Abstracts of the 1988 EuroFORML Conference

September 22 – 24, 1988
Southampton University
Southampton, England[1]

Specification and Optimization

B. Stoddart
Teeside Polytechnic
School of Information Engineering

This paper provides a tutorial introduction to some formal aspects of computer science in the context of the Forth development environment. We hope to show that the techniques described can help Forth users in the following ways.

i. To specify exactly what their Forth words do and when it is safe to use them.

ii. To reason about the correctness of their code.

iii. To clearly separate the specification of systems components from their implementation.

A mathematical notation is introduced for writing the specifications (glossary entries) of Forth words, and we show how such specifications can be composed to obtain the specification of a new definition from the specification of its parts. The meaning of sequential composition is then considered, with reference to a technique for optimizing compiled Forth.

Source Stack list Utility 'slist'

C. J. Jakeman
50 Grimshaw Road
Peterborough PE14ET

This utility operates on blocks of Forth source code producing lines of output in a similar way to the **LIST** word but appends to each line of a definition a summary of the stack contents. It checks that the stack effect intended matches the stack effect of the definition and also checks the following, issuing warnings where appropriate:

- for attempts to pop an empty parameter stack,
- for an unbalanced use of **>R** or **R>**,
- for an incomplete or unbalanced **IF**, **BEGIN** or **DO** construct.

It is written to the Forth F83 standard.

Forth Implementation on the Acorn RISC Machine

B. W. D. Larkin
10 Granville Park
London SE13 7EA

This paper describes some aspects of Forth on the Acorn/VCSI Technology 32-bit Reduced Instruction Set Microprocessor. This processor has a number of characteristics that made it a good Forth machine including stack instructions, a single word subroutine call and a large register file.

Implementation details and timing results for language primitives are given. The methods discussed here form the basis of a commercially available Forth-83 system for this machine.

[1]Proceedings are available from MPE, 133 Hill Lane, Southampton S01 5AF, England. VISA or Mastercard accepted, £15. plus £5. shipping and handling.

Target Compiling — The Need For An Environment

C. L. Stephens
Comsol
Canada Road
Byfleet KT14 7HQ

After outlining the differences between a conventional Forth compiler, and a cross target compiler the paper will show the technical and commercial advantages as well as the problems inherent in these techniques. It will show how many of the problems of cross compilation can be overcome by careful design of the environment in which the cross compiled codes development takes place. It will introduce a number of powerful and novel techniques to optimize the development of code under these circumstances.

A Floating Point Package With Sizable Accuracy

C. Lavarenne
Authorized MPE Dealer/Consultant
12 rue du Docteur Vuillieme
92190 MEUDON, France

An original Forth83 software floating point package will be presented.

The CORDIC unified algorithm on which it is based will first be reviewed, showing its simplicity, its ability to cope easily with any given accuracy, and its power for complex functions.

The choices made for its implementation will then be discussed, mainly on the topics of efficiency, portability and compatibility with IEEE standards, Intel 80287 floating point coprocessor, and previous floating point packages.

Finally, some timings will be given and compared with other floating point packages. This floating point package is available under Modular Forth.

Who Needs the Text Interpreter Anyway?

R. Crawford
MPE
133 Hill Lane, SOUTHAMPTON S01 5AF, England

The behavior of Forth words is shown to be different in three time frames: Compilation, Execution and Interpretation. A new system is described which replaces the outer interpreter and compiles with a single, unified mechanism. The implication of this approach with respect to immediate execution, real-time debugging and user interface design are discussed.

In-line Code Generation for Optimized Data Structure Access

C. Hainsworth
Chairman of Forth Interest Group

A novel approach for manipulating variables within Forth words is described. The technique employed is based upon in-line code generation. It is shown how this approach can be extended to provide optimizing compilers for classes of words.

An Augmented Transition Network Compiler and State Machine

R. Crawford
P. Sammons
MPE
133 Hill Lane, SOUTHAMPTON S01 5AF, England

A notation for describing Augmented Transition Networks is introduced. It is shown that this notation can be easily compiled into a simple data structure. The trade-offs between size of structure versus speed of execution are discussed. An algorithm for traversing such networks is developed.

Forth In Space!

R. Vahrman
Brunel Institute of Bioengineering

The Brunel Institute of Bioengineering had several contracts with the European Space Agency, concerned with the development of experimental biological facilities to be flown on unmanned space missions. The systems being designed must work reliably, without intervention, for several months at a time and as a consequence require the development of fault-tolerant hardware and software.

A modular approach has been adopted in order to simplify the design and development of the equipment. Each subsystem is under the control of a dedicated microprocessor board communication via dual port memory with an IBM/PC host. The function of the host then largely becomes one of activating the subsystems in the correct sequence at appropriate times. The microprocessor boards, based on the 6303 single chip computer were designed at the Institute, and both the boards and the host are programmed in Forth.

Modules that have been developed using this method include environmental life-support systems, plant handling workstations, and an image recognition system based on an optic RAM camera. A system that allows investigators to model their experiments in the absence of the hardware is currently under development.

Application of the Harris RTX Processor in Medical Imaging

R. Marriott

SMIS

The Harris RTX processor is used in the SMIS Programmable Pulse Generator. The generator gives the user the ability to create accurate for use in the Medical Resonance Imaging (MRI). The pulse generator is equally applicable to other areas of engineering that require generation of precise time intervals ranging from 200nS to several seconds, hours or days.

The RTX RISC processor was chosen because of the capability to perform a Forth instruction in a single processor cycle. This is important for building accurate delays for pulse width control; delays are the fundamental element in generating pulse sequences for MRI.

The RTX board was designed to use the processor in an IBM PC or compatible computer. A language called PPL was defined to enable the user to rapidly develop pulse sequences using simple statements and control structures. The PC is used to provide an easy-to-use interface for writing and compiling these structures.

The board hosts the proposed ANSI standard Forth-83 operating system supplied by MPE. The power of the RTX processor combined with the facilities offered on the board and the Forth operating system gives the user the flexibility to use the board in areas such as signals and speech processing, control systems and data acquisition.

Acquisition and Real-Time Display of Spectrophotometer Data Using Forth

M. Thomas

Cairn Research Ltd.

Our company produces a spectrophotometer system that makes rapid sequential measurements at different wavelengths, by using a series of optical interference filters in a rotor that can spin at up to 300 revolutions per second. We have designed our own computer interface for the system, allowing bidirectional data transfer, and the software has been written for an IBM/AT (or compatible) computer using Laxen and Perry's Forth83. The rotor can contain up to eight filters, and the computer interface has four independent data channels for each filter position, giving up to 32 channels in total. The software includes a fast assembler, that allows on-line display and manipulation of up to 16 channels at the same time. The software is written for the EGA, and is designed for easy porting to the VGA, where it will be able to make use of the higher vertical resolution of this card. Individual data channels can be displayed in different colours for ready identification. The principles of operation will be discussed, and examples of source code will be shown The software also has extensive interfacing with MS/DOS for control of memory allocation and disk file access. It is fully menu driven, but direct access to Forth is available as a menu option. The system will be demonstrated during the conference.

"Go! Forth", with "Come hither"?

R. Goddard BSc

Applications Engineer

MPE

133 Hill Lane, SOUTHAMPTON S01 5AF, England

This paper looks at a range of user interfaces currently in use in commercial software, commenting on their benefit and ease to both novice and experienced users. It then looks at the ergonomic requirements for the interfaces outlined, giving some ideals. This is followed by some guidelines for implementing such user interface in Forth, and extends these to implementing the Forth environment for programmers and the development cycle.

A C-to-Forth Translator

A. Winfield

S. Kelley

Advanced Processor Design

The authors have rewritten the final 'Code Generation' phase of a full ANSI specification C language compiler to generate code for the APD MF1600 series processors. These processors are characterized as fast two-stack machines with an assembly language which closely resembles Forth; it follows therefore that the C Compiler is effectively an ANSI C into Forth 'translator'.

This paper focuses on the interesting problems of code generation which arose during this project and their solutions. Of particular interest are the storage allocation strategies for the various classes of C variable (register, auto and static). This paper illustrates the Forth code output by the translator with a number of examples of the actual code generated.

Some Application Packaging Considerations

C. L. Stephens
Managing Director
Comsol
Canada Road
Byfleet KT14 7HQ

Forth's capabilities make it a good language for developing systems using a prototyping methodology.

However the final stages of this technique can be very wasteful of time and manpower, as the product is fine tuned to meet the changing requirements of users and sales departments. Frequently the overhead of describing such changes to contractor greatly exceeds the time taken to make the change.

For this reason the ideal project team consists of a programming expert who knows (or learns on the job) the application coupled with an applications expert who, given the package and its design documentation, knows enough Forth to be able to perform these minor changes. The latter needs to be sufficiently expert to know when the changes required are more complex than his capabilities will allow. Clearly to achieve this ideal co-operation all the source code produced by the programmer must be available to the applications expert.

A natural extension of this philosophy is a package designed for OEMs which is a fully functioning solution to an application problem and which is supplied to them as source. In this way modifications may be done prior to compilation if that is the most efficient point rather than having to provide many complex run-time options and the package may be readily modified or extended to suit differing market requirements.

Examples will be given from such a package developed for the control and instrumentation industry.

A Text File Syntax For Screen File Users

S. Pelc
A. Waters
MPE
133 Hill Lane, SOUTHAMPTON S01 5AF, England

A syntax for using text files is described. The approach does not treat files in the conventional monolithic way. Related items may be grouped together using an elastic chunking technique — Pages. Pages may be accessed randomly and are of variable size.

The Use of State Machines in Forth Applications

R. Borrell
Software Control

The concept of a state machine is introduced and several state machine applications are described.

The Use of Multi-Tasking in Forth Applications

H. Oakford

Several Forth applications which use multi-tasking are described.