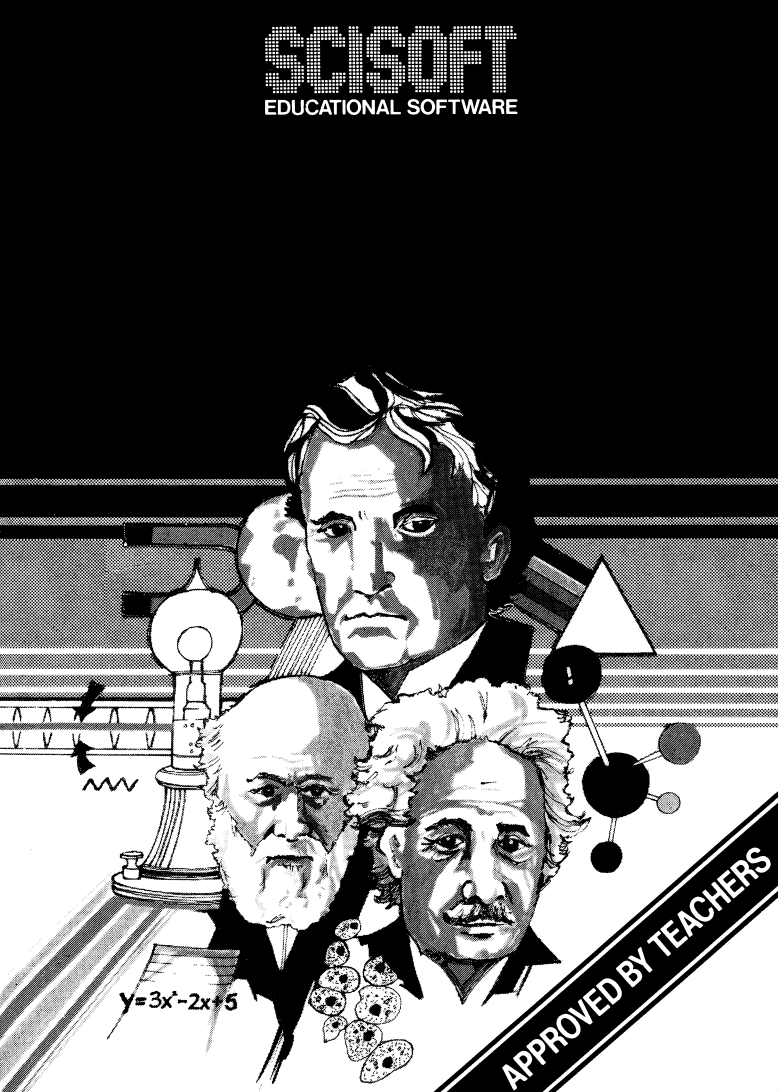
SCISOFT

EDUCATIONAL SOFTWARE



COMPUTER STUDIES

Computer studies is a comprehensive package for pupils undertaking a computer studies course in the 13-19 years age range. It is also of general interest to anyone requiring a deeper understanding of the subject. For example the track "CES" will help to provide an insight into machine code. Certain elements of the package will also be useful for students studying Mathematics and Electronics.

The package consists of

1. Six extensive programs covering a wide range of topics.
2. A program giving hints on revision technique which includes a 50 question examination paper.
3. A set of notes.

SIDE A

"FLOW" (flowcharts)
"CES" (testing CESIL - an insight to machine code)
"REVEX" (revision and mock examination)

SIDE B

"GATES" (logic gates)
"LOGIC" (Karnaugh maps, truth tables and an application of gates)
"PROCESSOR" (logic processor)
"BASES" (number bases)

Wherever possible the programs are interactive. For example "Processor" allows the pupil to investigate their own circuit designs even to the extent of mimicing faults in the circuit, whilst "Truth" section of the track "Logic" lets the pupil experiment with different Boolean expressions.

This suite of programs is not simply a series of question/answer type problems but a real advance in educational computer software.

LOADING INSTRUCTIONS

SPECTRUM 48K	LOAD " "
DRAGON 32	CLOAD
BBC & ELECTRON	CHAIN " "
COMMODORE 64	LOAD " "

Details of other programs in this series and of other educational programs can be obtained from your software supplier or from Scisoft ltd. 5 Minster Gardens, Newthorpe, Eastwood Notts. U.K. NG16 2AT.

FLOW

Most peoples introduction to computing is through the manual that came with their computer. These manuals surprisingly gloss over flowcharts. The advanced programmer may only occasionally use flowcharts but to the beginner they should be the foundation to a well structured program. For this reason Computer Studies courses generally begin with writing flowcharts prior to writing a program. Futhermore a flowchart is generally required with any Course Work that may be submitted.

"FLOW" enables the pupil to test his understanding of flowcharts. On loading the following MENU appears.

MENU

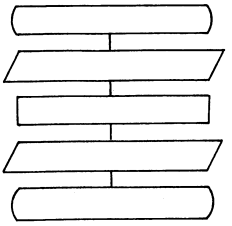
1 Logical sequence
2. Correct symbol
3. Name that symbol
4. Follow that flowchart

On pressing 1, steps in a flow chart are displayed. They need to be rearranged logically and then a flowchart produced. For example the following instructions will if correctly arranged convert temperature in degrees Celsius to Fahrenheit.

1. INPUT Temp. (T)
2. Calc. (T* 9/5) + 32
3. END
4. Print F
5. START

The instructions at this stage should be noted down. The appropriate flowchart then appears.

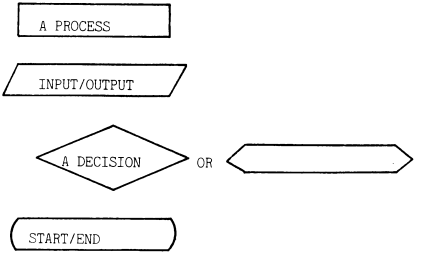
Print 'F

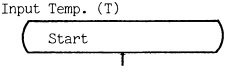


The instructions in random order appear in the top left-hand corner of the screen. These must then be fitted into the flowchart in the correct sequence, furthermore clues to the wrong order are highlighted at the end. Obviously the above example is straight forward, but more complicated programs for example using decisions are also encountered. After 5 flowcharts the pupil is returned to the menu.

Correct symbol

The pupil is given each step in a flowchart in the correct order and has to choose the correct symbol.



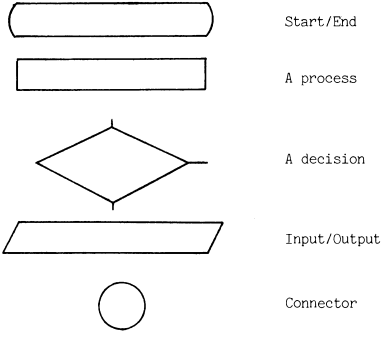


KEY 1. 2. 3. 4.

Here the correct start symbol has been used and the computer is waiting for the symbol for the second instruction - Input Temp. (T)
After 5 flowcharts the pupil is returned to the menu.

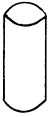
Name that symbol

The following symbols are used: PROCESSING DATA

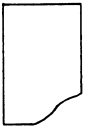


STORED DATA

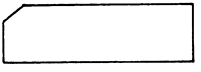
Paper tape



Magnetic disc

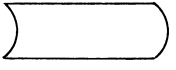


Written/Printed document

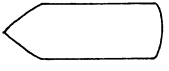


Punched cards

Magnetic tape



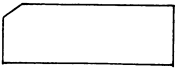
Computers memory



Visual display unit

Stored data - Visual display

MISSES
Ø



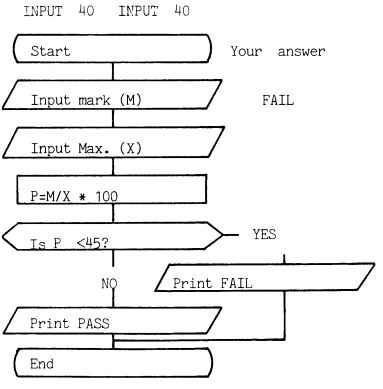
Key M - next step
Z select answer

If the pupil misses the correct answer then at the end of the round this is shown as well as his score. After 5 symbols the pupil is returned to the menu.

Follow that flowchart

The pupil is given a flowchart with inputs and outputs, one of which is missing. The pupil has to work out and key in the missing input/output.

In the flowchart below the answer is PASS. 40 has been put in as the mark followed by a maximum mark of 40. The calculation results in 100%.



My output=PASS. Are you right?

In this example the pupil failed to appreciate that a percentage was calculated. Simply but wrongly assuming that 40 was below 45 therefore the mark was below the pass rate.

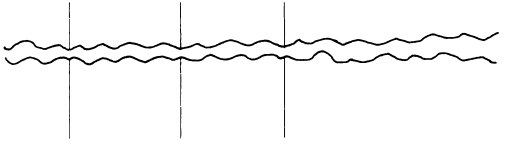
After 5 problems the pupil is again returned to the menu.

CES

This program test your understanding of CESIL (Computer Education in Schools Instructional Language) Unlike BASIC, CESIL is an example of a low level language, it is therefore very useful as a first step to understanding machine code.

CESIL programs are written on coding sheets thus:

label operation operand comment or PRINT text



%

DATA

*

Load the program and initially choose level 1. The headings above will appear together with instructions in the operation column and a list of numbers opposite the DATA heading (The % sign indicates the end of program instructions.)

The instruction IN causes the computer to wait for a number to be entered into the computer, the number then being stored in the accumulator. The numbers to be entered, sequentially, are shown opposite the DATA heading. The OUT instruction causes the computer to print out the number held in the accumulator. (A list of CESIL instructions is appended).

Suppose this program is produced by the computer.

```
label     operation     operand        PRINT
          IN
          IN
          OUT
          HALT
          %


DATA   5   23
```

This will result in initially 5 being "stored" in the accumulator then 23 the 23 overwriting the 5 already in the accumulator) and finally 23 being printed out. All programs and data produced by CES is randomised hence whenever you run this simulation of CESIL you will obtain different programs. The correct response to the program above would be:

```
                    23    ENTER
                    Z     ENTER
```

If you are wrong then the correct response is supplied. Now continue onto level 2. Level 2 progresses to STORE and LOAD instructions A number held in the accumulator can be stored in memory locations. Two memory locations with BEN and TAMS as identifiers, are used in CES. (Upto 6 alphanumerics provided that the identifier begins with a letter can be used in CESIL).

Consider the following program:

```
label     operation     operand        PRINT
          IN
          OUT
          STORE         BEN
          IN
          STORE         TAMS
          LOAD          BEN
          OUT
          HALT
          %
DATA   8   15
```

Initially 8 is stored in the accumulator and then this is printed out before a copy of it is stored in memory location BEN The next IN

instruction causes the 8 in the accumulator to be overwritten by 15, a copy of which is stored in TAMS. The 15 in the accumulator is then overwritten by the contents of BEN which is then loaded into the accumulator. This is then printed out. Thus your response to this program would have been:

8 ENTER
8 ENTER
Z ENTER

The next two levels progress onto carrying out arithmetic using CESIL. The following program is a simple addition program, storing each number before adding them together and printing out the answer.

```
label     operation     operand       PRINT
          IN
          STORE         TAMS
          IN
          STORE         BEN
          ADD           TAMS
          OUT
          HALT
          %

DATA   7   5
```

Your response to this program should have been:

12 ENTER
Z ENTER

Subtraction, multiplication and division are carried out in a similar way. Positive and negative numbers can be used but CESIL can only deal with whole numbers, hence 11 ÷ 4 would result in 2 being printed out. Furthermore -11 ÷ 4 gives -2 (Take care since if you typed, in immediate mode, PRINT INT(-11/4) then your computer would respond with -3)

Constants can be entered into programs thus:

```
label     operation     operand       PRINT

          SUBTRACT      +25
```

This will result in 25 being subtracted from the number held in the

accumulator. NB Constants can be either positive or negative but the sign must be included. They can be used with the instructions ADD, SUBTRACT, DIVIDE, MULTIPLY and LOAD. Constants are incorporated in the programs generated by level 5.

Consider the following program:

```
label     operation     operand            PRINT
          IN
          ADD           -4
          SUBTRACT      +2
          OUT
          HALT
          %

DATA   -6
```

Here -6 is placed into the accumulator to which -4 is added giving -10 and +2 subtracted resulting in an output of -12

Level 6 involves jumping to other parts of the program.

JUMP is an unconditional jump which causes the computer to jump to the label specified by the operand. In CES, END is always used as the label. Consequently the following program will result in -2 only being printed out, since after printing out -2 there is a jump to HALT.

```
label     operation     operand            PRINT
          IN
          OUT
          JUMP          END
          ADD           -4
          LOAD          -3
          OUT
END       HALT
          %

DATA   -2
```

Two conditional jumps are allowed.

JIZERO - Jump if zero

JINEG - Jump if negative

Thus in the following program the JINEG is ignored since the number in the accumulator is positive (+4)

```
label     operation     operand          PRINT
          IN
          JINEG         END
          SUBTRACT      +3
          OUT
END       HALT
          %

DATA 4
```

Thus the correct response should have been:

```
1  ENTER
Z  ENTER
```

Please note that in this simulation of CESIL programs print statements and comments would be meaningless they are however very necessary when CESIL programs are written. Should you wish to pursue the subject SCISOFT have produced a CESIL interpreter written in BASIC.

REVEX

This program contains advice gained from many years of teaching experience. There are comments for those who start early on a program of revision and for those panicking at the last minute

```
PLEASE KEY ONE OF THE FOLLOWING
1.  >6   MONTHS TO EXAM
2.  3-6 MONTHS TO EXAM
3.  2-3 MONTHS TO EXAM
4.  1 MONTH TO EXAM
5.  THE BIG DAY
6.  MOCK EXAM
```

The program includes a mock examination with a unique revision programme based upon an analysis of your answers It highlights the gaps in your knowledge of the subject. The more questions that you get wrong the more topics are suggested for revision. If you have performed particularly badly your revision programme could fill 2 or 3 screens. To obtain a hard copy press N in response to SCROLL?, COPY the screen and then press CONTINUE. Repeat this as many times as necessary. If you do not have a printer connected simply note down the screen contents prior to scrolling.

GATES

The fundamental building bricks of logic circuits are logic gates. The basic gate is an electronic device with two inputs and a single output. The output is dependent on the relative states of the two inputs There are only two states allowed either an electrical signal present (represented by 1) or not present (represented by Ø).

On loading the program you are presented with a menu.

MENU

1. AND
2. OR
3. XOR
4. XNOR
5. NAND
6. NOR

Enter number of gate simulation required.

The programs show a simulation of electron movement through these gates when the input signals are present or not. In this way you are

provided with visual evidence of the funcion of each gate.

1. "AND"

On loading, press 1 and yourscreen will look as follows:

Pulse at 1 - Press 1 or Ø?

Truth Table AND gate

1 2 3

The output from an
AND gate is true if
and only if its
inputs are all true.

r=reset p=print
q=quit

At the top of the screen is the AND gate, shown with 1 and 2 as inputs and 3 as the output. Below this, is the question - Pulse at 1 Press 1 or Ø?. You now decide whether the 1 input into the gate is to have a signal or not. Suppose that you decide "yes", then type ENTER. The 1 will now appear under 1 in the Truth Table. You will now notice that the question has changed to - Pulse at 2. The computer is now asking you to make the same decision about input 2 of the AND gate. Let us again put in 1 ENTER. Now watch the gate and you will see simulated electrons move through the gate. You will see pulses in inputs 1 and 2 and that these have produced a pulse at 3. The output at 3 is registered in the Truth Table under 3. The computer's question now changes back to - Pulse at 1, inviting you to input the other combinations of inputs. Common sense will tell you that there are only four possible combinations for two inputs:

1	2	3
1	1	1
Ø	1	
1	Ø	
Ø	Ø	

We have now generated the first possibility i.e. at 1 and 1 at 2. You can now generate the complete truth table (a Truth Table is simply a statement of how output varies with input), by keying in the other values.

Your screen should now show the following picture:

```
1-----| A |
      | N |-----3
2-----| D |
```

```
Pulse at 1 - Press 1 or Ø?
Truth Table
 1   2   3
 1   1   1   The output from an
 Ø   1   Ø   AND gate is true if
 1   Ø   Ø   and only if its
 Ø   Ø   Ø   inputs are all true
                r=reset p=print
                    q=quit
```

Thus as shown on the screen the output from an AND gate is true only if its two inputs are simultaneously true.

The AND gate can therefore make a simple decision of the type : "If it's not raining and I have sufficient money; I will go to the cinema". Input 1 could thus represent "not raining" input 2 could be "have sufficient money". If both are true then, you go to the cinema. This decision can be represented by an AND gate.

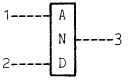
You can now try the various other gates and in turn experiment with the inputs in order to find out what the outputs are. We have given a summary for each gate type below. In circuit diagrams gates are usually represented by boxes with the name of the gate type in the box. e.g.

AND GATE

NOT AND or NAND GATE

A circle is placed on the output to represent NOT.

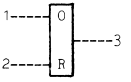
1. AND



Truth Table

1	2	3
1	1	1
Ø	1	Ø
1	Ø	Ø
Ø	Ø	Ø

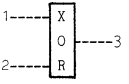
2. OR



Truth Table

1	2	3
1		1
Ø	1	1
1	Ø	1
Ø	Ø	Ø

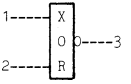
3. XOR



Truth Table

1	2	3
1	1	Ø
Ø	1	1
	Ø	1
Ø	Ø	Ø

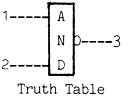
4. XNOR



Truth Table

1	2	3
1	2	3
1	1	1
Ø	1	Ø
1	Ø	Ø
Ø	Ø	1

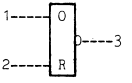
5. NAND



Truth Table

1	2	3
1	1	Ø
Ø	1	1
1	Ø	1
Ø	Ø	

6. NOR



Truth Table

1	2	3
1	2	3
1		Ø
Ø	1	Ø
1	Ø	Ø
Ø	Ø	1

LOGIC

This program is divided into three parts each of which is accessible via a MENU.

MENU

1 TRUTH
2. MAP
3. CONTROL

Press any key to continue.

1. TRUTH

A Truth Table is simply a statement of how output varies with input. This program will generate either automatically or you can generate the table manually. The program has a pre-programmed expression for demonstration purposes but the pupil is encouraged to try out his own expressions and some suggestions are made.

On selecting 1 from the menu, you will be presented with the question - "Do you wish to use the auto-generate facility"? The program has a facility to generate ascending binary numbers and feed these into the expression evaluater. The variables we have used are - A, B, C and D and for the output or dependent variable - F. Do not worry too much about understanding the expression, it is just to demonstrate the program. If you now key Y, all the possible combinations of A, B, C and D will be calculated by the computer and output F will be evaluated. Your screen should now show the following table:

A	B	C	D	F
Ø	Ø	Ø	Ø	Ø
1	Ø	Ø	Ø	Ø
Ø	1	Ø	Ø	Ø
1	1	Ø	Ø	Ø
Ø	Ø	1	Ø	Ø
1	Ø	1	Ø	Ø
Ø	1	1	Ø	1
1	1	1	Ø	Ø
Ø	Ø	Ø	1	Ø
1	Ø	Ø	1	Ø
Ø	1	Ø	1	1
1	1	Ø	1	Ø
Ø	Ø	1	1	Ø
1	Ø	1	1	Ø
Ø	1	1	1	1
1	1	1		Ø

Logical expression to evaluate

LET F=NOT A AND B AND (C OR D)

The number of unique entries in the Truth Table is 2^n, that is if there is one variable there are 2^1 unique combinations. If there are two variables then there are 2^2 unique combinations etc.

The demonstration expression has 4 variables and so we would expect 2^4 or 16 unique entries in the Truth Table. Now select from the MENU "TRUTH" but this time use the manual mode - answer N to the auto-generate question. You can now experiment with the inputs in any order. To change the expression, you simply EDIT LINE 148Ø of the program. If you also wish to show this new expression on the screen then you will also have to EDIT the PRINT statement in LINE 1Ø6Ø to read appropriately.

Try the following expression:

LET F = NOT A

By doing this you have made F dependent on one variable only (A) and by using NOT, you have created an INVERTER. Now run the program and you will obtain the following table:

A	B	C	D	F
Ø	Ø	Ø	Ø	1
1	Ø	Ø	Ø	Ø
Ø	1	Ø	Ø	1
1	1	Ø	Ø	Ø
Ø	Ø	1	Ø	1
1	Ø	1	Ø	Ø
Ø	1	1	Ø	1
1	1	1	Ø	Ø
Ø	Ø	Ø	1	1
1	Ø	Ø	1	Ø
Ø	1	Ø	1	1
1	1	Ø	1	Ø
Ø	Ø	1	1	1
1	Ø	1	1	Ø
Ø	1	1	1	1
1	1	1	1	Ø

Logical expression to evaluate

```
LET F = NOT A
```

Ignore columns B, C and D because they do affect F. You will see that if A = Ø, F = 1 and if A = 1, F = Ø. This is exactly how an inverter functions. Since there are only two combinations of input i.e. A = Ø or A = 1, then this is repeated eight times down the screen.

Now try : `LET F = A AND B`

Ignore outputs C and D and you will have simulated an AND gate in which the Truth Table is repeated four times.

A	B	C	D	F
Ø	Ø	Ø	Ø	Ø
1	Ø	Ø	Ø	Ø
Ø	1	Ø	Ø	Ø
1	1	Ø	Ø	1
Ø	Ø	1	Ø	Ø
1	Ø	1	Ø	Ø
Ø	1	1	Ø	Ø
1	1	1	Ø	1
Ø	Ø	Ø	1	Ø
1	Ø	Ø	1	Ø
Ø	1	Ø	1	Ø
1	1	Ø	1	1
Ø	Ø	1	1	Ø
1	Ø	1	1	Ø
Ø	1	1	1	Ø
1	1	1	1	1

You will see that as before, the output from the AND gate is only true if A and B are both true.

You can now move onto three and four expression functions with multiple terms and if you bear in mind that equivalent truth tables relate to equivalent functions, then you should be able to reduce complex expressions to simpler evaluation functions. For instance try the following expression:

```
LET F =  A AND (NOT A OR B)
```

The truth table is identical to the simpler AND expression. From the equivalence of the truth tables you have shown that:

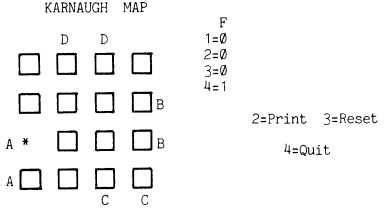
```
A AND (NOT A OR B) = A AND B
```

2. MAP

It is also possible to take a truth table and to work out the expression that produced it, using a technique called Karnaugh Mapping. Let us try to obtain the map from the following truth table:

A	B	F
Ø	Ø	Ø
1	Ø	Ø
Ø	1	Ø
1	1	1

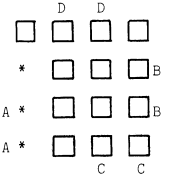
From the MENU select 2. As long as A and B are in ascending binary sequence (and if you used auto-generate in "TRUTH" then they will be), then all you need to input are the F values i.e. Ø Ø, Ø, 1. You will obtain the following map:



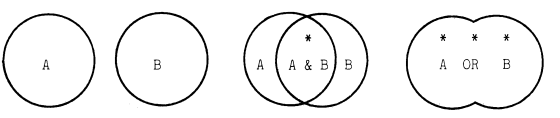
You will notice that a star has appeared on a square which is in the horizontal line shared by A and B. The expression is therefore A and B. Try :

A	B	F
Ø	Ø	Ø
1	Ø	1
Ø	1	1
1	1	1

Just input F as before and you should obtain:



The area now mapped is A or B, since two stars appear in the B field and two appear in the A field. In terms of a Venn diagram the following would be obtained:



You can obtain many tables from the "TRUTH" program and experiment with mapping these until you gain experience at map interpretation.

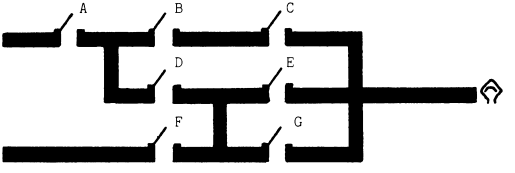
3. CONTROL

We aim to prove that logic is not a purely theoretical subject and this final part of the program demonstrates that use of logic to control a switching circuit. Again initially, suggestions on which switches to open or close are made. You can then experiment to find out which combinations will light the bulb. The power of logic is clearly demonstrated by the simplicity of the expression required to control all the various permutations of switching states that are possible.

From the MENU press 3 and you will obtain the following diagram:

EXPRESSION USED
IF a=1 AND b=1 AND c=1 OR a=1
AND d=1 AND e=1 OR a=1 AND d=1
AND g=1 OR f=1 AND g=1

CONTROL USING LOGIC



Is switch A open (Ø) or closed (1)

Presented before you is a circuit for a light bulb operated by seven switches (A to G). You are presented with the question: "Is switch A open (Ø) or closed (1)?" Let us enter 1 to indicate that the switch A is closed. You will see the switch close on the screen and the computer asks you to program switch B. Let us leave B open, so enter Ø. Also leave C open, close D, leave E and F open and close G. When this is complete the bulb will flash since current can reach it through A, D and G.

You can now experiment with all the combinations of switches to see if the bulb lights or not. (There are 128 possible combinations).

You will find that logic is so powerful that the single expression in LINE 351Ø of the program:

```
351Ø  IF a=1 AND b=1 AND c=1 OR a=1 AND d=1 AND e=1 OR a=1 AND d=1 AND
g=1 OR f=1 AND g=1 THEN GOTO 356Ø
```

controls and simulates the whole switching circuit for all 128 permutations of switching states.

PROCESSOR

The processor allows you to create computer circuits in software. These behave in exactly the same way as the actual hardware would. It is extremely simple to operate : The circuit diagram and it's connections are numbered in a logical sequence, starting from 2 upwards. 1 and Ø are forbidden as wire numbers). The gate types are entered, together with these wire numbers and the processor connects the circuit together in the computer's memory. The computer then does a series of checks on the data, works out the loose inputs and the loose outputs from the array of gates (logic system) You then vary the inputs and the computer generates the outputs (it also shows you all the internal states) Truth tables can therefore be generated from single gates upto multi-gate systems. A simulated fault mode is provided in which output from a gate can be held 'high' or 'low' in order to watch the effect on the truth table. In this way you can mimic real circuits in order to locate faults. A circuit summary feature is also provided which prints out tabulated summaries of the circuit to be analysed.

Load the program which on loading autoruns to the circuit summary feature. (Allowing you to experiment with the pre-stored circuit). You are asked the following:

```
Do you want a print out of the
circuit data (Y or N)?
```

Press Y and you are given this summary

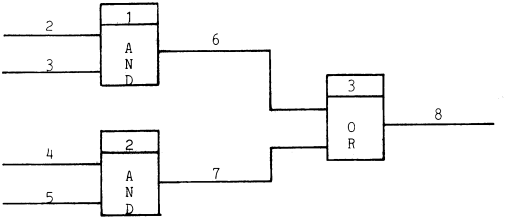
```
LOGIC SYSTEM TO BE ANALYSED

N/TYPE INPUTS        OUT

1 and  2  3          6
2 and  4  5          7
3 or   6  7          8
```

Press C ENTER

This is a tabulation of the following simple circuit:



Compare the circuit with the summary.

Now to continue press C ENTER. Other responses will wipe out the stored DATA and thus to run the demonstration you will then have to reload "PROCESSOR".

The computer gives you the following message:

```
Input data received - thank you.
I will check it for errors and
report


Data consistent   will process
```

There will be no error message since the circuit is pre-recorded within the program "PROCESSOR". N.B. Do not use BREAK and RUN as this destroys the data. The screen will then clear and present:

```
                TRUTH-TABLE
I N P U T S          O U T P U T S
2 3 4 5              8
```

```
2=copy 3=clear data 4=new

INPUT:2  (1 OR Ø)
```

You will note that the computer has worked out the inputs to this logic system are 2,3,4 and 5 and that it s only output is wire 8 You can now hold the input wires at 1 or Ø as you desire and the computer will display the output. In addition, at the bottom of the screen it will show you the logic levels (i.e. Ø's or 1's for the internal wires - those not interfacing with the outside world)

Two frames of a session in progress are shown below:

```
                TRUTH-TABLE
I N P U T S          O U T P U T S
2 3 4 5              8
Ø Ø Ø Ø              Ø
```

```
Internal ---- 6=Ø   7=Ø   8=Ø
states and
output for    2=copy  3=clear data  4=new
1st entry
              INPUT :2  (1 OR Ø)?
```

TRUTH-TABLE

I N P U T S				O U T P U T S
2	3	4	5	8
Ø	Ø	Ø	Ø	Ø
1	1	1	1	1
1	Ø	1	1	1

Internal--- states and output for last entry

6=Ø 7=1 8=1

2=copy 3=clear data 4=new

INPUT:2 (1 OR Ø)?

N.B. At any time the use of 2 ENTER, provides a hard copy; 3 clears the screen in order for you to continue with this logic system and 4 allows you to enter an entirely new logic system.

Now press 4 ENTER, this clears the stored data. The following is the input sequence for a simple 3 gate circuit. You are asked:

Number of inputs into this logic system?

The whole system has four inputs i.e. wires 2, 3, 4, 5 - so key in 4 ENTER. Now the question is:

Number of outputs?

Wire 8 is the only output therefore press 1 ENTER. The next question is:

How many gates?

There are three gates in the total system therefore press 3 ENTER. The computer now askes for specific data about each gate:

GATE 1

```
Gate number:  1

Gate type?    (lower case please)
and
Number of inputs for this gate?
2
Input line numbers?
Type ENTER after each number
2 3
Output line number for the gate?
6
Fan out?
1
```

The above sequence shows the data entry for gate one - note that input line numbers are each followed by ENTER and that Ø and 1 are not allowed.

The sequence for the remaining two gates is given below:

GATE 2

```
Gate number:  2

Gate type?
and
Number of inputs for this gate?
2
Input line numbers?
Type ENTER after each number
4 5
Output line number for the gate?
7
Fan out?
1
```

GATE 3

```
Gate number:  3

Gate type?
or
Number of inputs for this gate?
2
Input line numbers?
Type ENTER after each number
6 7
Output line number for the gate?
8
Fan out
1
```

You can now use the processor to enter your own circuits. The training objective, is that you should be capable of predicting the internal states and the output(s) before making the appropriate entry in the truth table. You should try the exercises given in the Appendix.

BASES

Understanding number bases fully is a major part in your comprehension of Computer Studies.

Our adoption of base 10 numbers for every day use is probably a result of man being provided with ten fingers. Had we been a race of intelligent octopuses, perhaps we would count using the octal system. For certain circumstances however bases other than 10 are more appropriate.

If we consider electron flow for instance, they generally flow or they do not. Switches are either closed or open and therefore perhaps the most convenient base for examining electrical systems is base 2 or BINARY. It is very convenient to build integrated circuits containing thousands of bi-state devices which can store an 'electrical 0 or an electrical' 1. These coupled with other logic devices form the whole basis of modern computing

Octal (8) and duodecimal (12) have also been used, so to has hexadecimal 16) for 'talking to' the Central Processing Units using assemblers.

It is therefore important that you achieve fluency in handling ALL bases but with some emphasis on BINARY, OCTAL and HEXADECIMAL. N.B. In "BASES" numbers upto 4096 BASE 10 are handled. Larger numbers generate an error message.

On loading the following menu appears:

```
   UNIVERSAL NUMBER BASE TUTOR
1  Convert denary to any base.

2. Convert any base to denary.

3. Convert any base to any base.

4. Number base test.

5. Number base - demonstration.

6. Quit

Press number of your choice.
```

a) Convert denary to any base

Note: this handles bases upto 16 only.
Press 1 on the keyboard and as an example convert the denary number 139 to base 2. A screen view of this completed version is shown below:

```
Input number in denary then press
ENTER. (integers only)

DENARY NUMBER=139


TO WHICH BASE?

BASE=2


DENARY 139 IN BASE 2
  =10001011

Press C to continue.
Press R to return to menu.
```

Now press C and try the following exercises:

i) 255 denary to base 16

ii) 8 denary to base 2

iii) 16 denary to base 2

b) Convert any base (upto 16) to denary

On returning to the menu press 2 and try converting FF (base 16) to denary. A typical screen view of the conversion is shown below:

```
                Input base you are working in.

                NUMBER BASE=16

This in the-- INPUT NUMBER IN BASE 16
case of
letters
must be
input in
lower
case            ff IN BASE 16 = 255 denary      [lower case is used for your
                                                 convenience].
                Press C to continue.
                Press R to return to menu.
```

Now press C and try the following exercises:

i) Convert 10101(binary) to denary

ii) Convert 100(base 3) to denary

iii) Convert 100(base 9) to denary

c) Convert any base to any base (maximum 16)

Select 3 from the menu and try converting 3476 (base 8) to base 2. As before the completed conversion is shown below:

```
                Input base you are working in.

                NUMBER BASE=8
                INPUT NUMBER IN BASE 8
                3476


                TO WHICH BASE?
                BASE =2

                3476 IN BASE 8
                =  10011111Ø IN BASE 2
```

As before press C and try the following:

i) Convert 3Ø (base 9) to base 3
ii) Convert 3Ø (base 8) to base 2
iii) Convert 1Ø1Ø11 (base 2) to base 4
iv) Convert 3Ø (base 6) to base 3

d) Number base - demonstration.

On returning to the menu press 5. This mode is used to help your comprehension on any base conversion, which you have difficulty in understanding. It illustrates fully, conversions of numbers in any base (upto 10), into denary (base 10), showing you how the mathematical process works.

A typical demonstration is shown below in which 1Ø1Ø base 2 is converted to base 10.

Number base demo--up to base ten

Power:	4	3	2	1	0
Base:2	2	2	2	2	2
Denary	16	8	4	2	1
	X	X	X	X	X
Number:		1	Ø	1	Ø
Products		8	Ø	2	Ø

Therefore denary number=10

To get the values for each column in a particular base, starting from the right, the base is raised to ascending powers, starting with zero and incrementing by 1. Thus in base 2 our columns are worth in denary:

16 8 4 2

These are then multiplied by the number themselves e.g.

Column Values	16	8	4	2	1
Times	X	X	X	X	X
Number in base 2		1	Ø	1	Ø
		8	Ø	2	Ø

The values are then added 8 + Ø + 2 + Ø 1Ø

Therefore 1Ø1Ø base 2 = 1Ø base 1Ø

e) Number base test

On returning to the menu press 4 and the following appears

Number base conversion test.

Answer the following questions-
take as much time as you need
use pencil and paper to work out
your answers the objective is
a perfect score.

Repeat the exercise if you get
any wrong.

Ten questions of the type shown below are presented:

14 is a denary number

What is it in base 4?

ANSWER=32

YOUR RESPONSE=32

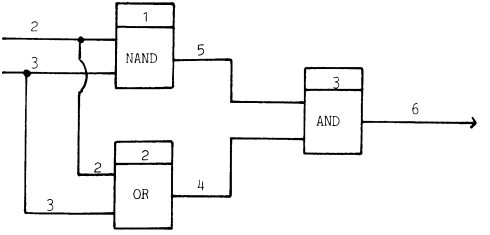
N.B. The data is random, so a new set of questions is presented each time you use this particular track.

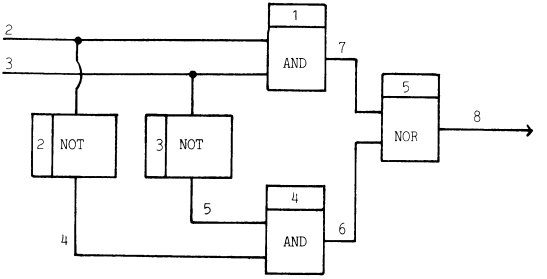
APPENDIX

CESIL INSTRUCTIONS

INSTRUCTION		OPERATION
IN		Data input is stored in the accumulator
OUT		Contents of the accumulator are printed out
HALT		End of program
STORE	BEN	Copy contents of the accumulator and store them in memory location BEN
LOAD	TAMS	Copy contents of memory location TAMS into the accumulator
ADD	BEN	Add contents of memory location BEN to the contents of the accumulator
SUBTRACT	TAMS	Subtract contents of memory location TAMS from the contents of the accumulator
MULTIPLY	BEN	Multiply contents of memory location BEN to the contents of the accumulator
DIVIDE	TAMS	Divide contents of the accumulator by the contents of memory location TAMS
JUMP		Unconditional jump - whatever the number held in the accumulator
JIZERO		Conditional jump jump if the number held in the accumulator is zero
JINEG		Conditional jump jump if the number held in the accumulator is negative
PRINT		Causes print text to be printed out
(		Indicates a comment

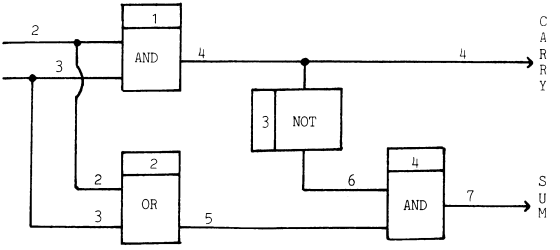
i) Show that the following two networks are equivalent.





Notes: Fan out is one for all gates, since the outputs are all connected to one input load. Circuits are equivalent if they have the same truth table. You should discover that both these elaborate circuits could be replaced by a single XNOR gate.

ii) The Half Adder (2 inputs)

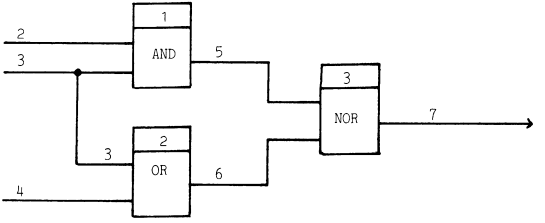


Note Fan out for gate 1 is 2.

You should obtain the following truth table

INPUTS	OUTPUTS
2 3	4 7
Ø Ø	Ø Ø
1 1	1 Ø
Ø 1	Ø 1
1 Ø	Ø 1

iii) Process the following circuit



You should obtain the following truth table.

INPUTS	OUTPUTS
2 3 4	7
1 1 1	Ø
Ø 1 1	Ø
1 Ø 1	Ø
Ø Ø 1	Ø
1 1 Ø	Ø
Ø 1 1	Ø
1 Ø Ø	1
Ø Ø Ø	1

You will notice that changing input 2 has no effect whatsoever on the output and that the whole circuit could therefore be replaced by a two input NOR gate.

General Notes

Be logical in your wire or line numbering sequence or the circuit syntax checker will throw it out.

Do not couple outputs back to inputs - the processor will not be able to stabilise a circuit which oscillates.

Gates with systems with large numbers of inputs will cause a collision on the screen of input and output numbers, both on the truth table and the circuit summary Split large circuits into smaller chunks and process separately.

Circuits are saved with the program. When you save use auto run to line 220 (SAVE "PROCESSOR" LINE 220).

The main applications of "PROCESSOR" are multiplexing, de-multiplexing, decoding and encoding. The processor will not handle flip flops or decode counters or other historical/time delay devices.

SCISOFT EDUCATIONAL SOFTWARE FOR MICRO COMPUTERS

All of Scisoft's programs have been comprehensively tested and are regularly used in schools. Please note that the pupil, parent or teacher should check with the syllabus to make sure that all the components of the packages are relevant. Some Boards may not require knowledge of all the contents of our packages.

PROGRAMS FOR YOUNGER CHILDREN

All available in Full Colour Presentation Boxes

NEW

STAR READER (6 - 11 YRS)

Improve your child's reading. Written by experts in reading skills. Designed to complement reading skills taught in schools. Simple to use, but extremely sophisticated programs utilising some machine code and most of a 48K machine.
PACK A — Space and Mountains Themes (2 Programs)
PACK B — Sea and Pyramids Themes (2 Programs)

WIZARD BOX ANY AGE

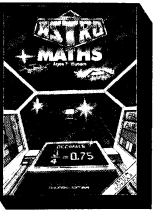
NEW

Set a test, or a quiz, learn foreign language vocabulary. Multi-purpose program which holds 10 sets of 10 words for 45 children (Spectrum version). A game can be played at the end of the test but only if the child obtains a high score. A versatile and useful program which can be used in all subject areas in a school, at home and even at parties. The Spectrum package also has 3 extra sets of the program containing tests on (1) French Vocabulary (2) Junior Spelling (3) General Knowledge.

JUNGLE MATHS (5 - 13 YRS)

Add, Subtract, Multiply, Divide. One of our most popular programs. The teacher or parent has full control over the level of difficulty, the type of problem and the time allowed. The child has to cross a jungle answering problems on the way. Getting an answer wrong results in a death too horrible to contemplate.

ASTRO MATHS (7 - 15 YRS)

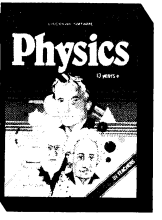


Fractions to Decimals and Percentages. For every two correct answers the child plays a space game and thus saves his ship from destruction. Level and time allowed is set by the teacher or parent.

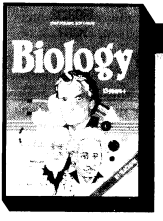
PACKAGES FOR PUPILS STUDYING FOR EXAMINATIONS

All the programs have been produced by very experienced teachers. They all contain problems to be solved on the major themes of the relevant examination syllabuses. Wherever possible the problems contain randomised data, so that the programs can be used time and time again with increasing benefit. Most packs also contain sets of multiple choice questions typical of those found in examination papers. Complete with a comprehensive book of REVISION NOTES. Extremely useful as examination day approaches. All packs contain a program of hints on HOW TO REVISE AT HOME. Written to be compatible with most O-Level Boards and also useful for the most able CSE pupil.

PHYSICS



Two programs full of problems. Ohm's Law, Refraction, Resistances, Linear Expansion, Moments etc., etc., 5 sets of multiple choice questions. Book of notes. Hints on revision. Fantastic value.

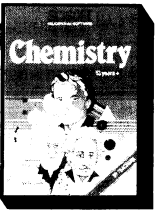


BIOLOGY

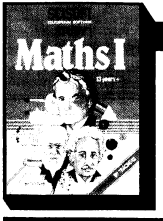
An identification program which requires knowledge of Vitamins, Minerals and Carbohydrates. Genetics program, multiple choice questions and superb diagrams plus questions on the alimentary canal, kidney, flower structure. Book of notes. Hints on revision.

REVISED EDITION

CHEMISTRY



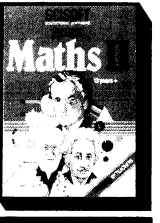
Two programs full of problems. Gas Laws, Molarity, Titrations, Mystery Compounds etc. 3 sets of multiple choice problems. Book of notes. Hints on revision. Very good use of graphics.



MATHS PART I

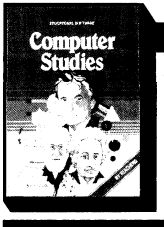
Problems with random data on:-Algebra, Equations, Functions, Areas, Perimeters and Volumes. Book of notes. Hints on revision.
A real must for all children.

MATHS PART II



A series of programs designed to complement PART I. Problems with random data on:- Proportions, Percentages, Differentiation, Integration, Trigonometry. Book of notes. Hints on revision.

Part III — Coming soon. **NEW**



COMPUTER STUDIES

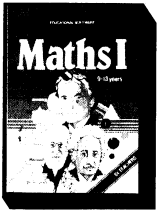
A revised edition of the well tested suite of programs on logic gates. Also including Boolean Algebra, Number Bases, Flow Charts etc.

TEACHERS MARK BOOK

Store up to 10 sets of marks for up to 700 pupils — sorts alphabetically (boys then girls) by form or in rank order. All options available from menu. Printer routines included. Gives individual pupil profile. Ideal for parental interviews.

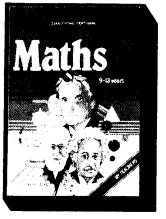
INTERMEDIATE LEVEL PROGRAMS (9 - 13 YRS)

MATHS PARTS I AND II



5 programs per pack. Exciting and stimulating program that makes maths fun to learn and understand.
Part I — draw symmetrical patterns; find the factors; compare fractions; use co-ordinates by playing battleships; 'guess' the shape.

NEW



Part II — run a lemonade stand; use logic in Towers of Hanoi; design a shape and view a 3D representation at different angles; 'guess' the percentage. Superb value for money.

Part III — Coming soon.

FRENCH

A truly remarkable package of programs. Describe a picture; test your vocabulary using a variety of novel ideas; put sentences in the correct sequence. A new and novel way of learning a foreign language.

GERMAN

The German version of the above package.

SCISOFT
EDUCATIONAL SOFTWARE