A.C.C NEWSLETTER VOL 2 ISS 3 AUG 74

THE 8008 ISSUE

This article describes the construction of a general purpose desk calculator I have built over the last few months. As a microcomputer it is able to perform any calculation on binary data so many games (e.g. LIFE) can easily be programmed.
Subroutines can be added (in a PROM) for computations with numeric data. The machine will then be able to solve all arithmetic problems and can be used as a mini.

Data can be entered by the 32 key keyboard, some mode switches, a paper tape reader and through the 8 bit input port which accepts signals with TTL levels from any source. Data output occurs through the 9 digit (8 digit + sign) display or through the 8 bit output port which is displayed on LED's and made available for external equipment (e.g. a TTY) with a TTL fan-out of 4. Input and output capacity can be easily expanded.

Cost

The whole system uses 7 10x12 cm boards. They contain 84 IC's - 58 SSI TTL, 17 MSI TTL, 2 low power TTL, 9 MOS LSI, 2 linear and a handful of resistors, capacitors and transistors. Overall costs including case, keyboard, display, power supply etc. were less than DM 2000 (about £ 300).

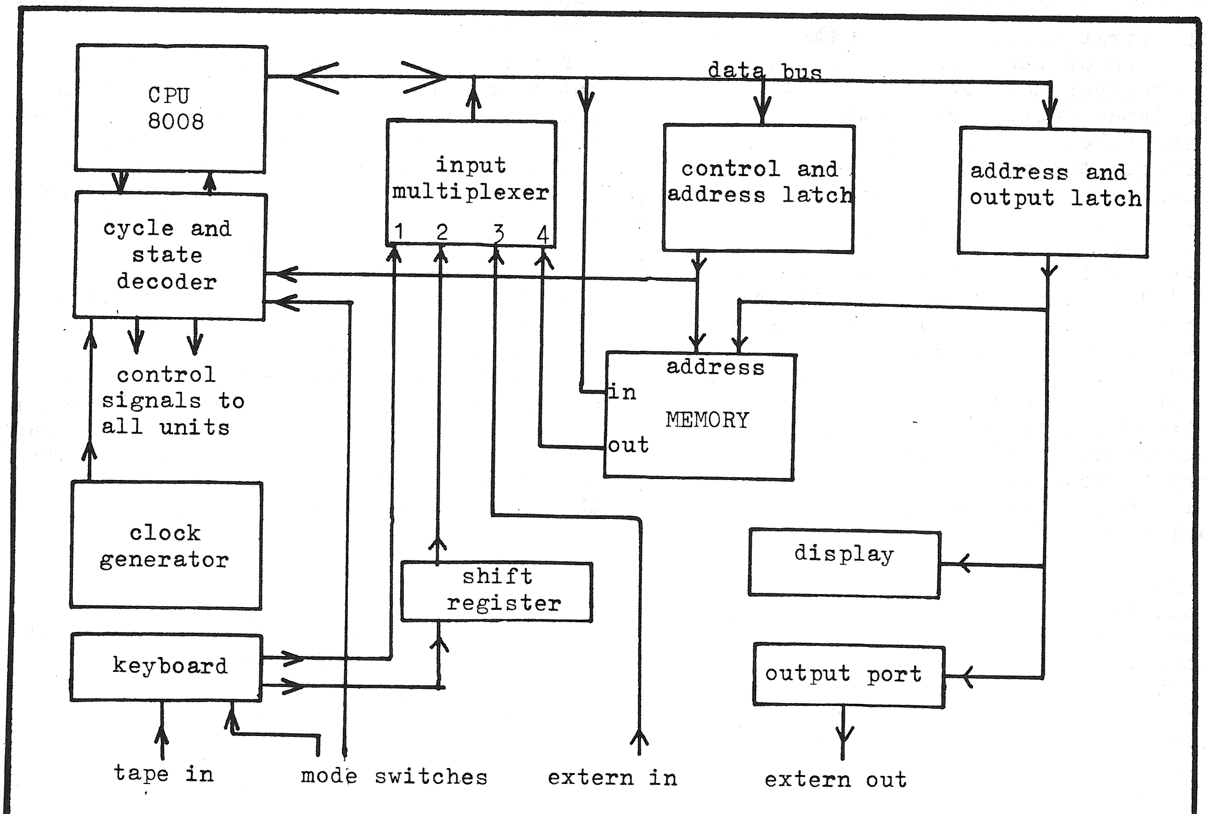
8008 CPU

The heart of the system is INTEL's eight bit 'computer on a chip', the 8008. This is an 8 bit parallel CPU with a repertoire of 48 instructions, seven working registers, an 8 level address stack, and the ability to address 16K bytes of memory directly.
Instructions include data manipulation, binary arithmetic and 'Jump to Subroutine'

For detailed information ask Walmore Electronics Ltd., London 01-836-0201 for the "MCS-8 Micro Computer Set Users Manual.

Execution of an instruction

The completion of an instruction requires one, two or three machine cycles. Each cycle consists of three to six states, each state of two clock periods. The 8008 works with a 500KHz clock so that the execution of a single instruction will take 16 - 44 uS. The processor communicates over an 8 bit data bus and uses two input leads and four output leads for control. Time multiplexing of the data allows control information, 14 bit addresses and data to be transmitted.



Block Diagram

State	Activity of CPU	Activity of csd
T1	sends out lower bits of address or output data	enables address and output latch
T2	sends out higher bits of address and/or control	enables control and address latch
wait	does nothing	waits for keyboard
T3	fetches instruction or data or sends data for memory	enables input multiplexer or write line of memory
T4	executes instruction	does nothing
T5	executes instruction	does nothing

Timing

The first cycle of all operations is an instruction fetch cycle. During its first state (T1) the 8008 sends out the lower 8 bits of the memory address of the instruction on the data bus. Like all states it is recognised by the 'cycle and state decoder' csd (see block diagram) which does all of the timing. It enables the 'address and output latch' to store the first part of the address.

During the following state (T2) the data bus contains the high 6 bits of the address and two bits for control. The csd now enables the 'control and address latch'. The whole address is now available for the memory. The two control bits which were also transmitted inform the csd about the type of cycle which is to be executed, which can be one of three types;

1) read instruction or data from mem.
2) write instruction or data into mem.
3) perform an I/O operation.

As the first cycle is an 'instruction fetch' it is of the first type. Depending on the stop/run mode switch the csd selects input 2(stop) or 4(run) of the input multiplexer. For normal execution of a program the switch will be in the 'run' position and data is fetched from memory. In the 'stop' position instructions come from the 8 bit serial in - parallel out shift register. Pressing keys '0' or '1' of the keyboard enters the corresponding logic levels into the reg. Thus it can be loaded with the code of any instruction by 8 key strokes. As the CPU is interrupted while the mode switch is in the 'stop' position the address is not incremented. This allows execution of programs by hand and even programming if the LMI (Load Memory Immediate with data) instruction is used.

The next state is called 'wait' and is optional. It is only carried out if the mode switches are in the 'stop' or 'step' position. In this case the CPU will remain waiting until a special key is pressed to finish the operation. The 'input multiplexer', the 'control and address latch' and the 'address and output latch' are displayed by LED's. The contents of the shift register are displayed when the program is stopped, and the memory address with its corresponding contents are shown while stepping. This makes it easy to find program errors.

The following state (T3) is used to enter the instruction into the CPU. The csd enables the three state output buffers of the input multiplexer as long as T3 is on. The last two states of the cycle (T4 and T5) are needed for the execution of the instruction. Some operations do not require these states and then they are omitted.

Most instructions are complete after the first cycle, but some require two or three. The 'Jump' instruction for instance needs two more cycles for transmitting the address at which the program will continue. Since they are also of the 'read instruction or data' type they are carried out in exactly the same way as the first cycle.

If a 'load memory' operation is executed the csd will recognise the second cycle as the 'write instruction or data into the memory' type. The address of the memory location to be loaded is sent out during T1 and T2 and the data is put on the data bus at T3. At that time the outputs of the input multiplexer are disabled and the 'write' line of the memory is activated.

If the second cycle of an input or output is performed the 'address and output latch' receives the output data at T1. Eight bits of control are stored in the 'control and address latch' during T2. The csd distinguishes between 2(input) and 4 (output) operations. For input operations input 1 or 3 of the input multiplexer is selected and input data is transmitted during T3.

Input 3 is available for external equipment. Input 1 is used for the keyboard. The keys are debounced, binary coded, and stored in a latch. An additional expander input allows for data entry from a paper tape reader.

For an output operation the keyboard latch is cleared,a flip-flop which is displayed by a LED (e.g. as an overflow indicator) is set, the output data is transferred to the output port register or the display is activated. The display logic is multiplexed by the CPU. It contains a digit counter which is automatically advanced with each output display operation and is connected to a decoder/driver to enable one of nine LED numeric displays. The output information is stored in a five bit latch (data and decimal point), decoded to seven segment and amplified to provide the high current required. If the CPU leaves the display loop the counter is reset after 400uS and the display is blanked.

Memory

The memory consists of eight 1024 bit RAM's. They are available from most semiconductor manufacturers (Intersil IM7552, Intel P2102, Microsystems MF2102, Signetics 2602 etc.) These memories are TTL compatible in inputs, outputs and single 5V power supply, static (no clocks or refresh) and the output is tri-state. If the cost of the external logic required for dynamic systems is taken into account these are the cheapest semiconducter memories and they save the user a lot of work.

Memory may be expanded up to 16K with any mix of RAM, PROM or Shift Register.

8080 CPU

Meanwhile Intel have developed an improved version of the 8008 - the 8080. It has many advantages and a lot of new extremely useful instructions. (See ACCN Vol 1 Iss 3 page 1). All new designs should be made using the 8080.

Questions

If you have any questions or if you are interested in detailed wiring diagrams of my machine write to;

T. Knauf
24 Luebeck
Hamburger Str. 85
West Germany

LETTERS

RECURSION COMEBACK

RECURSIVE PROGRAMMING

I must take up the gauntlet thrown down by our editor and treasurer in the June '74 issue. 'Can anyone suggest a real application for recursion?' he asks. I think I can.

Firstly, recursion is particularly useful in a list-processing environment, where it is easy to set up the necessary mechanism for recursive calls and passing of arguments. My first example is therefore a list sort, to sort the items on a list into 'key' order. The method is to choose an arbitrary item on the list and divide the list into three sublists, one, list A, with items with smaller keys than the chosen item, one, list B, with items with equal keys, and one, list C, with items whose keys are larger than the chosen item. If list A has more than one item, it is sorted in the same manner (ie recursively), and similarly list C if it has more than one item. List B, of course, has items of equal keys, and does not need further sorting. List C is then appended to list B which is appended to list A, and the sort is complete. I believe that this algorithm, called Quicksort, is due to C A Hoare.

Secondly, I have used recursion in a computer-aided logic design program. The technique was to take a general combinational expression and recursively factorise by a small number of variables (two or three), generating the necessary logic, until only '0's and '1's remained.

Thirdly, I happened across an application of recursion in a most unexpected place: an octal number output routine - but perhaps I'm cheating here, as this was within an implementation of BCPL. Here it is, anyway, in a language called TASC.

```
PROCEDURE WRITEO(OCT, WIDTH);
CONSTANT L13 = 017777, L3 = 07;
      IF WIDTH>0 THEN
           (    WRITEO((OCT>>3 & L13),
                           (WIDTH-1));
                WRITEC((OCT&L3 + "0"))    );
END;
```

note that (and) also serve as begin and end in ALGOL 60, >> means shift right, & means logical AND, and WRITEC is the character output routine.

I don't know anything about the history of the discovery (invention?) of recursion, but I suspect that is was initially acclaimed in the way we now talk about structured programming.

Lest anyone gain the contrary impression, my main interest is in hardware, particularly the control functions, the providing of control signals to registers, store, peripherals and so on. Is there anyone with similar interests?

Alasdair Rawsthorne

COMPUBOOK

A few more books on computers;

K.J.Deane - An Introduction to Counting Techniques & Transistor Circuit Logic. Chapman & Hall (I think).

J.B.Dance - Electronic Counting Circuits Illiffe (mainly cold cathode devices).

J.C.Bartee - Digital Computer Fundamentals. McGraw Hill

R.M.M.Oberman - Electronic Counters McMillan

E.H.W.Hersee - A simple Approach to Digital Electronic Computers. Blackie.

D.Metcalf

16 BITTE

The German periodical "Elektor" started a series of articles called "Computer 74" with edition 42 May 1974. They describe the hardware of a 16 bit microcomputer which can be built up with standard TTL devices. Storage may be a semiconductor memory or even a normal tape recorder. The machine is suitable for demonstration and training but may also be adapted to other applications. If you are interested in copies (no charge) write to:

T. Knauf
24 Luebeck
Hamburger Str. 85
West Germany

** SUPPORT THE ACC **

WANTED

MEMBERS & ARTICLES

SIN IS WRONG

There is an error in the flow diagram for sin(x) in Vol 2 Iss 2.

In the section which performs range reduction from $\left(-\frac{PI}{2}, \frac{PI}{2}\right)$ to $\left(-\frac{PI}{4}, \frac{PI}{4}\right)$

use is made of the equation;

$$SIN(x) = \sqrt{1 - SIN^2(x \pm \pi/2)}$$

however this equation is valid only for positive x within the range $-\pi/2$ to $\pi/2$, for negative x the negative square root should be used. Apossible modification to the flow diagram is illustrated in Fig. 1.

Another point is the use of a value (PI= 3.14159) which is an approximation to an irrational number (π). Perhaps it should have been made clear for those with little experience in such areas that numbers like π ,e, $\log_e 10$ etc. should be incorporated into programs to the maximum precision possible.

Finally , the flow diagram illustrates a convential and inefficient method for computing the MacLaurin series. An alternative method can be obtained by re-arranging the sin and cos series to give

$$\cos(x) = 1 - \frac{x^2}{1\times2}\left(1-\frac{x^2}{3\times4}\left(1-\frac{x^2}{5\times6}(etc)\right)\right)$$

$$\frac{\sin(x)}{x} = 1 - \frac{x^2}{2\times3}\left(1-\frac{x^2}{4\times5}\left(1-\frac{x^2}{6\times7}(etc)\right)\right)$$

setting $y= -x^2$ gives the recurrence relationship

$$R := 1+(y\times R)/(J\times(J-1))$$

valid for both series. The only difference lies in the initial and final values of J and the need for the extra calculation

$$\sin(x) = x \times (\sin(x)/x)$$

This method (Fig 2) can be very easily used with a calculator, particularly if the machine has a constant in which $-x^2$ can be stored, and so may be of interest to ACCN readers.

D.England

FIG 1

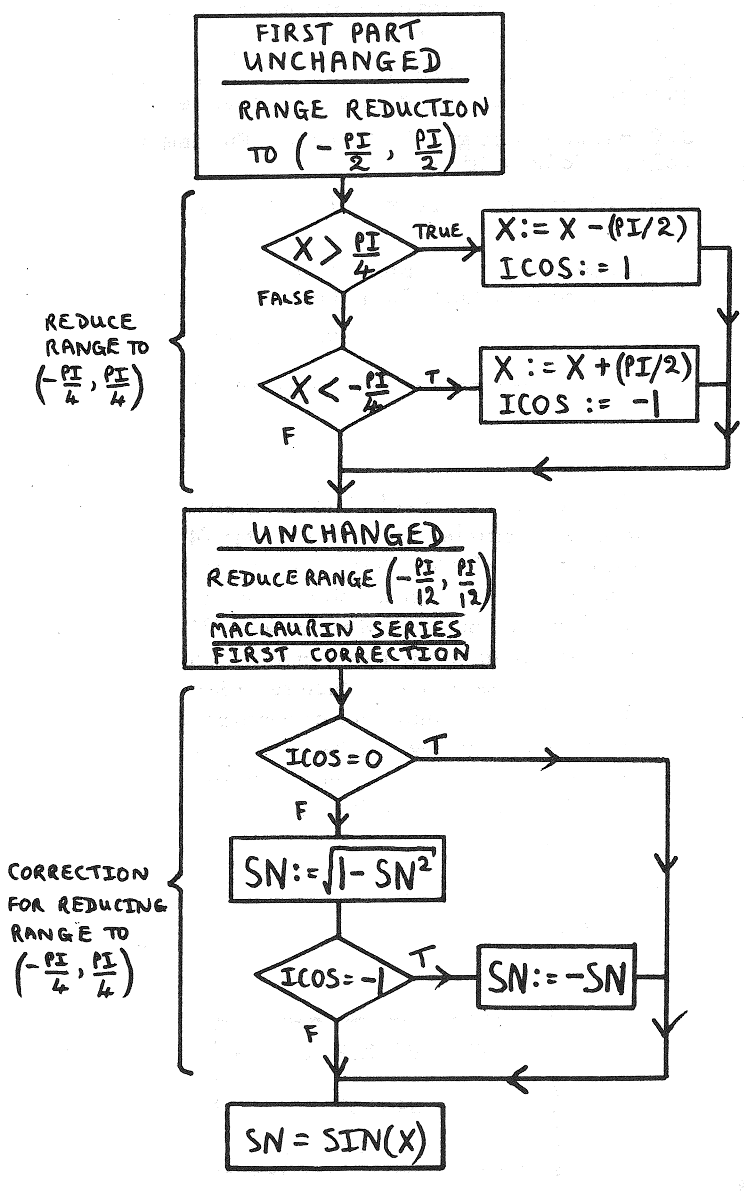
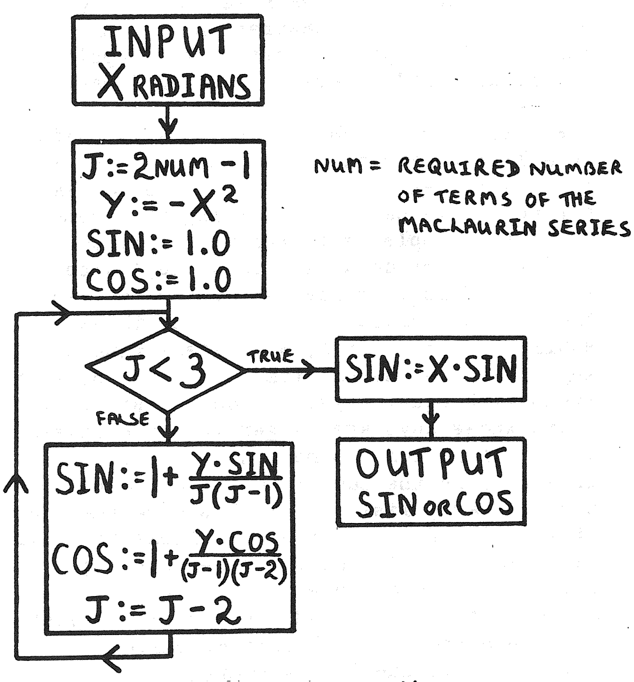


FIG 2

MACLAURIN SIN OR COS SERIES



SIN SERIES ERROR $\leq \frac{|X|^N}{N!}$ $(N = 2NUM + 1)$

COS SERIES ERROR $\leq \frac{|X|^{N-1}}{(N-1)!}$

IF $|X| \leq 1$ THEN SERIES ERROR $< 10^{-10}$ FOR NUM = 7

LET'S GET TOGETHER IN SOUTH LONDON

As you will have read in ACCN the question of local groups has been raised on several occasions at Committee meetings and it is really the lack of information about what our members want which has delayed the formation of such groups. On the other hand, it is difficult to find out what people want without personal contact. So let's break the vicious circle and start a group in South London.

Ring me at 01-674-1205 if you want to join a group of ACC members in South London interested in both software and hardware.

What can we start out with? To begin, software is no problem; as a lecturer at the Polytechnic of the South Bank I have access to computing equipment which can be used. We may have to form ourselves into an official course to straighten things out with the ILEA, but that is no problem. Incidentally did you realize that you have access to the educational computing equipment of most local education authorities if you form yourselves into an evening class? Of course you will have to pay the fee of 75p per term (5p if you already follow a class), but then by appointing one out of your midst as the lecturer you can recoup your fee out of his salary!

Back to South Bank. The equipment is as follows: Remote terminal (LP, CR, TP, TR) to 1905E (64K, 3 EDS, 4 MT, Plotter); 5 TTY (Basic to OU HP); and more interesting, GT44 = PDP11/40, 16K, 2 EDS, TTY, PDP11/5 + Mickey Mouse terminal. What is the proper name for a VDU which allows you to erase selectively? We have Alpha Scopes, we have Storage Tubes, so what do you call these things that allow you to give cartoon shows?

I am insufficiently knowledgeable (but very interested) in hardware to suggest projects in this area though there must be many. It would be particularly nice if we could work at the interface of hardware and software. My view of the computing world is that of course everybody knows about soft ware. Perhaps a similar mistaken notion exists among the hardware fraternity. If that is so, let's join forces.

Hope to hear from you soon!

Jaap Creutzberg
135 Thornton Road, London SW12 OLJ

Software

Jaap Creutzberg

Like many terms in computing, the word software means different things to different people. I don't apologize therefore for giving my definition; indeed this article is largely about the concept of software which is a useful one so long as it is not equated with the notion of a program.

In the first programmable computers you wrote your program in the machine's wired-in language and the program performed a job which was the end product, e.g. trajectory calculations. The machine's language (machine code, m/c for short) is difficult to use because it forces the programmer to think in bit patterns. The programmer would therefore design his program in mnemonic form and translate this by hand in m/c. Such a purely administrative task could easily be performed by the computer and programs (assemblers) were written to perform the task.

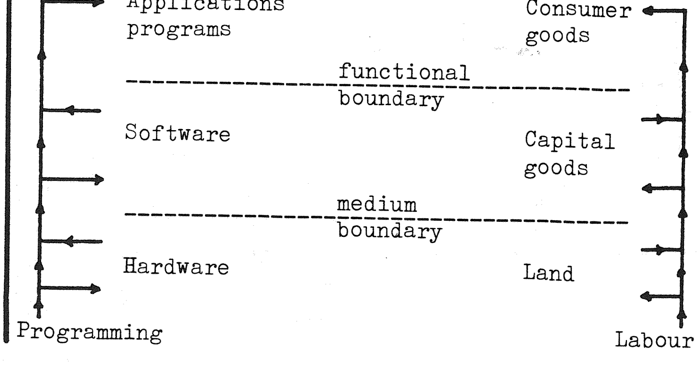
Thus a new class of programs was born which did not deliver an end product but instead produced other programs. An apt analogy may be drawn with the distinction an economist draws between consumer goods and capital goods. Just as capital is the investment of labour which allows more efficient production from labour and land, so the new class of program is the investment of programming which allows more efficient production from programming and computer. The new class of programs which seemingly changed the nature of the computer was termed software and it is a concept which however fuzzy at the edges, is still a very useful one.

So here then is my definition: Software is the set of programs which assists in generating and running other programs. A payroll program is not software under this definition because it neither generates nor assists in the running of other programs. It has a definite end product which is of use outside the computer environment. There are some boundary cases, but so long as one does not look at the hardware/software interface, these cases are usually quite clear. For instance, an editor used to alter a program is software; on the other hand an editor for use of typesetting is not. The operating system module which produces the running statistics of an installation is software because its use is limited to the computing environment, it is a management tool to assist in the running of other programs. These boundary cases lie on the interface between software and applications programs.

Let's turn now to the boundary between hardware and software. Most people would agree that a computer, at least at the drawing board stage, is a program. So what is it that makes a program soft or hard? It cannot be that one is easy to change and the other is not. Most software is as difficult to change as hardware if not more so. Some hardware can be very easy to alter. Seen in this way the only difference between hardware and software is the medium which carries the program. Thus if alteration of the program entails discarding the medium or part of the medium, the program would be regarded as hardware. A program on paper tape would be hardware!

This view of the difference between hardware and software does not however go to the heart of the matter. The essence of hardware programming seems to be that here one has to deal with the real world full of imperfections. It is difficult to understand for the APL programmer who lives in a world of abstract perfection that there are programmers who live in a noisy environment of stray fields and not-so-stabilized voltage supplies.

The computer designer builds the same kind of programs that I functionally defined as software but he has the added burden of having to mould his medium out of the raw materials. With the advent of IC, LSI, µP and bubble stores however, this burden is becoming less and it is clear in more recent designs that the hardware programmer has become more closely related to his colleague the software programmer.



(fig 1) Diagrammatic summary and analogy

The INTEL 8008 is an 18 lead integrated circuit Central Processing Unit. It can be interfaced with virtually any type of memory and peripherals to form a complete computer - as described elsewhere in this newsletter -. It currently costs around £50 (one off) from Walmore Electronics Ltd., 11-15 Betterton St., Drury Lane, London WC2H 9BS. We have tried to negotiate reduced prices for ACC members but without success.

Hardware

Fig 1 shows a block diagram of the circuit of the IC. The main features are;

-8 bit word.

-7 data registers accessible to the programmer. Reg A is the main accumulator. H & L are used to hold the memory addr. for memory reference data manipulation instructions.

-8 14 bit registers used to form an address stack to permit nesting of up to seven levels of subroutine. The 14 bit address length allows addressing of up to 16K bytes.

-Interrupt capability.

-Use with any speed memory. If a slow memory is used it can cause the CPU to pause while information is being stored or retrieved.

-18 lead package. IC packages with more leads are expensive. Unfortunately the restriction in the number of leads has resulted in the Data in, Data out and Address information being multiplexed onto one 8 bit timeshared bus. This slows the processor down considerably and also means that complementary multiplexing and de-multiplexing circuits have to be provided externally by the user - see the Basic System Diagram.A simple (non memory ref) instruction takes 20uS.

-All inputs and outputs are LOW POWER TTL compatible.

INSTRUCTION SET

An instruction may be 1,2 or 3 bytes.

`11 DDD SSS` Load data from SSS into DDD If SSS or DDD are 000 to 110 they refer to the data registers A to H. If SSS = 111 then data is loaded from the memory location addressed by H & L. If DDD = 111 then the data is loaded into the memory location addressed by H & L.

`00 DDD 110` `XX XXX XXX` The second byte of the instruction is loaded into register DDD or (if DDD = 111) the mem loc addressed by H & L.

`00 DDD 00X` where DDD is not 000 or 111. Increment reg if X = 0,decrement if X = 1.

Accumulator (reg 0) instructions

Status bits C (carry), Z (zero), S (sign) and P (parity) are set based on the result of the operation. In all cases the result of the operation is placed in the accumulator (A).

`10 III SSS` SSS is register or (if SSS = 111) the contents of the memory location addressed by H & L

III = 000 : ADD SSS to A
001 : ADD SSS plus carry (C) to A
010 : SUB SSS from A
011 : SUB SSS and C from A
100 : AND SSS with A
101 : EXCLUSIVE OR SSS with A
110 : INCLUSIVE OR SSS with A
111 : COMPARE SSS with A (A & SSS unchanged)

`00 III 100` `XX XXX XXX`
As above but source is the byte X - X immediately following the instruction.

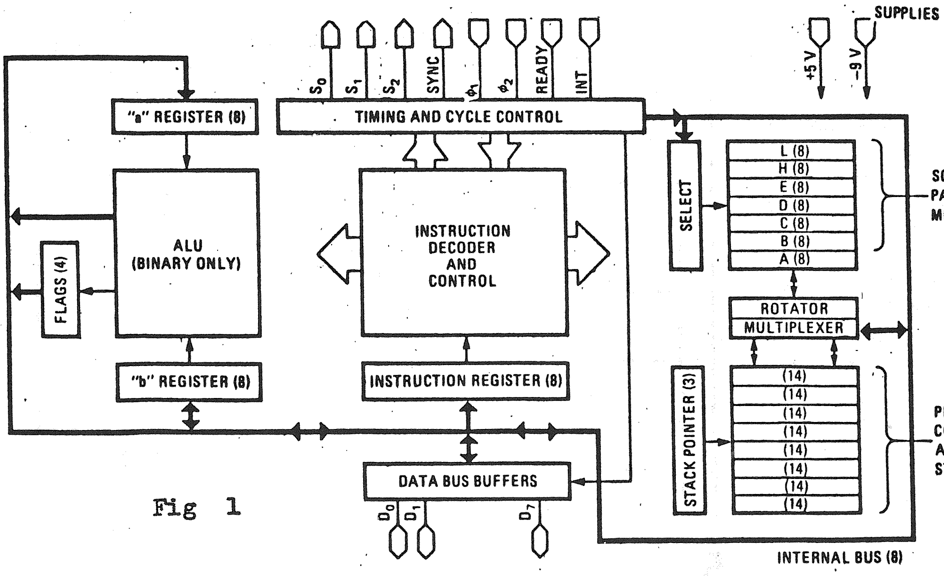
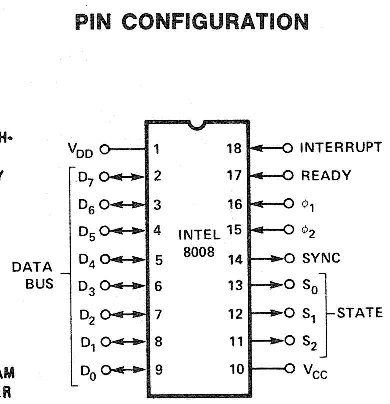


Fig 1



00 0II 010

II = 00 : Rotate A left one bit
     01 : Rotate A right one bit
     10 : Rotate A left through Carry
     11 : Rotate A right through Carry

Program Control Instructions

JUMP:

01 ACC D00 | YY YYY YYY | EE ZZZ ZZZ

CALL SUBROUTINE :

01 ACC D10 | YY YYY YYY | EE ZZZ ZZZ

RETURN:

00 ACC D00

In all cases the jump, call or return is unconditional if D = 1 (in which case ACC have no meaning). If D = 0 then the jump call or return is conditional according to the following:

A = 0 : jump, call or ret if status bit CC is 0.
    1 : jump, call or ret if status bit CC is 1.

where CC = 00 : Carry
           01 : Zero
           10 : Sign
           11 : Parity

If the jump or call are performed then they transfer control to location :

ZZZ ZZZ YY YYY YYY

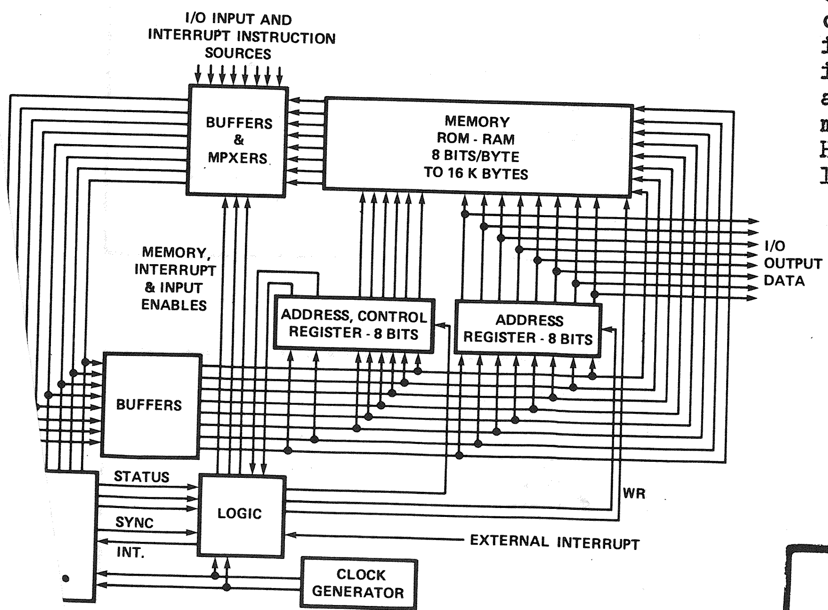
Bits EE are meaningless.

When a Jump is performed the return address is put onto the stack.

Return removes the return address from the stack.

00 AAA 101 — Calls subroutine at addr 00000000 00AAA000

00 000 000
00 000 001 — HALT
11 111 111



Basic System Using 8008

Input - Output Instructions

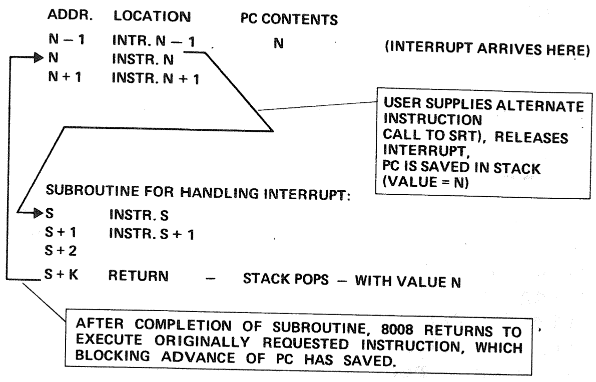
01 00M MM1 — Input data from dev MMM

01 RRM MM1 — Output data to dev RRMMM (RR can not = 00)

Interrupts

Are handled on the following basis:

When an interrupt request is received by the CPU it finishes the current instruction then sends out the address of the next instruction but does not increment the program counter. External circuitry can then be used to apply any required instruction to the CPU, overriding the instruction that would normally be fetched. The single word 'Call Subroutine' instruction is particularly useful for this purpose.



SUMMARY

The 8008 was the first real 'CPU on a chip'. Better ones exist now (including the 8080, of which more in a later issue) but they are far more expensive.

The 8008 is useable, cheap and makes a good basis for an amateur machine.

Main drawbacks, as far as I can tell, are the low speed, the limitation to 7 levels of subroutine nesting, and the difficult data addressing method. Data can be manipulated easily as long as it is in a register or the 'immediate' mode is used (data byte follows immediately after the instruction). All other data must be addressed indirectly through the H & L registers, which must first be loaded with the address of the data. This takes an extra 4 bytes (load H immediate, load L immediate) plus perhaps 2 more if you need to save the old values of H & L. Of course this limitation can be overcome to some extent by clever programming and careful structuring of the data but it does mar what is otherwise a good instruction set.

Diagrams reproduced by permission of INTEL.

COMPUTER SHOP

Dear Sirs;
We read the recent articles on your activities in Computer Weekly.
As our name implies we are computer brokers and buy and sell hardware professionally. However, sometimes we have old equipment on our hands which could be of interest to your members. At present we have a complete range of Elliot 803B equipment and we supply spares etc. to the various companies maintaining such machines.
Should your members require any parts perhaps you could put them in touch with us.

DATA RESALES
PO Box No 1
Chepstow, Mon

I have surplus to my requirements one Welmec reader complete (8 track) but in need of adjustment.

J.Florentin   203 Old Marylebone Rd
London NW1

GOOD HOME WANTED

James Abrahams Abrahams & Co (Birmingham) Ltd. placed an advertisement in the May edition of Office Equipment News saying that they would be giving away their NCR 500 computer - with paper tape punch & reader and automatic magnetic ledger card reader in October.
Their address; Brama Teams Glassworks, Gateshead.
Contact; Max Boucher 549 1366 x 481

ITV SCHOOLS COMPUTER PROGRAMME
9.50 - 10.10 am on October 24, Nov 5.

Babbage - Where did you put it ?

Having received the newsletter of the ACC I wonder whether you ra your club members have any knowledge or interest in old computers.

I am at present attempting to discover whether any machines built prior to 1960 (the earlier the better) are still in existence. If you have any information I would be most grateful if you would contact me through the address below or telephone 580 3953

yours sincerely

C.G.Furey   Staff Writer
'Computing', Gillow House,
5 Winsley St., London W1A 2HG

A. F. A. C. O.
ASSOCIATION FRANÇAISE des AMATEURS
CONSTRUCTEURS D'ORDINATEURS
DÉCLARÉE SUIVANT LA LOI DE 1901
Secrétariat : J.-C. RIBES
4, Avenue de Trivaux, 92190 MEUDON

← A NEW CLUB

WE HAVE HEARD

That a 'minicomputer' will be loaned to each of the 1000 or so Open University students taking the "Digital Computer" course next year.
The computer - which is an eight bit machine with 8 registers & 120 words of memory - will be put on open sale by the Open University next year.
If anyone has any further inf would they please send it along.

THE NEXT MEETING OF THE ACC

Will be held in Oxford on either Sat 26 Sep or Sat 5 Oct.

Exact details were not available at the time this newsletter went to the printer, but should be settled by the time you receive it. To find out ring;
R.Cowderoy at Oxford 53225 (after 6pm)
or
J.Aslett at Huddersfield 56712

ALL ARE WELCOME

I MUST APOLOGISE

For the delay in bringing out this issue of the newsletter.
Hoping to get back on schedule and distribute the October issue actually in October could I prevail upon all contributors to let me have their inputs by October 12 at the latest.

mike lord

|←——— 95 mm ———→|

Books of the Month (Year ?)

* 101 BASIC COMPUTER GAMES *

£3 from 'Software Services Dept., Digital Equipment Co. Ltd.
Fountain House, Butts Centre
Reading RG1 7QN

* MY COMPUTER LIKES ME
(When I speak in BASIC) *

A teach yourself to play with (and program) BASIC.

S 1.49 plus postage (say £1) from;
DYMAX , PO Box 310, Menlo Park,
California 94025 USA

Amateur Computer Club Officers ;

Chairman	J.Creutzberg
Treasurer & ACCN editor	M.Lord
Secretary	J.Aslett

Committee members ;
R.Cowderoy
G.Hankey
M.Reeve

All communications to the Club
sent to 7 Dordells, Basildon,