easily be modified to include such a signal. When E = 1 the circuit is disabled. After feeding-in the inputs, E is pulled down to zero, enabling the device.

Acknowledgement

The author would like to thank Dr Robert Burger for his help in this research.

D. SALOMON
Computer Science Department, California State University, Northridge, CA 91330, USA.

References

1. M. Abd Alla and A. Meltzer, *Principles of Digital Computer Design*. Prentice-Hall, Englewood Cliffs, N.J. (1978).
2. J. Cavanagh, *Digital Computer Arithmetic*. McGraw-Hill, New York (1984).
3. J. Cavanagh, *Digital Computer Arithmetic*, p. 105.
4. B. Gilchrist, J. H. Pomerence and S. Y. Wong, *Fast Carry Logic for Digital Computers. IRE Trans. on Electronic Computers*, vol. EC-4 (Dec. 1955), pp. 133–136.
5. D. Lewin, *Theory and Design of Digital Computers*. Halstead Press, New York (1972).
6. H. C. Hendrickson, *Fast, High-Accuracy Binary Parallel Addition. IRE Trans. on Electronic Computers*, vol. EC-9 (Dec. 1960), pp. 465–469.

Correspondence

Leith and Legal Logic Programming

Dear Sir,

In his article 'Fundamental errors in legal logic programming' (vol. 29, no. 6, 1986), Philip Leith attacks our use of PROLOG to formalise the provisions of the British Nationality Act 1981. The basis of his attack is the claim that we identify the legal process with the rigid, formal application of rules embodied in legislation. Leith is quite mistaken. Nowhere in our original draft or in the later published paper[3] do we ourselves make such a claim. In the paper we state explicitly that our British Nationality Act program can only be used to determine what follows if the rules of the Act are applied literally. Since there is more to legal reasoning than the literal application of the letter of the law, we did not propose that our program could be used to decide questions of British citizenship autonomously. On the contrary, we explain why it could not be used for this purpose, except in quite unrealistic circumstances. Elsewhere we have consistently emphasised the need for embedding the use of logic within a flexible framework for assimilating knowledge, for revising beliefs and for comparing alternative systems of belief.[1, 2, 4] It is precisely because reasoning in law demands such great flexibility that we believe it is an ideal domain in which to test the application of these developing techniques. We do contend, however, that systems like our British Nationality Act program can be of substantial use, even though they address only one relatively trivial aspect of the whole legal process. Some of these uses are described in our paper. The possibilities are argued at greater length in more general accounts of our work.[4]

Leith's central contention seems to be that there is no such thing as a clear legal rule, or more accurately that there is no set of circumstances in which a legal rule could be applied routinely. In making such a claim, Leith is attacking not simply the use of logic programming techniques in law, but the very idea that computer programs of any kind could ever be used in the routine administration of law, whether these programs are written in PROLOG, in FORTRAN, in COBOL, or whatever. If Leith is right, then there are no clear legal rules. The mistake must be in thinking that there are, and not in choosing some particular programming language to express them. However, in the day-to-day practice of law, there are mundane and routine tasks that have to be performed. That is why there are computer programs, like payroll systems for example, which are used every day to perform these tasks. To suggest that the administration of law can be reduced to some routine application of fixed legal rules is a massive oversimplification. To suggest that legal rules are never applied routinely is to oversimplify to the opposite extreme.

Philip Leith's attack seems to be based on a mistaken impression of our work. We would argue that it is also based on a mistaken impression of the jurisprudential material which he cites. Less easy to overlook, however, is the general tone of the article. We take particular objection to several quite outrageous claims that are made in the article, none of which Leith has bothered to substantiate. Leith draws attention to 'racist implications' of the British Nationality Act and suggests that our work gives support to the racist cause. He goes so far as to hint that this might explain in some part the funding that our work on logic programming has attracted. If the Act is indeed racist, then a rigorous derivation of its logical consequences can only make its racist character more apparent. In fact, it was precisely such ethical considerations that contributed to our initial choice of the British Nationality Act project.

Philip Leith concludes

> 'I believe that I have cast substantial doubt on the claimed success of the Imperial team in their use of logic programming in law. The next question should be whether the team have made similar claims in other areas outside computer science which are open to the same challenge.'

Leith is, of course, entitled to uncover and expose errors in our work. We welcome such challenges when they are based on technical considerations. However, articles such as this one by Leith, with its personal overtones, contribute little to what would otherwise be an important and stimulating debate.

Yours faithfully

R. KOWALSKI and M. SERGOT
Imperial College of Science and Technology,
University of London,
180 Queen's Gate, London SW7 2BZ

References

1. R. A. Kowalski, *Logic for Problem Solving*. North Holland-Elsevier, New York (1979). (See especially Chapter 13.)
2. R. A. Kowalski, *Logic-based Open Systems*. Department of Computing, Imperial College (1985).
3. M. J. Sergot, F. Sadri, R. A. Kowalski, F. Kriwaczek, P. Hammond and H. T. Cory, The British Nationality Act as a logic program. *Comm. ACM* **29** (5) (1986).
4. M. J. Sergot, *Representing Legislation as Logic Programs*. Department of Computing, Imperial College (1985). To appear in *Machine Intelligence* 11, Oxford University Press.

Editor's note:

The above letter draws attention to a comment made in the paper by Leith that there are racist implications in the British Nationality Act. Authors and potential authors should remember that *The Computer Journal* is devoted solely to reports of new technical developments in computer science and computer applications. In general, items which are not directly relevant to the technical aspects of work undertaken should not be included in papers submitted to the *Journal*.

Ada's Fixed-point Types

Dear Sir,

I feel that I must make a comment on the fragment of Ada which appears in A. J. Cowling's paper on type checking in *The Computer Journal*, **29** (6) 541, where the fixed-point type 'currency' is declared. (A minor point is that 'delta' should be in the bold font, not in italics.) I believe that this fragment perpetuates a misunderstanding about Ada (which fortunately in no way invalidates the technical content of the published paper), which I should like to clear up.

Currency comes in indivisible units. If I divide £100 between 3 Scotsmen, each gets £33.33 and I keep 1 penny. If I divide 1 000 000 lire between 3 Italians each gets 333 300 lire and

I keep 100 lire. In either case, I need to map the indivisible unit on to unity, and use integer division with a remainder, to get the right answer. The 'Steelman' specification for Ada's fixed-point arithmetic could be read as expecting such discrete 'scaled integers'.

In the event the Ada designers chose not to provide either the exactness or the operations such as remainder needed for this: they recognised that Ada is intended for embedded applications not financial ones. Ada's fixed-point types are classified as 'real types' and are intended to be used for the representation of variables which ideally would be continuous, not discrete.

As well as having obvious use on microcomputers which lack floating-point hardware, fixed-point types are important in embedded applications for communicating with the outside world, because the transducers that are used to read sensors and control actuators rarely accept floating-point data representation. These are the areas that Ada's fixed-point types address.

So Ada's fixed-point types are best implemented as 'scaled fractions', with any 'spare bits' used to provide more accuracy, not more range. There is certainly no guarantee that the 'delta' or 'small' of a type is indivisible: only that the 'small' is represented exactly.

Unfortunately, Ada's fixed-point declarations betray their evolution from Steelman, by still concentrating on the 'delta' and 'small' rather than mapping the 'range' on to pure fractions and specifying the minimum 'mantissa'.

A. J. Cowling is not the only one to be misled by this evolution: most casual readers of the Ada reference manual are, and I have even seen misleading 'dollars and cents' examples in Ada textbooks. It would be nice to see this and other problems with Ada's fixed-point types sorted out, some day.

Yours faithfully
T. J. FROGGATT
Senior Consultant
Ada Division
Software Technology
Systems Designers plc

Announcements

1–3 July 1987

International Workshop on Designing for Yield, University of Oxford

Aim

The aim of this workshop is to bring circuit designers, VLSI architects and IC process engineers together with test and yield modelling experts in order to discuss mutual objectives of designing ICs to yield targets. The worshop will be held in an informal environment with panel, regular-paper and short-paper sessions along with time for impromptu discussions and seminars. The number of delegates will be restricted to allow maximum interaction.

Background

The workshop is organised by the University of Oxford Department of External Studies in conjunction with the Department of Engineering Science. It is sponsored by the University of Oxford in association with the IEEE Computer Society.

The workshop is the third in a series on topics in VLSI started at Southampton in 1986. Edited proceedings of the former workshops have been published by Adam Hilger, Bristol BS1 6NX.

Wafer Scale Integration by C. R. Jesshope and W. R. Moore (1986).

Systolic Arrays by W. R. Moore, A. P. H. McCabe & R. B. Urquhart (1987).

Scope

The scope of the workshop includes, but need not be limited to, the following topics.

Advanced processing techniques, quality control, process monitoring.
Defect mechanisms and fault analysis.
Design rule optimisation.
Statistical defect models and yield estimation.
Fault models and test coverage.
Design for testability. BIST and test equipment.
Fault-tolerance, WSI and robust architectures.
Semi-custom and full custom circuits.
Repair techniques, discretionary wiring and high-density hybrids.
Packaging yield.
Reliability.
Cost models, case studies and further problems.

Local details

The workshop will be held at the University of Oxford from 1 to 3 July 1987 with meals and accommodation available in Balliol College on the nights of 30 June to 2 July. The cost of the workshop, proceedings, meals and accommodation will be around £295 (to be confirmed).

The University and its Colleges are located in the centre of this historic city, which has fast connections to London and its airports. Oxford and the surrounding area has numerous cultural and tourist attractions and has plenty to interest accompanying partners. Special visits can be arranged if there is sufficient demand.

For further information please contact:
Secretary, Designing for Yield Workshop, Department for External Studies, 1 Wellington Square, Oxford OX1 2JA (Tel: 44-865-270360).

11–13 July 1987

International Conference on Data Bases in the Humanities and Social Sciences, Montgomery, Alabama, USA. ICDBHSS '87 will be hosted by Auburn University at Montgomery (AUM) Library and Schools of Liberal Arts and Sciences, with events scheduled at other sites as well. Co-sponsors include: Air University, Maxwell Air Force Base; Alabama Library Association; American Bibliographical Center/Clio Press; AUM Lecture Series; Friends of the AUM Library; Huntingdon College; National Archives and Records Administration; Veterans Administration Hospital, Tuskegee; Tuskegee University School of Veterinary Medicine.

Accommodation includes a wide selection of nearby motels or inexpensive dormitory rooms at Huntingdon College. Depending on personal choices, conference costs will be modest: i.e. *c* \$180 (room and board, local transportation, registration, banquet, receptions and social hours, theatre) plus transportation to Montgomery. Delta Airlines is offering a 30% discount on all domestic coach services to ICDBHSS '87.

Plans for a three-day conference, from a welcome Friday evening until 2 p.m. Monday, include these special events:

- Reception, cash bar, and banquet with keynote speaker at the Officers Club, Maxwell Air Force Base.
- *Taming of the Shrew*, performance by the Alabama Shakespeare Festival Theatre, followed by a wine and cheese party at the state theatre.
- Plenary sessions and keynote addresses by:

  Dr Toni Carbo Bearman, Dean of the School of Library and Information Science, University of Pittsburgh; and former executive Director, National Commission on Libraries and Information Services (NCLIS), on libraries and data bases in the Humanities and Social Sciences.

  Dr Frank Burke, Executive Director of the National Historical Publications and Records Commission (NHPRC) and Acting Archivist of the United States, on archives and data bases in the Humanities and Social Sciences.

  Professor Ching-chih Chin, Associate Dean, Graduate School of Library and Information Science, Simmons College; and project director of the Emperor I Videodisc project, on image data bases in the Humanities and Social Sciences.

  Professor Neil Freeman, Department of Theatre, York University, on data bases