SECTION 1 - Computer Description

List of Contents

	Page
Introduction	1: 1
Peripherals	1: 1
Software	1: 1
Core Memory	1: 3
Accumulators	1: 4
Carry Flag	1: 4
Greater Than Flag	1: 5
Program Counter	1: 5
Memory Address Register (MA)	1: 5
Memory Buffer Register (MB)	1: 5
Instruction Register	1: 5
DCH Memory Address	1: 6
Base Register	1: 6
Limit Register	1: 6
Addressing Mode Memory	1: 6
Arithmetic Logic Unit (ALU)	1: 7
Phase Control Register	1: 7
Memory Organisation	1: 9
Auto-Increment and Auto-Decrement	1: 9
Interrupt Stack Pointer	1:10
Interrupt Addresses	1:10
Input/Output Hardware	1:11
Octal Notation	1:12

COMPUTER DESCRIPTION

Introduction

The BCL Molecular 18 enhanced version is a high speed, modular processor designed to provide maximum computing power at minimum cost to the user. It is a general purpose computer system able to operate in a normal office environment; the cycle time is 1.2, 1.4 or 1.6 microseconds and an 18-bit word length is incorporated, the 18th bit being used as a parity check.

The primary storage of the computer is a "core memory", composed of magnetic cores which record binary information by the direction in which they are magnetized. The core memory system is organised into Pages, each containing 1024 words (1K). There are several sizes of core available ranging from 4K to 32K in modules of 4K.

Peripherals

The input/output hardware allows a program to address up to sixty-three peripherals, and Section 4 of this Manual describes many of these in detail. The following list shows some of the peripherals available with the enhanced version :-

various types of input/output writers
paper tape or card input/output
discs
magnetic tape units
optical mark readers
etc.

Software

The Molecular 18 enhanced version is supported with a programming package to ease the task of the programmer. Basic software programs include the Assembler, Operating System, Linkage Editor, Library Update and Debug.

Assembler

The MK.II Assembler allows the user to write efficient and time-saving symbolic programs in modular form, which are translated by the Assembler into Object Modules. The Assembler accepts a source module consisting of user-coded instructions, and outputs it as an object module which can then be linked to Library modules, etc. by the Linkage Editor.

The Assembler is designed to operate with a minimum of 16K of core memory, and can be adapted to almost any configuration of I/O devices to optimise assembly speed. In addition to op-code mnemonics, Assembler Mk.II has the capability of accepting user symbols, and has a flexible constant definition format.

Operating System

The operating system provides an efficient loading and input/output control capability for relocatable programs produced by Assembler Mk.II. The system is modular in design and may be constructed to fit each user's hardware configuration. Its general purpose is to take over from the operator and programmer the execution of a number of ordinary error-prone tasks, and to organise the running of each program in the most efficient way possible. The main functions of the operating system are to :-

a) Monitor all peripheral transfers
b) Organise and control multi-programming
c) Control, execute or initiate all communications with the operator via a control peripheral device.
d) Execute Extracode functions.

Linkage Editor

The Linkage Editor provides a means of combining the relocatable object modules generated by Assembler Mk.II with each other, automatically linking in any required Library Modules, to form a loadable program.

Library Update

The Library is a set of standard modules for use by any program. The Library can be created and updated by use of the Library Update routine, which can also delete any out-of-date library modules.

Debug Package

Debug is an interactive octal debugging aid which contains many valuable program testing features. Among these are the display and modification of memory locations and/or the accumulators. Also included is a breakpoint feature which allows the programmer to set, reset and recognise breakpoints throughout his logic and enter his program at any point.

COMPUTER DESCRIPTION

The basic Molecular 18 system includes the Central Processing Unit or CPU, the core memory and an input/output coupler to connect the processor to the peripheral chassis.

The CPU is the control unit for the whole system: it governs all peripheral in-out equipment, performs all arithmetic, logical and data handling operations and also sequences the program. It is connected to the core memory by a memory bus and to the peripheral equipment by an in-out bus. In order to store, retrieve, control and modify information and to perform the required logical, arithmetic and data processing operations, the core memory and the CPU employ the logic complement described in the following paragraphs :-

Core Memory

Core stores are used primarily for holding program, but also provide a fast random-access storage medium for data to be processed or distributed. Each core stack used in the processor has a 4K capacity, holding 4096 18-bit words. A word of 17 binary digits or bits (plus a parity bit which may be disregarded) can represent signed or unsigned binary quantities from Ø to 1111111111111111 (i.e. 65,535 in decimal). The bits of a word are numbered 17 to 1, left to right, as are the bits in the registers or accumulators that handle the words. The 'leftmost' 1 in a binary number is called the most significant digit, the least significant digit being the extreme right digit (Bit 1).

Core stores used for numeric data use Bit 17 as a sign bit, its presence indicating that the number is negative (see Fig.1.1 below).

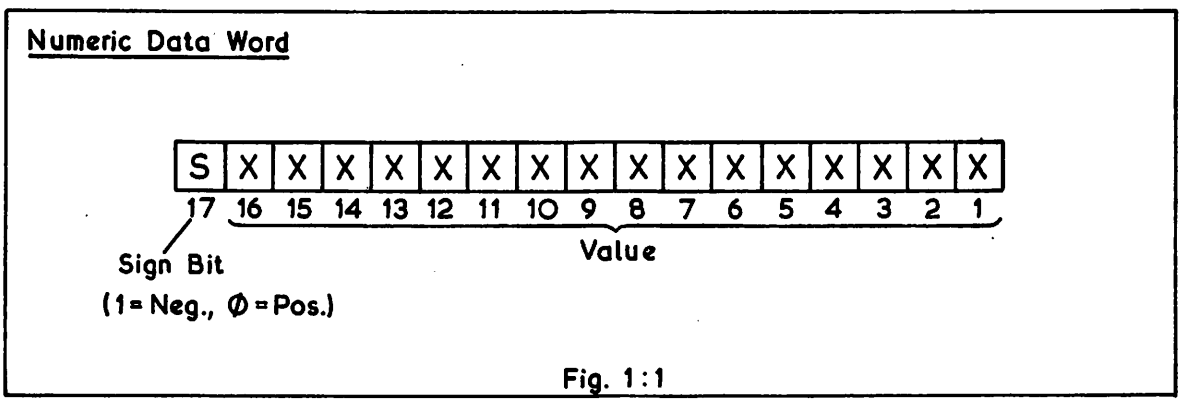

Fig. 1:1

Accumulators

Two internal registers of importance to the programmer are the 17-bit accumulators, Registers A and B, which are used by program for logic and arithmetic operations and can also be addressed as Steps 00 and 01 respectively on Zero Page. It is important to note that these are in fact flip-flop hardware registers and not part of the core storage. They are registers each capable of storing a number made up of sixteen binary digits plus a sign bit, and are called accumulators because they can accumulate partial sums during the operation of the computer. Data can be moved in either direction between any core memory location and either accumulator. Although words in the core memory can be incremented or decremented, all other arithmetic and logic operations are performed via the accumulators, the accumulator receiving the result. In addition, all programmed information transfers between the processor and the peripherals are passed through these registers, with the exception of data-channelling.

Accumulator A - this holds the results of arithmetic and logical operations such as Boolean logic operations, comparison for equality with any core store, shifting or rotating of bits left or right, testing individual bits, complementing, swapping the left and right halves (Bits 16 to 9 with Bits 8 to 1) and testing the contents for a skip. It is also used for transfers to and from peripheral devices, and by the Privileged Instruction 'Mask' for masking out.

Accumulator B - this has almost the same capabilities as Accumulator A, except that the three Boolean logic instructions (ANDA, IORA and XORA) and also the Privileged Instruction 'Mask' can apply only to Accumulator A, while the Privileged Instruction 'ACKINT' can apply only to Accumulator B. Also, on a JSBR instruction, the return address is always placed in Accumulator B.

Carry Flag

Associated with the Accumulators is the Carry flag (sometimes referred to as the Overflow). This is a one-bit register which is used to extend the arithmetic facilities of the Accumulators. When it is in a non-zero state it could indicate either that a carry has occurred from Bit 16 to Bit 17 of an accumulator, or that an overflow has occurred from Bit 1 downwards (as in Right Shift, see Page 3:25). Under program control the Carry flag may be cleared, complemented, tested or rotated as part of an Accumulator.

Greater Than flag

There is also an indicator known as the Greater Than flag, which is associated with the CMPA and CMPB memory reference instructions (see Page 3:14). Under program control, this flag may be tested for, cleared or set by certain Register instructions.

Program Counter (PC)

The PC is a 16-bit register which is used to control the program sequence; in other words, the order in which program instructions are performed is determined by the PC. The PC register contains the current relative program address, the address of the core memory location from which the instruction will be taken. When the instruction has been completed, the PC is incremented by one, the information in the PC then being transferred to the Memory Address Register (MA) to determine the core memory address from which each instruction is taken. Sequential program flow can be altered by changing the contents of PC, either by incrementing it by two, as in a test skip instruction, or by replacing its contents with the address specified by a JUMP or JSBR instruction (see Page 3:5).

Memory Address Register (MA)

The MA is a 16-bit register which is used for absolute core addressing. When using the Operating System, the MA = PC + Base. Bits 12 to 1 are used for selection of core words, while Bits 13 to 16 are decoded to select the correct core stack. The MA will differ from the PC during the processing of multi-phase instructions, since the MA will be changed by memory references in the instruction (which may be several in the case of indirect addressing), whereas the PC remains constant until completion of the instruction.

Memory Buffer Register (MB)

The MB is an 18-bit register that is used for all information transfers between CPU registers and core memory. The 18th bit is used for parity checks.

Instruction Register

This is a 5-bit register which contains the operation code (Bits 17 to 13) of the instruction currently being performed by the machine. The five most significant bits of the current instruction are loaded into the Instruction Register from the MB. The contents of the Instruction Register are decoded to produce the conditions for program execution. This register is not accessible by program or via the control panel.

DCH Memory Address

This is a 16-bit register which is used purely for core access during data channelling; it is not accessible either by program or via the control panel.

Base Register

This is a 16-bit register which contains the start address of whichever program is running in the machine at the time. The Executive always occupies an area of core starting from the beginning of the Zero Page. However, each program written for the machine is assembled as if from Address Zero also, so that all memory instructions will refer to addresses 'relative' to the start of the program. In order that the hardware can convert these 'relative' addresses to actual machine addresses, the absolute address is obtained by adding the contents of the base register to the relative address. In normal operation the base register will be set to any value which is a multiple of 128.

When an interrupt occurs, the base register is set to zero after stacking.

Limit Register

This is a 16-bit register which, on Unstacking, is loaded with Word 00 of the program, which always specifies the length of the program at present running in the machine. If a program attempts to access a relative address which exceeds this limit, an internal interrupt will be called, eventually causing the program to go to location 000214 (see Page 1:10).

Addressing Mode Memory

This is a one-bit memory which is provided for purposes of indicating and controlling the addressing mode, such that absolute addressing or relative addressing may be achieved. The Addressing Mode Memory will be set either to 1 or Ø :-

Privileged Mode (i) When set at 1, absolute addressing is achieved (and Privileged Instructions enabled).

User Mode (ii) When set at Ø, the relative addressing mode is achieved, such that any address contained in an instruction has the contents of the Base Register added to it, and is further checked against the set 'Limit' before that instruction is carried out. Should an instruction be found to exceed the set limit, an internal interrupt occurs.

When mains is first switched on, the Addressing Mode Memory will be set to the 1 (absolute addressing) state. Thereafter :-

a) When any interrupt occurs, the contents of the Addressing Mode Memory is propagated into the sign bit (Bit 17) of the 2nd word of the appropriate Interrupt Stack. The Addressing Mode Memory itself will then be set to the '1' state.

b) The 'Unstack' process will use the sign bit of the 2nd word in the appropriate Interrupt Stack to set the Addressing Mode Memory back to the correct state.

Arithmetic Logic Unit (ALU)

The ALU is that part of the processor which can perform various logic and arithmetic functions on the data retrieved from store locations addressed by the PC; the operands are located in the accumulators and/or core memory. Any operands specified are fetched from core or accumulators, the operation carried out and the results then stored. Arithmetic data is fixed point and is treated as a signed sixteen-bit integer. Extended precision arithmetic may be accomplished via the Library software routines. Negative numbers are represented in two's complement form with a sign bit of one, the numeric value of zero, however, always being deemed non-negative.

The time taken to 'fetch' a word of data from core, operate on it and to regenerate it in core is called the machine cycle time; for the Molecular 18 processor a typical time is 1.6 μs. Each machine cycle enables a data transfer from and to core of one word, the significance of which is determined by the setting of the Phase Control Register, which governs the application of this data :-

Phase 0 - Idle - There is no processing in this phase; the machine waits for a Start signal from the Control Panel switches (see Page 2:2) to continue processing. The Load/Examine switches may be operated in this phase (see Phases 6 and 7 below).

Phase 1 - Fetch - Assuming the computer is in a fully automatic, running condition, core access in this phase results in a word transfer from core to the Memory Buffer and Instruction Register, from where it is decoded to set the required processing conditions for successful operation. During the 'Fetch' phase, Control, Register and I/O instructions may be performed, as no further access is required. At the end of this phase for these instructions, the PC is incremented and the CPU restarts Phase 1 for the next instruction.

For memory instructions, further core cycles must be initiated to retrieve the operand as specified by the address in Bits 10-1 of the instruction. However, if Bit 12, the indirect bit, is set, an additional core cycle is required prior to execution to locate the true address of the operand. This will cause Phase 2 to set.

Phase 2 - Indirect - This phase is used if a binary 1 is present in Bit 12 of a memory reference instruction decoded in Phase 1. This state causes the CPU to obtain the full 16-bit address of the operand from the address in either the current page or Page Zero, as specified by Bits 11 to 1 of the instruction. Should the word accessed during this phase have the indirect bit (Bit 17) set, Phase 2 will be repeated; this can go on almost indefinitely, except that if more than 15 indirects are specified an internal interrupt will be called, causing the program to go eventually to location 000213. If the indirect bit of the accessed word is not set, the processor will set Phase 3 at the end. The one exception to this is an Indirect Jump or JSBR which can be executed in Phase 2 and will then return to Phase 1 for another instruction.

Phase 3 - Execute - The Execute phase is entered for all memory reference instructions except JUMP and JSBR. Memory instructions may be completed in this phase after accessing core for an operand or modifying a core word, depending on the instruction to be performed. At the end of this phase the PC will be incremented, the processor then returning to Phase 1 for the next instruction unless there is an Interrupt or Data Channel Request pending.

Phase 4 - Interrupt - When an interrupt occurs, this phase sets the Memory Address Register to 000030 to access the Interrupt Stack Pointer address, which is read from core and sets the MA to the address of the current Interrupt Stack. The next five machine cycles are taken up by the stacking of the Interrupt Stack, the Interrupt Stack Pointer address is then incremented by five, and the MA set to the appropriate Interrupt Address (see Page 1:10).

Phase 5 - Data Channel - Devices requiring high speed I/O data transfers may 'steal' machine cycles at any time during processing by raising Data Channel Request, which for one machine cycle enables one core word to be accessed for direct transfer to or from a device. Other control lines determine the direction of data flow and the core address involved.

Phases 6 and 7 - Control Panel - These two phases are used for Control Panel Load/Examine operations (see Page 2:4), and can only be reached from Phase 0. i.e. while there is no processing in progress. When the Load/Examine Mem. Next switch is operated, Phase 6 increments the MA, and then sets Phase 7 to access core.

Memory Organisation

Some areas of memory are dedicated to certain hardware and software functions when in the Absolute Addressing Mode. The following paragraphs described these areas, giving their purposes and specific locations in octal :-

000000 Accumulator A, as described on Page 1:4

000001 Accumulator B, as described on Page 1:4

000002 The Memory address for Auto-Start.

When power is restored to the Molecular 18 after a Mains fail, interrupt is off.

If the Key switch is in Normal Mode, PC and MA will both be set to 000002 and the program will immediately commence from this point. If Interrupt is then turned on without first performing the instruction 'Skip if Mains Return Interrupt' (see Page 3:21), a Mains Return Interrupt will occur, which will eventually send the program to the Interrupt Address 000207 (see Page 1:10).

If the Key switch is in Manual Mode, PC and MA will be set to 000002 as above, but the machine then awaits further orders.

000010 to 000017

When a location between and including 000010 and 000017 in Page Zero of the core memory field is addressed indirectly (when in Absolute Addressing Mode) by any Memory Reference instruction except JUMP and JSBR, the contents of that location are taken as the effective address of the instruction, then incremented by one and rewritten in the same location. This feature is called Auto-Increment.

e.g. If location 000012 contains the octal number 001234, and the instruction STA I 000012 is given, the contents of Accumulator A will be deposited in core memory location 001234 and the number 001235 then rewritten in location 000012.

000020 to 000027

When a location between and including 000020 and 000027 in Page Zero of the core memory field is addressed indirectly (when in Absolute Addressing Mode) by any Memory Reference instruction except JUMP and JSBR, the contents of that location are taken as the effective address of the instruction, then decremented by one and rewritten in the same location. This feature is called Auto-Decrement.

000030 - The Interrupt Stack Pointer

Associated with each interrupt line (level) is a 5-word set of consecutive memory locations situated anywhere in the core memory. Accordingly, on the Molecular 18 enhanced version, 40 words of core are reserved as there are eight interrupt levels. These reserved core addresses are referred to as 'the Interrupt Stack' and are accessed from the Interrupt Stack Pointer at 000030. When the processor honours an interrupt, it saves the following information in the appropriate position in the Interrupt Stack :-

Word 1 - Carry and Greater Than flags (Bit 1=Carry, Bit 2=G.T.)
Word 2 - Program Counter (+ Addressing Mode in Bit 17)
Word 3 - Base Address
Word 4 - Contents of B Register
Word 5 - Contents of A Register

The Interrupt Stack Pointer address (000030) is incremented by five, and the computer then causes the next instruction to be read from a fixed memory location, according to the type of interrupt it was. This location in memory is referred to as the 'interrupt address', and is reserved for that particular type of interrupt. A routine can thus be entered to perform the process required by the interrupt. In the enhanced version of the Molecular 18, the built-in Executive program will handle all hardware interrupts.

000205 to 00217 - Interrupt Addresses

000205	Continuous Interrupt Switch Interrupt
000206	MA=SW Interrupt
000207	Mains Return Interrupt
000210	Extra Code Interrupt
000211	I/O Interrupt
000212	Illegal Operation Interrupt
000213	More than 15 Indirects Interrupt
000214	Memory Boundary (Limit) Interrupt
000215	Memory Parity Interrupt
000216	Mains Fail Interrupt
000217	Overstack (Stack Full) Interrupt

Input/Output Hardware

The input/output hardware allows the program to address up to sixty-three peripheral devices, which are connected to the CPU through a highway known as the input/output bus. The 'bus' system allows one set of data and control lines to communicate with all I/O devices; no additional connections to the computer are required.

The peripheral devices are 'slow' compared with electronic operations within the CPU. The control unit deals with them by starting them and then turning to another instruction while waiting for them to complete their operation. A single instruction can transfer a word between an accumulator and a device and at the same time control the device operation. Each device is linked to the I/O bus by an interface board, which usually holds buffers for temporary storage of data in or out. Through this buffering, Input/Output can be overlapped with processing to take full advantage of available processing time. The devices empty or fill these buffers at their own pace, the buffers then transferring their contents at very much higher electronic speeds to or from the Accumulators; this avoids the tying up of a working register during the relatively 'long' transfer periods. The actual number of buffer bits (up to 17 flip-flops) will depend on the device for which the interface is intended. If the buffer is less than 17 bits in length, data is transferred to or from the least significant bits of Accumulators A or B (bits 8-1). Data is transferred one character at a time in most cases, between the interface buffers and core via the Accumulators, the transfer logic being controlled by the CPU.

Included in the in-out system, as mentioned before, are facilities for program interrupts and high speed data transfers. The interrupt system facilitates processor control of the peripheral equipment by allowing any device to interrupt the normal program flow on a priority basis.

While most devices are addressed directly by the CPU, disc drives and magnetic tapes are accessed through a device controller, controlling up to four devices, although only one of these may be accessed at any one time.

High-speed devices such as disc, magnetic tape, line-printer, can gain direct access to memory under data channel control. This is connected to the appropriate devices through interface boards which contain buffers and also addressing logic, allowing more than one character at a time to be transferred directly to or from core without using the accumulators. The CPU carries out the transfer by 'stealing' machine cycles during instructions without disturbing program execution.

E.C.M.A. (European Computer Manufacturers' Association) No.6 code is used wherever possible for the exchange of information between various peripherals and between peripherals and processor. This code is the U.K. version of I.S.O. No. 7 and I.T.A. No. 5., all these codes being based on ASCII.

Peripheral devices which operate directly connected to the CPU via the I/O bus are said to work 'on line'. No operations at all may be carried out 'off program'. The input and output on writers are completely separated, and no writer can be used when the CPU is switched off. Likewise, no regeneration of tape can be achieved 'off program'.

Octal Notation

It will be apparent that the binary number system, although suitable for computers, is rather complex from the programmer's point of view. To overcome this problem the Octal or Base 8 numbering system can be used to make the 17-bit words of the Molecular 18 easier to handle; indeed in Debugging or in object program print-outs all references to instructions or addresses are made in octal notation. Each binary word is divided into 6 groups, 5 3-bit and 1 2-bit, each group being represented by a single octal digit, using the table of equivalents shown in Fig. 1:2 :-

Decimal-Octal-Binary Equivalents

Decimal	Octal	Binary
0	0	000
1	1	001
2	2	010
3	3	011
4	4	100
5	5	101
6	6	110
7	7	111

Fig. 1:2

A binary number 1101111010111101 is separated into groups as stated previously, by starting from the least significant digit end (the extreme right hand end) of the number, and supplying leading zeros if necessary.

i.e. 01 101 111 010 111 101

Each binary group is then replaced by its octal equivalent :-

01	=	1
101	=	5
111	=	7
010	=	2
111	=	7
101	=	5

the binary number 1101111010111101 thus being converted to its octal equivalent, 157275.

An octal number can be expanded to a binary number using the same system and table of equivalents.

e.g. 105307 Octal = 01000101011000111 (Binary)

Note :-

It should be remembered that only binary numbers are recognised by the computer, and that the Octal representation used for purposes of simplicity does not itself exist within the computer.