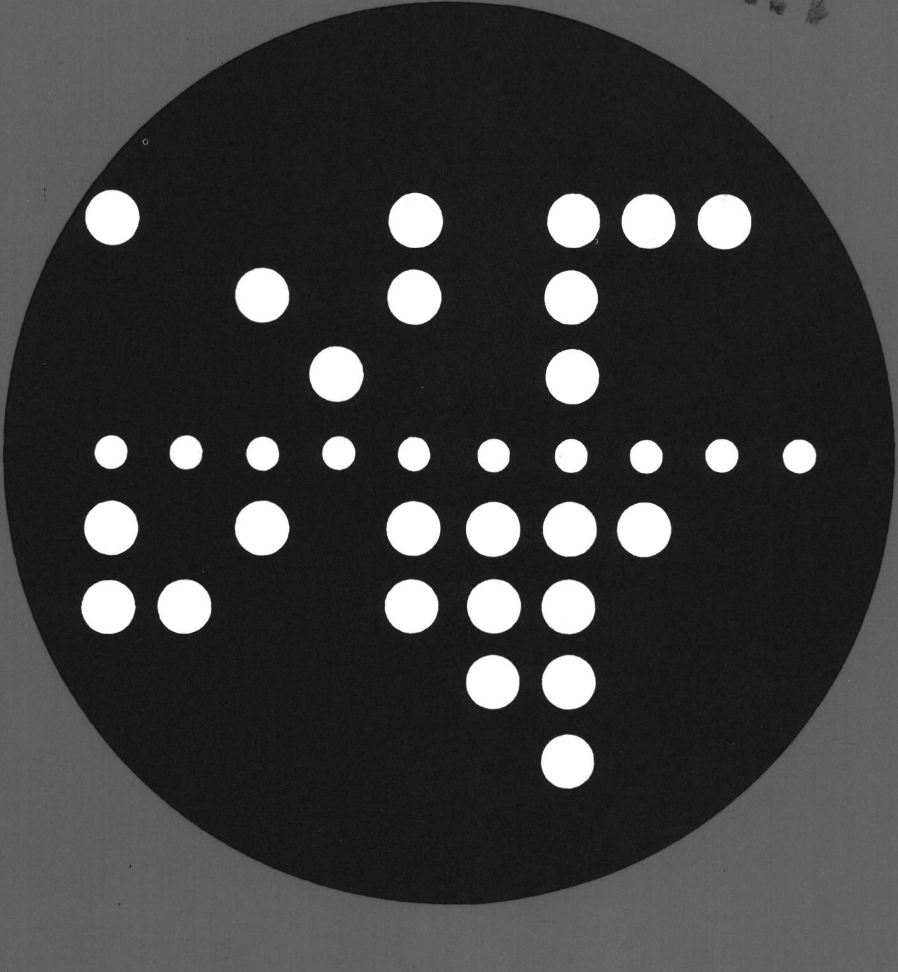
COMPUTING CENTRE NEWSLETTER

October 1981 · N. **55**



Commission of the European Communities

JOINT
RESEARCH
CENTRE

Ispra Establishment

CONTENTS

EDITORIAL NOTE

The Computing Centre Newsletter is published monthly except for August and December.

It describes developments, modifications and specific topics in relation to the use of the computing installations of the Joint Research Centre, Ispra Establishment.

The aim of the Newsletter is to provide information of importance to the users of the computing installations, in a form which is both interesting and readable.

The Newsletter also includes articles which are of intellectual and educational value in order to keep the users informed of new advances in computer science topics.

Administration and contact address:

Ms. A. Cambon (tel. 5446)
Support to Computing
Building 36
J.R.C. Ispra Establishment
21020-ISPRA (Varese)

LEGAL NOTICE:

Neither the Commission of the European Communities nor any person acting on behalf of the Commission is responsible for the use which might be made of the information in this Newsletter.

A NOTE ON VECTOR PROCESSORS

H. I. de Wolde

Introduction

This ariticle is mainly based on the material as presented at the Seminar:

"Efficient Fortran Techniques for Vector Processing"

This seminar was given by the Pacific Sierra Research Corporation in Bologna, 28 Sept.-2 Oct. 1981.

The scope of this article is only to give an illustration of the problems and the possibilities of the new developments in computer hardware.

The examples given refer commonly to the CRAY-1S and the CYBER 205 computers as these machines are the best known in this field.
This article is by no means a quality evaluation of the two computers but uses the information on this hardware only as an illustration to the applied techniques.

Improving Computing Power

There are many hardware approaches to improve the speed of scalar computers:

- Extensive use of high-speed registers, as the operands can be obtained from the registers much faster than from memory.

- Multiple functional units, which allow for a simultaneous execution of tasks: add, shift, multiply, logical operations, divide, population.

- Segmenting funtional units: for example an Add may be divided in four separate hardware functions which cover each just one clock cycle. In this way several Add's may overlap each other.

- Instruction word stack; by fetching groups of instructions from memory the executions of the command may be accelerated.

- Memory Banks; segmenting of the memory allows for the simultaneously loading of elements from different banks.

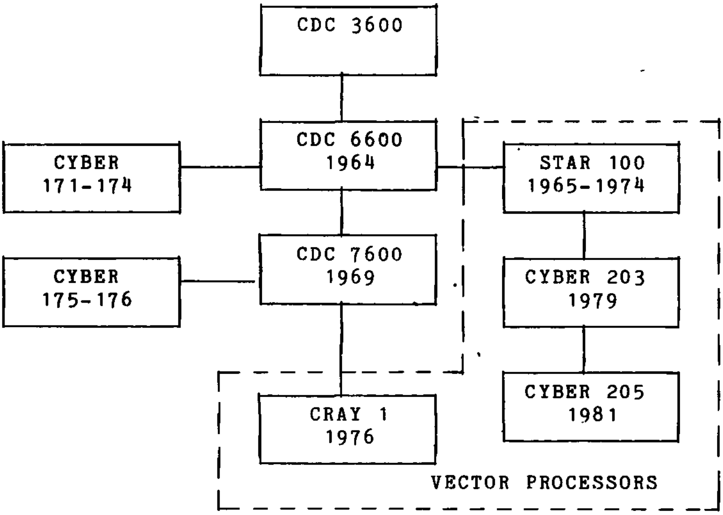
- Reducing the physical dimensions of the computer gives an improvement of the transfer rates.

The CRAY and the CYBER computers apply the listed techniques and, additionally, have the vectorization capability:

- Vector instructions perform many operations by just one single command.
  For example: A=B+C, in which A,B and C are one-dimensional arrays, require one instruction.
  This option allows also for a more effective use of the hardware enhancements already mentioned for scalar computers.

The CDC Computer Family

The next diagram gives the descent relations between the CDC computers and also the CRAY which may conceptionally be considered as also belonging to the CDC family.



The CDC 7600, CYBER 175 and 176 have some parallel functions, i.e. the nine functional units may be addressed simultaneously. The CYBER 205 is a vector processor with two completely separated CPU's; one for scalar operations and one for vector operations.

The characteristics of this family of processors are very different from IBM type computers.
To mention just a few of these differences:
- CDC does not have a byte structure.
- Disk organisations are totally incompatible.
- The Fortran evolution has produced offshoots with little respect for standardization. The conversion of the dialects is very difficult.
  CDC Fortran and IBM Fortran show more than seventy essential language incompatibilities.

The next table gives a summary of some technical data.

	CDC 7600	CYBER 203	CYBER 205	CRAY
Year	1969	1979	1981	1976
Cycle time in nanosec.	27.5	20	20	12.5
Result rate scalar MFLOPS	4.3	8-12	8-12	8-12
Result rate vector MFLOPS	(8)	37	50-200	80
Registers	24	256	256	144+8*64
Word size (bits)	60	64	64	64
Max. memory size	2M	2M +virtual	4M +virtual	4M

There exists an essential difference between the CRAY and the CYBER 203/205 in the way in which the vectorization is implemented.

The CRAY uses registers to load the arrays and to perform the operation. To execute a multiplication of two vectors, A and B, the elements are put into a series of vector registers. Eight registers of each 64 words are available.

```
V1 contains A1,A2,.....,An
V2 contains B1,B2,.....,Bn
```

After the one-instruction multiplication:

```
V3 contains C1,C2,.....,Cn
```

However if the vectors are longer than 64 elements, the operation has to be repeated for consecutive parcels of 64 elements.

The CYBER fetches the elements directly from memory which avoids the limitation of 64 words per instruction.
However, the vectorization of a DO-loop with non unit stride is not possible. These operations have to be performed in scalar mode:

```
   DO   10   I=1,N,2
10 C(I)=A(I)*B(I)
```

Multiple Functional Units

Functional units are hardware components dedicated to the execution of one specific operation.

The CRAY disposes of the following functional units:

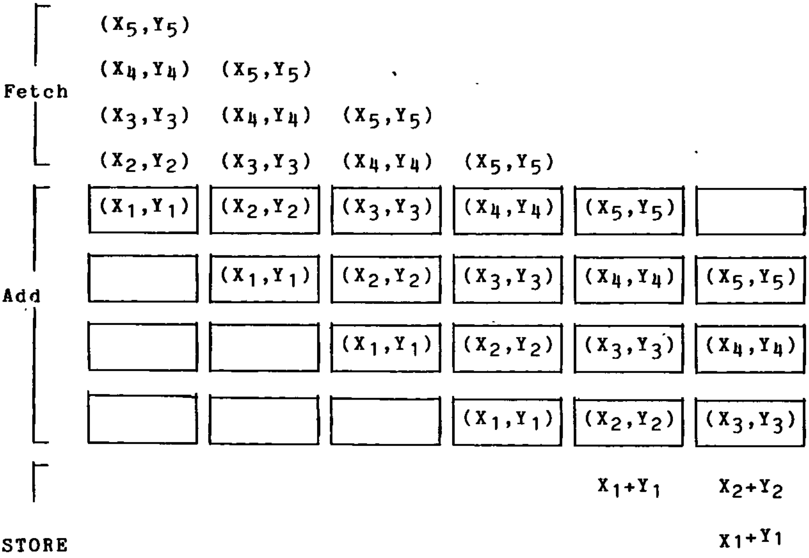
Scalar operations:
- Add
- Logical operations
- Shift
- Population and leading zero count

Scalar and vectorial operations:
- Add, floating point
- Multiply
- Reciprocal approximation

Vector operations:
- Add
- Logical
- Shift

The CYBER is in fact a composition of two independent processors; a scalar unit and a vector unit.
Both processor have their own set of functional units, being:
- Add
- Multiply
- Shift/Logical
- Divide/Square Root/Convert

On both computers the functional units are segmented to allow for overlapping of operations.
The following diagram gives the example of a sequence of Add operations for a functional unit which is composed of four segments.



Fetch	(X_5,Y_5)					
	(X_4,Y_4)	(X_5,Y_5)				
	(X_3,Y_3)	(X_4,Y_4)	(X_5,Y_5)			
	(X_2,Y_2)	(X_3,Y_3)	(X_4,Y_4)	(X_5,Y_5)		
Add	(X_1,Y_1)	(X_2,Y_2)	(X_3,Y_3)	(X_4,Y_4)	(X_5,Y_5)	
		(X_1,Y_1)	(X_2,Y_2)	(X_3,Y_3)	(X_4,Y_4)	(X_5,Y_5)
			(X_1,Y_1)	(X_2,Y_2)	(X_3,Y_3)	(X_4,Y_4)
				(X_1,Y_1)	(X_2,Y_2)	(X_3,Y_3)
					X_1+Y_1	X_2+Y_2
STORE						X_1+Y_1

The process proceeds one step at each clock cycle.
Although an addition of two values takes four clock cycles to be performed, the system delivers, after the start-up period of four cycles, the sum of two values on each clock cycle.

Performance estimates

It is difficult to express the power of a computer system in a concise format. Before the appearance of the vector processors the MIP expression was used to give an indication of the CPU speed:

1 MIPS = one million of instruction per second

However, since a single vector operation is the equivalent of many scalar instructions this constant is no longer a good measure of execution speed.
Now, instead of MIPS the MFLOPS is used:

1 MFLOPS = one million of floating point operations per second

It must be well understood that such specifications of speed is related to the CPU only. The fastest CPU will not improve the process time for an I/O bound program!

The following table gives an estimate for the calculation time of the so-called Sandia Kernels on some computers. These kernels are typical batch tasks.
The average figure gives a reasonable approximation for a general workload in a scientific environment.

The result rate is expressed in MFLOPS.

	IBM 3303	AMDAHL V7A	AMDAHL V8	CDC 7600	CYBER 203	CRAY
1 Linear equation solver	1.4	2.0	2.8	3.6	3.9	6.3
2 Ordinary differential equation solver	1.7	1.9	2.6	2.9	5.9	11.2
3 Vortex dynamics	2.0	2.2	2.9	3.1	5.3	5.4
4 Lattice relaxation	1.4	1.8	2.6	3.0	4.2	3.7
Average	1.6	2.0	2.7	3.2	4.9	6.6

Vectorprocessing

As has been explained, the techniques of increasing the speed of computers are manifold and vectorization is just one item on the list of options.
However, this technique includes serious consequences for the application programmer in contrast to the other mentioned hardware enhancements which normally are incorporated in a transparent way.
Only the operations in a DO-loop may be handled by the vectorial functional units.
However, many restrictions limit this capability as is shown in the next table.
If any of these situations occur in a DO-loop, the compilation product will define the operation as scalar.

vectorization limitations		
	CRAY	CYBER 205
CALL	-	-
IF	-	-
GO TO	-	-
I/O	-	-
non-linear array reference	-	-
indirect addressing	-	-
recursion	-	-
CII[1]) other than loop index	+	-
CII used before being set	+	-
CII used in expression	-	-
stride	+	- [2])
very long vectors	+	-
equivalences	+	-

+ the compiler accepts the function for vectorization
- the function is executed as scalar operation

[1]) CII = Constant Increment Integer, an integer value that has a constant value added to it at each loop cycle.
[2]) Stride may be incorporated by some particular routines.

The shown limitations are especially serious for those classes of programs which use extensive decision trees. For example Monte Carlo programs and reliability programs cannot be vectorized.

Optimization of Programs

Almost any hardware approach to increasing computer speed requires some adaption of Fortran semantics for an optimal effect.
However, this adaption is generally not a very essential interference, except for the optimization of the vector capabilities.
Here the Fortran programmer needs a good knowledge of the computer architecture to obtain the desired results.
Adaption of a Fortran program for a scalar computer to a vector processor is a highly specialized job.
The following case study may illustrate this.
A Westinghouse nuclear reactor model was very frequently used and required optimization.
After running with a time-scheduler it appeared that one particular routine used 90% of the CPU time. This routine contained a triple-nested 600 records DO-loop, recursive along all dimensions. The following steps were taken to speed up this routine:

- loop independent IF's were removed from inner loop
- smaller loops were formed with simple flow of control according to the requirements of the special compiler of the vector processor
- loop independent calculations were pulled outside

The total effort took 4 man-month and the result for the scalar computer was a CPU time reduction of 50%.
For a vector processor it was calculated that the code runs about ten times as fast.
This example shows that optimization of Fortran programs may have a large effect for a vector processor. However this means also that a program which is not written according to the special preferences of the compiler of the vectorprocessor will function extremely poor.
These compilers are much more sensitive for programming style than in the case of scalar computers.

A typical example of optimization of a DO-loop for the CRAY computer is given here:

```
      DO  100  I=1,N
      A(I) = (B(I)+C(I))*S
      D(I) = S*B(I)*C(I)
  100 E(I) = C(I)*(C(I)-B(I))
```

By overlapping of operations this piece of program may be accelerated at execution.

The rules for improving the speed by "chaining" are as follows:

- A Functional Unit can only appear once in a chain
  The Functional Units are: floating add, floating multiply, floating reciprocal, integer add, shift, boolean, memory access
- A specific operand may only appear once in the chain
- Fetch and store operations start a new chain

The required time is expressed in "chimes"= chained vector time. This means roughly the time to fetch or to store a vector and is thus only a relative unit depending on the length of the array.

The above mentioned piece of Fortran requires 7 chimes:

operation	chimes	
LOAD B	1	
LOAD C	1	
ADD B+C	0	overlapping
MULTIPLY S*(B+C)	0	"
STORE A	1	
MULTIPLY S*B	0	"
MULTIPLY C*S*B	1	
STORE D	1	
SUBTRACT C-B	0	"
MULTIPLY C*(C-B)	1	
STORE E	1	
Total	7 chimes	

Respecting the above mentioned rules for chaining, the program may be also written as:

```
       DO   100   I=1,N
       E(I) = C(I)*C(I)-B(I)*C(I)
       D(I) = B(I)*C(I)*S
   100 A(I) = (B(I)+C(I))*S
```

The applied modification is not obvious at all and can only be justified by the calculation of the chimes.
It is contrary to the logic of programming because, in the second statement, a subtraction + multiplication is replaced by a subtraction and two multiplications.
The account of the necessary time to complete the DO-loop shows an improvement of 40%.

Operation	Chimes	Vector register	
LOAD C	1	V_1	
MULTIPLY C*C	0	V_2	overlapping
LOAD B	1	V_3	
MULTIPLY B*C	0	V_4	"
SUBTRACT C*C-B*C	0		"
STORE E	1		
MULTIPLY B*C*S	0		"
STORE D	1		
ADD B+C	0	V_5	"
MULTIPLY (B+C)*S	0		"
STORE A	1		
Total	5 chimes		

Concluding, it may be stated that adapting programs for vector processors,

- is a tedious job which requires a sound knowledge of the computer architecture
- requires considerable manpower investment
- decreases the readibility of the programs
- may increase execution time for scalar computers.

USING THE NAG AND IMSL LIBRARIES

M. Dowell

A new "green book" entitled "Using the NAG & IMSL Libraries" has recently been produced by the staff of the Computing Centre. This document includes the following type of information:

. Why the mathematical Libraries should be used.

. How the relevant subroutines should be used on the TSO system and in batch jobs (with useful examples).

. How the relevant Library documentation should be used.

. Which Library should be used for a particular mathematical or statistical problem (with a comprehensive comparative analysis).

Copies of this document will shortly be circulated to all relevant registered holders of other green books. If you wish to obtain a copy of this document and you are not on the relevant mailing list then please complete and send the form at the back of this Newsletter.

Note.

The introduction of this "green book" makes the following Newsletter articles obsolete:

issue	title of article	author
31	The NAG Library	M. Dowell
38	The NAG Library is Available	M. Dowell
41	NAG Note	M. Dowell
42	IMSL Library - New Edition	M. Dowell
45	IMSL and NAG Libraries on TSO (Errata Corrige in N. 46)	M. Dowell
48	IMSL Edition 8	M. Dowell
50	NAG Library Mark 8	M. Dowell

STATISTICS OF COMPUTING INSTALLATION UTILIZATION.
REPORT OF COMPUTING INSTALLATION EXPLOITATION
FOR THE MONTH OF SEPTEMBER 1981.

	YEAR 1980	YEAR 1981
General		
Number of working days	22 d	22 d
Work hours from 8.00 to 24.00 for	16.00h	16.00h
Duration of scheduled maintenance	26.00h	20.34h
Duration of unexpected maintenance	18.17h	8.00h**
Total maintenance time	44.17h	28.34h
Total exploitation time	307.83h	323.66h
CPU time in problem mode	278.63h+	297.72h*
Batch Processing		
Number of jobs	8026	8320
Number of cards input	1355300	243590
Number of lines printed	28091000	27012000
Number of cards punched	231000	16430
CPU Time	247.95h+	234.62h*
Number of I/O (Disks)	25155000	25305200
Number of I/O (Magnetic tape)	3886600	5751300
T.S.O.		
Number of LOGON's	4403	5674
Number of messages sent by terminals	319500	374730
Number of messages received by terminals	2114500	2503630
CPU time	29.17h+	53.66h*
Number of I/O (Disk)	4309250	4469100
Connect time	2976.68h	4021.92h
ADABAS		
Total time service is available	-	194.98h
CPU time	-	6.01h*
Number of I/O (Disk)	-	849023
IMS		
Total time service is available	30.09h	136.16h
CPU time	0.16h+	3.43h*
Number of I/O (Disk)	88400	667960

+ Real CPU has been multiplied by a factor of 2.0 to indicate the increased throughput of the AMDAHL 470/V7A.

* Real CPU has been multiplied by a factor of 2.5 to indicate the increased throughput of the AMDAHL 470/V8, in comparison with the IBM 370/165.

** Covering all the configuration.

UTILIZATION OF COMPUTING CENTRE BY OBJECTIVES & APPROPRIATION ACCOUNTS FOR THE MONTH OF SEPTEMBER 1981.

		AMDAHL 470/V8 work units in hours
33001	Reactor Safety	303.06
33002	Plutonium Fuel and Actinide Research	-
33003	Safety of Nulcear Materials	4.07
33004	Fissile Materials Control and Management	17.73
33005	Super-SARA Test Programme SSTP	67.46
33011	Solar Energy	1.10
33012	Hydrogen Production, Energy Storage and Transport	4.98
33013	Thermonuclear Fusion Technology	20.05
33014	High Temperature Materials	2.05
33021	Protection of the Environment	23.27
33022	Remote Sensing from Space	4.92
33041	Informatics	39.85
33043	Support to the Community - Bureau of References	-
33044	Training and Education	-
33046	Provision of Scientific and Technical Services	17.51
1.20.1	General Administration - JRC	63.45
1.20.2	General Services - Administration - Ispra	
1.20.3	General Services - Technical - Ispra	0.46
1.30.0	Central Workshop Ispra	2.61
1.40.2	ESSOR	3.69
	TOTAL	576.26
1.94.0	Services to External Users	4.10
	TOTAL	580.36

BATCH PROCESSING DISTRIBUTED BY REQUESTED CORE MEMORY SIZE

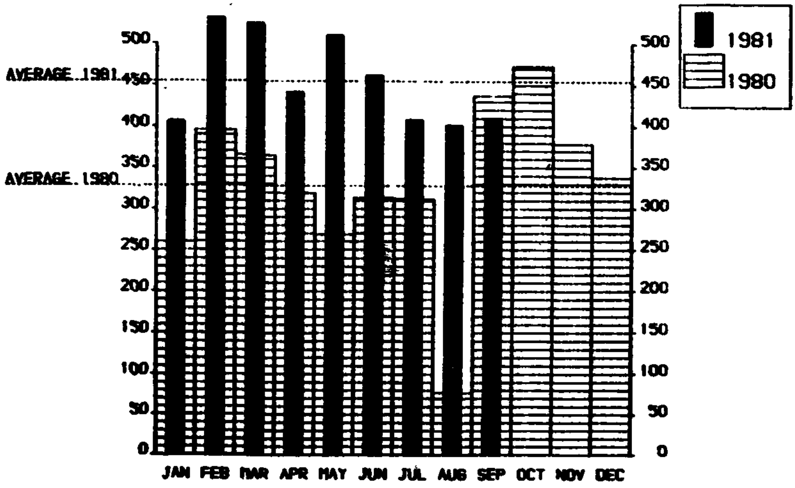
	100 k	200 k	300 k	400 k	600 k	800 k	1000 k	1200 k	1400 k	1400 > k
No. of jobs	2497	1724	1249	1079	665	306	83	21	26	66
Elapsed time	73	156	175	204	144	138	50	13	9	34
CPU time	7.7	27.2	32.6	23.0	33.8	47.3	30.7	4.2	4.3	23.2
"Equiv" time	24	48	70	67	59	68	35	6	4.7	26.4
"Turn" time	0.5	1.4	2.2	2.2	2.0	3.5	3.5	3.0	3.2	4.0
I/O (disk)	1670	2780	4594	5984	3078	2998	589	263	49	459
I/O tape	1637	415	1773	670	1165	14	32	-	-	-

NOTE.
All times are in hours.
"Equiv" means equivalent.
"Turn" means turn around.
All I/O transfers are measured in 1000's.

PERCENTAGE OF JOBS FINISHED IN LESS THAN:

TIME	15mn	30mn	1hr	2hrs	4hrs	8hrs	1day	2day	3day	6day
%year 1980	33	50	65	76	87	94	96	100	100	100
%year 1981	31	46	60	73	88	97	99	100	100	100

HISTOGRAM OF TOTAL EQUIVALENT TIME(HRS)



Projected total for 1981 = 5455 hours(using average)

Total for 1980 was = 3936 hours

REFERENCES TO THE PERSONNEL/FUNCTIONS OF THE COMPUTING CENTRE

Manager of the Computing Centre		J. Pire		
Responsible for User Registration	Ms.	G. Rambs		

Operations Sector

Responsible for the Computer Room		A. Binda-Rossetti
Substituted in case of absence by:		
Responsible for Peripherals		G. Nocera

Systems Software Sector

Responsible for the sector		D. König
Substituted in case of abscence by:		P.A. Moinil
Responsible for TSO Registration		C. Daolio

Informatics Support Sector

			Room	Tel.
Responsible for the Sector	(f.f.)	H. de Wolde	1883	5787
Secretary	Ms.	G. Hudry	1873	5787
Responsible for User Support		M. Dowell	1886	5419
General Inf./Support Library	Ms.	A. Cambon	1871	5446
Advisory Service /List of Consultants(See Note 1)			1870	5446

A. Inzaghi

R. Meelhuysen

A. A. Pollicini

H. I. de Wolde

M. Dowell

Note 1. The advisory service is available in the same room as the Computing Support Library (room 1870). Exact details of the advisory service times for a specific week can be found at the head of any output listing (for that week).
Any informatics problem may be raised. However, the service is not designed to help users with problems which are their sole responsability. For example, debugging of the logic of programs and requests for information which can easily be retrieved from available documentation.
If necessary, other competent personnel from the informatics division may be contacted by the consultant but not directly by the users.
The users should only contact the person who is the consultant for that specific day and only during the specific hours.
Outside the specific hours general information may be requested from Ms. A. Cambon in the Computing Support Library.

HOW TO OBTAIN COMPUTING CENTRE DOCUMENTATION

Person interested in receiving copies of the Computing Centre "green books" or in receiving regularly the "Computing Centre Newsletter" are requested to complete the appropriate part of the following form and send it to:

Ms. A. Cambon

Support to Computing

Building 36

Tel. 5446.

Indicate with a (✓) which option are required.

Please add my name to Newsletter mailing list ( )

Please send me copies of the following "green books":

JRC-TSO Primer	( )
JRC Computer Graphics (new version)	( )
Towards a New Programming Style	( )
LIBRARIAN	( )
Using the IMSL & NAG Libraries	( )

NAME ....

ADDRESS ....

....

....

TELEPHONE ....