R77-84—Mavaddat, Farhad and Behrooz Parhami, "A Data Structure for Family Relations" (21 pp., Arya-Mehr University of Technology, Tehran, Iran)

A data structure is proposed which enables the determination of all family relations of common interest with the minimum amount of information on each individual. The problem of updating of information due to births, deaths, marriages, and divorces is considered. Algorithms for determining the immediate relatives of each individual are given, and a framework is established for writing procedures to determine other relatives.

R77-85—Dubois, Eric and A. N. Venetsanopoulos, "The Discrete Fourier Transform Over Finite Rings With Application to Fast Convolution" (26 pp., University of Toronto, Toronto, Canada)

Necessary and sufficient conditions for a direct sum of local rings to support a generalized discrete Fourier transform are derived. In particular, these conditions can be applied to any finite ring. The function $O(M)$ defined by Agarwal and Burrus for transforms over $Z_m$ is extended to any finite ring $R$ as $O(R)$, and it is shown that $R$ supports a length $m$ discrete Fourier transform if and only if $m|O(R)$. This result is applied to the homomorphic images of rings of algebraic integers. The motivation behind this work is to obtain algorithms for the computation of digital convolutions which have low bit-operation complexities, thus making them attractive for special-purpose hardware implementations. An example of the application of these concepts is given by a Fermat number transform over the ring of residues of the integers of the quadratic field $Q(\sqrt{-3})$.


A joint FAA/Industry Task Force has recommended that area navigation become the primary method of aerial navigation in the United States by 1982. This mode of navigation has already gained widespread acceptance in commercial aviation but, due primarily to high cost, has not yet become established in general aviation. This thesis investigates the possibility of utilizing an off-the-shelf microprocessor as a low-cost alternative to current special-design systems. The Microprocessor-based Area Navigation (MAN) system is designed around the MOS Technology 6502 microprocessor. The software makes extensive use of the CORDIC algorithm to evaluate the trigonometric functions associated with the area navigation computations. The MAN system is constructed, programmed, and tested using simulated VOR and DME inputs in a static ground environment. Computation times on the order of 0.2 seconds are achieved, and the system operates well within the tolerances specified by the FAA. The total system cost is low enough to offer a significant economic advantage over systems that are currently available.

R77-87—Riseman, Edward M. and Michael A. Arbib, "Computational Techniques in Visual Systems, Part II: Segmenting Static Scenes" (98 pp., University of Massachusetts, Amherst, Massachusetts)

In the first part of this paper, "The Overall Design," we looked at our evolving understanding of computational techniques—both in analyzing the visual system of animals and in building computer vision systems. We divided the computations between low-level systems and high-level systems, and sketched the interaction between the two system types. In Part II, we focus upon specific tasks of the low-level systems—feature extraction and segmentation, as well as their competition and cooperation. We continue to emphasize an integrated system design, with interaction of multiple processes resolving ambiguous and noisy data. A survey of processes which operate on a single static, but colored, image shows how segmentation can proceed via boundary
formalization, and by formalization of regions on
the basis of color and texture cues. We
present extensive experimental data on the
results of applying segmentation tech-
niques.

R77-88—Murugesan, S., “On Negabinary
Arithmetic Circuits Using Binary
Algorithms” (10 pp., ISRO Satellite Systems
Project, Bangalore, India)
A new type of negabinary arithmetic
algorithms/circuits for negative negabinary
addition, polarization, and negabinary
addition, subtraction, and comparison are
presented. The results are based on
an arithmetic algorithm. They use standard MSI
binary adders/ALU and look-ahead carry
modules and are very simple compared to
the circuits proposed earlier. Also, this
result is very well suited for fast
operation.

R77-89—Matelan, Mathew N., “Auto-
maton Design of Dedicated Real-Time
Control Systems” (222 pp., General
Dynamics, Fort Worth, Texas)
A methodology for reducing the com-
plexity of designing dedicated real-time
control systems is developed in this thesis. The
basis of the methodology is an ab-
stract model of the control system design
process. The CSD model reveals three
areas amenable to automation: 1) the
selection and configuration of hardware,
2) the production of software, and 3) the
adaptation of a monitor to maintain real-
time integrity of the entire system. The
concept of hardware binding is intro-
duced, and it is shown that delaying the
point in the design cycle where hardware is
functionally bound allows a new ap-
proach to machine independence. Concepts
which allow expression of repetitive control
situations are described, and a realization-
independent language (CSDL), based on
these concepts, is defined. Methods for
automatically selecting a time-wise cor-
correct monitor are classified, and techniques
for specifying the realization capabilities
of digital processors are discussed. These
concepts and techniques are brought
together to form a design automation system
for the production of a complete controller
design from a behavioral description. An
example description is traced through the
CSD system, producing a software listing
and hardware configuration document for
an actual microprocessor (the Intel 8080).
The CSD model provides a structured
description of the control system design
process, allowing a unified perspective in
the realization of controllers for applica-
tions previously considered to be unrelated.
The concepts developed in this thesis, both
in the model and in the system, define a
new direction in the production of real-
time control systems.

R77-90—Franklin, Ben S., “Microcom-
puter Keyboard/Display Interfacing Made
Easy” (15 pp., Syracuse University, Syra-
cuse, New York)
Scanned keyboards and displays are a
part of many low-cost microprocessor de-
signs. The interface to the keyboard and
display can represent one of the most
complex programming jobs a designer
faces if he has to program the keyboard
scan, debounce, and rollover functions as
well as the display refresh function. How-
ever, that programming headache is now
solved through the availability of inex-
ensive keyboard/display interface chips that do all of the keyboard
and display interfacing automatically.
Through the use of such devices, the pro-
gram to support the display executes
write operations to the interface chip
display RAM, and then the interface chip
takes over the display refresh function.
The keyboard program reads out a valid key
character code from the interface chip’s
FIFO RAM after the CPU has been noti-
fied of the end of a valid character
through an interrupt generated by the
interface chip. The net benefit to the
system designer of the keyboard/display
chip is a smaller program, greatly
improved system throughput, reduced CPU
RAM requirements, and a significantly
accelerated development schedule.

R77-91—Brodie, M.L.M., ed., “ZETA: A
Prototype Relational Data Base Manage-
ment System” (96 pp., University of
Toronto, Toronto, Canada)
This report describes the current status
of the design and implementation of
ZETA—a prototype relational data base
management system. ZETA is composed of
two principal levels. The lowest level,
MINIZ, provides primitive facilities for
representing and manipulating single
relations on a tuple oriented basis. The
intermediate level, the EXECUTOR, uses
the facilities of MINIZ to provide a higher,
relation-oriented view of the data base.
The language facilities are provided at the
highest level in order to permit access
to the data base through a query language
facility and a programmable interface.

R77-92—Hehner, Eric C.R., “Matching
Program and Data Representations to a
Computing Environment” (159 pp., Uni-
versity of Toronto, Toronto, Canada)
This thesis is about the advantages of
tailoring the representation of
machine instructions and data to the
distributions of usage found in a computing
environment. Variable-length encodings
can save memory and at the same
time eliminate overflow in all its forms. We
identify common dependencies and con-
tributors to redundancy in machine lan-
guage and develop techniques for mea-
suring and removing them. Using these
techniques on a sample of machine language
compiled from a high-level source, we are
able to trim 75% from the space taken by
a contemporary machine representation,
and 60% from the space taken by a
language-directed machine representation that
did not employ our techniques. The gain
in space will be accompanied by a reduc-
tion in execution time due to more effi-
cient use of available bandwidth. The cost
is an increase in hardware complexity. We
present a range of data organization models
that saves space and greatly reduces the
probability of overflow, and give example
evaluations of two models. We evaluate
one model with actual data to find the
best representation of variables within the
model.

R77-93—Hehner, Eric C.R., “Merlin:
Towards an Ideal Programming Language”
(91 pp., University of Toronto, Toronto,
Canada)
Accepting the premise that machines
should be designed to suit the languages
that will be implemented on them, and not
vice versa, this project explores issues of
language design not influenced by imple-
mentation considerations. Our concern is
only convenience for the development
and clear expression of algorithms. Under
this we include simplicity, a minimum of
terminology, freedom from senseless re-
strictions, lack of useless redundancy, in-
cclusion of useful redundancy, and affinity
of language structures to thought struc-
tures. An attempt is made to introduce a
discipline of language design, rather than
sticking together favorite language features.

R77-94—Hehner, Eric C.R., “DO Con-
considered OD: A Contribution to the
Programming Calculus” (62 pp., University
of Toronto, Toronto, Canada)
The utility of the repetitive DO con-
struct is challenged. Recursive refinement
is claimed to be semantically as simple
and superior for programming ease and
clarity. Some programming examples are
offered to support this claim. The rela-
tion between the semantics of predicate
transformers and the mathematical seman-
tics of Scott and Strachey is presented.

R77-95—Lasker, David M., “An Investi-
gation of a New Method of Constructing
Software” (62 pp., University of Toronto,
Toronto, Canada)
This thesis describes an interactive
text editor which stores, displays, and
modifies a program’s parse tree using a
CRT display terminal. The programmer
builds the parse tree of his program
top-down, by selecting valid productions
of the grammar with a light pen. This
method makes it impossible to construct a
program which is syntactically incorrect.
The usage of the editor is briefly de-
scribed, followed by a description of the
algorithms and data structures used to
implement it.

R77-96—Bernstein, Philip A. and Catriel
Beeri, “An Algorithmic Approach to
Normalization of Relational Data Base
Schemas” (72 pp., University of Toronto,
Toronto, Canada)
It has been proposed that the description
of a relational data base can be formulated
as a set of functional relationships among
data base attributes. These functional rela-
tionships can then be used to synthesize
algorithmically a relational schema. In
Parts I and II we present an effective and
efficient procedure to perform such a
synthesis. An abstract view of the
procedure is presented in Part I. The schema
that results from this procedure is proved
to be in Codd’s third normal form and to
contain the fewest possible number of
relations. Problems with earlier attempts
at constructing such a procedure are also
discussed. A basic step in the synthesis
algorithm is to check for membership of a
functional dependency in the closure of a given set of functional dependencies. In Part II we present a linear time membership algorithm, and we show how it can be used for an efficient implementation of the synthesis algorithm. In Part III we treat problems related to Boyce-Codd normal form and to key finding. In particular, the problems of whether a schema is in Boyce-Codd normal form and of whether additional keys exist in a relation are shown to be NP-complete.

R77-97—Czarnik, B. and D. Tsichritzis, eds., "A Student Project for an Operating Systems Course" (105 pp., University of Toronto, Toronto, Canada)

Perhaps the most valuable method of learning is through experience. This report presents a course project in operating systems involving the design and implementation of a "toy" operating system. The first part of the report gives the complete description of the project as well as an example. The second part of the report gives a manual for the implementation language, TOPPS—a language which provides basic synchronization primitives.

R77-98—Weissman, Larry, "Psychological Complexity of Computer Programs: An Initial Experiment" (87 pp., University of Toronto, Toronto, Canada)

Many ideas and techniques have been expounded to measure the complexity of programs. However, there is no quantitative evidence that the quality of the programs has indeed been improved. We believe that experimental studies should be performed to measure those factors which make programs complex. An initial experiment has been conducted to measure the effects of three such factors: comments, paragraphing, and mnemonic variable names. This report summarizes the results of this experiment.

R77-99—Fung, K.T., "A Parallel Clustering Method" (14 pp., University of Windsor, Windsor, Canada)

A method for the generation of a fixed number of nonoverlapping clusters for a given set of items with multiple numerical key characterizations is described. The method is based on an application of the principal component analysis of vectors, and the clustering procedure obtained is capable of being processed in parallel in a multiprocessing system. The procedure essentially attempts to produce the best results for a fixed number of computational stages used.

R77-100—Donahue, J.E., J.D. Gannon, J.V. Guttag, and J.J. Horning, "Three Approaches to Reliable Software: Language Design, Dyadic Specification, and Complementary Semantics" (131 pp., University of Toronto, Toronto, Canada)

This report consists of three papers, prepared for separate presentation at conferences, that have a common thread: they discuss techniques that can be expected to improve the reliability of software. The first surveys language design principles and presents empirical evidence that even small language changes can significantly affect reliability. The other two are more theoretical and discuss specification techniques for programs and programming languages, respectively.

R77-101—Barnard, David T., "Automatic Generation of Syntax-Repairing and Paragraphing Parsers" (134 pp., University of Toronto, Toronto, Canada)

This thesis presents the design of an efficient top-down no-backup parser. The parser does syntax repair of invalid programs; it modifies any syntactically invalid program so that it becomes a valid one and displays the modified version of the program. It automatically produces formatted source listings and is machine independent. This parser is currently in use for a practical teaching subset of PL/I, called SP/I. The parser is controlled by tables which describe the syntax, formatting, and error repair for the language. With appropriate tables, the parser can be used for the LLI(1) programming languages. The language to be processed is described by syntax charts, which are hand translated to a linear syntax language that is equivalent to the corresponding charts. The syntax language is processed by a table generator to produce tables for the parser. The table generator itself uses the parser and generates its own parsing tables.