Dagstuhl seminar 98351: Architectural and Arithmetic support for Multimedia

31.8 - 4.9.1998

organized by

G. Even
P. Kornerup
W. Paul

1 Content

2 Foreword

With the widespread dissemination of PCs and workstations with Internet access in homes and offices, there has been an increasing quest for computational resources to support ever more demanding multimedia applications, like audio, video, virtual reality and games employing advanced graphics. Beyond increasing the performance of CPUs through pipelining and superscalar execution, in the past few years a number of CPU manufacturers have enhanced their instruction sets with special instructions to support such applications. In particular SIMD-types of basic arithmetic operations have been introduced, exploiting parallelism on new and typically shorter data types, to support video decompression and other compute intensive applications. But also special arithmetic instructions like fast reciprocal and root-reciprocal operations are emerging to support special graphics applications. The increasing CPU processing power places extra demands on the overall system architecture, on cache, bus, memory and I/O bandwidth, problems exaggerated by this development.

The purpose of this seminar was to bring together researchers and practitioners from three different topic areas: Multimedia architectures, the general purpose systems and processor architecture area, and the arithmetic design and implementation area. Our invitations were well received, many expressed their enthusiasm about this particular combination of topics. Unfortunately some had to decline participation due to other obligations, in particular some people from industry could not attend due to urgent work, but expressed their hope that this type of event could be repeated later. A total of 29 people attended the seminar, 27 formal talks were given, together with an "open problem session" and numerous informal discussions in smaller groups. The facilities of Schloss Dagstuhl were extensively utilized, even such that a formal conference submission was conceived, prepared and submitted during the week, meeting an imminent deadline.

The talks presented covered a wide spectrum of problems from the three areas, ranging from problem identifications to specific algorithmic or implementation designs, from MPEG video decompression to multimedia and hypertext books (with publication issues), basic arithmetic to implementation of standard functions, floating point arithmetic to signal- and image processing, and issues like pipelining, hardware scheduling, precise interrupts, multi threading, encryption and compiler optimizations.

The organizers would like to express their thanks to the Dagstuhl organization for hosting the seminar, to the staff at the Dagstuhl office for their help in preparing and running the seminar, and to the staff of Schloss Dagstuhl for an excellent servicing of the human needs of the participants during a memorable week.

Also thanks are due to a number of invitees for helping by providing new names to approach for participation, in particular to Bruce Shriver for good contacts and personal efforts in this respect. Finally – but not least – thanks to all participants for interesting and well prepared presentations, lively discussions after talks and during off-program hours, new personal contacts established, and for the general, social being-together at such a wonderful place.

3 Participants

K'Andrea Bickerstaff	Cirrus Logic, Embedded Processor Products Division
	4210 South Industrial Drive, TX 78744 Austin, USA
	Tel: +44-1223-464466, Fax: +44-1223-400-410
	e-Mail: kbickers@crystal.cirrus.com
Norbert Bierlox	Universitaet Karlsruhe, Inst. fuer Angewandte Mathematik
	Kaiserstr 12, Postfach 69 80, D-76128 Karlsruhe, Germany
	e-Mail: norbert.bierlox@ieee.org
Gerd Bohlender	Universitaet Karlsruhe, Inst. fuer Angewandte Mathematik
	Kaiserstr 12, Postfach 69 80, D-76128 Karlsruhe, Germany
	Tel: +49-721-608-2839, Fax: +49-721-695283
	e-Mail: Gerd.Bohlender@math.uni-karlsruhe.de
	URL: http://www.uni-karlsruhe.de/~Gerd.Bohlender/
Javier D. Bruguera	University of Santiago de Compostela, Dept. of Elec. & Comp. Eng.
	E-15706 Santiago de Compostela, Spain
	Tel: +34-981-563-100 ext. 13557, Fax: +34-981-599-412
	e-Mail: bruguera@dec.usc.es
Johnny Choe	Advanced Micro Devices
	5900 E. Ben Whihte Blvd., MS 585, TX 78741 Austin, USA
	Tel: +1-512-602-7400, Fax: +1-512-602-6836
	e-Mail: johnny.choe@amd.com
Marc Daumas	Ecole Normale Superieure de Lyon, Laboratoire LIP
	46 allee d'Italie, F-69364 Lyon. France
	Tel: +33-4-72 72 83 52, Fax: +33-4-72 72 80 80
	e-Mail: marc.daumas@ens-lyon.fr
	URL: http://www.ens-lyon.fr/~daumas/
Alexander Domay	Universitaet Karlsruhe, Inst. fuer Angewandte Mathematik
	Kaiserstr 12, Postfach 69 80, D-76128 Karlsruhe, Germany
	Tel: +49-721-377-934, Fax: +49-721-385-979
	e-Mail: alex.domay@ieee.org
	URL: http://www.uni-karlsruhe.de/~Alex.Domay/
Guy Even	Tel Aviv University, Dept. of Electrical Engineering
	Ram-Aviv, 69978 Tel Aviv, Israel
	Tel: +972-3-640-7769, Fax: +972-3-640-7095
	e-Mail: guy@eng.tau.ac.il

Michael Flynn
Stanford University, Dept. of Electrical Engineering
340 Panama Street, CA 94305-3090 Stanford, USA
Tel: +1-650-723-1450, Fax: +1-650-725-6949
e-Mail: flynn@ee.stanford.edu

Cristina S. Iordache
Southern Methodist University, Computer Science & Engineering
P.O. Box 750122, TX 75275-0122 Dallas, USA
Fax: +1-214-768-3085
e-Mail: cristina@seas.smu.edu

Matthew D. Jennings
North Carolina State University, Dept. of Electr. and Computer Eng.
Room 438 EGRC, Box 7911, NC 27695-7911 Raleigh, USA
Tel: +1-919-513 2013
e-Mail: mdjennin@eos.ncsu.edu

Joerg Keller
FernUniversitaet-GH-Hagen, FB Informatik, LG Technische Informatik II
D-58084 Hagen, Germany
Tel: +49-2331-987-376, Fax: +49-2331-987-308
e-Mail: Joerg.Keller@FernUni-Hagen.de
URL: http://ti2server.fernuni-hagen.de/~jkeller/

Reiner Kolla
Universitaet Wuerzburg, Institut fuer Technische Informatik
LST Informatik V, Zwinger 3, D-97070 Wuerzburg, Germany
Tel: +49-931-312-660, Fax: +49-931-312-662
e-Mail: kolla@informatik.uni-wuerzburg.de

Peter Kornerup
Odense University, Dept. of Computer Science
Campusvej 55, DK-5230 Odense, Denmark
Tel: +45-65 57 23 57, Fax: +45-65 93 26 91
e-Mail: kornerup@imada.ou.dk
URL: http://www.imada.ou.dk/~kornerup/

Ulrich Kulisch
Universitaet Karlsruhe, Inst. fuer Angewandte Mathematik
Kaiserstr 12, Postfach 69 80, D-76128 Karlsruhe, Germany
Tel: +49-721-608-2680, Fax: +49-721-695-283
e-Mail: ae15@rz.uni-karlsruhe.de

Holger Leister
Universitaet des Saarlandes, FB 14 - Informatik, Geb. 45 - Zi. 318
PF 15 11 50, D-66041 Saarbruecken, Germany
Tel: +49-681-302-5557, Fax: +49-681-302-4290
e-Mail: holy@cs.uni-sb.de

David W. Matula	Southern Methodist University, Computer Science & Engineering TX 75275-0122 Dallas, USA Tel: +1-214-768-3089, Fax: +1-214-768-3085 e-Mail: matula@seas.smu.edu
Jean-Michel Muller	Ecole Normale Superieure de Lyon, Laboratoire LIP 46 allee d'Italie, F-69364 Lyon, France Tel: +33-4-72 72 82 23, Fax: +33-4-72 72 80 80 e-Mail: jmuller@lip.ens-lyon.fr
Silvia M. Mueller	Universitaet des Saarlandes, FB 14 - Informatik, Geb. 45 - Zi. 318 PF 15 11 50, D-66041 Saarbruecken, Germany Tel: +49-681-302-5557, Fax: +49-681-302-4290 e-Mail: smueller@cs.uni-sb.de
Wolfgang J. Paul	Universitaet des Saarlandes, FB 14 - Informatik PF 15 11 50, D-66041 Saarbruecken, Germany Tel: +49-681-302 2436, Fax: +49-681-302 4421 e-Mail: paul@cs.uni-sb.de
Thomas Scholz	TU Braunschweig, Institut fuer Datenverarbeitungsanlagen Hans-Sommer-Str. 66, D-38106 Braunschweig, Germany Tel: +49 531 391 37 31, Fax: +49 531 391 45 87 e-Mail: scholz@ida.ing.tu-bs.de URL: http://www.ida.ing.tu-bs.de/people/thomass/
Michael Schulte	Lehigh University, EE & CS Dept. PA 18015 Bethlehem, USA Tel: +1-610-758-5036, Fax: +1-610-758-6279 e-Mail: mschulte@eecs.lehigh.edu URL: http://www.eecs.lehigh.edu/~mschulte/
Peter-Michael Seidel	Universitaet des Saarlandes, FB 14 - Informatik, Geb. 45 - Zi. 306 PF 15 11 50, D-66041 Saarbruecken, Germany Tel: +49-681-302-4130, Fax: +49-681-302-4290 e-Mail: pmseidel@cs.uni-sb.de URL: http://www.cs.uni-sb.de/~pmseidel/
Bruce D. Shriver	GENESIS 17 Bethea Drive, NY 10562 Ossining, USA Tel: +1-914-762-8030, Fax: +1-914-941-9181 e-Mail: shriver@genesis2.com

Brett Tischler	CYRIX Corporation 1351 South Sunset Street, CO 80501 Longmont, USA Tel: +1-(303) 774-5168, Fax: +1-(303) 774-5801 e-Mail: brett.tischler@cyrix.com
Theo Ungerer	Universitaet Karlsruhe, Fakultaet fuer Informatik, Geb. 20.20 , Raum 159 Am Fasanengarten 5, Postfach 6980, D-76128 Karlsruhe, Germany Tel: +49-721-608-6048, Fax: +49-721-370455 e-Mail: ungerer@ira.uka.de URL: http://goethe.ira.uka.de/people/ungerer/
Julio Villalba Moreno	Universitat de Malaga, Dept. of Computer Architecture P. O. Box E4114, E-29080 Malaga, Spain Tel: +34-5-213-2787, Fax: +34-5-213-2790 e-Mail: julio@ac.uma.es
Eberhard Zehendner	Universitaet Jena, Institut fuer Informatik Ernst-Abbe-Platz 1-4, D-07743 Jena, Germany Tel: +49 3641 946385, Fax: +49 3641 946372 e-Mail: zehendner@acm.org URL: http://www2.informatik.uni-jena.de/∼nez/
Dan Zucker	TriStrata Security P.O. Box 8611, CA 94309 Stanford, USA Fax: +1-650-723-0033 e-Mail: dan@tristrata.com URL: http://umunhum.stanford.edu/∼zucker/

4 Workshop schedule

MONDAY, 31.8.98

09:00-09:10	Introduction
09:10-09:55	Dan Zucker, *Improving performance for multimedia*
10:00-10:40	Silvia Mueller, *The impact of hardware schedulers on the cost and performance of processors*
11:15-11:35	Peter-Michael Seidel, *Variable position rounding after multiplication*
11:40-12:00	Julio Villalba Moreno, *Multimedia applications based on rotations*
16:30-17:15	Micheal Flynn, *Rapid evaluation of estimates of high level functions*
17:20-17:50	Reiner Kolla, *IAX: Interval Arithmetic Evaluation*

TUESDAY, 01.09.98

09:00-09:40	Brett Tischler, *Benefits and challenges of integration for low cost multimedia*
09:45-10:30	Bruce Shriver, *Core Technologies in Hardware and Software*
11:00-12:00	David W. Matula, *Reciprocal Functions: Standards and efficient pipelined implementations*
16:30-17:15	Cristina Iordache, *3DNow! reciprocal and root reciprocal instructions: the AMD K6-2 implementation*
17:20-17:50	Marc Daumas, *Expansions: light weight multiple precision method*
20:00-20:45	Bruce Shriver, *Multimedia Technical Books*

WEDNESDAY, 02.09.98

09:00-09:30	Jean-Michel Muller, *A few results on table-based methods*
09:35-10:05	Wolfgang Paul, *Cost effectiveness of Booth recoding*
10:10-10:50	Ulrich Kulish, *Advanced arithmetic for the digital computer design of arithmetic units*
11:10-12:00	Theo Ungerer, *Simultaneous multi-threading and multimedia*
13:30	trip to Trier

THURSDAY, 03.09.98

09:00-09:45 Holger Leister, *Precise interrupts in processors with out-of-order execution*

09:50-10:20 Johnny Choe, *AMD 3DNow! Technology for Multimedia*

10:20-10:55 Thomas Scholz, *Utilizing Memory Transport Units in embedded DSP Arrays for Video Signal Processing*

11:20-12:00 Silvia Mueller, *Hardware scheduler for controlling variable latency functional units*

16:30-17:00 Eberhard Zehendner, *Signed Digit Number Representations for High-Speed Arithmetics*

17:00-18:00 Michael Schulte, *Arithmetic units for multimedia and DSP*

20:00-20:30 Joerg Keller, *Optimizing Arithmetic for Encryption*

20:30-21:00 Matthew Jennings, *Challenges in Compiling for Subword Extension to Microprocessors*

FRIDAY, 04.09.98

09:00-09:30 Peter-Michael Seidel, *Quantitative analysis of IEEE compliant FP implementations*

09:35-10:05 Peter Kornerup, *Fast computation of moments*

10:35-11:05 Cristina Iordache, *Infinitely precise rounding for division, reciprocal, square root, and square root reciprocal.*

11:10-11:40 Guy Even, *A fast IEEE FP adder*

11:40-12:00 Ulrich Kulish, *Shortcomings of Existing Processors and Standards*

5 Abstracts

5.1 Architecture and Arithmetic for Multimedia Enhanced Processors

Daniel F. Zucker, TriStrata Security, Inc. and Stanford University

In the past, displaying video on desktop systems has required high cost special purpose hardware to handle the computationally intensive task of video compression. Recently, special purpose multimedia instruction sets have allowed software-only real time video decompression without extra hardware. This means video functionality can now be handled by a general purpose CPU. With this low cost video capability, the video data type is becoming truly ubiquitous. Most major CPU vendors have adopted this strategy of enhancing a general purpose processor for multimedia applications. Examples include Hewlett Packard's MAX instruction set, Sun's VIS (Visual Instruction Set), and Intel's MMX instructions.

This work investigates similar techniques for applying cost-effective enhancements to a general purpose processor. Using public domain MPEG implementations as benchmarks and trace based simulation, we investigate performance for typical MPEG video decompression applications.

Beginning with a system level breakdown of execution time, we propose techniques to improve execution time in three separate architectural components: I/O, arithmetic, and cache memory. For I/O, we show how applying traditional techniques of I/O cache prefetching can reduce the time for reading compressed video data. For arithmetic, we propose software-only techniques to pack multiple data words into a single floating point operand to achieve SIMD type parallelism. For cache memory accesses, we define a stream cache that can eliminate approximately 80% of cache misses across a range of cache sizes. Finally, we show how the stream cache motivates a technique for inserting software prefetch instructions that can improve performance with little or no extra hardware cost. Combining these techniques, we can achieve a speedup of 1.5x for a typical MPEG application.

5.2 The Impact of Hardware Scheduling Mechanisms on the Performance and Cost of Processor Designs

Silvia M. Mueller, (H. Leister, P. Dell, N. Gerteis, D. Kroening), University of Saarland

Current processors comprise multiple functional units which can work in parallel. The latency of the functional units varies by a lot. For a better performance and hardware utilization, many designs therefore allow instructions to overtake each other (*out-of-order execution*). The instruction execution and the resources are then governed dynamically by hardware schedulers, most of which are based on the result shift register, the Scoreboard and the Tomasulo Algorithm.

The major difference between the schedulers is how early in the processing they allow instructions to leave the in-order execution, and their support for renaming and forwarding. In case of *out-of-order completion*, the instructions and their operands are passed to the functional units (FU) in-order, but the results might be provided in a different order due to the non-uniform latencies of the FUs. Whereas with *out-of-order dispatch*, the execution of the operation on a FU is already started out-of-order. In-order execution is usually enforced by a result shift register.

In combination with a reorder buffer, the result shift register enables out-of-order completion. The Scoreboard and the Tomasulo scheduler support out-of-order dispatch; the latter scheduler is considered to be more powerful because of its renaming capabilities.

Since hardware schedulers are widespread nowadays, one would expect plenty of studies which quantify the impact of the different type of schedulers on the performance and cost (gate count) of processors. Papers rather focus on the impact of design changes, i.e., for a given scheduler, they study the speedup gained by varying the degree of superscalarity, the forwarding mechanism, or the number of buffers. However, one gets the notion that the more flexible the execution of the instructions is, the higher is the performance improvement of the scheduler. This would imply that out-of-order dispatch is more powerful than out-of-order completion, and that both concepts are more powerful than in-order execution. In this talk, we check this hypothesis and quantify the factor to be gained in cost and performance when switching from one scheduler to another.

On a single-issue design, the hardware schedulers increased the cost of the processor (CPU and 16KB cache) by about 20%, and the cycle time remained the same. Out-of-order completion reduces the CPI and run time by 20%, and the Tomasulo scheduler reduces it by another 25%. That conforms with the original hypothesis, but for the Scoreboard, the matter is quite different. Despite its out-of-order dispatch capability, the Scoreboard scheduler turns out to be 75% slower than the out-of-order completion scheduler, and for most of the benchmarks, it is even significantly slower than the in-order design. The talk also identifies the aspects which are responsible for the poor performance of the Scoreboard. Two thirds of the performance loss are due to the lack of result forwarding. For each functional unit and result register, the original Scoreboard only supports a single outstanding instruction, and the Scoreboard also stalls on false dependencies. That accounts for the other third of the performance loss.

5.3 Variable position rounding

Peter-Michael Seidel, University of Saarland

In this talk the design of an IEEE compliant floating-point multiplier that computes the correctly rounded result for all representable floating-point values is presented. This multiplier can be implemented in 2 clock cycles, whenever its adder tree fits in one clock cycle. The multiplier operates on a normalized format with the property, that the LSB of the significand could be placed at each position of the significand representation. As rounding has to be done at the LSB of the significand, the use of the normalized format complicates the multiplication rounding for denormal results. We solve this problem by an injection based rounding technique. According to the rounding position, the rounding injection and fix bits are adjusted in parallel to the multiplication, so that cycle time and latency is not enlarged due to the rounding of denormal results.

For the same operation recent multiplier designs need at least 4 clock cycles including the pre- and post-processing for denormalized operands and multi-step rounding for the support of different precisions.

5.4 Multimedia applications based on rotations

Julio Villalba Moreno, University of Malaga

In this paper we present the formulation in terms of rotations of some image processing and communication applications. Rotations can be efficiently implemented by means of the CORDIC algorithm. First we give a brief overview of the main researching areas of our group. We are working with in place architectures for signal processing fast transforms. We are also working with variable and high precision elementary functions computation based on both digit–by–digit algorithms and lookup table algorithm. We have designed a CORDIC processor for variable precision which supports a new algorithm to perform the interval sine and cosine functions with time results close to point function computation time.

The first image processing algorithm which we have obtained a formulation in terms of rotations is the Hough Transform. The formulation in terms of rotations has been obtained by splitting the initial interval $[0, \pi]$ into two subinterval $[0, \pi/2]$ and $[\pi/2, \pi]$, and taking into account the symmetry properties of the angles which differ by $\pi/2$. Therefore, it can be implemented using a CORDIC rotator in circular coordinates. This formulation leads to two independent Hough subspaces which allows exploiting the inherent parallelism of this transform. Similarly, we can obtain four independent Hough subspaces when the angles which differ by $\pi/4$ are considered. Since the rotation angles are known beforehand, it is possible to precompute the coefficients, eliminating the hardware corresponding to the z coordinate. We have designed a full radix–4 CORDIC processor which performs the Hough Transform.

The second application presented is the rendering based on rotations. One of the highest computational cost stages corresponds to the linear transformations involved. For the viewing system proposed, we need to obtain the coordinates of a point in the view coordinate system from the coordinates of that point in the world coordinate system. The transformation matrix is very complex, but it can be derived in three steps: translation, rotation through $90 - \theta$, and rotation through $180 - \phi$. Consequently, it is possible to compute the transformation matrix by means of a CORDIC rotator. Finally, we present the formulation of the QAM modulation in terms of rotations.

A lot of algorithms used in multimedia applications can be reformulated in terms of rotations. CORDIC–based rotators need two parallel adders and two parallel shifters to be implemented efficiently. In the other hand, multimedia extensions of the last generation microprocessor manage parallel additions and shifts. Therefore, it does not seem too complicated that the future version of the multimedia extensions of the processors can incorporate the CORDIC algorithm (or a restricted version of it) to speed multimedia applications up.

5.5 Adaptive Arithmetic for HLF computation

Michael T. Flynn, Stanford University

A multiplier is a functional unit that takes two known operands X and Y and produces a product P. A number of operations resemble (or can be made to resemble) an inverse multiply, where (say) X and P are known and Y is unknown. Such operations include

- reciprocal ($1/X = R$; $XR = 1$; R is unknown)
- quotient ($Y/X = Q$; $QX = Y$, Q is unknown)
- square root ($\sqrt{X} \cdot \sqrt{X} = X$; $\sqrt{X}$ is unknown)

Other operations f(X), can be put into a form of an inverse multiplication by differentiating the function, f(X), so that it can be put into multiplicative form e.g.

$$\begin{aligned} w(X) &= ln(v(X)) \\ w(X)' &= v(X)'/v(X) \\ w(X)' \cdot v(X) &= v(X)' \end{aligned}$$

where $w(X)$, $v(X)$ are binary bit polynomials in 2^n with $(0, 1)$ coefficients. Similarly:

$$d(sin^{-1}(X))/dX = 1/\sqrt{1 - X^2}.$$

These inverse problems can be solved by writing a series of simultaneous bit equations defining each product bit, then simultaneously back solving the equations for the unknown multiplier bit values. The resultant equations can be reduced and manipulated, so that they consist of a series of positive bit variable terms (b_i logical op b_j) which are to be summed. These can be summed simply in the partial product summation tree of an ordinary multiplier. A 53 b multiplier array provides at least 2 b accuracy for reciprocal and 16 bits for square roots. Other functions are accurate to $8 - 20$ bits.

5.6 IAX - An Interval Arithmetic Extension

Reiner Kolla, (Ana Vodovipec, Fuergen Wolff v. Endenberg), University of Wuerzburg

The advantage of interval arithmetic is, that we can always keep track of precision. However, the memory overhead is a factor of 2 and the runtime overhead is much more than a factor of 2 if interval arithmetic is only supported in software.

In this talk we demonstrate, that it is easy to extend existing architectures and floating-point units to support interval arithmetic. In particular we show that we can have single precision interval arithmetic at floating-point double precision performance and cost by a small amount of hardware overhead.

5.7 Benefits and Challenges for Integration in Low Cost Multimedia

Brett Tischler, Cyrix

Today, low cost multimedia solutions do not have adequate performance. To lower the cost of the system, integration provides optimal usage of resources and provides a balanced architecture. Cyrix's MXI integrated processor integrates an x86 processor, 3D/2D/MPEG2 graphics and memory controller functions on a single die. This provides the benefits of UMA, low latency to system memory, SIMD floating point, and balanced partitioning.

In the future higher levels of integration will be necessary to meet the needs to low cost markets. This includes methods to optimize on chip resources between unrelated units, provide special purpose programmable engines and leverage the latest in high speed data path design.

5.8 Core Technologies in Hardware and Software

Bruce Shriver, Genesis 2, Inc and University of Tromso

Continued, significant advances in technology have dramatically affected the computer industry itself, particularly in the past 5-6 years. Legacy systems of only a decade ago have largely been transformed into interconnected networks of networks of systems forming the Internet and corporate intranets based on clusters of servers. Much of this transformation has been fueled by relentless performance improvements in high performance microprocessors. There are also fundamental changes in the nature of telecommunications and the services it provides. In this talk, several core technologies are identified that require continued research and development laboratories in order to be competitive in this changing environment. Examples are given of some of the myriad of open problems in this area. This talk explores these technology changes and the impacts they are having on the globalization of the computer industry. Implications about curricula, research, development, and technology transfer in universities will also be discussed.

5.9 Reciprocal Functions: Multimedia Applications Standards and Efficient Pipelined Implementations

David W. Matula, Southern Methodist University, Dallas

Popular multimedia applications such as 3D graphics in computer games have created a need for hardware implementations of floating point reciprocal and root reciprocal functions. The design challenge is to provide fast single precision reciprocal functions satisfying appropriate rounding standards. This paper argues that ulp accuracy, monotonicity, and ulp accurate differences should serve as appropriate minimal standards for these floating point functions. Piecewise linear approximations are shown to satisfy these reciprocal function standards. Several closed form finite precision linear and quadratic interpolations are derived which allow standardized single precision reciprocals to be pipeline computable with a table assisted multiply cycle and from zero to two adds. Our preferred solution demonstrates that a 3 1/2 Kbyte table and a single 29x35 multiply is sufficient to obtain a one ulp, monotonic single precision reciprocal function.

5.10 AMD K6-2 Reciprocal and Square Root Reciprocal Instructions: Performance Evaluation and Analysis

Cristina Iordache, (David W. Matula), Southern Methodist Univeristy, Dallas

Reciprocal and root reciprocal functions at “half” and IEEE single precision formats are specified in the AMD 3DNow instruction set. Implementations in the recently released AMD K6-2 microprocessor are analyzed herein by exhaustive computation and timing loops to ascertain the accuracy and monotonicity properties of the output and throughput/latency cycle counts. Periodicities in stepwise function output were observed and employed to construct an underlying bipartite table that can serve as the core of the respective reciprocal function outputs. The recommended RISC instruction macros generated single precision reciprocals and root reciprocals accurate to a unit in the last place. However, the root reciprocal functions failed to satisfy the desirable monotonicity property typically implemented as an industry standard for elementary functions on x86 floating point units. Reasons for the failure are provided and an adjusted table is shown to satisfy the monotonicity standard.

5.11 Expansions: Lightweight multiple precision arithmetic

Marc Daumas, CNRS, CiP, EWS Lyon

In modern computers, the floating point unit is the part of the processor delivering the highest computing power and getting all the attention from the design team. Performances of any multiple precision application will be dramatically enhanced by an adequate usage of the floating point expansions. We present in this work some new multiplication algorithms faster and more integrated than the stepwise algorithms proposed earlier. We have tested these new algorithms on an application that computes the determinant of a matrix. In the absence of overflow or underflow, the process is error free and possibly more efficient than its integer based counter part.

5.12 Authoring a Multimedia Technical Book: Design and Implementation Considerations

Bruce Shriver, Genesis 2, Inc and University of Tromso

A new set of book publishing and distribution technologies has recently emerged. Among them are electronic books, paperless publishing systems, the Web (both the Internet and intranets), CD-ROMs, and DVDs. Using a newly released book, ”The Anatomy of a High-Performance Microprocessor: A Systems Perspective,” by Shriver and Smith as a case study, this talk discusses various issues that arise in authoring multimedia books, particularly those intended for small, technical audiences. After a brief demonstration of the electronic version of the book, content development (e.g., the hypertext version of the book itself, video clips, audio clips, technical articles from professional society periodicals and conferences, material from vendors and the trade press, technical presentations, and simulators), multi-purposing source material, authoring tools, indexing and searching documents, time commitments, protection mechanisms, and copyright issues will be discussed. A set of recommendations will be presented to those who are embarking on similar projects.

5.13 A few results on table-based methods

Jean-Michel Muller, Ecole Normale Superieure de Lyon

Table methods are frequently used for computing functions (reciprocals, square roots, transcendentals) where precision is low. We recall some conventional methods, then we give a generalization of the bipartite table method (originally due to Das Sarma and Matula). For instance, we can evaluate a function f with n-bit accuracy (provided f is regular enough) using 5 look-ups in $2n/5$ bit address tables, then a final addition. More can be found at the URL: www.ens.lyon.fr/∼jmmuller

5.14 On the complexity of Booth Recoding

Wolfgang J. Paul, (Peter-Michael Seidel), University of Saarland

We formalize and prove the folklore theorems that Booth recoding improves the cost and cycle time of standard multipliers by certain constant factors. We also analyze the number of full adders in certain 4/2-trees.

5.15 Advanced Arithmetic for the Digital Computer Design of Arithmetic Units

Ulrich W. Kulisch, University of Karlsruhe

Advances in computer technology are now so profound that the arithmetic capability and repertoire of computers can and should be expanded. Of the elementary floating-point operations +,-,x,/ nowadays is required that for any choice of operands, the computed result coincides with the rounded exact result of the operation. Advanced computer arithmetic extends this accuracy requirement to all operations in the usual product spaces of computation; the real and complex vector spaces as well as their interval correspondents. This enhances the mathematical power of the digital computer considerably. A new computer operation, the scalar product, is fundamental for the development of advanced computer arithmetic. The talk discussed the design of arithmetic units for advanced computer arithmetic. Scalar product units were developed for different kinds of computers like Personal Computers, Workstations, Mainframes, Super Computers and Digital Signal Processors. The new expanded computational capability is gained at modest costs. It is obtained by putting a methodology into modern computer hardware which was already available on old calculators before the electronic computer entered the scene. In general the new arithmetic units increase both, the speed of computation as well as the accuracy of the computed result. The circuits developed in the paper show that there is no way to compute an approximation of a scalar product faster than the correct result. A collection of constructs in terms of which a source language may accommodate advanced computer arithmetic is described in the paper. The development of programming languages in the context of advanced computer arithmetic is reviewed. The simulation of the accurate scalar product on existing, conventional processors is discussed. Finally, the theoretical foundation of advanced computer arithmetic is reviewed and a comparison with other approaches to achieve higher accuracy in computation is given. Shortcomings of existing processors and standards are discussed.

5.16 Multimedia and Simultaneous Multithreading

Theo Ungerer, University of Karlsruhe

We survey microarchitecture models for a general-purpose processor with multimedia enhancements. Such enhancements are provided by one or more multimedia units that employ subword parallelism (data parallel "SIMD" instructions), saturation arithmetic and additional arithmetic, masking, selection, reordering and conversion instructions.

We start with a wide-issue superscalar processor model, enhance it by multimedia units, by the simultaneous multithreading technique, and by an additional on-chip RAM storage.

A multithreaded processor, in general, stores multiple thread contexts in different register sets on the processor chip and multiplexes the functional units between the contexts. Context switching is very fast. Thereby all kinds of latencies can be bridged by switching to another thread. A simultaneous multithreaded processor, in particular, can issue instructions from several threads simultaneously.

Our workload is a multithreaded MPEG-2 video decompression algorithm that extensively uses multimedia units. The simulations showed that a single-threaded, 8-issue maximum processor (assuming an abundance of resources) reaches an IPC (instructions per cycle) count of only 1.60, while an 8-threaded 8-issue processor is able to reach an IPC of 6.07. A more realistic processor model reaches an IPC of 1.27 in the single-threaded 8-issue vs. 3.21 in the 8-threaded 8-issue model. So, the generalizing conclusion is that an 8-threaded 8-issue processor may yield up to threefold performance increase over the single-threaded 8-issue model.

5.17 Precise interrupts in Processors with Out-of Order Execution

Holger Leister, University of Saarland

Current microprocessors achieve high performance by exploiting instruction level parallelism. Several instructions can be executed in the functional units in a single machine cycle. Since the functional units have different latencies, instructions can complete out-of-order and therefore it is no longer guaranteed that the status of the processor is updated in sequential program order. Up-to-date microprocessors provide fast I/O, virtual memory and support the full IEEE floating point standard. All that calls for a powerful mechanism for the support of precise nested interrupts. The design of such a mechanism is one of the main design challenges in a processor. It becomes even more complex in processors with out-of-order execution. In order to make the interrupts precise, the processor must be recovered from the out-of-order state to the in-order state. In commercial microprocessors, the problem of combining precise interrupts and out-of-order execution was first solved in 1995. In this talk, we present several mechanisms for the solution of this problem: the reorder buffer, the future file and the history buffer.

5.18 AMD 3DNow! Technology for Multimedia

Johnny Choe, Advanced Micro Devices, Austin

A set of computational modules has been analyzed for the performance measurement of 3DNow! Technology. Each module is vectorized through hand-optimization techniques as well as a vectorizing compiler. The speedups from the optimization indicate that the compiler offers as much performance as the hand-optimization. Our experiments also indicate that the hand-optimization is relatively easy to accomplish through the in-line assembly of a high-level programming language such as C or C++. Since the in-line assembly is unable to directly access the registers allocated for C variables, the compiler often introduces an overhead in the in-line assembly block. This limits the performance gain from the hand-optimization. Employing rigorous and exhaustive vectorization techniques, the compiler offers performance that is competitive to, or even better than the hand-optimization. With such compiler techniques, the 3DNow! technology provides multimedia applications a speedup ranging from 1.2 to 1.6 with CD quality audio and high-quality image.

5.19 Utilizing Memory Transport Units in embedded DSP Arrays for Video Signal Processing

Thomas Scholz, Technical University of Braunschweig

Due to increasing resolution and throughput rates of emerging standards for broadcast television, algorithms from the domain of video signal processing are very demanding concerning computational performance. Therefore, even if it is possible to utilize arrays of high performance signal processors, the computational resources of the processors are still a bottleneck of systems in this domain, while the memory throughput rates of these processors exceeds the demands by an order of magnitude.

Within this scenario, in order to meet hard time constraints, any kind of blocking of the computational units due to memory stalls is not tolerable. By utilizing memory transport units of modern DSP that operate concurrently to the processor core it is possible to hide all memory transfers between local and external memories and thus to avoid any delays in memory access. The presented technique has been adapted from the parallelizing SUIF compiler where it is called "Loop Tiling". As an important property this technique supports multi processing.

5.20 A Hardware Scheduler for Controlling Variable Latency Functional Units

Silvia M. Mueller, University of Saarland

In order to improve the performance of functional units, one tries to make the common case fast and simple. That often results in functional units with variable latency, i.e., the latency depends on the operation and the operands. There are several sources for variable latency, like extra cycles for handling denormal operands and arithmetical exceptions, or exploiting the characteristics of an operation, as it is done in the multi-path adder of Flynn and Oberman.

Those functional units (FUs) introduce special scheduling problems. The global scheduler, which governs the interaction between the FUs and the other data paths, does not know the latency of an instruction during dispatch, but that is a solved problem. The schedulers which are based on the Tomasulo Algorithm already account for that. However, there is also a scheduling problem within an FU with variable latency due to structural hazards. There are several paths through such an FU; these paths have different latency, but they still share some of the hardware resources, i.e., instructions might compete for circuits and busses. Thus, a scheduler has to ensure that there are no contentions on the busses and that no data are lost.

In the talk, we present a scheduling mechanism which can control the data flow within any variable latency FU, and which can be implemented with reasonable hardware overhead. If several instructions compete for a resource, the scheduler always picks the oldest instruction. It is shown that such an arbitration mechanism is well defined, and that it never increases the latency of the FU. With simple FIFO queues, this latency cannot be guaranteed.

5.21 Signed Digit Number Representations for High Speed Arithmetic

Eberhard Zehendner, University of Jena

Signed digit number representations are frequently used for speeding up arithmetic operations, for instance in multiplier recoding, as an intermediate format for division and elementary functions, and in carry-free addition or subtraction. Designing arithmetic modules for such representations is far more difficult than for standard representations since the behavior of the circuit is not fixed by the problem specification and thus its derivation constitutes a mandatory part of the design process. Even the choice of the representation at the bit level may be open beforehand.

Former work on adders for signed digit number representations deliberately constrained the design space by prescribing an inner structure for the adder and some very restrictive conditions for the cells of this structure. In our work we drop all these constraints except for some regularity. To master the huge design space thus opened we develop a theory that explicitly characterizes all possible solutions. This theory can be used to construct adders, to enumerate adequate subsets of the design space, and to devise implementation-invariant properties of the adder. For instance, for each representation of the digits we can determine the minimal fan-in of the boolean functions calculating the result bits. The fact that the minimal fan-in depends heavily on the chosen representation emphasizes the importance of choosing the representation with care.

5.22 Arithmetic Units for multimedia and Digital Signal Processing

Michael J. Schulte, Lehigh University

This research presents the design and analysis of arithmetic units for multimedia and digital signal processing applications. Designs for constant correction and variable correction multipliers, and high-speed reciprocal and reciprocal root units are presented. By allowing the computer result to differ from the true result by at most one unit in the last place, a significant reduction in area, delay, and power consumption is achieved.

5.23 Optimizing Arithmetic for Encryption

Joerg Keller, (A. Mahraous), Open University Hagen; (I. Biehl), University of Darmstadt

As software encryption of multimedia streams with high bandwidth does not deliver enough throughput, one needs hardware support. Programmable devices are more favorable than custom devices because they are cheaper and can be used for acceleration of other applications as well, thus further amortizing cost.

The slow speed of programmable devices will be less of a problem if one needs to encrypt only for a particular key and this can optimize the arithmetic; e.g. a multiplication with part of the key turns into a multiplication with a constant.

The rationale behind the assumption of a fixed key, is the following. In public key schemes, a key server will offer the descriptions of optimized encryption hardware instead of the corresponding keys. In secret key schemes, instead of generating a session key, some time in advance one generates the corresponding optimized hardware. For decryption, similar schemes are possible.

Consideration of popular encryption algorithms show that IDEA and RSA are promising candidates while DES is not.

Our Future work includes

- experiments to find lower bounds on performance increase,
- finding a compact format independent of the type of device used,
- investigating optimized software encryption in Java.

5.24 Challenges in Compiling for Subword Extension to Microprocessors

Matthew Jennings, North Carolina State University

While subword extension to general-purpose processors provide for improved performance in executing multimedia code, the use of such instructions is difficult. Examples of subword extensions to general-purpose processors include HP's MAX, Intel's MMX, AMD's 3DNow!, UltraSparc's VIS, Motorola's AltiVec, and MIPS's MDMX. In general today, the use of these instructions is limited to game applications in which hand assembly programming is used for loop kernels. Also, some compilers today use these instructions through macro level calls in a high level language. The use of macro calls in a high level language requires a programmer to work at an "assembly level" to take full advantage of subword execution. The difficulty in programming results in the use of subword extensions only for select applications in which the programming effort is warranted.

With recent advances in compiling for instruction level parallelism (ILP), it is natural to hope for a compiler to utilize subword instructions mostly transparently to the high level programmer. It may even be possible for a compiler to outperform an experienced assembly level programmer as many sources of ILP are not obvious to a programmer. Challenges in compiling for subword extensions include determining the amount of precision each variable requires and identifying sources of single instruction multiple data (SIMD) parallelism inherent to subword extensions. In determining the amount of precision required either language extensions can be used (to allow the programmer to explicitly declare data size) or the compiler can determine data size by context. Requiring a compiler to determine data size without programmer input seems difficult. For determining sources of SIMD parallelism, existing vectorization techniques could be leveraged.

With successful compilers for exploiting subword extensions, their use could expand to areas other than multimedia games. For example, subword extensions could be useful in the general-purpose domain for 8-bit character code and also for executing legacy 32-bit code on new 64-bit architectures. In addition their use in multimedia applications could be expanded and made easier.

5.25 Quantitative analysis of IEEE compliant floating-point implementations

Peter-Michael Seidel, University of Saarland

In this talk a quantitative analysis of various IEEE compliant floating-point (FP) implementations is presented. In particular, 3 different rounding architectures are compared and the use of half-sized multiplier adder-trees regarding cost and performance is analyzed. In combination with this, we regard five different division implementations. This combines to a comparison of 30 different FPU versions.

The analysis of the FPUs are performed integrated in a pipelined RISC-architecture under the same conditions. To analyze the systems performance, a trace driven runtime-simulator on the SPECfp92 floating-point benchmark suite is used.

Among the different FPU variants the costs differ by a factor of 2.16 and the performance by a factor of 1.42. When looking separately at the different design options, the rounding optimization turns out to be the most efficient. In this way the largest speed-up with only little additional cost can be achieved.

5.26 Computing Moments by Prefix Sums

Peter Kornerup, (Feng Zhou), Odense University

Moments of images are widely used in pattern recognition, because in suitable form they can be made invariant to variations in translation, rotation and size. However the computation of discrete moments by their definition requires many multiplications which limits the speed of computation. In this paper we express the moments as a linear combination of higher order prefix sums, obtained by iterating the prefix sum computation on previous prefix sums, starting with the original function values. Thus the p'th moment $m_p = \sum_{x=1}^{N} x^p f(x)$ can be computed by $O(N{\cdot}p)$ additions followed by p multiply-adds. The prefix summations can be realized in time $O(N)$ using $p+1$ simple adders, and in time $O(p{\cdot}\log N)$ using parallel prefix computation and $O(N)$ adders. The method of prefix sums can also be used in the computation of two-dimensional moments for any intensity function $f(x, y)$. Using a bit-serial addition architecture, it is sufficient with 13 full adders and some shift registers to realize the 10 third order moments $(m_{00}, m_{01}, m_{10}, m_{02}, m_{20}, m_{12}, m_{21}, m_{03}, m_{30})$ for a 512*512 size image at the TV rate. In 1986 Hatamian published a computationally equivalent algorithm, based on a cascade of filters performing the summations. Our recursive derivation allows for explicit expressions and recursive equations for the coefficients used in the final moment calculation. Thus a number of alternative forms for the moment computation can be derived, based on different sets of prefix sums. It is also shown that similar expressions can be obtained for the moments introduced by Liao and Pawlak in 1996, forming better approximations to the exact geometric moments.

5.27 Infinitely Precise Rounding for Reciprocal, Quotients, Square Roots and Square Root Reciprocals

Cristina Jordache, (David W. Matula), Southern Methodist University, Dallas

In order to round an approximation the same way as the infinitely precise result, the correct round and sticky bits must be obtained. We present theoretically proven upper bounds on the minimum accuracy needed for each of these functions, so that infinitely precise rounding is achievable. While the bounds for division and square root are tight (∼2p), the accuracy needed in practice for the root reciprocal appears to be lower than the proven bound of 3p-1. We also show the rules based on which hardware implementable rounding algorithms can be developed.

5.28 A fast IEEE FP adder

Guy Even, Tel Aviv University, (Peter-Michael Seidel), University of Saarland

We present an algorithm for IEEE floating-point addition. The latency of the addition algorithm for double precision is roughly 24 logic levels, not including delays of latches between pipeline stages. The algorithm accepts normalized numbers, supports all four IEEE rounding modes, and outputs the correctly rounded sum/difference in the format required by the IEEE Standard. The presentation of the algorithm is technology independent and can serve as basis for evaluation and comparison with other floating-point addition algorithms.

5.29 Shortcomings of Existing Processors and Standards

Ulrich Kulish, University of Karlsruhe

The talk was supposed to draw attention to a very interesting and important article by John Gustafson: Computational Verifiability and Feasibility of the ASCI Program (THEME ARTICLE, March 1998, IEEE Journal: Computational Science & Engineering). The Advanced Strategic Computing Initiative is a $ 0.2 billion per year federal program carried out at major US laboratories (Sandia, Los Alamos, Lawrence Livermore). Page 43 of John Gustafsons article shows the side bar: "We would do well in the ASCI program to echo the fervor of the Germans for numerical verifiability" and on the same page he says: "Germany, specifically the University of Karlsruhe, remains the center of activity on interval arithmetic and verifiable numerical computation." Then he claims that their methods and language PASCAL-XSC are extremely slow. The talk tried to explain that not the methods but existing processors which are not designed for verified computing are responsible for this slowness. Three major shortcomings of existing processors and standards have been discussed: 1. The IEEE arithmetic standard separates the rounding from the operations. This slows down interval arithmetic on existing processors unnecessarily by a factor between 8 and 50 in comparison with floating-point arithmetic. 2. No one of the existing processors that perform IEEE arithmetic delivers the double length product to the outside world. This slows down an optimal scalar product by a factor between 50 and 100, while a hardware support would speed it up by a factor around 4. 3. The IEEE arithmetic standard does not provide a reasonable interface to the programming languages. Thus arithmetic operations for intervals and for those in the product spaces require clumsy and slow function calls.