·O·U·G·

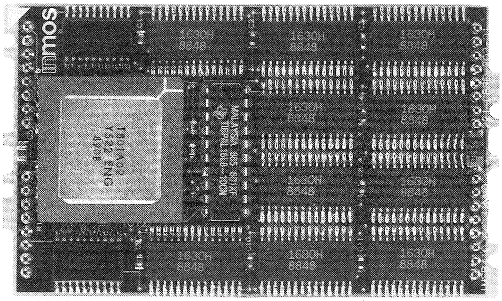
occam® user group · newsletter

for all users of occam and the transputer

No 11 July 1989

Contents

continued on back cover



INMOS B410 TRAM module (see page 53)

EDITORIAL

This issue of the newsletter forges a new link between the group founded in Britain, and the North American group whose members we welcome to the readership. There has been a flurry of activity about the world this summer, indeed I expect that some of you are reading this in mid-Pacific, on your Grand Tour of a succession of user group meetings in the Netherlands, Utah, Japan, New Zealand, and Australia. Plans are afoot for the staging of international joint meetings, but more of that anon.

Much of this activity reflects a growth of interest in applications: the theme of the Enschede meeting (see page 17) was the application of transputer based parallel machines; the Edinburgh meeting (see page 8) will focus on design, development and industrial applications; Peter Welch reports that he saw 'a lot of good applications aimed at mass business/consumer markets' on his visit to the Japanese group's recent meeting (see page 24). There seems clear evidence that the occam/transputer model is paying off in special purpose computers and embedded applications.

This year has also seen the ownership of INMOS pass into the hands of SGS Thomson. Announcements from the company (see page 52) are always very positive about this development – but then you would expect that. It may be more significant that discussion in the bar at Enschede was also positive about the future.

Once again, time to record appreciation of the work done by a retiring officer. The committee, and I am sure the whole of his readership, thanks Derek Paddon for his work editing recent issues of the newsletter.

It seems I am getting my just deserts for saying that I thought the newsletter was important. Henceforth – perhaps until this wears me out too – it will be me that encourages you to contribute to your newsletter. May as well start as we mean to go on then. *Geraint Jones, 30 May 1989*

CONTRIBUTIONS TO THE NEWSLETTER

You have been invited, you are now being urged, and you will eventually be pestered to contribute announcements, articles, letters about anything that looks as though it belongs in your newsletter. In particular we welcome letters, short articles or news about work being done with occam or transputers; calls for, discussion of, and reports on meetings of the group or related societies; ideas for new ways the group could help its members, or better ways of organizing what we do; details of material published elsewhere in books and journals; information about new products and courses.

Life would be easiest for the editor if you were able to submit material of longer contributions by electronic mail to `oug-news@uk.ac.oxford.prg`, or to send either unformatted ASCII files on an IBM PC compatible floppy disk or clean camera-ready copy to the editor at the address below. Camera-ready copy should be arranged not to look out of place when its linear dimensions are reduced to about 70%, i.e. from A4 to A5.

Copy for the next edition must arrive by *24th November 1989.*

A short, fat competition

The newsletter is also open to the offer of appropriate short, fat, black-and-white pictures for the front cover of the next newsletter. Of course, if things go on as they have been doing the list of contents may soon take up both covers, but I might bring the contents list inside for a really good picture.

Geraint Jones
Programming Research Group
11 Keble Road
Oxford OX1 3QD
United Kingdom

Tel: +44 865 273851
Fax: +44 865 273839
oug-news@uk.ac.oxford.prg

OCCAM USER GROUP PUBLICATIONS

Proceedings of occam user group technical meetings 7, 8, 9 and 10 – dated September 1987 to April 1989 – have been published and are available direct from the publisher, International Organisations Services BV at

IOS
Van Diemenstraat 94
1013 CN, Amsterdam
The Netherlands
Tel: +31 20 38 21 89
Fax: +31 20 22 60 55

or in the USA and Canada:
IOS
PO Box 2848
Springfield VA, 22152-2848
USA
Tel: +1 703 323 9116

OUG Newsletters 1–7 are now out of print; any requests for copies of newsletters 8–10 should be sent to

Claire Hale
INMOS Limited
1000 Aztec West
Almondsbury
Bristol BS12 4SQ

A directory of members is in preparation and a copy will be sent to all members when it becomes available. A bibliography is also being put together by Information Services staff at INMOS; any queries on this should be addressed to Zena Woodley at the INMOS office.

Michael Poole

ELECTRONIC GRAPEVINES

If you are an electronic mail user, you may want to know about two electronic mailing lists, carrying discussions on occam and the transputer. These offer you a mechanism rapidly to distribute information, short papers, programs, problems, even gossip about INMOS, to the sort of people who may be interested. You may even want to read this sort of thing. We even have subscribers from INMOS who can sometimes be goaded into authoritative declarations.

Each list has distribution points both in the UK and the USA. To join try making contact with the appropriate address: for the occam mailing list contact

`occam-request@uk.ac.oxford.prg` (in the UK)
or `occam-request@sutcase.case.syr.edu` (in the USA);
for the transputer list contact
`transputer-request@tcgould.tn.cornell.edu` (in the USA)
or `transputer-request@uk.ac.oxford.prg` (in the UK).

Please choose the contact the address that is nearest you, to reduce duplicated traffic across the Atlantic. The transputer list traffic is also available in newsgroup `comp.sys.transputer` on USENET.

A WORD ABOUT NAMES AND NUMBERS

I have tried to be reasonably consistent about addresses and telephone numbers in the newsletter. Human fallibility excepted, the telephone numbers are all given in the international form: so for example a UK caller should replace the +44 of my number by an initial nought, and in the USA you would just drop the +1 from Lyle Bingham's number.

Would that electronic mail was as simple. Again I have tried to be reasonably consistent: UK addresses are quoted big-end first, but in other parts of the world `geraint@uk.ac.oxford.prg` for example, would be given little-end first as `geraint@prg.oxford.ac.uk` and in the UK they prefer American addresses like `csa@adam.byu.edu` the other way, in this case as `csa@usa.edu.byu.adam`.

I have been told that if you are at a BITNET site, turning a big-endian address around does not work for all UK addresses, and in particular that it does not work for addresses at `uk.co.inmos`. It ought to be the case that all UK commercial domain addresses are known at Canterbury – `uk.ac.ukc` – so you may be able to render, for example `oug@uk.co.inmos`, as `oug%uk.co.inmos@ukc.ac.uk`. *GJ*

FORTHCOMING

AUSTRALIAN TRANSPUTER AND OCCAM USER GROUP CONFERENCE AND EXHIBITION

6–7 July 1989, Melbourne, Australia

In conjunction with the Centre for Advanced Technology in Telecommunications within the Department of Communication and Electrical Engineering at the Royal Melbourne Institute of Technology, the second Conference and Exhibition of the Australian transputer and occam user group will be held at the Glasshouse Theatre at the Royal Melbourne Institute of Technology. The Conference and Exhibition will provide an opportunity to focus on the latest research and developments in transputers, transputer-based technology and applications using transputers in the Australian environment.

Keynote Speaker

A special feature of the Conference will be the keynote speaker, Mr Peter Cavill, from INMOS (Bristol, UK). Mr Cavill is the Technical Director of INMOS, and has been with INMOS for ten years, joining from Fairchild. He has been responsible for all Transputer, DSP, SRAM and graphics developments at INMOS. He has managed the Transputer project from the start, being the board member responsible for new product definition and development.

Mr Tony Dent, General Manager, Asia Pacific from INMOS Japan, will also be in attendance to assist in answering questions.

Exhibition

In conjunction with the Conference, an exhibition will demonstrate the latest equipment being developed with the transputer family. The following exhibitors have been invited:

- ▹ Hawk Electronics Pty Ltd (Australian INMOS distributor)
- ▹ Unitronix Pty Ltd (PARSYTEC range of boards)
- ▹ Graphics Computer Systems Pty Ltd (SUN workstation add-in boards)
- ▹ Reptechnics Pty Ltd (Definicon Boards)
- ▹ Lionel Singer Corp. (A new transputer-based machine)
- ▹ Adaptive Pty Ltd (Microway boards)
- ▹ Atari Computers Pty Ltd (Atari Transputer Workstation)
- ▹ Prentice-Hall Book Publishers
- ▹ Quintek Ltd
- ▹ Parallel Publishing Ltd (publishers of 'Parallelogram')
- ▹ EPL Training

Pre-Conference Workshop

A feature of this year's Conference and Exhibition will be the provision of a one day pre-Conference Workshop on Wednesday 5 July 1989 to provide an overview of transputer hardware and software. This Workshop is particularly directed at an audience which has no prior knowledge of the transputer, transputer-based technology and parallel processing with the transputer. The Workshop will include the following topics:

- ▹ What is a transputer, and what can it do for me?
- ▹ Parallel processing and the transputer
- ▹ The occam programming language for transputers
- ▹ Other high level language support for transputer-based systems

The cost of attendance at this one day workshop will be $A95 (including workshop notes, morning, afternoon teas and lunch) is not included in the cost of the Conference itself, but will be an additional charge to participants. Those wishing to attend this pre-Conference Workshop should book early, as numbers will be limited to thirty-five, and so that they can be informed as to the actual location of the Workshop on the RMIT campus.

Booking

Make cheques or orders payable to *Centre for Advanced Technology in Telecommunications* for:

pre-conference workshop attendance fee	$A 95·00
conference attendance fee (if received before 22 June 1989)	$A245·00
late fee	$A 20·00
guests for dinner (per guest)	$A 35·00

Further Information

Information about the Conference and Exhibition can be obtained from

John Hulskamp
Department of Communication
and Electrical Engineering
Royal Melbourne Institute of Technology
GPO Box 2476V
Melbourne 3001 Australia

Tel: +61 3 660 2453
+61 3 660 2090
Fax: +61 3 662 1060
rcojh@au.oz.rmit.gecko

ARTIFICIAL INTELLIGENCE AND COMMUNICATING PROCESS ARCHITECTURE

an international conference and a technical meeting of the oug AI SIG
17–18 July 1989, at Imperial College, London

Parallel Processing and Artificial Intelligence are two key themes which have risen to the fore of technology in the past decade. This international conference brings together the two communities.

Communicating Process Architecture is one of the most successful models for exploiting the potential power of parallel processing machines. Artificial Intelligence is perhaps the most challenging application for such machines. This conference explores the interaction between these two technologies.

The carefully selected programme of invited talks and submitted papers brings together the very best researchers currently working in the field.

Topics include: Robotics, Neural Networks, Image Understanding, Speech Recognition, Implementation of Logic Programming Languages, Information management, The Japanese Fifth Generation Project, Transputers and occam.

Speakers

Keynote speaker

Prof. Iann Barron, *Transputer Technology and AI.* Iann Barron is co-founder of INMOS, and is Corporate Development Director currently leading the design of next generation transputers.

Invited speakers

- ▷ Prof. Igor Aleksander – *Myths and Realities about Neural Computing Architectures.*
- ▷ Prof. Colin Besant – *Application of Artificial Intelligence to the Programming of Robots.*
- ▷ John S. Bridle – *Automatic Speech Recognition and Parallel Processing.*
- ▷ Prof. David Gelernter – *Information Management in Linda.*
- ▷ Dr Atsuhiro Goto – *Research and Development of the Parallel Inference Machine in the FGCS Project.*
- ▷ Prof. Tosiyasu Kunii – *A Communicating Process Architecture Model of Primitive Image Understanding – A Case Study.*
- ▷ Dr Rajiv Trehan – *Concurrent Logic Languages for the design and implementation of Parallel AI Systems.*
- ▷ Prof. J. A. Robinson – *Functional and Relational reasoning with a fine-grain parallel reduction system.*
- ▷ Prof. Les Valiant – *Bulk-synchronous Parallel Computing.*

Conference organising committee

Dr Mike Reeve, Imperial College, London, UK; Steven Ericsson Zenith, INMOS Limited, Bristol, UK; J. T. Amenyo, Centre Telecoms Research, Columbia University; Jean-Jacques Codani, INRIA, France; Dr Atsuhiro Goto, Institute for New Generation Computer Technology (ICOT), Japan; Dr med. Ulrich Jobst, Ostertal–Klinik für Neurologie und Klinische Neurophysiologie; Dr Peter Kacsuk, Multilogic Computing, Budapest, Hungary; Pasi Koikkalainen, Lappeenranta University of Technology, Finland; Prof. T. L. Kunii, University of Tokyo, Japan; Dr Heather Liddell, Queen Mary College, London; Prof. Y. Paker, Polytechnic of Central London; Prof. L. F. Pau, Technical University of Denmark; Prof. Bernd Radig, Institut Für Informatik, München; Prof. Alan Robinson, Syracuse University, USA; Kai Ming Shea, University of Hong Kong; Prof. David Warren, Bristol University, UK; Chung Zhang, Brighton Polytechnic, UK.

Proceedings

The edited proceedings includes invited and submitted papers and is intended for publication in a new book series on Communicating Process Architecture published by John Wiley and Sons.

Conference dinner

The conference dinner will be held at London Zoo, with before dinner sherry in the Aquarium. Coaches will transport delegates.

Accommodation

Accommodation is available on the Campus of Imperial College. Campus accommodation is available for Sunday and/or Monday night. Hotel accommodation can be arranged separately by writing to the conference secretary.

Payment

Cheques or bankers drafts in pounds sterling should be made payable to: OUG AI Conferences. Registration should have been received by June 16th. Late registration incurs a £20 surcharge.

Non-residential	£200
Residential (1 night)	£225
Residential (2 nights)	£250
Conference dinner	£42

All enquiries should be addressed to the conference secretary:

The Conference Secretary
OUG AI Conferences
INMOS Limited
1000 Aztec West
Almondsbury
Bristol BS12 4SQ

Tel: +44 454 616616 x503
zenith@uk.co.inmos

This conference is underwritten by INMOS Limited,
to whom the organising committee wish to extend their thanks.

ELEVENTH OCCAM USER GROUP TECHNICAL MEETING

Monday and Tuesday 25–26 September 1989
University of Edinburgh, Scotland

The occam user group invites all those interested in concurrent systems using transputers to attend its 11th Technical Meeting at the University of Edinburgh. The meeting includes lectures, an exhibition, a panel session with key speakers, and meetings of the Special Interest Groups. The focuses for the eleventh meeting are:

- ▷ Productive design and development of concurrent software
- ▷ Industrial applications

Timetable

The meeting will take place in the Appleton Tower, George Square, in the University area of central Edinburgh. It will begin on Monday 25th September at 09:00, and the final session – Special Interest Group meetings – will end at 18:30 on Tuesday 26th September.

A conference dinner will take place on Monday 25th September.

Accommodation will be provided in the University's Pollock Halls for the nights of Sunday 24th, Monday 25th and Tuesday 26th September, with breakfast and (optionally) lunch and evening meal.

Booking

The conference fee of £50 (inclusive of VAT) will include: admission to all sessions; the conference dinner; coffee and tea during breaks; and one copy of the proceedings.

All accommodation and meals are separately charged so that delegates can choose the number of nights' accommodation to suit their individual travel arrangements, and those who prefer can use any of Edinburgh's wide range of hotels and restaurants. For those who take the full three nights accommodation with all meals, the total cost (including the conference fee) will be around £125. Precise details will be available shortly, at the same time as the full programme of the meeting.

Exhibition space will be available at a price of £360 per unit of 3 × 2 metres, which includes the conference fee (but not accommodation) for one person.

All enquiries should be made to:

John Wexler
Edinburgh University Computing Service
The King's Buildings
Edinburgh EH9 3JZ
United Kingdom

Tel: +44 31 667 1081 x2635
J.Wexler@uk.ac.edinburgh

EDINBURGH CONCURRENT SUPERCOMPUTER SECOND ANNUAL SEMINAR

Wednesday 27 September 1989, Edinburgh, Scotland

The Edinburgh Concurrent Supercomputer Project is based on a very large MEiKO Computing Surface, running a multi-user service as a national facility for academic and research applications, and also supporting an industrial affiliation scheme with currently some eighteen members. There are active user projects in many different disciplines. The project's own staff, as well as managing the facility, provide courses, documentation and user support, and are engaged in software development and acquisition and in-house research.

The Project's second annual seminar will take place at Edinburgh University on Wednesday 27th September 1989, and will include presentations of results achieved in many of the research areas using the Supercomputer, as well as reports on the status of the project as a whole. There will also be demonstrations of the MEiKO system and the AMT DAP, and software developed on them.

The seminar follows on immediately from the occam user group meeting, allowing delegates to combine them in a single visit. It will be held in the James Clerk Maxwell Building – where the Supercomputer is housed – in the King's Buildings, which is the University's 'science campus' in South Edinburgh.

Booking

The seminar fee is unchanged from 1988 at £55·20 (inclusive of VAT) with a reduced rate of £34·50 for early booking.

John Wexler
Edinburgh University Computing Service
The King's Buildings
Edinburgh EH9 3JZ
United Kingdom

Tel: +44 31 667 1081 x2635
J.Wexler@uk.ac.edinburgh

NORTH AMERICAN TRANSPUTER USERS GROUP FALL MEETING

18–19 October, 1989, Duke University, Durham North Carolina

The 1989 Fall meeting of the North American Transputer Users Group will be hosted by Duke University, Durham, North Carolina, on 18th and 19th October. Contributions are being solicited in the areas of hardware, software and applications. Presentations will be allotted twenty minutes each, with ample additional time for questions; this format will allow for approximately twenty-five talks. All accepted papers will appear in a published proceedings. Abstracts must be submitted by 15 July 1989. Notice of acceptance will be sent out by 15 August, and final drafts of accepted papers will be due by 15 September. Abstracts should be submitted – electronic mail submissions are preferred – to:

Professor John Board, NATUG 2 Program Chair +1 919 684 3123
Electrical Engineering Department jab@dukee.egr.duke.edu
Duke University
Durham, NC 27706
USA

TRANSPUTERS FOR INDUSTRIAL APPLICATIONS II

Second International Transputer Conference
23–24 October 1989, Antwerp, Belgium

Following the success of the 1988 event, the Belgian Institute for Automatic Control (BIRA) will organize its second international 'Transputers for Industrial Applications' conference on 23rd and 24th October 1989.

During this two day international transputer seminar, several speakers from all over the world (UK, USA, France, Belgium, etc.) will present a number of new and revolutionary applications of transputers in signal processing, real-time control, robotics, image processing, simulation, pattern recognition, telecommunications, computer aided design, artificial intelligence, supercomputers, etc.

The main goal of this conference is to show a number of advanced transputer-based products and applications, clearly demonstrating why transputers were chosen and how this specific application is implemented. Purely commercial presentations will *not* be accepted, neither will purely academic ones.

The BIRA aims at stimulating and propagating the science and technology of automation in the broadest possible sense. Their constructive rôle in this field has granted BIRA a reputation which extends far beyond the national boundaries. This specific seminar will attract approximately 200 (mainly industrial) people from all over Europe.

Exhibition

There will also be a specialized exhibition of transputer technology (last time we had fourteen companies, making it one of the largest exhibitions in Europe) that can be visited by all participants. Interested companies can already book space in the exhibition room at a reasonable price.

Patrick Van Renterghem
seminar coordinator
Automatic Control Laboratory/
The Transputer Laboratory
State University of Ghent
Grotesteenweg Noord 2
B-9710 Ghent-Zwijnaarde
Belgium

Tel: +32 91 22 57 55
Fax: +32 91 22 85 91
Email: PVR@bgerug51.bitnet

WORKING CONFERENCE ON DECENTRALIZED SYSTEMS

under the auspices of IFIP WG 10.3 Concurrent Systems
Lyon (France), 11–13 December, 1989

This conference will address recent issues in the implementation and evaluation of decentralized computer systems. It will be mainly focused on architectures and programming systems with no central point of control or communication.

List of topics (not limitative)

- ▷ Communication networks
- ▷ Formal methods for design and verification
- ▷ Non shared memory processors
- ▷ Fault tolerance
- ▷ Distributed operating systems
- ▷ Performance evaluation of parallel systems
- ▷ Distribution and communication concepts
- ▷ Hardware and software tradeoffs
- ▷ Applications
- ▷ Dedicated programming languages
- ▷ Fully scalable systems
- ▷ Distributed data bases and expert systems

Date and Place

The conference will be held in École Normale Supérieure de Lyon, France, on 11–13 December, 1989.

Deadlines

Prospective authors should submit five copies of their contribution in its final form (full paper), to the Chairman of the Scientific Programme Committee to arrive before

20 June 1989 at the following address:

Professor Claude Girault
Laboratoire MASI
Université Paris 6, Tour 65
4 Place JUSSIEU
F-75252 Paris Cedex 05
France

Tel: +33 1 43 36 25 25 x43 63
Fax: +33 1 43 29 41 96
Telex: UPMCSIX 200 145 F
Email: ...!mcvax!inria!litp!cg

Paper submission

Papers should be no longer than sixteen pages. The first page of the paper should indicate the author's name, affiliation, address, telephone, fax or telex numbers and electronic mail address. It should also contain a short abstract, some key words and the main topic (from the above list). A submission letter that contains a commitment to present the paper at the conference if accepted should accompany the paper. Authors will be notified of acceptance or rejection of their paper before the end of September 1989.

Publication

Copies of the papers will be available at the conference. Camera ready copies are due at the conference.

The proceedings will be published by North Holland, after the conference, in the series of the WG 10.3 Conferences: Highly parallel computers, Nice, France, 1986; Distributed processing, Amsterdam, The Netherlands, 1987; Parallel processing, Pisa, Italy, 1988.

Provisional Scientific Committee

C. Girault (Chairman) Paris, France; J. L. Baer, Seattle, USA; F. Baiardi, Pisa, Italy; M. Barton, Bristol, UK; G. Carlstedt, Göteborg, Sweden; G. Chiola, Torino, Italy; Y. Chu, Maryland, USA; V. Cordonnier, Lille, France; M. Cosnard, Lyon, France; E. L. Dagless, Bristol, UK; E. Dirkx, Brussels, Belgium; R. D. Dowsing, Norwich, UK; K. Ebcioglu, New York, USA; E. Feustel, Massachussets, USA; W. Giolo, Berlin, FRG; C. Hamacher, Toronto, Canada; M. Legendi, Budapest, Hungary; E. S. Lee, Toronto, Canada; R. W. Marczinski, Warsaw, Poland; K. Melhorn, Saarbrücken, FRG; H. Perros, Raleigh, USA; R. Puigjanner, Palma, Spain; M. Raynal, Rennes, France; G. L. Reijns, Delft, Netherlands; B. Samani, New Jersey, USA; B. Sanders, Zürich, Switzerland; N. Santoro, Ottawa, Canada; O. Spanniol, Aachen, FRG; J. Tiberghien, Bruxelles, Belgique; M. Vanneschi, Pisa, Italy; C. Whitby-Strevens, Bristol, UK.

DISTRIBUTED SIMULATION

part of the 1990 Western Multiconference
17–19 January 1990, San Diego, California

All papers related to the execution of continuous or discrete simulation programs on multiple processor computing systems are invited. Papers may deal with simulation on systems ranging from geographically distributed computing systems to tightly coupled multiprocessors and SIMD machines. Specific topics of interest include, but are not limited to:

- Methods for distributed concurrent simulation, discrete or continuous;
- Concurrent simulation methods for particular classes of systems (e.g. neural networks, electronic circuits, weather, computer systems and communication networks, military applications);
- Concurrent graphics and animation for simulation;
- Concurrent real time simulation;
- Machine architectures for concurrent simulation;
- Programming constructs and languages for concurrent simulation;
- Performance evaluation methods for concurrent simulation;
- Empirical performance evaluation studies.

Submission details

Papers must be in English and will be limited to six pages in the conference proceedings. Papers must contain original contributions to the field that have not been previously reported in the literature. Papers that contain incremental improvements of previously published work should clearly indicate new aspects of the work in the abstract. Extended abstracts will be accepted if it is not possible to submit a full paper by the 31st July deadline, however, full papers will receive priority.

Each submission should include the name, complete address and phone number of each author. Send six copies of the full text of papers to the program chairman at the address below. Papers are due 31 July 1989. Authors will be notified of acceptance 30 September 1989. Final papers are due 31 October 1989.

General Chairman

Dr Richard Fujimoto
Department of Computer Science
University of Utah
Salt Lake City, Utah 84112
Tel: +1 801 581 4845
fujiomoto@cs.utah.edu

Program Chairman

Dr David M. Nicol
Department of Computer Science
College of William and Mary
Williamsburg, VA 23185
Tel: +1 804 253-4748
nicol@cs.wm.edu

Program Committee

Jon Agre, Rockwell; Rassul Ayani, Swedish Royal Institute of Technology; William Bain, Intel; Mark Davoren, University of Edinburgh; Geoffrey Fox, California Institute of Technology; David Jefferson, UCLA; Walter Karplus, UCLA; Ed Lazowska, University of Washington; Greg Lomow, Jade Simulations; Daniel Reed, University of Illinois, Urbana; Paul Reynolds, University of Virginia; Lisa Sokol, MITRE; Carl Tropper, McGill University; Brian Unger, Jade Simulations.

TWELFTH OCCAM USER GROUP TECHNICAL MEETING

2–4 April 1990, in Exeter, England

Advance notice is given of the 12th technical meeting of the occam user group, which will take place at the University of Exeter from 2nd to 4th April 1990. The provisional deadline for extended abstracts of papers is 1st December 1989. For further information about this meeting contact:

Dr Stephen Turner   Tel: +44 392 264048
Department of Computer Science   Fax: +44 392 263108
University of Exeter   Email: steve@uk.ac.ex.cs
Prince of Wales Road
Exeter EX4 4PT
England

THIRD TRANSPUTER/OCCAM INTERNATIONAL CONFERENCE

organised by the occam user group Japan
17–18 May 1990, Tokyo, Japan

OUG Japan is pleased to announce the third transputer/occam international conference to be held on 17–18 May 1990 in Tokyo. This is a place to find wide varieties of transputer-based parallel processing systems through original refereed papers, invited talks, tutorials and commercial exhibits. The official language of the conference is English. Simultaneous translation from English to Japanese is provided whenever necessary. The registration fee is ¥20 000 and this includes:

- ▹ admission to all technical sessions, tutorials and exhibits;
- ▹ admission to all lunches;
- ▹ admission to the final party.

The conference organising committee consists of:

Prof. Dr Tosiyasu L. Kunii (Chairman), University of Tokyo;
Prof. Dr Tadao Nakamura, Tohoku University;
Prof. Dr Eiichi Miyamoto, Hokkaido University;
Prof. Dr Shinji Tomita, Kyushu University;
Associate Prof. Dr Mitsuru Ishizuka, University of Tokyo;
Research Associate Dr Shigeki Sugano, Waseda University.

Full papers in English are solicited and should contain previously unpublished original high-quality results. The submitted paper should be typed, double spaced, about 10–20 pages in length with an abstract of 100–200 words and a maximum of 10 keywords. All papers will be peer reviewed and will be assessed with regard to their quality and relevance. The proceedings will be published for distribution at the conference and also for later dissemination. Papers must be submitted by 1st January 1990. Acceptance or rejection of the papers will be notified by 1st February 1990. Final camera-ready papers must be provided by 1st April 1990.

For more information contact the secretary of OUG Japan:

Mr Kazuto Matsui Tel: +81 3 505 2840
INMOS Japan KK Fax: +81 3 505 2844
4th Floor, Nº 1 Kowa Building
11-41, Akasaka 1-chome
Minato-ku, Tokyo 107
Japan

REPORTS

TRANSPUTERS, UNIX, AND FUTURE SUPPORT ENVIRONMENTS

a workshop of the OUG UNIX SIG and OUG operating systems SIG
21 February 1989, at Canterbury

Peter Welch, University of Kent at Canterbury

This joint workshop of the UNIX and Operating System special interest groups took place at the University of Kent on the 21st February 1989. The meeting was oversubscribed, but we found room in the end for over sixty participants.

Four presentations were made in the morning by speakers from INMOS, MEiKO, Parsys and Perihelion. Michael Poole, from INMOS, described the history (and some politics!) of the development of the TDS and forecast some of the most likely improvements that were currently planned. He invited proposals for new ways forward. That invitation, of course, applies to all concerned people – not just those who attended this particular workshop.

Simon Turner (MEiKO), Kevin Collins (Parsys) and Charlie Grimsdale (Perihelion) then explained the approaches their companies were taking to providing a more 'industry standard' approach for distributed (operating system) support services for multi-transputer applications. These are respectively known as Meikos, Idris and Helios.

All three of these approaches incorporate a strong level of UNIX compatibility, together with special support for the distributed nature of the transputer implementation. At the application level 'most' UNIX system calls are available and the parallel hardware can be hidden or made explicit to the application developer or user. The environments provide support both for development (through familiar UNIX tools) and execution of transputer applications (e.g. file services, message routers, etc.). All environments also provide support for 'raw' occam applications and the INMOS development tools.

Finally these approaches enable transputer hardware to be incorporated smoothly with more traditional servers (e.g. SUNs, VAXes, PCs, etc.), the 'join' being largely hidden by the software support services.

After lunch, the participants divided into three working groups (about twenty in each). No special themes were allocated to these groups, but they were left to thrash out freely their own worries 'spontaneously'. Finally, we all reconvened for a short plenary session at which the working group chairmen summarised what had been discussed.

Issues raised ranged from low-level matters ('Why can't we `fork()` on transputers?') to higher-level philosophy ('What is "UNIX"?', 'Why do we want it?', 'How much system support do we really need for parallel processing applications on transputers?', 'Why won't applications be directly portable between these environments?', ...). As can be seen, the higher-level philosophy tended to dominate!

A report on this workshop is (still) being prepared – we hope that it will be available by the time this newsletter appears. This report will primarily consist of copies of the slides used during the morning presentations and summaries of the afternoon working-group discussions. This will be mailed to the workshop participants, but further copies will be available (for a modest fee) from:

Mrs J. Farmer
Computing Laboratory
The University of Kent
Canterbury
Kent CT2 7NF
England

UNIX is a trademark of AT&T Bell Laboratories in the USA and other countries.

FRENCH TRANSPUTER USERS' WORKING GROUP ON PARALLEL SYSTEMS

T. Muntean, IMAG-LGI Laboratory, University of Grenoble

Research and development work on parallel programming and multiprocessor systems began in France in the early seventies. Since 1982, under a CNRS initiative the research activity in parallelism has been structured as a national action: *Programme de Recherche Concerté C3: Cooperation, Concurrence, Communication.* Work on parallel architectures and programming languages has been greatly supported within this framework.

The interest in using transputers for the design and application of parallel systems is now quite significant in France. Several research and development projects have been started since 1986 and this interest keeps growing with major development projects emerging. There have also been pioneering contributions to several 'big' ESPRIT projects using transputers.

Participants from France at OUG meetings have been noticeable since 1985. The University of Grenoble was very pleased to host the first OUG meeting outside the UK in September 1987. This OUG meeting was an opportunity to express our willingness to set up a local interest group. The first meeting of the working group took place in Grenoble on 22nd March and assembled a significant number of participants who presented their work. Nearly 150 people expressed their interest in such a forum for the exchange of ideas, tools and experience.

The group decided to act as a local working group and includes industrial, academic and research participants. The complementarity with the OUG has been emphasized since this first meeting.

The actual topics of work include:

▷ programming environments,
▷ software engineering,
▷ theory of parallel systems
▷ future architectures of parallel machines,
▷ scientific applications,
▷ simulation,
▷ real time and reactive systems,
▷ parallelism and neuroscience,
▷ various applications (CAD, AI, image analysis, graphics, robotics, DB, etc.),
▷ education.

The topics for future meetings (the next is planned in Sophia Antipolis, near Antibes, in July) will be determined by the interests of the participants.

TENTH OCCAM USER GROUP TECHNICAL MEETING

'Applying transputer based parallel machines'
3–5 April 1989, Enschede, The Netherlands

Albert L. Schoute, University of Twente

The 10th occam user group technical meeting was held at Enschede in the Netherlands between 3–5 April 1989. The conference was organized by the Control Systems and Computer Engineering Group of the department of Electrical Engineering at the University of Twente, under the dedicated control of conference chairman André Bakkers. This was the second meeting outside the UK – after Grenoble in September 1987 – but this time the environment and weather had been arranged to be less distracting. Because of the on-site accommodation and smooth organization at the DISH-hotel, the attendees could concentrate fully on the compact program: twenty-five papers, SIG-meetings, commercials, INMOS product information, forum discussion, even a short joke session, all within only two and a half days.

As is traditional, during the conference the familiar debates about belief and non-belief in occam, TDS and the transputer emerged at many occasions. At the forum discussion headed by Peter Welch questions were raised which apparently had no simple answers: Will the transputer be outmarked by the new Intel 860 chip? What will be the future of occam when alien languages get better development environments? Does success not depend on portability and is it not necessary for INMOS to adopt systems like Express or Helios as standards?

At some of the SIG-meetings questions about (knowledge of) available tools appeared still to be main items. At the 'Environments' SIG (Tools, OS, Unix and Networks) it was suggested that presentations on available software tools be made at future OUG meetings.

Despite all disputes over good and bad things, certain and uncertain developments, future survival or obsolescence, the OUG meeting made clear that – in applying transputer based parallel machines – practitioners eagerly absorb the new possibilities offered by the transputer and the occam way of thinking.

It was nice to see in the corridors not only the advances on the commercial front, but also a number of pretty demonstrations from the control engineering lab of the University of Twente.

The proceedings of the conference are published under the title *Applying Transputer Based Parallel Machines*, ed. André Bakkers, published by IOS (see page 3). An annotated summary of the presentations is given below (only speakers are mentioned).

EXPERIMENTS IN ALGORITHMIC PARALLELISM (PETER CAPON) As an experiment in speeding-up applications by exploiting parallelism inherent in the algorithm, a compiler has been parallelized as a pipeline. Using seven transputers a speedup factor of more than three could be reached. In optimizing, monitoring tools – such as GRAIL and 'matrix-of-meters' – played an important rôle.

PDS: ADVANCED PROGRAM DEVELOPMENT SYSTEM FOR TRANSPUTER BASED MACHINES (TRAIAN MUNTEAN) A description is given of a system which extracts the execution structure from an occam program and automatically partitions the program into processes that map onto a transputer network. A resident operating system takes care of dynamic allocation and routing of communication messages.

DESIGN, ABSTRACT DATA TYPES AND OCCAM (JON KERRIDGE) A design methodology is presented whereby abstract data types are mapped onto occam processes. For each function a command and reply protocol is specified. Attention is paid to the formal specification. The method may be useful in obtaining a parallel implementation of dataflow diagrams as produced by systems like SSADM.

CONFIGURATION TOOLS FOR A TRANSPUTER WORKSTATION (PETER CROLL) A configuration language is described which specifies a virtual machine configuration according to which the 'utility cluster' of a transputer workstation can be configured. Part of the workstation is a system ring providing the operating system support for the utility programs.

HIGHLY TRANSPARENT MONITORING OF PARALLEL SYSTEMS USING 'LOGICAL CLOCKS' (STEPHEN TURNER & WENTONG CAI) Language based monitoring influences program behaviour because it affects the order of process interaction. To eliminate this effect logical clocks are maintained which reflect the real time behaviour without monitoring. With these clocks 'slow motion replay' is possible without changing the real time event ordering.

A TRANSPUTER BASED VISUAL SYSTEM (AD LANGENKAMP) A visual system for training and simulation must be flexible and must give high real time response and image quality. A parallel architecture has been developed based on a graphics pipeline.

VISUALISATION OF 3D EMPIRICAL DATA: THE VOXEL PROCESSOR (WIM HUISKAMP) A parallel system architecture is described for the visualisation of 3D picture elements (voxels). Object data is divided in subcubes assigned to different transputers. Extra speedup is obtained due to some implementation tricks (incremental addition instead of multiplication, BYTEs to INT32 mapping).

MARVIN – MULTIPROCESSOR ARCHITECTURE FOR VISION (CHRIS BROWN & MICHAEL RYGOL) A multiprocessor architecture for machine vision is presented which is designed for the recognition of objects within 3D stereo images. A network of worker transputers is used in conjunction with a SUN 3/110 host. A software infrastructure written in parallel C provides the application environment.

MEMORY MANAGERS FOR TRANSPUTER NETWORKS (NEIL CARMICHAEL) Memory manager processes placed on every transputer of a network may provide a number of services to user processes, such as: concurrent file access, allocation and use of (heap) memory, caching etc. These functions may even be extended to include all kinds of process interaction. According the speaker, within the Shell company transputers may be useful in the workstation area, but will not replace CRAY computers (even if transputers are given for free) because of the investments already made.

PARX: A PARALLEL OPERATING SYSTEM FOR TRANSPUTER BASED MACHINES (YVES LANGUE) PARX is developed as a UNIX-like operating system for the Supernode architecture. It supports several levels of parallelism, including massive (light process) concurrency within user programs at little overhead. A kernel based on message passing implements homogeneous communication, making the actual processor allocation transparent to the user. The kernel offers multiple interfaces, thereby supporting different programming models, for example both occam and UNIX types of process management.

A COMMUNICATION PROCESSOR ON THE TRANSPUTER (HERMAN ROEBBERS) The presentation deals with the use of transputers as communication processors in a packet switching network. A deadlock-free routing algorithm for a mesh topology has been implemented in occam on a 3×3 transputer network, for which performance measurements are obtained.

ISSUES RAISED WHILE IMPLEMENTING LAYERED PROTOCOLS USING OCCAM AND THE TRANSPUTER (ROGER PEEL) A TCP/IP protocol interface to Ethernet is described, implemented on a transputer board with a LANCE Ethernet controller. At the level of the Internet Protocol (IP) double buffering by means of shared memory is employed. Presently, single stream TCP is supported.

AN OCCAM 2 IMPLEMENTATION OF HIGHER-LEVEL NETWORK PROTOCOLS: A CASE STUDY IN INTERFACING A MULTI-USER, MULTI-TRANSPUTER SYSTEM TO A LOCAL AREA NETWORK (MARK HEAPS) At the University of Kent the local area network is based on the Cambridge Ring. In order to connect the Meiko Computing Surface to this ring a VME-bus oriented front-end processor is used as interconnecting medium. Transport network protocols are implemented in occam, prompting the speaker to some remarks about occam and TDS.

TOPOLOGIES FOR LARGE TRANSPUTER NETWORKS: THEORETICAL ASPECTS AND EXPERIMENTAL APPROACH (FRANÇOISE BAUDE) Connection topologies and routing strategies are important issues in large networks. An evaluation tool has been developed for performing measurements under different message handling strategies.

It has been applied to a hypercube of dimension three. Other topologies (N-cube, N-star, De Bruijn and Kantz graphs) are investigated.

TRANSNET – A TRANSPUTER-BASED COMMUNICATION SERVICE (PETER WELCH) A simple message distribution service is described based on transputers as communication servers. The problem of a receiver blocking all communication can be solved by an end-to-end handshake, allowing a sender to output only on request. A 'free-flow' algorithm on a ring topology with private packets is recognized as satisfying a number of desirable properties in the best way.

USE OF OCCAM FOR VALIDATION OF DISTRIBUTED DISCRETE EVENT DRIVEN SIMULATION (A. H. DJANHANGUIR) The advantage of parallel architectures with respect to event simulation is obtained only when the event list handling is distributed too. Then the global time needs to be distributed along channels by attaching time-stamps to messages and – in some cases – sending null messages with timing information to avoid deadlock.

DISCRETE EVENT SIMULATION USING OCCAM (CHRIS NEVISON) Tools for discrete event simulation are developed in occam. With respect to the previous presentation, instead of null messages, more sophisticated deadlock avoidance methods, such as Nicol's appointment method, are investigated.

A STRUCTURAL DYNAMICS PROBLEM ON A NETWORK OF TRANSPUTERS (ALAIN COSNUAU) A configuration of forty T800-20 transputers is considered as hardware for finite element computations. The nature of the numerical equations gives rise to a ring topology.

APPLICABILITY OF A 16-NODE TRANSPUTER ARRAY WITHOUT EXTERNAL MEMORY (PATRICK VAN RENTERGHEM) The transputer array is presented as cost-effective for many applications. A graphical configuration tool is badly needed. The speaker expresses his enthusiasm for the Express system of Caltech (in contrast to other parallel C language environments).

PIPES: A TRANSPUTER-BASED PARALLEL ARCHITECTURE FOR AI REAL-TIME APPLICATIONS (GIAN PAOLO BALBONI) An architecture for AI applications is developed by an ESPRIT project. The application in mind is speech recognition using a parallel PROLOG dialect. The architecture provides local communication along a ring topology using links and provides non-local any-to-any communication by interconnecting local memories by a buffered, multistage, packet switched interconnection network (delta-topology). This communication structure is built out of 2 × 2 switching elements, which are made available as custom designed chips.

BIONIVISION: A TRANSPUTER BASED LASER SCANNER (KLAAS WIJBRANS) This embedded system application using transputers detects scratches and edge faults on glass plates. The application is characterized by a high pixel data rate. The computation can be shaped into a combination of parallel processing and pipelining. An appropriate hardware configuration has been built out of T414s and T212s. The speaker emphasized that the 16-bit processing capability of the T212 should not be underestimated.

A NOVEL ARCHITECTURE FOR DATA ACQUISITION AND ON-LINE ANALYSIS IN HIGH-ENERGY PHYSICS EXPERIMENTATION (J. C. VERMEULEN) In the context of the ZEUS experiment of the HERA electron-proton collider at Hamburg (planned to be operational in 1990) a two-transputer VME-module has been designed to read-out a large number of data channels and selectively trigger other detectors. Messages between these transputer modules are routed by a specially designed control and switch box, providing direct connections (via crossbar switches) and broadcast facilities.

TRANSPUTER BASED DATABASE ORGANISATION – AN EXAMPLE PROTEIN DATABASE IMPLEMENTED USING PIPELINE AND HYPERCUBE CONFIGURATIONS (L. C. WARING) A database containing protein sequence information is distributed over a transputer based system to allow parallel searching for specific patterns. A ring topology appears to be superior to a hypercube configuration. Use is made of an IMS M212 controlled hard disk for which a file system is developed.

AN OPERATIONAL PATTERN RECOGNITION SYSTEM ON TRANSPUTERS (EVERT BUITENWERF) A new generation of recognition systems for reading handwritten cheques is developed by the Dutch PTT. In order to satisfy the real time requirements, the reading process is parallelized and mapped on a tree configuration of transputers. Existing VAX-Pascal programs were easily embedded in an occam communication framework.

GRACEFUL TERMINATION – GRACEFUL RESETTING (PETER WELCH) Stopping any parallel application program might be a problem if not done properly. Seven examples are reviewed of how not to do it. Spreading 'poison' by putting it into the last message sent along all channels appears to be the correct solution under the constraint that all processes are reached by the (or any) initiator. This method also solves the more important problem of (partially) resetting a parallel system.

WHEN IS A TRANSPUTER NOT A JOKE?

Once again, and despite the sad absence of Roger Shepherd, there was a transputer-joke competition held at the end of the 10th occam user group technical meeting in Enschede. The services of the group's stand-up comic were secured at great expense – mainly to his self-respect – and amongst the contributions that David May read out were:

a: I say, I say, I say, my transputer's got short legs!
b: Your transputer's got short legs?
a: Yes, it SKIPs well but is hopeless at the long jump.

Moses came down from the mountain carrying the ten formal semantics, and said 'I was offered eleven, but I am sure there's a hitch with the parallel sea.'

Q: What's the difference between the transputer and the Titanic?
A: The Titanic had a band on it when *it* went down.

IN is a research student hoping to make it in parallel processing, OC is a majority shareholder in a transputer board company, while DAVID is an eminent scientist. IN arrives to find DAVID and OC standing at the bar, so who buys the next round? Well, you know, it's the old story: DAVID May, OC-can, but IN-must.

Q: Why is the new INMOS fab plant going to be in Amsterdam?
A: 'Cos it's easier to dope the silicon there.

Q: Why should you buy transputers from INMOS (Holland)?
A: 'Cos they will peddle you more cycles per chip.

Q: Why do INMOS benchmark all their transputers in Amsterdam?
A: 'Cos you get a better speed there.

1st Transputer: Have you heard about the new topology-independent, deadlock-free communications harness?
2nd Transputer: . . . .
. . . .
. . . .
. . . . yes!

Q: What's the difference between a *Guardian* writer and the INMOS C compiler?
A: Not a lot.

In this international medium, I feel obliged to explain a local reference: the *Guardian* is a London (formerly Manchester) newspaper with an affectionate but unenviable and wholly justified reputation for the accuracy of it sty pography.

Q: What's the similarity between a transputer and a very unsuccessful lover?
A: Their mega-flops.

Q: What's the similarity between a dung-beetle and a transputer?
A: They're both crap processors.

1st Transputer: Knock, Knock!
2nd Transputer: Knock, Knock!
(both together): . . . .

It would appear that those last two are deadlocked.

There was also a trade in acrostics; one that went through a number of β-test versions in the bar the previous night was 'Iann Now Makes Outdated Systems'. David's production release version was something like, 'Iann Never Makes Outdated Systems.' There were also a number of attempts at 'Only C Creates A Mess'; and a quotation attributed to David May himself, 'I Never Make Outlandish Statements'.

Surface mounting J-bend packaging has rather spoilt the '101 Uses for a Dead Transputer', and there were no particularly innovative contributions although I have been told that a pin-grid packaged one still makes a good bed of nails for a meditating pygmy shrew.

Extra "speakers' gratuities" were dispensed to the winners. The first bottle went, for the topology-independent deadlock-free communications harness, to S. Duncan of Southampton. *GJ*

NORTH AMERICAN TRANSPUTER USERS GROUP MEETING

5–6 April 1989, Salt Lake City, Utah

Dyke Stiles, Utah State University

The first full-fledged meeting of the North American Transputer Users Group was held in Salt Lake City, Utah, on 5th and 6th April. The meeting attracted approximately ninety participants. There were twenty-one papers presented and six vendors displayed hardware and software products. The meeting was hosted by Brigham Young University, Utah State University, and the University of Utah. Computer Systems Architects was the corporate sponsor.

Professor Hoare, in the keynote speech, traced the development of occam and its foundation in the need for reliable real-time systems. Hoare was in Utah to present the Organick Memorial Lectures at the University of Utah; these lectures covered the use of formal methods in the design of CMOS logic circuits.

The NATUG presentations were divided roughly into the areas of software, hardware, and applications. Operating and development systems dominated the software section, with papers on Trollius, Idris (a POSIX compatible OS developed by Real Time Systems Ltd), the Linda-based XTM machine from Cogent, and the Topologix package; demos of a fifth system, Parasoft's Express, were being handed out in the vendor area. Questions from the audience were directed at whether these systems might seriously slow down application programs, but the speakers indicated that experience so far has shown the overhead to be insignificant.

Chris Phillips described work underway at the University of Liverpool on the development of occam and Fortran numerical libraries for systems of transputers (see page 63). Other talks in the software section covered Modula and Prolog implementations, data and algorithmic parallelism, task allocation on reconfigurable systems, and verification of the transputer design.

In the hardware area, Fred and Barry Schlereth of Syracuse described the design of a 1000 node system aimed at numerical problems; this system has been simulated successfully on a Connection Machine. Fred made the interesting point that older algorithms may be more amenable to parallelization than their newer versions. Substantial up-and-running hardware projects were discussed by Mark Smith of Hewlett Packard, the Taylors from General Dynamics, and Tom Henderson of the Naval Ocean Systems Center. Henderson won the award for the most dense system with his picture of an IBM PC plug-in board holding thirty-two T800s and four C004s.

The applications papers covered a variety of topics. Martin Buhler of Yale described a working robotics vision system, and John Baker and Steven Seidman of George Mason University discussed natural language processing. The remaining talks addressed several areas of scientific computation and the use of transputers in computer engineering education.

A Proceedings, containing the papers presented at the meeting, is now being assembled and should be available in the near future.

On the administrative side, a permanent committee has finally been formed to keep NATUG moving ahead (see page 86). Dyke Stiles of Utah State University will serve as chair. Secretarial support will be provided through Mark Hopkins of INMOS' Colorado Springs office; Lyle Bingham of CSA will be in charge of soliciting NATUG contributions to the OUG newsletter, which we intend initially to distribute to NATUG members. If the NATUG submissions eventually reach a sufficiently high level, we shall consider publishing a separate NATUG newsletter.

The Fall NATUG meeting has been scheduled for 18–19 October in Durham, North Carolina (see page 10). The meeting next Spring will be 10–12 April 1990 on the west coast, and will be held in conjunction with the meeting of Working Group 10.3 of IFIP.

SECOND JAPANESE TRANSPUTER AND OCCAM INTERNATIONAL CONFERENCE

24–25 April 1989, Tokyo, Japan

Kazuto Matsui, OUG Japan, and INMOS Japan KK

The second transputer and occam international conference was held in Tokyo on 24th and 25th April 1989. This time we invited Prof. Peter Welch from the University of Kent, UK. He talked about some of the important ideas of transputers and SIG activities. This gave us an excellent opportunity to understand what European people are doing. We appreciated his coming and talking.

There was also good news for us in the tutorial session: Dr Yamamoto (Osaka Industrial University) showed us the video tape of TDS. The tape is very cheap because it is actually made by the transputer image processing system. I believe this video tape will be a bible, especially for beginners.

The technical sessions were as follows:

- *Transputer-based communications service*, Peter Welch (University of Kent, UK).
- *Fast calculation of FEM using parallel computer*, Toshirou Miyoshi, Naoki Takano (University of Tokyo).
- *The NOVI-II super parallel computer for signal processing*, Yashuhiro Ono, Naohisa Ohta (NTT Laboratory).
- *Transputer based neural network simulation*, Takashi Ohmori (The Tokyo University of Agriculture and Technology).
- *Transmission delay based design specification in VLSI components – link interconnections for highly parallel processing*, Takao Ichiko (Yamagata University).
- *Parallel processing system in computer integrated manufacturing*, Atsuo Takeda, Naouki Motomura, Mitsunori Kawabe (Research Laboratory, Yasukawa Electric Co.),
- *Managing hard read-time demands on transputers*, Peter Welch (University of Kent, UK).
- *A real-time implementation of a parallel algorithm for model-based control of a robot manipulator*, Kouichi Hashimoto, Kenji Ohashi, Hideki Kimura (Osaka University).
- *Image communication processor CP-200*, Tosiharu Saitou, Hirohisa Yamaguchi, Koji Kinuhata (KDD Co.).
- *Numerical analysis using Parallel C and Fortran*, Shunji Ido (Saitama University).
- *Development of parallel neural net simulator*, Hirosi Takada (Nippon Steel Co.).
- *Sensor fusion utilizing transputers*, Akihiko Takahasi, Masatosi Isikawa, Motoyuki Akamatsu (Industrial Products Research Institute).

SPECIAL INTEREST GROUPS

FORMAL METHODS SIG

Michael Goldsmith, Programming Research Group, Oxford

In the absence of Bob Stallard, I was persuaded to take the chair at the Formal Methods SIG meeting at Enschede. Some dozen people attended the meeting.

From Oxford came the news that our long promised course in *Formal Methods for Parallelism* had come one step closer to realisation, in that we had presented a week-long course with a mix of Z, CSP, and occam transformation to INMOS in January. We had hoped to produce a 'public' version in late June/early July, but had failed to find college accommodation for then. Southampton Transputer Support Centre pointed out that they have accommodation for hire, which we might turn to in case of future difficulty; and the transatlantic participants registered a plea that we should aim at dates close to the next OUG meeting so that trips could be combined. We are looking in to the possibilities.

I also reported limited progress with the occam Transformation System: Sun operating system changes breaking the window interface to the existing (proto-occam) tool have distracted me from the development of the occam 2 version, which is still some months from completion. Interest was also expressed in Z manipulation tools at the PRG, in particular a package called *f*UZZ (as being lint-like!) which typechecks the formal material in a LaTeX document prepared with Mike Spivey's Z macro package.

People interested in formal methods and occam would probably also be interested in Geoff Barrett's paper, *Verifying the Transputer*, which was being delivered (concurrently!) to the North American transputer users' group meeting in Utah. An article by David Shepherd and Greg Wilson under the title *Making chips that work* explaining the approach of the T800 verification project has also appeared in the British popular science journal *New Scientist* (Nº 1664, 13 May 1989, pp. 61–64).

One area where the use of more or less formal methods seems to be active is in the design of self-timed logic circuitry, using occam, CSP or CSP augmented with a *probe* construct. Sites known to be involved in such research include Southampton (sorry, I am not sure who to contact), Carnegie-Mellon (Brunfeld), CalTech (Martin) and Eindhoven/Phillips (Rem).

Helmut Zwittlinger, of Interkantonales Technikum in Switzerland, mentioned some work involving the use of occam to encapsulate interface interactions for the MASCOT real-time program design tool. Klaas Wijbrans (Twente) described a problem arising from analysis of simulations into basic blocks, and then eliminating parallelism from the resulting host of small processes; hand-worked exercises show promise, but a transformation strategy (and heuristics) are needed. I seem to remember that some time ago (three years? – perhaps at the Canterbury OUG meeting, or at STACS'87) somebody else mentioned a similar problem arising from emulation of electronic circuitry with occam processes describing the behaviour of each component: if anybody knows whether that research got anywhere, perhaps they could let us know.

There followed a discussion of graphical tools for the representation of occam programs, without much agreement on whether they were useful or what they should be like, for example how and at what level program text should be integrated with pictorial presentations. One cogent objection was that the 'standard' graphical representation of a network, as a number of boxes joined by lines representing channels, precluded the possibility of expressing even the limited dynamic parallelism that occam offers; there is no notation for the synchronisation implicit in the termination of a PAR construct.

A pertinent topic which was not actually aired during the SIG meeting, but was the subject of some whispering in the back row of the preceding Education and Training SIG (which had a substantial overlap of attendees) and lay behind some of the Panel Discussion questions, was a worry about INMOS' internal commitment to Formal Methods and occam for their own software, with the advent of a new generation of tools written in C. The answers in the panel session may have eased that disquiet somewhat.

GRAPHICS AND IMAGING SIG

Hugh Webber, RSRE, Great Malvern

There was a meeting of the Graphics and Imaging SIG at the tenth OUG meeting held in Enschede, Holland. The interest divides up into the two groups in the heading, i.e. graphics which seems to be generally the visual realisation of computational results, ray-tracing and other solid modelling; and imaging which is tied to processing and extraction of information from images captured by some sensing system.

There is no clear boundary between these two areas and they have common problems. Two of the problems discussed were the apparent lack of high resolution graphics hardware (i.e. minimum 512 × 512 pixel, and 8 bits per colour gun) required for realistic images and the associated software to drive it. If people would like to send me information on any hardware and software that they know of I will compile a list and either distribute it at the next OUG or put it into this newsletter (or both, let me know what you want). I can be contacted at the address (postal or e-mail) given at the back of this newsletter. Obviously any experience, good or bad, of using the hardware or software would be useful feedback. I can edit comments to be non-attributable if required.

At the other end of the resolution scale we (RSRE) have developed a low cost graphics system consisting of video capture and display devices controlled by T2s. These are capable of capturing and displaying 256 × 240 pixel images at about 20 frames per second. Display can be 256 greyscales or G170 controlled colour. The boards were designed to be cheap, total component cost of less than £150, and are ideal for student use etc. because of the low cost and because no special high performance monitors are required. Details can be made available if there is enough interest. Also I will collect and distribute any other graphics hardware ideas.

To make the next Graphics and Imaging SIG (Edinburgh) more interactive if you have any burning issues you would like to see discussed please drop me a line beforehand and I will try to come prepared with some info/ideas. The effectiveness of a SIG is really dependent on those who attend so it is up to you.

IMAGE PROCESSING AND VISION SIG

Neil Carmichael, Shell Research

The Imaging SIG now has a chairman. He is Neil Carmichael of Shell Research. He would like some feedback on imaging activities of interest to occam and transputer users. Such feedback might concern

- ▹ hardware, e.g. interfacing, video rate i/o;
- ▹ software, e.g. toolkits for image processing and analysis, prototype applications;
- ▹ algorithms, especially (of course) parallel ones.

Communications – conventional mail or FAX – should be addressed to

Neil Carmichael Fax: +31 70 11 39 10
KSEPL – Shell Research
Volmerlaan 6
2288 GD Rijswijk ZH
The Netherlands

Those who get in touch, and also those who are currently identified in the OUG membership records as being interested in the Imaging SIG will receive further material on the SIG.

HARDWARE SIG

Denis Nicole, University of Southampton

The SIG meetings at Enschede were held on Monday evening, with the hardware meeting attracting over a hundred participants. I started the meeting off with a short account of visits to India by Jon Kerridge and myself. The Indians have a large project at the Centre for Development of Advanced Computing in Pune, near Bombay, building a thousand transputer machine. They intend to use it mainly for oil reservoir and land resource modelling. The hardware team seems as strong as any in the West, but they claim that their main strength will be in software. There are an awful lot of graduates in India, and this group and its software team in Bangalore are definitely people to watch.

We were lucky to have several Americans at the SIG. Craig Davidson of Levco described the Linda machine from the US company Cogent. This is designed to support the Linda global tuple-space data model and includes a fast interprocessor parallel bus as well as transputer links. He also suggested the use of a cache chip from Fujitsu as a possible way of exploiting the new fast memory interface in the T801.

Ronald Cok of Kodak had some interesting results on occam code performance which ran counter to the accepted advice to put data in the on-chip memory of a transputer and code in the off-chip memory. A program of his which took unit time with both code and data on chip showed a factor of 1·5 times slowing with code off chip compared to only a 1·2 times slowing with data off chip. Both off chip resulted in a 1·8 times slowing. These results are supported by some recent work on functional language compilation by Hugh Glaser at Southampton. He found that tight loops and frequent function calls cause a lot of flushing of the prefetched code word (four instructions) and substantially increase the cost of off-chip code.

There was much discussion about providing through-routed communications for transputers. Despite a lot of speculation, INMOS declined to make any announcement about their plans, but there was general agreement that a link through-routing chip is urgently required for general-purpose transputer computers. On the circuit switching front, the substantial delays associated with C004 switches also attracted attention; a three-layer C004 switch halves the bandwidth of a 20 MHz link. Such switches are found in larger Parsytec and several other machines. Some custom switches do not perform better; during one of the main sessions Peter Welch of Kent reported that the 10 MHz links of his MEiKO box deliver 840 kbytes/sec with direct connection but only 660 kbytes/sec when routed through two MEiKO switches.

The INMOS product presentation on Tuesday had a few imminent offerings: a fully functional 25 MHz T425, sampling in April of the T801 – an enhanced T800 with a two-cycle non-multiplexed memory interface and a T810 promised for 1990 with more links, faster links (with the current protocol) and more single cycle instructions.

Many thanks to our hosts at the University of Twente for a very enjoyable meeting, and for an impressive display of an active compensation of bending in a driven beam. See you all in Edinburgh!

EDUCATION AND TRAINING SIG

Roger M. A. Peel, University of Surrey

The recently-renamed Education and Training SIG meeting in Enschede was attended mainly by members of various universities, with contributions also from representatives of two of the UK Transputer Initiative support centres. The meeting started with the complaint frequently heard in academic circles that occam 2 is not available on multi-user machines, and is therefore expensive to support for large classes.

The majority of the meeting was spent discussing parallel software design methods and how they were taught. Teaching occam in conjunction with either Petri Nets or CSP or both was common and highly recommended. CCS was often included in concurrency courses, too.

The engineering need for proof and verification techniques was highlighted by reports of recent air disasters. Difficulties with demonstrating the effects of parallel occam processes for military purposes were noted, as was the increasing awareness of safety-critical applications and standards to monitor them.

The continuing controversy on the benefits of occam versus parallel versions of conventional languages was aired. One particular concern was the announcement that INMOS was writing the new occam compiler in C. The security implications of this were questioned by many attendees. It was agreed that occam programming was highly justifiable in terms of rapidity of design and implementation, ease of maintenance, performance and security.

Finally, it was agreed that the Education and Training SIG should consider how best to teach parallel program design. One of the most promising ideas was that the E&T SIG should organise a workshop at which a real engineering problem is implemented in parallel over a couple of days so that educators can discover the problems of running such courses themselves. Any ideas for suitable problem areas would be appreciated by the Chairman.

TECHNICAL CONTRIBUTIONS

WHAT MAKES OCCAM SO INTERESTING?

Ian Glendinning, Fidelio Software GmbH

For some time now, I have been somewhat dismayed by the lack of positive things being said about occam on the occam and transputer electronic mailing lists. Recent discussion on the mailing lists about 'what makes transputers interesting?' reproduced in OUG newsletter Nº 10, seems to have been generally along the lines that transputers are really neat, but that occam is lousy. While it is true that occam has limitations, I personally consider them to be far outweighed by the advantages it has to offer.

I cannot really argue that occam has not been an obstacle to the general acceptance of transputer systems, but I would argue that this is really a problem of education of the consumers. In this respect, although INMOS appear to regard themselves primarily as chip manufacturers, I tend to think it would benefit them to invest a little effort in making occam available on machines other than transputers. If people got hooked, they would surely want to buy transputer chips, since they make such good occam engines, and make distributing occam programs so easy.

For me, the most exciting and interesting thing about occam is not just that it is a language designed specifically with parallelism in mind. Even more important is that it is based on a well-defined mathematical model of concurrency, which gives the language exceptionally clean semantics. Few other languages can lay claim to such a combination of qualities. The mathematical model in question is that of Communicating Sequential Processes (CSP), due to Professor Tony Hoare[1]. Because of this formal basis, there exist a large number of algebraic laws relating different programs which have the same meaning. For instance, a rather simple and intuitive law is that the process

```
PAR
  P
  Q
```

has the same meaning as the process

```
PAR
  Q
  P
```

where P and Q are arbitrary processes. This equivalence can be written with an equals sign, as

```
PAR       PAR
  P   =     Q
  Q         P
```

Another simple, but interesting, equivalence is that

```
PAR
  c ! e   =   v := e
  c ? v
```

where *c* is a channel, *e* is an expression, and *v* is a variable. A slightly less obvious example is that

```
ALT
  c ? x          PAR
    d ? y   =      c ? x
  d ? y            d ? y
    c ? x
```

It should be emphasised that the above laws express identities between the processes on either side of the equals sign. They are not just rough equivalences, which break down in certain circumstances. They always hold. Thus, laws which involve arbitrary processes (like *P* and *Q* in the first example) are extremely general. This is to be contrasted with the semantics of most commonly available languages, which typically contain many exceptions and irregularities.

This is all very elegant, but so what? How does this help us? Well, it has been shown that a set of laws exist, which completely characterise the semantics of the language[2]. (Actually, the reference cited shows this for occam 1, since that untyped version of the language is easier to deal with, but it is expected that the same ideas can be extended to include occam 2.) In particular, for `WHILE`-free programs, it is shown that it is possible to use the laws to transform any program into a 'normal form', containing only assignments, `IF`s and `ALT`s (also obeying some other constraints). The special thing about the normal form is that any two programs which are semantically equivalent (have the same meaning) will have the same normal form. Thus it is possible to check whether two different programs really do the same thing! What is more, the steps to transform a program into its normal form can be (almost) automated. If a tool existed which could check the equivalence, or otherwise, of two programs, it would be immensely valuable. For instance, it would enable you to check with one hundred percent confidence that that 'small edit' really did not change the meaning of a program. From a slightly different standpoint, the laws could be used to make transformations to a program, which would be guaranteed to be meaning-preserving. This could be helpful, for instance, when trying to improve its execution efficiency.

On a more down to earth level, the formal properties of the language enable the occam compiler to perform extremely strong checking of code, which is a great help in making sure a program really does what you think it does. For instance, it is the only compiler I know which actually enforces the complete absence of variable name aliasing. This particular kind of checking is required by the simple substitution semantics of abbreviations and procedure parameters. On the other hand, the programmer is not prevented from doing low level things such as accessing memory mapped peripherals, or accessing the same machine word as more than one different type. Those things are allowed, but treated more hygienically than in most languages, with the `PORT` and `RETYPES` declarations, respectively. The strictness of the compiler can be irritating at times but, for me at least, this is outweighed by the extra confidence I have in a piece of code which has passed its stringent requirements.

One criticism of occam which is often raised, is that it is not recursive. It is interesting to note that the underlying CSP model is indeed recursive, and there is no reason in principle why a recursive version of occam could not be implemented. No syntactic changes to the language would be necessary. The reason the definition in the occam 2 reference manual[3] is not recursive is simply to allow an efficient and secure implementation of the language on the transputer. For the same reason, there is no real reason why the process count on a replicated PAR construct could not be allowed to be variable. However, the current definition does have the advantage that the amount of memory needed to execute a process is known at compile time, and so there is no possibility of a program crashing at run time, with stack or heap overflow. If this were not the case, the program transformations mentioned above would no longer, in general, be meaning preserving, and it would not be possible to make use of the formal properties of the language (at least, not so easily).

Occam is not everything, of course, and I hope that we will be seeing, in due course, an 'occam 3', containing some of the abstract data types which were left out of occam 2. (Give the guys at INMOS a chance!) Records, more general arrays, and an enumeration type might be expected, for instance. Looking further ahead, what might be the line of development? It has been said that occam is an 'assembly language' for parallel processing. This is not meant to imply that the conventional aspects of occam are at the level of an assembly language, as it is clearly a 'high level' language (whatever that means) in that sense. Rather, the communication facilities provided by channels are sufficiently primitive to allow higher level communication constructs to be efficiently built on top of them. At the moment, it is not entirely clear what such constructs ought to be, but when it is, a new language will presumably be designed around them. Channels might be regarded as the GOTOs of parallel programming, which will eventually be replaced by more structured ideas.

In conclusion then, it is really the formal properties of occam which make it such an interesting language, and it is the things which the occam compiler does not allow you to do, almost as much as those which it does, that make it such a useful tool. I look forward to further occam inspired developments, and, in the mean time, sincerely hope that the elegance and security of the language will come to be appreciated by a wider circle within the computing community. By the way, I'm also a fan of TDS[4], but that, as they say, is another story!

References

[1] C. A. R. Hoare, *Communicating Sequential Processes*, Prentice Hall International series in Computer Science, 1985, ISBN 0-13-153271-5, ISBN 0-13-153289-8 (pbk).

[2] A. W. Roscoe and C. A. R. Hoare, *The Laws of occam Programming*, in Theoretical Computing Science, 60(2) pp.177–229, September 1988, *also* Oxford University Computing Laboratory Programming Research Group, Technical Monograph PRG-53, February 1986, ISBN 0-902928-34-1.

[3] INMOS Limited, *occam 2 Reference Manual*, Prentice Hall International Series in Computer Science, 1988, ISBN 0-13-629312-3.

[4] INMOS Limited, *Transputer Development System*, Prentice Hall International, 1988, ISBN 0-13-928995-X.

Ian Glendinning
Fidelio Software GmbH
Widenmayerstraße 6
D-8000 München 22
West Germany

Tel: +49 89 2123080
JANet: ian@uk.ac.soton.esp.vl

COSMOS - COMPUTER ONLINE STRUCTURAL MODEL OF OCCAM SYNTAX

Harold J. Curnow, Advanced Technology Group, CCTA

This paper describes a model of the syntax of the occam2 language which can be examined by a user of an INMOS Transputer Development System (TDS2) concurrently with program development work. The intention of the model is that the user should be able to access directly the definition of any of the syntactic objects in the language, and from that point examine the definitions of the objects used in the first definition, and so on. *Yes*, since the language definition is recursive, so is the model. That this can be achieved simply is a consequence of the design of the TDS editor, with its concept of folded files and filed folds. The model has been set up and can be examined using only the standard features of the TDS, although it could be examined by an occam program running under TDS if a more elaborate display were required. Once the basic syntax has been represented it becomes possible to add information about semantics, to provide examples, perhaps even to run demonstrations, turning the model into an online language reference and tutor.

The Transputer Development System (TDS)

For those not familiar with the TDS folding editor, the following description will cover those features used in the syntax model; those familiar with TDS may safely skip to the next section.

The TDS editor is a program running on the Transputer. It provides an occam program development environment in which source texts can be created and edited, compiled and run, but it can process any text, not just occam. Documents are held on the hard disk of the host PC as PC-DOS files, and a server program on the host gives TDS access to them as well as to the PC keyboard and screen.

The editor treats any document as a series of lines. Each line may be either a simple text line, or a *fold* header line. It is possible to fold up any number of lines and represent them by a single fold header line. When a fold is displayed on the screen by the editor, only its own simple text lines and any fold header lines that it contains are shown. The contents of the folds which the header lines represent may be seen by instructing the editor to *open* or to *enter* them – with single keystrokes. (The difference between 'open' and 'enter' lies in the way the fold contents are displayed in relation to the containing folds; easier to see than explain.) By opening all the folds in a document the whole document would be displayed. When required, folds can be closed or exited, returning the outer fold to its original state.

Folds are of two kinds, *simple* and *filed.* These appear to the user to behave similarly, but there is a difference of implementation. The contents of a simple fold are hidden behind the header line, but are still part of the containing document and are stored with it. The contents of a filed fold, on the other hand, are held in their own DOS file, and the header line conceals a reference to that file. This means that it is possible to have any number of appearances of the same filed fold, in the same or different documents, from each of which access can be obtained to the same set of lines. This is a feature which is exploited in the TDS, for example to provide access to libraries, and the editor and file handling utilities make it very easy to set up such arrangements.

The syntax model

Each object in the syntax is represented by a filed fold within the outermost syntax fold. So the model resides in (about seventy-five) DOS files, each named for the object whose definition it contains. When entered or opened by the editor each fold reveals the text of the definition of the object, additional syntactic and semantic comments, and filed fold headers which are attached to the files containing the definitions of all the other objects referred to in this fold. By opening or entering these inner folds, and then the folds they contain, the whole structure of the syntax definition related to the first object can be explored and displayed.

The model has been set up on an INMOS TDS2 system, version D700D, running on a Sension Transputer Evaluation System with a T414 transputer connected to an IBM PC-AT. The principles on which it is based should, however, apply to any TDS system. The model represents the occam 2 language according to the INMOS *occam 2 Reference Manual*, and is in effect a transcription of the formal syntactic definition of the language given therein. It is intended to add to the model other useful information from the reference manual, concerning the semantics and the finer points of the syntax, in the form of comments or additional enfolded objects.

The model can be used in a number of different ways, either as a stand-alone reference and tutor, or as an online tool accessible during program development. The following might be the way for a programmer to check on the definition of *process* while editing his program, using a technique suggested by K. Everett (a television clown who draws a door on the wall, opens it and goes through it).

Example – online use

Instructions for examining the syntax of *process* while editing an occam source text:

1. Create an empty fold [2 keystrokes]
2. Name it process [7 keystrokes]
3. Attach it (to the DOS file of that name) [1 keystroke]
4. Enter Fold [1 keystroke]

The screen would appear as shown in figure 1. The text above the line gives the eight alternative definitions of *process* ('::=' means 'is defined as', '|' means 'or'), while the six lines beginning '...F' are the filed fold header lines leading to the six objects used in the definitions. I have chosen that *process* itself does not appear here since its contents are already visible.

```
{{{F process
  ::=  SKIP              -- starts, does nothing, terminates
  ::=  STOP              -- starts, does nothing, never terminates
  ::= action | construction | instance
  ::= caseinput
  ::= specification      -- declarations etc  = block head
      process
  ::= allocation         -- PLACE ...
      process
  ------------------------------------
  ...F action
  ...F construction
  ...F instance
  ...F caseinput
  ...F specification
  ...F allocation
}}}
```

Figure 1: the *process* fold

To illustrate the model further, suppose the programmer now investigates the *construction* option by opening the second filed fold. Then within this he selects the *sequence* option, opens that, finds that a sequence contains one or more processes, so opens *process*; the screen looks figure 2. The notation `{ process }` means 'a list of processes written under one another'. You cannot see all that on the screen at once of course, but you can scroll through it. I have inserted the comments on the close-fold '}}}' lines for clarity.

When the programmer has finished exploring the syntax, he can exit all the syntax folds, detach the *process* file and delete the fold (in that order!), and he will be back looking at his program as when he started.

Conclusion

This has been a quick look at the COSMOS model in its present state. Another tribute to the TDS is that it has taken almost as long to write this note as to set up the model – seems like it anyway. Future developments, if anyone shows an interest, would be to hang more of the reference material onto the model, to organise ways of accessing the model from different existing TDS installations, and consider whether it would be worth creating an occam front-end program to traverse the model and present a better interface to the user. Finally there is the possibility of putting demonstration executable programs into the model, to illustrate the semantics – an instructional exhibition with recursive spatial geometry!

Caveats

▷ There are implementation limits to the number of folds that can be opened or entered at one time, so recursion cannot in practice be infinite.

```
{{{F process
  ::=  SKIP              -- starts, does nothing, terminates
  ::=  STOP              -- starts, does nothing, never terminates
  ::= action | construction | instance
  ::= caseinput
  ::= specification      -- declarations etc  = block head
      process
  ::= allocation         -- PLACE ...
      process
  -------------------------------------
  ...F action
  {{{F construction
    ::= sequence |conditional |selection |loop |parallel |alternation
    [[  SEQ       IF           CASE       WHILE  PAR       ALT       ]]
    ------------------------------------------------------------------
    {{{F sequence
      ::= SEQ
            { process }
      ::= SEQ replicator
            process
      -------------------
      {{{F process
        ::=  SKIP        -- starts, does nothing, terminates
        ::=  STOP        -- starts, does nothing, never terminates
        ::= action | construction | instance
        ::= caseinput
        ::= specification  -- declarations etc  = block head
            process
        ::= allocation     -- PLACE ...
            process
        -------------------------------------
        ...F action
        ...F construction
        ...F instance
        ...F caseinput
        ...F specification
        ...F allocation
      }}}                        -- end of "process"
      ...F replicator                   -- <SEQ>
    }}}                          -- end of "sequence"
    ...F conditional
    ...F  etc etc
  }}}                            -- end of "construction"
  ...F instance
  ...F caseinput
  ...F specification
  ...F allocation
}}}                              -- end of "process"
```

Figure 2: the *process* fold opened at *construction*, *sequence*, and *process* again

- Because of the recursion, some of the TDS facilities and utilities will not work. *Search* is one; I suspect listers would have a hard time too – they try to open everything inside everything! The only facilities needed to construct the model were TDS commands and the file handling utilities Attach/Detach File.
- The model is set up according to my interpretation of the occam 2 syntax etc. and I may be wrong.

Acknowledgements

The original idea for COSMOS came from observing a Hypercard based demonstration of the structure of a Government IT Method under development, created by Logica (Cambridge) Ltd for a feasibility study for CCTA. This paper is published by permission of CCTA.

Transputer and occam are registered trade marks of the INMOS Group of Companies. The INMOS Ltd *occam 2 Reference Manual* is published by Prentice Hall, ISBN 0 13 629312 3. TDS2 is described in the INMOS manual, *Transputer Development System*, also published by Prentice-Hall, ISBN 0 13 928995 X. All other trade marks acknowledged.

Harold J. Curnow   Tel: +44 1 217 3209
Advanced Technology Group   harold@uk.co.cctal
Central Computer and Telecommunications Agency   or . . . !ukc!cctal!harold
Riverwalk House
157-161 Millbank
London SW1P 4RT

EFFICIENT MULTIPLE BUFFERING IN OCCAM

Geraint Jones, Programming Research Group, Oxford

One of the virtues, it is said, of the transputer is that it can do all that communication *at the same time* as the all that arithmetic – or whatever it is a processor does. The great virtue, it is also said, of occam is that it is ideal for programming such concurrent activity. This is an exploration of these assertions, prompted by a fragment of code in a paper[5] presented at the tenth occam user group meeting.

Keeping things going

Suppose *T* is the type of some fixed size buffer – it will probably be [N]BYTE for some *N* or other – and that *in* and *out* are channels with that protocol, one placed at an input link and one at an output link of a single transputer. A program like

```
WHILE TRUE
  T x :
  SEQ
    in ? x
    p(x)
    out ! x
```

reads an infinite sequence of buffer values from the input link, applies p to each of them, and then copies them out of the other link.

The input link is working while the input from *in* is executing, the processor is working while the call of p is executing, and the output link is working while the output to *out* is executing. However, since only one of these three processes is executing at a time, only one of the three *engines* – the link drivers or the processor – is in use at a time. (I am assuming that N is so large that the time spent by the processor on loading the link registers is insignificant compared with the time it takes to ship the data along a serial link.) If executing p takes a time about equal to the time it takes to ship a value of type T over a link, and nothing else is happening in the transputer, you are getting only a third of the potential performance of your microprocessor.

The answer, since each engine 'uses up' a process to keep it busy, is to write a program with three parallel processes in it. The program in figure 3 also reads an infinite sequence of buffer values from the input link, applies p to each of them, and copies them out of the other link. However it can be doing these three things at once. In order to do that it is capable of reading up to three more items than it has output, unlike the earlier sequential process which could only read one item ahead.

That, it would seem, is that: all three engines can now be running at once, so this has clearly tripled the performance of the previous worst case, has it not? Well, no, regrettably not quite: although it is possible for all three engines to run at once, it is not possible for all three engines to be running at once all the time.

The problem is that between each of what you might call the useful activities, the three processes have to spend some time in a communication on *c.xy* or *c.yz*. Even if the processes are keeping up with each other, so that no time is spent waiting for the other partner in the synchronization, some time has to be spent on the transfer of data: over *c.xy* from x to y, and over *c.yz* from y to z. Although these are channels internal to the transputer – so the transfers are block moves and are presumably much faster than the link transfers – each link engine is guaranteed to be idle for one block move time per link communication, and the processor is guaranteed to be 'otherwise occupied' for two block move times per call of p.

Keeping things going at all costs

The solution that a number of people have adopted is, in effect, to replace the communication of T values by communication of pointers to variables containing those values. The pointer can be quite small in comparison to the value, so the time overhead is greatly reduced.

Of course, there are no *pointers* as such in occam, so you have to use indices into an array instead of pointers, and must explicitly manage the pool of variables to which they point. For the three place pipeline, three buffer variables are needed at a time so an array $[3]T$ is declared to be the pool of shared variables. Each T variable in the program is replaced by an `INT` variable which will serve as a pointer into that pool, and most accesses to the T variable are replaced by accesses to the pool indexed by the `INT` variable. The exception is that instead of communicating a T value over an internal channel, it is necessary only to pass the value of its index in the pool.

```
CHAN OF T c.xy, c.yz :
PAR
  WHILE TRUE
    T x :
    SEQ
      in ? x
      c.xy ! x
  WHILE TRUE
    T y :
    SEQ
      c.xy ? y
      p(y)
      c.yz ! y
  WHILE TRUE
    T z :
    SEQ
      c.yz ? z
      out ! z
```

Figure 3: a three stage pipeline

```
VAL INT n IS 3 :
[n+1]CHAN OF INT c :
PAR
  PAR i = 0 FOR n
    SEQ
      c[i] ! i
      WHILE TRUE
        INT ix.wi :
        SEQ
          c[i+1] ? ix.wi
          c[i] ! ix.wi

  CHAN OF INT new IS c[0] :
  CHAN OF INT c.xy, c.yz :
  CHAN OF INT used IS c[n] :
  [n]T b :  -- * shared * --
  PAR
    WHILE TRUE
      INT ix.x :
      SEQ
        new ? ix.x
        in ? b[ix.x]
        c.xy ! ix.x
    WHILE TRUE
      INT ix.y :
      SEQ
        c.xy ? ix.y
        p(b[ix.y])
        c.yz ! ix.y
    WHILE TRUE
      INT ix.z :
      SEQ
        c.yz ? ix.z
        out ! b[ix.z]
        used ! ix.z
```

Figure 4: a pointer-passing pipeline

A mechanism has to be provided to allocate the buffers on first use, and to collect them after last use. In our case, the program ends up looking something like figure 4, where the replicated PAR arranges that the sequence of INTs available at *new* is $0, 1, 2$ followed by the sequence returned at *used*. (Exercise for the reader: show that only one of *ix.x*, *ix.y*, *ix.z* has any particular value at any one time when the corresponding process indexes the array *b*.) Roger Peel's program[5] was actually, apart from an allocation mechanism different in detail, the same as this program except that he had declared n to be two rather than three. This has the effect of throttling the pipeline so that at most two of the three activities can happen at once. (Exercise: what does a T-passing program equivalent to Roger's pointer-passing program with $n = 2$ look like?)

The striking similarity between the programs in figures 3 and 4 is meant to inspire your confidence that they have a similar meaning. I hope that you can see that this pattern might be followed with any program where a number of values of the same type were being passed between a number of processes, provided all the processes were on a single transputer so that the pool of buffers was in a shared store. In general the pool handler process has to hand out indices over a number of different *new* channels, to a number of different processes: wherever the pooled values are created or copied a new buffer must be allocated. Similarly, wherever a value is destroyed a buffer must be released so the pool handler has also to collect indices from a number of *used* channels.

On the assumption that the time taken to do something with an INT is dominated by the time taken doing something to a T, the overheads are now negligible, and we get the more-or-less tripled performance we expected.

The cost of keeping things going

The problem with a program that passes array indices and shares an array is precisely the problem with any program using pointers: since the identity of a particular variable can be determined by data, it is not always possible to check statically that anti-aliasing and disjointness conditions hold. In this particular case, for example, the only way to find out which part of *b* is accessed by

```
INT ix.y :
SEQ
  c.xy ? ix.y
  p(b[ix.y])
  c.yz ! ix.y
```

is to run the program and see what turns up on *c.xy*.

This means that there is no way that a compiler can check that the parallel processes in figure 4 make disjoint use of the locations of *b*. I should emphasize that it *is* possible for the programmer to check this disjointness: that was the essence of an exercise in the last section. So if you are prepared to take the burden of responsibility for proving that your program obeys the rules, albeit in an uncheckable way, you can turn off the usage checks of the compiler and plough on. You can spend the time that the compiler takes to generate code, laboriously proving by hand that it is indeed a safe program.

```
CHAN OF T c :
PAR
  T x :
  SEQ
    P(x)
    c ! x
    Q(c)
  T y :
  SEQ
    R()
    c ? y
    S(c, y)
```

Figure 5

```
T x, y :
SEQ
  PAR
    P(x)
    R()
  y := x
  CHAN OF T c :
  PAR
    Q(c)
    S(c, y)
```

Figure 6

```
T x, y :
SEQ
  PAR
    P(x)
    R()
  CHAN OF T c :
  PAR
    Q(c)
    S(c, x)
```

Figure 7

In this respect, it is a pity that the INMOS compiler does not (as far as I know) let you suppress the checking of just one variable – in this case b – without suppressing the checks on all variables in a compilation. The price for using the pointer trick would seem to be that the programmer is obliged to do all the checks the compiler would otherwise have done. (But, of course, a disciplined programmer is doing that anyway while writing the program!)

Keeping things going with a clean conscience

The strategy for getting around this cost is to eliminate the transfer of data by 'taking the process to the data' rather than the other way about.

Recall that each of the pairs of communications in an occam program serves two purposes: as well as the data transfer, there is also a synchronization. This is encapsulated in the equivalence of the program in figure 5 to that in figure 6. Now, since x is not free in the second **PAR**, the assignment can be discarded by replacing the uses of y by uses of x, so this program is in turn equivalent to that in figure 7. If Q and S have the same form, you can proceed to eliminate the communications on c, one by one.

Suppose you have a program consisting of a PAR of two WHILE loops, the bodies of which communicate with each other exactly once, like that in figure 8. You can unwind each of the loops once to get a program with a distinguished first communication on the channel, which you can then eliminate. That makes a program with one communication fewer, and a component which is again a PAR of WHILE loops, in which you can unwind the loops again. Eventually you can get the program into a form where you can see how to wind up the loops again, and show the PAR-of-WHILEs program equivalent to the WHILE-of-PARs program in figure 9. (Exercise: do so; you will need to unwind the loops about three times.)

We have seen that a two-stage pipeline (figure 8) can be implemented by a sequence of parallel processes (figure 9), and since this has the form of figure 10 it is tempting to try replacing the three-place pipeline in figure 3 by figure 11. Indeed this is the closest approximation to the original of which I know, which uses no internal copying of data but can be checked by a compiler. (It is to be found in,

```
CHAN OF T c :
PAR
  WHILE TRUE
    T x :
    SEQ
      in ? x
      c ! x
  WHILE TRUE
    T y :
    SEQ
      c ? y
      out ! y
```

Figure 8

```
T x, y :
SEQ
  in ? x
  WHILE TRUE
    SEQ
      PAR
        in ? y
        out ! x
      PAR
        in ? x
        out ! y
```

Figure 9

```
VAL INT n IS 2 :
[n]T b :
SEQ
  in ? b[0]
  WHILE TRUE
    SEQ i = 0 FOR n
      PAR
        in ? b[(i+1)\n]
        out ! b[i]
```

Figure 10

```
VAL INT n IS 3 :
[n]T b :
SEQ
  in ? b[0]
  PAR
    in ? b[1]
    p(b[0])
  WHILE TRUE
    SEQ i = 0 FOR n
      PAR
        in ? b[(i+2)\n]
        p(b[(i+1)\n])
        out ! b[i]
```

Figure 11

for example, reference [7].)

The reason this program is different from the original pipeline is that it is not 'a buffer' in the sense of reference [6]. Just like the pipeline, this one can input up to three more items than it has output; however once it has taken any input, it can never have input *fewer* than one more item than it has output.

So long as the additional latency does not matter, this is a statically checkable program which can keep two links and their processor busy at the same time without any additional copying of data. If a call of p takes the same length of time as the transfer of a value of type T over a link, this program has the same tripled performance as figure 4; in fact it should be marginally better.

Why a clean conscience cannot be cheap

The following is an informal argument that there is no reasonable statically checkable program with the same behaviour as the pipeline in figure 3 and the same efficiency as the pointer-passing program in figure 4. By *behaviour* I mean the possible orders

$$\begin{tikzcd}
& {} \arrow[d] & {} \arrow[d] & {} \arrow[d] & {} \arrow[d] & \\
{} \arrow[r] & ix_n \arrow[r] \arrow[d] & iy_n \arrow[r] \arrow[d] & iz_n \arrow[r] \arrow[d] & ix_{n+1} \arrow[r] \arrow[d] & {} \\
{} \arrow[r, dashed] & px_n \arrow[r, dashed] \arrow[d] & py_n \arrow[r, dashed] \arrow[d] & pz_n \arrow[r, dashed] \arrow[d] & px_{n+1} \arrow[r, dashed] \arrow[d] & {} \\
{} \arrow[r] & ox_n \arrow[r] \arrow[d] & oy_n \arrow[r] \arrow[d] & oz_n \arrow[r] \arrow[d] & ox_{n+1} \arrow[r] \arrow[d] & {} \\
{} \arrow[r] & ix_{n+1} \arrow[r] \arrow[d] & iy_{n+1} \arrow[r] \arrow[d] & iz_{n+1} \arrow[r] \arrow[d] & ix_{n+2} \arrow[r] \arrow[d] & {} \\
& {} & {} & {} & {} &
\end{tikzcd}$$

Figure 12: the sequencing of activities in figures 3 and 4

in which observable communications – those on *in* and *out* – can happen; and by *efficiency* I mean the throughput and latency, as measured by the passage of real time, and the amount of storage consumed.

A reasonable program can use no more than three T variables, and must use at least three variables to have enough internal concurrency. There are therefore three variables – call them x, y and z – and so nine distinct types of activity – for each variable an input from *in*, a processing activity and an output to *out* – which are repeated over the passage of time. I will refer to these activities as, for example ix_n for the nth input into x, px_n for the corresponding processing activity, and so on.

The inputs have to happen in a sequence, and there is no point doing anything but using the variables in rotation, which implies an ordering: ix_n before iy_n before iz_n before ix_{n+1} and so on (up to a relabelling of the variables). The same holds for outputs, which clearly have to happen in the same order. There is also a sequence implied by the history of each variable: for example ix_n before px_n before ox_n before ix_{n+1} and so on.

These orderings are shown in figure 12. Each solid vertical arrow indicates an order enforced by sharing of a variable; each solid horizontal arrow is an order enforced by the sharing of a channel. For disjointness, orderings imposed by sharing a variable or channel must be implemented by executing the corresponding processes in sequence, either in a `SEQ` or something equivalent like a `WHILE`-loop. The broken arrows are orderings which might be implemented by communication; these are not important for the argument, and indeed it would not help even if we were prepared to relax these constraints. Notice that the same activity – ix_{n+1} – appears twice in the figure to make the picture clearer: the whole picture should be imagined as being wrapped around a cylinder.

Now, consider how you might implement these constraints. The effect of the `SEQ` in figure 7 is to constrain Q to happen when and only when both P and R have terminated, and similarly that S happen only when both P and R have terminated. In a picture there would be arrows from each of P and R to Q, and in general a process pointed at by some arrows must appear at the beginning of a `PAR` which follows immediately in a `SEQ` after a `PAR` which ends with the processes from which the arrows start.

Consider iy_{n+1} in the figure: it must come just after the end of a two branch `PAR` in which the branches finish with ix_{n+1} and oy_n. Similarly, oz_n must come just after the end of a two branch `PAR` in which the branches finish with oy_n and pz_n. But this means that oy_n must be the last process in two different `PAR` constructs in the program, which is not possible.

If there were a checkably disjoint program with the behaviour described by the graph sketched in figure 12, it would have to be possible to implement by `SEQ` ordering the constraints shown as solid lines. Assuming that such a program exists has led to a contradiction, so we conclude that there is no such program. (Exercise: show that the equivalence of the programs in figures 8 and 9 does not contradict this argument because the sequencing graph corresponding to figure 8 is a special case.)

A question of philosophy

There seems to be a growing body of opinion that the usage checker is an optional extra to occam and that whenever the programmer convinces himself that the restrictions are too confining but that his dubious program will work he should turn it off and still claim his program to be occam. This is, in some people's opinion, a similar level of 'language abuse' to that of the Fortran programmer who discovers that he can pass a routine as an argument to itself, and so introduce recursion. It can be argued that the occam definition is too restrictive and should not have specified, as it does[9, p. 75]:

> An array may be used in more than one component of a parallel, if and only if the subscripts used to select components of the array can be determined at compile time. Otherwise the array may only be used in one component of the parallel.

but until a revised constraint allowing justifiable exceptions to the rule is promulgated, anything which does not satisfy the Fourth Law of Parallel Usage is presumably not occam.

References

[5] Roger Peel, *Issues Raised while Implementing a Layered Protocol Using occam and the Transputer*, in ed. André Bakkers, OUG-10 *Applying Transputer Based Parallel Machines*, IOS, 1989.

[6] S. D. Brookes, C. A. R. Hoare, A. W. Roscoe, *A theory of communicating sequential processes*, Journal of the ACM, **31**(3), July 1984, pp. 560–599.

[7] Phil Atkin, *Performance maximisation*, INMOS Technical Note 17, INMOS document 72-TCH-017-00; also the last chapter of [8].

[8] INMOS Limited, *Transputer Technical Notes*, Prentice Hall, 1989.

[9] INMOS Limited, *occam 2 Reference Manual*, Prentice Hall, 1988.

Acknowledgement

This note started life in a conversation in the back of the lecture theatre at the Enschede meeting, and contains material contributed to a subsequent electronic mail conversation. There have been contributions from – amongst others – Roger Peel, Michael Goldsmith, Roger Shepherd, John Wexler, Phil Winder, Philip Ross, Tony Lomax, Mark Edmonds, Adrian Lawrence and Steven Beard; but since I have not given them an opportunity to correct me, I lay a personal claim to my mistakes.

Geraint Jones Tel: +44 865 273851
Programming Research Group geraint@uk.ac.oxford.prg
11 Keble Road
Oxford OX1 3QD
United Kingdom

GRAPHICAL ENVIRONMENT FOR OCCAM PROGRAMMING

Fabrice Mourlin, Laboratoire de Recherce en Informatique,
University of Orsay, France

Concurrent programming has become important in recent years, but the debugging of these applications has always led to complex problems. The most time-consuming part is, of course, finding errors, especially when the application is written in a language with parallel operations. This introduces another dimension of complexity to the already difficult task of debugging, for instance:

- ▷ communications between processes through methods such as *rendezvous* or the use of shared global variables,
- ▷ non-deterministic control of some processes and especially problems with *deadlock* and *starvation.*

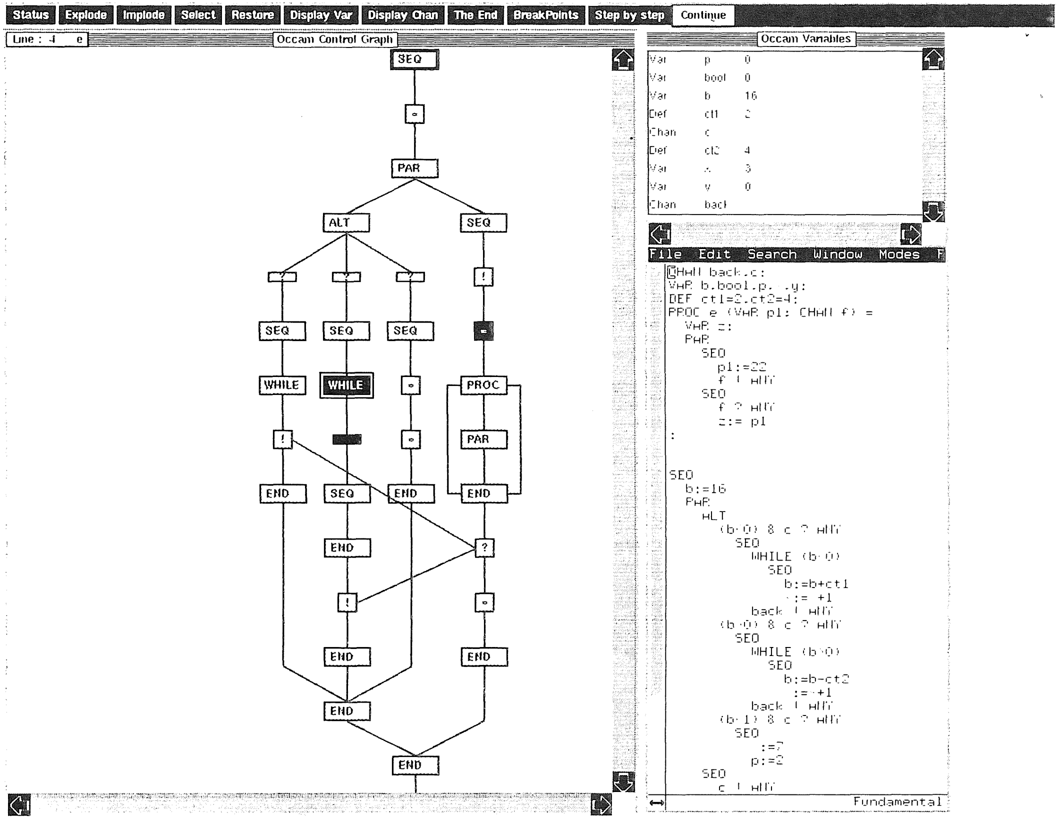
In the light of such problems, using a graphic tool adapted to a language like occam is a valuable weapon for the users. Our approach is based on observation of the occam program. This technique stresses the structure of the algorithm, e.g. the synchronization, and improves the understanding of it.

Once an occam program has been created – like that in figure 13 – it is necessary to analyse it. There are two complimentary stages. The first one is a static analysis: a control graph is first built (see the left window of figure 13); this graph represents the occam program in the form of block diagrams. Each node corresponds to an occam block which can be exploded to see its contents. The user can examine the structure of his program, the possible communications, the scope of the variables, the status of instruction in source program, etc. Afterwards, through animation, this graph can be used to synthesize program execution.

The second stage of the debugging makes visible an execution, or one of the possible executions. In this way, the states of all or part of the variables can be analysed no matter what the program level is. Moreover, the user can be confident that this animation represents an exact history of all the states.

Animation is implemented through close collaboration between a scheduler and the graphic interface. These two modules form the basis of the dynamic part. After the complete compilation of the occam program, the scheduler executes a part of

Graphical Environment for Occam Programming



Var	p	0
Var	bool	0
Var	b	16
Def	ct1	2
Chan	c	
Def	ct2	4
Var	x	3
Var	y	0
Chan	back	

```
CHAN back.c:
VAR b.bool.p. .y:
DEF ct1=2.ct2=4:
PROC e (VAR p1: CHAN f) =
  VAR z:
  PAR
    SEQ
      p1:=22
      f ! ANY
    SEQ
      f ? ANY
      z:= p1
:


SEQ
  b:=16
  PAR
    ALT
      (b·0) & c ? ANY
        SEQ
          WHILE (b·0)
            SEQ
              b:=b+ct1
              :=  +1
          back ! ANY
      (b·0) & c ? ANY
        SEQ
          WHILE (b·0)
            SEQ
              b:=b-ct2
               := +1
          back ! ANY
      (b·1) & c ? ANY
        SEQ
          :=7
          p:=2
    SEQ
      c ! ANY
```

Figure 13

the compiled code and transmits the new state to the graphic module. This graphic interface represents the new state (changes of parameters, allocated lines, etc.). Once the active point (or points) has been marked on the graph, that part of the graph is displayed in reverse video. This mechanism continues until the end of the user's program is reached.

Our purpose is to simulate the operation of a network of processors that can share time among several processes, each of which executes a number of instructions. The memory of the entire system must be divided among all existing parallel processes. Each part contains the local variables of its process, and links with the surrounding blocks. This backup structure was designed in the form of an evaluation stack in which each modification of a variable by a process must be propagated to its definition, even if it belongs to a surrounding block.

To start an execution, the user selects the command *continue.* Then, the program is executed and stops when it encounters a breakpoint; the user can then resume the static analysis. The operation will continue if the *continue* command is selected again. A single-step mechanism can be used instead of normal execution.

During execution, the channels are displayed in two different ways:

▷ in the variables window, in which their condition is indicated;

▷ on the control graph.

The channels are integrated into the control graph and thus acknowledged as soon as the corresponding graph is constructed in the occam source program. This justifies building them during the static phase and simply highlighting them during execution (see figure 13). If, when displaying a channel, the sender node (respectively, receiver) communicating with the receiver node (respectively, sender) is not visible, the channel will be displayed at the first visible parent construct node.

If during execution the program falls into *deadlock*, the executable processes will have stationary active points. That characteristic indicates the presence of errors which were not detected in the static phase.

The user can also select a specific execution: he can set parameters, such as the number of instructions per process, to obtain a particular execution of his program; he can fix the order in an ALT statement to check all cases.

The present version of the system is running under the X Window system, versions 10 and 11, and is written in C. The visible interface is shadowed by an intermediate layer that formats objects to be manipulated by the interface in graphic form. The tools are UFO (*User Friendly Objects*) which allows the construction and management of complex objects in graphic form, and GLIB (*Graphic Library*) which includes graphic functions for window management, external events and display of graphic objects. These two graphic tools are developed by M. Beaudouin-Lafon in LRI.

The system demonstrates that a graphical environment is particularly valuable to parallel programming to help understand how a program really works. The structure of the occam language is also well suited to a graphical representation of programs.

We plan to upgrade the system to run with occam2, incorporating all the improvements that have been made, such as the use of functions, multiple assignments, data types and so on. Improvements are also planned to the graphical interface: it is necessary to manage several windows each containing part of the control graph, to be able to study large programs.

OPERATING TRANSPUTERS AT LOW TEMPERATURES

D. A. Nicole and M. P. Moore,
Department of Electronics and Computer Science,
Southampton University

This item appeared as 'Southampton Transputer Support Centre Technical Note N1', and was the origin of some controlled mirth at the ninth occam user group technical meeting in Southampton where it was discussed by the Hardware SIG.

Background

We have recently performed some experiments on a direct way to provide faster transputer systems. The performance of CMOS circuitry improves dramatically when it is cooled. The dominant effect[10] is an increase in carrier mobility due to decreased phonon scattering. (Mobility obeys a $T^{-3/2}$ rule over the temperature range of interest.) There are also improvements in the electrical conductivity of the metal interconnect and in the thermal conductivity of the silicon substrate. Furthermore, parasitic bipolar transistor effects disappear at low temperatures, leading to improved switching characteristics (higher sub-threshold slope) and resistance to latch-up. Overall, an improvement in performance by a factor of about 1·5 to 3 can be expected[11, 12] as we cool a circuit from 300 K to 77 K, the temperature of liquid nitrogen. Liquid nitrogen immersion is, in fact, a feature of the American *ETA^{10}* high speed 'vector' processor.

Technique

For those with access to liquid nitrogen, these experiments are very easy to perform. A simple circuit was constructed using standard wire-wrap techniques, and using a reduced board size so as to fit the neck of a one litre Dewar. The circuit was then simply immersed in the Dewar full of liquid nitrogen. The rate of boiling is such as to permit operation for several hours from one litre of nitrogen.

For some of the measurements, a simple 'gearbox' consisting of two INMOS link adaptors joined by their parallel ports was used to provide variable speed link signals which tracked the varying clock rate supplied to the transputers. No attempt was made to vary the power supply voltage away from the standard five volts.

Transputers

As described above, it is both interesting and easy to chill transputers and investigate the changes in their performance. Nevertheless, this is a rather risky procedure. First, the packaging and bonding is not designed to tolerate sudden cooling. Second, the transputer has on-chip phase locked loop circuits. Attempts to clock the transputer at higher than normal speeds would result in these operating well outside their designed frequency range.

In view of the possible trouble with the phase locked loops, it is interesting to examine how their performance tracks on cooling. For a typical component, the maximum operating frequency of the processor clock phase locked loop at room temperature was 39 MHz. At 77 K it operated up to 92 MHz. We are thus extremely fortunate – the phase locked loop operating frequency tracks with the expected processor performance. Direct tests show the maximum operating frequency of a sample T414 transputer rising from 27 MHz at 300 K to 54 MHz at 77 K whereas that for a T800 rises from 27 MHz to 46 MHz. We have successfully operated a network of T800s in liquid nitrogen with their processors running at 45 MHz and their links at 40 MHz.

Overall, chilling seems to provide the possibility of a cost-effective doubling in performance for future machines based on CMOS technology.

We have subjected only a small number of transputers to this extreme cooling; we are rather concerned that the standard packaging will not tolerate repeated cycling to 77 K. Similarly, we have as yet made no use of the external memory interface.

References

[10] S. M. Sze, *Physics of Semiconductor Devices, 2nd edition*, John Wiley and Sons, ISBN 0-471-05661-8, pp. 28, 453.

[11] J. D. Plummer, *Low temperature CMOS devices and technology*, IEEE Proc. IEDM86 (1986).

[12] R. W. Keyes, E. P. Harris and K. L. Konnerth, *The rôle of low temperature in the operation of logic*, Proc. IEEE **58** (1970) 1914.

D. A. Nicole and M. P. Moore
Department of Electronics and Computer Science
The University
Highfield
Southampton S09 5NH

TRANSPUTER TIMING

Krste Asanovic, GEC Hirst Research Centre, Wembley

This note was written in reply to an observation by Erick Bizouarn of LRI, Université d'Orsay, France – that adding redundant instructions to a program can make transputers run them faster. Briefly, the original question was over execution times for a loop coded as follows:

```
        ldc    10000
        stl    2
.align  /* Put L0 on a word boundary. */
L0:
        /* Insert variable number of "ldc 0" instructions. */
        ldl    2
        adc    -1
        stl    2
        ldl    2
        eqc    0
        cj     L0
```

The number of 'ldc 0' instructions is varied and the total execution time for the run is measured. Below are Erick's original timings (perhaps for a Parsytec card?) together with some I gathered on the Tadpole card in our Sun. The execution time *decreases* with added 'ldc 0' instructions!

Number of ldc 0s	Erick's timings in μsec.	Tadpole card
0	16382	20352
1	15422	19328
2	14398	17856
3	15870	17856
4	18430	22848
5		21888

As Erick suggested, this occurs because of the operation of the instruction prefetch buffer. Below, I have attempted a more detailed analysis based on my own understanding of the transputer's architecture but would appreciate any more precise information.

Instructions are pre-fetched in parallel with program execution. On a given cycle, the CPU is taking instructions from one four-byte buffer while the pre-fetch may be filling another. When the CPU takes the last instruction from its buffer, the buffers are swapped and another instruction pre-fetch cycle is started. The transputer has only one memory space (i.e. non-Harvard architecture) and so instruction pre-fetches compete with data accesses for the memory bus.

When a jump occurs, an instruction fetch must occur before the CPU can begin executing code at the new location. However, a second pre-fetch cycle is started immediately to fill the pre-fetch buffer. If the user code contains instructions which access memory (ldl, stl, etc.) then these will be delayed until after the second pre-fetch cycle completes.

If the 'cj' instruction is the last byte in a word, then a new pre-fetch is started even if the jump is taken. This pre-fetch will be for the instructions five bytes after the 'cj' byte and must be completed before the instructions at the jump destination can be fetched. The 'cj' instruction takes four cycles if taken and these can overlap with the useless pre-fetch, so this end of the pipeline break is not too disastrous.

I have drawn some diagrams which show the interaction between the pre-fetch's use of the memory bus and the program's data accesses, and these illustrate how extra instructions can cause faster execution of the loop. The diagrams are for the Tadpole card which has a 20 MHz T414 with 250 ns DRAM cycles. Each horizontal step represents one clock cycle of 50 ns. Instructions codes are written vertically and are placed on the cycle on which they are read from the pre-fetch buffer. The periods labelled *execution* show when the transputer is running program code. The periods labelled *data access* and *pre-fetch* indicate the times for which the memory bus is occupied in fetching the program's data, or in reloading the pre-fetch buffer, respectively.

Figure 14, with no 'ldc' instructions, takes 42 cycles but figure 15, with three 'ldc' instructions, takes 38 cycles. These diagrams are not entirely accurate: I believe greater overlap occurs than these diagrams suggest. I do not have the time to set up a logic analyser, and INMOS keep quiet about the transputer micro-architecture.

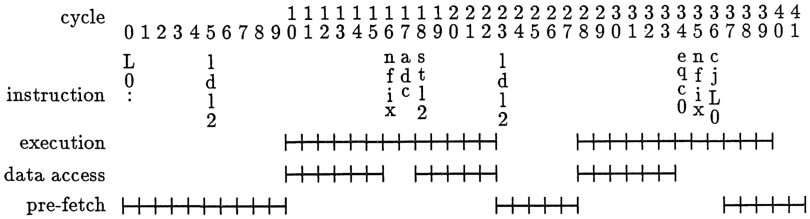


Figure 14: loop timing with no `ldc 0` instructions

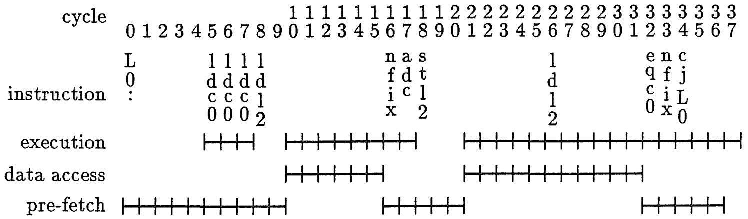


Figure 15: loop timing with three `ldc 0` instructions

However, they are within a cycle or two of the measured figures.

Conclusions

In general, the '`ldc`' instructions can be replaced with any instruction which does not access external memory. These will then be executed in parallel with the second instruction pre-fetch. A compiler could re-arrange code to take advantage of this, by placing code which does not reference memory at the head of word-aligned looping constructs. Also, within a section of code it is theoretically possible to optimise code (perhaps by adding '`pfix 0`' no-ops) so that instruction pre-fetches occur when non-memory instructions are executed, but this is very tricky and linked to a given hardware implementation's memory cycle.

Unnecessary pre-fetches at the tail of a loop are avoided if the '`cj`' instruction is not the last byte of a word. This might also be tricky to arrange and doesn't gain you much given that '`cj`' takes four cycles in any case.

The second execution trace shows how well the transputer can saturate its memory bus, but also shows how tightly its performance is coupled to the speed of external memory. How do you fit a Lisp system into internal RAM? Expect to see caches on future transputers!

Krste Asanovic
GEC Hirst Research Centre
East Lane
Wembley
Middlesex HA9 7PP

Tel: +44 1 908 9662
Fax: +44 1 904 7582
Telex: 923429 GECLAB G
Email: krste@uk.co.gec-rl-hrc

REVIEW

CONCURRENT PROGRAMMING IN OCCAM2

John Wexler, pub. Ellis Horwood, 1989,
pp. 208, hb £29·95, ISBN 0 7458 0394 6

In my view *Concurrent Programming in occam 2* is one of the better, if not the best, introductory level texts on the occam 2 programming language that I have read.

The book may be broadly divided into four parts; an introduction discussing the rationale for concurrency, a tutorial-style description of the occam 2 language, a discussion of concurrent programming in occam 2, and a small collection of worked examples. Additionally, a list of suggestions for further reading is included as an appendix.

Of these four parts the first is by far the worst. Normally, by reading the introduction to a book you can obtain a good impression of the author's style, and a feel for how you will react to the remainder of the text. In this case, the opposite is true, the introduction appears 'tacked' on to the front, and whilst providing an overview of SIMD versus MIMD in less than a page is possible – as the author demonstrates – I would have preferred either a more thorough treatment of this topic or its total omission. It should be noted however that the introduction accounts for only four pages and is therefore relatively insignificant.

The second part in comparison is probably the best section of the book. It comprises around 100 pages of the 208 page book and alone would make *Concurrent programming in occam 2* a worthwhile purchase for most people attempting to learn occam 2. The author assumes that the reader has some experience of programming in a traditional sequential language, and draws comparisons with Fortran and Pascal where appropriate.

If any criticisms of this section were to be made they would be that the author whilst attempting very successfully to write without reference to any particular development environment consistently uses the notation:

```
.. This is some code not written in full
```

whilst not referring to the folding editor until page 151. Whilst the author claims that the book is not written for any specific development environment, the Transputer Development System is mentioned briefly. This is however not unreasonable, since as the author himself states, the majority of development environments are either TDS or TDS derived.

Furthermore, the author too frequently for my liking introduces topics without a full description, instead referring the reader further forward into the text to where the topic is discussed more fully. It is obviously a case of the personal tastes of the reader deciding whether this sort of approach is appreciated or not, however I found it distracting. Overall, except for the two very minor criticisms I have made, I found this section excellent.

The third section of the book discusses the practicalities of concurrent programming in occam 2, together with issues relating to programming transputers and networks of transputers in particular. This section of the book is written at an introductory level and as such is very readable and quite informative. It covers most topics of importance – for example, non-determinancy, deadlock, PRI PAR/ALT, network topologies and load balancing – but in many cases relatively superficially and not in as much depth as I would have liked. Additionally, it does not cover how you actually specify the distribution of processes to processors in a network, although as the author states this was omitted deliberately due to it being implementation specific.

The final section of the book consists of seven useful but relatively small occam 2 programs, together with a discussion of their purposes and how these are achieved. This section again is rather short and could well benefit from being expanded, although as it stands it is more than useful. A few more sizeable examples might have improved it considerably however.

Generally, John Wexler has written a book which provides an excellent introduction to occam 2, and as such I will certainly recommend this book to my students. Overall, the impression I received was that the book is too short, which is not necessarily bad for an introductory level text. *Concurrent Programming in occam 2* attempts to cover a lot of ground and does so successfully although superficially in some areas.

For people attempting to learn occam 2 for the first time – as the author himself comments – information on the specific development system being used will be required. Additionally, although the author states that the formal definition of the occam 2 language has been omitted on the grounds of cost, I personally find this a shortcoming. If the language definition had been included then I would have been tempted to recommend this text as an alternative to Pountain and May.

Considering the book solely as an introduction to the occam 2 language, (to which over half of its content is devoted), I certainly have not yet read any text which I would consider superior. *Ian Johnson, Transputer Centre, Bristol Polytechnic*

PRODUCTS, SERVICES AND ANNOUNCEMENTS

INMOS MARKETING UPDATE

Mark Jones, INMOS Bristol

Since the last newsletter, the major event has been the formal announcement that SGS-Thomson are the new owners of INMOS, creating a broad-based semiconductor supplier with a combined turnover of over $1 billion.

SGS-Thomson have committed to heavy new investment for INMOS and the transputer family with the goal of firmly establishing the transputer as an industry standard and, for example, we are now recruiting over a hundred new people.

New manufacturing plant

Important news for customers is that the transputer will have a second silicon fabrication plant overcoming the issues of being a single sourced product.

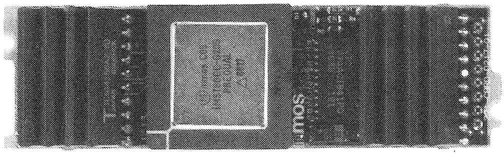
We will now be producing transputers at the SGS-Thomson factory at Carolltown, Texas, USA in addition to the Newport factory.

New products

T425 Enhanced T414 with 4K SRAM, faster links and improved instruction set.
The T425 is in production at 17, 20 and 25 MHz in an 84 pin PGA package.
We are also sampling the 17 and 20 MHz versions in a low cost PLCC surface mount package targetted at the volume market.

T801 This is a T800 with a non-multiplexed external memory interface giving 2 cycle access to external memory. It is produced in a 100 pin PGA package in two speed versions and 20 and 25 MHz. The 25 MHz parts support a data rate to external memory of 50 MByte/sec.

T801 TRAM For users who want to evaluate the T801 quickly, a size 2 TRAM with 110 KByte is available now from our distributors (Part number B410-11, illustrated on the front cover).



INMOS B411 TRAM module

Other new TRAM products include:

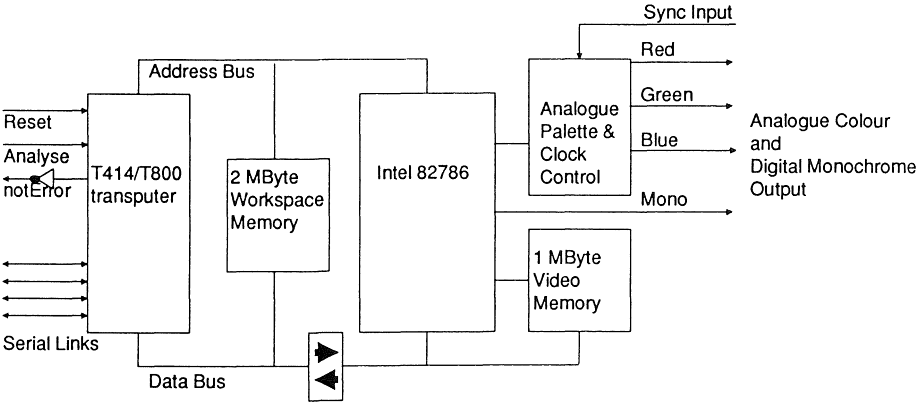
- ▹ B411-3: T800-20, 1 MByte DRAM, size 1,
- ▹ B411-7: T425-20, 1 MByte DRAM, size 1,
- ▹ B401-5: T800-25, 32 KByte SRAM, size 1,
- ▹ B401-8: T425-25, 32 KByte SRAM, 2 MByte DRAM, size 2,
- ▹ B404-5: T800-25, 32 KByte SRAM, 2 MByte DRAM, size 2,
- ▹ B404-8: T425-25, 32 KByte SRAM, 1 MByte DRAM, size 2.

Software The new parallel compilers incorporating the INMOS `iserver` are now available. These compilers are compatible with the forthcoming toolset products due for release over the next few months.
Available now:

- ▹ Parallel C for PC, SUN, VAX,
- ▹ Parallel FORTRAN for the PC.

TEKTITE XTRAM GRAPHICS HARDWARE MODULE

Tektite Ltd, a British company marketing transputer-based graphics products, announces the Xtram – an ultra-compact transputer module which incorporates the processing, drawing, and display functionality of a traditional workstation into a 9·3 × 11·0 cm, 16-pin package (a size 4 TRAM). The Xtram has been tailored to suit the requirements of windowing interfaces such as the MIT X Window System.



XTRAM architecture

The Xtram contains a 10 Mip, 1·5 MFlop INMOS T800-20 transputer provided with 2 MByte of main memory, a 20 MHz Intel 82786 drawing and display processor with 1 MByte of display memory, a palette allowing the display of 256 colours from a selection of over 16 million, and a fully programmable timing generator with integrated genlock capabilities.

The Intel 82786 drawing and display processor supports hardware windowing, hardware BitBlt, and hardware drawing of lines, arcs, circles, rectangles, polygons, polylines, points, and patterns.

1 MByte of graphics memory allows the storage and manipulation of images up to 4000 × 2000 pixels at 1 bit per pixel, or 1000 × 1000 pixels at 8 bits per pixel. It also allows the application to store and use multiple fonts, sprites, and icons.

The programmable timing generator operates with all common analogue monitor systems, including Multiscanning types, allowing many users to upgrade their graphics without a major investment in a new monitor.

For more information, contact

Andrew Talbot
Tektite Ltd
PO Box 5
Felixstowe
England IP11 7LW

Tel: +44 394 672117
Telex: 987458 TKTITE G

ADVANCED 3D MEDICAL GRAPHICS DISPLAY WORKSTATION PROJECT

Wolfson Transputer Application Group, MGI Section
Department of Medical Physics & Bioengineering
University College London

The project is inviting collaboration with medical imaging equipment manufacturers, and with academic and research collaborators.

Background

The Wolfson Transputer Application Group has been involved in the development of a transputer based workstation for 3D medical graphics applications for several years. The system aims to perform the tasks of standard image management, scientific visualisation of medical data, surgical simulation, clinical diagnosis and radiotherapy planning. Techniques include both surface and volumetric rendering, active ray tracing and Monte Carlo simulation. The group also has extensive experience in handling data from a variety of medical imaging system. These include CT, MRI, digital ultrasound and laser surface scans.

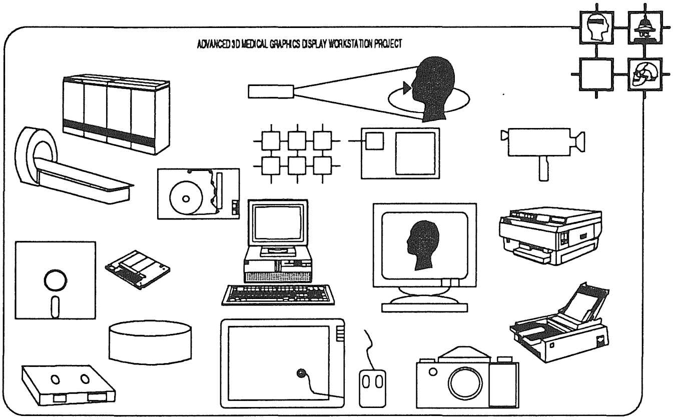
We are now at an advanced stage of development where working prototypes have been demonstrated and the system has been purchased by hospitals on a direct basis. Considerable support is expected to come from the health service and the commercial possibilities are good in this sector.

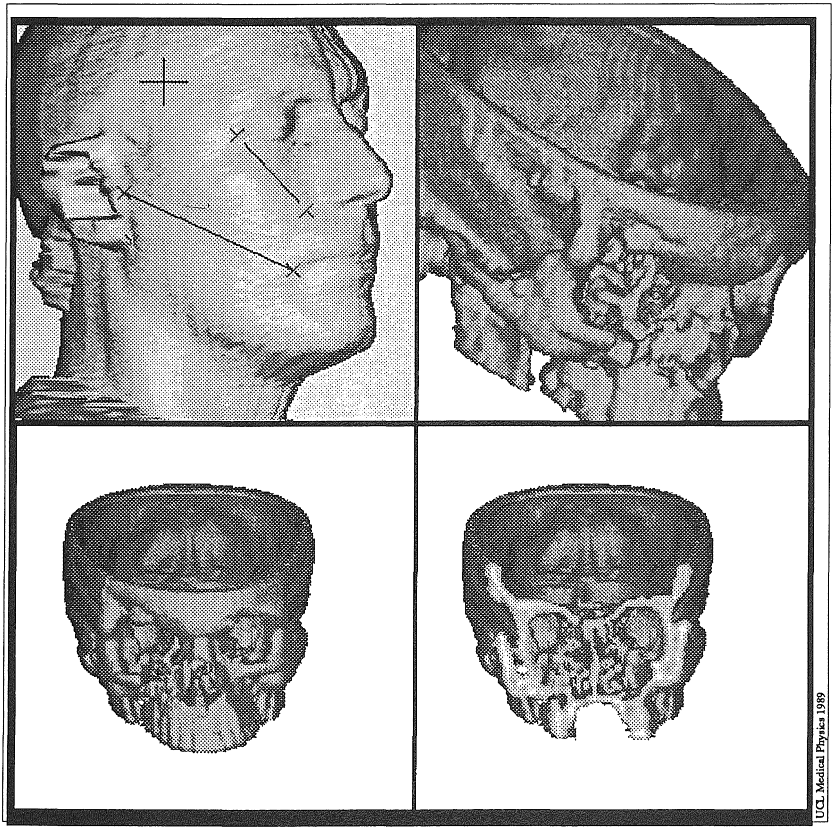
Technical specification

HOST COMPUTER The advanced 3D medical graphics display workstation is based on an industry standard PC-AT compatible. An optional half inch tape subsystem, intended for loading CT and MRI data can be provided as an additional desktop unit. Interaction using an extended keyboard with separate cursor keys comes as standard. Digitiser tablet and mouse interfaces are currently in preparation and will be available in future releases.

TRANSPUTER IMAGE GENERATION ACCELERATOR High speed calculations and image generation are handled by a transputer based coprocessor. This consists of a 20 MHz T800 transputer operating up to 10 Mips and 1·5 MFlops. The standard configuration has 4 Mbyte of RAM installed, with the option of upgrading to 8 Mbyte when required. Future upgrade options include extra transputer modules in the combination of one, two or four processors. A transputer based video card is also installed to provide high speed rendering for pixel intensive graphics. The operating system supported on this workstation is DOS which is the industry standard environment for many PC based application packages.

LASER SCANNER A laser scanner for acquiring accurate surface measurements of the face can be provided as an option with the workstation for 3D display and analysis of line profile data. The interface to this unit is housed in the same workstation case. The unit consists of a rotating chair and a fixed floor tower. It occupies a floor space of 6 feet by 8 feet.


ADVANCED 3D MEDICAL GRAPHICS DISPLAY WORKSTATION PROJECT


UCL Medical Physics 1989

Software specification

LASER SCANNER Both single and multiple laser line profiles are supported. Calibration software is provided to maintain accurate operation. Display of single or multiple views of the laser data in the form of line profiles, wire frames and fully rendered surfaces is available. Facilities for measurement have been implemented for clinical analysis of the data. Most of the viewing functions such as rotation, zooming and altering light conditions can be operated at less than ten seconds per interaction.

CT SCAN & MRI DATA Special tape reading software has been written to allow users of major CT or MRI installations to transfer data to the workstation via the half inch tape drive connected to the system. Standard functions for working with the data include 2D slice display, highlighting and windowing functions, and some image processing facilities. For volumetric rendering of data sets, active ray tracing techniques are used to present quality 3D images for diagnostic purposes. Both single and multiple views are supported, each view is independently rotated and magnified. A cut plane can be moved along the viewing direction to expose internal details of the anatomy. A limited set of surgical simulations and 3D data measurement facilities are also provided as supportive research tools on the workstation.

For further information, please contact:

Mr A. C. Tan
Wolfson Transputer Application Group
MGI Section
Department of Medical Physics & Bioengineering
University College London
1st Floor, Shropshire House
11–20 Capper Street
London WC1E 6JA

Tel: +44 1 380 9700
Fax: +44 1 380 9577
actan@uk.ac.ucl.sm.mph

CAMBRIDGE INTERNATIONAL SERIES ON PARALLEL COMPUTATION

a series of advanced textbooks and research monographs
published by Cambridge University Press
Cambridge, New York, New Rochelle, Melbourne, Sydney

The purpose of this new series is to publish advanced textbooks and research monographs on parallel computation. It will be complementary to the recently established series 'Cambridge Tracts in Theoretical Computer Science'. The aim of the editors will be to develop a series of books which will have a continuing relevance as the field of parallel computation matures. All aspects of the subject will be covered. The following list of topics is therefore not meant to be exhaustive, but merely to indicate the breadth of the series.

▷ *Parallel Algorithms and Architectures*: Design and analysis of parallel algorithms, complexity of parallel computations, parallel numerical algorithms, VLSI algorithms and architectures, systolic algorithms, cellular automata, connectionist architectures and neural networks.

- *Parallel Program Design*: Specification and verification of communicating systems, protocols, deadlock, fault tolerance.
- *General Purpose Parallel Systems*: Interconnection networks and routing methods, mapping of parallel algorithms, data parallelism, parallel implementation of functional and logic programming languages, operating systems and compilers for parallel architectures.
- *Advanced Applications*: Scientific computing, graphics, image processing, vision, robotics, CAD, database searching, combinatorial optimization, symbolic computation, AI.

Managing Editor

Editorial Board

Publisher

Cambridge University Press, founded in 1534, is a major international academic publisher, with branches in Britain, the USA and Australia, and offices elsewhere in the world. It is one of the world's leading publishers in mathematics, science and engineering, with an extensive list of major titles in these areas. The Press is part of the University of Cambridge and as such is in close liaison with the academic community, publishing works of lasting value to education and research. Further details on the publisher can be obtained from

David Tranah dt105@uk.ac.cam.phx
Cambridge University Press
The Edinburgh Building
Shaftesbury Road
Cambridge CB2 2RU
United Kingdom

Submission of Proposals

Book proposals should contain a general outline of the material, a list of chapters, and a sample section. Authors should also give some indication of the likely length of the book. Proposals should be sent to the Managing Editor or any other member of the Editorial Board.

T2 SYSTEMS LIMITED LAUNCH TRAM ROUTE TO DISC DATA

T2 Systems Ltd recently launched the Paradise-1 TRAM, which offers the user an easy interface into the popular SCSI bus used by all optical disc systems and most Winchester drives.

The size 4 low profile TRAM offers 800 Kbyte/sec sustained data transfer (disc to INMOS serial link), firmware support for initiator or target rôle, dip-switch selectable ID, and an easy to program non-variant link protocol. The firmware provides the option of auto-booting a transputer network on reset from a disc drive.

Incorporating this TRAM into a transputer network can, at a stroke, eliminate all the frustrating delays associated with data storage in the VAX or PC environment. A system's data storage capacity can grow with the system by adding more Paradise-1/Disc systems. The Paradise-1 is manufactured by T2 Systems Ltd who also offer custom disc interface systems. For more information contact the Managing Director:

Patric Pope Tel: +44 703 641276
T2 Systems Ltd
62 Longmead Avenue
Bishopstoke
Eastleigh
Hants SO5 6ET

PLANTAGE

a message passing system for farms
Karl Junk, Karlsruhe

Lots of transputer farms are operated all over the world.
Here comes the software that turns a transputer farm into a plantation:
PLANTAGE
efficient – easy to use – flexible
The specialized message passing system for programs that work according to the farm principle – may be linked to existing or new programs.

Features:

- The object program accepts every possible configuration of transputer networks without knowing about it before it is started.
- A minimum path length from the managing transputer to each worker is achieved by using flood-filling through the existing link connections.
- Messages are only copied via links, not via internal channels. (The C implementation does not use internal channels at all for the communication between the message passing processes.)
- All link channels used by Plantage are operated simultaneously, if required.
- Message passing and result passing are completely independent of each other.
- The manager may send messages that reach all transputers (broadcasts).
- A working process may or may not run on the managing transputer.
- Large messages and results may be split and recombined at the destination.
- No limitation on message buffer sizes and number of buffers; everything is configurable, with non-occam versions at runtime,
- Easy-to-use but powerful application program interface.
- Of course, no deadlocks ever!

Why should I use Plantage and not my traditional farm software?

1. With only a few transputers or any configuration where the message passing overhead is less than 1%, gain of performance is not significant, but the Plantage standard software makes life easier when implementing the next application.
2. For every farm application there is a maximum number of worker transputers that may be covered by the managing transputer. Efficient message passing software like Plantage can significantly increase this maximum.

Transputer plantations need Plantage!

Plantage versions:

- Type Scotch for C, Fortran and Pascal compilers of 3L
- Type Bristol for occam toolsets and TDS
- Type Oregon for C compiler of Logical Systems

Production, Information and Distribution:

Karl Junk   +49 721 699552
Kaiserstraße 66
D-7500 Karlsruhe
West Germany

Plantage was originally developed on a PROTEUS machine.
3L, occam, TDS and PROTEUS are (registered) trade marks.

TROS – A REAL TIME KERNEL FOR A FAULT TOLERANT MULTI-PROCESSING SYSTEM

Eric Verhulst, Intelligent Systems International

TROS is an operating system kernel with built-in fault tolerance for transputers. The system is able to recover from transient as well as permanent failures of the communication links or of the memory/processor pair. Software errors are trapped as well. TROS is currently aimed at the real time systems market where transputers are used as embedded components and where high reliability is desirable. A remarkable property of TROS is that the degree of fault tolerance can be tuned.

TROS is topology independent by the use of argument flow, a pragmatic implementation of coarse grain dataflow developed by ISI in collaboration with the University of Leuven. In argument flow the size of the procedures can be adjusted according to the user's needs.

The current system makes exclusive use of occam, resulting in a small kernel size (30 kbyte) and exploiting the security and efficiency of occam.

A pre-release version is available for a limited number of interested users. It consists of a number of pre-compiled occam procedures to be linked with the user's application processes under TDS. A manual detailing the usage by way of a working example is provided as well. This example demonstrates how in a small network of four transputers, up to 50% of the hardware might be failing without the user application being interrupted.

In the near future, TROS will gradually be expanded to support alien languages, dynamic load-balancing, and multiple users on a single node.

For more information, contact:

ISI
Zavelstraat 142
3200 Kessel-Lo
Belgium

Tel: +32 16 25 95 86
Fax: +32 16 20 77 10

OCCAM 2 AND TRANSPUTER ENGINEERING

Computing Laboratory, University of Kent at Canterbury

Course Objectives: To acquire technical knowledge, insight and practical experience of parallel system design using *occam* and *transputer* networks.

Further Details: Harnessing the potential processing power of *transputer* networks requires the development of a fluency in parallel systems design equal to our traditional skills for sequential logic. *Occam* is a simple, small but powerful language which enables such fluency. Software engineering principles, load-balancing techniques, real-time applications and various embedded and super-computing issues will be covered. The strengths, weaknesses and likely future developments of *occam* and *transputer* technologies will be discussed.

Course Members: If you have picked up basic *occam* syntax and semantics and are wondering how to use it to engineer high-performance high-security systems, this course is for you. If you have never seen any *occam* before, so much the better! Hardware engineers are especially welcome. *C* programmers beware – this course will change your life!! *[Since September 1986, this course has attracted over 170 participants from Industry and Academia worldwide.]*

Course Methods: Informal lectures with a large proportion of 'hands-on' experience being provided through practical exercises and a 'mini-project'. Practical work will be on the MEiKO Computing Surface and will be supervised at the ratio of one tutor for every six attendees. The MEiKO provides a multi-user multi-transputer development and applications environment. Our system will support up to 20 simultaneous users, each with dedicated access to a private network of transputers including at least two T800s. The full system comprises over 80 transputers (including 56 T800s) with a gigabyte distributed file store and three high resolution graphics workstations.

Length & Cost: Five days & £450.

Dates: Course Nº 12: 10–14 July 1989
Course Nº 13: 4–8 September 1989.

Contact: For a full syllabus, application forms, fees, special arrangements and accommodation, please contact:-

Dr P. H. Welch
Computing Laboratory
The University
Canterbury
Kent CT2 7NF

Tel: +44 227 764000 x7695
Fax: +44 227 762811
Telex: 965449 UKCLIB
Email: phw@uk.ac.ukc

EEC Recognition: This course is one of the foundations for a series of courses and technical workshops entitled '*Training for Transputer Technologies*'. This is being developed under contract with the EEC as part of the Communities Action Programme for Education and Training for Technology (COMETT).

occam is a trade mark of the INMOS Group of Companies;
MEiKO and the Computing Surface are trade marks of Meiko Limited.

TRANSPUTER PROGRAM DEVELOPMENT ON THE ACORN ARCHIMEDES

Gnome Computers Ltd

A complete, low-cost, transputer development environment for the Acorn Archimedes workstation has been released by Gnome Computers Ltd. It is based on two add-on boards, a Link Adapter Board and a TRAM Motherboard, both designed by Gnome Computers.

The Link Adapter Board allows the Archimedes workstation to communicate with external transputer systems using the INMOS standard link protocol at speeds of 10 MHz or 20 MHz. The TRAM Motherboard provides similar facilities and also houses up to four transputer modules (TRAMs). The Archimedes workstation can contain up to four such boards allowing a maximum of sixteen transputers to be housed within the workstation. The motherboard provides access to unused links on the TRAMs to allow connection to external transputer systems.

Software is available to allow the boards to be used to support a variety of transputer programming environments. The TDS Server program supports occam program development via the INMOS Transputer Development System (TDS). The AF Server program supports program development of other languages such as C, Pascal and Fortran. The TDS for the Archimedes is the same as that for the PC and programs may be readily transferred between the two environments.

The Link Adapter Board or TRAM Motherboard in conjunction with Gnome Computers TDS and AF servers can be used to quickly develop both occam and other high-level language programs on internal and external transputer systems.

The Acorn Archimedes workstation is an ideal platform for transputer development. Being RISC based, the Archimedes can provide the rapid disc access and screen update which is required for a productive programming environment. The overall performance when running systems such as the TDS is two to three times faster than a mid-range PC. The workstation also has the additional benefit of displaying 32 lines of text and high resolution graphics as standard.

Principal features of the environment include:

- ▹ High speed, RISC based workstation,
- ▹ TDS support for occam programs,
- ▹ AF Server support for 'conventional' high-level languages,
- ▹ Up to 16 transputers housed within the workstation,
- ▹ Connections to external transputer systems,
- ▹ Link Adapter and TRAM Motherboards conform to INMOS hardware standards.

For further information, please call Chris Stenton or Steve Temple at:

Gnome Computers Ltd Tel: +44 223 461520
16 Histon Road
Cambridge CB4 3LE

THE LIVERPOOL PARALLEL TRANSPUTER LIBRARIES

N. G. Brown
Centre for Mathematical Software Research
University of Liverpool

Libraries for Transputer Arrays

A recurrent need in scientific and engineering computing is for routines to solve standard numerical problems: solution of sets of equations, ordinary differential equation integrators, optimisation routines, etc. Most scientific and engineering sites provide collections of such routines for their users; the best known, and in Europe by far the most widely used of these, is the Nag Fortran library, now at Mark 13 and containing over 750 user-callable routines.

Now, numerical libraries are beginning to appear for transputer arrays. Several are being actively advertised; most however run only on a single transputer – and if this is what you need, the best buy must still be the Nag Fortran library, available in a single transputer version for INMOS (or 3L) Fortran. However, a genuine parallel library, with versions interfaced to both Fortran and occam, is being developed at Liverpool University; and Mark 1 of these libraries is now available, and is being marketed commercially by N. A. Software Ltd, a Liverpool University 'spin-off' firm.

The Liverpool Parallel Transputer Libraries are a collection of carefully tailored numerical algorithms, which incorporate explicit parallelism in a form suitable for computer systems with an 'MIMD local memory' architecture, such as those based on arrays of transputers. The versions being marketed are interfaced to Fortran and (for those who use it) to occam; that is, the routines are callable from a master Fortran or occam procedure. They will run on almost any transputer array, of almost any configuration; considerable care has gone into making them as portable as possible. So far, the available implementations are for machines with a PC front-end, running either MSDOS or stand-alone Fortran; a version for MEiKO single-user machines is also available. Versions interfaced to other operating systems and hosts (Sun, Helios, etc.) are under development. At the current Mark (Mark 1) all routines are in fact written in occam, for maximum efficiency; but the Fortran user need not be aware of this.

The libraries have been designed to be as easy to use as possible; the user need do little other than `#use` the library in his/her master procedure, and make a straightforward call to the appropriate routine, together with provision for dealing with errors which may be detected by the routine. The library design assumes that the user program (or that part which calls the library) resides on a single transputer, referred to as the *Master Processor*, which is linked to the array on which the library runs by a single transputer link. This assumption mirrors the most common form of setup, in which the Master Processor is an INMOS B004 board or clone, residing in a PC or other front-end. However, the Master Processor may be a mass memory board, or indeed any other transputer; the important assumption is that only one processor makes a call to the library. Thus, the library can be called directly from a serial Fortran program; existing programs which use serial numerical libraries, such as the Nag library, can be parallelised merely by replacing calls to the Nag library with calls to the Liverpool Parallel Library. If the library calls were dominating the time taken, this is indeed all that is necessary; if the library calls were only a minor part of the time, then of course parallelising them will not help much, and you will be better off calling the serial Nag library and hand-parallelising the actual bottleneck in your program.

The Liverpool Parallel Library

The library is written to make minimal assumptions about its environment; hence, it is designed to run on essentially bare transputers. Each library routine must therefore make an assumption about the way in which the array is configured, i.e. the array topology. Most routines in the Mark 1 library expect a linear array (a daisy chain). However, some routines assume either a square array or a binary tree topology. It is up to the user to make sure that the appropriate topology is available, together with information on the way in which individual processors are wired together (i.e. processor numbering and link numbering). If a given topology is not available, those routines which need it should not be called. However, on most arrays, especially those with arbitrary reconfigurability such as the Parsys or Telmat Supernode, the array can be configured so that all of the required topologies coexist, and the library uses a configuration file which can be prepared by the user (a handy Configure program is provided for this purpose) and which describes the details of the array: its size, connectivity, etc. Changing to a larger or smaller array, or re-wiring an existing array, requires only that the corresponding changes be described in the configuration file. During program execution, a library manager keeps track of the needs of individual routines, and will raise an error if the configuration is unable to meet the needs of the routine (for example, if a square grid is needed but the array is configured only as a daisy chain). The library management system consists of several parts:

1. The *library sleeper* is the name given to a general-purpose harness which is loaded onto the library array statically. Slave code for library routines is down-loaded to the array dynamically, when the routine is called; this down-loading is controlled by the library sleeper.
2. The *library manager* is a process which has knowledge of the state of the array: its detailed connections, size, a load path for the slave code, the memory available on each slave, etc. It carries out the dynamic checking needed at library calls.

The library comes with full documentation (users of the Nag library may recognise the style: not surprisingly, since Liverpool University has worked with Nag on library development for almost twenty years). Each routine is provided in single and double precision form, and each routine document contains a full description of the routine, together with an example program, showing how (one variant of) the routine being described may be called. These example programs are provided with the library in machine-readable form, and are useful as templates from which to construct valid calls of the routine. The Mark 1 library contains twenty-four routines covering a variety of problems in the areas of:

- ▷ Basic Linear Algebra,
- ▷ Solution of linear equations (Dense, banded, and sparse),
- ▷ Eigenvalue problems,
- ▷ Fast Fourier Transforms,
- ▷ Polynomial Equations,
- ▷ Linear Programming,
- ▷ Sorting.

The library cannot compare in size with the serial Nag Fortran library, but will grow rapidly over the next few years: Liverpool University and Nag are committed jointly to a fifty man-year project – within the Esprit 2 funded Supernode 2 project – to carry out further development leading to a larger library which will be marketed also by Nag Ltd. Even at its current size it provides the easiest route available to the porting of existing Fortran codes; and with a typical parallel routine taking six man-months to develop, we think that its the purchase price would be recouped by using just one routine once!

Acknowledgements

The research and development on which this library is based were carried out by staff in the Centre for Mathematical Software, University of Liverpool, with funding from:

1. The CEC under Esprit project P1085 ('Supernode'), led by Dr G Harp, RSRE Malvern.
2. The DTI and SERC under the UK Transputer Initiative, via two EMR contracts.
3. The University of Liverpool, via support for CMSR and for the North West Regional Transputer Support Centre set up under the Transputer Initiative at the University of Liverpool.

For more information phone, or write to:

Jean Hadfield +44 51 708 0123
Transputer Division
N. A. Software Ltd
Merseyside Innovation Centre
131 Mount Pleasant
Liverpool L3 5TF

MICROPROCESSORS AND MICROSYSTEMS

a journal published by Butterworth Scientific Limited

Two special issues of this journal were devoted to the transputer: 'Transputer Enabling Technology' in March 1989:

- ▷ *Solving problems with transputers: background and experience*, J. Wexler and D. Prior, Edinburgh University, UK.
- ▷ *Distributed operating systems for transputers*, C. H. R. Grimsdale, Perihelion Software, UK.
- ▷ *Multitransputer graphics subsystem*, J. D. Nicoud and Ph. Schweizer, EPFL, Switzerland
- ▷ *Mercury: an operating system for medium-grained parallelism*, H. Oakley, UK.
- ▷ *CS-PROLOG on multitransputer systems*, I. Futo and P. Kacsuk, Multilogic Computing, Hungary.
- ▷ *Message passing in a transputer network*, A. Knowles and T. Kantchev, University of Manchester, UK.
- ▷ *Peripheral handling techniques for the transputer*, P. Croll and G. Wilson, Sheffield University, UK.
- ▷ *Support system for occam objects on transputers*, I. Thomas, Southampton University, UK.
- ▷ *Commercial issues: parallel processing and the transputer*, N. Tucker, Paradis Consultants, UK.

and 'Transputer Applications' April 1989:

- ▷ *A programmable, digital waveform generator*, R. M. Smith, Interamerican Observatory, Chile.
- ▷ *Hybrid architecture paradigms in a radar data processing application*, R. D. Benton et al., Plessey Research, UK.
- ▷ *Automatic compilation of parallelism in visual object recognition*, G. B. Shoom and D. B. Skillicorn, Queen's University, Canada.
- ▷ *Transputer based simulation tool for performance evaluation of wide area telecommunications networks*, R. T. Clarke, S. J. Nichols and P. Mars, Durham University, UK.
- ▷ *Free-text retrieval on transputer networks*, M. Waldén and K. Sere, Åbo Akademi, Finland.

- ▷ *Transputer implementation of multiple target tracking*, E. Gul and D. P. Atherton, Sussex University, UK.
- ▷ *Development of a distributed algorithm for a robotics application on transputers*, W. van den Broek and H. de Boer, PTT Dr Neher Laboratories, The Netherlands.
- ▷ *Mapping image processing operations onto transputer networks*, R. F. Browne and R. M. Hodgson, University of Canterbury, New Zealand.
- ▷ *Implementation of the Hough transform for intermediate-level vision on a transputer network*, S. Eghtesadi and M. Sandler, King's College, London.
- ▷ *Transputer-based dataflow multiprocessor for robot-arm control*, S. Geffin and B. Furht, Modcomp Inc., USA.
- ▷ *Transputer control of a flexible robot link*, A. C. J. Stavenuiter, G. ter Reehorst and A. W. P. Bakkers, University of Twente, The Netherlands.

Copies of each issue are available at £12·00 each (£14·00 overseas) from Butterworth Scientific Limited at the address below.

Call for papers

To extend the theme of the recent 'transputer' special issues Microprocessors and Microsystems is starting a new featured series on 'Parallel processing and transputers in practice'.

The journal invites contributions of papers – design and development reports, review papers and tutorials, and descriptions of practical system applications – for each of the featured series in addition to papers that meet the existing requirements of the journal's established coverage. Microprocessors and Microsystems publishes fully refereed technical papers describing practical computing design and development experience and engineering applications. Papers should be sent to the Editor at:

Microprocessors and Microsystems
PO Box 63
Guildford
Surrey GU2 5BH
United Kingdom

REFERENCE

BIBLIOGRAPHY UPDATE

Articles authored by INMOS personnel

Debugging Transputer networks, Phillip Mattos, Microsystem Design, November 1988.

Global positioning by satellite, Phillip Mattos, Electronics & Wireless World, February 1989, pp. 137–142.

Articles authored by others

A Transputer Implementation of a McCulloch & Pitts Network, Francesco Abbruzzese, E. Chiricozzi and A. D'Amico (Editors), Parallel Processing and Applications, Elsevier Science Publishers (North-Holland), 1988.

Memory Managers in a Transputer Network, J. C. Admiraal, Shell International Research, Publication 875, December 1988.

Foundational Issues in Concurrent Computing, Gul Agha, Sigplan Notices, April 1989, pp. 60–65.

High Performance Ada Using Transputers, John Barnes & Colin Whitby Strevens, Defense Computing, Sept.–Oct. 1988, pp. 45–49.

Communication protocols & concurrency: an OCCAM implementation of X.25, Bouillat, J. E., *et al.*, 1988 Int. Zürich Seminar on Digital Communications, IEEE, Zürich, Switzerland, 8–10 March 1988, pp. 99–102.

Transputer-based upgrading of a laser photon correlator, F. Bruge, *et al.*, Rev. Sci. Instrum. 60(2), 1989, pp. 222–225.

Architectural Support for Synchronous Task Communication, Burkowski, F. J., Cormack, G. V. and Dueck, G. D. P., ACM Operating Systems Review, April 1989, pp. 40–53.

Pulling Together for power on demand, Mike Charnley-Fisher, Workstation Magazine, Dec.88/Jan.89, pp. 27–30.

A Closer Look at the Transputer, Pete Chown, Elecktor Electronics, May, 1989, pp. 39–43.

Real-Time Parallel Processor Systems using OCCAM and Transputer: A Case Study, Ciccarella, G., *et al.*, E. Chiricozzi and A. D'Amico (Editors), Parallel Processing and Applications, Elsevier Science Publishers (North-Holland), 1988.

View from the Ivory Tower: Transputers – Where do they fit in to Information Retrieval?, Janey Cringean, IT Link, December, 1988, pp. 10–13.

The Parallel Interpretation of Logic Programs in Distributed Architectures, P. Corsini, G. Frosini and G. Speranza, The Computer Journal, Vol 32(1), 1989, pp. 29–35.

Parallel Processing of the Square Root Free Givens Rotations by means of a Transputer Network, J. Duato, D. Chiricozzi and A. D'Amico (Editors), Parallel Processing and Applications, Elsevier Science Publishers (North-Holland), 1988.

Transputer – A component of supersystems, V. Dvořák, Sdelovaci Technica, 36(9), 1988, pp. 325–27), (In Czech).

Simulating microprocessor systems using occam and a network of transputers, I. Gorton, J. Kerridge and B. Jervis, IEE Proceedings, Vol. 136, Pt. E, Nº 1, 1989, pp. 22–28.

Distributed Data Processing Systems Based on Uniform Specialized Processors, L. Glukhin, *et al.*, Soviet Journal of Computer & System Sciences, Vol 26 (2), 1988, pp. 67–77.

Parallel Programming and Transputer Networks in C, Peter Guthseel, Mikrocomputer Zeitschrift, September 1988, pp. 78–81, (In German).

Parallel processing, the transputer and the future, Chris Jesshope, Microsystem Design, 13(1), 1989, pp. 33–37.

Configurable transputer architectures – the supercluster series, O. Kramer, Chip, 8, 1988, pp. 14–18.

Learning as the High Level Language for a Neural Network, Francesco E. Lauria, Cybernetics and Systems '88, pp. 383–390.

Transputer Computer Juggles Real-Time Robotics, Forrest Levin, ESD, 19(2), 1989, pp. 77–82.

Les supercalculateurs a transputers, Traian Muntean, La Recherche, No. 204, November 1988, pp. 1309–1319, (in French).

An Extended Occam model for timed parallel systems: Application to environmental robot programming, Traian Muntean & Michel Riveill, International Conference on Intelligent Autonomous Systems, 3–11 December, 1986, pp. 470–476.

Speed Up Transputers, Howard Oakley, Tech PC User, January 1989, pp. 26–36.

Image Processing on Transputer Arrays Programmed in OCCAM, G. C. Parodi, *et al.*, E. Chiricozzi and A. D'Amico (Editors), Parallel Processing and Applications, Elsevier Science Publishers (North-Holland), 1988, pp. 181–188.

Il Rasoio di Occam e le ricorrenze frattali, Rinaldo Reni, Sistemi e Automazione, Nº289, Febbrao 1988, pp. 104–109, (In Italian).

Architecture for a Massively Parallel Database Machine, N. Rishe, D. Tal and Q. Li, Microprocessing and Microprogramming, Vol. 25, 1989, pp. 33–38.

Simulation and performance evaluation of parallel software on multiprocessor systems, Luigi Rizzo, Microprocessors and Microsystems, 13(1), 1989, pp. 39–46.

Transputer 'Parallelizes' Computers, Gail M. Robinson, Design News, 19.12.88, pp. 82–93.

Art + 2 Years = Science, Phillip Robinson, Byte, 14(2), 1989, pp. 255–264.

Communication Protocol Specification in Occam, S. Sawyers, Factory 2000: IERE, Cambridge 1988, pp. 325–9.

Automatic Process Configuration in Occam2, A. Schutte, *et al.*, Informationstechnik IT 30 (1988) 4, pp. 272–284, (In German).

A Taxonomy for Computer Architectures, David B. Skillcorn, IEEE Computer, November 1988, pp. 46–57.

Understanding Multi-Transputer Execution, S. Stepney, UK IT 88 Conference, Swansea, UK, 4–7 July 1988, pp. 239–241.

An Environment for Concurrent Programming in Pascal, N. Q. Vasconcelos and E. S. T. Fernandes, Microprocessing and Microprogramming, 25, 1989, pp. 381–386.

Load Balancing in a Network of Transputers, Oliver Vornberger, 2nd International Workshop on Distributed Algorithms, Amsterdam, June 1988, pp. 116–26.

Parallel Processing and Database Architecture, N. Winterbottom, Int. Specialist Seminar on the Design & Application of Parallel Digital Processors, Portugal, 1988 (IEE), pp. 48–52.

The INMOS Transputer and Occam, No Author Given, Elektor Elektronics, November 1987, pp. 50–52.

NEW MEMBERS

The names below are of people who have joined the group or changed address in the five months up to 8th May 1989. They appear here as they are entered in the group's mailing list.

PROF J M ADAMO, Ecole Nationale Superieure, de Lyon, 46 Allee d'Italie, 69364 Lyon, Cedex 07, France

MR M AINSWORTH, Gemini Computer Systems Ltd, Unit 4, Springfield Road, Chesham, Bucks, HP5 1PW

JAMES AITKEN, Aitken Computer Systems, 16 Prebendal Court, Oxford Road, Aylesbury, Bucks, HP19 3EU

FRED A AKL, Associate Professor, Ohio University, Dept of Civil Engineering, Stocker Center, Athens, Ohio 45701-2979 USA

DR S C ALLCOCK, ISTEL Ltd, Ravensbank Drive, North Moons Moat, Redditch, Worcs, B98 9HB

ROBERT V ALLEGRETTI, Wandel, 1030 Swabia Ct, RTP, NC 27709, USA

MR MICHAEL L ANDREW, Falkirk College of Technology, Dept. of Elect. & Elec. Eng., Grangemouth Road, Falkirk, FK2 9AD

DR DAVID R ANDREWS, Kings College London, Strand, London, WC2R 2LS

DR K S ARABADJIYSKI, Dept of Computing, Imperial College, University of London, 180 Queen's Gate, London, SW7 2BZ

HEINRICH ARENS, Doruier GmbH, EKD, Postfach 1360, 7990 Friedrickskafeu 1, West Germany

JOHN CHARLES ASH, Marconi Underwater Systems Ltd, Elettra Avenue, Waterlooville, Hampshire, PO7 7XS

DR ARNOLD BACA, IFS, Dept. of Biomechanics, Auf der Schmelz 6, A-1150 Vienna, Austria

MR SERGEY BACHTEYAROV, International Research, Inst. for Man. Sciences (IRIMS), 9 Prospect 60-let Oktyabria, 117312 Moscow, USSR

ANTON BADER, MicroNet, Postfach 1205, 6458 Rodenbach, West Germany

MR PHILIP BAKER, Lee Colortran, Manchester Road, Kearsley, Bolton, BL4 8RL

S A BAKER, RSRE, St Andrews Road, Malvern, Worcs, WR14 3PS

MR P J BAKKES, University of Stellenbosch, Stellenbosch 7600, South Africa

J BALTAXE, Computer Services Centre, Victoria Univ of Wellington, P O Box 600, Wellington, New Zealand

MICHAEL BANE, 59 Hartington Street, Manchester, M14 4RP

LAURENCE E BANNER, Lead Scientist, Engineering Research Assoc, 1555 Springhill Road, Vienna, VA 22180, USA

MANUEL MARTINS BARATA, Rue M Lote 162, 7E, Casal De Sao Bra's, 2700 Amadora, Portugal

JAN BARDINO, RC/Jydsk Telefon, Forskerparken i Aarhus, Gustav Wiedsvej 10, DK 8000 Aarhus C, Denmark

JEFF J BARRATT, Navy Information Systems, The MITRE Corporation, Washington C3I Operations, 7525 Colshire Drive McLean, Virginia 22102-3184 USA

IAN BASS, Rapid Silicon, Denmark Street, High Wycombe, Bucks, HP11 2ER

J E BASS, University of Arkansas, Engineering Research Center, West 20th Street, Fayetteville, AR 72701, USA

DR A BASU, C-DAC, Pune University Campus, Pune - 411 007, India

RUEDI BAUMANN, Aarbergstrasse 123, CH-2502 Biel, Switzerland

DR. DOMINIK BAUMGARTNER, CELSI AG, Industriestr. 20, CH-8117 Fallanden, Switzerland

DR RAMON BEIVIDE, Dept de Informatica, Universidad del Pais Vasco, Apartado 649, 20080 San Sebastian, Spain

MR N BENJAMIN, FPC 267, Sowerby Research Centre, British Aerospace, P O Box 5, Filton Road, Bristol BS12 7QW

J T BENNETT, Ultrasonic & Transducer Systems, P O Box 395, Pretoria 0001, South Africa

M E BERRY, 100 Church Street, Enfield, Middlesex, EN2 6BQ

DR H E BEZ, Loughborough University, Department of Computer Studies, University of Technology, LE11 3TU

MR LOKESH BHASIN, C-DAC, Udyog Bhavan, Ganeshkhind Road, Pune 411 007, India

PROF P C P BHAT, Head, Dept. of Comp Sci, I.I.T. Delhi, Hauz Khas, New Delhi 110016

DR VIJAY P BHATKAR, Executive Director, C-DAC, Udyog Bhavan, Ganeshkhind Road, Pune 411 007, India

DAVID BICKNELL, Computer Weekly, Quadrant House, The Quadrant, Sutton, Surrey, SM2 5AS

Lionel Bidwell, Trent Polytechnic, Dept of Computing, Burton Street, Nottingham, NG1 4BU

Shawn Miles Bilodeau, 49 Elmhurst Avenue, Albany, NY 12203-3222, USA

Rainer Bischoff, Prinz-Handjery-Str. 32a, D-1000 Berlin 37, West Germany

Morten L Bjarnar, Seatex A/S, Pirsenteret, Brattora, N-7010 Trondheim, Norway

John Blairfish, E.V.C.S, Kings Buildings, Mayfield Road, Edinburgh, EH9 33Z

Harold Blanchard, Concept Moderne SA, 13 Av Ste-Clotilde, CH-1205, Geneva

Tony Blighe, Flight Electronics Ltd, Flight House, Ascupart Street, Southampton, Hants, SO1 1LU

Bernhard Bloechl, Rupperst. 8, 8000 Munich 2, W-Germany

Prof F Boeri, Lab de Sigmaux et Systems, 41 Boulevard Napoleon III, 06041 Nice, Cedex, France

Karl H Bohle, FR-FBERT-Str.9.177, D7032 Sindelfingeh, West Germany

Ronalo Boontje, Nikhef-k CS6, Postbus 4395, 1009 AJ, Amsterdam

Christopher Booth, RIPR - RSRE, St Andrews Road, Malvern, Worcs, WR14 3PS

L J Van Dem Bos, W Druckerstraat 50, 7741 RM Coevorden, Netherlands

Paolo Brazzale, ESA Estec, P O Box 299, 2200 EG Noordwijk, Holland

Francois Bret, Dept Informatique, Universite de Tours, Park Grandmont, 37200 Tours, France

Keith Bromich, 25D Oxford Road, Ealing, London, W5 3SP

Paul Bromley, INMOS Ltd, 1000 Aztec West, Almondsbury, Bristol, BS12 4SQ

Per Owind Brosstad, Seatex As, Pirsenteret, Brattora, 7010 Trondheim, Norway

E D Browne, University of Adelaide, Dept of Mechanical Engineering, GPO Box 498, Adelaide SA 5001, Australia

Janet Bunn, Ministry of Defence, Royal Signals, St Andrews Road, Malvern, Worcs, WR14 3PS

John Burnell, Applied Maths Division, P O Box 1335, Wellington, New Zealand

Dr Alan Burns, University of Bradford, Bradford, West Yorkshire, BD7 1DP

Mr Butterfield, Hatfield Polytechnic, Div. of Elect. Eng., Box 109, Hatfield, Herts, AL10 9AB

Dr D Caban, Institute of Eng. Cyber., Wroclaw Tech. University, Worclaw 50-372, ul Janiszewskiego 11/17, Poland

Renato Campanini, c/o Dipartimento di Fisica, Via Irnerio 46, 40126 Bologna, Italy

Ole Caprani, Computer Science Dept, Aarhus University, Ny Muukegade, DK-8000, Denmark

Trevor Carden, Thorn EMI CRL, Dawley Road, Hayes, Middlesex, UB3 1HH

Torgny Carlsson, Y/ORDC, Ericsson Radar Electronics AB, 43184 Molndal, Sweden

Dr Geoffrey F Carpenter, Aston University, Aston Triangle, Birmingham, B4 7ET

Dale Cass, GEC Transmission DPL, Dept 734, P O Box 27, Lichfield Road, Stafford, ST17 4LN

Antoni Llorens Castello, Valencia, 93 Pral 19, 08029 Barcelona, Spain

G B Cecchet, I.N.F.N., University of Pavia, V. Bassi 6, I-27100 Pavia, Italy

Ugo Cei, Universita di Pavia, Dip Informatica e Sistemistica, Via Abbiategrasso, 209 Pavia 27100, Italy

Miss Carolynn Chalmers, Computer Science Dept., University of Natal, Pietermaritzburg, 3201 Natal, South Africa

Mr Kap-luk Chan, Computer Vision Lab, Dept of Mechanical Eng., Imperial College, Exhibition Road, London, SW7 2AZ

Khee Chan, Sattelite International Ltd, The Paddock, Hambridge Road, Newbury, Berks, RG14 5TQ

Paul Chapman, Room 0.56, Queens Building, University Walk, Bristol, BS8 1TR

U P Cheah, 116 Ridge Road, Letchworth, Herts, SG6 1PT
G W Chege, Dept. of Electronics, University of York, Heslington, York, Y01 5DD
Dr Y S Cheung, Senior Lecturer, Dept. of EEE, University of Hong Kong, Pokfulam, Hong Kong
Jean-Marie Chretien, Faculte de Medecine, Laboratoire de Biologie, Theorique, 1 rue haüte de Reuter, 49045 Angers Cedex, France
Bill Christmas, BL Research Centre, Chertsey Road, Sunbary, Middlesex, TW16 7LN
Mr P J C Clare, Thorn EMI, Central Research Laboratories, Dawley Road, Hayes, Middlesex, UB3 1HH
Charles Clayton, 4A Berners Road, London, N22 5NE
Ron Coates, Barleyfields Cottage, Newton Purcell, Oxon, MK18 4AN
Adrian Cockcroft, Sun Microsystems Ltd, Merlin Place, Milton Road, Cambridge, CB4 4DP
Dr Malcolm Coe, Dept of Physics, The University, Southampton, SO9 5NH
P M Conlan, Systems Programmer, Imperial Chemical Industries PLC, Room R203,Topfloor Block R, P O Box 14, The Heath, Runcorn, Cheshire, WA7 4QG
Mauricio de Menezes Cordeiro, LCDR Brazilian Navy, Rua Joao Geraldo Kuhlman 156, Barra oa Tijuca - Rio de Janeiro, 22793 Brazil
J E Cowey, Dept of Electronic Engineering, Leeds Polytechnic, Calvesley Street, Leeds, LS1 3HE
Chris Craig, Mimansa Systems, Lower Tytherleigh Farm, Tytherleigh, Axminster, Devon, EX13 7AZ
Angel Creus, ISEL, Auda. Gran via de les Corts, 184 6, Barcelona 08004, Spain
S J Cronk, 7 Church Lane, Royston, Herts, SG8 9LG
N A Cropper, MEL (Philips), Manor Royal, Crawley, West Sussex, RH10 2PZ
B Crouch, Thorn EMI Electronics Ltd, Computer Systems Division, Penleigh Works, Wookey Hole Road, Wells, Somerset BA5 1AA
E H Croucher, 4 Sea View Park, Whitburn, Sunderland, SR6 7JS
Paul Daniels, Qsys Consultancy Ltd, 96 Appledore, Great Hollands, Bracknell, Berks, RG12 4QZ
Mick Davies, Dept. of Med. Physics, 1st Floor, Shropshire House, 11-20 Lapper St, London, WC1E 6JA
James L Davis, Data Base Administrator, Parklawn Computer Center, 9004 Manchester Road, Silver Spring, Maryland 20901, USA
Mark Davoren, Edinburgh Concurrent, Supercomputer project, Edinburgh University, EUCS Rm 2408 JCMB, Edinburgh, EH9 3JZ
C E Dawe, Easams Ltd, Lyon Way, Frimley, Camberley, Surrey, GU16 5EX
C Demopoulos, 5-7 Mikalis Str, 162 31 Byron, Athens, Greece
Dr Keith Dimond, Electronic Engineering Lab, University of Kent, Canterbury, Kent, CT2 7NT
Fraser Donachie, Marconi Defence Systems Ltd, The Grove, Warren Lane, Stanmore, Middlesex, HA7 4LY
R Downhill, McDermott Eng London, 140 Wembley Park Drive, Wembley Park, Middx, HA9 8JD
Ms Sue Downhill, SNP Ltd, 7 Meadway, Ashford, Middlesex, TW15 2TJ
Will Dracup, Jumb Beck Farm, Burley Woodhead, Ilkley, W Yorks, LS29 7AX
David W Drell, United Technologies/USBI, 520 Old Spanish Trail, Slidell, Louisiana 70458, USA
Sarah Duckworth, A.I. Ltd, Greycaire Road, Watford, Herts, WD2 4JP
Dr Evgeny Dudnikov, Internation Research, Inst. for Man. Science (IRIMS), 6 Prospect 60-let Oktyabria, 117312 Moscow, USSR

STEPHEN DUNCAN, Department of Physics, Astronomy Group, University of Southampton, Southampton, SO9 5NH

A DYKSTRA, State Univeristy of Utrecht, Dept. of Computer Science, P O Box 80089, 3508 TB Utrecht, Holland

KARLHEINZ EBERHARDT, Klingerstr. 34, 8000 Muenchen 70, Munich, Germany

TIM ECCLES, Tektite, P O Box 5 Felixstowe, Suffolk, IP11 7LW

NIGEL EDWARDS, Hewlett Packard Labs, Filton Road, Stoke Gifford, Bristol, BS12 6QZ

PETER J H VAN EETVELDE, State University of Ghent, Grote Steenweg Noord 2, B-9710, Ghent-Zwynaarde, Belgium

E EID, Pixellar Imaging Systems Corp, 21 Antares Drive Suite 123, Nepean, Ontario, Canada, K2E 7T8

B C ELLIS, Computing Sercies Centre, Victoria University of Wellingto, P O Box 600, Wellington, New Zealand

GRAHAM K ELLIS, 1103 Horseshoe Lane, Blacksburg, Va 24060, USA

EIVIND ENGEBRETSEN, Norweigen Defence Res Est, P O Box 25, N-2007 Kjeller, Norway

FERDINAND ENGLBERGER, Universitat der Bundeswehr Munc., Fakultat fur Elektrotechnik, Institut fur Nachrichtentechnik, Werner-Heisenberg-Weg 39, 8014 Neubiberg, Germany

ANDREAS EPPINGER, Merkleweg 4, 7000 Stuttgart 70, West Germany

ERGUN ERGIN, School of Computer Science, Polytechnic of Central London, 115 New Cavendish Street, London, W1M 8JS

MEHMET RASIT ESKICIOGLU, University of Alberta, Dept of Computer Science, 615 General Service Bldg, Edmonton, Alberta, Canada, T6G 2H1

DR C ESSERY, Dept of Computing Science, University of Ulster, Jordanstown, Newtownabbey, Co Antrim, BT37 0QB

DANIELA EUFRI, Agusta Sistemi S.P.A., Via Isonzo 33, 21049 Tradate (VA), Italia

TERRY EVANS, Rapid Silicon, Denmark St, High Wycombe, Bucks, HP11 2ER

ERNST FABRICUS, Heinrich-Lanz Str 3, D-6900 Heidelberg, Germany

CRAIG FARRELL, c/o School of Computing Science, Curtin University, G P O Box U1987, Perth 6001, Western Australia

DR K V FERNANDO, NAG, Wilkinson House, Jordan Hill, Oxford, OX2 8DR

ROBERT L FERREIRA, Parker Hannifin Corp, 14300 Irvine, CA 92718-1814, MS K141, USA

ROGER FEYEREISEN, 46 rue Pierre Wiser, L-9092 Ettelbruck, Gd Duchy of Luxembourg

CHARLES FINLAY, M/S 23-3, NASA Lewis Research Center, 21000 Brookpark Road, Cleveland, OH 44135, USA

JOHN W FINNEGAN, Lecturer, Engineering School, National University College, Galway, Republic of Ireland

PAOLO FINOCCHIARO, Laboratorio Nazionale del Sud, V.le A.Doria Ang. V.S. Sofia, 95123 Catania, Italy

G FORSSTROM, AUT/KHX, ABB Automation, S-72167 Vasteras, Sweden

ROY FORSSTROM, Public Works Computer Services, 25 West 4th Street, St Paul, MN 55102, USA

CHIAVETTA FRANCO, Vi2 F. Di Giorgi No 4, 90145 Palermo, Italy

DAVID FRANKLIN, Plessey Research, Roke Manor, Romsey, Hants, SO51 0ZN

DR T L FREEMAN, Dept of Mathematics, University of Manchester, M13 9PL

LEE FREITAG, Woods Hole Oceanographic, Institution, Woods Hole, MA 02543, USA

DR MANFRED FUCHS, Kaiser - Joseph Str 191, D7800 Freiburg i.Br, West Germany

NIGEL GAMBLE, Modular Computer Systems Inc, 1650 West McNab Road, P O Box 6099, Ft Lauderdale, FL 33340-6099, USA

JEFF GANAPOSKI, Seven Mountains Scientific Inc, P O Box 650, Boalsburg, PA 16827, USA

R GEARY, Glasgow College, Cowcaddens Road, Glasgow, G4 0BA, Scotland

MR H E GEORGE, George Consultants Ltd, International House, 2-4 Wendell Road, London, W12 9RT

CUTTONE GIACOMO, Laboratorio Nazionale del sud, V.le a Doria Ang. V.S. Sofia, 95123 Catania, Italy

KLAUS GIERE, c/o IABG. Abt. WTS, Einsteinstr.20, 8012 Ottobrunn, West Germany

BALBIR GILL, 89 North Crescent, Southend-on-sea, Essex, SS2 6TH

MR MARC GIROD, Instrumentarium Imaging, Teollisuuskatu 27, SF 00510 Helsinki, Finland

KEITH R L GODFREY, Dept. Elect. & Elect. Eng., The University of W.A., Western Australia 6009, Australia

GERHARD GONTER, Guenser Strasse 3E, A-2700 Wiener Neustadt, Austria

N J GOODENOUGH, Dept of Electronic Engineering, Sheffield University, Mappin Street, Sheffield, S1 3JD

DR IAN GORTON, Dept. of Comp. Science, University of Wollongong, P O Box 1144, Wollongong, NSW 2500, Australia

ERNST GOSTELI, Baumgartenweg 22, CH - 8471 Rutschwil, Switzerland

J F M GRAAT, Fideliohof 74, 2402 ET Alphen A/D rijn, Netherlands

VITTORIO GRANDI, Via Bezzecca 9, 19100 La Spezia, Italy

YVES GRENIER, Enst, Dept Signal, 46 rue Barrault, 75639 Paris, Cedex 13, France

PHILIPPE GUGLIELMETTI, Institute d'Automatique, Ecole Polytechnique Federale, 1015 Lausanne, Switzerland

RAFAL GUTKOWSKI, Institute of Computer Science, Policy Academy of Sciences, 00-901 Warsaw, P O Box 22, Poland

A M HACKING, GEC Research, Marconi Research Centre, Room B265, West Hanningfield Rd, Great Baddow, Chelmsford Essex CH2 8HN

JOHN HADJIOANNOU, White Cross Systems, 90 London Road, London, SE1 6LN

IVAN HALLWORTH, 49a Berwick Road, Marlow, Buckinghamshire, SL7 3AS

B OLDE HAMPSINK, Universiteit Twente, Afd. Informatics, Postbus 217, 7500 AE Enschede

DR L K HANSEN, Andrex, Halfdansgade 8, DK-2300 Copenhagen s, Denmark

STEVEN HARRIS, 47 Park Road, Gloucester, GL1 1LR

STEPHEN HART, 24132-A Hollyoak, Laguna Hills, CA 92656, USA

SIMON J HART, Combat Systems Trainer Facility, Fleet Base Asset 902, Garden Island, NSW 2070, Australia

KOICHI HASHIMOTO, Dept. of Mechanical Eng., Faculty of Engineering, Osaka University, 2-1 Yamadaoka, Suita, Osaka 565, Japan

JOHN HAYWOOD, Access Electronic Components, Jubilee House, Jubilee Road, Letchworth, Herts, SG6 1QH

HELMUT HELLER, 8046 Garching, FRG, James-Franck-Strasse, Physics Dept. T30, Munchen, Germany

STEFAN HELLWIG, GFTT, Emil-Figge-Str. 76, 4600 Dortmund 50, West Germany

CAROLE HENSHAW, Rapid Silicon, 28 High Street, Nantwich, Cheshire, CW5 5AS

N J HERROD, Laboratory for Medicinal, Chemistry, Level 4 RTC, Addenbrookes Hospital, Hills Road, Cambridge CB2 2QQ

SHAUN HERROW, Inst. of Chartered Accountants, Gloucester House, 399 Silbury Boulevard, Central Milton Keynes, MK9 2HL

PROF R D HERSCH, EPFL Laboratoire de Systemes, Peripheriques, Av de Cour 37, CH-1007 Lausanne, Switzerland

Dr Markus Heuer, AC Laboratory, CH-3700 Spiez, Switzerland

Mr P M Highams, 1 Brookfield Avenue, Larkfield, Maidstone, Kent, ME20 6RT

Kenneth C Ho, IBM T J Watson Research Center, P O Box 704, MS 71-D13, Yorktown Heights, NY 10598, USA

M V Hobden, Ministry of Defence, RSRE, St Andrews Road, Malvern, Worcs, WR14 3PS

T Hobden, Seascan Technology Ltd, Unit 6, Crown Avenue Ind. Estate, Tredegar, Gwent, NP2 4EF

Helge Hoen, Miros A/S, Postbox 364, N-1371 Asker, Norway

Frank Hoffman, Institut fur Theoretische, Physik, D 2300 Kiel, Olshausen str. 40-60, Germany

Aimo Hoffren, R & D Engineer, Instrumentarium Corp, Imaging Division, PL 357, SF-00101 Helsinki, Finland

J Hofstede, Dept. Computer Science (INF/SPA), PB 217, 7500 AE Enschede, Netherlands

Mark Holden, Dept. of Computer Science, University College, Gower Street, London, WC1E 6BT

Professor Bernard Hon, Manufacturing Systems Res. Grp., University of Dundee, Dundee, DD1 4HN

Bob Howard, National Transputer Support, Centre, Sheffield Science Park, Arundel St, Sheffield, S Yorks S1 2NS

Fred Howell, 3 Southern Road, East Finchley, London, N2 9LH

Wen-Shyong Hsieh, National Sun Yat-sen University, Kaohsiung, Taiwan (80424), R.O.C.

Daniel Huber, SOHARD AG, Galgenfeldweg 18, Postfach 235, CH-3000 Bern 32, Switzerland

S J Hurst, Askew House, Milton Road, Repton, Derbyshire, DE6 6FZ

Jeremy M Hutson, Department of Chemistry, University of Durham, Durham, DH1 3LE

Roland Hutter, Breithaupstr. 1, 7528 Karlsdorf-Neuthard 1, W Germany

Ing. Giulio Iannello, Instituto FMI - Fac. Ingegneria, Universita di Salerno, 84084 Fisciano SA, Italy

Takao Ichiko, Dept. of Information Engineering, Faculty of Engineering, National Yamagata University, 3-16 Johnan 4-Chome, Yonezawa City, Yamagata 992 Japan

Michael J G Innes, Advanced Computing Dept, Thorn EMI Central Research Labs, Dawley Road, Hayes, Middx, UB3 1HH

Niels Jul Jacobsen, Haraldsgade 6 st. th., 5000 Odense c, Denmark

J L Jacquemin, Departement GEII, IUT de Nimes, Rue Raimu, 30 000 Nimes, France

C T F Jansen, Victoria University of, Wellington, Computing Services Centre, P O Box 600, Wellington, New Zealand

Colin Jenkins, Vision Information, Systems Limited, 15 Western Road, Hurstpierpoint, West Sussex, BN6 9SU

Bernd Jessen, GPA mbH, Gruberstr. 46b, D - 8011 Poing, Germany

Royer Joel, Humelec Informatique za Courtabo, No 521 Local Quebec, 91946 Les Ulis, Cedex, France

Gerald C Johns, 3615 Dry Folk Road, Imperial, Missouri 63052, USA

Joanne Johnson, Access Electronic Components Ltd, Jubilee House, Jubilee Road, Letchworth, Herts, SG6 1QH

Timothy Carl Johnson, 1530 Fulham Street, St Paul, MN 55108, USA

A M Jones, Ferranti International, Bird Hall Lane, Cheadle Heath, Cheshire, SK3 OXQ

Geraint Jones, National Transputer Support, Centre, Sheffield Science Park, Sheffield, S Yorks, S1 2NS

PETER JONES, Dept of Computer Science, University of Manchester, Oxford Road, Manchester, M13 9PL

WOLFGANG JUNG, University of Kaiserslautern, P O Box 3049, Erwin-Schrodinger-Str, D-6750 Kaiserslautern, West Germany

KARL JUNK, Kaiserstr. 66, 7500 Karlsruhe, West Germany

SCHEIBER JURGEN, Petrusgasse 1614, 1030 Wien, Austria

DR THEODORE KALAMBOUKIS, The Athens School of Economics, Dept. of Statistics and Inf., 76 Patission Street, Athens 104 34, Greece

DR UNSAL KALAYCIOGLU, Kfupm Box 1770, Dhahran 31261, Saudi Arabia

EUTHYMIOS KAPPOS, Dept. of Elect/Elect Eng, The University, Newcastle-Upon-Tyne, NE1 7RU

MAG. PETER KARLSREITER, EDV-Zentrum, Boltzmanng. 5, A - 1090 Wien, Austria

S K KARMARKAR, Centre for Development of, Telematics, Sneha Complex, 71/1 Miller Road, Bangalore 560 052, India

DR A M KARNIK, B-4/150/2 Safdarjung Enclave, New Delhi 110 029, India

DIANE N KATO, Ford Aerospace Corporation, Ford Road, Newport Beach, CA 92658, USA

GUO YI KE, Dept of Computing, Imperial College, 118 Queens Gate, London, SW7 2AZ

D KELLY, 91 Repton Street, Leicester, LE3 5FD

ROB KENNEDY, General Dynamics Pomona Division, MZ 4-41, P O Box 2507, Pomona, CA 91769, USA

B M S KHALAF, School of Computer Studies, Leeds University, Leeds, LS2 9JT

M KHOUDI, Room 0.65, Queens Building, University Walk, Bristol, BS8 1TR

JAMES KIDD, ALTIS, 9 Brackley Street, Stockton Heath, Warrington, Cheshire, WA4 6DY

PAUL J KING, Struan Thie, Sulby Glen, Lezayre, Isle of Man, British Isles

TONY KING-SMITH, 30 The Street, Uley, Nr Dursley, Glos, GL11 5SY

DR R P KNOTT, Dept of Computer Studies, University of Loughborough, Leics, LE11 3TU

MR MARK W H KNOWLES, MEL (Philips), Manor Royal, Crawley, Sussex, RH10 2PZ

MR KOHLER, Brosius, GmbH, Berliner Strasse 23, D-5600 Wuppertal 2, W Germany

J B KOOLMEES, Dept R & D, RM, Haagsteeg 2, 6700 AA Wagenimgen, The Netherlands

K KORDI, Department of Computing, Trent Polytechnic Nottingham, Burton Street, Nottingham, NG1 4BU

P K KOURMOULIS, 61 Salisbury Road, Crookes, Sheffield, S10 1WA

PETER A KRAUSS, Universitat der Bundeswehr, Informatik, Werner-Heisenberg-Weg 39, D-8014 Neubiberg, West Germany

DR.-ING. TH. KREIS, BIAS, Ermlandstrasse 59, D-2820 Bremen 71, West Germany

A A M KUIJK, CWI Department IS, Kruislaam 413, 1098 SJ Amsterdam, The Netherlands

MR LIN KUNHUA, P O Box 2704-11, Beijing, Postal code 100080, P R China

PAUL KWAN, Graduate Office, Dept. of Electrical Engineering, University of Toronto, Toronto, Ontario, Canada M55 1A4

MR SIU-WAH LAU, Dept. of Computer Science, University of Hong Kong, Pokfulam, Hong Kong

DR MARK LAUDER, Oxford Computing Services Ltd, 52 St Giles, Oxford, OX1 3LU

RUDY LAUWEREINS, Katholieke Universiteit Leuven, ESAT Laboratory, Kardinaal Mercierlaan 94, B-3030 Heverlee, Belgium

DR A M LAYFIELD, Dept. of Elect. Engineering, City Poly. of Hong Kong, Tat Chee Avenue, Kowloon Tong, Hong Kong

DR ALEX F LEHAR, Ektron Applied Imaging, 23 Crosby Drive, Bedford, Massachusetts 01730, USA

GUY LEMSTROM, Kaupralantie 25B15, 02700 Kauniainen, Finland
S LEUTWYLER, Institut fur Anorgan, Analyt., Freiestrasse 3, CH 3000 Bern 9, Switzerland
DR G LIBERT, Faculte Poly. de Mons, 7000 Mons (Belgigue), Rue de Houdain, 9, Belgium
KRISTIAN LINDGREN, Physical Resource Theory Group, Chalmers University of Tech., S-41296 Goteborg, Sweden
KEN LINTON, c/o Audio Eng. Res. Group, SEAS, Durham University, South Road, Durham, DH1 3LE
JOSE M LLABERIA, Facultad de Informatics (UPC), Pau Gargallo, 5, 08028, Barcelona, Spain
SCOTT LLOYD, 950 N. University Avenue, Provo, Utah 84604, USA
STEVE LLOYD, Bioengineering ONIT, UHW, Heath Park, Cardiff, CF4 4XW
G LOAEC, Infremer, BP 70, 29263 Plouzane, France
H J LORMANS, Trilogic R & D B.V., P O Box 90, Systeren 6114 ZH, The Netherlands
MICHALIS LOUPIS, 43 Ellanikou Str., 116 35 Athens, Greece
DR N P LUCKMAN, Dept. of Medical Physics, Bristol General Hospital, Guinea Street, Bristol, BS1 6SY
SIMON P H LUI, University of Hong Kong, Dept. of Elect & Elect Eng, Pokfulam Road, Hong Kong
ZAFFALON LUIGI, E.I.G., 4 rue de la Prairie, 1202 Geneve CH, Switzerland
RUDOLF LUKAS, Romerstrasse 27, 5000 A A R A U, Switzerland
RUPERT LUNG, R G B Computer Graphics Ltd, 27 Warple Way, London, W3 0RQ
JOACHIM LUTZ, J.W. Goethe University, Prakt Informatik, VSFT, P O Box 111932, D-6000 Frankfurt, West Germany
DR R W LYTTLE, Dept. of Computer Science, Queens University, Belfast, BT7 1NN
RICHARD MCGUIRE, Hybrid Memory Products Ltd, Elm Road, West Chirton Ind Estate, N Shields, Tyne, NE29 8SE
MR H J MACKENZIE, Dept of Mech. Eng., Swinburne Inst. of Tech, P O Box 218, Hawthorn 3122, Australia
S J MCKENZIE, Niche Technology, Jigsaw Business Centre, Centre Gate, Colston Avenue, Bristol
W MACKEOWN, Dept of Computer Science, University of Bristol, Queen Building, University Walk, Bristol, BS8 1TR
IAIN MCWILLIAMS, INMOS Limited, 1000 Aztec West, Almondsbury, Bristol, BS12 4SQ
LIAM MAGUIRE, The Queens University of Belfast, Dept of Elect, Ashby Building, Stranmillis Road, Belfast, BT9 5AH
WALTER MAISEL, Universitat Karlsruhe, Inst. fur Algorithmen, und Kognitive Systeme, Haid-und-Neu-Strasse 7, 7500 Karlsruhe, West Germany
TOR-ERIK MALEN, Abo Akademi, Dept. of Computer Science, Lemminicaiseng 14, 20520 Turku, Finland
A K MALIK, EAPS-II, University of Sussex, Falmer, Brighton, Sussex, BN1 9QT
MATILE MALIMABE, E Floor, Richmond Building, University of Bradford, Richmond Road, Bradford, BD7 1DP
DALIA MALKI, IBM Research, IBM T.J. Watson Research Centre, P O Box 218, Yorktown Heights, NY 10598, USA
DR PETER MANI, GRD, NC-Lab, CH-3700 Spiez, Switzerland
F C MARCH, Computing Services Centre, Victoria University of, Wellington, P O Box 600, Wellington, New Zealand
WINFRIED P MARMIT, Uhlandstrasse 14, 2000 Hamburg 76, West Germany
S J MARSH, MASD, GEC Avionics, Airport Works, Rochester, Kent, ME1 2XX
MICHAEL S MARSTON, 2311 Bisontine, Friendswood,, TX 77546, USA
TERRY MAWBY, Rutherford Appleton Lab., Chilton, Didcot, Oxon, OX11 0QX

J J MERELU, Dpto. Electronica, Colegio Universitario, 04071 Almeria, Spain

JOHN MERLIN, Southampton University, Department of Physics, The University, Southampton, SO9 5NH

MR MICHAEL MERNAGH, Access Electronic Components, Jubilee House, Jubilee Road, Letchworth, Herts, SG6 1QH

MR J H PAUL MILLAR, Instrument Section, BLD 376, UK Atomic Energy Authority, Springfields Laboratories, Salwick, Preston, Lancs, PR4 0PR

GLENN MILLER, National Transputer Support, Centre, Sheffield Science Park, Arundel Street, Sheffield, S1 2NS

DANIEL MILLOT, LRI, Bat 490 - Orsay, France

GARY MILLS, 845 Waller Street, Apt #3, San Francisco, CA 94117, USA

GEIR MILLSTEIN, Norw. Telecomm. Adm., Research Dept., Instituttvn. 23, N-2007 Kjeller

MR A T MITCHELL, Mullard Space Science Lab, Holmbury St Mary, Dorking, Surrey, RH5 6NT

DR J J MODI, 7 Rushey Close, Leicester, LE4 7PT

THOMAS MOHR, DMA, EPFL, CH-1015 Lausanne, Switzerland

DR AFONSO MOLINA, Research Centre for Social, Sciences, University of Edinburgh, 56 George Square, Edinburgh, EH8 9JU

SAM MONACO, Hughes Aircraft Co, 1600 Randolph CT SE, Albuquerque, NM 87106, USA

MR ERIK MONCRIEFF, 17/5 The Gallolee, Redford Road, Colinton, Edinburgh

MARID MONTANARI, Hs Elettronica, Via Verdi 47/2 - 49, 40050 Villanova di Castenaso, Bo, Italy

PETER C MORRIS, Merino Wool Harvesting Pty Ltd, Innovation House, Technology Park, Pooraka, South Australia 5095, Australia

JEAN MOSZKOWSKI, Supor s.a.r.l, Chemin Des Clos 1, 78830 Bonnelles, France

H T MOUFTAH, Queens University at Kingston, Department of Electrical Eng., Queens University, Kingston, Ontario, Canada K7L 3N6

FABRICE MOURLIN, L.R.I., Bat 490, 91405 Orsay, Cedex, France

DR PETER MYERS, Phase Separations Ltd, Deeside Ind Park, Deeside, Clwyd, CH5 2NJ

JONG HO NANG, Dept. of Computer Science, Kaist, P O Box 150, Cheongryang, Seoul 131-650, Korea

JOHN R NAYLOR, AMS Industries PLC, AMS Industrial Park, Billington Road, Burnley, Lancs BB11 5ES

KEVIN NEAL, Doplar Designs Ltd, 9 Harris Road, Porte Marsh Industrial Estate, Calne Wiltshire

ERLING NIELSEN, Glaciset 23, 2800 Lyngby, Denmark

MR KAZUO NOGUCHI, 1st Advanced Tech Dev Dep, Advantest Corp, 1-16-1 Fujimi-cho, Gyoda 361, Japan

DONALD NOID, Oak Ridge National Lab, Post Office Box 2008, Oak Ridge, Tennessee 37831, USA

T H NORMANTON, Software Research Engineer, Smiths Industries, Bishops Cleeve, Cheltenham, GL52 4SF

PROF. JADRANKO NOVAK, Univ. of Trieste, Dept of Elect. Electronic and, Computer Engineering (DEEI), Via A. Valerio 10, 34127 Trieste, Italy

GREG OAKES, Mail Station 1517, Honeywell, DASD, 9201 San Mated Blvd, N.E., Albuquesque, NM B7113, USA

S OMAROUAYACHE, Dept of Info. Engineering, The City University, Northampton Square, London, EC1V 0HB

LUIS OSORIO, R. Alves Redol 9, 1000 Lisboa, Portugal

ANDERS OSTERBERG, Bofors Aerotronics AB, S-181 84 Lidingo, Sweden

DANIEL GUSTAVO OTERO, Zabala 1664 Z B, (1426) Capital Federal, Buenos Aires, Argentina

DR RICHARD E OVERILL, Department of Computing, Kings College London, Strand, London, WC2R 2LS

LARRY D OWENS, 1113 Nottingham Drive, Glen Burnie, MD 21061, USA

J M PARKER, Advanced Crew Stations, MUD PE, Rarde (Chertsey), Chobham Lane, Chertsey, Surrey KT16 0EE

WILL PARKER, K-Par Systems Ltd, 40 St Thomas Street, Bristol, BS1 6JZ

GIANCARLO PARODI, c/o DIBE - Universita GE, via All'Opera pia 11/A, 16145 Genova, Italia

DR MINESH PATEL, Logica Cambrige Ltd, Betjeman House, 104 Hills Road, Cambridge, CB2 1LQ

DR A PAULRAJ, Director, C-DAC, Udyog Bhavan, Ganeshkhind Road, Pune 411 007, India

ANDREW PAYNE, Wellcome Research Labs B76H, Langley Court, Beckenham, Kent, BR3 3BS

MR ROBERT PEARCE, Rm 0.27 Queen's Building, University Walk, Bristol, BS8 1TR

PHILIP JOHN PEARSON, c/o Smith Associates Ltd, Surrey Research Park, Guildford, GU2 5YP

MR ZIE PENG, University of Twente, Dept. of Informatics, SPA, P O Box 217, 7500 AE Enschede, Holland

ALAN PENNY, Rutherford Appleton Labs, Space Science Dept, Chilton, Didcot, Oxfordshire, OX11 0QX

M PERIASAMY, Centre for Dev of Telematics, Sneha Corporation, 71/1 Millers Road, Bangalore 560 052, India

CHRIS PETERSON, Rapid Silicon, 28 High Street, Nantwich, Cheshire, CW5 5AS

MR C I PHILLIPS, Dept. of Elec. Engineering, Queen Mary College, Mile End Road, London, E1 4NS

H PICHLIK, P T D S, Hersbrucker Str. 3, 8503 Altdorf bei Nurnberg, West Germany

LLUIS PINA, ISEL, Gran via de les Corts, 184 6, Barcelona 08004, Spain

DR ANDRZEJ PLASKOWSKI, ul. Anielewicza 33m 27, 01-057 Warszawa, Poland

R SCOTT PLAYTER, Electronic Forms Systems, 2395 Midway Road, Station 656, Carvolton, Texas 75006, USA

D PLUMMER, GEC-Marconi Research, Great Baddow, Chelmsford, CM2 8HN

KEN POOLE, 2631 Riding Ring, Norco, CA 91760, USA

MR N POORE, Gems of Cambridge Ltd, J1 The Paddocks, 347 Cherry Hinton Road, Cambridge, CB1 4DH

HEIKKI PUTKONEN, University of Oulu, Institute of Inf. Proc. Science, Linnanmaa, SF-90570 Oulu, Finland

N B QUIN, 54 Westbury Road, New Malden, Surrey, KT3 5AX

MARTIN RABER, c/o Prof Dr R Wilhelm, Fachbereich 10. Informatik, Universitat des Saarlandes, D-6600 Saarbrucken, West Germany

ANTONIO RACCIU, Corso Re Umberto,10, 10121 Torino, Italy

DR S RAFAILIDIS, c/o Department of Chemical, Process Engineering, University of Surrey, Guildford, Surrey, GU2 5XH

A RAMESH, Sneha Complex, 71/1 Miller Road, Centre for Dev. of Telematics, Bangalore 560052, India

A E RANTALA, Tech Research Centre of Finland, Lab for Info. Proc., Lehtisaarentic 2A, SF-00340, Helsinki, Finland

A RATTER, Robert Gordons Institute of Tech, School of Elect, RGIT, Schoolhill, Aberdeen, AB9 1FR

CERI REES, GEC-Marconi Research, Great Baddow, Chelmsford, CM2 8HN

MATTEO SONZA REORDA, Politecnico di Torino, Dip. Automatica e Inform, C.SO Duca degli Abruzzi 24, 10129 Torino, Italy

LARS RIEMER, Gudrunsvej 3, DK-2920 charl, Denmark

DARIO RINGACH, Israel Institute of Technology, Dept. of Elect. Eng., Box 59, 32000 Haifa, Israel

DR LUIGI RIZZO, Istituto di Elet. e Tele., Universita di Pisa, via Diotisalvi, 2, 56126 Pisa, Italy

MR DAVID A ROBINSON, P O Box 11871, Honolulu, Hawaii 96813-0871, USA

MR GRAHAM C ROBINSON, Cleveland ITEC, Sun Alliance House, 16-22 Albert Road, Middlesborough, Cleveland, TS1 1PR

KEVIN ROGERS, Merino Wool Harvesting Pty Ltd, Innovation House, Technology Park, Pooraka, South Australia 5095, Australia

JOHN J ROOD, Matheware Company, 23 Saunders Road, Norwood, MA 02062, USA

P R ROOZENDAAL, Postbus 421, Room C131 - RNL TWS, 2260 AK Leidschendom, Nederland

MR M ROPER, British Aerospace Dynamics, FPC 84, P O Box 5, Filton, Bristol, BS12 7QW

GARY RUMAIN, 102 Morris Drive, APT 101, Laurel, MD 20707, USA

M SABBATINI, ESTEC - WGSG, European Space Agency, Keplerlaan 1, 2201 AZ, Holland

DR A B SAHA, C-DAC, Udyog Bhavan, Ganeshkhind Road, Pune 411 007, India

THOMAS D G SANDFORD, Rolls Royce PLC F, P O Box 31, Derby, DE2 8BJ

NEELAM RAVI SANKAR, CDOT, 71/1 Millers Road, Bangalore - 560052, India

DIPL. ING. MARTIN SAUR, ibso GmbH, Fahrenheit Strasse 1, 28 Bremen 33, 0421/2208130, West Germany

STEFAN SAXER, Grossenbacher Elektronik AG, Elestastrasse, CH-7310 Bad Ragaz, Switzerland

I SAYERS, Newcastle University, Dept Electrical Eng, Merz Court, Newcastle, NE1 7RU

NAN SCHALLER, Rochester Inst. of Technology, School of Computer Science, 1 Lomb Memorial Drive, Rochester, NY 14623-0887, USA

IR J SCHOLTEN, University of Twente, Dept. of Informatics I/TH213, P O Box 217, 7500 AE Enschede, The Netherlands

IR KEES SCHUERMAN, ECRC, Arabellastrasse 17, D-2000 Muenchen 81, Germany

DR ANSELM SCHUSTER, Perimos, Schillerstrasse 18, D 7900 Ulm, Germany

PETER SCHWAB, Lab. for High Energy Physics, Sidlerstr. 5, CH-3012 Bern, Switzerland

HENRI SEYDOUX, B S C A, 6 rue Lockroy, 75011 Paris, France

DR P M SHARKEY, Dept. Engineering Science, Parks Road, Oxford, OX1 3PJ

MR ASHOK K SHARMA, C-DAC, Udyog Bhavan, Ganeshkhind Road, Pune 411 007, India

CAPT M M SHARMA, C-DAC, Udyog Bhavan, Ganesh Khind Road, Pune 411 007, Pune, India

MR MIKE SHARROTT, Engineering Research Group, Smiths Industries, Winchester Road, Basingstoke, Hants, RG22 6HP

S K SHRIVASTAVA, Computing Laboratory, University of Newcastle, Claremont Tower, Claremont Road, Newcastle-upon-Tyne, NE1 7RU

HOLLANDSE SIGNAALAPPARATEN B.V., Department OA-BS, H.G.W. Morsink, P O Box 42, 7550 GD Hengelo, The Netherlands

DAVID SINCLAIR, Dataproducts (Dublin), Clonshaugh Industrial Estate, Dublin 17, Ireland

STEVE SMERKER, 900 N. Riedel, Fullerton, CA 92631, USA

AVRIL SMITH, Oxford Polytechnic, Dept of Comp. & Math. Sci., Gipsy Lane, Headington, Oxford OX3 0BP

E R Smith, Mathematics Department, L A Trobe University, Bundoora, Victoria 3083, Australia

L G Smith, GEC Measurements Ltd, Dept 427, St Leonards Ave, Stafford, ST17 4LX

N D B Smith, British Telecom, Research, logy, BT 4244 Room 80 MLB 2, BTRL, Martlesham Heath, Ipswich IP5 7RE

Antonio Augusto de Sousa, Inesg (Norte), Largo Mompilher 22, 4000 Porto, Portugal

Harold De Souza, Dept of Elect Eng, University of Nairobi, P O Box 30197, Nairobi, Kenya

Norbert Speda, Gesellschaft fuer Prozessautomat, Gruberstrasse 46b, 8011 Poing, Germany

R P Stallard, Racal Milgo Ltd, Bartley House, Station Road, Hook, Herts, RG27 9PE

Jan Stappenbelt, Hoofdstraat 158, 7921 AS Zuidwolde, Holland

Karkanis A Stavros, 45 Hymettus, Holargos, Athens 15561, Greece

Mr J Steel, Ground Floor, Davis House, Oasis Park, Eynsham, OX8 1UT

Fred Steiner, Sentibuehlstr. 53, CH-6045 Meggen, Switzerland

George Stemmler, Jaegerstr. 22, 6653 Blieskastel, F R Germany

Roland Stern, c/o Contraves AG, Dept ETS-5, Schaffhauser Str. 580, 8052 Zurich, Switzerland

Alan Stevens, Transport and Road Research Lab., Old Wokingham Road, Crowthorne, Berks, RG11 6AU

Mr Ian Stewart, Plessey Research Ltd, Roke Manor, Romsey, Hants, SO51 0ZN

Paolo Stofella, A.C.S. S.r.l., Via Rombon 11, 00134 Milano, Italy

Mr Mark Stoker, Research Dept, Middlesex Polytechnic, Bounds Green Road, London, N11 2NQ

Paul Strickland, Portsmouth Polytechnic, Elect. and Elect. Eng. Dept., Anglesey Building, Anglesey Road, Portsmouth, PO1 3DJ

Jan Stubstad, MOVIX, Automatic Systems Des. ANS, P O Box 30, 4301 Sandnes, Norway

David Suter, Dept. Comp. Science, La Trobe University, Bundoora 3083, Australia

A M R Sutton, B.T.I., Goonhilly Downs, Helston, Cornwall, TR12 6LQ

Miroslaw Szturmowicz, Institute of Computer Science, Polish Academy of Sciences, 00-901 Warsaw, P O Box 22, Poland

Francis Pak-Twan Tam, Lecturer, City Polytechnic of Hong Kong, Dept of Electronic Engineering, Argyle Centre, Tower II, Mongkok Kowloon Hong Kong

Simon Taylor, Dept of Computer Science, Leeds Polytechnic, The Grange, Beckett Park, Leeds, LS6 3QX

Dr Steven Temple, Gnome Computers Ltd, 16 Histon Road, Cambridge, CB4 3LE

Nguyen Thanh-Son, School of Computer Science, Polytechnic of Central London, 115 New Cavendish Street, London, W1M 8JS

Dr R Thomas, Didas Entwicklungs GmbH, Schatzbogen 50, 8000 Munchen 82, Germany

B Thurnherr, Bodenacherstrasse 76, CH 8121 Benglen, Switzerland

Tom Tollenaere, Catholic University of Leuven, Neuro and Physchofysiology Lab, Campus Gasthuisberg, Herestraat, B3000 Leuven, Belgium

Mr Touman, IBM UK, 3-9 Leazes Park Road, Newcastle-upon-Tyne, NE1

David Tracey, Data Products Dublin, Clonshaugh Ind. Est., Dublin 17, Ireland

Mr S C Tran, Automotive Electronics Dept., T & N Technology Ltd, Cawston House, Cawston Lane, Cawston, Rugby, Warwickshire CV22 7SA

Robert Tredidgo, Dept of Elect. Syst. Eng., University of Essex, Wivenhoe Park, Colchester

Luuk Trip, Trip Advies, Postbus 133, 3730 AC De Bilt, Holland

Alan Turnbull, 65 Vicarage Road, Cale Green, Stockport, SK3 8HL

A M Tyrrell, Senior Lecturer, Coventry Polytechnic, Dept of EES, Priory Street, Coventry, CV1 5FB

Thomas A Ubriaco, ITT/Avionics, 390 Washington Avenue, Nutley, NJ 07110, USA

Jacques Vautherin, L R I, bat. 490 Universite Paris-sud, 91405 Osray, France

Mr P Verity, c/o British Steel, Brinsworth Strip Mills, P O Box 69, Sheffield Road, Rotherham, S60 1SZ

J C Vermeulen, NIKHEF-H, P O Box 41882, 1009 DB Amsterdam, Netherlands

J S Vertegaal, Stork X-CEL B.V., AFD Jet Printing, Sd30 AC Boxmeer, The Netherlands

Paul R M C Verzele, State University Gent, Sem en Lab voor Informatica, Campus Ardoyen, Grote Steenweg noord 2, B-9710 Gent, Belgium

Sjur Vestli, Postgraduate Student, Dept of Electrical Engineering, University of Salford, Salford, M5 4WT

Martorana Vincenzo, Bagheria 90011, Via Mole CC.PP.SC.6, Italy

Mag. Hartwig Vogel, HTL Rankweil, Negrellistr. 50, 6830 Rankweil, Austria

Christian Voirol, c/o CSEM, Maladiere 71, 2007 Neuchatel/CH, Switzerland

Thor Vollset, Miros A/S, PB 364, N-1371 Asker, Norway

Archie Waddell, Stuart Electronics, Bothwell Road, Castlehill Carlute, Strathclyde, ML8 5UF

Mr J F Waddington, MOD (PE), Rarde, Chobham Lane, Chertsey, Surrey, KT16 0EE

Mark Wadey, Thorn EMI Electronics, Wookey Hole Road, Wells, Somerset, BA5 1AA

Eyal Waldman, Electrical Eng. Department, Technion City, Haifa 32000, Israel

A W Walker, Dept 431, Marconi Instruments Ltd, P O Box 10, Six Hillsway, Stevenage, Herts SG1 2AN

Dalan Wang, Dept of Physics, Aberdeen University, Aberdeen, AB9 2UE

Kuanshen H Wang, Lockheed EMSCO, 2400 Nasa Rd 1, C-18, Houston, TX 77048, USA

Simon Wardrop, Magdalen College, Oxford, OX1 4AU

Gavin A Warren, 2 Holts Meadow, Redbourn, Herts, AL3 7BW

Etsuo Watanabe, Matsushita Electric Industrial, Components Department, 26th Floor World Trading Center, 4-1 Hamamatsu-cho, 2-chome, Minato-ku, Tokyo 105, Japan

Mark Webb-Johnson, Houston Crest Co., Ltd, Unit C4, Star House 2/F, I, Tsim Sha Tsui, Kowloon, Hong Kong

Mitch Webster, Electrical Engineering Dept, University of Natal, King George 5th Avenue, Durban 4000, South Africa

Ralf Wehrmann, Konradin Strasse 16, D8000 Muenchen 90, West Germany

Roman Weilguny, 1100 Wien, Herzg 47, Austria

Prof Dr Ralph Weissel, Sudetenweg 26, D2150 Buxtehude, Fed Rep Germany

John Wexler, Edinburgh University, James Clerk Maxwell Building, The Kings Buildings, Mayfield Road, Edinburgh, EH9 3JZ

Mr K R Whight, Philips Research Laboratories, Cross Oak Lane, Redhill, Surrey, RH1 5HA

Matthew Whillock, MRC, Molecular Genetics Unit, Mills Road, Cambridge, CB2 2QH

F R Whiteley, Wellcome Foundation PLC, Langley Court, Beckenham, Kent, BR3 3BS

Sue Whittle, Elec Eng Dept, Bolton Institute of Higher Edu., Deane Road, Bolton, BL3 5AB

Dr Francis Wong, Institute of Systems Science, University of Singapore, Singapore 0511

M G Wood, 7 Shawhurst Lane, Drakes Cross, Wythall, B47 5HL

Michael Woodacre, INMOS Limited, 1000 Aztec West, Almondsbury, Bristol, BS12 4SQ

Mr C Woodhouse, Infrared Engineering Ltd, Galliford Road, The Causeway, Maldon, Essex, CM9 7XD

TIM WORTLEY, Dept of Elect & Elect Eng, Heriot Watt University, 31-35 Grassmarket, Edinburgh, EH1 2HT

JOOSEN WOUTER, K.U.L. Dept. of Computer Science, Celestynemlaam 200A, 3030 Leuven, Belgium

CHRIS WRIGHT, INMOS Limited, 1000 Aztec West, Almondsbury, Bristol, BS12 4SQ

MARK B WRIGHT, c/o Computing Devices Co, M S 175, P O Box 8508, Ottawa Ont. C, Canada, K1G 3M9

DR B WURDENWEBER, Technische Datenverartbeitung, Hella KG, Postfach 2840, D-478 Lippstadt, W Germany

SHEILA WYLIE, Rapid Silicon, Denmark St, High Wycombe, Bucks, HP11 2ER

MR MARTTI YLIOJA, Pormestarinrinne 2 B 22, SF-00160 Helsinki, Finland

HUANG YONGGEN, The School of Computer Science, Polytechnic of Central London, 115 New Cavendish Street, London, W1M 8JS

HENRY E YOUNG, William Penney Laboratory, Dept. of Computing, Imperial College, 180 Queens Gate, London, SW7 2BZ

PHILIPPA YOUNG, NUTIS (NERC), Dept of Geography, University of Reading, Whiteknights, Reading, RG6 2AB

DR ROBERT YVES, Lab. Tim 3, INPG, 6 av. Felix Viallet, F-38031 Grenoble, Cedex, France

JURG ZBINDEN, c/o Contreves AG ETS1, Schaffhausen str. 580, 8052 Zurich, Switzerland

KLAUS ZEPPENFELD, Universitat Osnabruck, Fachbereich Matem./Informatik, Albrechtstr. 28, D-4500 Osnabruck, Germany

GISELA ZIEGLER, Steinhofstr.17, 7500 Karlsruhe 41

MICHAEL ZIELENBACH, Schlehdorn weg 44, 59, Siegen, West Germany

ROMAN F ZUNKER, Dept. U9321, Siemens UB-KWU, Hammerbacher Str. 12, 8520 Erlangen, Germany

CONTACTS FOR RELATED GROUPS

Australian Transputer and Occam User Group

In Australia, the Users Group is still at an embryonic stage, and at the July meeting, intend to formalise a Committee etc. The person to contact is:

John Hulskamp
Department of Communication
and Electrical Engineering
Royal Melbourne Institute of Technology
GPO Box 2476V
Melbourne 3001 Australia

Tel: +61 3 660 2453
Fax: +61 3 662 1060
rcojh@au.oz.rmit.gecko

French Transputer Users Working Group

Traian Muntean
IMAG-LGI
b.p. 68
38402 St Martin d'Heres CEDEX
France

traian@fr.imag.imag

Deutschen Occam-Interessengemeinschaft der Transputeranwender

DO IT can be contacted through its secretary:
Heinz Ebert
Im Heidigen 3
5206 Neunkirchen-Seelscheid 2
West Germany

The president is:
Joachim Stender
℅ Brainware GmbH
Gustav-Meyer-Allee 25
1000 Berlin 65
West Germany

and vice presidents are:
Frank Heinemann
℅ Fraunhofer-Institute
Kleiststraße 23–26
1000 Berlin 30
West Germany

Peter Eckelmann
℅ INMOS GmbH
Danziger Straße 2
8057 Eching b. München
West Germany
+49 89 319 10 28

Occam User Group Japan

Contact the Secretary:
Mr Kazuto Matsui
INMOS Japan KK
4th Floor, Nº 1 Kowa Building
11-41, Akasaka 1-chome
Minato-ku, Tokyo 107
Japan
Tel: +81 3 505 2840
Fax: +81 3 505 2844

The chairman is:
Prof. Tosiyasu L. Kunii
Department of Information Science
University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo 113
Japan
+81 3 505 2840

New Zealand Transputer Users' Group

The NZTUG is still only a small organisation. The Chairman in Bob Hogson, Professor of Production Technology at Massey University. Contact the secretary and treasurer:

Dr Ian Graham
Department of Computer Science
University of Waikato
Private Bag
Hamilton
New Zealand

Tel: +64 71562889 x8204
Fax: +64 71384066
CSNet: i.graham@waikato.ac.nz
JANet: i.graham@nz.ac.waikato

North American Transputer Users Group

NATUG have finally (their word) formed a permanent organization; Dyke Stiles will be serving as the chair of a committee of about fifteen members. INMOS (Colorado Springs) will be providing secretarial support. Contacts for this group are: the Chair, Dyke Stiles; the Secretary of the North American Transputer Users Group, care of Mark Hopkins at INMOS Colorado; and the local agent for newlsetter submissions, who is Lyle Bingham. Their addresses appear on page 86.

SPECIAL INTEREST GROUP CHAIRMEN

Artificial Intelligence

Steven Ericsson Zenith
INMOS Limited
1000 Aztec West
Almondsbury
Bristol BS12 4SQ
+44 454 616616
zenith@uk.co.inmos

Environments

Formerly 'Operating Systems' and 'Unix'

Gordon Manson
Department of Computer Science
University of Sheffield
Sheffield S10 2TN
+44 742 768555 x5580

Formal methods

Bob Stallard +44 25 672 3911
Racal Milgo Ltd
Bartley House
Station Road
Hook
Hants RG29 9PE

Graphics and image processing

George Staniewicz +44 73 085 876
Weirlord Ltd
The Meeting House
High Street
South Harting
Hants GU31 5QB

Hardware

Denis Nicole
University of Southampton
Department of Electronics
and Computer Science
The University
Highfield
Southampton SO9 5NH
+44 703 787167
dan@uk.ac.soton.ecs

Image processing and vision

Neil Carmichael
KSEPL – Shell Research
Volmerlaan 6
2288 GD Rijswijk ZH
The Netherlands
Tel: +31 70 11 39 11
Fax: +31 70 11 39 10

Education and training

Formerly known as 'Learning'

Roger Peel
Department of Electrical Engineering
University of Surrey
Guildford
Surrey GU2 5XH
+44 483 509284
roger@uk.ac.surrey.ee

Networks

This group has been disbanded

Numerical methods

Derek Paddon
Department of Computer Science
University of Bristol
University Walk
Bristol BS8 1TR
+44 272 303030 x4336
derek@uk.ac.bristol.compsci

NATUG STEERING COMMITTEE

Dyke Stiles
Electrical Engineering Department
Utah State University
Logan, UT 84322-4120
Chair
+1 801 750 2806
Stiles@USU.Bitnet
Stiles@CC.USU.edu

Mark Hopkins
INMOS Corporation
PO Box 16000
Colorado Springs, CO 80935-6000
Secretary
+1 719 630 4000

Lyle Bingham
Computer Systems Architects
950 N. University Avenue
Provo, UT 84604
Newsletter contributions
+1 801 374 2300
csa@adam.byu.edu

Jim Favenesi +1 205 837 5282
c/o SPARTA, Inc.
4901 Corporate Drive
Huntsville, AL 35805

Jim Newhouse
FMC Advanced Systems Center
1300 South Second Street
Minneapolis, MN
+1 612 337 3242

David L. Fielding
Cornell University
265 Olin Hall
Ithaca, NY 14853
+1 607 255 8686
fielding@tcgould.tn.cornell.edu

Colin Whitby-Strevens
INMOS Limited
1000, Aztec West
Almondsbury
Bristol BS12 4SQ
United Kingdom
+44 454 611500
colin@inmos.co.uk

Linda Pollard +1 503 222 7080
Regis McKenna Inc.
220 NW 2nd, #1150
Portland, OR 97209

Gerald C. Johns
Computer Systems Laboratory
Washington University
724 S. Euclid Avenue
St. Louis, MO 63110
+1 314 362 3123
gerald@wuibc.wash.edu

Gordon Harp +44 684 894824
RSRE jgh@rsre.mod.uk
St Andrews Road
Great Malvern
Worcs WR14 3PS
United Kingdom

Ernest Miller
Computer Science Dept.
East Stroudsburg University
East Stroudsburg, PA 18301
+1 717 424 3447

John Board
Electrical Engineering Dept.
Duke University
Durham, NC 27706
+1 919 684 3123
jab@dukee.egr.duke.edu

Paul Smith
University of California at San Diego
Center for Research Language
CRL-C-008
La Jolla, CA 92093
+1 619 534 2695
Paul@amos.VESD.edu
PSSmith@UCSD.bitnet

Gerd Beckmann
Rensselaer Polytechnic Institute
Associate Director
Center for Manufacturing
110 8th Street
Troy, NY 12189
+1 518 276 6010

INFORMAL OCCAM USER GROUP COMMITTEE

Peter Welch
Computing Laboratory
The University
Canterbury
Kent CT2 7NF
Chairman
+44 227 764000 x3629
phw@uk.ac.ukc

André Bakkers
University of Twente
PB 217
7500 AE Enschede
The Netherlands
+31 53 892790
elbscbks@henut5.earn

Gordon Harp
RSRE
St Andrews Road
Great Malvern
Worcs WR14 3PS
0684 894824
jgh@uk.mod.rsre

Geraint Jones
Programming Research Group
Oxford University Computing Laboratory
11 Keble Road
Oxford OX1 3QD
Newsletter editor
+44 865 273851
geraint@uk.ac.oxford.prg

Jon Kerridge
Department of Computer Science
University of Sheffield
Sheffield S10 2TN
+44 742 768555 x5580
ac1jmk@uk.ac.sheff.primea

Derek Paddon
Department of Computer Science
University of Bristol
University Walk
Bristol BS8 1TR
+44 272 303030 x4336
derek@uk.ac.bristol.compsci

Roger Peel
Department of Electrical Engineering
University of Surrey
Guildford
Surrey GU2 5XH
+44 483 509284
roger@uk.ac.surrey.ee

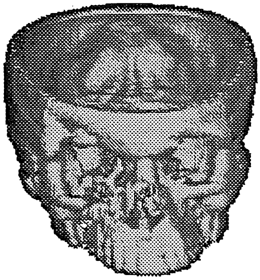
Michael Poole
INMOS Limited
1000 Aztec West
Almondsbury
Bristol BS12 4SQ
Secretary
+44 454 616616
oug@uk.co.inmos

Simon Turner
MEiKO Limited
650 Aztec West
Almondsbury
Bristol BS12 4SD
+44 454 616171

Colin Upstill
Plessey Electronic Systems Research Ltd
Roke Manor
Romsey
Hants SO5 0ZN
+44 794 515222

Hugh Webber
RSRE
St Andrews Road
Great Malvern
Worcs WR14 3PS
Program exchange
+44 684 89728
hcw@uk.mod.rsre

John Wexler
Edinburgh University Computing Service
The King's Buildings
Edinburgh EH9 3JZ
+44 31 667 1081 x2635
J.Wexler@uk.ac.edinburgh



... continued from front cover

Facilities to prepare this newsletter were provided by Oxford University Computing Laboratory; printed and distributed on behalf of the Occam User Group courtesy of INMOS Limited.