Abstracts of the 1989 SIGForth Workshop on Real Time Software Engineering[1]

February 17 – 19, 1989
Austin, Texas

Systematic High-Level Interrupts in Forth

Bapi Ahmad
Druma Incorporated
6488 Highway 290 East, E103, Austin, Texas 78723
Ashok Adiga
Dept of Computer Sciences
University of Texas at Austin, Austin, Texas 78712

This paper presents a high-level abstraction mechanism for managing interrupts in Forth. The abstraction mechanism allows interrupts to be treated as named entities and allows associated interrupt service routines to be manipulated within Forth definitions in a systematic manner. Service routines may be defined using either low-level code or high-level colon definitions. Multiple service routines may be declared and selectively associated with an interrupt based on program execution state.

The paper discusses data structures and provides definitions for creating and manipulating interrupt service routines. Compiler extensions to implement these definitions are described. Examples along with relevant implementation details are also provided. Although discussions in the paper are in the context of Forth, the concepts are relevant to other computer languages and systems.

Forth Interface to MS-DOS Interrupts

Robert H. Davis
3400 Greencastle Rd., Burtonsville, Md. 20866

A set of words for a Forth implementation on INTEL™ 8086 series microprocessors under the MSDOS operating system is described which allows high-level definition of tasks to be executed in response to hardware or software interrupts. A defining word **EVENT** creates a symbolic interface to one of the hardware interrupt vectors in page zero. Any Forth word may then be attached to the interrupt event to be executed each time the interrupt occurs. The mechanics of saving the processor state and constructing a valid Forth stack environment is automatically performed by the code.

A 16 Bit Forth Model for a 32 Bit Addressable Host

Robert H. Davis
3400 Greencastle Rd., Burtonsville, Md. 20866

An implementation model of a 16 bit Forth system is described for hosts with fast, efficient access to a preferred 64 kilobyte memory space and less efficient access to a larger memory space addressable by 32 bit pointers. The memory model enforces a strict separation of the Forth virtual machine implementation and the memory space used by that machine. The advantages of such a model are less necessity for implementations on different hosts to deviate from a standard abstract high-level definition of the Forth language, and therefore more transportable code. The model allows efficient use of the underlying 16 bit hardware together with a consistent use of the larger memory space available. The model has been implemented on the INTEL™ 80x86 microprocessors under the MS-DOS™ operating system. The model should be inherently applicable to Forth engines such as the Harris RTX2000™.

[1]The SIGForth '89 Proceedings are available from: ACM Order Department, P.O. Box 64145, Baltimore, MD 21264. Prices are $11.25 for SIGForth and ACM members, and $15.00 for non-members.

Creating Conditional Transform Words

Gary Feierbach
Intergraph Corp., Advanced Processor Div.,
2400 Geng Rd., Palo Alto, CA. 94303

This paper demonstrates how Forth words may be created that will transform acquired data only when data changes. This is helpful in reducing processor load when many slowly changing measurements must go through complex transformations.

Forth in the USSR

Lawrence P. Forsley
Institute for Applied Forth Research, Inc.
70 Elmwood Avenue, Rochester, New York 14611

There is a growing Forth community in the USSR with much of its interest focused through Dr. Sergei Baranoff of the USSR Academy of Sciences and the Leningrad Institute for Informatics. Dr. Baranoff published the first text on Forth in Russian last year and watched 100,000 copies sell out in 2 weeks. This is indicative of a potentially large demand for Forth in the USSR which has been traditionally weak in computer hardware resources. However, Soviet scientists and engineers may be more than an equal match for their Western counterparts in terms of training and available thinking time. It may not be a coincidence that the long Russian winters have given us poets like Pasternak and authors like Tolstoy. Forth is a particularly poetic programming language.

Object Oriented Thinking in Forth

Lawrence P. Forsley
Institute for Applied Forth Research, Inc.
Rochester, New York 14611

The object-oriented programming paradigm, of classes, instances of classes and methods allowing execution of class specific functions has become a powerful programming paradigm. Originating with Smalltalk 80, and now leaking into Lisp (Flavours, Loops), C++ and ADA, it has gained attention within Forth circles. Although several Object oriented extensions have been implemented by Davis, Duff and Pountain, this concept has been latent within Forth since it's inception. This paper will briefly examine Forth's properties which lend themselves easily to this programming paradigm.

A Forth Neural Net Controller

Paul Frenger M.D.
A Working Hypothesis, Inc.
820 Threadneedle #248, Houston, TX 77079

This paper describes the use of a high integration Forth microcomputer to control a system of analog "neurons" in a neural net computer. The microcontroller performs many essential functions: user input and output; mass storage; system setup; saving of current state; restoration of saved state; network training; and problem execution. A single Forth microcomputer can control multiple analog neurons. This technique may be used to create a wide variety of neural net architectures.

The Design of a Real-Time Multi-tasking Operating System Kernel for the Harris RTX 2000

Harvey Glass
University of South Florida
Mike Mellen
Tom Hand
Harris Semiconductor

This paper describes the design of a multi-tasking kernel for the Harris RTX 2000 microcontroller. The RTX 2000 is a 16-bit processor that is particularly well-suited to support applications in demanding real-time embedded systems. The multi-tasking kernel acts as a resource manager and provides services for the applications programmer to coordinate the activities of a set of asynchronous tasks. A major emphasis in the design was to provide the applications programmer with a set of consistent facilities that integrate simply with the existing development environment and provide synchronization among concurrent tasks in a natural manner. Each task is assigned a priority and the real-time kernel dispatches the highest priority task from among those tasks ready to run. The system supports several mechanisms to facilitate synchronization and the sharing of resources. These include semaphores, regions (for mutual exclusion), pipes (FIFO buffers), and mailboxes. These features are described, along with design decisions that were made in their implementation.

Architecture Impact on Compiler Construction

Tom Hand
Harris Semiconductor
Melbourne, Florida 32902

This paper first examines the architectures of several RISC machines, including the Harris RTX 2000, and illustrates how they directly impact the construction of compilers for those machines.Second, this paper shows how the Harris RTX2000 C Compiler generates code for some constructs of the C language.

This C Cross Compiler, from Harris Semiconductor, is a full implementation of the proposed ANSI Standard for the C Programming Language. It is the first commercial C Cross Compiler that is available for the RTX 2000.

In addition, the integrated environment on which C runs contains both a C source statement symbolic debugger as well as a C source statement profiler.

Metrics for Real-time Systems

Tom Hand
Harris Semiconductor
Melbourne, Florida 32902

In the book *Elements of Software Science*, Maurice Halstead presented the first systematic analysis of computer programs. He illustrated that computer programs are governed by natural laws, both in their preparation and in their ultimate form. Halstead's work does not address real-time systems.

This paper identifies four aspects of real-time systems for which metrics are appropriate and then indicates a general method for defining customized metrics. Since the performance of a real-time system is directly tied to both its underlying hardware and software, these metrics will incorporate aspects of both the system hardware and the system software.

Load Time Forth Capabilities

Rick F. Hoselton
Technology Information Center
2900 Wilcrest Houston, TX. 77042

Conventional wisdom says that an interpreted language, even a with a very efficient address interpreter like Forth's, just can't run as fast as compiled code. For a given algorithm in a given hardware environment, Forth users give up performance as the price of convenience. However, Forth's unique load-time (or target compilation time) capabilities can sometimes be used to greatly increase run-time performance. Perhaps faster than other compiled languages, possibly even faster than conventional assembler code.

Asynchronous Designs for Systolic Arrays

Moon S. Jun
Department of Computer Science
University of Maryland, Baltimore County

In this paper, we present new methodologies for design of systolic arrays and asynchronous arrays that implement recursive algorithms efficiently. Using the new methods, we propose a systolic array with very simple local interconnections for matrix multiplication which achieves optimal performance without using any undesirable properties such as preloading input data or global broadcasting. An asynchronous array for matrix multiplication which can speed up the total computation time significantly is also presented.

Multitasking/Multiuser Systems

Forth in a Multiprogramming Operating System Environment

Yong M. Lee
Edward Conjura
Computer Science Department
Trenton State College

A multiprogramming operating system is implemented first. Then a Forth `DICTIONARY` is to be viewed as a general database access issue. An approach to access a word in the `DICTIONARY` is to make it content addressable using the content of the `NAME` field of the `WORD`.

A High Speed Foreground Multitasker for Forth

Howard Leverenz
Texas Microsystems Inc.
10618 Rockley Road, Houston, Texas 77099

Forth systems are frequently called upon in one application to perform time-critical hardware control functions as well as calculations and data manipulation. The standard Forth multitasker cannot guarantee finer temporal granularity than the maximum loop time of the round-robin multitasker. An analog to the Forth multitasker is presented which controls the execution of assembly language foreground tasks with a temporal resolution equal to the period of a foreground interrupt time base. With a time base of 1 kHz a foreground task on a small microprocessor can be reliably activated within 1 μS of the requisite activation time.

A New Process Controller: A Study in Modularity

L. Greg Lisle
Fairchild Industrial Products
1501 Fairchild Drive, Winston-Salem, NC, 27105

Fairchild Industrial Products has developed a new family of process controllers (DCM-2020) based on the Forth language. These process controllers were designed for a wide range of continuous processes found in industry. While focusing on the software design, this paper will also look at some of the hardware features of these devices.

I will discuss the development and future growth of this product line. A particular emphasis will be on advantages Forth has brought to this project. Software features to be highlighted will include interrupt handling, multi-loop context switching, and a simplified menu construction technique.

Development of a Forth Based Remote Weather Station

Brian C. Mikiten
B.C.M. Designs
Shawn Mikiten
B.S.C.S.

The need to provide up-to-date information on weather conditions exists in both the scientific and non-scientific communities. The researcher uses weather monitoring systems to provide forecasting information or data on conditions affecting a device or system under development. The lay person has concerns more general and recreational in nature, but nonetheless important.

Anonymous "Things" Used as Locals

Leonard Morgenstern
304 Rheem Blvd., Moraga, CA 94556. GEnie LMORGENSTERN

A scheme of headerless Forth words will be presented, embracing variables, constants, and colon definitions. The idea can be extended to almost any class of Forth words. By adding automatic nesting of scope, it can form the basis of a system of locals that make it possible to pass information back and forth among a group of consecutively defined Forth words, not only with respect to data (local variables and constants), but also program (local colon definitions).

Safety Nets for Error Trapping

Leonard Morgenstern
304 Rheem Blvd., Moraga, CA 94556. GEnie LMORGENSTERN

Error handling is the most difficult aspect of programming. A scheme of 'safety nets' is presented in this paper that permits jumping back to predetermined points, which can be set and reset at run time. A net is effectively a `GOTO`, or more accurately, a `GOBACKTO`. The net takes care of the stack and instruction pointers.

Real Time Signal Processing with the RTX 2000

Karl-Dietrich Neubert
Michael Ulbrich
Physikalisch-Technische Bundesanstalt
D-1000 Berlin 10, Federal Republic of Germany

Timing measurements are presented for the real time FFT, using the AT/FORCE Coprocessor board of Silicon Composers and an 386-AT compatible host. The data from a 100 kHz A/D converter are read interrupt driven by the RTX 2000 on the AT/FORCE board. We shall address the following aspects: optimal distribution of the workload between

coprocessor and host for performing the FFT and preparation of the data for graphic display. The block transfer between coprocessor and host is described. A cost analysis for heavy use of look-up tables versus actual computation, where appropriate, is given.

Forth Real-Time Automata for Experiment Control

John L. Orr
Southwest Research Institute
6220 Culebra Road, San Antonio, TX 78284

Finite automata are useful definitions of real-time experimental procedures. This paper describes components of a system to implement complex finite automata for experiment control. Forth colon definitions are used as the states or nodes and `IF` ... `ELSE` ... `THEN` words are used to trigger state transitions.

Mammography Information and Reporting System in Forth

Virgil O. Stamps
Houston, Texas 77068

Description of a Forth implementation of MIRS, a Mammography Information and Reporting System. This paper describes program structure, data base approach and features of this PC program.

Cooperative Multitasking on the RTX 2000

Rick VanNorman
Harris Semiconductor
Melbourne, Florida

This paper describes a simple cooperative multitasker for the Harris RTX 2000. The execution of the task switch is implemented in high-level Forth for portability and in optimized RTX 2000 instructions for speed. The usefulness and limitations of an extremely simple multitasking environment are explored. The speed of the context switch on the RTX 2000 is compared with the same context switch implemented on other processors.

Virtual Memory on Systems without Hardware Support

David Weinstein
9036 N. Lamar #274, Austin, TX, 78753

It is a rare Forth application which requires even 64k of code space. However, it is very easy to use more than 64k of data. And even if the decision is made to use a 32bit forth (with the resulting fattening of code), lack of usable memory on the system can still be a problem. In this paper, a virtual memory system is described which allows the data space to far exceed the actual system memory, by using a device as the virtual memory.

Directory Replication in Distributed Systems

K. C. Wong
Joseph Cornacchio
Department of Mathematical Sciences
The University of Akron, Akron, Ohio 44325. Tel: (216) 375-7193.

In this paper, a simple algorithm, which essentially determines the level of replication for those directories existing in a particular directory structure based on the determined level of replication for files and directories, is proposed. The algorithm is devised in such a way that the purposes concerning the reachability of all of the files and directories replicated in a distributed environment and the optimality (minimization) of storing the corresponding directory structure are achieved simultaneously. Moreover, the algorithm is found to perform in the order of n, where n is the total number of files and directories contained in a particular directory structure.

An approach adopted to estimate the amount of storage space saved in storing a particular directory structure indicates that grouping those files of higher level of replication near the root will save not only file access time, but also the storage space for storing directory structures. The maintenance of storage space for storing directory structures in an optimal way during file/directory creation, deletion, and modification is also discussed.