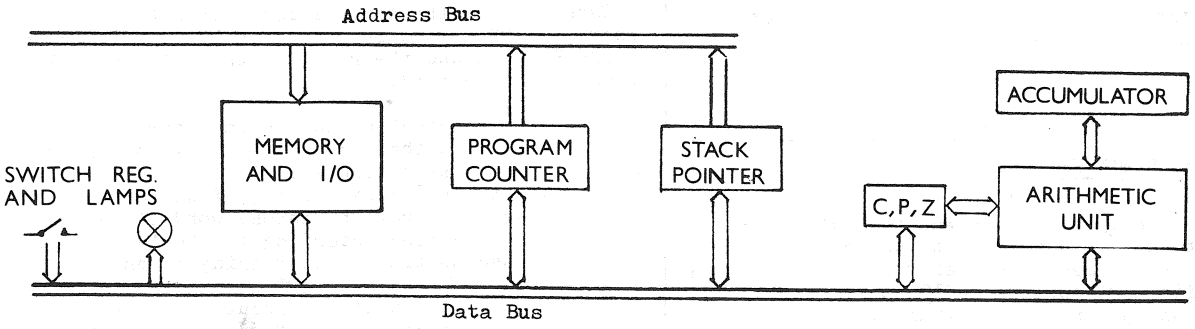
A.C.C NEWSLETTER VOL 3 ISS 3 AUG. 75

* THE WEENY-BITTER * * * *



INTRODUCTION

The WEENY-BITTER is a simple 256. word, 8 bit machine which can be built for around £50, without peripherals, but which is capable of enhancement by the addition of more memory and some extra instructions. In the enhanced form (which can handle up to 64K bytes of memory) it is a fairly powerful mini; the basic machine can be used to play simple computer games, as an educational unit; a controller/sequencer for household devices, model train layouts; a sophisticated code converter, peripheral controller for a larger machine etc.

This first article describes the fundamental machine characteristics and instruction set. Future articles will describe the hardware of the basic machine (WB-1), programming techniques hardware enhancements to upgrade to the WB-2, and possibly various I/O devices.

FUNDAMENTAL MACHINE CHARACTERISTICS

The block diagram above shows the major units of the Weeny-Bitter as it appears to the machine language programmer - the actual hardware configuration is somewhat different as will be explained in a subsequent article.

Memory

This consists of a number of locations (256. for the WB-1, up to 65536. for the enhanced version) each capable of holding one eight bit word. The word may be either data or instructions, the memory does not differentiate between them. Each word in the memory has a unique address - thus the WB-1 has 256. locations numbered 0 to 255. (0 to 377 in octal). The low 4 words of memory are represented in the diagram below;

octal address	word
003	123
002	001
001	001
000	377

In this example, location 003 (binary 00000011) contains the eight bits 01010011 (octal 123), location 000 contains all 'ones' (octal 377), and locations 001 and 002 both contain octal 001.

The contents of specified locations may be modified, read out or replaced by new words.

Certain memory locations are reserved for special uses; these are detailed later.

Program Counter

This is a register which contains the address of the next instruction word in memory. It is automatically incremented by one as each instruction word is read from memory and the contents may be changed to jump to another part of the program.

Stack Pointer

This register is not equipped on the basic WB-1.

It contains the address of the top word of the stack (in memory). When a word is 'pushed' onto the stack the contents of the stack pointer are incremented by one and are then used to define the memory location into which the data word is stored. Data is 'popped' from the stack by reading it from the location defined by the Stack Pointer after which the contents of the Stack Pointer are decremented by one. For example, if the SP contains the address 100, pushing a data word A onto the stack will cause A to be stored at location 101 and the SP contents changed to 101. A subsequent push of data word B will store it in location 102 and change the SP to 102. Popping will then get word B and change the SP back to 101.

The stack is mainly used to store the return address for subroutine and interrupt calls.

Accumulator

This is an eight bit register used to hold one of the operands in certain operations.

Arithmetic Unit

This unit performs the arithmetic/logical operations involved in instructions such as ADD, INC, AND.

C, P,& Z

These are one bit registers used in conjunction with the arithmetic unit. Their functions are;

C holds the carry bit resulting from an addition, subtraction or shift operation.

P holds the inverse of the most significant bit of the result of the latest operation. When data is treated as a two's complement signed binary number this bit is the sign bit (one for negative) therefore P is one for positive, zero for neg.

Z is set to one if the result of the last operation (ignoring the C bit) was all zeroes. If not it is cleared to zero.

Input / Output

There are no special Input/Output instructions, but all I/O ports are made to look like memory

locations and are thus handled with the standard memory reference instructions.The address chosen for an I/O port may be the same as one of the existing memory locations, in this case the memory is inhibited during operations involving that particular address.

Switch Register

Eight switches form a primitive input port and allow the user to enter instructions and data into the machine.

Bus Display

Eight lamps can be used as a primitive output port to allow the user to examine the contents of memory locations and various registers.

OCTAL , DECIMAL & BINARY

One difficulty that appeared when people first started to use binary computers was that of representing the binary data as used by the machine in a way that mere humans can easily use - a string of binary digits e.g. 1001101100010 is not easily remembered. Translation into decimal gives a number (4962 in this case)which can be handled easily by humans but the translation between binary and decimal is tedious and rather error prone. To overcome these difficulties the Octal number system is now used almost universally to represent binary numbers.

Octal is a number system to the base 8, using the digits 0 to 7 inclusive, so you count;

0, 1, 2, 3, 4, 5, 6, 7, 10, 11, . . .
17, 20, 21, . . 77, 100, 101 . .

The big advantage is that each octal digit is exactly equivalent to 3 binary digits;

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

The binary number is converted to octal by dividing it into groups of 3 bits, starting at the right (least significamt) end, and translating each group of 3 bits into an octal digit, for example the binary bumber given above is broken down into

1, 001, 101, 100, 010

which gives octal

1 1 5 4 2

Thus the 8 bit word used in the WB divides into three groups; XX, XXX, XXX and can therefore be represented by a three digit octal number in the range 000 to 377.

To avoid confusion, all decimal numbers in this article are written with a decimal point (eg 255.) to distinguish tham from octal numbers unless the number is obviously decimal i.e. it contains one of the digits 8 or 9 which are not valid in octal.

OCTAL-DECIMAL-BINARY CONVERSION TABLE

Octal	Decimal	Binary	Decimal	Octal	Binary
10	8.	1000	10.	12	1010
20	16.	10000	20.	24	10100
30	24.	11000	30.	36	11110
40	32.	100000	40.	50	101000
50	40.	101000	50.	62	110010
60	48.	110000	60.	74	111100
70	56.	111000	70.	106	1000110
100	64.	1000000	80.	120	1010000
200	120.	10000000	90.	132	1011010
300	192.	11000000	100.	144	1100100
400	256.	100000000	200.	310	11001000

INSTRUCTION SET

There are four basic classes of machine language instruction;

- Two operand e.g. A plus B
- One operand e.g. Increment A
- Program control e.g. Go to
- Miscellaneous e.g. Halt

Although the instructions are basically binary (or octal) words, mnemonic equivalents are given for ease of use.

Two Operand Instructions

These are of the general form OP P Q where the OPeration is performed between two data words P and Q and the result then replaces Q, P remaining unchanged.

P may be;

- the content of the Accumulator (A)
- data in the word immediately following the instruction word (D)
- data in memory location X where X is derived from the second word of the instruction according to the 'mode' bits (see section on addressing modes)

Q may be;

- the content of the Accumulator (A)
- data in memory location X where X is derived from the second or third word of the instruction.

The allowable forms are;

mnemonic	octal	
OP #D A	11R DDD	Acc := Acc OP data DDD
OP X A	n2R XXX	Acc := Acc OP mem
OP #D X	n5R DDD XXX	mem := mem OP data DDD
OP A X	n6R XXX	mem := mem OP Acc

where n defines the addressing mode and R the operation. For example,using the SUBtract operation with addressing mode 0 (direct ,first 256 words) we can have;

SUB #D A	112 222	subtracts octal 222 from the Acc.
SUB X A	022 222	subtracts the contents of memory location 222 from the Acc.
SUB #D X	052 222 333	Subtracts octal 222 from memory location 333
SUB A X	062 222	Subtracts the contents of the Acc from memory location 222

Operations

MOV T Q (**0)	Q is replaced by T, T remains unchanged. C becomes zero, P & Z set to one or zero depending on T.
ADD T Q (**1)	T is added to Q, T remaining unchanged. Result put in Q. C, P & Z set to one or zero depending on the result.
SUB T Q (**2)	T is subtracted from Q, result in Q, T remains unchanged. C, P & Z set to one or zero depending upon result.
AND T Q (**3)	Q is replaced by the logical AND of T and Q. T remains unchanged. C is set to zero, P & Z set to one or zero depending on result.
IOR T Q (**4)	Q is replaced by the logical inclusive OR of Q & T. T remains unchanged. C is set to zero. P & Z set to one or zero depending upon result.
XOR T Q (**5)	Q is replaced by the logical exclusive OR of Q & T. T remains unchanged. C is set to zero. P & Z set to one or zero depending upon result.
BIT T Q (**6)	As XOR T Q except that the result is not put in Q, which remains unchanged.
CMP T Q (**7)	As SUB T Q except that the result is not put in Q, which remains unchanged.

One Operand Instructions

These are of the general form OP Q where the OPeration is performed on the single data word Q.

Q may be;

- the content of the Accumulator (A)
- data in memory location X where X is derived from the second word of the instruction.

The allowable forms are;

mnemonic	octal	
OP A	10R	Acc := Acc OP
OP X	n4R XXX	mem := mem OP

where n defines the addressing mode and R the operation. For example, using the INCrement operation with addressing mode 0 (direct, first 256 words) we can have;

INC A	101	Increments the Accumulator
INC X	041 222	Increments the contents of memory location 222

Operations

CLA Q (**0)	Clear Q to zero. C is set to zero, P and Z are set to one.
INC Q (**1)	Increment Q by one. C, P & Z are set to one or zero depending on result.
DEC Q (**2)	Decrement Q by one. C, P & Z are set to one or zero depending on result.
ADC Q (**3)	Add C bit to Q, C, P & Z are set to one or zero depending upon result.
TST Q (**4)	Q is examined and P & Z set accordingly. Q remains unchanged. C set to zero
INV Q (**5)	Invert all bits in Q. C set to zero, P & Z set to one or zero depending on result.
SHR Q (**6)	Shift Q right one place, most significant bit (B7) becomes 0, least significant bit (B0) shifted into C. P set to one. Z set to one or zero depending on result.
SHL Q (**7)	Shift Q left one place. Most significant bit (B7) shifted into C, B0 becomes 0. P & Z set to one or zero according to result.

Program Control Instructions

The GO TO and JMS instructions control the flow of the program, either unconditionally or according to the state of the C, P & Z bits. They are both 2 word instructions of the form

OP X

where X (with the memory address mode bits) defines the memory location to which the program is to go. e.g. using the unconditional GO TO and mode 0 (direct, first 256 words) we can have;

GTO X	030 222	which causes the program to go to location 222 by loading this number into the Program Counter.

The JMS instruction (not implemented in the basic machine) puts the program counter contents, which are the address of the instruction after the JMS instruction, onto the stack before loading the Program Counter with the new value. The GTO instruction loses the old value of the Program Counter.

Miscellaneous Instructions

These are all single word instructions;

HLT (000) stops program execution

NOP (001) is a 'filler', the machine accepts it but does nothing before passing to the next instruction.

CLC and SEC (002 & 003) are used to set and clear the C bit

ION and IOF (004 & 005) are not implemented on the basic machine

RTS and RTI (006 & 007) , not implemented in the basic machine, return program control from a subroutine or interrupt by loading the program counter with the address previously stored on the stack by the JMS instruction or by the interrupt.

RESERVED MEMORY LOCATIONS

Certain memory locations are reserved for special functions ;

address	function
000	Accumulator
001	Switch Register
002	Low 8 bits of Program Counter
+003	High" " " " "
+004	Low 8 bits of Stack Pointer
+005	High " " " " "

+not used in the basic machine

Locations 007 to 037 should also be reserved, for use by I/O ports

MEMORY ADDRESSING MODES

The eight address bits (X) assosciated with memory reference instructions of the form;

OP X A
OP #D X
OP A X
GTO/JMS X

are interpreted according to the 'mode' bits of the instruction (B7 & B6) as follows;

mode	B7	B6	
0	0	0	Direct. The number X is the required address (one of the first 256 words)
1	0	1	Indirect. The required address is located in memory locations X and X+1 (low 8 bits in X, high 8 bits in X+1)
2	1	0	Relative. The required address is obtained by adding (X-200) to the current contents of the Program Counter.
3	1	1	Indirect Relative.

Only mode 0 is implemented in the basic WB-1.

Note that the indirect modes (1&3) generate a 2 word (16 bit) address, allowing the enhanced machine to use up to 64K bytes of memory.

BINARY ARITHMETIC

Addition

This is done according to the rules;

0 + 0 = 0
1 + 0 = 1
1 + 1 = 10 (0, carry 1)

Thus;

```
  00 000 101   (5)
+ 00 000 110   (6)
  00 001 011   (11.)
```

Any carry resulting form the most significant bit position is put into the C register ;

11 111 110 + 00 000 001 = 11 111 111 ,C=0
11 111 110 + 00 000 010 = 00 000 000 ,C=1
11 111 110 + 00 000 011 = 00 000 001 ,C=1

Subtraction

Is done according to the rules;

0 - 0 = 0
1 - 0 = 1
1 - 1 = 0
0 - 1 = 1 ,borrow 1

Any 'borrow' resulting from the most significant bit position is put into the C register;

```
00 000 100 - 00 000 001 = 00 000 011  ,C=0
00 000 100 - 00 000 100 = 00 000 000  ,C=0
00 000 100 - 00 000 101 = 11 111 111  ,C=1
```

Note that the last example (4 - 5) gives a negative answer (-1) which is represented in binary form as 11 111 111 . This is known as the 'Twos Complement' form which is the way the WB represents negative numbers.

The Twos Complement of a number is defined as that number which when added to the original number will result in a sum of zero (ignoring the carry out to the C reg). Thus the positive number 00 000 001 has a twos complement equal to 11 111 111 as shown by the following addition

```
      00 000 001
    + 11 111 111
  1   00 000 000
```

The easiest way to generate a twos complement is to first invert each bit then add one to the result

```
00 000 010  number
11 111 101  inverted
11 111 110  twos complement
```

In Twos Complement form all negative numbers have the most significant bit (B7) equal to one, while all positive numbers have it equal to zero. The range of numbers which can be expressed by a eight bit word is;

```
11 111 111  (-1)
11 111 110  (-2)
.........
10 000 000  (-128.)
01 111 111  (+127.)
.........
00 000 001  (+1)
00 000 000  ( 0)
```

BINARY LOGIC

AND
is performed according to the rules;

```
0 AND 0 = 0
0 AND 1 = 0
1 AND 0 = 0
1 AND 1 = 1
```

thus;

```
      11 000 111
AND   10 100 011
 =    10 000 011
```

Inclusive OR
the IOR function is performed as;

```
0 IOR 0 = 0
0 IOR 1 = 1
1 IOR 0 = 1
1 IOR 1 = 1
```

thus;

```
      11 000 111
IOR   10 100 011
 =    11 100 111
```

Exclusive OR
the XOR function is performed as;

```
0 XOR 0 = 0
0 XOR 1 = 1
1 XOR 0 = 1
1 XOR 1 = 0
```

thus;

```
      11 000 111
XOR   10 100 011
 =    01 100 100
```

Complete list of WEENY-BITTER instructions

Mnemonic symbol	Octal code	Operation
HLT	000	Halt, stops program
NOP	001	No Operation
CLC	002	Clear C bit to zero
SEC	003	Set C bit to one
+ION	004	Interrupt enable on
+IOF	005	Interrupt enable off
+RTS	006	Return from subroutine
+RTI	007	Return from interrupt
CLA A	100	Clear Acc
INC A	101	Increment Acc
DEC A	102	Decrement Acc
ADC A	103	Add C bit to Acc
TST A	104	Test Acc
INV A	105	Invert all bits of Acc
SHR A	106	Shift Acc right one place
SHL A	107	Shift Acc left one place
MOV #D A	110 DDD	Move data D to Acc
ADD #D A	111 DDD	Add data D to Acc
SUB #D A	112 DDD	Subtract data D from Acc
AND #D A	113 DDD	And data D with Acc
IOR #D A	114 DDD	Inclusive OR D with Acc
XOR #D A	115 DDD	Exclusive OR D with Acc
BIT #D A	116 DDD	Bit Test data D with Acc
CMP #D A	117 DDD	Compare data D with Acc
MOV X A	n20 XXX	Move mem X to Acc
ADD X A	n21 XXX	Add mem X to Acc
SUB X A	n22 XXX	Subtract mem X from Acc
AND X A	n23 XXX	And mem X with Acc
IOR X A	n24 XXX	Inclusive OR mem X with Acc
XOR X A	n25 XXX	Exclusive OR mem X with Acc
BIT X A	n26 XXX	Bit Test mem X with Acc
CMP X A	n27 XXX	Compare mem X with Acc
GTO X	n30 XXX	Go to X
GIZ X	n32 XXX	Go to X if zero (Z=1)
GNZ X	n33 XXX	" " " " not zero (Z=0)
GPL X	n34 XXX	" " " " positive (P=1)
GMI X	n35 XXX	" " " " negative (P=0)
GCS X	n36 XXX	" " " " C bit set (C=1)
GCC X	n37 XXX	" " " " C bit clear (C=0)
CLA X	n40 XXX	Clear mem X
INC X	n41 XXX	Increment mem X
DEC X	n42 XXX	Decrement mem X
ADC X	n43 XXX	Add C bit to mem X
TST X	n44 XXX	Test mem X
INV X	n45 XXX	Invert all bits of mem X
SHR X	n46 XXX	Shift mem X right one place
SHL X	n47 XXX	Shift mem X left one place
MOV #D X	n50 DDD XXX	Move data D to mem X
ADD #D X	n51 DDD XXX	Add data D to mem X
SUB #D X	n52 DDD XXX	Subtract data D from mem X
AND #D X	n53 DDD XXX	And data D with mem X
IOR #D X	n54 DDD XXX	Inclusive OR D with mem X
XOR #D X	n55 DDD XXX	Exclusive OR D with mem X
BIT #D X	n56 DDD XXX	Bit Test data D with mem X
CMP #D X	n57 DDD XXX	Compare data D with mem X
MOV A X	n60 XXX	Move Acc to mem X
ADD A X	n61 XXX	Add Acc to mem X
SUB A X	n62 XXX	Subtract Acc from mem X
AND A X	n63 XXX	And Acc with mem X
IOR A X	n64 XXX	Inclusive OR Acc with mem X
XOR A X	n65 XXX	Exclusive OR Acc with mem X
BIT A X	n66 XXX	Bit Test Acc with mem X
CMP A X	n67 XXX	Compare Acc with mem X
+JMS X	n70 XXX	Jump to Subroutine X
+JIZ X	n72 XXX	JMS if zero (Z=1)
+JNZ X	n73 XXX	" " not zero (Z=0)
+JPL X	n74 XXX	" " positive (P=1)
+JMI X	n75 XXX	" " negative (P=0)
+JCS X	n76 XXX	" " C bit set (C=1)
+JCC X	n77 XXX	" " C bit clear (C=0)

Memory Reference Instruction Addressing Modes;

	n	mnemonic
Direct (first 256 words)	0	X
+Indirect (first 256 words)	1	@X
+Direct relative	2	+X
+Indirect relative	3	@+X

+ not implemented in the basic machine

LETTERS

TRIP

I was recently on a three day course at Salford University (An Introduction to Electrical Eng.) Part of the course involved a 'project' and I was lucky to be assigned to a project involving computing (analogue).

The project involved finding a suitable model to simulate a shunt wound d.c. motor.Our tutor helped us to find an equation (from first principles) which related input voltage to speed.

We connected a motor up to a 'scope and observed that on switch-on the speed would slowly rise, overshoot, then settle down; on lower speeds the rise was slower and there was less overshoot. We therefore knew how our model ought to behave.

We were introduced to oue analogue computer, a £1500 Vidac (Vidac 273 I think it was) and I will give brief details in case you are not familiar with this machine;

The computing elements are fairly standard, integrators, multipliers and inverters with the addition of some logic elements; 'flip-flops', gates and, more important for an analogue machine, comparators (giving logic output which can be gated or stored in order to further control the system). There were 14 10 turn pots which could be used as dividers or for presetting voltages.

Output is fed to a 'scope or x-y plotter (with ramp for the x direction if required) or to a built-in 3½ digit DVM (which can also monitor pot settings etc.).

Also available were several push-buttons; Pot-set (self explanatory), Reset (resets the system), Initial Conditions (allows the initial conditions to be set up), Hold, Compute and Repeated Operation. The last allows continuous monitoring of the system whilst making adjustments (equivalent to on-line work with a digital !) and the sequence of operations is Compute, Reset, Initial Conditions etc. The speed can be set by an internal timer which has a 5/10/20 mSec/sec range, and which selects integration times (and x timebase if required).

With Compute the system computes continuously and needs to be reset manually if another operation is required.

One good feature is that nearly every unit (integrators etc) has an overload warning light which comes on at about 1.3 units (on the DVM). In normal use the computer should only be operated between 0 and 1 units. (the DVM is calibrated to arbitrary 'machine' units as opposed to a particular standard voltage).This overload warning gives indication of which part of the 'circuit' has failed and in complex equations is very useful since overload of one part may not necessarily show at the output (although it may well have a detrimental effect) and troubleshooting may prove difficult.

We set up our equation and used a storage 'scope to plot our results. Unfortunately there were three parameters which we didn't know (relating to magnetic field strangth in the motor) and we found it was taking a long time to optimise these by 'guesswork'. Unfortunately also, the parameters were interlinked which meant if one was adjusted the other two had to be adjusted also. Finally we did, however, manage to get a reasonable approximation but we couldn't get any decent measurements from the system (we had hoped to find % overshoot etc).

In the 'Simulation' department there were quite a lot of analogue and digital machines. Some of the analogue machines were early valve types (huge chassis and PSU's) but one or two were more modern!

The digital machines were PDP8's (F I think) and "Lab 8" which I presume are similar to PDP8's. Some of the machines were linked to analogue machines for hybrid work. Facilities included a teletype, VDU's, disc and tape readers.

I managed to see (and play) a 'Space War' game, in which two players have to control rockets and 'shoot' the other's rocket. The control system was specially built (no details since the unit was 'second-hand' and none were supplied) and featured left,right rotate, forward thrust and 'fire'. Output was on a standard VDU. I believe the program can be run on a PDP8 & VDU without the control (yoystick board) but I think data has to be entered via the switch register which might prove rather difficult.

P.D.Maddison

DISPLAY

TELEWRITER*

If anyone else is interested in building it I would be interested to know in case it is possible to get components cheaper for more than one off.

About the timing, this will have to be altered to suite UK TV standards won't it ? The UHF modulator used in the PW Teletennis article could be used.

I'm also working on a unit for storing data on a cassette recorder. If anyone else is interested in these topics I should be grateful to hear from them.

I have an idea for using a calculator to work out operations and then produce the answers on a screen, not too difficult but very useful I forsee. The code for '+' could be tied into a calculator as could '-' and "=" etc and the output decoded from 7 segment to ASCII. The way I envisage doing this is to convert the 7 segment calculator display to BCD, stored in a shift register maybe, and then use codes 48 to 57 to enter digit data. An off the cuff way of testing all codes in the ROM could be to connect a binary counter + 64 to the inputs of the ASCII data entry points, the complete option set would then be seen. Also note that the BCD code for numbers preceeded by two 'ones' gives the ASCII code - an easy way to get number devices to interface with the system.

Mike Hewitt
25 Spring Lane,Sprotborough,Doncaster,Yorks

* TV TYPEWRITER article, see V3, 11, p4 ed.

SALE!

Hundreds of bits for sale; 30cps 8 track punch, 110 cps reader, transformers, veroboard, IC etc.
J Florentin
44 Burleigh Rd., Hemel Hempstead, Herts HP2 4PP

USA

On March 8, I got a PDP-11 in a trade for some other equipment, I also got a teletype. It was not in working order and I've spent all my spare time up to now bringing it up.It should be running by the end of the month.That would be great!

I also got a D-112 card set. The D-112 is a copy of the PDP-8 made by Digital Computer Controlls. I hope to eventually bring it up alongside the PDP-11 for a dual processor system, but of course that's a dream and at least a year off.

I'd like to correspond with some of your members who'd be willing to just gab about computers. I'd be interested in learning a bit more about the amateur computing scene than you cover in your newsletter. I come across a lot of computer equipment here in the States and have increased my correspondence list to include 42 people around the country. I like to write letters and would be happy to further US/British relations any way I can.

Gary Coleman
PO Box 7089 Arlington , Va , 22207 USA

STRUCTURE

STRUCTURED BASIC TRANSLATOR

I have used the structured programming technique (though unbeknown for most of the time) in the design of a translator which enabled a BASIC subset to be used on a 4K 16 bit minicomputer.

The hierarchy of main program and subroutine levels I denoted as a Format Analyser, Controllers, Modules and Subroutines.

The Format Analyser determines the user's input to be a statement, a command, a deletion, or a load of rubbish, viz;

```
120 LET A = B + C
RUN
100
*!AZ99.,!?
```

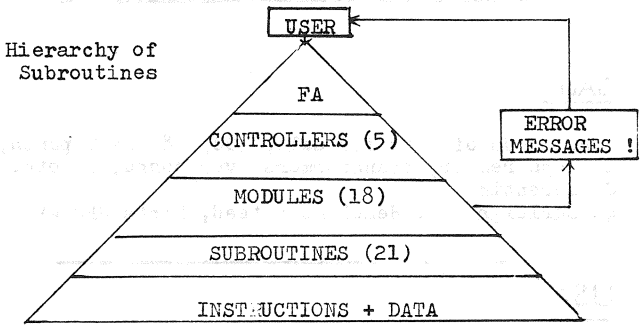
The FA brings in one of five controllers which interpret the user's commands to RUN or LIST the program, to say good-BYE, to delete a statement or to store a statement (includes syntax analysis and encoding).

There were six statements in the BASIC subset; GOTO,IF,INPUT,LET,PRINT and END. Each of the 18 modules handles one type of statement in one of three modes - RUN, LIST and storage of a statement. In the event of user errors,a branch is made out of the module to give an error message. Control then returns to the user.

The Subroutines do the actual work of testing and manipulating data.

A call and reply technique was used between Module and Subroutine. The Module places data in the accumulator, then calls the Subroutine. Now the given data may be out-of-range, the wrong type or OK. The Subroutine informs the Module of either state by 'replying' with a '1' or a '0' in the most significant bit of the accumulator.

The advantage of the structured programming technique here is that Controllers, Modules and Subroutines can be modified therebye altering the power of this BASIC subset e.g. the types of statement can be increased; each type can be altered in versatility eg PRINT could print variables and/or commentary, LET could perform simple addition and subtraction and/or more complex arithmetic functions.



S.L.Thompson

RAM CLUB

L.Elmer will be buying some 4K bit RAM, probably Motorola MCM 6605 L, and would be interested to hear from anyone interested in ordering with him to raise the quantities and so lower the price. Cost is about £11 for one-off, dropping to around £7 if you buy more than 100. Ring him (weekends only) on 078 887 324 or write, but quickly, to 6 Atterbury Close, West Haddon, Northampton

DATA

Intel (of 8080 fame) publish a huge catalogue of their memories/microcomputer components, at 15p each seems like a bargain. They are at Broadfield House, Between Towns Road,Cavendish, Oxford tel 771431. Their official distributors publish a list with prices, they are Rapid Recall and are at 9 Betterton St.,London WC2H 9BS, tel 01-379-6741. In fact I got mine straight from Intel.

General Instrument Microelectronics publish a MOS Condensed Catalogue: ROM's, RAM's, PROM's, calculator chips, counter/drivers etc. They are at 57-61 Mortimer St., London W1N 7TD tel 01-636-2022.

Sintel (53 Ashton St,Oxford) do a lot of MOS/CMOS chips-they publish a list/catalogue and will answer queries etc by phone (Oxford 43203).They seem very helpful to me...

Be warned that the cheap I/C supplier called DeMa (who advertise in Wireless World)wholive in the USA do not include 15% duty and 8% VAT in their prices!

A shortform catalogue of CMOS made by RCA (with connection data etc.) can be obtained from Celdis Ltd.,37/39 Loverock Rd,Reading RG3 1ED, tel Reading 582211.

Texas Instruments (Manton Lane, Bedford) publish an application note for the TMS0100NC series of calculator chips and a MOS/LSI specifications short form for design engineers.

P.S. Anyone around here who would be willing to answer my queries/help/drop in & chat ?
Jeremy Wyatt
The Rooks Nest,Charlbury,Oxon

SALE!

I have for sale 3 blocks of 10 planes each of 48 x 52 bit core originally from 1301.

A cheap way of buying TTL is at radio rallies where you can buy a board of many devices, the individual price being a few pence each.

I am interested in connecting a small machine with teleprinters via amateur radio. Any one else?
P Staton
847 Lincoln Rd., Peterborough PE1 3HG

BOOKLIST

MICROPROGRAMMING
G.Boulaye £7.50 Macmillan

COMPUTER AIDED DESIGN OF ELECTRONIC CIRCUITS
E.Wolfendale Illife

MINICOMPUTER SYSTEMS : ORGANISATION AND PROGRAMMING THE PDP-11
Prentice-Hall series in automatic computation.
1975 £9.80

CYBERNETIC SERENDIPITY: THE COMPUTER & THE ARTS
Ed. J Reichardt 1969 Frederick A Praeger

MacDONALD COMPUTER MONOGRAPHS:
9 USE OF FILES D.Judd
18 PROGRAM DEBUGGING A.Brown & W.Sampson

DIGITAL COMPUTER STRUCTURE & DESIGN
R.Townsend 1975 £4.20 Newnes-Butterworths

MODULAR PROGRAMMING
J.Maynard 1972 £1.50 Newnes-Butterworths

```
      INTEGER A,B
      LOGICAL LEAP
      DIMENSION A(7,6,12),B(504),MONTH(12),NAME(12),DAY(21),
     *          IDAY(28),LEAP(28)
      EQUIVALENCE (A(1,1,1),B(1))
      DATA I,J,K,L,M,N,J1,J2,K1,K2/0,0,0,1,0,0,0,0,0,0/
      DATA IDAY,LEAP/3,4,5,7,1,2,3,5<6,7,1,3,4,5,
     *               6,1,2,3,4,6,7,1,2,4,5,6,7,2,
     *               F,F,T,F,F,F,T,F,F,F,T,F,F,F,
     *               T,F,F,F,T,F,F,F,T,F,F,F,T,F/
      DATA MONTH/31,28,31,30,31,30,31,31,30,31,30,31/
      DATA NAME/'JAN','FEB','MAR','APR','MAY','JUN',
     *          'JLY','AUG','SEP','OCT','NOV','DEC'/
      DATA DAY/'SU','MO','TU','WE','TH','FR','SA',
     *         'SU','MO','TU','WE','TH','FR','SA',
     *         'SU','MO','TU','WE','TH','FR','SA'/
 6000 FORMAT('1')
 6010 FORMAT(//)
 6020 FORMAT(34X,3(A4,29X),/,1X)
 6030 FORMAT(18X,3(5X,7(I2,2X)),/)
 6040 FORMAT(18X,3(5X,7(A2,2X)),/)
 6050 FORMAT(/,97X,'CALENDAR NUMBER ',I2)
      CALL YEARS
      DO 5 II=1,28
      DO 10 I=1,504
      B(I)=100
   10 CONTINUE
      K2=IDAY(II)
      PRINT 6000
      PRINT 6050,L
      PRINT 6010
      DO 40 K=1,12
      NDAYS=MONTH(K)
      IF((K.EQ.2).AND.LEAP(II))NDAYS=NDAYS+1
      DO 20 K1=1,NDAYS
      B(K2)=K1
      K2=K2+1
   20 CONTINUE
   30 K2=K2+7
      IF(K2.LE.42*K)GO TO 30
   40 CONTINUE
      DO 60 J1=1,10,3
      J2=J1+2
      IF(J1.NE.7)GO TO 41
      PRINT 6000
      PRINT 6010
   41 PRINT 6020,(NAME(J),J=J1,J2)
      PRINT 6040,(DAY(M),M=1,21)
      DO 50 J=1,6
      PRINT 6030,((A(I,J,K),I=1,7),K=J1,J2)
   50 CONTINUE
      PRINT 6010
   60 CONTINUE
      L=L+1
    5 CONTINUE
      CALL EXIT
      END
      SUBROUTINE YEARS
      INTEGER*4 YEAR,N
 1000 FORMAT(/,11X,'FOR ',I4,' USE CALENDAR NUMBER ',I2)
 2000 FORMAT('+',47X,'FOR ',I4,' USE CALENDAR NUMBER ',I2)
 3000 FORMAT('+',83X,'FOR ',I4,' USE CALENDAR NUMBER ',I2)
      DATA YEAR,N,J/1755,10,0/
      DO 10 I=1,1245
      J=J+1
      IF(YEAR.EQ.1800)N=N-17
      IF(YEAR.EQ.1801)N=N-3
      IF(YEAR.EQ.1900.OR.YEAR.EQ.2300.OR.YEAR.EQ.2700)N=N+5
      IF(YEAR.EQ.1901.OR.YEAR.EQ.2301.OR.YEAR.EQ.2701)N=N+11
      IF(YEAR.EQ.2100.OR.YEAR.EQ.2101
     *.OR.YEAR.EQ.2200.OR.YEAR.EQ.2201
     *.OR.YEAR.EQ.2500.OR.YEAR.EQ.2501
     *.OR.YEAR.EQ.2600.OR.YEAR.EQ.2601
     *.OR.YEAR.EQ.2900.OR.YEAR.EQ.2901)N=N-6
      IF(J.EQ.1)PRINT 1000,YEAR,N
      IF(J.EQ.2)PRINT 2000,YEAR,N
      IF(J.EQ.3)PRINT 3000,YEAR,N
      IF(J.EQ.3)J=0
      YEAR=YEAR+1
      N=N+1
      IF(N.EQ.29)N=1
   10 CONTINUE
      RETURN
      END
```

THIS PROGRAM PRINTS OUT A SET OF CALENDARS. AND A TABLE SHOWING WHICH ONE TO USE FOR THE YEARS

1755 – 2999

P. RUTHERFORD

ED'S BIT

So the Weeny-Bitter seems to have got off to a good start, next issue we should be able to start featuring the hardware design. Thanks to all those who are contributing to the design of both hard and soft ware aspects.

As you can see from the adjacent column we now have an ACC library, thanks to Mr. Doherty-Bullock. He has asked me to print a request for further contributions to the library - books, equipment manuals, programs etc. So lay it on him folks.

While I'm still in an appealing mood, could I prevail upon someone to write us an article on Data Base techniques and applications ? Seems to be the latest 'thing', wish I knew something about it.

Membership has now reached 246, compared with last year's final figure of 281, and without any extra special recruiting drive we look like ending this year with around 260.At this figure and taking into account the £52 surplus from last year we should just about break even if inflation does not hot up again. Nevertheless, to be on the safe side we should try to recruit more. Jon Aslett has been contacting the electronic & computer magazines to try for some more free publicity but without success so far.

Inputs for the October issue by Oct 11 please.

m.lord

WHAT DO YOU THINK ?

A 'mind reading' system being developed at the Stanford Research Institute, California, monitors ECG signals and uses pattern recognition techniques to relate them to specific words. At present they are achieving around 50% accuracy with a 7 to 15 word vocabulary, not spectacular but its a start.

THE SCHOOL SYSTEM

A 64K English Electric System 4/30 with six tape drives, line printer & paper tape reader is now working at Imberhorne School, East Grinstead having been donated by the Standard and Chartered Bank last year. The school is now seeking a teletype with punch and, after the hot summer, some form of air conditioning.

CODE CONVERSION

The following table gives the normal commercial code equivalents for the short alphanumeric codes sometimes found on 'surplus' IC's.

P.STATON

A	7400	LN9	LM311N/8	M26	74175
B	7420	M2	7481	M27	74S158
C	7451	M4	74180	M28	74S194
D	7474	M6	74151	M29	74283
E	7486	M7	74157	N	74121
F	7430	M8	7485	P	7428
G	7440	M9	74191	Q	75451
H	7402	M10	7495	S	7413
I	74112	M11	74155	U	7410
J	7401	M12	7489	V	7404
K	74122	M14	74118	W	7406
L1	74181	M15	74200	WZ	74S05
LN2	75108	M16	74148	Y2	74S11
LN4	LN340	M17	7496	Y3	74S64
LN5	72710	M19	74164	Y4	74S133
LN6	72709	M20	74S138	Y6	7438
LN7	72741	M21	74S175	Y7	7414
LN8	74S124	M25	74S153		

Generally suffix X means high speed
" Z " ultra high speed

e.g. A = 7400
AX = 74H00
AZ = 74S00

16 BIT MICROPROCESSOR

A 16 bit microprocessor chip, designed by General Instruments for Honywell, is now being made available for general sale in the UK.
Called the 1600 CPU, it costs £121 (one off).

LIBRARY LISTING

DEC 'Introduction to programming' (PDP-8)
DEC 'Small computer handbook ' "
SHOPLAN Dataskill production control system, leaflets
General Dataskill software sales literature, leaflets including details of operating systems etc.
COBRA leaflet
IBM System 7 functional characteristics
Extracts from Elektor, german electronics magazine describing hardware & instruction set considerations.
Index catalogue of Teletype KSR,ASR & RO 33 page printer parts.
Teletype ASR,KSR33 service manual, Vol I
Literature about Ferranti's Argus machines.
Ferranti press releases about past projects including traffic control & computer controlled manufacture.
Ferranti in offshore oil.
Ferranti - CEGB control centre.
" - CP6 controllers.
" - Packet switching .
" - Computers in an information network.
" - Details of PT7 programmable terminal.
" - Description of support services.
" - Dartford # 6 paper machine computer control.
Univac 1106/1108 Algol reference manual.
Univac 9400 software conventions reference manual.
Univac OS/4 disc mapping program manual.
Univac Uniscope 100 display terminal reference.
Xerox Fortran IV-H language reference manual.
" Fortran IV-H operations reference manual.
" BASIC language reference manual.
" APL reference manual.
" CIRC-AC (circuit analysis) manual.
" Extended Fortran IV reference manual.
" Volume initialisation reference manual.
" CP5 (Control Program 5) reference manual.
" Universal time sharing OS
" Sort/Merge reference manual.
TSL Telcomp 2 user manual.
" Superstat user manual.
Olivetti TE 300 manual.
Xerox Peripheral Switching Equipment.
" CIRC-TR
" CIRC-DC
" Text formatting prog user guide.

In addition, Control Data have put their library at my disposal.

LIBRARIAN : F. DOHERTY-BULLOCK
CRC Information Systems Ltd.,
83 Clerkenwell Rd
London EC1R 5HP

MEMBERSHIP of the ACC for the year 1 April 1975 to 31 March 1976 plus subscription to Vol 3 of the Newsletter costs £1 (50p for UK members who are aged 16 or under on 1 April 1975).

Vol 3 of the newsletter will consist of six issues, to be published at two month intervals.

AMATEUR COMPUTER CLUB NEWSLETTER
Vol 3 Iss 3 August '75

m.lord
7 Dordells, Basildon, Essex
tel; 0268 411125 home
0268 3040 x 117 work