ACC AMATEUR COMPUTER CLUB NEWSLETTER
VOL.4 ISS 5 DEC 1976

300 BAUD CASSETTE STORE

INTRODUCTION

For the cost of approximately £15 it is possible to build a bulk storage system using the cheapest and nastiest type of cassette recorder (in my case Marconiphone 4266 which had deteriorated beyond listenable quality). It has in about a month's use (loading programs into a 6800 microprocessor) produced no errors, and on an earlier test ran for 45 minutes at 200 baud without error.

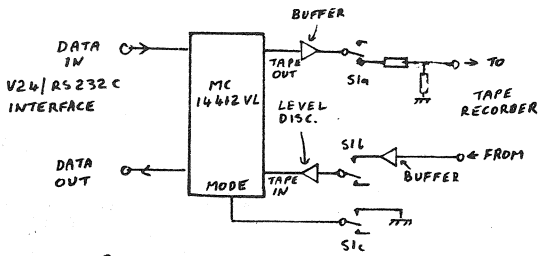
Its heart is a Motorola chip which does all the tricky stuff and is capable of working up to 600 baud. The main point of using this particular chip is that it uses international standard frequencies and so is potentially compatible with commercial equipment which may one day be available as scrap. A similar machine could be made possibly cheaper using circuits outlined by John Compton in 'Electronics' page 83 April 1st 1976. The disadvantages are that it is very slow compared to the full speed potential of cassettes.

THEORY

Logic '1' is modulated to a 980 Hz tone
Logic '0' is modulated to a 1180Hz tone
which can be attenuated and fed into the microphone socket of the recorder. This corresponds to channel 1 of CCITT standard V21.

On playing back the demodulator on the chip needs to be set for receiving channel 1 (this automatically means the modulator goes from channel 1 to channel 2 (1650Hz & 1850 logic '1' & '0')). This is achieved by setting the mode control high or low. this modulating process is called Frequency Shift Keying (FSK).

Now CCITT standard V24 can be crudely translated as logic '1' = -12V into 3000 to 7000 ohms, and logic '0' as +12V into 3000 to 7000 ohms. This is compatible with RS232-C. This may seem to be a very strange arrangement but is a firm favourite rivalled only by the 20mA current loop. It unfortunately requires a massive 2½W power supply. It can be met using Texas chips 75150 line driver and 75154 line receiver or Motorola MC1488 line driver and MC1489 line receiver. A discrete component design was used to lower the cost.



BLOCK SCHEMATIC

CIRCUIT DESCRIPTION

Based on Motorola chip MC14412VL

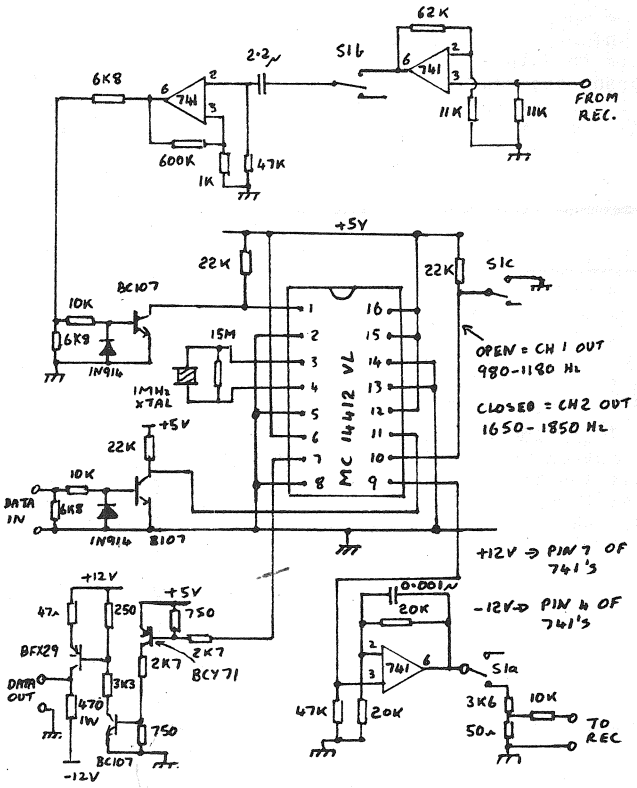
As it is a CMOS circuit and expensive, great care is taken to buffer it from the outside world, particularly the +12V which is offered to all its inputs. The choice of component values 'happened' more than designed so I'm sure alternate values could easily work just as well but be warned about tinkering with the DATA OUT circuit as there are several jokes in it. The two buffer amplifiers are straightforward, one including a simple filter. The level discriminator however uses positive feedback to produce a +12V signal out, tripping at approximately +20mV input. This is for a signal of about 1½V p-p. It was found that about 10% phase distortion is caused by the FSK demodulator and a low trip level helps to minimise it.

CONCLUSIONS

It works.

I'm sure a more elegant arrangement could be designed using less switches and components but that's the one I built. If anyone wishes to discuss it they can contact me on 0628 29073.

Tim Moore. 24 College Rd.,Maidenhead,Berks SL6 6BN



POWER CONSUMPTION : +12V @ 84mA, +5V @ 4mA, -12V @ 60mA

CIRCUIT

IN THIS ISSUE

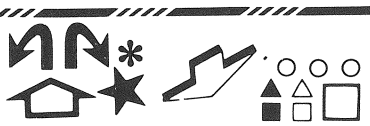
1180
980
200Hz

ACC GENERAL MEETING

Thursday 27th January at 7.00pm

Room 468

South Bank Polytechnic
(Elephant & Castle)



COMPUTER ARTWORK COMPETITION

Free 1977/8 ACC membership for the lucky winner.
Entries to be original computer generated artwork of any size, and will not be returned. The winning entry will be printed in the Feb. newsletter, so please ensure that a high quality, high contrast print is provided.
Closing date 14th Feb.

ED'S BIT

A bumper edition this time - in fact some contributions have had to be kept for the next issue. But keep writing !

Yet another cassette interface description appears in this newsletter. While there is no doubt that these circuits have been valuable and of great interest to many, I feel that it is rather a pity that no-one has yet come up with a CUTS compatible device. After all, although it may well be possible to devise a technically better system, CUTS has emerged as the 'standard' system in the US, and I can't see any reason for following a different path. So, will anyone who has a CUTS design please publish it.

Three CUTS interface designs have been published in BYTE; Don Lancaster's and Harold Maunch's circuits in March and the STP AC-30 in Dec '76. A folder containing all three articles is now available on loan to UK ACC members; send a couple of 8½p stamps to join the queue.

Those members who attended the Motarola talk on 15th July may recall a suggestion that Motorola donate some 6800 software to the ACC.

Unfortunately nothing has come of this to date - mainly because the person concerned has since left Motorola . But Mick Reeve is still trying and any progress will, of course, be reported in the newsletter.

PUZZLE BITS Philip Rutherford

1) A circle, a square and an equilateral triangle all have the same length perimeter. If the circle is rotated around the other two without slipping in which case will it rotate through the largest number of degrees ?

2) Among 100 applicants for a certain technical position it was discovered that ten had never taken a course in chemistry or in physics. Seventy five had taken at least one course in chemistry. Eighty three had taken at least one course in physics.

How many of the applicants had had some experience in both chemistry and physics ?

No prizes - answers next issue

COMPUTER ARTWORK T SHIRTS

SAE for details from R J Baker
54 Brixton Rd.,London SW9 6BS

SYMBOL TABLES

I have not attempted to give an exhaustive study of Symbol Tables, only an account of some of the techniques involved.

When writing programs, it is desirable to have locations represented by names; your home-made compilers and assemblers must take these names and convert them into numbers. One way of doing it would be to manipulate the characters as numbers, and get a result in the range 0-4095 (or something). The troubles with this are: that the algorithm must give a unique result for each name(=> a crippling length restriction for a start), which is difficult to engineer; and that the results of this operation are not necessarily compact or nicely spread out, i.e. your program gets splattered all over store.

Another answer (very simple and not very good) is to say all identifiers must be only one character long, allot 26 consecutive locations in the global area of core and address arrays indirectly. All very well until you write a program with 27 variables. Nor does this allow labels within the program. A slightly better idea for addressing variables is to let each letter represent a (fixed) address, and then have a number tacked onto the end of the identifier to say how much must be added to this to get the location required. This is the method employed, in a modified form, by the KDF9 Usercode assembler. Note that here, the real address may be calculated by the compiler, the linkage editor, or the object program. This last alternative is very inefficient both in store and time requirements: moral- let the compiler do all the work.

Whilst the last symbol type is not very exciting, it does illustrate one thing- the use of relativizers, which in this case are the initial letters of symbols (in KDF9 Usercode, for example, Y was the relativizer for the main data area of store, E meant 0, P was the program origin, V was the start of the literal pool, etc.). Relativizers are the compiler's glorified equivalent of index registers; they might be used implicitly, while compiling into relocatable binary, or explicitly as above or as determined by the architecture of the machine (e.g. the 1900 series assembler needs a relativizer for the upper data area, stored after the program, as it does not know in advance how long the program will be). In a mini, they might be concerned with addressing modes: usually, their brief existence ends with consolidation. It is useful and easy to allow for their existence in symbol tables, just in case they are needed.

Simplest of the fully fledged symbol tables is the linear search table. As each new symbol is encountered it is stuck onto the end of a vector of names: to find a symbol, work along the table from the beginning until the symbol is found: if you drop off the end, it does not exist. This method is terribly inefficient- for a vector of length N symbols, the mean search length is >N/2. The most effective type of symbol table for program defined symbols is the hash table. The basic version of this operates as follows:

(i) a numerical value (Hash function) is calculated from the symbol to be found or entered
(ii) it is reduced mod the table length
(iii) the entry in the table corresponding to the reduced value is examined:
  if zero, go to (v)
  if = required symbol, go to (vi)
(iv) add a fixed number (1,3,5) to the reduced function, go to (ii)
(v) the symbol has not been found. If it is to be entered, replace the zero value in the table with the symbol,a value, a relativizer and return. if not, return NOT FOUND.
(vi) the symbol has been found. If it was being searched for, return the value and the relativizer from the table. If not, return ALREADY EXISTS.

This algorithm is not complete. A routine may be incorporated to check the relativizer against one supplied, and another to extend the table when it becomes full (keep a count of the number of symbols defined). The problem with the algorithm as it stands is that symbols must have a fixed length or be space filled to that length by the symbol reading routine. To overcome this problem, some hash tables may have insteat each entry a pointer to the beginning of the symbol in a separate dictionary, and the number of characters in the symbol. The search time may then be reduced by checking the length of the entry with the search symbol length prior to comring the symbols. Search time may be further reduced by comparing the unreduced Hash function with one recorded in the table. If either of these disagree, the checking of the whole symbol is obviated.

However, user defined symbols are not the only ones encountered in programs: there are key words- such as begin, real etc., and of course the mnemonics in an assembler. If these are arranged in order of numerical magnitude, from

AAAA to ZZZZ, together with their values, a very simple, efficient logarithmic search is possible; here is one version as an algol procedure:

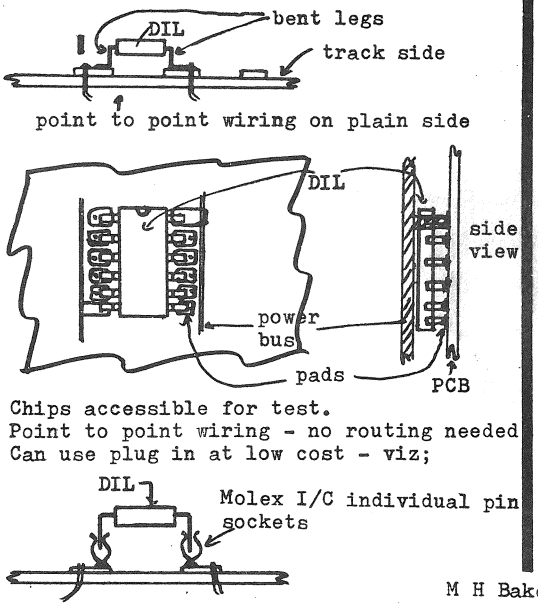
```
integer procedure logsearch(sym);
  integer sym;comment characters packed as an integer,
      tab,len,val are global variables. tab is the table,
      val is a vector of associated values, len is the
      no of symbols in tab+1 (= an integral power of two);
begin integer step,try;
  step:=len;try:=entier(step/2);
repeat:if sym=tab[try] then goto found;
  step:=entier(step/2);
  if step=0 then goto notfound;
  try:=if sym>tab[try] then try+step
                       else try-step;
  goto repeat;
found:logsearch:=val[try];
notfound:end
```

The main restriction here is that it is very difficult to change the table because the order of entries must be preserved.

A logarithmic search is much more efficient than a hash table search. A Hash table is much more efficient at collecting symbols. It would seem to be best, then, to write compilers as at least two-pass: the first pass for creating a hash table, the second for re-ordering and padding this out into a logsearch table and performing the object code conversion. Or perhaps you prefer to program in binary.....

O.F.Morgan

LOGIC BOARD CONSTRUCTION METHOD



Chips accessible for test.
Point to point wiring - no routing needed
Can use plug in at low cost - viz;

M H Baker

SHOP

Creed 75 teleprinter with 5 track tape punch and reader (Ferranti code). Working. Offers.
G.R.Cobb, 89 Eastwick Park Ave.,Bookham,Surrey
tel; Bookham (0372) 53787

8080A microprocessor. Assembled and working in 22 slot card frame complete with NCR paper tape punch high speed reader. Clare-Pendar keyboard and ETI VDU boards built and tested. The unit although complete and working is sold as parts i.e. 8080A chip with 8224 clock 4 x 8216 bus drivers. 3 x 8208 decoders many 8212 I/O ports. 3k of P2102A-6 and 1k of P2112. 4k of 1103 dynamic RAM (unused). Two power supplies give ample spare capacity. Supplies of unused paper tape and 8080A manuals are included.

ASR33 TTY of Data Dynamics origin. Good order with box of spare rolls of paper and ribbons.

8080A best offer over £100 (worth £200 in chips alone)
ASR33 best offer over £125 (worth 3x this)

Ring 0455 35621 evenings/weekends
D.V.Goadby 2 Lupin Close, Hinckley, Leics LE10 2UJ

I wish to sell a Creed '54 teletype with paper tape punch & reader as its not quite compatible with my system (WB, hopefully controlling I/O, & a 1600) but as I can directly address 64 Kbytes of memory using the 1600 and combined with WB using an interface of not exactly original design I should be able to address megabytes & so am looking for a mass storage system e.g. mag tape, disc or drum etc. If any member is willing to sell any of these I would be willing to pay a reasonable price.
A.Deas
6 Lime Close, Turnpike Est.,Newbury, Berks RG13 2PW

ICL 1901A
Processor 2010/2 16K
Console/Desk F.1094
Printer 2405/2 600 lpm
Twin exchangeable disc stores 2821/1
Approx 95 twin exchangeable discs
Offers to either;

Neil Pearce	or	Nick Mannerings
1 Chapman Ave		47 Gladstone Rd
Maidstone		Maidstone
Kent		Kent

CHEAP MOSTEKS

I have been talking to my distributor of MPU components and he has agreed to let members of the ACC have MPU components at the distributor's 25-99 off price.

His only conditions were that all orders should be placed through me and that we did not disclose our source in the newsletter. He will dispatch all orders direct to the individual members. He also said that if the total orders for any component should reach the 100+ region he will give members the 100+ price.

I have negotiated prices for the following components, members should add VAT at 8% to the total order value and a contribution of 50p towards postage and packing.

All components supplied are 'factory fresh' and will be covered by the manufacturer's warranty.

Members wishing to take advantage of this offer are requested to send me a duplicate list of their requirements, a label containing their name and address and a cheque for the full amount including VAT and postage. I will re-imburse any surplus monies to the members concerned should the orders for any component reach the 100+ price.

The prices agreed with my distributor yesterday, 16th December, are as follows. I have given his 1 off prices also as a guide, some prices from other distributors are in fact higher than these.

	1 off	members price
Z-80 CPU ceramic package	£65.00	£47.20
Z-80 PIO ceramic package	18.95	14.10
1702A type EPROM, pin for pin replacement;		
MK3702-T-1 1uS	£9.60	8.00
3702-T-2 750nS	11.40	9.30
3702-T-3 550nS	13.20	11.50
I can arrange to have these programmed for a charge of £3.00 each.		
2102 type 1k RAM plastic package, 5V supply;		
MK4102N-1 450nS	£2.30	1.75
4102N-6 270nS	2.85	2.20
If we can get towards 100+ of these devices the 450nS version's price falls to £1.50 each.		
4k dynamic RAM,low power; 300mW active,0.6mW standby.		
MK4200P-11 350nS	£11.85	8.90
4200P-16 300nS	12.61	9.45
4k dynamic RAM low power;380mW active,24mW standby		
MK4096P-11 350nS	£11.13	8.17

16k dynamic RAM are also available and have access speeds of either 250nS or 200nS. I can let anyone who is interested know the price for these devices. They are not cheap and are in very short supply.

Also available from the same source are components for the F8 MPU and I list the prices below;

MK3850N-1	F8 MPU crystal control	£15.10	10.20
MK3850N-3	F8 MPU R-C control	12.70	8.45
MK3852N	dynamic memory interface	8.70	6.30
MK3854N	DMA	7.00	5.50
MK6820N-1	peripheral interface adaptor	6.00	4.70
MK3853N	static memory interface	8.70	5.50
MK3861N	peripheral input / output	8.70	5.50
MK3851N	program storage unit 1k ROM + 2x8 bit I/O ports programmed with the DDT-1 software monitor	£24.65	22.50

The MK6820N is a pin for pin replacement for the Motorola M6820 PIA, it is available in two speed versions.

F8 Survival Kit -assembled MPU unit containing a 3850 MPU, a 3851 PSU preprogrammed DDT-1, 1kx8 static RAM, 3x8bit I/O ports available to user, 1x8bit I/O port dedicated to a TTY/RS232 interface also supplied are the F8 manual,F8 Programming Manual,DDT-1 listing and as a bonus a FORTRAN ASSEMBLER on 80 col punched cards. The price for the F8 kit is £123 + 8%VAT + £2 postage and packing. Bywood are asking £165 for this kit. I have purchased one for myself and am designing a power supply for this, plus up to 63k x 8 RAM/ROM. The board needs +5V at 750mA and +12V at 250mA.

I also have arranged for any member who is interested a source for a low cost TTY/CRT substitute. This is a 'low cost' visual display unit keyboard entry terminal. It will drive a domestic TV and is available in two versions;
Model 401 with a video output 2.25 volts p-p with composite synch (your TV will need a minor mod and details will be supplied with this model) £250.00 + 8% VAT + £3.00 p&p
Model 403 with UHF output channels 21-68 tunable. (this will plug into your aerial socket) £275 + 8% VAT + £3.00 P&P.
The specs for this terminal are;

Power supply	- mains 230 - 250V 50Hz
Memory	- 1024 characters
Format	-16 lines by 64 characters per page
Character set	- each character is made up of a 7x9 dot matrix. There are 128 different characters, upper and lower case, descenders. (alternative character sets also available)
Keyboard	- full teletype functions available with 7 bit ASCII code output and control characters. The keyboard has N key roll over.
Baud Rate	- pre-selectable from 110 to 9600
Cursor	- a full addressable cursor is provided with blinking facilities.
Display	- white on a black ground.
MPU interface	- 20mA current loop or RS232 V24

I have tried out the prototype of this terminal and it works well. I have one on order and expect delivery at the end of January. The production models will use a F8 MPU as controller. If there is sufficient demand for this terminal I expect I may be able to get an improvement in the price.

If any member is interested I have the details for interfacing a MOSTEX Expandable Calculator Set (see Sept '76 issue of ETI) to a MPU. There is an improved version of this set available which with the addition of two extra ROM's can be made into a 100 step programmable calculator. The part numbers for the Financial chip set are MK50075 (ALU) MK50101/MK50102 and the Scientific chip set MK50075 (ALU) MK50107/50108 the programmable chips are MK50109 (50 step programmable) + MK50110 to convert this chip set to 100 step programmable. The MK50075 (ALU) will 'drive' up to 16 ROM's from the MK501XX series. Prices are;
MK50075 (ALU) £8.50 one off each
MK501XX (ROM) £6.45 one off each
I expect I could get an improvement in this price if anyone is interested.

The prices quoted in this letter are open for an indefinite period, but would be subject to revision should the manufacturer increase his price to my distributor. If any member would like a copy of the product data sheets, I am expecting a limited supply of these from my distributor and if the member will send me a 9" x 12" envelope stamped (9p) and addressed I will send him/her a copy.
William J Whitehouse
Flat 2, Hean Castle, Saundersfoot, Dyfed SA69 9AL
tel; Saundersfoot (0834) 813075 (preferably between 5pm and 6.30pm)

DIVISION

Here is a method of division by addition. It amounts to long division in binary. It can easily be implemented in hardware, if you want that sort of thing, or alternatively I have written a little routine in WB Usercode, which although slow is certainly faster than repeated subtractions. To illustrate the technique, here is how to evaluate 122/5 (in binary, 0111010/0000101):

i invert the numerator (take the one s complement)
01111010 becomes 10000101
ii align the first non zero bit of the denominator with the beginning of the word (in fact, you could align it with the fist zero bit of the inverted numerator, but this is time consuming in software)
00000101 becomes 10100000
iii add the numerator and shifted denominator:
10000101
10100000
100100101
iv if overflow occurs, as it did above, ignore the result of the addition.
if overflow does not occur, replace the num. with the result, and enter a 1 in the quotient s least significant bit.
v shift the quotient left one place, the denominator right one place, and repeat steps 3,4 until the denominator starts falling off the end of the word.
vi the numerator inverted is the remainder.

Here is the complete division tabulated (steps 3,4,5):

count:	5	4	3	2
quot:	0	01	011	0110
num:	10000101	10000101	11010101	11111101
denom:	10100000	01010000	00101000	00010100
sum:	100100101	11010101	11111101	00010001

count:	1	0
quot:	01100	011000
num:	11111101	11111101
denom:	00001010	00000101
sum:	00000111	00000010

quotient=00011000=24; remainder=not(11111101)
= 00000010
= 2.

The WB Usercode routine is really basic. As given, it is for division of unsigned 8 bit words, but I think it s easy enough to upgrade to 16 bits, and to add on sign routines. A stands for tha accumulator. Written alongside is an explanatory version in BCPL.

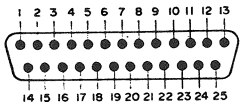
```
        MOV DENOM A
        GIZ ERROR       if denom=0 then finish
        CLA COUNT       count:=0
ALIGN   INC COUNT       f(count:=count+1
        SHL DENOM       denom:=denom<<1 f)
        GPL ALIGN       repeatwhile denom>=0
        CLA QUOT        quot:=0
        INV NUM         num:=not num
DIVL    MOV NUM A
        ADD DENOM A     f(a:=num+denom
        GCS OVER        if not carry then
        MOV A NUM       f(num:=a
        INC QUOT        quot:=(quot+1)<<1 f)
        SHL QUOT
OVER    SHR DENOM       denom:=denom>>1
        DEC COUNT       count:=count-1 f)
        GNZ DIVL        repeatwhile count<>0
        MOV A REM
        INV REM         rem:=not num
ERROR   HLT             finish
```

O.F.Morgan

RS-232 CONNECTOR PINS (also V24)

	pin		function
*	1	FG	Frame Ground
*	2	TD	Transmit Data
*	3	RD	Receive Data
	4	RTS	Request To Send
	5	CTS	Clear To Send
	6	DSR	Data Set Ready
*	7	SG	Signal Ground
	8	DCD	Data Carrier Detect
	9		+ve DC Test Voltage
	10		-ve DC Test Voltage
	11		not assigned
	12	(S)DCD	Secondary Data Carrier Detect
	13	(S)CTS	Secondary Clear To Send
	14	(S)TD	Secondary Transmit Data
	15	TC	Transmit Clock
	16	(S)RD	Secondary Receive Data
	17	RC	Receive Clock
	18		Receive Dibit Clock
	19	(S)RTS	Secondary Request To Send
	20	DTR	Data Terminal Ready
	21	SQ	Signal Quality Detect
	22	RI	Ring Indicator
	23		Data Rate Select
	24	ETC	External Transmit Clock
	25		Busy

note; you rarely find all of these signal connections used (or provided). Important ones for low speed modems are marked *



GIM EVENING

Approximately 30 members turned up to a very interesting evening lecture on the GIM series of microprocessors, held at the South Bank Polytechnic on the 25th of November.

Peter Rush of GIM described in some detail the advantages of the CP1600 (one of the few 16 bit MPU's at present available) one off price £32,and then went on to mention the LP8000, their el cheapo 8 bit CPU at £8.33. Of considerable interest were the range of EAROM's produced by GIM.

The Q & A session at the end was largely spent discussing TV games chips, a field in which GIM have at present a virtual monopoly.

Bob Warren

My LIFE program is being written in Mercury Autocode, this being the only high-level language available at Galdor.

Whilst still on the subject of LIFE, I have experimented with a triangular version;

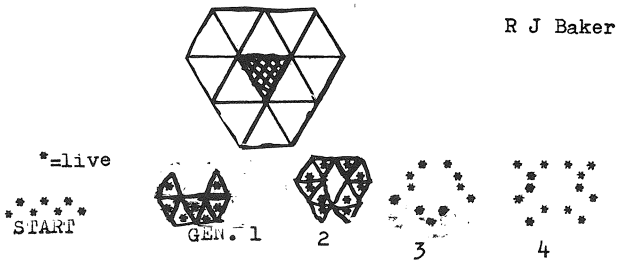
The neighbours of a cell are as shown below, and the growth rules are;

A live cell adjacent to 3,4 or 5 neighbours survives.

A dead cell adjacent to 4 live cells is 'born'.

I have run the traditional 7 cell pattern in this triangular grid, with these results;

R J Baker



Compelec Electronics Ltd.

310 Kilburn High Road, London NW6. Tel:01-328 1124 & 624 7744

ALTAIR	MAIN FRAMES
8800A	ALTAIR 8800a Computer
8800B	ALTAIR 8800b Computer
680F	ALTAIR 680 Computer
680T	ALTAIR 680 Less Front Panel
	TERMINALS
COMTER II	Terminal w/audio Cassette I/O
CT-8096	CRT Terminal
88-VLCT	Terminal for Binary to Octal Conversion
88-80LP	Line Printer and Controller 80LP w/pin-feed
88-TTY	Teletype ASR-33 and Controller
	8800 MEMORY
88-4MCS	4K Static Memory
88-16MCS	16K Static Memory
88-4MCD	4K Dynamic Memory
88-DCDD	Disc Controller & 1st Disc
88-DISC	Disc Drive in Cabinet
	680 MEMORY
680-BSM	16K Static Memory Board
	MISCELLANEOUS
88-V1	Vectored Interrupt
88-RTC	Real Time Clock
88-PMC	PROM Memory Card (Holds 2K bytes)
PROM	PROM's (256 x 8 bytes) for 8800 or 680
88-A/D	Analog to Digital Converter
88-MUX	Multiplexor and Buffer
88-PPCB	Prototype Board
88-ACR	Audio Cassette Record Interface

Haywood Electronic Associates Ltd.,
11 Station Approach,
Northwood,
Middlesex.
Tel: 01 428 9831
Tel: 09274 28301

COMPUTER KITS HEA 6800

HEA 68 Computer System with 2K RAM and Control Interface

MP–M	4K Board with 2K static RAM devices
MP–MX	Additional 2K static RAM devices for above
MP–S	Additional Serial Interface Boards
MP–L	Additional Parallel Interface Boards

HEA CT–1024 Visual Display Terminal (includes assembled, tested key tronic keyboard 1466)

HEAC–30	Audio Cassette Interface
HEDCR	Decca Cassette Recorder
HEA–12	Monitor Unit 12 ins.
HEA–15	Monitor Unit 15 ins.
HEA–40	Low cost portable T.V. set

Completed Computer System HEA–68

Completed Visual Display Terminal HEA CT–1024

Floppy Disc Unit and Interface Card for HEA–68

Cassette Unit and Interface Card for HEA–68

Keyboards in desk top consoles:–

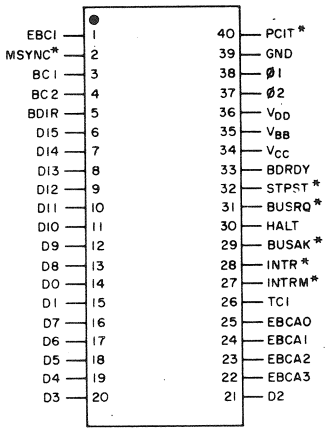
DTA–0330-100	Teletype and ASC11 53 key
DTB–1313-100	Teletype and ASC11 52 key
DTC–1066-100	Teletype and ASC11 78 key
DTD–1466-100	Capacitive 53 Key ASR33

CP1600

16-BIT MICROPROCESSOR

FEATURES

- 8 program accessible 16-bit general purpose registers
- 87 basic instructions
- 4 addressing modes: immediate, direct, indirect, relative
- Conditional branching on status word and 16 external conditions
- Unlimited interrupt nesting and priority resolution
- 16-bit 2's complement arithmetic & logic
- Status word: carry, overflow, sign, zero
- Direct memory access (DMA) for high speed data transfer
- 64k memory using single address
- TTL compatible/simple bus structure
- 400 ns cycle time with 5MHz 2-phase clock



*ACTIVE LOW LEVEL

CP1600 – INSTRUCTION SET SUMMARY

		MNEMONICS	OPERATION	MICROCYCLES Dir.	Indr.	Imm.	Stack	COMMENTS
External Reference Instructions	Arithmetic & Logic	ADD	ADD	10	8	8	11	
		SUB	SUBtract	10	8	8	11	
		CMP	CoMPare	10	8	8	11	Result not saved
		AND	logical AND	10	8	8	11	
		XOR	eXclusive OR	10	8	8	11	
	I/O	MVO	MoVe Out	11	9	9	9	
		MVI	MoVe In	10	8	8	11	
Internal Register Instructions	Register to Register	ADDR	ADD contents of Registers	6				Add one cycle
		SUBR	SUBtract contents of Register	6				if Register 6 or 7
		CMPR	CoMPare Registers by subtr.	6				Result not saved
		ANDR	logical AND Registers	6				
		XORR	eXclusive OR Registers	6				
		MOVR	MOVe Register	6				
	Single Register	CLRR	CLeaR Register	6				XORR with itself, except*
		TSTR	TeST Register	6				
		JR	Jump to address in Register	7*				PC ← (RRR)
		INCR	INCrement Register	6				
		DECR	DECrement Register	6				
		COMR	COMplement Register	6				One's Complement
		NEGR	NEGate Register	6				Two's Complement
		ADCR	ADd Carry Bit to Register	6				
		GSWD	Get Status WorD	6				
		NOP	No OPeration	6				Two Words
		SIN	Software INterrupt	6				Pulse to PCIT pin
		RSWD	Return Status WorD	6				
		PULR	PULl from stack to Register	11*				PULR = MVI@R6
		PSHR	PuSH Register to stack	9*				PSHR = MVO@R6
	Register Shift	SLL	Shift Logical Left	6				one or two position shift capability. Add two cycles for 2-position shift
		RLC	Rotate Left thru Carry	6				
		SLLC	Shift Logical Left thru Carry	6				
		SLR	Shift Logical Right	6				
		SAR	Shift Arithmetic Right	6				
		RRC	Rotate Right thru Carry	6				
		SARC	Shift Arithmetic Right thru Carry	6				
		SWAP	SWAP 8-bit bytes	6				2-position=SWAP twice
Control Instructions		HLT	HaLT	4				
		SDBD	Set Double Byte Data	4				Must precede external reference
		EIS	Enable Interrupt System	4				to double byte data
		DIS	Disable Interrupt System	4				
		TCI	Terminate Current Interrupt	4				Not Interruptible
		CLRC	CLeaR Carry to zero	4				
		SETC	SET Carry to one	4				
Jump Instructions		J	Jump	12				
		JE	Jump, Enable, interrupt	12				
		JD	Jump, Disable interrupt	12				
		JSR	Jump, Save Return	12				Return Address
		JSRE	Jump, Save Return & Enable	12				saved in R4, 5 or 6
		JSRD	Jump, Save Return & Disable Interrupt	12				
Conditional Branch Instructions		B	unconditional Branch	7				Displacement in PC+1
		BC, BLGE	Branch on Carry, C=1	7				PC ← PC ± Displacement
		BNC, BLLT	Branch on No Carry, C=0	7				Add 2 cycles if test condition
		BOV	Branch on OVerflow, OV=0	7				is true.
		BNOV	Branch on No OVerflow, OV=0	7				
		BPL	Branch on PLus, S=0	7				
		BMI	Branch on MInus, S=1	7				
		BZE, BEQ	Branch on ZEro or EQual	7				Z=1
		BNZE, BNEQ	Branch if Not ZEro or Not EQual	7				Z=0
		BLT	Branch if Less Than	7				S∀OV=1
		BGE	Branch if Greater than or Equal	7				S∀OV=0
		BLE	Branch if Less than or Equal	7				Z ∨ (S∀OV)=1
		BGT	Branch if Greater Than	7				Z ∨ (S∀OV)=0
		BUSC	Branch if Sign ≠ Carry	7				C ∀ S=1
		BESC	Branch if Sign = Carry	7				C ∀ S=0
		BEXT	Branch if External condition is True	7				4 LSB of Instruction are decoded to select 1 of 16 external conditions.

1 MICROCYCLE = 2 CLOCK CYCLES

BUS CONTROL SIGNALS

BDIR	BC1	BC2		DECODED FUNCTION
0	0	0	NACT	- No ACTion, D0-D15 = high impedance
0	0	1	IAB	- Interrupt Address to Bus, D0-D15 = Input
0	1	0	ADAR	- Addressed Data to Address Register, D0-D15 = high impedance
0	1	1	DTB	- Data To Bus, D0-D15 = Input
1	0	0	BAR	- Bus to Address Register
1	0	1	DWS	- Data Write Strobe
1	1	0	DW	- Data Write
1	1	1	INTAK	- INTerrupt AcKnowledge

D0-D15=Output

PROCESSOR CONTROLS

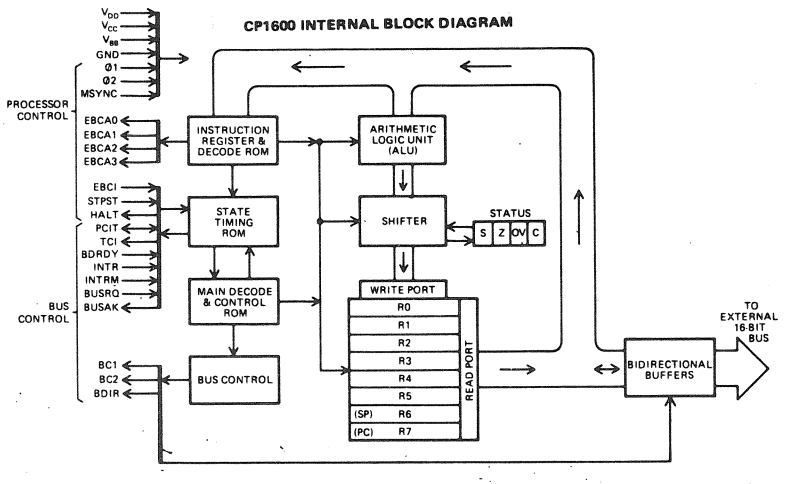
STPST* (Input)
STop-STart: Edge-triggered by negative transition: used to control the running condition of the microprocessor.

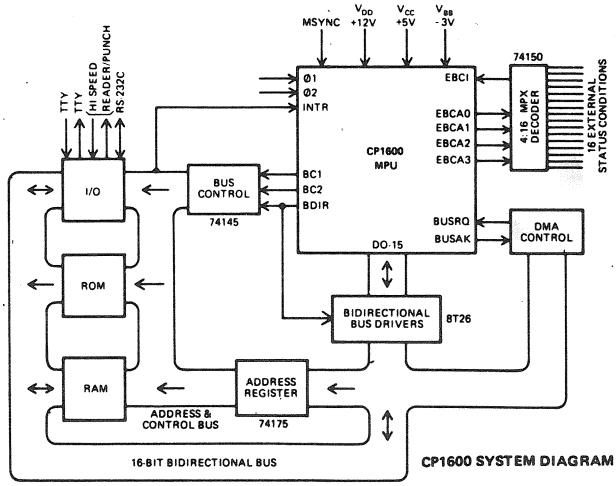
HALT (Output)
HALT: Indicates that the microprocessor is in a stopped mode.

MSYNC* (Input)
Master SYNC: Active low input synchronizes the microprocessor to the ϕ1, ϕ2 clocks during power-up initialization.

EBCA 0-3 (Outputs)
External Branch Condition Addresses 0-3: Address for one-of-16 external digital state tests via the BEXT (Branch on External) instruction.

EBCI (Input)
External Branch Condition Input: Return signal from the one-of-16 selection made by EBCA 0-3.





The CP1600 is one of the 16 bit minicomputer-like MPU's that must surely be more attractive to the amateur computer enthusiast than a lot of the 8 bit chips around, and for £32 is not much more expensive.

From a hardware point of view it seems very straightforward - see the 'System Diagram' on the facing page. Address and data are multiplexed onto a 16 bit bus, allowing GIM to squeeze the CP1600 into a 40 pin package (the TI 9900 uses a 64 pin package).This is no disadvantage as buffers will in practice be needed for data and address lines, and these can provide the de-multiplexing function. +12 and -3 V supplies are required as well as +5V, but any worthwhile computer system must have supply rails other then +5V anyway if only to drive 20mA current loop or RS232 interfaces. The Ø1 and Ø2 clocks are fairly undemanding; nominally +12V non-overlapping. It is not a static device, so the clock period must not be more than 2uS (0.2uS minimum), but by pulling the BDRDY line low, the MPU can be held for up to 40uS to wait for slow memory. Two interrupt lines (masked and non-masked) are provided.

The most peculiar feature of the CP1600 is that it only uses 10 of the available 16 bits for instruction words. Generally these ten bits are divided into three fields;

15 - - - - - 10	9 8 7 6	5 4 3	2 1 0
Not used	Operation	F1	F2

It is said that this allows one to store a program in 10 bit wide ROM. It is also rumoured that the CP1616 - an enhanced MPU to be released mid 1977 - will make use of bits 10 - 15.

Of the 8 16 bit registers, R6 is a general purpose stack pointer, and R7 the program counter.

For register to register operations, F1 and F2 specify the registers to be used. For external reference instructions, the source is external memory as specified by F1, and the destination one of the 8 registers as given by F2. F1 determines the memory address as;

```
F1 = 000  Direct address, in next mem location
     001  Indirect, address in R1
     010    "          "    "  R2
     011    "          "    "  R3
     100    "          "    "  R4, R4 + 1 → R4
     101    "          "    "  R5, R5 + 1 → R5
     110  Stack,thru R6
     111  Immediate data, in next location
```

The Set Double Byte Data instruction allows the MPU to obtain a 16 bit word from 2 consecutive 10 bit wide words (if you are using 10 bit ROM).

The Branch instructions use two consecutive 16 bit words, the first contains the Op Code, the second a 16 bit displacement which may be added to or subtracted from the PC. This allows relative branching over the whole of the address space (65k words) As shown in the System Diagram, up to 16 external conditions can be tested by the Branch instructions.

The Jump instructions occupy three consecutive memory locations (10 bit ROM's again !) and can allow you to save the old value of the PC in register 4,5 or 6.

The CP1600 is available from Semicomps Ltd., and SDS Components Ltd. A faster version, the CP1600A, is also available. A manual costs £10.

BOOKLIST

At the present moment you print a list of books that have been recommended by members. If they are willing to do so, and you have the space to print them, perhaps they could give a short review of the book. I enclose a couple of reviews for two books that I would like to recommend, to set the ball rolling as they say. (printed elsewhere in this issue ed)

Since they are interested in computers I think it would be fair to say that most of the members are in a scientific enviroment of some kind. Therefore why not include articles that are scientific in nature, but not necessarily computer oriented, e.g. a mathematical member could write about probability in games of chance, or explain the birthday date paradox which most people know about, but not very many can explain or nkow the reasoning behind it.

Philip Rutherford

BOOK REVIEWS

MATHEMATICAL PUZZLES AND DIVERSIONS
(ISBN 0 14 020713 9)
MORE MATHEMATICAL PUZZLES AND DIVERSIONS
(ISBN 0 14 020748 1)

Both are by Martin Gardner and published by Penguin

A couple of interesting books from Martin Gardner. They consist of collections of his more popular articles from 'Scientific American'. He examines and analyses various games and puzzles - some well known and some not so familiar - with names like Hexaflexagons, Polyominoes, and the Icosian Game.

The book is very readable and despite the rather ominous title, the most complex mathematical formula used is Pythagoras Theorem.

His second book carries on from the first and mentions brain teasers like diabolic squares, the Soma cube, mazes and magic squares; and anyone who doesn't know the Generalised Ham Sandwich Theorem has never lived and just can't afford to be without these books.

These books are guaranteed to keep you away from the TV for hours, and as such their meagre pittance of a price is the best 30p you will ever spend.

Philip Rutherford

INTEL VISIT

The premises of Intel at Oxford were inundated by 50+ members on Saturday the 27th of November. A slightly overwhelmed Keith Chappell (Marketing) and Howard Kornsteil (Applications) gave descriptions of the present 8080 range of MPU & peripherals and unveiled the new 8085 series.

The 8085 requires only a 5V power supply and may in turn replace the 8080 in the way that it replaced the 8008.

After a long and useful question and answer session we were allowed to play with the various computer systems scattered about the workshop, and finally loaded with lavish amounts of freely given bumph, limped homewards happy & contented.

During the visit Keith Chappell mentioned the possibility that Intel may be able to offer ACC members the SDK80 kit at an advantageous price. If this comes to anything details will be printed in the ACCN.

Bob Warren

HELP

Does anyone have the pin connections for the Cannon plug on the back of a Flexo model FV - or even a complete manual ?
O Morgan
Norcombe, Tile Barn, Woolton Hill, Newbury, Berks RG15 9UZ. tel; (0635)253412

I have just bought a 13 bit by 16k Plessey 1uSec core store type 68/1339, Ser No 20715, Ref No 13 bit fast, made for ICL, and would be most grateful if any ACC member could supply or lend some data on it (any costs reimbursed). It is very tightly packaged (10"x15"x2½") and unintelligent probing could be fatal. The supplier was Servo & Electronic Sales (Wireless World Nov.)

I plan to use it with the IM6100 kit (ACCN Aug,) when this arrives. There seems to be some considerable delay due to 'production difficulties' at Intersil.

I have an ASR35 teletype ex Chiltmead (now out of stock). This turned out to have an obscure 'Gothic' ASCII coding and to convert to 'real' ASCII the typebox must be rearranged and the function decode

bars (CR.LF etc.) altered with the aid of a blow-lamp. Nevertheless it was a good buy at the price.

I recently acquired two ancient dictaphones (Rex recorders) which record on 8½" dia plastic magnetic coated discs. If the dropout rate isn't too high (rather hopeful, I suspect!) they could make useful 'random' access bulk storage devices having faster rewind and file search capabilities than cassettes.

Chris Doran
89 Lennard Road, Penge, London SE20 7LY

Recently I bought two stacks of Plessey 13 x 16k core from Servo & Electronics in Lydd. My problem is that Servo did not supply any data on the core (simply because they had none).

On writing to Plessey and asking for any information on the core, I was told that I had part of a 'Series 1000 core store' and that a drawing package and test specification would cost me £30. (Presumably to cover staff time etc.)

The thought occurs to me that some other member of the ACC may already have paid the amount that Plessey asks and have the data I need. So, if I could contact him (or her) it might be possible to come to some arrangement to pass on information.

Failing that, there might be others in the same position as me, so perhaps we might club together to spread the cost.

p.s. Eventually I hope to use the core as part of an Intersil 6100 based version of the PDP8. Suprisingly enough very little is heard about this in the amateur electronics mags - everyone seems to be concentrating on the more recent 8 bit creations from Motorola, Intel etc. What puzzles me is WHY ? Have I overlooked a major fault of the 6100 or has this interesting machine been overlooked by authors of 'Build it Yourself' articles ?

Richard Cogliatti
6 Woodcote Lane, Purley, Surrey CR2 3HA

LETTERS

TTL BUGS

I'm building an 'electronic animal' using TTL logic for its 'brain'. It's fairly crude but it has sparked my interest to try and expand on it. Is anyone else interested in this sort of thing.

G Simpson,20 Wright Ave.,Stanground,Peterborough

6800 MICROPROCESSOR TIPS

Following the purchase of the kit I have learnt the following;

1) That expensive 43 way 0.156" pitch connector is not necessary to start with as only 0V and +5V are used initially and 0.15" offcuts 3 pins deep (2 off) are quite adequate.

2) Hidden extra RAM is available. Memory locations A04A to A07F can normally be used.

3) The ROM Mikbug has plenty of hidden goodies via the Jump to Subroutine command (JSR)=BD

a) BD E1AC inputs one character to Acc A

b) BD E07E outputs all the characters in the mem pointed at by the index register until 04 is reached.

c) BD E1D1 outputs one character from Acc A

d) BD E0CA outputs 2 characters pointed at by the index register.

e) BD E0C8 outputs 4 characters pointed at by the index register.

f) BD E0E3 returns control to MIKBUG without using RESET button.

I do not claim to have discovered these but I have tested them. I'm sure that there are plenty more useful items within MIKBUG to be found.

I would very much like to contact and exchange software with other 6800 users. Perhaps a sub-unit within the ACC ? It would be a great shame to let an opportunity to organise a software pool pass. It would be great if some maniac would mod WB1/2 to be software compatible. Now there's a challenge !

Tim Moore
24 College Rd., Maidenhead, Berks SL6 6BN
tel; (0628) 29073

CASSETTE AGAIN

I am trying to build a cassette interface, inspired by the HITS system and an article on analogue to digital plus some help from Don Lancaster's TTL Cookbook etc. The system will be based on 2kHz carrier, FM modulated up or down depending upon data '1' or '0'. I expect I will use a crystal controlled clock with TTL countdown and a switch on front panel to cater for various baud rates.

I emphasise that although inspired by HITS it is not a HITS system and the usual 8 bits of data will end up as 11, extra '1's on the end and all data has '0' start bit. Should work, anyway.

By the way, I have never seen the CUTS article in Popular Electronics, I gather it was in the March & May 1976 issues - could anyone let me have photostat copies of this article plus any amendments they know of , or sell the mags ? I shall be plaesed to refund postage costs etc. It may save me a lot of trial and error.

J J Smith,7 Kettlebaston Rd,Leyton,London E10 7PE
tel; 01 556 3368

STATES-SIDE

I have just got back from a three month trip to the States (Boston Mass) and so have had a chance to sample the American computer scene. I attended several meetings of the New England Computer Society (about 200 members) and was astounded to see so many people from one area attending, and about three-quarters seemed to have their own machines (mostly 6800 and 8800). I also went to a smaller club - the Boston Computer Club - which was very interesting.

In the area I was staying there were 2 computer shops for the hobbyist and they had in stock everything from processors, floppy discs, displays etc. down to magazines and books. Needless to say I was unable to stop several hundred dollars escaping from my wallet. Actually the main things I bought were a video display from the Digital Group as was described in one of the issues of Byte, plus a South West Tech. keyboard kit which I have now finished, and by the way this kit is very easy to build and the instructions are very good. I also bought a lot of wirewrapping tools, boards, sockets and components plus lots of books and magazines. For anyone interested I would like to recommend the TV Typewriter Cookbook by Don Lancaster ($9.95) this is very good, and The Best Of Creative Computing by David H Ahl has many interesting notes and programs (mostly in BASIC) $8.95.

Ian D Spencer

OTHER-SIDE

I thought you might like to hear some news and views from this part of the world. Well, to begin with there are not many computer hobbyists in Singapore (total population is a little over 2 million). Most of the chaps who dabble in computers are professionals or students at the local University or Polytechnic. In fact I seem to be an odd-ball, at least to the local electronic parts shops, for I keep asking them for odd components like ASCII keyboards, which they have never even heard of.

Recently, however, I discovered that the local Motorola agent was offering the M6800 Design Evaluation Kit at the equivalent of US $149. I bought one of them and they're worth the money for the kit includes the MPU, ROM, 2 RAM's, 2 PIA's ACIA and

documentation (Instruction, Programming and Applications manuals) to boot. I'm now on the way to building a system round the 6800 MPU.

The snag is that the kit requires a terminal (TTY or RS232) to work and some odd IC's that I'm unable to buy locally. I've sent for the CT-1024 Video Terminal kit which was mentioned in a recent newsletter and hope to get it working in a couple of months.

My view is that there is little point in publishing articles on the VDU project in the newsletter as there are enough of them in the commercial mags, I feel the newsletter should be more for information interchange between members. I would be pleased to correspond with other ACC members who are using the M6800 MPU.

Soh Chio Siong
67E Blk 3 Marine Vista, SINGAPORE 15

SPRECHEN SIE MACRO ?

Because the larger magazines (e.g. Electronics Today International) are now supporting micro processor designs the impact of ACCN in the hardware field is considerably lessened. Rather than let the ACCN become just a gossip column, it would be much more preferable to change direction and lead into the field that everyone, once they have built their micro processor, will be crying out for - software.

Previous attampts to write general software in the ACCN have come to grief because not everyone has the same high level language, let alone machine configuration. It should be remembered that we are interested in software produced for the hobbyist and therefore, because of the restrictions on core size, code efficiency is essential. To this end, ideally the best choice is a language specially constructed for each particular machine. But if we are to achieve any form of software portability this approach is impractical.

Whilst almost all computers have numerical operators (addition, subtrction etc.), they all have different methods for manipulating data and characters, hence it is not a simple job to formulate a series of non-numeric operators that would be generally acceptable. Because of the inherent problems in constructing a high level language that could be used to encode non-numerical algorithms, we must seek a different solution.

To combine portability with efficiency we should not be attampting to write in a pre-defined high level language, but rather in a descriptive language in which its operators are defined to be exactly those required by the software being described. For example, if a sample of software used queues, then the operators of the descriptive language should include queuing operators. These could take the form;

```
LOADQ    argument
UNLOADQ  argument
```

An ideal solution to achieve this descriptive language is to use a macro processor (a macro is a means of exchanging one series of symbols for another). The function of a macro processor is to scan a source text for occurrences of the macros defined and replace the code accordingly.

Therefore taking the example of the queuing operators, for each implementation macros would be written to map these operators into the appropriate assembly language instructions.

For each statement type there needs to be a corresponding macro in the descriptive language. Most assembly languages permit the use of identifiers as the variable names and therefore no changes would be needed in this area.

Example routine;

Routine function is to add parity to a character prior to output. The type of parity, whether odd or even, is declared within the program, as are the bits parity is to be calculated on, and the bit position of the parity bit.

The routine leaves the character (with parity) in the accumulator.

The routine is passed the character to be output in the variable 'CHAR'.

A local work space 'WS' is declared to calculate the parity bit.

```
        DEF     WS
        LAL     +PTYPE
        STO     WS
        LAV     CHAR
        ANDL    +MVAL
        STO     CHAR
PAR:    INC     WS,     +1
SHIFT:  LEFT    CHAR,   +1
        JMPZ    FINISH
        JMPN    PAR
        JMP     SHIFT
FINISH: LAV     WS
        ANDL    +1
        JMPZ    EXIT
        LAL     +PVAL
EXIT:   AAV     CHAR
```

macro named literals used ;

PTYPE -defines type of parity to be formed
value '0' for odd parity
value '-1' for even parity

MVAL -defines which bits to be used to form the parity bit

PVAL -defines value of parity bit (i.e. if parity bit in eighth bit then value = 128)

the following is a list of the statements in the example routine;

AAV	V	add variable to acc.
ANDL	signed integer	'and' accumulator with literal
DEF	V	define variable
INC	V,signed integer	increment variable by integer
JMP	label	unconditional jump
JMPN	label	jump if negative
JMPZ	label	jump if zero
LAL	signed integer	load acc with literal
LAV	V	load acc with variable
LEFT	V,signed integer	shift variable logical left by a literal
STO	V	store acc in variable

Apart from those macros used in the example routine the following need including as a bare minimum;

AAL	signed integer	add literal to acc
DEC	V,signed integer	define variable and equate to integer
END		exit from procedure
EQU	V,V	equate two variables
FINISH		end of logic
JMPRO	procedure name, parameters	jump to procedure
MULT	signed integer	multiply acc by literal
NB	'characters'	comment
NULL	V	set variable to zero
PROC	procedure name, parameters	declare procedure
RIGHT	V,signed integer	shift variable logical right by literal
SAL	signed integer	subtract literal from acc
START		start of logic
STR	'characters'	output the string
SUB	V,signed integer	subtract literal from the variable

These are by no means all the macros required to write all types of software, merely basic building blocks. Additional macros can be defined as required.

N.Pearce & N.Mannerings

SMAB & SUPER-SMAB

I wonder if I might use the columns of ACCN to do some 'consumer research'. I am at present involved in an A-level Computer Science project. This is a program, written in BASIC, to translate a higher-level version of BASIC into normal BASIC.

The features of the higher level BASIC (SMAB - Super Macro Assembler Basic) are as follows;

1) No line numbers but symbolic labels. Labels are suffixed by a colon. Addresses prefixed by a decimal point.

2) IF (arg) THEN PRINT A , or any other statement following 'THEN'. This is translated into;

```
      IF NOT (arg) THEN .XXX
      PRINT A
 XXX: REM
```

3) LIF (exp) .ABC,.DEF,.XYZ

```
(exp) < 0  - GOTO .ABC
      = 0  - GOTO .DEF
      > 0  - GOTO .XYZ
```

4) SOUT A$(50)
outputs the first 50 elements of A$ (1 char per element) as a single string

These are the statements incorporated in the program at present but I have plans to extend it to other functions. At present the ideas I have include a software stack (an array with a stack pointer) and block data. This would be a statement ('BLOCK' for instance) which would cause all following lines to be copied across with the word 'DATA' preceeding them, until an 'UNBLOCK' statement was encountered.

I hoped some people might be inspired to write to me with their hang-ups about BASIC so that I can write the software to get around them. Although languages such as Algol 68 lack these problems, they also have the disadvantage of non-conversationality. With this system, which is not pretended to be efficient, the object code is in a readily comprehensible form.

If the system proves especially useful, maybe someone would like to write some software to run the new language directly ? !

Paul M Jessop (G8KGV)
1157 Warwick Rd.,Solihull,West Midlands B91 3HQ

2650 BEATS WB

When it seemed that WB2 was not on, I thought hard and long about WB-1, with the result that I decided not to build it. Study of available MPU yielded the 2650, and I have now obtained one such chip.

It is easy to interface and is static. No weird clocks. The most minicomputer like MPU there is. Currently I am planning the whole system i.e.;

Front Panel
CPU board (simple TTY I/F) } INITIAL SYSTEM
RAM board (1 or 2 k)

Core Store (1k)
Serial Interface } LATER ENHANCEMENT
Cassette Interface

and of course PSU

I obtained an old Teletype 35RO and a keyboard which uses cores. It is my intention to fit 20mA current loop to the 35RO & build PSU, also to rebuild keyboard and later to fit simple scrolling type 24 line by 40 char/line VDU into spare space. I already have a small 5" closed cct type TV monitor. Also have 8 hole PTR but await some results from CHIC for 8 hole punch (aren't we all ! ed.)

The plan is to expand store as time passes. Hopeful future configuration will allow two pass assembler. i.e. key in program via keyboard, display on VDU, dump to PTP (source prog). Read paper tape via PTR on first pass to obtain symbol table, 2nd pass to obtain output to cassette in object code. Listings taken as required, using perhaps a third pass if necessary. During the keying in phase it would be possible to use cassette as backing store since dump to produce source prog tape could be cassette via CPU to PTP, thus saving store required for source listing which will of course have comment fields as well as source code names.

Punch and reader may well end up as serial (V24) devices to make the interface between system components compatible. This seems to be the modern trend according to the electronics mags.

I like the ACCN but wish there were more software contributors.

Useful idea from Byte magazine; a way to use a single 4k RAM ; make a single bit wide store and use to store data bytes in 8 consecutive addresses. Great for sense anf flag bits on 2650 when using software 'UART'.

Mike Baker

CHIC

The story so far - -

As no doubt you will remember, the CHIC system was announced in the June 1976 edition of the ACCN. To date a total of approximately 20 members have sent in their SAE's for the various categories, and five 'For Sale' notices have been circulated to the appropriate subscribers. I am only firmly aware of one successful transaction, but no doubt there were others. Small beginnings, but worthwhile, I think. As suggested by one of the subscribers, an eighth category; CORE MEMORY, is to be added, so if you are interested please let me know.

Bob Warren

ETI VDU PLUS

ETI VIDEOWRITER MODIFICATIONS & ADDITIONS

D E Howland

To fit a keyboard with cursor control to the ETI Video Writer, the following modifications and additional circuitry are needed;

1) BOARD B MODIFICATIONS (CURSOR)

No additional components are needed as this modification uses spare gates already present. The new circuitry is shown in Fig 1.

First, the inverter in IC6 which combines the outputs of gates IC'a,b,c d and IC9 is replaced by a NAND gate (IC9c plus a 2k2 resistor). Then IC15b and IC9d are used to gate together RCLK, CURSOR ON/OFF, BLNK (ungated - IC28 pin 3) and COMP (see Board C mod) to produce a white underlining cursor. Due to the existing circuitry, this cursor is under the last character printed, not the next character to be printed, but this is not a serious handicap.

2) BOARD C MODIFICATIONS (KEYBOARD STROBE AND 'CLEAR SCREEN')

Only one extra 7400 NAND gate pack is needed here, as there are two gates unused on the board. The circuit, shown in Fig 2, firstly allows a strobe input from the keyboard (via Board D), and secondly gives a continuous write condition to the RAM's if the CLEAR input is low. The signal from CCLK is used to prevent double characters appearing sometimes as happened on the original.

3) BOARD D (NEXT CHARACTER POSITION AND CURSOR CONTROL)

This is in two parts as follows;

a) D(1) - LF,RET & CLEAR PAGE decoder

For this, the standard codes are used for return (RET) and line feed (LF), but the CLEAR PAGE key must be designed to give 1100000 (B6 is MSB, B0 LSB). The decoder is shown in Fig 3. This gives the appropriate LF,RET and CLEAR outputs and also a strobe output which is inhibited when LF and RET are present. This is to prevent the next character counter from being incremented by these commands.

b) Next Character position counter and control

See Fig 4. This uses 3 SN74193N up/down counters as the position counters. They are wired so that the first two give the position of the character on the line, and the third one gives the line number. The gating arrangement allows the cursor to be moved up, down, left, right or to the home position (top left corner). Also, RET will reset the first two counters, and LF increments the third counter. The monostable produces very narrow pulses (approx 1uS) to overcome switch contact bounce.

4) KEYBOARD (Fig 5)

There are also a few keyboard modifications which are desirable. Firstly, two 7475 latches are used to hold the keyboard data to prevent the data changing before the strobe pulse returns to a low level. Secondly, the 7493 counter & 74155 decoder arrangement is a further method of removing persistent contact bounce, and also makes sure that the data on the latch outputs is

stable before the strobe occurs. The circuit might not need to be so complicated, but this has been found to be very efficient in practice.

5) MICROPROCESSOR INTERFACING

The easiest method, which requires no extra software in the microprocessor, is to use a Universal Asynchronous Receiver/Transmitter.

For this, the keyboard is disconnected from the VDU and connected to the Transmit side of the UAR/T. The receive side is then connected to the VDU (see Fig 6).

These circuits have not yet been built, but are in the process of being so. However they should work in their present form. If not - further details will be sent to the ACC Newsletter.

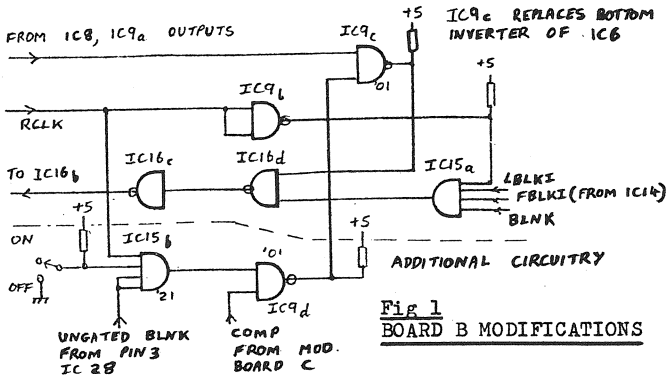


Fig 1
BOARD B MODIFICATIONS

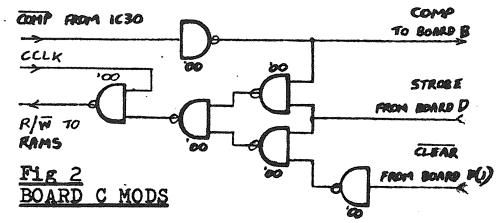


Fig 2
BOARD C MODS

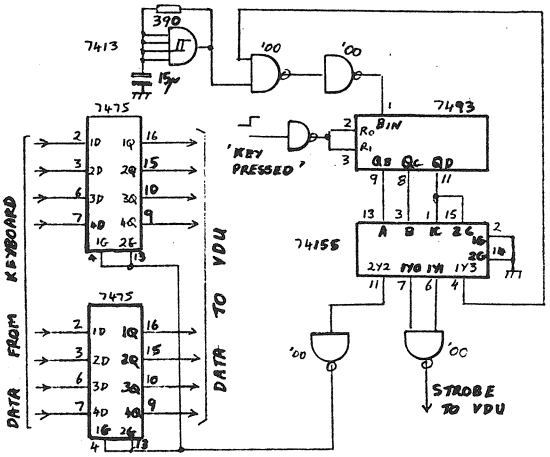


Fig 5
KEYBOARD ADDITIONAL CIRCUITRY

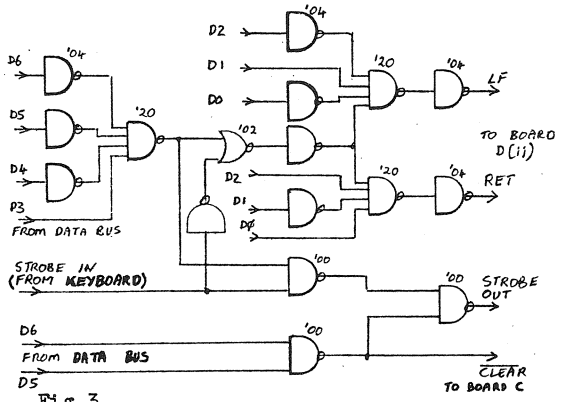


Fig 3
BOARD D(i) LF, RET & CLEAR SCREEN DECODER

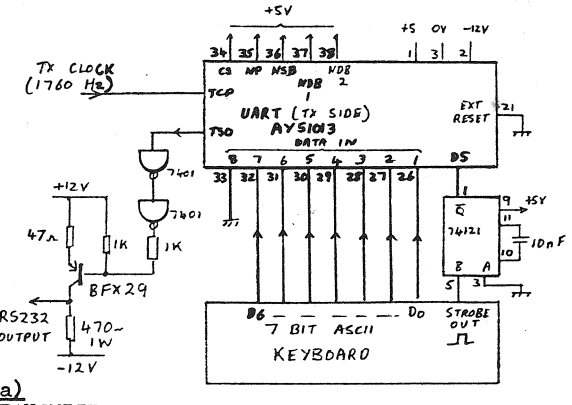


Fig 6(a)
UART TRANSMITTER

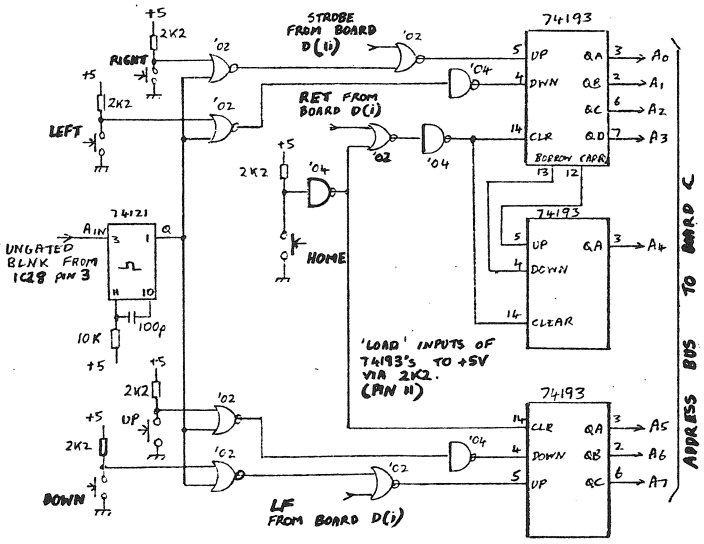


Fig 4
BOARD D(ii) - NEXT CHAR & CURSOR CONTROL

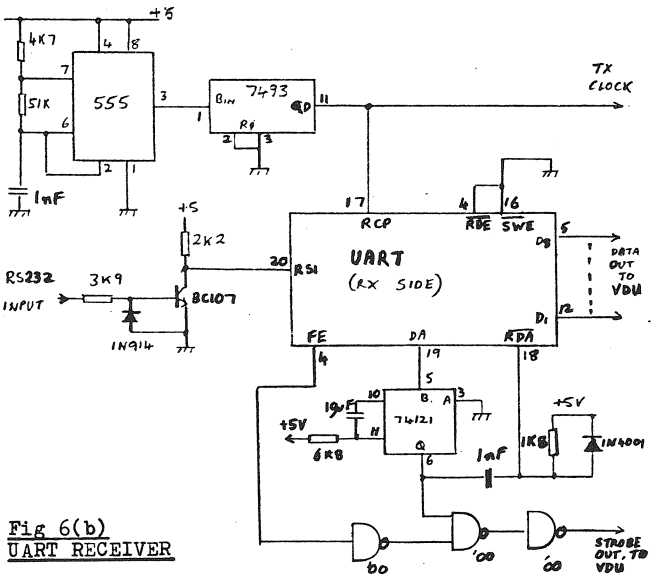


Fig 6(b)
UART RECEIVER

WHITHER ACC

In another part of this issue of the newsletter you will find notice of an extraordinary general meeting of the club. This meeting has become necessary because of the pressing need to increase the annual subscription fee paid by members, and this can only be done by a summoned meeting.

The main expense in running the Amateur Computer Club is the cost of producing and distributing the newsletter. Inflation is adding to these costs continually and in order to maintain the present number of pages and distribute worldwide a rise in subscription fee of at least 50% is required.

The exact nature of the future role of the club must be considered because if activities are to be increased and sponsored projects are to be undertaken then the subscription fee must be increased much more than that required just to counter inflation.

Please, if you can, attend this meeting, and in the meantime reflect upon the type of ACC that you wish to have in the future. Do you want just a correspondence club as we are at the moment, or a more active organisation ?

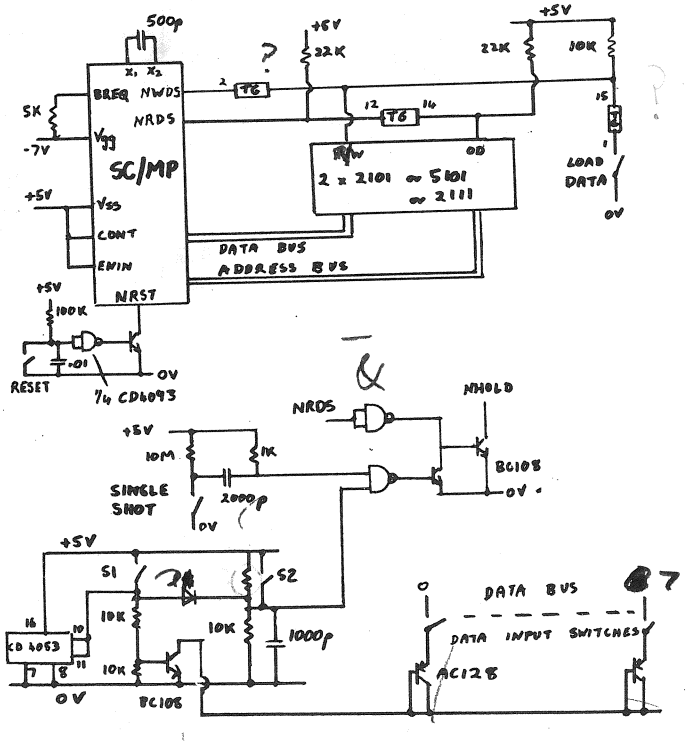
Come along, and speak your mind !

Bob Warren, Chairman

SCAMP

SIMPLE SC/MP BASED COMPUTER

With S1 on and Data input switches all on, processor will read a HALT instruction when single shotted, and Address bus will increment with every other single shot. In between, data can be entered with input switches and Load Data switch. To load data into high memory a Jump instruction could be entered with the data switches rather than incrementing all the way. A store instruction would make the processor write data on the bus with the switches enabled but this is unlikely to do any harm as the hold only operates on a read cycle.



TG Through Gates of a CD4053 (Nos = pin Nos)

S1 & S2 Off = run
S2 On = Hold - memory to data bus for debugging program

S1 On or
S1 & S2 On = Hold - switches to data bus for loading memory

Also 3 x CD4069 Hex Inverters driving LED's for address bus and data bus display. 10k pull-ups are probably needed on the address bus, not shown.

J Owen

WB CORNER

WB-1 MODS

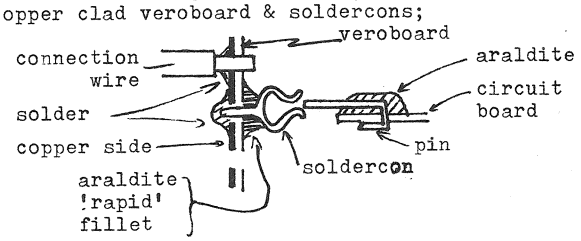
Thought you'd like to know the WB has been up & running since Sep 6. I've had a lot of fun programming sequential flashes for front panel lights etc. as well as 2 digit multipliers and dividers. The latter have been used in the usual 'starter' programs - prime numbers, factrials etc. I also programmed the 'Computer Dating' algorithm from a past N/L, plus a lot of other odds & ends.

I ran out of storage once or twice, so the following changes were evolved to allow storage - storage instructions. They are defined as 210-217, and save 1 byte per logical instruction. Thus;

	MOV X,A	020 XXX	
	MOV A,Y	060 YYY	4 bytes
becomes	MOV X,Y	210 XXX YYY	3 bytes

The main limitation is that these new instructions are direct only. For indirect, it would have to be made clear which of the operands, if not both, were to be treated indirectly, & I guess we'd run out of bits. A number of new states would have to be defined too.

Some comments on construction; first I must agree with the remark in the October N/L regarding lists of pin connections. This is absolutely mandatory. I also drew up lists of internal & external connections, crossing off each joint as made. I must admit to one or two wiring arrors, but I'm sure there would have been more but for this system. For those with difficulty accquiring veroboard of the right size, Trampus do unclad sheets of 0.1" matrix 17" x 3¾". You cut them in half & Evostick them edge to edge, & they are surprisingly rigid. I also had problems with edge connectors, which I solved with copper clad veroboard & soldercons;



Bit fiddly, but its cheap & it works. I have no idea of the availability of orthodox 0.1" pitch edge connectors. Incedentally, the system works best for upwards of about 6 connections to a strip.

Herewith a Dice program. Start it up at address 005, knock out SW0 and SW2, and set SW7. It stops. Unset SW7, press Start (shake the dice) then set SW7 again to throw. Data display shows your score.

G.D.Hayes

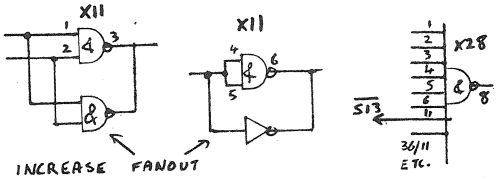
DICE

005	L1	BIT	#200,001	056 200 001
010		GIZ	L2	032 023
012		INC	A	101
013		CMP	#007,A	117 007
015		GNZ	L1	033 005
017		MOV	#001,A	110 001
021		GTO	L1	030 005
023	L2	TST	A	104
024		HLT		000
025		GTO	L1	030 005

note; Loc 001 is switch register.

Hardware Mods.

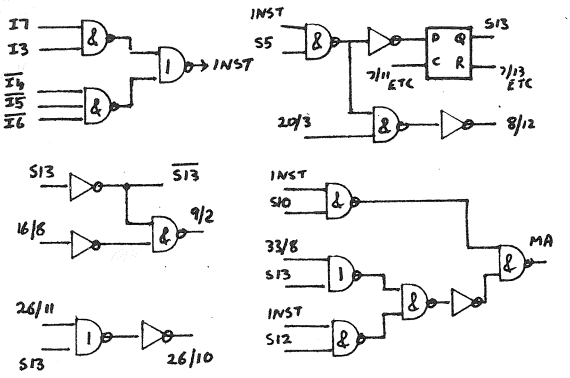
These could probably be more elegant, but the junk box came into play !



AMATEUR COMPUTER CLUB NEWSLETTER
Vol 4 Iss 5 December 1976

Data !!!

Castle Lloyod, Pendine, Dyfed SA33 4PT.



WB-2 THOUGHTS

I am greatly heartened by the recent enthusiasm for WB2 and thought you might like to hear about the thoughts I have been having on a suitable implementation. I'd be grateful for comments, either privately or through the ACCN.

I propose modifying the addressing modes specified originally, and adding a couple of instructions. It is likely that typical small programs or subroutines will not extend beyond one 'sector' of 256 bytes; accordingly I am anxious to retain all 4 addressing modes originally described, all referring to single byte - present sector- addresses. Call this the short form address mode; most programming can take advantage of it and not waste bytes on redundant addressing. Clearly, sector boundaries can be crossed using the relative address mode. In addition I suggest an extended address mode, with the following implementation;

Mode bits	Effect
0	the immediate 2 byte address is the relevant location.
1	the immediate 2 byte number is the address of the low byte of the address, high byte in XX XXX +1
2	the immediate 1 byte number is to be added to the PC (2's complement form for negative)
3	the immediate 2 byte address is the location of a 1 byte value to be added to the PC.

I see no need for relative jumps of greater than ± 128 bytes ! The short form / extended address mode is selectable under program control using the new (miscellaneous) 1 byte instructions;

SSA set short form addressing flag
SEA set extended addressing flag

These codes might be effected in state 4 with the present configuration.

My other new instructions concern the stack necessary for subroutines/interrupts. I envisage using a separate TTL stack 1f 16 words by 16 bits (7489's). This stack length seems adequate for most purposes, and can be expanded easily enough anyway. Such a stack could be added to an existing WB-1 without reducing the available store. Use of the stack by subroutines and interrupts will be automatic, but in addition the stack can be used as workspace with the following commands;

POP A top of stack→A; move stack up
PSH A move top of stack down; A→top of stack
STX A A ⇄ top of stack

By using high speed memory for the stack, stack operations should be able to be completed within the existing states.

The extended addressing - as I see it at present at least - will centre araound a dedicated adder unit, say using 7483's, for determining relative addresses. I would like to minimise the number of extra states required so that I can use much of my existing WB1 circuitry. The trade off against microprogramming is that the memory boards will be substantially different to those in WB 1.

Rom Mount, 13 Senlac Rd.,Romsey,Hants SO5 8RE

My WB-1 is now finished and working. I've not tested all the instruction set yet but it runs through the short programs from ACCN. Incedentally R Mount's program FLASH (ACCN 4.2 p5) seems to have instruction at address 13 missing. Putting 000 here makes the thing run.

I now have a Tally BP30 punch and a R50 reader. Is there any bootstrap I can key in to get reader to run ? Is there any loader tape that will enable me to load programs from tape into memory starting at a defined address ? I suppose I could work this out for myself but I would like to keep in step with other WB users, and not lead to a proliferation of operating modes.

I entirely agree with the ED'S BIT about the fun of building a mini up from bits rather than doing it by MPU. I certainly am interested in helping to develop a WB2.

My ultimate hope is to put my model railway under WB control. Are there any other ACC members with this interest ? I am keen to get in touch with them.

Tony Cassera
140 Tilehurst Rd., Reading, Berkshire.

WB-1/2 & PROGRAMS

I would welcome a WB-2 based on similar techniques to WB-1. One thing I'm not clear about is whether a WB-2 would be a 16 bit machine (i.e. PDP-11 'ish, with a word/byte structure) or an 8 bit, able to address more than 256 bytes. My feeling is for the latter.

With regard to enhancing the WB-1 as it is, I've some sketchy ideas for an interrupt system. It would involve dedicating two more storage bytes, for old program counter and interrupt handler address. An interrupt would set an interrupt latch, which would be serviced after S12. The PC would then be dumped (say to loc 003) & loaded (say from loc 004), the latch reset, and processing continue as normal. Peripherals would have status registers, which would be polled by the IH to ascertain who interrupted. During handling, interrupts would be diasbled, and re-enabled upon return to mainstream processing (reload PC from loc 003). Snags centre around S1, where a wake-up clock pulse would have to be generated, and the restored PC would have to be PC-1. The data display would be corrupted.

Herewith a few WB programs which I hope will be interesting and/or amusing.(one in this issue, others held due to lack of space - ed). The SUB/GMI problem has caused no end of heartache ! Once the MSB is set, the GMI instruction after a SUB can give unexpected results. The C bit setting on subtract seems to be the inverse of the original spec (Vol 3 Iss 3 Aug '75) and, checking this against the truth tables for the 74181 sems to indicate that the original spec was incorrect. Anyway, it consistently sets C when it should clear it and vice versa.

G.D.Hayes

BITOP

Presents storage one bit at a time to I/O port at loc 037. Might be useful to produce software serialising of a program for an external storage medium.

Entry @ 003

Program to be output is loaded above 040.

```
LOC   CODE          SOURCE STATEMENT
003   020 041       L1      MOV P1,A
005   106           L2      SHR A
006   043 035               ADC OUTBYTE
010   060 036               MOV A,SAVE
012   020 035               MOV OUTBYTE,A
014   060 037               MOV A,OUTPUT
016   040 035               CLA OUTBYTE
020   020 036               MOV SAVE,A
022   042 040               DEC CT
024   034 005               GPL L2
026   041 004               INC L1+001
030   050 007 040           MOV  007,CT
033   030 003               GTO L1
035   000           OUTBYTE HLT
036   000           SAVE    HLT
037   000           OUTPUT  HLT
040   007           CT      DEF  007
041   000           P1      HLT
```

ACC 6800 LIBRARY

6800 SOFTWARE POOL

A start has been made in gathering together a collection of short programs for the Motorola 6800 microprocessor and a set of brief instruction sheets on how to use and assemble 6800 microcomputer kits and associated bits and pieces. For those of you who want to use this service send a stamped addressed envelope A4 size to the address below for any particular item and an up to data index. The published index will always be a month or so out of date so for the latest information write or phone. Members who have 6800 experience do please send in items of software however trivial or details of hardware applications even if it is only to light a single LED.

This pool can only be as useful as the members of the ACC want it to be - so let's share our collective knowledge and help those who are just starting to get launched without too many catastrophic traumas !

Certain items cannot be duplicated legally and so are for loan only. These are marked 'L' in the index.

Tim Moore 24 College Rd, Maidenhead, Berks. SL6 6BN
tel; 0628 - 29073

INSTRUCTION SHEET INDEX

L B1 SYSTEMS REFERENCE AND DATA SHEETS (Motorola)
details of most chips used in kit.

L B2 USER'S ASSEMBLY GUIDE FOR MEK6800D1 KIT (Celdis)
component by component description.

L B3 CIRCUIT DIAGRAM FOR MEK6800D1 KIT (Motorola)

L B4 6800 INSTRUCTION SET (Motorola)

L B5 SINGLE STEP FEATURE FOR 6800 (New Electronics, page 24 Jan 13 1976)

L B6 HEX VALUES OF MACHINE CODES (Motorola)

L B7 ENGINEERING NOTE 100 (Motorola)
listing of Mikbug

B8 THE PARTS YOU DO NOT NEED FOR MEK6800D1 (ACC)
the bought kit is not complete but many 'recommended' parts are not required.

B9 USE OF SUBROUTINES WITHIN MIKBUG (ACC)
many goodies are hidden within 512 bytes of the Mikbug ROM.

B10 EXTRA MEMORY WITHIN MIKBUG RAM (ACC)
at least 53 extra bytes are available.

L B11 M6800 MICROCOMPUTER SYSTEM DESIGN DATA (Motorola) 165 pages, recommended.

L B12 UNDERSTANDING MICROPROCESSORS (Motorola)
113 pages, recommended

L B13 FROM THE COMPUTER TO THE MICROPROCESSOR (Motorola) 58 pages, slightly elementary

L B14 ASSEMBLY INSTRUCTIONS FOR MEK6800D1 EVALUATION KIT (Motorola) 16 pages, essential

L B15 Large number of component data sheets - see separate index.

L B16 Data sheets on Polyvalent development system, see separate index

L B17 EXORCISER DATA SHEETS - HARDWARE (Motorola)

L B18 DATA SHEETS ON SOFTWARE (Motorola)

L B19 MICROPROCESSOR CLOCK APPLICATIONS (Motorola)

L B20 INSTRUCTION SET CRIB SHEET (Motorola)

B21A/B PRICE LISTS CRAMER/LOCK

L B22 MPU NEWSLETTER (Motorola)
8 pages, light overall view

L B23 M6800 µPROCESSING (Motorola)
14 pages, even lighter view

L B24A/B MOTEST-1 COMPONENT TESTER (Motorola)
Factory check NO/YES on component IC

L B25 µPROC PROGRAMMING MANUAL (Motorola)
hundreds of pages, very detailed description of instruction set. From Mick Reeve tel 01-446-1045

L B26 APPLICATIONS MANUAL (Motorola)
from Mick Reeve as above. 500-600 pages

L B27 SUMMARY OF AVAILABLE KITS (Motorola)
concise and candid

L B28 MPU APPLICATIONS IN INDUSTRIAL ELECTRONICS (Motorola) describes A/D converter and lift system, a trifle dry

SOFTWARE INDEX

S1 LISTER (THOMSON) (BYTE)
lists a program in machine code, one instruction per line. 68 Hex bytes.

S2 INPUT/OUTPUT EXAMPLE (ACC)
inputs 11 characters, prints 4 spaces & prints 11 char. 43 Hex bytes.

S3 INPUT/OUTPUT EXAMPLE (ACC)
inputs 26 characters and then outputs them,20 Hex bytes

S4 SUBROUTINE FOR TESTING FOR N OR Y INPUT (ACC)
sets carry flag for N input, clears for Y and prints ? for any other input. 16 Hex bytes.

S5 PRINT OUT ALPHABET SLOWLY (ACC)
divides execution speed by 65000 e.g. one letter printed every second (approx). 1C bytes.

S6 1 TO 9 ADDITION (ACC)
integer calculator for numbers 0 to 9, provided sum does not exceed 9. 2B Hex bytes.

S7 ASCII TO BINARY CONVERSION (ACC)
inputs numbers 1 to 127 decimal and loads one byte of store. Prints message for number out of range. 40 Hex bytes.

S8 '2' COMPLEMENT TO ASCII CONVERSION (ACC)
takes care of minus sign and then prints out in decimal. 36 Hex bytes.

S9 ADDS TWO 16 BIT NUMBERS AND TESTS FOR OVERFLOW
14 Hex bytes.

S10 MULTIPLIES TWO 8 BIT NUMBERS (ACC)
3E Hex bytes.

L S11 16 BIT BINARY TO DECIMAL ASCII (Motorola)
PGM5 Motorola Programming Manual page 10 .3D Hex

S12 TO TEST PROGRAM S11 (ACC)
1A Hex bytes

S13 MEMORY CHECK (ACC)
loads and checks all available memory.1CHex bytes.

S14 CLEAN FINISH AND RESTART OF A PROGRAM (ACC)
resetting of program counter not needed to get program going again.

L S15 MULT16 - MULTIPLIES 2 16 BIT '2' COMPLEMENT NUMBERS (Motorola) Applications Manual p 2-15.
4D Hex bytes.

S16 CHARACTER STRING LOADER (ACC)
loads any length character strings in a formatted manner and then lists with starting address of each string. Uses Mikbug routines. 73 Hex bytes.

L S17 RANDOM NUMBER GENERATOR (GRAPPEL - BYTE)
simulates tapped shift register. uses Mikbug routines. 18 Hex bytes.

AMATEUR COMPUTER CLUB NEWSLETTER

Vol 4 Iss 5 December 1976

Editor; Mike Lord
7 Dordells, Basildon, Essex
Basildon (0628) 411125

.... sorry it's late folks ...