The Complete Machine Code Tutor

by

Malcolm Evans

Amstrad CPC 464

LOADING INSTRUCTIONS

This program is arranged on four sides of tape.

To load the program press CTRL and small ENTER keys and start tape recorder on side 1. This will load in the screen and program/simulator. Do NOT press ESC during the loading of the simulator.

You will then be asked whether you wish to load lessons.

If NO (N) you will enter straight into the editor part of the simulator. This is NOT recommended for beginners. You may escape by pressing ESC.

If YES (Y) you will be requested to load the required group of lessons. Choose the desired section and start the tape recorder. The group you have chosen will appear on the screen. If you have selected the wrong group pressing ESC will stop the load, and enable you to load another section.

The program and lessons are arranged on the cassette tapes as follows:

SIDE 1	-	Program/Simulator	:	Lessons 1 - 9
SIDE 2	-	Lessons 10 - 17	:	Lessons 18 - 25
SIDE 3	-	Program/Simulator	:	Lessons 18 - 25
SIDE 4	-	Lessons 26 - 35	:	Lessons 1 - 9

ASSEMBLER INSTRUCTIONS

The assembler accepts all Z80 instructions as included in the lesson summaries. In addition it will accept the following instructions itself:

DEFB n : Define Byte. This allocates the next memory location for storage, (this may be labelled), and initially loads it with n. The simulator will continually display the contents of the location in decimal or hex.

DEFW nn : Define Word. This allocates the next two memory locations for storage, (may be labelled), and initially loads it with the two byte number nn. The simulator will continually display the contents of these locations as a single decimal or hex number.

BIN n : This is the same as DEFB, except that the simulator displays the memory contents in Binary form. n is a decimal or hex number.

The assembler will also accept labels. A label is defined in relationship to numbers as follows:

Decimal Number : Any string containing only 0-9

Hexadecimal Number : Any string containing only 0-9 or A-F. It may be distinguished from decimal by following it with an H.

Label : Labels a memory location. Any string not covered by decimal or hexadecimal numbers above, register names or conditional letters i.e. C, NC, Z, NZ etc. The maximum length is 6 characters.

ERROR MESSAGES - ASSEMBLER

Instruction Unknown - First part of mnemonic not recognised

Space missing - Space missing after first part of mnemonic i.e. LDA,5

Missing space or , - Space or , missing after second part of mnemonic i.e. LD A 5 or SUB AM

Error after Inst. - Error after first part of mnemonic which the assembler was unable to recognise as any other defined error

Number missing - Part of mnemonic missing could be a number

Bracket missing - A bracket missing

Number too large - 16 Bit number greater than 65535

Label too long - Labels can only have upto 6 characters

Label not found - Label appears in instruction but not declared in label column

No DEFB or DEFW - Label with no instruction

Can only ADD IX/IY - SBC IX/IY or ADC IX/IY found

Offset too big - Offset in (IX+d) or(IY+d) Instruction greater than 255

Offset missing - No offset in (IX+d) or(IY+d) instruction

ERROR MESSAGE - SIMULATOR

Program counter has jumped to non valid address

- Your program has caused the PC to jump to an address which is not the beginning of one of your instruction lines

You are trying to run code in allocated storage area

- Your program has caused the PC to jump to an address which has been allocated as storage

YOU ARE ABOUT TO AFFECT MEMORY AREA NOT ALLOCATED TO YOU

- YOU ARE ABOUT TO LOAD A MEMORY LOCATION NOT ALLOCATED TO YOU. YOUR ALLOCATED AREA IS 3840(0F00H) TO 4032(0FC0H)

YOU ARE ABOUT TO WRITE TO MEMORY WHICH WILL AFFECT YOUR PROGRAM

- YOU ARE ABOUT TO LOAD A MEMORY LOCATION WITHIN YOUR PROGRAM INSTRUCTIONS.

THE STACK POINTER IS OUTSIDE ALLOCATED MEMORY AREA

- THE STACK POINTER HAS MOVED OUTSIDE THE AREA 4033(0FC1H) TO 4095 TO 0FFFH

THERE ARE TOO MANY REGISTERS CALLED UP

- YOUR PROGRAM USES MORE REGISTERS THAN THE SIMULATOR CAN DISPLAY. THE PROGRAM CAN BE RUN BUT ONLY THOSE REGISTERS DISPLAYED CAN BE SHOWN.

THE LESSONS IN THIS PROGRAM ARE ARRANGED IN FOUR GROUPS. UPON LOADING OF EACH SECTION AN INTRODUCTION APPEARS ON THE SCREEN. THE MENU FOR THE GROUP OF LESSONS CAN THEN BE OBTAINED BY PRESSING SPACE. AT ANY TIME WHILST IN A LESSON OR EXAMPLE PRESSING ESC WILL RETURN YOU TO THE MENU.

THE FOLLOWING IS A COMPLETE LIST OF ALL KEYS USED AT VARIOUS STAGES WITH A FULL DESCRIPTION OF THEIR FUNCTION.

MENU

ENTER : WILL ENTER ANY LESSON OR EXAMPLE HIGHLIGHTED ON THE MENU.

SPACE : PRESSING THIS KEY ALLOWS YOU TO CHOOSE WHICH ITEM TO ENTER

LESSON AND EXAMPLE TEXT

SPACE : PRESSING THIS KEY WILL DISPLAY THE NEXT PAGE. AT THE END OF EACH LESSON IT WILL EFFECT A RETURN TO MENU.

ESC : WILL RETURN YOU TO THE MENU AT ANY TIME

SIMULATOR

: ALL KEYS AUTO-RUN.
ALL KEY PRESSES FOR ON SCREEN INSTRUCTIONS HAVE BEEN ABBREVIATED. [ ] DENOTES "WITH CAPS SHIFT".

RUN [R] : THIS CLEARS ALL REGISTERS AND STARTS PROGRAM RUNNING.

ANY KEY : IF IN RUNNING STATE WILL PERFORM THE HIGHLIGHTED INSTRUCTION.

STOP [S] : THIS STOPS THE RUNNING OF THE PROGRAM

EDIT [E] : PRESSING THIS KEY ENTERS THE EDITOR, ALLOWING YOU TO MODIFY OR RE-WRITE THE PROGRAM.

[T] : THIS SWOPS THE DISPLAY BETWEEN DECIMAL AND HEXADECIMAL NOTATION. THIS KEY IS OPERATIVE ONLY WHEN THE PROGRAM IS NOT RUNNING.

ESC : WILL RETURN YOU TO THE MENU.

ONCE COMPLETED IF YOU WISH TO REPEAT THE EXERCISE THEN PRESS [R]

EDITOR : ALL KEYS AUTO-RUN

CURSOR KEYS : PRESSING THESE WILL ALLOW YOU TO MOVE THE CURSOR IN THE DIRECTION OF THE ARROWS.

CLEAR [C] : THIS CLEARS THE EDITOR SCREEN AND MEMORY AND RETURNS THE CURSOR TO THE START OF THE SCREEN.

SPACE : TABS TO BEGINNING OF INSTRUCTION WHEN IN LABEL COLUMN. ELSEWHERE A SPACE WILL BE CREATED.

ENTER : THIS MOVES THE CURSOR TO THE START OF THE NEXT LINE

DELETE : THIS DELETES THE CHARACTER TO THE LEFT OF THE CURSOR AND SHIFTS THE CURSOR ONE SPACE TO THE LEFT.

ASSEMBLE [A] : INITIATES ASSEMBLY OF THE PROGRAM ON SCREEN. IF ASSEMBLED CORRECTLY THE SIMULATOR MODE IS ENTERED. IF AN ERROR IS FOUND THEN AN ERROR STATEMENT IS DISPLAYED AGAINST THE APPROPRIATE LINE OF THE PROGRAM AND THE EDITOR WAITS FOR CORRECTION.

ESC : WILL RETURN YOU TO THE MENU.

THE EDITOR ALWAYS DISPLAYS LETTERS IN CAPITALS. HOWEVER CAPS SHIFT IS NOT REQUIRED.

INTRODUCTION TO LESSONS

The Complete Machine Code Tutor contains 35 lessons covering all the instructions on the Z80 processor, which is the processor in your Spectrum Computer.

All the lessons are dealt with in great detail on screen, and in many cases are followed by example programs, which you can use as exercises by modifying them yourself. There is no danger that you might crash the system.

There now follows a list of all the lesson headings, together with an indication of those lessons that are followed by examples. Under each heading there is a summary of instructions which will become clear to you as you progress through the Tutor. These summaries are intended as a permanent record of instructions, to which you can easily refer for revision purposes, without having to refer back to the screen text.

LESSON 1	-	Registers and Memory
		Initially we only consider A,B,C,D, E,H, and L registers.
LESSON 2	-	Simple Load Instructions
		The following instructions are covered:
LD R,R'		where R and R' are any of the following: A,B,C,D,E,H and L.
LD R,N		where N is a number 0 - 255
LD A,(NN)		where NN is a memory location 0 - 65535
LD (NN),A		Examples follow this lesson
LESSON 3	-	Register Pairs
		A number in a register pair is 256 x the high Byte + the low Byte.
		The following instructions are covered:
LD DD,NN		where DD is any register pair BC, DE, and HL. NN is a number 0 - 65535
LD DD,(NN) LD (NN),DD		where NN is address of a memory location 0 - 65535.
EX DE,HL		exchanges register contents.
		Examples follow this lesson

LESSON 4 -	Indirect Addressing
	The following instructions are covered:
LD r,(HL)	where r is any single register A,B,C,D,E,H, or L.
LD (HL),r	
LD A,(BC)	
LD A,(DE)	
LD (BC),A	
LD (DE),A	Examples follow this lesson
LESSON 5 -	Additions and the Carry Flag
	Additions with Accumulator and HL register pair are discussed as well as ADD with Carry.
	The following instructions are covered:
ADD A,n	where n is a number 0 - 255
ADD A,r	where r is any single register
ADD A,(HL)	
ADD HL,BC	
ADD HL,DE	
ADC A,n	ADD with Carry
ADC A,r	
ADC A,(HL)	
ADC HL,BC	
ADC HL,DE	
	2 Examples follow this lesson
LESSON 6 -	Subtraction and the Carry Flag
	Subtraction with and without Carry on the Accumulator and HL register pair are discussed.
	The following instructions are covered:
SUB n)	
SUB r)	SUBtract from A, n, r, or (HL)
SUB (HL))	
SBC A,n)	
SBC A,r)	SuBtract from A with carry
SBC A,(HL))	
SBC HL,BC)	SuBtract from HL with carry
SBC HL,DE)	
SCF	Set Carry Flag
CCF	Compliment Carry Flag
	2 Examples follow this lesson

LESSON 7 - Increment and Decrement Instructions

The following instructions are covered:

INC r
INC (HL)
INC dd

DEC r
DEC (HL)
DEC dd

Examples follow this lesson

LESSON 8 - The Zero Flag

No new instructions are covered in this lesson, which is included to show you the effect on the Zero Flag of all the instructions considered in previous lessons.

A table of the effects of all instructions on all flags is given in Appendix (A)

Examples follow this lesson

LESSON 9 - Compare

If n is the number with which A is compared, then the following results:-

	Carry	Zero
A greater than n	0	0
A equals n	0	1
A Less than n	1	0

The following instructions are covered:

CP n	Compares A with n (0-255)
CP r	Compares A with register r
CP (HL)	Compares A with memory location (HL)

Examples follow this lesson

LESSON 10 - Conditional & Unconditional Jumps

The following instructions are covered:

JP nn
JP (HL)

JP NC,nn	jump if carry flag not set
JP C,nn	jump if carry flag set
JP NZ,nn	jump if zero flag not set
JP Z,nn	jump if zero flag set

Examples follow this lesson

LESSON 11 - RELATIVE JUMPS

THE FOLLOWING INSTRUCTIONS ARE COVERED:

JR E — WHERE E IS THE DISPLACEMENT IN THE RANGE 127 TO -128

JR NC,E
JR C,E
JR NZ,E
JR Z,E

DJNZ E — DECREMENT AND JUMP ON NON ZERO

2 EXAMPLES FOLLOW THIS LESSON

LESSON 12 - THE STACK

THE STACK AND THE STACK POINTER ARE INTRODUCED.

THE FOLLOWING INSTRUCTIONS ARE COVERED:

PUSH DD — WHERE DD IS AF, BC, DE, OR HL. FROM NOW ON DD CAN BE CONSIDERED AS BC, DE, HL, OR SP.

POP DD

LD SP,NN
LD SP,(NN)
LD (NN),SP
LD SP,HL

ADD HL,SP
ADC HL,SP
SBC HL,SP

INC SP
DEC SP

EX (SP),HL

EXAMPLES FOLLOW THIS LESSON

LESSON 13 - CALLS TO SUBROUTINES

THE FOLLOWING INSTRUCTIONS ARE COVERED:

CALL	RET	UNCONDITIONAL
CALL NC,NN	RET NC	NO CARRY
CALL C,NN	RET C	CARRY SET
CALL NZ,NN	RET NZ	NOT ZERO
CALL Z,NN	RET Z	ZERO SET

2 EXAMPLES FOLLOW THIS LESSON

LESSON 14 - BINARY NOTATION

THIS LESSON IS ABOUT A WHOLE NEW NUMBER BASE - BINARY. THIS IS A SYSTEM OF USING ONLY TWO DIFFERENT NUMBERS, ONE AND ZERO, IN EACH DIGIT COLUMN. A FLAG IS AN EXCELLENT EXAMPLE OF A BINARY DIGIT. THIS IS BECAUSE IT CAN ONLY HOLD A VALUE OF ONE OR ZERO. AFTER READING THE LESSON ABOUT BINARY, YOU MAY FIND THIS REFERENCE CHART USEFUL:

Values of Bit 7 to Bit 0:

Bit Number:	7	6	5	4	3	2	1	0
Values :	128	64	32	16	8	4	2	1

Examples follow this lesson

LESSON 15 - Hexadecimal Notation

Hexadecimal notation is discussed at great length in this lesson, but the following table will prove an invaluable reference:-

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Examples follow this lesson

LESSON 16 - Binary Coded Decimal Notation

The instruction DAA (Decimal Adjust Accumulator) is introduced.

Examples follow this lesson

LESSON 17 - Positive & Negative Number Notation

The instructions introduced are CPL (Complement) and NEG (Negate), as well as the overflow and sign flags. A table of the effects of all instructions on the overflow and sign flags is given in Appendix (A)

Examples follow this lesson

LESSON 18 - Parity

The Parity flag and its uses are introduced.

A table of the effects of all instructions on the parity flag is given in Appendix (A).

LESSON 19 - The Flag Register and AF Register Pair

LESSON 20 - Sign & P/V Flags in Instructions

The following instructions are covered:

JP PO,nn	CALL PO,nn	RET PO	Parity odd = 0
JP PE,nn	CALL PE,nn	RET PE	Parity even = 1
JP P,nn	CALL P,nn	RET P	Sign +ve
JP M,nn	CALL M,nn	RET M	Sign -ve

If the condition is not met the program will not jump, call a subroutine or return

Examples follow this lesson

LESSON 21 - Bit Manipulation

The following instructions are covered:

SET N,r
SET N,(HL)

where N is the bit number 0-7

RES N,r
RES N,(HL)

BIT N,r
BIT N,(HL)

Examples follow this lesson

LESSON 22 - Logical Instructions

The following instructions are covered:

AND n
AND r
AND (HL)

OR n
OR r
OR (HL)

XOR n
XOR r
XOR (HL)

Examples follow this lesson

LESSON 23 - Shift Instructions

Shift Instructions are pictorially illustrated in Appendix (B).

The following instructions are covered:

SRA r SRA (HL)	divides +ve and -ve numbers by 2
SRL r SRL (HL)	divides +ve numbers 0 - 255 by 2
SLA r SLA (HL)	multiplies +ve and -ve numbers by 2

Examples follow this lesson

LESSON 24 - Rotate Instructions

Rotate Instructions are pictorially illustrated in Appendix (B)

The following instructions are covered:

RLC r RLC (HL)	rotate r left, carry duplicates
RLCA	rotate A left, carry duplicates
RL r RL (HL)	rotate r and carry left
RLA	rotate A and carry left
RRC r RRC (HL)	rotate r right, carry duplicates
RRCA	rotate A right, carry duplicates
RR r RR (HL)	rotate r and Carry right
RRA	rotate A and Carry right

Examples follow this lesson

LESSON 25 - Decimal Rotate

Decimal rotate instructions are pictorially illustrated in Appendix (B)

The following instructions are covered:

RLD	Rotate Left Decimal (x10)
RRD	Rotate Right Decimal(/10)

Examples follow this lesson

LESSON 26 - Index Registers

The IX or IY register can replace the HL register in all instructions except ADC HL,dd SBC HL,dd and EX DE,HL

The following instructions are covered:

LD r,(IX+d)	LD IX,nn	LD SP,IX
LD (IX+d),r	LD IX,(nn)	
LD (IX+d),n	LD (nn),IX	EX (SP),IX
ADD A,(IX+d)	INC (IX+d)	AND (IX+d)
ADC A,(IX+d)	DEC (IX+d)	OR (IX+d)
SUB (IX+d)		XOR (IX+d)
SBC A,(IX+d)	CP (IX+d)	
ADD IX,dd	INC IX	DEC IX
SLA (IX+d)	SRA (IX+d)	SRL (IX+d)
RLC (IX+d)	RL (IX+d)	RRC (IX+d)
RR (IX+d)		
SET n (IX+d)	RES n,(IX+d)	BIT n,(IX+d)
JP (IX)		

Examples follow this lesson

LESSON 27 - The Alternative Set of Registers

THE FOLLOWING INSTRUCTIONS ARE COVERED:

EX AF,AF' EXCHANGES THE CONTENTS OF AF AND AF'

EXX EXCHANGES BC,DE AND HL, WITH BC',DE' AND HL' RESPECTIVELY.

Examples follow this lesson

LESSON 28 - Input and Output Instructions

The following instructions are covered:

IN A,(N) WHERE N IS THE NUMBER OF THE INPUT PORT (0-255)

IN R,(C)
IN F,(N)

OUT (N),A
OUT (C),R

Examples follow this lesson

LESSON 29 - Block Instructions

LESSON 30 - Block Transfer Instructions

The following Instructions are covered:

LDI POINTER INCREMENTED
LDIR POINTER INCREMENTED AND REPEATED UNTIL NUMBER FOUND OR BC=0
LDD POINTER DECREMENTED
LDDR POINTER DECREMENTED AND REPEATED UNTIL NUMBER FOUND OR BC=0

Examples follow this lesson

LESSON 31 - Block Search

The following Instructions are covered:

CPI POINTER INCREMENTED
CPIR POINTER INCREMENTED AND REPEATED UNTIL NUMBER FOUND OR BC=0
CPD POINTER DECREMENTED
CPDR POINTER DECREMENTED AND REPEATED UNTIL NUMBER FOUND OR BC=0

Examples follow this lesson

LESSON 32 - Block Input/Output Instructions

The Block Input instructions covered are:

INI INCREMENTING
INIR INCREMENTING AND REPEATING
IND DECREMENTING
OTDR DECREMENTING AND REPEATING

The Block Output Instructions covered are:

OUTI	Incrementing
OTIR	Incrementing and repeating
OUTD	Decrementing
OTDR	Decrementing

LESSON 33 - Processor Control Instructions

The following instructions are covered:

NOP	
HALT	
RST n	Where n = 00H, 08H, 10H, 18H, 20H, 28H, 30H, or 38H

LD A,R
LD R,A
LD A,I
LD I,A

Examples of the use of the Refresh register follow this lesson.

LESSON 34 - Interrupts

The following instructions are covered:

EI	Enable Interrupts
DI	Disable Interrupts
IM0 IM1 IM2	Interrupt Modes
RETI	Return from Interrupt
RETN	Return from Non-Maskable Interrupt

LESSON 35 - Finale

APPENDIX A

EFFECT OF INSTRUCTIONS ON FLAGS
SINGLE REGISTER INSTRUCTIONS

INSTRUCTION	CARRY	ZERO	P/V	SIGN
ALL LD	+	+	+	+
EXCEPT LDA, I & LDA,R	+	!	!(1)	!
ADD,ADC,SUB,SBC,CP	!	!	V	!
INC,DEC	+	!	V	!
DAA	!	!	P	!
NEG	!	!	V	!
SHIFT & ROTATE SPECIFICALLY ON A	!	+	+	+
RLD,RDD	+	!	P	!
ALL OTHER SHIFT & ROTATES ON R	!	!	P	!
AND,OR,XOR	0	!	P	!
BIT	+	!	?	?
SET,RES	+	+	+	+
CCF	!	+	+	+
SCF	1	+	+	+

NOTATION	+	NOT AFFECTED	V	P/V INDICATES OVERFLOW	!	AFFECTED ACCORDING TO INSTRUCTION
	P	P/V INDICATES PARITY	0	FLAG RESET		
	1	FLAG SET	?	FLAG STATE UNKNOWN		

APPENDIX A

EFFECT OF INSTRUCTIONS ON FLAGS
REGISTER PAIR OPERATIONS

INSTRUCTION	CARRY	ZERO	P/V	SIGN
ALL LD INSTRUCTIONS	+	+	+	+
ALL EXCHANGE INSTRUCTIONS	+	+	+	+
ADD	!	+	+	+
ADC,SBC	!	!	V	!
INC,DEC	+	+	+	+
PUSH,POP	+	+	+	+

NOTATION	+	NOT AFFECTED	V	P/V INDICATES OVERFLOW	!	AFFECTED ACCORDING TO INSTRUCTION
	P	P/V INDICATES PARITY	0	FLAG RESET		
	1	FLAG SET	?	FLAG STATE UNKNOWN		

APPENDIX A

EFFECT OF INSTRUCTIONS ON PAIRS
MISCELLANEOUS INSTRUCTIONS

INSTRUCTION	CARRY	ZERO	P/V	SIGN
ALL JP,JR,CALLS,RET & DJNZ	+	+	+	+
LDI,LDD	+	+	!(2)	+
LDIR LDDR	+	+	0	+
CPI,CPIR,CPD,CPDR	+	!(3)	!(2)	!
IN A,(N); OUT(N),A)	+	+	+	+
OUT (C), R)				
IN R, (C)	+	!	P	!
INI,IND, OUTI, OUTD	+	!(4)	?	?
INIR,INDR,OTIR,OTDR	+	1	?	?
NOP, HALT, DI, EI, IM	+	+	+	+

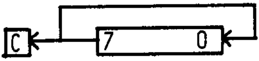
NOTES:
(1) P/V DISPLAYS STATE OF INTERRUPT ENABLE FLAG
(2) P/V = 0 IF BC = 0 IF NOT P/V = 1
(3) Z = 1 IF A = (HL) IF NOT Z = 0
(4) Z = 1 IF B = 0 IF NOT Z = 1

APPENDIX B

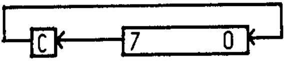
SHIFT & ROTATE INSTRUCTIONS
PICTORIAL DESCRIPTION

MNEMONIC

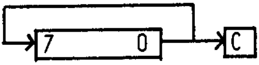
RLC RLCA



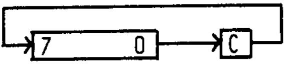
RL, RLA



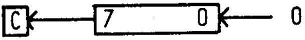
RRC, RRCA



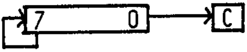
RR RRA



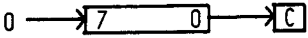
SLA



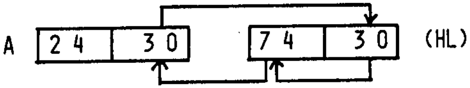
SRA



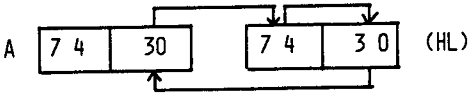
SRL



RLD



RRD



GLOSSARY

ASSEMBLY LANGUAGE - A LANGUAGE USING MNEMONICS TO REPRESENT MACHINE CODE OPERATIONS. A LOW-LEVEL LANGUAGE. AN ASSEMBLY LANGUAGE PROGRAM CAN NOT ITSELF BE RUN UNTIL IT IS ASSEMBLED.

BINARY - TWO. IN BINARY ARITHMETIC THE DIGITS 0 AND 1 ARE USED TO REPRESENT NUMBERS.

BINARY CODED DECIMAL (BCD) - A SYSTEM WHERE A NYBBLE REPRESENTS ONE DECIMAL NUMBER. THEREFORE A BYTE CAN REPRESENT TWO DECIMAL NUMBERS.

BIT - ONE SINGLE BINARY DIGIT, EITHER A ONE OR A ZERO.

BUG - AN ERROR OR UNDESIRABLE ASPECT IN A PROGRAM, WHICH PREVENTS A PROGRAM FROM WORKING CORRECTLY OR NOT AT ALL.

BYTE - A GROUP OF BINARY BITS, USUALLY 8, CONSIDERED AS ONE UNIT.

CHARACTER - AN ELEMENT OF A SET OF SYMBOLS, SUCH AS A LETTER OR NUMBER, OR SPECIAL SYMBOL.

CHIP - COMMON NAME FOR INTEGRATED CIRCUIT, DERIVED FROM THE SMALL PIECE OF SILICON ON WHICH THE INTEGRATED CIRCUIT IS CHEMICALLY FORMED.

COMPUTER - A MACHINE THAT ACCEPTS DATA, ACTS UPON IT, AND SUPPLIES THE RESULTS OF THE PROCESSING AS A RESULT OF CERTAIN INSTRUCTIONS. A COLLECTIVE NOUN DESCRIBING THE PROCESSOR AND I/O DEVICES.

CRASH - TERM USED TO DESCRIBE THE COMPUTER 'LOCKING UP' AND NOT ACCEPTING ANY INPUT FROM THE KEYBOARD. THE ONLY SOLUTION IS TO TURN THE COMPUTER OFF AND THEN ON AGAIN.

CURSOR - A FLASHING THIN LINE USED TO INDICATE WHERE DATA IS EXPECTED TO BE ENTERED ON A VDU.

DATA - A PIECE OF INFORMATION WHICH THE COMPUTER CAN PROCESS.

EDITING - THE PROCESS OF CHANGING DATA BEFORE IT IS COMMITTED TO THE PROCESSOR.

EXECUTE - TO CARRY OUT THE INSTRUCTIONS IN A PROGRAM. A MICROPROCESSOR EXECUTES A PROGRAM BY READING AND ACTING ON THE INSTRUCTIONS.

GRAPHICS - TERM DESCRIBING THE DISPLAY OF DATA IN PICTORIAL FORM. PICTURES ON SCREEN ARE DISPLAYED USING PIXELS.

HARDWARE - PARTS OF THE COMPUTER THAT PHYSICALLY EXIST, THE COMPUTER AND A PRINTER FOR EXAMPLE.

HEXADECIMAL - A NUMBER BASE USING 16 DIFFERENT DIGITS FOR EACH NUMBER COLUMN. THE DIGITS 0-9 AND A-F ARE COMMONLY USED.

INSTRUCTION - A CERTAIN ACTION TO BE TAKEN BY THE PROCESSOR. A MACHINE CODE PROGRAM IS MADE UP OF INSTRUCTIONS.

MACHINE CODE - BINARY REPRESENTATION OF THE INSTRUCTIONS OF THE MICROPROCESSOR. MACHINE CODE CAN BE ACTED UPON BY THE MICROPROCESSOR WITHOUT ANY FURTHER TRANSLATION.

MEMORY - COLLECTION OF INTEGRATED CIRCUITS IN WHICH DATA IS STORED. EACH BINARY BIT IS STORED AS AN ELECTRICAL SIGNAL WITHIN THE IC. MEMORY IS CLASSIFIED AS ROM OR RAM AND ITS SIZE IS MEASURED K (KILOBYTES).

MICROPROCESSOR - AN INTEGRATED CIRCUIT THAT CONTAINS ALL THE COMPONENTS TO PERFORM THE BASIC DATA PROCESSING OPERATIONS, ALL IN ONE PACKAGE. A MICROPROCESSOR MUST BE CONNECTED TO MEMORY AND 1/0 DEVICES BEFORE IT CAN BE USED.

MNEMONIC - A GROUP OF 3/4 CHARACTERS REPRESENTING A MACHINE CODE INSTRUCTION. EACH MNEMONIC IS TRANSLATED BY AN ASSEMBLER INTO A MACHINE CODE INSTRUCTION.

NYBBLE - A GROUP OF FOUR BITS. THERE ARE TWO NYBBLES PER BYTE.

OBJECT PROGRAM - A PROGRAM IN MACHINE CODE. THE SOURCE PROGRAM, WHICH CANNOT BE EXECUTED BY THE PROCESSOR, IS ASSEMBLED BY THE ASSEMBLER WHICH GENERATES AN OBJECT PROGRAM. THIS OBJECT PROGRAM RESIDES IN MEMORY, AND CAN BE EXECUTED BY THE PROCESSOR.

OPERATING SYSTEM - A MACHINE CODE PROGRAM, PART OF THE SYSTEMS SOFTWARE, WHICH ENABLES THE PROCESSOR TO PERFORM THE DATA PROCESSING AND CONTROL FUNCTIONS.

PAGE - WHEN USED IN CONJUNCTION WITH MEMORY, MEANS 256 BYTES OF MEMORY.

PROGRAM - A collection of instructions to make the microprocessor perform a certain task.

RAM - Random Access Memory. This kind of memory may be written to or read from. This kind of memory used to store the program that is being developed. If you turn the computer off, all data contained in RAM will be lost.

ROM - Read Only Memory. This kind of memory is set up at the factory where the computer is made. It usually houses the operating system and other programs necessary each time the computer is turned on. Turning the computer off and then back on again has no effect on ROM.

SOFTWARE - A non-physical part of a computer such as a program.

SOURCE PROGRAM - The program that consists of mnemonics that can be understood by humans. This program cannot be executed until it is assembled.

New Generation Software Ltd.
Bath,
Avon.