INTRODUCTION

The Department of Computer Science was created in 1966 when the then Computer Unit was sub-divided as a consequence of the Flowers Report on computing for the Universities and Research Councils. The Department inherited responsibility for teaching and research in Computer Science. There already existed a Postgraduate Diploma and a first-year course in the subject, together with a major research and development project, the Edinburgh Multi-Access Project, from which sprang the interactive computing system (EMAS) which provides the service now enjoyed by the University. The Flowers Report established major Computing Centres at Edinburgh, London and Manchester and the Edinburgh Regional Computing Centre was soon active in the role of providing service to the University and research institutes in the Edinburgh area.

As funds became available, the Department purchased its own small computers and the number of staff was increased from the original half-dozen. Second and third year courses were introduced in 1968 and 1970 - the latter made possible by the co-operation of the Mathematics Department in setting up a joint Honours school. At this time the Department moved to new accommodation in the James Clerk Maxwell Building, thereby acquiring the laboratory space required for the kind of teaching that it regarded as central to courses in Computer Science. By 1972 it became possible to launch a Final Honours year in the subject and an Honours degree in Computer Science alone became available. Joint degrees with Electronics, Management Science, Physics and Statistics are also now available. Undergrduate teaching was expanded in 1980 to provide a general first-year course Information Systems. In 1978 an M.Sc. course in Computer Systems Engineering was started and this is now part of the modular M.Sc. programme within the School of Information Technology.

Since then, teaching effort has been devoted to improving the structure and content of the course, not least as a result of the feedback from the first contingents of students to go through it. The Department has been fortunate in the level of support it has been able to achieve and the laboratory facilities have been continuously upgraded.

On the research side, the project on multi-access systems came to an end and was largely replaced by work on software for mini-computers. Early work on Computer-Aided Design has now been succeeded by a group actively working on design methodology for Very Large Scale Integrated (VLSI) circuits. In the last few years, work in the field of Theory of Computation has expanded and this is now a major research focus. Other areas include distributed systems, communications, graphics, advanced personal machines, databases and micro- processor control.

The Department is housed in the James Clerk Maxwell Building at The King's Buildings, which is the Science campus at Edinburgh University situated about two miles from the city centre. Few would claim that the appearance of the building is worthy of Edinburgh's architectural tradition, but it does provide extensive and modern accommodation for the six departments which occupy it, including lecture theatres, machine halls, a library and common-rooms.

Alistair Sinclair	Computational complexity: probabilistic models of computation.
Allen Stoughton	Semantics of concurrent languages.
Mads Tofte	Operational Semantics.
Rod Widdowson	VLSI design methodology.
Eric Wilson	Computational stylometry.

UNDERGRADUATE DEGREES IN COMPUTER SCIENCE

Directors of Studies: R. Candlin, A. Wight, F. Stacey

Several Honours degrees are available at Edinburgh; a single Honours degree in Computer Science, and five joint Honours degrees in which Computer Science is combined equally with another subject. The following degrees are offered:

Computer Science
Computer Science & Electronics
Computer Science & Management Science
Computer Science & Mathematics
Computer Science & Physics
Computer Science & Statistics

An Honours degree normally lasts four years. During each of the first two years, students take three subjects, of which one is Computer Science. Students who intend to take a joint degree also have to take prescribed courses relevant to that subject. There is usually the possibility of taking one or two "outside" subjects, which are not essential components of a given Honours course, but which give students the opportunity to broaden their interest. In their final two years, students follow courses in their chosen speciality.

The degree structure is very flexible, and students can to a certain extent keep their options open until the end of their first or second year, as far as their choice of degree is concerned.

For example, a student registered for a joint degree can easily change to a single Honours degree in one or other component by opting to do so at the beginning of the third year. Note however that it is not possible to switch from Computer Science and Management Science to a B.Com. degree in Social Science. Many students take the first and second year courses of Computer Science as outside subjects for other Honours degrees (for example in Engineering or Business Studies). In some cases, they become so interested in Computer Science that they decide to transfer to Computer Science Honours. This can usually be done quite easily, provided that Computer Science has been included as a subject from the first year.

Computer Science is often an important component of the Ordinary B.Sc. degree, which provides a three-year course for those who do not wish to specialise. A recently introduced compromise between the unspecialised Ordinary B.Sc. degree, and the highly specialised Honours degree is the Ordinary B.Sc. in a designated discipline. For this degree, students continue their study of Computer Science into their third year, by following selected parts of the full Honours course.

Overall course description

For their first two years, students spend one third of their time on Computer Science. Many students have had experience of working with computers at school, but there are many who have not, and the first year forms a broad introduction to the subject. From the very first, students are expected to obtain a practical experience of using a computer, which in this case is the University's large multi-access system, 2900 EMAS. In the second and subsequent years they have the opportunity of working with a variety of other computer systems ranging from the Department's VAX multi-access system, down through small mini-computers, to microprocessor systems. In the early years, students write software for existing hardware configurations; later they construct their own hardware from standard

components, and have the opportunity to design their own VLSI chips.

Throughout the course, emphasis is laid on the idea that a computer system is a fusion of hardware and software design. The fundamental ideas behind computers and computing can also be represented in mathematical terms, and this abstract and theoretical treatment forms an important part of the third and fourth year courses.

Assessment

There is a written examination in June each year, which is referred to as a "degree" examination. Passes in the first, second and third years are counted towards an ordinary B.Sc. The Honours degree is awarded on the basis of the third and fourth year examinations. In all these examinations, work carried out during the year contributes towards the assessment.

In the first three years there are also less formal "class" examinations during the year, which enable students to check their own progress.

Entry Requirements

Prospective students must satisfy the University's "general entrance requirement", which demands a certain level of achievement over a range of subjects. Details can be found in the University Undergraduate prospectus. For most Computer Science Honours courses, qualifications above the minimum are required. Each application is considered on its own merits, but as there are more applicants than places, a good standard of performance in Higher or A-level examinations has become necessary for an applicant to be offered a place. In the past, grades of AABB at Higher, or BBC at A-level have been required. There is a considerable amount of mathematically-based material taught in the third and fourth years, so we require a student to have gained a good grade in Mathematics at higher or A-level. Applicants for joint degrees must of course satisfy the requirements of the other subject as well. In exceptional circumstances it may be possible to admit candidates whose background does not match the above requirements.

Admissions

Candidates for undergraduate degrees must apply through the Universities Central Council on Admissions. The UCCA Handbook - 'How to Apply for Admission to a University' - and an application form may be obtained from schools or from:

The Secretary,
UCCA,
P.O. Box 28,
CHELTENHAM, GL50 1HY

Further enquiries about admission should be addressed to:

Faculty of Science Office (Admissions),
University of Edinburgh,
West Mains Road,
EDINBURGH EH9 3JZ

POSTGRADUATE STUDENTS - FULL-TIME Ph D/M Phil

Duncan Baillie	VLSI
Marek Bednarczyk	Logics of programming.
David Berry	Functional programming.
Gavin Brelstaff	Computer vision.
Simon Brown	Theory of computation.
Ilaria Castellani	Concurrent processes and their semantics.
Murray Cole	Architectures for concurrent language implementation.
Frank Cringle	VLSI design methodology.
Mads Dam	Theory of computation.
Bruce Davie	VLSI
Mark Davoren	Distributed computing systems.
Alex Deas	VLSI design methodology.
Tatsuya Hagino	Theory of computation.
Tim Hopkins	Computer architectures
Tom Horton	Stylometry of 16,17 and 18th century poetry and drama.
Martin Illsley	Programming methodology and correctness.
Claire Jones	Theory of computation.
Thomas Kean	VLSI.
Andreas Knobel	Logic and complexity.
Laurent Langlois	VLSI design methodology.
Kim Larsen	Theory of computation.
Gary Law	Computer architectures
Richard Marshall	The construction of silicon compilers.
George McCaskill	Incremental programming environment for VLSI.
Eugenio Moggi	Theory of computation.
Faron Moller	Theory of computation.
Ian Nixon	VLSI design methodology.
Brian Ritchie	Programming languages with concurrent processes.
Oliver Schoett	Theory of computation.

COMPUTER SCIENCE 1

Aims - The aims of the course are principally
-- To develop skill and knowledge in computer programming;
-- To understand and classify the algorithms (i.e. methods) which underlie all computer programs;
-- To group some of the great variety of structure present in the data handled by computer software;
-- To study the main hardware and software components which go to make up a typical computer system.

The main vehicle for this work is PASCAL. In the first half of the course this programming language is learned in depth and used as a medium for expressing solutions to non-trivial computing problems. Later the emphasis gradually becomes more abstract, more concerned with the structure and design of both the algorithms and the data objects which are expressible in the language..

Topics
Basic principles and Programming
Computer Systems
Data Structures
Programming Techniques and Other Languages

COMPUTER SCIENCE 2

There are three "fat" half courses available: Computer Systems in the first half of the year, Foundations and Real-Time Systems in the second half. Depending on the intended degree students take

Computer Systems + Foundations
or Computer Systems + Real-Time Systems

Some students may take all three half courses.

Computer Systems 2h - This half course continues the study of programming and computer architecture begun in Computer Science 1.
Topics - Programming in the applicative language ML, Compilers, Computer Graphics, Computer Systems.

Foundations 2h - An introduction to some of the basic mathematical ideas required for a formal treatment of computer hardware and software systems.
Topics - Mathematical Foundations, Automata, Semantics and Program Verification, Design and Analysis of Algorithms.

Real-Time Systems 2h - An introduction to the characteristics of real-time systems and techniques for implementing them. A study of the industrial application of computers for control and scheduling.
Topics - Real-Time programming, The Computer and its Environment, System Control, Industrial Applications.

COMPUTER SCIENCE 3

Introduction - The full-time third year course provides a basic foundation for the design and implementation of computer systems. The software oriented lectures concentrate on the overall design of software systems and their interaction with the underlying hardware. The hardware lectures reflect recent developments in methods of hardware implementation. Throughout the course the theoretical foundations of computation are examined.

Structure - The eight lecture modules each consist of 18 lectures. There are two major practical exercises associated with the other six modules. Joint Honours courses include a subset of the lecture modules and practical exercises. While the practicals are set in relation to the particular courses, students are encouraged to read as widely as possible and draw from their experience in other courses. Both examinations and coursework are assessed. A combined figure is carried forward to be used together with the final Honours year assessment in awarding the appropriate class of degree.

Lecture Modules

Computer Structures 1 - Detailed anatomy of computers. Microprogramming. Gate level primitives and logic design. Implementation methods including VLSI. Computer arithmetic. I/O mechanisms and bus organisation.

Computer Structures 2 - The influence of high level languages and systems on mainframe architecture. Evolution of the microprocessor. Asynchrony. The influence of technology on architectures. Cost and performance evaluation. The computer as an embedded component. Micro-computers in systems.

Formal Languages and Computability - Finite Automata and Regular Languages. Context Free Languages. Turing machines. Undecidability, the Halting and other problems. Simulation arguments.

Analysis of Algorithms - Recurrence relations. Sorting, merging and selection problems. NP-completeness. Fast algorithms for multiplying integers and polynomials. Discrete Fourier transform. Matrix problems, algorithms and reductions.

System Modelling - Probability. Queueing theory. Markov processes. Network models. Discrete-event simulation. Pseudo-random numbers. Statistical analysis of simulation results.

Programming Methodology - Program design. Formal and informal specification. The ML language. The ADA language and project.

Operating Systems - Concurrent processes and their synchronisation. Virtual addressing. Operating system kernels. Design and implementation of time- sharing multi-access systems. Single-user systems.

Database Systems - Roles, functional requirements and application design. Data models - relational, network and functional. Access structures. Query processing.

Major Practicals

Operating Systems. Microprogramming.

SECRETARIAL STAFF

Heather Carlin	Secretary to Head of Department (part-time)
Kate Duncan	Secretary to Professor Michaelson
Alison Fleming	Secretary to Professor Ibbett (part-time)
Eleanor Kerse	Secretary to Professor Burstall
Dorothy McKie	Secretary to Professor Milner
Margaret Melvin	Secretary/typist

TECHNICAL STAFF

Ian Conkey	Junior technician
John Dow	Laboratory Superintendent
Jimmy Johnstone	APM production
Peter Lindsay	Real-time and special systems
Ian Marr	Junior Technician
Michael Warburton	Communications installation and maintenance
Stuart Wasson	APM Servicing
Tom Wigham	VDU Servicing

RESEARCH STAFF

Neil Bergmann	VLSI
Pascal Bernard	Programming methodology (visitor for 1 year)
Bob Harper	Machine assisted proof
Kevin Mitchell	Machine assisted proof
Sassan Mohseni	VLSI verification
Andrew Morton	Stylistic Analysis
Janet Procter	VLSI
George Ross	Computer systems, graphics and personal machines
Hiroshi Sakuma	VLSI (visitor for 1 year)
Bob Tennent	Semantics of programming languages (visitor for 1 year)
Nigel Topham	Computer Architectures
Niklas Traub	Hardware verification
Xu-Sheng Zhang	VLSI

Gordon Plotkin	Reader. The denotational semantics of programming languages with emphasis on concurrency. Computational and inductive logic.
Rob Procter	Lecturer. Micro-computer systems. New technology for graphical devices. Social impact of computers.
David Rees	Senior Lecturer. VLSI deisgn and associated design tools. Design and implementation of multi-access operating systems.
Don Sannella	Lecturer (jointly with AI). Mechanised reasoning.
Peter Schofield	Senior Lecturer. Chairman of Department. Programming techniques. Data structures.
Frank Stacey	Lecturer. Systems software, combinatorial mathematics, parallel computation.
Colin Stirling	Lecturer. Computational theory.
Alex Wight	Lecturer. Computer performance evaluation. Computer systems modelling. Workload characterisation. Capacity planning. Network performance.

COMPUTING OFFICERS

David Baines	Software development and maintenance for real-time systems teaching.
John Butler	APM systems. Real-time systems teaching.
Ken Chisholm	Graphics and VLSI design tools.
George Cleland	Management of VAXes.
Lise Desjardins	Computing Support Officer
Archie Howitt	Hardware.
Kathy Humphry	First year teaching (part-time).
Fred King	Hardware design for APM. Hardware and software for very high performance personal computers.
Ann Macintosh	First year teaching. Micros and personal machines.
Man-Chi Pong	Software systems. Programming environments. Interactive computer graphics.
K.V.S.Prasad	Part-time
Alistair Scobie	Systems software.
Rainer Thonnes	Maintenance and development of communications and system software. CAD.

COMPUTER SCIENCE 4

Introduction- The final year honours course in Computer Science is intended to provide students with an opportunity to: add to the core curriculum of the first three years, study subjects from earlier years to greater depth and undertake a project involving a major implementation or research work.

Assessment - For full-time Computer Science students the following restrictions apply:

- VLSI Design is compulsory;
- One Theory of Computation course is compulsory;
- Digital Communications is compulsory;
- At least two and at most three other courses must be done.

Restrictions for joint degrees vary, depending on the other subject. Assessment is based on two equal parts, namely the best five results and the project. Honours are awarded on third and fourth year results.

Courses

Advanced Graphics - The course consists of a section of core material followed by a choice between two further sections. The core aims to teach fundamental techniques for generating line-drawings and realistic shaded images of three- dimensional scenes. One of the further sections extends this to include "ray-tracing" techniques colour and texture. The other section discusses software design of CAD systems and window managers.

Compiler Construction - The course deals with the problems of producing a production quality compiler. Groups of students work to produce an ALGOL60 compiler on the Amdahl EMAS service.

Computational Complexity - Machine-based complexity theory involves studying the resources, primarily time and space, required to solve problems using computing machines.

Computer Systems Performance Evaluation- The problems of measuring, predicting and improving the performance of computer systems. Tools, techniques, workload characterisation and capacity management.

Conceptual Modelling - Representing knowledge and capturing the meaning of data. Data abstractions. Conceptual models. Conceptual languages. Applications.

Denotational Semantics- Abstract syntax. Sets and semantic domains. The denotational format. Definitions using stores. Recursion and iteration. Environments. Procedures, functions and other compound data types. Continuations.

Digital Communications - The techniques used to implement networking, starting from transmission of data bits along physical connections and building up to the distribution of computations over many processors. Local and wide-area networks.

High Performance Computer Systems - Instruction formats. Pipelines. Instruction buffering. Parallel function units. Performance limitations. Vector processors. Multiprocessors.

The Theory of Communicating Systems - The course discusses the formalisation of languages for the specification of concurrent systems where several independent but interconnecting processes are active.

Very Large Scale Integrated Circuit Design - MOS devices and circuits. Integrated system fabrication. Data and control flow in systematic structures. Circuit topology. Patterning geometry. Wafer fabrication. Overview of an LSI computer system. Architecture and design of system controllers.

VLSI Circuit Design Practical- The goal of this practical course is the design of a digital subsystem using the VLSI design techniques introduced in the previous lecture course.

MEMBERS OF STAFF

TEACHING STAFF

Andrew Blake	Lecturer and Royal Society/IBM Research Fellow. Computer vision: description by computer of visible surfaces.
Gordon Brebner	Lecturer. Computational complexity. Parallel computation. Computer networks. VLSI layout algorithms.
Rod Burstall	Professor. Programming methodology: correctness proofs, program transformation, specification languages. Semantics using an algebraic/categorical approach.
Rosemary Candlin	Lecturer. Introductory teaching. Specialised micro-processor systems. Real-time systems.
Igor Hansen	Lecturer (part-time). Microcoded hardware description and analysis. Computer system architecture. Micro-programmable processors for graphics, communications and high level languages. Multiprocessor systems.
Matthew Hennessy	Lecturer. The mathematical semantics of programming languages and the design of program-oriented logical proof design.
Roland Ibbett	Professor. Computer architecture, local area computer networks.
Mark Jerrum	Lecturer. Complexity of computation: combinatorial algorithms, algebraic models of computation.
Kyriakos Kalorkoti	Lecturer. Complexity of computation, algebraic models of computation.
Clemens Lautemann	Lecturer. Complexity of computation, probabilistic algorithms.
David McCarty	Lecturer (jointly with Cognitive Science). Mathematical logic. Theories of computation.
Eric McKenzie	Lecturer. Computer architecture, VLSI and graphics.
Sidney Michaelson	Professor. The study of literary style with special reference to problems of authorship and chronology. Queue-related models of computing systems. Distributed computing systems.
George Milne	Lecturer. Formal models for the description of circuit behaviour.
Robin Milner	Professor. Semantics of programming languages. Application of mathematical logic to formalise the statement and proof of assertions concerning programs. Abstract models of concurrent computation.
Moira Norrie	Lecturer. Operating Systems. Database systems.

COMPUTING FACILITIES

The Department is rapidly moving to the point where the principal computing facility will be powerful personal work-stations with common filestores on an Ethernet-type communication network. The Department has designed and built a number of these systems and a number of commercial devices are also used.

The Advanced Personal Machine was designed in the Department and 50 are now in service in offices and public areas. Another 14 are being built. The machines are currently based on an M68000 processor and 2 megabytes of memory. Some are equipped with fast high-resolution graphics processors.

The workstations use central filestores rather than local disk storage. There are currently 3 of these filestores, also based around 68000 processors, providing close to 1 Gigabyte of file space.

Another substantial departmental computing facility is a VAX 11/780 housed in the Department's machine halls. This machine was installed at the end of 1978 and has since been upgraded to 4 Megabytes of main store and 1200 Megabytes of disk storage. It supports a maximum of around 40 simultaneous users under the VMS operating system and is connected to both the Departmental and ERCC networks. The main languages available are Pascal, IMP, ML and C.

Under a collaborative agreement with Gould Corporation the Edinburgh University School of Information Technology (the Departments of Artificial Intelligence, Computer Science and Electrical Engineering) have been given a large UNIX system to be used primarily for teaching.

Other machines include a VAX 11/750, 2 SUN 2/120, a Sun 3-75 and 5 PERQs all running Unix. There are also three high-quality graphics terminals based on PDP-11 processors.

The Department and ERCC were jointly concerned with the setting up of the micro-computer laboratory in the Appleton Tower of which the Information Systems course is a major user.

The Real-time systems laboratory is situated in the machine halls and is equipped with a variety of mechanisms and microprocessor prototyping kits which are used for experiments in automation. There is also a well-equipped electronics workshop which contains a selection of electronic measuring equipment and hand tools.

The computing facilities provide access to a variety of special-purpose peripherals including high quality laser document printers, plotters and two robot arms.

For first year teaching and certain specialised applications, the department also uses the main University computing service provided by the Edinburgh Regional Computing Centre. The principal mainframe facility is based around twin ICL 2976 systems linked with two ICL Distributed Array Processors (DAPs), two ICL 2988 systems and an IBM compatible Amdhal 470. all housed in the James Clerk Maxwell Building. These systems all run the Edinburgh Multi-Access System (EMAS) to support in excess of 200 simultaneous users at interactive terminals sited throughout the University and connected by a local network. This network which is one of the largest academic networks in the world, as well as providing access to 33 hosts from over 2,000 terminals provides other facilities like plotting and phototypesetting.

INFORMATION SYSTEMS 1

The motive in presenting this first level course is to increase the number of people in society who are well informed about computers. Decisions about the applications of computers cannot become or remain the prerogative of the computer scientists alone. It is important that future managers, politicians, lawyers, doctors, journalists etc, should be well equipped to make or influence such decisions. It is the aim of this course to nudge them into taking an interest in this field and initiating them into the acquisition of the relevant knowledge. Students from any faculty, in any year with any background (other than having already attended a Computer Science course) will be welcome.

Topics -
Computer Technology, Pascal Programming, Computer Operating Systems and Tools, Computers and Society, Principles of Information Retrieval, Graphics, Large Scale Systems, Management Information, Computing and the Professions, Natural Language and Speech, Expert Systems.

POSTGRADUATE STUDY

The Department offers facilities to study for the Research degrees of **M Phil** and **Ph D** and to participate in a course leading to an **M Sc** degree in Information Technology: Computer Systems Engineering.

Postgraduate Admissions - For further information or application forms, write to:-

Miss Eleanor Kerse,
Department of Computer science,
University of Edinburgh,
King's Buildings,
Edinburgh EH9 3JZ,
Scotland or phone (031) 667-1081, ext: 2782

RESEARCH DEGREES

For the Research degrees, the normal period of registration is two years for the **M Phil** and three years for the **Ph D**. Candidates are normally registered in the category of 'Supervised Postgraduate Student' for the first year of study, prior to transfer to the degree course considered appropriate, with back-dating of registration.

During their first year, postgraduate students are expected to participate in an appropriate study programme. For those interested in computational theory a specific course is provided. Students intending to pursue systems and other research may be directed to particular advanced courses, and are expected to participate in postgraduate study seminars. It is normally expected that, during their first year, students will focus their attention on one of the programmes of research being pursued within the Department. Some accommodation of interests is possible, but it is usually in the student's interest to be associated with one of the existing active research areas.

M Sc in INFORMATION TECHNOLOGY: COMPUTER SYSTEMS ENGINEERING

Introduction- This one-year postgraduate course consists of eight lecture modules examined by written examinations followed by a full-time project of six months duration examined by dissertation. A minimum of four lecture modules are chosen from those offered by the Computer Science Department. The remainder may be chosen from any of four departments: Computer Science, Electrical Engineering, Artificial Intelligence and Physics.

Course Modules

VLSI design (I and II) (two modules)
CAD tools for VLSI
Digital Communications
Programming techniques and software tools
Introduction to operating systems
Modelling and system performance
Database systems
Advanced graphics

In addition to the modules above the following modules are available from other Departments.

Department of Electrical Engineering-
Silicon wafer fabrication (I and II)
LSI circuit design (I and II)
Microprocessors
Test and reliability.

Department of Artificial Intelligence-
Artificial intelligence programming (I and II) (Prolog and LISP)
Natural language processing techniques
Expert systems
Machine vision
Knowledge Representation
Robot control.

Department of Physics -
Computer Architectures for Parallel Processing (jointly with Computer Science)

Part-time study - There are two possible schemes of part-time study spread over a period of up to three years.

CAD

The work in this theme is led by David Rees and the overall aim is towards Silicon Compilation. That is, systems which accept high-level descriptions of either the structure or the required behaviour of circuits and from them synthesize complete designs ready for fabrication. Considerable progress has already been made in this direction. Two such silicon compilers have appeared. The so-called 'First' silicon compiler was aimed at bit-serial signal processing applications (produced jointly with the EE department) and 'U2' which is aimed at nucleonic instrumentation applications. Work is also in progress at an intermediate level of the design process, that of textual language based composition systems which specify the floorplanning and layout of chips in such a way that composition can be verified for correctness.

Hardware Verification

Hardware verification is a research area of increasing importance due to the technological advances of VLSI fabrication and the resulting complexity of potential designs. Techniques are being developed which allow the correctness of a circuit design to be established by mathematical proof prior to fabrication. Central to this formal approach to circuit validation is a behavioural model in which to naturally and accurately represent the inherent concurrency of circuit behaviour.

Current research, funded by the Alvey Directorate(Software Engineering) explores the modelling capabilities of CIRCAL, a model of concurrent systems, both hardware and software, developed by George Milne. This has resulted in the discovery of a number of powerful validation techniques which allow both verification by proof and validation by simulation to be performed in a way which reflects the constructive, modelling philosophy adopted.

Behavioural Languages for VLSI Design

This project is funded by the Alvey Directorate(VLSI) and involves the collaboration of Edinburgh University (George Milne et al) with Ferranti Electronics Ltd and Lattice Logic Ltd.

New VLSI design languages are required to support the design of verifiably correct devices using both manual and automatic design techniques. Such languages should permit the expression of behaviour as well as structure with the design process being viewed as the iterative replacement of behaviour by structure. A hierarchical design methodology is adopted with every design step requiring to be shown to be correct; that is the design meets its specification.

Related research involves the design of correct silicon compilation algorithms where the correctness of the mapping algorithm ensures that the class of automatically generated designs are correct with respect to their source code specifications. This research builds on work done by George Milne and others at Edinburgh on models of concurrent computation.

SEMANTICS of ABSTRACT DATA TYPES

The aim of this project, carried out by Gordon Plotkin and his students is to develop further the application of Scott's theory of computation to the study of the synthetic approach to abstract data types. It is intended to pursue a wide variety of topics ranging from the detailed study of practical examples to theoretical problems and to include systematic comparisons with other approaches.

SEMANTICS of NON-DETERMINISTIC and CONCURRENT COMPUTATION

This research, which is being conducted by Robin Milner, Gordon Plotkin, Matthew Hennessy and Colin Stirling concerns the foundations of non-deterministic and concurrent computation. The aim is to provide a uniform framework containing mathematical models for the intuitive ideas of an event, of process communication and of synchronisation. The mathematics involved is continuous, as advocated by Scott, and uses tools from algebra and category theory.

STYLISTIC ANALYSIS

Stylometry has been defined as the scientific study of the usage of words in an attempt to resolve literary problems of authorship and chronology. The traditional methods of stylistic analysis have often been based upon subjective evaluation of internal textual evidence, often with unsatisfactory results. With the availability of computers as tools, new ideas have burgeoned and new approaches to these age-old problems have led to an increased need for statistical analysis of observational data. For example, it has now become practicable to study the usage of the most common words (such as determiners, conjunctions and prepositions) in a text: because of their number, such function-words have previously been neglected by literary scholars. Study of the positions of such words within the sentence has revealed that authors can be distinguished in this way. A new approach to problems of authorship and chronology is being brought into being by the study of such habits of composition.

Before any new technique is applied to disputed texts, it must be tested on texts of known provenance - as with other observational sciences. A wide range of texts is available for such testing, as well as an ever-growing collection of unanswered questions ranging from the composition of the Iliad to disputed wills in the U.S.A. Current work is concentrating on the thorny problems of authorship of Elizabethan and Jacobean drama and on the development of techniques for investigating poetry as well as prose.

This research into the methodology of answering such questions necessarily involves the development of further techniques and associated software. The amounts of data to be handled are often large and the processing poses interesting problems for the computer scientist.

This work is being carried out by Sidney Michaelson, Andrew Morton and two research students.

VERY LARGE SCALE INTEGRATION

The two main themes of activity are firstly Compter Aided Design (CAD) tools and secondly Formal Verification techniques. In addition, however, VLSI architectures for concurrent systems are attracting increasing interest and this may become a more significant aspect of the research in the future.

POSTGRADUATE STUDY PROGRAMME IN COMPUTATION THEORY

Outline and purpose of the programme

The Department of Computer Science at Edinburgh University offers a postgraduate study programme in the Theory of Computation, both pure and applied. In their first year, students attend an informal course of about 150 lectures, plus seminars, designed to give them a suitable grounding for research in this area. As the first year proceeds they are also guided towards a research topic.

These lectures are open to all, and a small number of one-year visitors and non-graduating students, who wish to attend but not to register for a degree may be accommodated.

Students require considerable mathematical training, for example an undergraduate degree in Mathematics, Mathematics and Computer Science or Mathematical Physics. Some computing experience is expected but a substantial knowledge of Computer Science is not a requirement.

The aim is to develop skill in applying mathematical ideas to computing problems, notably the proof of properties of programs, the quantitative analysis of algorithms, the complexity of computing tasks and the semantics of programming languages. Much progress has been made in the last ten years using ideas from mathematical logic, algebra, analysis, recursion theory, combinatorics and other branches of mathematics. This work is now beginning to affect methods of developing reliable and efficient software. Students will develop both mathematical understanding and practical programming skills.

Course Structure

The lectures, numbering about 150 in all, are given in the Autumn term and early Spring term, and are divided into three broad sections: Complexity, Program Methodology and Semantics. More advanced topics are covered in seminars in the second half of the Spring term. Theoretical and programming exercises are set in conjunction with the lectures. Most of the formal teaching is finished by Easter, to enable students to concentrate fully upon their research topics thereafter.

Topics -

Complexity- Analysis of algorithms. Computational complexity. Algorithmic graph theory. Research surveys.

Program Methodology - Program design. Formal tools for program development. Program logics and automatic deduction. Program specification. Prolog.

Semantics - Denotational semantics. Theory of communicating systems. Domains. Algebras and categories. Models of parallelism. Operational semantics.

RESEARCH

COMPUTATIONAL COMPLEXITY and ALGORITHMS

This research, carried out by Gordon Brebner, Mark Jerrum, Kyriakos Kalorkoti and Clemens Lautemann studies the fundamental limits on the resources such as time and space used by computations. Topics explored include a unified algebraic theory of the complexity of algebraic and combinatorial problems, general purpose parallel computers, algorithms with good probabilistic behaviour, and upper and lower bounds on problem complexity.

COMPUTING SYSTEMS,GRAPHICS and PERSONAL MACHINES

The Department has been active in basic system software - operating systems and language support - over many years. More recently it has built up expertise in the area of hardware design and implementation, particularly in relation to local area networks, micro-programmed controllers and high-speed processors. These strands have been drawn together in a Departmental project, now well advanced to implement a complete modular computer system. This provides substantial local computing capability but relies on a network for services such as filing and printing.

A major aim of this development has been to provide maximum flexibility of configuration, so that it would be possible within the overall framework to experiment with a variety of architectures and processors. Accordingly , the system is conctructed round a high-performance memory bus, shareable by several processors and permitting easy expansion of memory capability. The bus supports full 32-bit data operands and 32-bit (byte) addresses, with separate data and address lines.

What we are calling the Advanced Personal Machine is one version of this system. The basic version of the APM has a single processor board utilising the Motorola 68000 microprocessor chip, memory as required up to 8 megabytes and a network controller board providing access to an ethernet-type Local Area Network. Usually the system will be configured with a graphical processing capability, which provides bit-map graphics under the control of a micro-programmed controller. In due course the processing capability of the system will be enhanced by the provision of separate user-level processor boards, with the basic processor board retained as an input/output controller and diagnostic monitor. Because of the generality of the design, the system is not tied to one particular microprocessor range and plans for user level processors await the availability of 32-bit processors with satisfactory arrangements for supporting virtual memory. We shall also see the machine used for experimentation with novel architectures.

The operating system for the Advanced Personal Machine is aimed at matching the modularity of the hardware. It consists of a small core concerned with process creation and synchronisation, together with an open-ended set of facility modules, selectable at will according to configuration and user requirements. This approach permits a number of different user interfaces to be provided and its extension to provide support for complete alternative operating systems is now being investigated. Emphasis is being laid on designing the programming support invironment on a multi-lingual basis, at least to cover a family of broadly similar languages.

The distributed system now provides the major contribution to the Department's computing power and supports a large part of the teaching and research of the Department.

Work is in progress to create a Processing Resource Network to which a variety of novel processing devices will be connected. The network will contain a number of different technological components including Ethernet, Centrenet (a high performance LAN) and a contention switch, and will support a variety of protocols. Among the processors will be a sparse vector processing computer and a number of parallel processing systems as well as conventional machines acting as personal workstations and dedicated servers (e.g. for cross-compilation).

COMPUTER VISION

Computer vision research is concentrating on the theme of the "2 1/2 D sketch" - generation of descriptions of visible surfaces from stereo images. The aim is that 3D vision processes should be able to recognise objects by matching stored object-models to such surface descriptions.

The project is led by Andrew Blake and involves Andrew Zisserman and a Ph.D. student funded by a CASE award with IBM(UK). The work is part of a large multi-site ALVEY project in AI/IKBS vision. This project, which involves both industrial and academic collaborators, is investigating the interpretation by computer of stereo and moving images. Expected applications include automatic assembly and vehicle guidance systems.

MACHINE-ASSISTED PROOF

Previous projects designed LCF, a fully interactive proof system in which properties of computations (for example, of programs) may be rigorously verified by a mixture of automatic and interactive methods.

The most recent project, carried out under the direction of Robin Milner , focussed upon case-studies of proof. The main aim was to evaluate the LCF methodology, consisting principally of (1) the organisation of problems and problem areas in a hierarchic structure of theories, and (2) the use of a powerful metalanguage ML (a high-level programming language in its own right) to raise the quality of interaction by programming and combining partial proof strategies. Of particular interest were the verification of compilers and parsers, and establishing properties of useful abstract types. Completed studies are: the verification of a simple but non-trivial compiler, the modelling and analysis of Backus' FP systems, and the investigation of some abstract data structures (for example, binary search trees). Further projects are planned to continue this work.

PROGRAMMING METHODOLOGY

This project, carried out by Rod Burstall, Don Sannella and postgraduate students, is funded by the SERC. The main topics are :-

a) Specification of problems and development of programs. We have developed specification languages such as Clear and ASL. We have studied the techniques for formal development of programs from specifications so as to obtain programs which are guaranteed correct relative to the specification.

b) Advanced programming languages, with particular emphasis on functional programming and modular structure of large programs. We have worked on ML and Pebble, both strongly typed functional languages. ML is the intended vehicle for much of our research work and Pebble is a kernel language developed with Lampson of DEC System Research Center, Palo Alto.

c) Applications of algebra and category theory. We have studied applications to program specification and development categorical programming and the categorical formulation of algorithms.

UNIVERSITY OF EDINBURGH

DEPARTMENT OF COMPUTER SCIENCE

Departmental Handbook

November 1985