has only minimal timing information. In particular, assignment of operations to specific clock cycles is not specified. On the software side, C code is used at that abstraction level. Timing is not described in this case, only sequencing and control flow are given.

A functional co-simulation allows to validate the global behavior of the software and hardware. Sequences of events and data computation and exchanges between H/W and S/W can be validated, but detailed cycle-accurate timing behavior is not simulated [52], [53]. The main requirement of this tool is that it must allow the same C source code to be used for co-simulation as well as the final compilation onto the target processor.

On the hardware side, the HDL process is transformed manually, or using behavioral synthesis tools, to produce an RT level HDL description.

For software, the C code of the application (linked with the RTOS code, if applicable) is mapped onto the target processor's instruction set to produce optimized assembly code. The compiler can be based on a traditional single target compiler or a retargetable compilation environment. The latter compiler approach is presented in detail in a companion paper in this issue [1]. An important requirement is that the resulting compilers offer source-level debugging capabilities and a complete assembler/linker back-end.

The instruction set definition is also used to generate a C (or HDL) bit-true and cycle-accurate model of the target processor's instruction set. This permits the execution of the object code on a virtual model of the processor. A C-based model is preferred over an HDL model in order to maximize model execution speed. This model can be run in standalone mode for a fast validation of the algorithm running on the architecture, or co-simulated with HDL models. In the former case, the cycle-accurate behavior is often not essential. However, the generation of accurate cycle counts is important in both cases.

A final tool in this environment generates a synthesizable HDL description of a processor from an instruction set description. An alternative is a reconfigurable processor HDL description which supports user-defined parameter settings. An example of this approach is given in the case studies below.

A key characteristic of this "ideal" environment, is that it allows for exploring ASIP architectures. A profile tool which measures static or dynamic code utilization and performance guides the selection and refinement of the processor instruction set. Validation against the hardware environment of the processor needs to be performed at the behavioral level, instruction set level, RT level, and gate level.

VI. Case Studies

We have chosen four industrial designs to illustrate some of the trends presented in the previous sections:

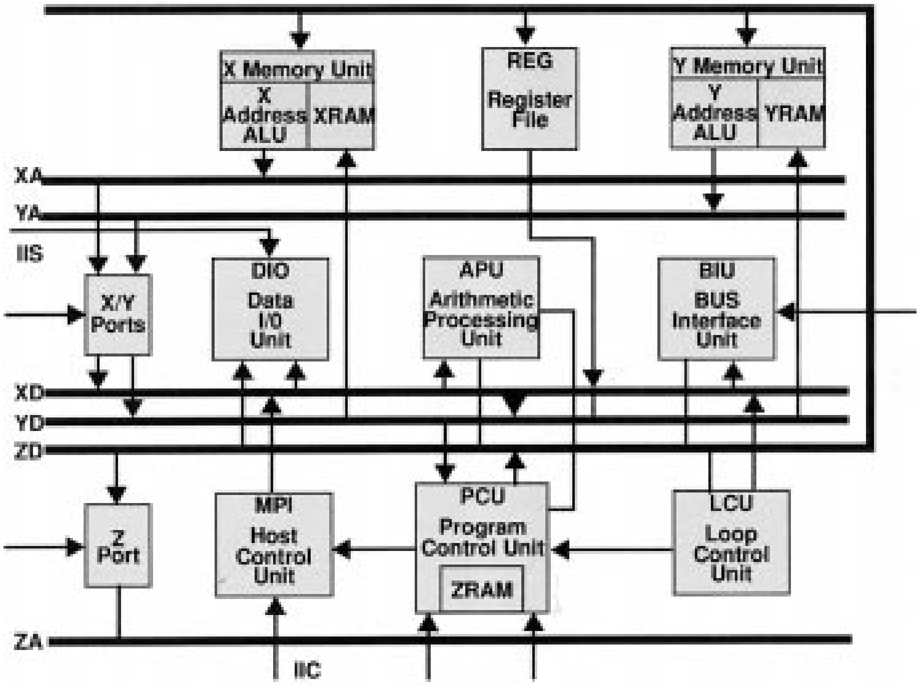

Fig. 9. EPICS architecture template.

1) the Philips EPICS family of configurable DSP's [32], [33];
2) a DSP ASIP designed by Northern Telecom [54];
3) a videophone designed at SGS-Thomson [21], [22], [51];
4) an MPEG2 audio decoder designed at Thomson Consumer Electronics Components (TCEC) [16].

The objective is to illustrate, via concrete industrial examples, the following key messages.

1) Reconfigurable, synthesizable DSP cores are being successfully used in industry as a method to quickly develop new ASIP's.
2) A key feature of an ASIP is the simplifications which can be obtained with respect to a standard DSP. These are achievable without performance penalty due to the well-defined application for which it is designed.
3) The increasing complexity and evolution of standards is driving the move from hardware to software. In many cases, an ASIP is needed in order to attain sufficient performance and/or cost.
4) In all the case studies, high-performance C compilation is an essential part of the design methodology.
5) New hardware-software co-design needs are emerging. This includes multilevel co-simulation and ASIP architecture exploration.

A. EPICS DSP Core

The EPICS reconfigurable DSP core architecture is presented in Fig. 9. This approach was developed by Philips in order to reduce the development time of ASIP's. Customized EPICS core versions are derived from a parameterizable architecture model by tuning the architecture configuration before synthesis. One of the key characteristics is the use of a technology-independent library of parameterized VHDL module generators which map onto standard-cell libraries in a variety of submicron technologies. Details of the architecture and the user-definable parameters can be found in [32], [33]. Typical characteristics of the EPICS DSP are shown in Table 6. The performance figure is given in millions of instructions per second (MIPS) and in millions of operations per second (MOPS). The latter figure is up to four times higher since an instruction can contain up to four parallel micro-operations.

Table 6 EPICS typical chip characteristics

Technology	Area	Clock Cycle	Performance
CMOS 5v, 1 μ	85 mm^2 (w 7 Kbit ROM 42 Kbit RAM)	33 MHz	33 MIPS (132 MOPS max)

The EPICS core has been applied successfully in wireless DECT terminal, consumer digital audio, and multimedia applications. The design objective is to deliver dedicated processor capabilities in a reduced design time. This approach can be seen as an intermediate point between an ASIP and an embedded standard core.

1) A Key Characteristic of the EPICS Case Study: The EPICS project has demonstrated a key capability, namely, the ability to quickly generate application-specific DSP cores from parameterizable and synthesizable VHDL descriptions. This is in strong contrast with the traditional approach of a well-defined list of standard DSP cores, as proposed by most semiconductor vendors [58], [59]. However, this new approach cannot be successful overall if

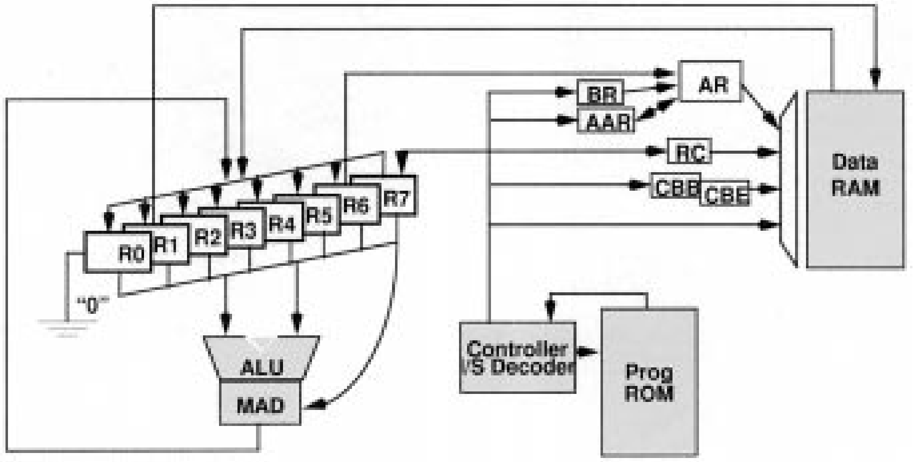

Fig. 10. BNR/Northern telecom DSP ASIP.

it is not associated with a retargetable embedded software development environment, such as the one presented in Fig. 8.

B. Northern Telecom ASIP

Fig. 10 gives the block diagram of a Northern Telecom DSP ASIP designed for a private local telephone switch, called a key system unit; a more detailed description can be found in [54]. This block diagram clearly highlights the simplifications with respect to a commercial general purpose DSP.

- There is a single data memory instead of the commonly used X and Y memories.
- Bus connections are reduced in many ways. There is only a single connection from register R1 when storing a value to the data memory. Register R7 is the unique possible destination of the immediate field of the microinstruction memory. Also, register R7 is the only possible right-hand source of the multiplier. Finally, R6 is the only register that can be used to transfer a computed address to the address calculation unit.
- The datapath bit widths are optimized to the application requirements (a combination of 8-, 12-, and 16-bit data busses).
- The instruction word width is 40 bits wide and supports up to five parallel operations: control, load/store, arithmetic and logical unit (ALU) operation, immediate value load, address calculation unit operation.

1) Tool Requirements: The key requirement in this design is for a high-performance compiler that can be adapted to this irregular architecture, with heterogeneous register structure. In turn, this implies a compiler development technology that offers:

- rich data structures to support a complex instruction set and algorithmic transformations required to exploit these effectively;
- a register allocation approach that can deal with the heterogenous registers and their irregular connectivity.

An optimizing C compiler for this processor was developed by Northern Telecom, using an extended template pattern base, a code selection approach based on dynamic programming [54], [55], and a register-class driven register allocation algorithm [54], [56]. These approaches are reviewed briefly in the companion paper [1]. The measured code size overhead with respect to manually written assembler was reported to be less than 20% [54].

2) A Key Characteristic of the Northern Telecom ASIP Case Study: This processor example clearly highlights the key feature of an ASIP, namely the simplifications which can be obtained with respect to a standard DSP. These are achievable without performance penalty due to the well-defined application for which it is designed. In this case, a detailed study of the most-used inner loops of the application allowed to strip out general connections which were not required. Furthermore, a wide instruction word allowed for the level of parallelism (up to five parallel micro-operations) which was required to meet performance constraints.

The effective use of recent compiler techniques demonstrates that the use of a low-cost, high-performance ASIP does not preclude good quality software development tools.

C. SGS-Thomson Videophone

The block diagram shown in Fig. 11 represents the design of a single chip videophone. This is an evolution of a previous chip, a video codec currently in production. The original chip simultaneously encodes and decodes 15 QCIF (144 × 176 pixels) images per second, according to the H.261 standard. A more detailed description of the existing H.261 chip's functionality and design is given in [21] and [22]. The recent design shown in Fig. 11 will support the new H.263 videophone standard.

As shown in the block diagram, there are five embedded processors:

- one standard DSP for the sound codec (SGS-Thomson's D950, a core version of the ST18950 stand-alone DSP);

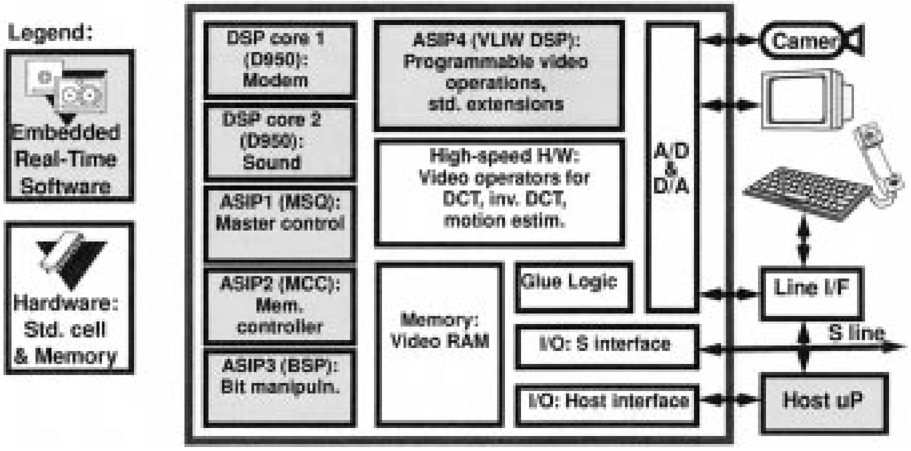
Legend:
Embedded Real-Time Software
Hardware: Std. cell & Memory
DSP core 1 (D950): Modem
DSP core 2 (D950): Sound
ASIP1 (MSQ): Master control
ASIP2 (MCC): Mem. controller
ASIP3 (BSP): Bit manipuln.
ASIP4 (VLIW DSP): Programmable video operations, std. extensions
High-speed H/W: Video operators for DCT, inv. DCT, motion estim.
Memory: Video RAM
Glue Logic
I/O: S interface
I/O: Host interface
A/D & D/A
Camer
Line I/F
S line
Host uP

Fig. 11. Integrated video telephone (H.263) block diagram.

- a MCU ASIP for global control (named MSQ);
- a second ASIP that controls the memory (the MCC);
- a third ASIP for bit-processing operations used in variable length coding and decoding, as well as audio/video multiplexing and demultiplexing (the BSP);
- and finally, a fourth ASIP, a very large instruction word (VLIW) DSP, used for motion vector prediction calculation.

Optionally, a second D950 DSP core may be used for modem functions in a variant of this design. There are four important trends when comparing this design (H.263) with the previous one (H.261):

- The increased number of embedded processor cores (five, in comparison with the first H.261 design that used only the MSQ and MCC cores).
- In the case of the BSP and VLIW DSP, these cores replace dedicated hard-wired logic realizations. The design constraint which forced this transition is the need for flexibility: 1) to adapt to the H.263 standard evolution and 2) to accommodate any specification or design errors, due to the complexity of the H.263 standard.
- The increased number of functions performed on the embedded cores. For example, much of the function currently performed on an external host microprocessor (the ST9 MCU) will be shifted to the internal MSQ ASIP in the next design.
- The significant evolution of the instruction set of ASIP's reused from the first design. This is to accommodate the more complex H.263 standard needs.

1) Tool Requirements: The presence of two standard cores and four ASIP's in this design implies important needs in tools for H/W-S/W co-design. The main requirements:

- high-performance C compilers for all the ASIP's (one already exists for the D950 DSP);
- a tool to manage the C-VHDL co-simulation for functional validation;
- another set of utilities to manage the H/W-S/W co-simulation at the RT level;
- high-level synthesis tools for the high-speed hardware blocks (e.g., direct DCT and IDCT, motion estimator).

The combined use of advanced tools for compilation, co-simulation, and hardware synthesis of the previous H.261 videophone is described in [51], while the co-simulation tool is detailed in [52] and [53], and the compiler development in [57] (see the companion paper [1] for a brief summary of the compiler approach). In particular, it was shown that the retargetable compiler approach led to near-zero code size overhead.

An important feature of the method used in this design is that the functional co-simulation at the behavioral C-VHDL level allowed to forgo the development of a source-level debugger. The assembly code running on the processor never needed to be examined. It was only necessary to compare simulation traces at functional and register-transfer levels.

2) Key Characteristics of the Videophone Case Study: This project highlights four important trends:

1) The transition from hard-wired implementations to a mixed implementation on hardware, ASIP's, and standard DSP's.
2) The need for programmability due to continuous evolution of the H.263 videophone standard.
3) The need for high-performance compilation and co-simulation tools since a significant part of the overall chip functionality is contained in the embedded software. A high-performance compiler and co-simulation tool were developed and used effectively.

D. TCEC: MPEG2 Audio DSP

Fig. 12 depicts a simplified block diagram of a VLIW ASIP that is used to perform MPEG2, Dolby AC-3, and Dolby Prologic^{TM} audio decoding. This is an evolution of the architecture of the MPEG1 audio decoder used in the satellite to set-top box DirecTV application [16]. Other target applications for this ASIP include DVD, DVB, and next-generation digital set-top box satellite systems. This is a good example of a low-cost, high-volume multimedia application.

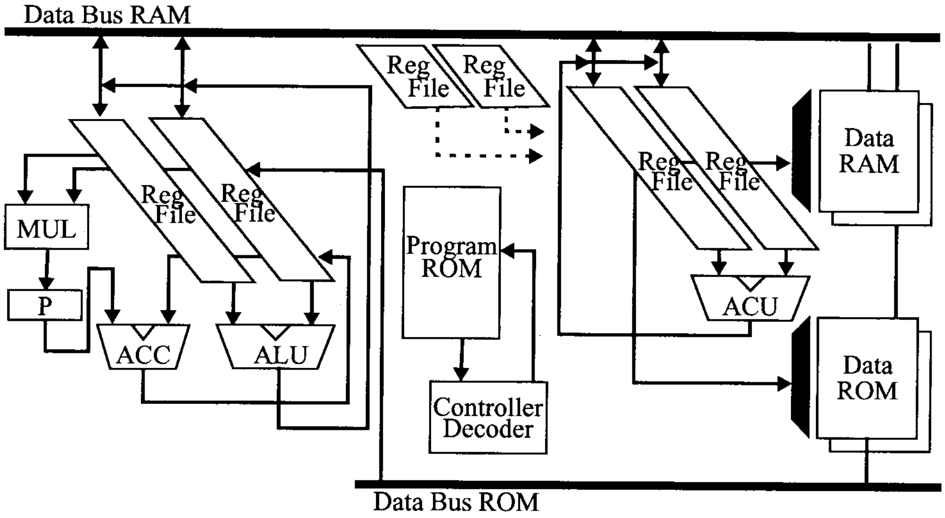

Fig. 12. TCEC MPEG2 audio VLIW block diagram.

The architecture shown in Fig. 12 is the result of a thorough analysis of the time-critical functions of the MPEG2 and Dolby-AC3 standards. At first glance, it is similar to many commercial DSP's. It is based on a load/store, Harvard architecture. Communication is centralized through a bus between the major functional units of the ALU, address calculation unit (ACU), and memories. The controller is a standard pipelined decoder with the common branching capabilities (jump direct/indirect, call/return), but also including interrupt capability (go-to/return-from interrupt) and hardware do-loop capability. Three sets of registers are used to provide three nesting levels of hardware loops; however, this can be increased without limit by pushing any of these registers onto the stack.

There are certain features of this architecture which set it apart and allow it to perform well in this application domain. The post-modify ACU includes custom register connections and a rich set of increment/decrement capabilities which allow it to efficiently access the special memory structures. The instruction encoding is designed so that a carefully selected subset of the ACU operations can be executed in parallel.

The ACU has been designed to work in concert with the memories. The memory structure has been developed around the data-types needed for the application. A first partition separates memory into ROM which is used mostly for constant filter coefficients, and RAM to hold intermediate data. For each of these memories, several data types are available, some are high precision for DSP routines, others are lower precision mainly for control tasks. This choice of data-types is key to the performance of the unit.

The multiply-accumulate (MAC) unit was designed around the time-critical inner-loop functions of the application. The unit has special register connections which allow it to work efficiently with memory-bus transfers. In addition, certain registers may be coupled to perform double precision arithmetic.

Finally, the VLIW format (61 bits) allows for far more parallelism than most commercial DSP's. This is crucial in obtaining the required performance for the MPEG2 and Dolby AC-3 audio standards.

1) Tool Requirements: The tool requirements here are fairly similar to those of the videophone: high-performance compilation, high-level functional validation of C descriptions with the testbench, and instruction set simulation. In addition, source-level debugging and performance profiling were considered essential.

The requirement on compiler performance was for zero execution time overhead in the inner loops, and less than 25% overhead elsewhere. The compiler developed met the 25% overhead target for the noncritical code. Furthermore, the selective use of pragmas in the ANSI C code were used to guide the compiler for the variable to register assignment in the critical inner loops. Under these conditions, the zero overhead target was also achieved [60].

There is an additional requirement for a tool to assist with ASIP architecture exploration. This would be guided by the designer who would propose different instruction set and parallelism configurations, and the tool would provide the resulting execution profiles for the application code on this selection.

2) Key Characteristics of MPEG Audio DSP Case Study: This is a good example of a high-volume multimedia application for which low cost is an essential feature. An analysis of the two main applications (MPEG2 and Dolby AC-3) allowed to design a low-cost dedicated architecture well suited to the critical inner loops. Three key characteristics distinguish this ASIP from conventional DSP's:

- the memory structure, which is developed around the data types needed for the application;
- the ACU design, which includes a rich set of increment/decrement operations, as well as custom connections to the register sets;
- the VLIW format (61 bits), which provides a level

of parallelism that is specifically tuned to the application.

A retargetable compilation environment was successfully used to meet stringent performance constraints. As a result, the software application was written in the C programming language. In order to achieve zero overhead in critical code portions, the inner loops required selective variable to register assignment. Finally, the main requirement for the future is a tool for ASIP architecture exploration.

VII. Conclusion

An increasingly large part of the functionality of embedded applications will be delivered via an embedded processor. There are many reasons for the rise of embedded processors, including the ones listed below.

- Design flexibility: In order to accommodate design errors, late specification changes and future product evolution. This flexibility also reduces design risk and contributes to a shortened design interval by reducing the number of chip iterations. This is the single most important reason cited by designers.
- Design reuse: It is simpler to adapt an existing programmable datapath than to redesign a hard-wired circuit. Design reuse usually leads to shorter development times, which translates to faster time to market.
- Design complexity: The presence of large reconfigurable, programmable building blocks can greatly simplify the design of a significant portion of today's heterogeneous systems-on-a-chip. Furthermore, complex signal processing and/or microcontrol is often better managed in software than in hardware.

Existing commercial microprocessors, MCU, and DSP processors will continue to account for the majority of design starts. However, the Northern Telecom survey shows that ASIP's account for the majority of the volume of the parts shipped. In the survey, ASIP's were used by only one third of the design groups, but these ASIP's represented two thirds of the total volume of shipped parts.

The survey of leading products in multimedia and wireless applications also confirm the importance of ASIP's as a cost-effective design style. Virtually every second- or third-generation GSM terminal uses an ASIP. MPEG2 decoders are based on ASIP's or dedicated hardware. Video games rely on ASIP's, RISC, and dedicated hardware. Finally, 3-D processors are all based on ASIP's or dedicated hardware.

There are two main motivations for using ASIP's. The first to replace an existing standard core for cost or power reduction in a second-generation design. This is the case for GSM terminals for example. First-generation terminals used standard DSP's to get a product to market quickly. An ASIP was developed for the second- or third-generation terminal in order to maintain a competitive advantage in cost, or more importantly, low power.

The second motivation for the use of ASIP's is to replace blocks previously designed in hardware that require more flexibility and to accommodate increased complexity. MPEG video decoders and 3-D processors fall into this category. Today, there is a coexistence of hardware-only solutions to these products as well as ASIP-based solutions. Due to the large performance gap between high-end video processing needs and what is available on standard processors, the evolution path of hard-wired solutions toward a programmable one will be necessarily via an ASIP.

1) Tools for Embedded Software Development: The following items are the most important requirements in this area.

1) High-performance C compilers. While this is especially true for DSP applications, it is also the case for most of the low-cost MCU's (8 and 16-bit). The C programming language is used for only 10% of the application code lines written for DSP's, and 25% of the code lines for MCU. Assembly language reigns elsewhere. For a compiler to be used by a design team, code size overhead must be 20% or less, with performance overhead very close to 0%.
2) Multilevel simulation. This includes C-VHDL functional co-simulation, as well as interactive instruction set level co-simulation with RTL hardware descriptions.
3) Source-level debugging tools with links to in-circuit emulation.
4) Finally, for those teams that choose to use an ASIP approach, there is a need for computer-aided exploration of processor architectures.

The companion paper in this issue [1] presents a survey of tools and approaches that address many of these requirements, with special emphasis on retargetable compilation.

Acknowledgment

The authors would like to thank the following people for their technical contributions: L. Bergher, J.-M. Gentit, and M. Froidevaux of TCEC, M. Harrand and O. Deygas of SGS-Thomson, and T. Rahrer and S. McClennon of BNR/Northern Telecom. The authors also thank J. Carson for the final paper editing and proofreading.

References

[1] G. Goossens, J. Van Praet, D. Lanneer, W. Geurts, A. Kifli, C. Liem, and P. G. Paulin, "Embedded software in real-time signal processing systems: Design technology," *Proc. IEEE,* this issue, pp. 436–455.

[2] P. Paulin *et al*., "Trends in embedded systems technology," *Hardware/Software Co-Design*, M. G. Sami and G. De Micheli, Eds. New York: Kluwer, 1996.

[3] *World Semiconductor Trade Statistics Annu. Rep.*, 1994.

[4] J.-P. Dauvin, J. Olliver, and D. Coulon, *Electronic Components and Their Industry*. Paris: Presses Univ. de France, 1995.

[5] A. De Geus, "Trends in CAD and design: System-on-a-chip," in *EuroDAC Conf.*, keynote address, Grenoble, France, Sept. 1995.

[6] G. House, "Technology trends," in *Europe. Design and Test Conf.*, keynote address, Grenoble, France, Mar. 1996.

[7] M. Gold and A. Bindra, "DSP world rocked to core," (Source: Forward Concepts), *Electron. Engineering Times*, pp. 1, 118, Mar. 18, 1996.

[8] *Rep. Int. Data Corp. (IDC) Study Group,* Apr. 1996.

[9] R. Camposano and J. Wilberg, "Embedded system design," in *Design Automation for Embedded Systems*, vol. 1. Amsterdam: Kluwer, Jan. 1996, no. 1, pp. 5–6.

[10] G. De Micheli, "Hardware/software codesign: Application domains and design technologies," in *Hardware/Software Co-Design*, M. G. Sami and G. De Micheli, Eds. New York: Kluwer, 1996.

[11] I. Gold, "Eight-bit microcontrollers forge ahead," *Electron. Engineering Times*, p. 54, Feb. 1, 1995.

[12] J. Turley, "Embedded CPU's shift to consumer usage," *Microprocessor Rep.*, pp. 13–16, Dec. 25, 1995.

[13] ——, "Multimedia chips complicate choices," *Microprocessor Rep.*, pp. 14–19, Feb. 12, 1996.

[14] ——, "MPEG choices for PC's abound," *Microprocessor Rep.*, pp. 9–12, July 31, 1995.

[15] P. Clarke and R. Wilson, "Philips preps VLIW DSP for multimedia," *Electron. Engineering Times*, p. 1, Nov. 14, 1994.

[16] L. Bergher *et al.*, "MPEG audio decoder for consumer applications," in *Proc. IEEE Custom Integrated Circ. Conf. (CICC)*, Santa Clara, CA, May 1995.

[17] D. Epstein, "Chromatic raises the multimedia bar," *Microprocessor Rep.*, pp. 23–27, Oct. 23, 1995.

[18] K. Otsu, F. Nakagawa, and Y. Sato, "Two-chip MPEG-2 video encoder," *IEEE Micro Mag.*, pp. 51–58, Apr. 1996.

[19] M. Thorn *et al.*, "A high-performance MPEG-audio decoder IC," in *Proc. 5th NASA Symp. on VLSI Design*, 1993.

[20] A. Bindra, "Audio DSP's fit home applications," *Electron. Engineering Times*, p. 64, Dec. 19, 1994.

[21] M. Harrand *et al.*, "A single chip videophone video encoder/decoder," in *Proc. IEEE Int. Solid-State Circ. Conf.*, Feb. 1995, pp. 292–293.

[22] SGS-Thomson Microelectron., "STi1100 videophone CODEC preliminary data specification," Aug. 1993.

[23] Y. Yao, "Competition heats up in 3D accelerators," *Microprocessor Rep.*, pp. 16–23, Mar. 5, 1996.

[24] D. Epstein, "IBM extends DSP performance with Mfast," *Micropocessor Rep.*, pp. 1–8, Dec. 4, 1995.

[25] L. Gwennap, "Intel's MMX speeds multimedia," *Microprocessor Rep.*, pp. 1, 6–10, Mar. 5, 1996.

[26] Y. Yao, "PC graphics reach new level: 3D," *Microprocessor Rep.*, pp. 14–19, Jan. 22, 1996.

[27] J. Turley, "Action-packed fray for newest video games," *Microprocessor Rep.*, pp. 15–20, May 30, 1995.

[28] P. Le Scan *et al.*, "A chip set for 7 kHz handfree telephony," in *Proc. CICC*, San Diego, May 1994.

[29] P. Vanoostende, "Retargetable code generation: Key issues for successful introduction," in *1st Workshop on Code Generation for Embedded Processors*, Dagstuhl, Germany, Aug. 1994.

[30] A. Picciriello, "Italtel activities in ASIP design," presented at *Digital Signal Process. in ASIC's Course*, Unterspremstatten, Austria, Apr. 1995.

[31] D. Weinsziehr *et al.*, "KISS-16V2: A one-chip ASIC DSP solution for GSM," *IEEE J. Solid-State Circ.*, vol. 27, no. 7, pp. 1057–1066, July 1992.

[32] R. Woudsma *et al.*, "EPICS, a flexible approach to embedded DSP cores," in *Proc. 5th Int. Conf. on Signal Process. and Applicat. Technol.*, Dallas, TX, Oct. 1994.

[33] R. Beltman *et al.*, "EPICS10: Development platform for next generation EPICS DSP products," in *Proc. 6th Int. Conf. on Signal Process. and Applicat. Technol.*, Boston, MA, Oct. 1995.

[34] P. Blouet, "DSP shifts to perform for the consumer," *Electron. Engineering Times*, Apr. 17, 1995.

[35] SGS-Thomson Microelectron., "D950-CORE specification," Jan. 1995.

[36] R. Beale, "Custom-fit DSP's suit more needs," *Electron. Engineering Times*, 1992.

[37] M. Gold and J. Yoshida, "GSM fuels IC ramp-up," *Electron. Engineering Times*, p. 1, Nov. 14, 1994.

[38] F. Grosvalet, "Wireless communications oriented DSP's," *Electronique Int.*, p. 25, May 18, 1995.

[39] L. Gwennap, "Multimedia API's spur hardware innovation," *Microprocessor Rep.*, p. 3, Nov. 13, 1995.

[40] B. Emerson and D. Greethham, "GSM's extraordinary growth," *Byte Mag.*, pp. 21–24, Mar. 1996.

[41] M. Slater, "The future of computing platforms," *Microprocessor Rep.*, pp. 15–21, Aug. 21, 1995.

[42] M. Gold and A. Bindra, "EDA, chip makers ally for DSP push," *Electron. Engineering Times*, p. 1, Oct. 24, 1994.

[43] J. Turley and P. Lapsley, "Motorola, TI extend 16-bit DSP families," *Microprocessor Rep.*, pp. 14–15, Mar. 5, 1996.

[44] F. Grosvalet, "A Low-cost DSP optimized for control functions," *Electronique Int.*, p. 20, Feb. 29, 1996.

[45] R. Goering, "Mentor to acquire microtec," *Electron. Engineering Times*, p. 1, Oct. 16, 1995.

[46] Texas Instrum. Press Release, "TI drives DSP time-to-market by doubling development support investment," Dallas, TX, May 6, 1996.

[47] B. C. Cole, "DSP embedded use grows," *Electron. Engineering Times*, pp. 80–82, Apr. 17, 1995.

[48] V. Zivojnovic, J. M. Velarde, and C. Schlager, "DSPstone: A DSP-oriented benchmarking methodology," *Internal Rep.*, Aachen Univ. Technol., Germany, Aug. 1994.

[49] V. Zivojnovic *et al.*, "DSPstone: A DSP-oriented benchmarking methodology," in *Proc. Int. Conf. on Signal Process. Technol. (ICSPAT)*, Dallas, Oct. 1994.

[50] P. G. Paulin, C. Liem, T. May, and S. Sutarwala, "DSP design tool requirements for embedded systems: A telecommunications industrial perspective," *J. VLSI Signal Process.*, vol. 9. Amsterdam: Kluwer, Mar. 1995, pp. 23–47.

[51] P. Paulin *et al.*, "High-level synthesis and codesign methods: An application to a videophone codec," in *Proc. EuroVHDL/EuroDAC*, Brighton, U.K., Sept. 1995.

[52] F. Nacabal, O. Deygas, M. Harrand, and P. Paulin, "Functional validation of software application in embedded systems: A VHDL-C co-simulation approach," in *Proc. EuroDAC (Designer Sessions)*, Geneva, Sept. 1996.

[53] C. A. Valderrama *et al.*, "A unified model for co-simulation and co-synthesis of mixed hardware/software systems," in *Proc. Europe. Design and Test Conf.*, Paris, France, Mar. 1995.

[54] P. G. Paulin, C. Liem, T. May, and S. Sutarwala, "FlexWare: A flexible firmware development environment for embedded systems," *Code Generation for Embedded Processors*, P. Marwedel and G. Goossens, Eds. Amsterdam: Kluwer, 1995.

[55] C. Liem, T. May, and P. Paulin, "Instruction-set matching and selection for DSP and ASIP code generation," in *Proc. Europe. Design and Test Conf.*, Paris, Feb. 1994.

[56] ——, "Register assignment through resource classification for ASIP microcode generation," in *Proc. ACM/IEEE Int. Conf. on Computer-Aided Des. (ICCAD)*, San Jose, Nov. 1994.

[57] C. Liem, P. Paulin, M. Cornero, and A. Jerraya, "Industrial experience using rule-driven retargetable code generation for multimedia applications," in *Proc. Int. Symp. on Syst. Synthesis*, Cannes, France, Sept. 1995.

[58] E. Lee, "DSP architecture survey," *IEEE ASSP Mag.*, vol. 8, p. 88, Jan. 1989.

[59] M. Levy and A. Coyle, "1996 DSP-chip directory," *EDN Mag.*, Mar. 1996.

[60] C. Liem *et al.*, "An embedded system case study: The firmware development environment for a multimedia audio processor," submitted to *Design Automat. Conf.*, Anaheim, CA, June 1997.

Pierre G. Paulin (Member, IEEE) received the B.Sc. degree in engineering physics and the M.Sc. degree in electrical engineering from the Université Laval, Quebec City, Canada, in 1982 and 1984, respectively. He received the Ph.D. degree in electronics engineering from Carleton University, Ottawa, Ontario, Canada, in 1988.

He is a Manager in the Central R&D division of SGS-Thomson Microelectronics in Grenoble, France, and is responsible for the Embedded Systems Technology group. The group's activities include retargetable compilers, debuggers, instruction-set simulators and hardware/software co-design tools for application-specific and standard embedded processors. During 1989–1994, he was with BNR/Northern Telecom, Canada, where he was responsible for embedded software development tools. His research interests are in the fields of hardware/software co-design, embedded systems, compilation, high-level synthesis, and system-level design tools.

Dr. Paulin was awarded the Design Automation Conference (DAC) best presentation award in 1986, and his 1987 and 1989 DAC papers were nominated for best paper awards. His research in high-level synthesis was recognized in "25 Years of Design Automation," a collection of seminal papers that have appeared at DAC over the last 25 years.

Clifford Liem received the B.S. degree in physics from St. Francis Xavier University, Antigonish, Nova Scotia, Canada, and the M.E. degree in electronics from Carleton University, Ottawa, Ontario, Canada, in 1989 and 1991, respectively.

He is currently completing the Ph.D. degree at the TIMA Laboratory, Institut National Polytechnique de Grenoble (INPG) in cooperation with SGS-Thomson Microelectronics on methods and tools for embedded processors. From 1991 to 1994 he held an engineering postion at Bell-Northern Research (now Nortel) where he worked on retargetable compilation, processor modeling, and RTL synthesis. His research interests include optimizing compilers, instruction-set design, and hardware-software co-design.

Marco Cornero received the M.S. degree in electronic engineering and the Ph.D. degree in high-level synthesis from the University of Genova, Italy, in 1991 and 1994, respectively.

In 1995 he joined the Embedded Systems Technology group at the Central R&D division of SGS-Thomson Microelectronics, where he is currently working on compilers for embedded processors. From 1993 to 1995 at the University of Genova on a European project for the automatic synthesis of real time operating systems for embedded applications. During 1991–1993 was with IMEC, Belgium, where he worked on on the Cathedral Second High-Level Synthesis project. His research interests include optimizing compilation techniques, real-time operating systems for embedded applications, and DSP architectures.

François Naçabal received the M.S. degree in microelectronics from the Pierre-et-Marie Curie University, Paris, France, in 1993. He is currently working toward the Ph.D. degree in the Institut National Polytechnique of Grenoble, France, in cooperation with SGS-Thomson Microelectronics. He has been working on hardware-software cosimulation and compilation for embedded software.

His research interests include hardware-software codesign, architecture exploration, and performance analysis for DSP's.

Gert Goossens (Member, IEEE) received the electrical engineering degree and the Ph.D. degree in applied sciences from the Katholieke Universiteit Leuven, Belgium, in 1984 and 1989, respectively.

Since 1996, he has been a General Manager and co-founder of Target Compiler Technologies, Leuven, Belgium. From 1992 to 1996 he headed the Interuniversity Micro-Electronics Centre's (IMEC) Retargetable Software Compilation Research Group, and from 1989 to 1992 he headed its High-Level Synthesis and Asynchronous Circuit Synthesis Research Group. From 1984 to 1989 he was with IMEC's VLSI Design Methodologies Division, where he was an architect of the "Cathedral" high-level synthesis environment for real-time signal processing.

Dr. Goossens received a best paper award at the 26th Design Automation Conference. In 1995 he received the Prize for Sciences of the Royal Belgian Academy of Sciences. Since 1991 he has served on the program committee of several conferences in electronic design automation, including ICCAD, ED&TC, and ISSS.