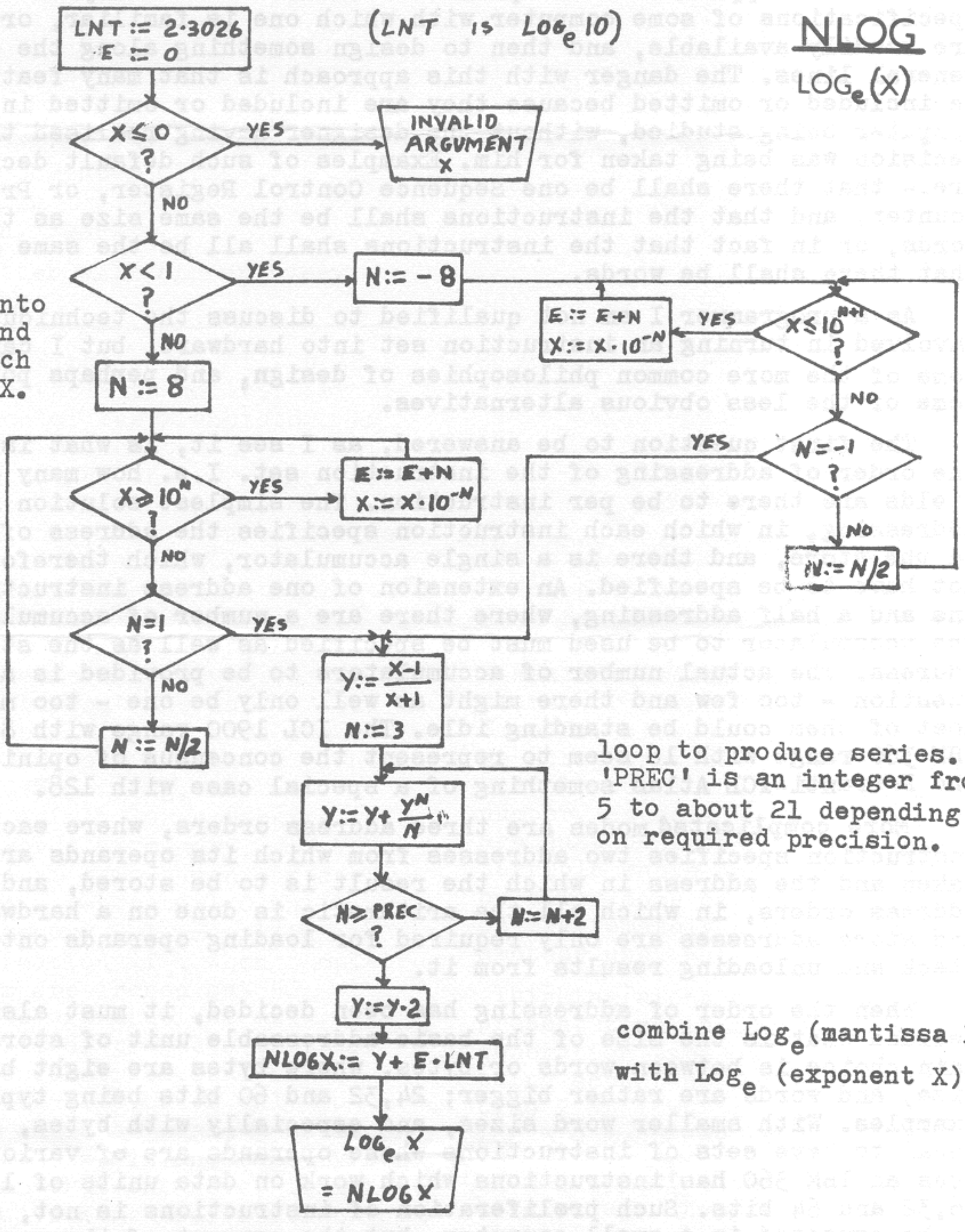
ACCN AMATEUR COMPUTER CLUB NEWSLETTER VOL I ISS 2 MAY 1973

NLOG
$LOG_e(X)$



Separate X into exponent E and mantissa which becomes new X.

loop to produce series. 'PREC' is an integer from 5 to about 21 depending on required precision.

combine Log_e (mantissa X) with Log_e (exponent X)

IF $Y = \frac{X-1}{X+1}$, THEN $X = \frac{1+Y}{1-Y}$

$$\therefore \ln X = \ln(1+Y) - \ln(1-Y)$$
$$= Y + \frac{Y^2}{2} + \frac{Y^3}{3} + \frac{Y^4}{4} \ldots \quad -\left(-Y + \frac{Y^2}{2} - \frac{Y^3}{3} + \frac{Y^4}{4} \cdots\right)$$
$$= 2\left(Y + \frac{Y^3}{3} + \frac{Y^5}{5} + \ldots\right)$$

S. PANTING.

~ Thoughts on Computer Design, J.D.Thewliss ~

How does someone set about designing a computer? I am thinking in particular of the amateur in the field; someone with some experience of electronics, and perhaps of programming, but not of the principles that lie behind the basic desigh of a machine. There are many decisions to be taken, and I propose to discuss those which relate to the definition of the instruction repertoire, or 'opcode set', of the processor.

A common approach to the problem seems to be to study the specifications of some computer with which one is familiar, or which are readily available, and then to design something along the same general lines. The danger with this approach is that many features may be included or omitted because they are included or omitted in the computer being studied, without the designer having realised that a decision was being taken for him. Examples of such default decisions are:- that there shall be one Sequence Control Register, or Program Counter, and that the instructions shall be the same size as the data words, or in fact that the instructions shall all be the same size or that there shall be words.

As a programmer I am not qualified to discuss the techniques involved in turning an instruction set into hardware, but I can compare some of the more common philosophies of design, and perhaps point out some of the less obvious alternatives.

The first question to be answered, as I see it, is what is to be the order of addressing of the instruction set. I.e. how many address fields are there to be per instruction. The simplest solution is single addressing, in which each instruction specifies the address of a word in the store, and there is a single accumulator, which therefore does not have to be specified. An extension of one address instructions is one and a half addressing, where there are a number of accumulators, and the accumulator to be used must be specified as well as the store address. The actual number of accumulators to be provided is a tricky question - too few and there might as well only be one - too many and most of them could be standing idle. The ICL 1900 range with 8 and the IBM 360 range with 16 seem to represent the concensus of opinion, with the Ferranti-ICL Atlas something of a special case with 128.

More complicated modes are three address orders, where each instruction specifies two addresses from which its operands are to be taken and the address in which the result is to be stored, and zero address orders, in which all the arithmetic is done on a hardware stack and store addresses are only required for loading operands onto the stack and unloading results from it.

When the order of addressing has been decided, it must also be decided what is the size of the basic addressable unit of storage. The main choice is between words or bytes, where bytes are eight bits in size, and words are rather bigger; 24,32 and 60 bits being typical examples. With smaller word sizes, and especially with bytes, it is usual to have sets of instructions whose operands are of various lengths, thus an IBM 360 has instructions which work on data units of length 4,8, 16,32 and 64 bits. Such proliferation of instructions is not, of course, to be expected in a small computer, but the concept of the word as the unit of size used by all instructions must not be regarded as universally binding.

A third area of decision is involved with the size and constitution of instructions. The order of addressing will obviously affect the size of instructions, as will the number of instructions provided, and the amount of store directly addressable. Various other factors will also play a part, and I think that the best way to consider their effects on instruction design is to compare the designs used on a number of commercially available computers, and to this end I have drawn up a table of the instruction formats of six computers. There are many important features o f these machines that are not mentioned in the table, but I think that it should serve its purpose of pointing out some of the breadth of choice available in instruction set design, and that was really the principal object of this article.

	word size	instr. size	opcode field	acc. field	addr. field	mod. field	extras
PDP8	12	12	3*		7		1 direct/indirect
903	18	18	4*		13	1	
4120	24	12	6		6		
		24	6		15		2 normal/literal/ modified/indirect 1 hardware/extracode
KDF9	48	8	8**				
		16	8		4,4		
		16	8			4,4	
		24	8		16		
		24	4		16	4	
1900	24	24	6	3	15		
		24	7	3	12	2	
		24	9	3	10	2	
360	32	16	8	4,4			
		32	8	4	12+	4,4	
		32	8	4,4	12	4	
		32	8		12	4	8 literal value
		48	8		12,12	4,4	8 length, or 4,4

* one or more opcodes do not take store addresses, but can specify a number of zero address operations on registers and/or peripheral channels.
** a hardware stack is used.
+ addressing is in bytes, not words.

BITS

M.E.Allard, 57 Priests Lane, Brentwood, Essex has use of a PDP 11 + 1.2M disc, VDU, high speed paper tape reader. He would like to hear from anyone who has any interesting programs suitable for this set up.

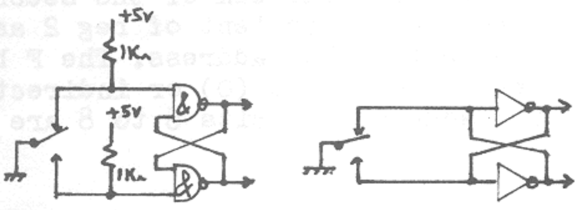
R.Palmer, 90 Stort Tower, Harlow, Essex has about 330 DTL's (mainly JK flip-flops) for sale or swap. He is in need of a Cannon plug for a F-V Flexowriter.

P.J.Stringer, Damar House, Ottershaw Park, Ottershaw, Chertsey, Surrey. has bought a Welmec 51108 paper tape punch ,for use on a 24 bit machine he is constructing, and would like any details on the punch.

N.Trewartha , 4790 Paderborn, Konrad-Martin Platz 5, B.R.D. / West Germany , would like to buy a 2nd. (or more) hand computer; 12-18 bit speed not very important , TTY with interface required.

BOUNCES

The circuit shown in Fig 1 is commonly used to eliminate the worst effects of switch contact bounce. (a problem when operation of the switch is required to trigger a single shot circuit). Fig 2 shows a simpler and cheaper version. When TTL inverters are used the second circuit even has an improved noise immunity , as both sides of the switch are always connected to low impedance points. (The output impedance of a TTL inverter is typically less than 100 ohms in both the '1' and '0' states).

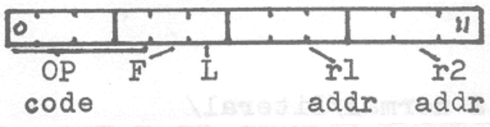


1 2

- 12 bit MACHINE R.I.Cowderoy -

12 bit words. 2's complement arithmetic. Linear store. 7 active accumulators which can be used as general purpose registers , index registers etc.

1 word instruction format ;



2 word instruction format as above with second word as the address or as a literal.

OP (octal)

00 LOR logical OR
01 AND logical AND
02 ADD
03 SUB
04 LDA load accumulator
05 STA store accumulator
06 MUL
07 DIV

For these instructions the F & L bits are used to specify the addressing mode;

F=0,L=0; reg to reg.
F=0,L=1; literal to reg, where the literal value is stored in the word following the instr.
F=1,L=0; store to reg, 2 word instruction with the store address in the second word.
F=1,L=1; store to reg, indirect.

Jumps

OP	L	
10	0	JMP unconditional jump.
10	1	JAZ jump if acc zero.
11	0	JAN jump if acc neg.
11	1	JAP jump if acc pos.
12	0	JAO jump if acc odd.
12	1	JAE jump if acc even.
13	0	JLS jump if link = 1
13	1	JLC jump if link = 0

There is an overflow or link bit for each register.

All jump instructions are 2 word, using the sum of the second word and the content of reg 2 as the destination address. The F bit specifies direct (0) or indirect (1) addressing. Bits 6 to 8 are used to address a register whose contents are to be tested, except in the case of a JMP when they specify a register in which a subroutine link can be placed.

OP
14 Logical shift
15 Arithmetic shift
16 Input / output
17 Special (Halt etc.)

Hardware;

Mainly serial operation, run from one master clock. TTL logic. EX Ferranti Orion 10us core store with own designed electronics. The only peripheral at present is a Creed model 75 teleprinter.

5 BIT TTY

Lettrs.	Figs.	5	4	3	2	1
A	-	0	0	0	1	1
B	?	1	1	0	0	1
C	:	0	1	1	1	0
D	8̸	0	1	0	0	1
E	3	0	0	0	0	1
F	!	0	1	1	0	1
G	&	1	1	0	1	0
H	£	1	0	1	0	0
I	8	0	0	1	1	0
J	'	0	1	0	1	1
K	(	0	1	1	1	1
L	)	1	0	0	1	0
M	.	1	1	1	0	0
N	,	0	1	1	0	0
O	9	1	1	0	0	0
P	Ø	1	0	1	1	0
Q	1	1	0	1	1	1
R	4	0	1	0	1	0
S	bell	0	0	1	0	1
T	5	1	0	0	0	0
U	7	0	0	1	1	1
V	;	1	1	1	1	0
W	2	1	0	0	1	1
X	/	1	1	1	0	1
Y	6	1	0	1	0	1
Z	"	1	0	0	0	1
line feed		0	0	0	1	0
carr. ret		0	1	0	0	0
space		0	0	1	0	0
figures		1	1	0	1	1
letters		1	1	1	1	1

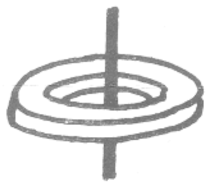
1 = hole ▲ sprocket

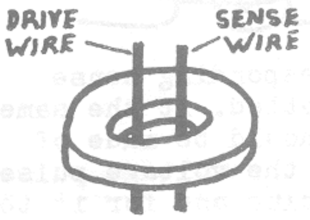
+ SUPPORT THE ACCN +
THE NEWSLETTER DEPENDS ON YOU
MEMBERS DATA AND ARTICLES WANTED

CORES FOR STORES

'surplus' magnetic core stores are about the cheapest form of fast bulk memory available to the amateur at present. Unfortunately it is not usually possible to get a complete working store unit with all the assosciated electronics, but only the core stack itself, which means that the user has to build the drive and sense circuits himself.

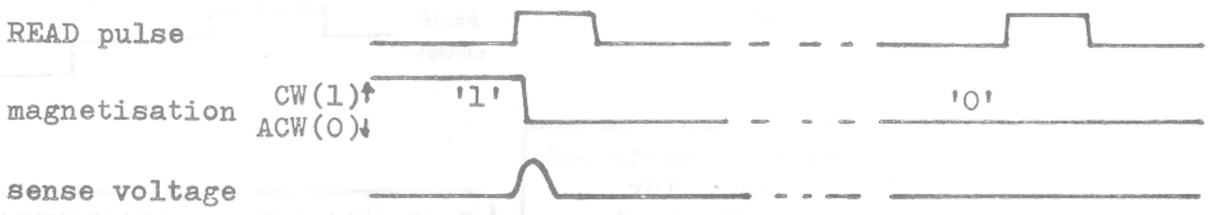
The basic p rinciples are very simple. A small 'polo mint' (or 'toroid' for the mathemeticians amongst us) is made of a material which can be magnetised in either of two directions which we can call 'clockwise' and 'anticlockwise' (and as long as we don't turn the core over and look at it from the other side then we won't get confused). To magnetise the core we put a wire through the centre then by passing a sufficiently large current through the wire we can magnetise the core in either the clockwise or the anticlockwise direction depending on which way the current is made to flow. The core material is such that it will remain magnetised when the current is removed, so hey-presto ! we have a memory. If we say that anticlockwise magnetisation represents a binary '0', and clockwise a '1', then by passing the current in the right direction we can set the store to the required value (1 or 0). To make a practical store all we need are thousands of cores (to store thousands of 'bits') and some circuits to pass the right currents in the right direction and some way of sensing the direction of magnetisation of each core in the stack.





Perhaps you could detect the direction of magnetisation with a compass needle, but a better way is to make use of the fact that if the core is magnetised in, say, the anticlockwise direction then further attempts to magnetise it anticlockwise will not have much affect, whereas if it had initially been magnetised CW then the attempt to 'anticlockwise' it would succeed and, in doing so would cause a large change (in fact a reversal) of the magnetic flux in the core. Now changes in flux induce voltages in wires so by putting a second 'sense' wire through the core we can detect this flux reversal as a voltage pulse across the sense wire.

By passing a 'read' current pulse through the drive wire in such a direction as to tend to magnetise the core ACW (to '0') the we will get a voltage pulse if the core had stored a '1', and no pulse if it had stored a '0'.

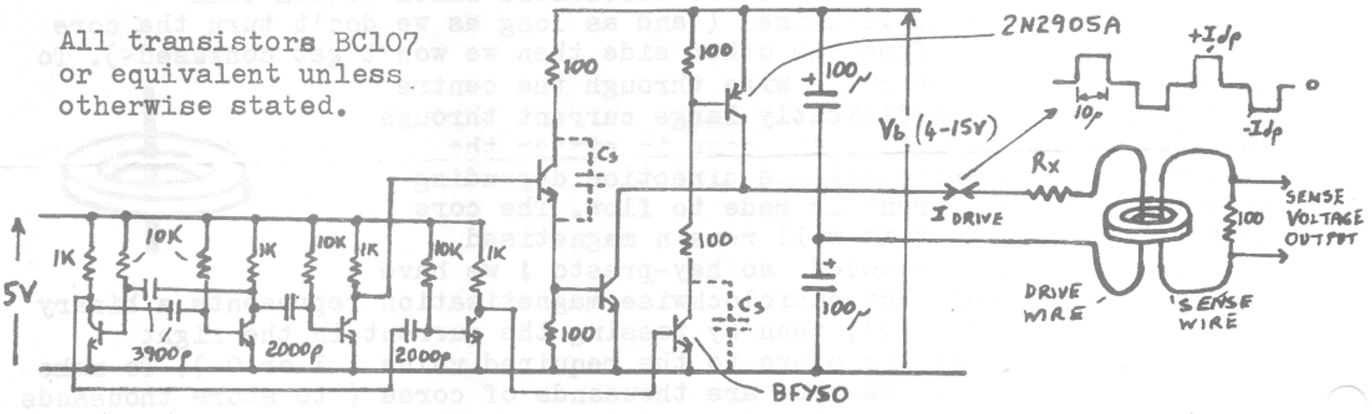


Note that the act of reading out actually erases the stored information by setting the core to 0; thus magnetic core stores are described as 'destructive read out'(DRO) stores and care must be taken to write the information back in after the read operation if it will be needed again.

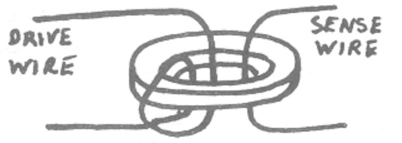
So far so good, but the practical problems of "how much current for how long & what value of sense voltage do I get " remain. There are two ways of getting this information; the easiest and best is by examining the manufacturers data sheets. When this is not possible we must experiment. The obvious thing to do is to get a core with a drive and a sense wire running through it and then put current pulses through the drive wire and see what happens.

The circuit shown below will deliver 10 uS current pulses of alternate polarity through a drive wire which should be passed through a single core (this is usually possible by using fine enammeled copper wire even when the cores are already wired into a stack). The size of the current pulse can be determined by measuring the voltage Vdp across Rx with a 'scope and altered by varying the value of Rx or the supply voltage Vb. The capacitors Cs (about 1000pF) may be needed to stop spurious oscillations.

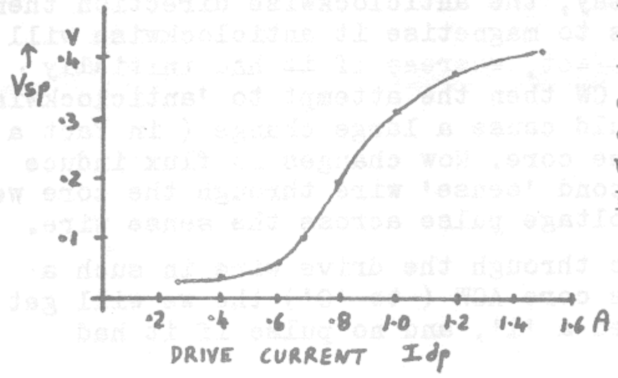
All transistors BC107 or equivalent unless otherwise stated.

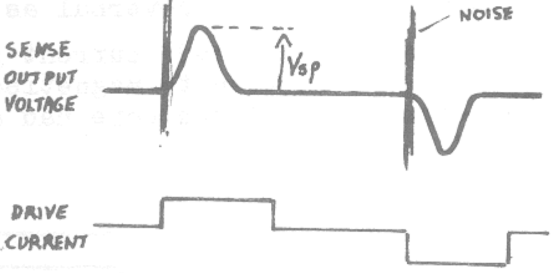


As it stands the circuit can push up to about 600mA through the drive wire. If more is needed, this can be effectively doubled by passing the drive wire through the core twice.



By varying the drive current and measuring the corresponding sense voltage (peak), a graph like that shown below may be plotted. At the same time measurements should be made of the times taken for the voltage pulse to reach its peak value and for it to die away. Care should be taken not to confuse the wanted sense voltage pulse with any noise spikes which may be present.





Continued next issue.

Several people have written to ask for details of books suitable for beginners. I am making my own review of those books I know and would be interested to hear of other peoples' recommendations.

A list of members' names and addresses will be included in the next issue. Anyone not wishing their name to be included please let me know.

Next issue - August.

THE AMATEUR COMPUTER CLUB IS OPEN TO ANYONE INTERESTED IN THE DESIGN, CONSTRUCTION OR PROGRAMMING OF COMPUTERS AS A HOBBY. FOR MEMBERSHIP OF THE ACC AND A SUBSCRIPTION TO VOL. I OF THE NEWSLETTER SEND 50p - 75p FOR OVERSEAS MEMBERS -. VOL. I WILL COMPRISE AT LEAST 5 ISSUES PRODUCED AT 2 TO 3 MONTH INTERVALS.