JUNE/JULY 1979 NO.3

Welcome to the third issue of C&A.

GENERAL

I am indebted for all of you who added a contribution to the first newsletters, I shall try to put as many of your letters as possible into other circulations.

Incidentally, there are now over 40 members of the MUC (MK Fourteen Users Club) two of which are living abroad. One is from Denmark, he actually 'phoned me! The other is from France, writing after seeing a letter in Personal Computer World.

I am grateful to Steve Birch for photocopying one newsletter and starting its distribution.

This issue includes a list of all members except the following late arrivals:

Mr. Brent Heard, Gwent

Mr. Graham Knott, Cambridge, Cambs,

Mr. B. Johnson, Newcastle on Tyne NE4 6BE.

Please note that Michael Gundy can be reached faster via Whetstone, Barnet.

Now you have got no excuse for not meeting the folk in your area. (Not that I can talk - my recent attempt to meet Mike Gundy ended up with me just missing him by a few minutes)

I managed to get another letter published in PCW recently, and also my Maze Program in Computing Today. (They haven't paid me yet).

I am hoping to contribute to a SC/MP column in Practical Computing, and possibly some program for PE's VDU via Mr. Berk who designed the system.

Coming back to the subject of meetings, would anyone like to arrange some meeting place and time in London? (I will appoint them M.C. (meeting Organiser)) I am too busy at the moment getting newsletters out and answering letters.

1979 COMPUTER SHOW

From 5-7th July, there is a Micro-Computer Exhibition being held. I certainly intend to go, probably on the 6th (it will be crowded on the Saturday) to see the exhibition and mess about on the PETs (and Mk14s if a For more information consult Practical Computing.

Cassette Interface

Originally (many months ago) I built a Cassette Interface module and found to my dissappointment that it didn't work.
So I (hopefully) sent it off to S. of C. and forgot all about it.

Since then I purchased another from them (via Greenbank) and got it working.

They have now returned the old one (free of charge!) which is now lying around doing nothing.

Would anyone who would like this C.I. (I haven't tested it) please send me a stamped addressed jiffy bag(!!) and 2 books of 9p stamps so that I can regain some of my costs.

If you 'phone me up first you can determine whether someone else has nabbed it before you!

4K NIBL BASIC

In the most recent issue of Elektor there was a listing of a Basic Interpreter for SC/MP which might help anyone who is trying to fit a mini Basic into 512 bytes.

This magazine has quite a lot of SC/MP based products, they don't seem to like other micros.

PE VDU

Since the newsletter no.2, I have managed to connect up a Practical Electronics VDU with 95 per cent success.

The VDU provides 16 lines of 64 characters mapped onto 1K of memory. These characters can be either white on black background or vice-versa. All upper case ASCII characters are provided, 64 in all, but I have added a lower case character generator which provides some peculiar graphics too!

The VDU is much easier to use than the MK14 display as you do not have to multiplex the output and you are not limited to 7 segments.

Most of the software I am writing, or have written should easily be adapted for S. of C's own VDU as it is also based on the memory mapped principle. (Paul from Kenton tells me they are going to charge £30)

However I have the problem that writing to a location in the VDU's ram occasionally causes a corruption in another location.

I have rigged it so that I can either have reliable memory or VDU. (The or is exclusive-or i.e. E4).

If anyone thinks thay have the solution, please let me know.

So far Mark Wyllie is the only one with a solution to the insert delete routines. His solutions were correct but only work for 256 bytes at a time, basically the programs are the same. I think I gave the solution in issue 2 but here they are anyway:

INSERT P2 points to start of where insertion is needed, extension reg. contains byte. Enter these at OFFB,C for address of insertion and OFFE for the byte. program(relocatable): C6 01 01 CA FF E4 88 9C F7 3F. The program must be terminated by 88.

In the same way: DELETE routine: C6 01 CA FE E4 88 9C F8 3F, again, P2 points to start, and the program finishes with 88.

PROBLEM FOR THE MONTH

The problem is a bit more tricky this time.

Find out how many ways there are to load zero into the accumulator using up to 2 bytes and making no assumptions about the current status of other registers.

I will start you off with Load Immediate 00! I can think of at least 4 others.

BAD MEMORY?

For those without the revised monitors, kindly leave the paragraph. Try the following 0F20 C4 00 37 c4 00 33 3F
Execute the prog from 0F20, and the display will show 0F26 3F.
So far so good? Right, now enter 0F20 GO, that is run it again.
The display will then show 0F27 nn. Why? Well examine 0F25 and you will discover that the 33 has turned into a 22!!

This did not happen with the old monitor and an examination of 0001 will reveal all! 0001 CF FF will load whatever is in the accumulator into (P3) - 1. Now P3 will contain 0F26 and the 33 instruction will leave the old value of P3 low in the accumulator.

So the moral of the tale is that whenever you want to abort after having used pointer 3, don't jump to 0001, jump to 0003 otherwise you will find yourself with a nasty program bug.

BAD MEMORY - 2

There is a quick and easy way of finding out any flaws in your memory and will test any 18 consecutive bytes.

The way is to persuade your obliging monitor to give up its 0F00 area of ram and move home to somewhere else, e.g. 0B00-12.

You see the monitor routine needs these bytes so that it knows what it is displaying, and needs many a loop counter for its housekeeping.

So to test e.g. 0B00-12, enter 0FFB 0B 0FFC 12 and run from 0040. Hopefully you should notice no difference, but if you get stutterings on the display or anything that is abnormal, you know that you have got gammy memory in the area 0B00-12.

BACK ISSUES

I have currently got 2 copies of newsletter no. 1, and probably in a month will have newsletter no.2.

If anyone would like to see them, send a stamped address envelope to me and return as soon as poss.

The rest of the newsletter will be individual items, so good luck with whatever you are doing and I hope the list of members will help some of you get in touch with each other.

More on Logic

Last issue looked at Boolean Algebra, that is the AND OR type functions.
This month I will go over some arithmetic operations.

Add

This function adds a number to the accumulator totally in Binary.
Try the following: (press reset first)

```
OF20 C4 03 (load 3)
     F4 03 (add 3)
     C8 02 (store after the 3F)
     3F    (return)
```

Run from OF20, the answer 6 will appear on the right.
Note that we are not talking about two's complement add any more, if you add 80 and 80 you will get 00 but you have now left a time bomb!
So enter 80 into locations OF21 and OF23 and run again.
The display will show 00.
Now put the 3s back where they were! Run again and you will be surprised.
Well it has always been maintained that Computers cannot make decisions!
Our display shows that 3 plus 3 is 7!!
Run again and our computer gets its sums right this time.

The time bomb I mentioned is called the Carrylink and enables you to add up numbers of any length.

When an add occurs, the carry link is also added. This is simply a one digit binary number and is stored in the most significant position in the Status Register.

So, to start off an add, clear this carry first with the 02 CCL instruction. Place an 02 at OF1F and retry the above examples.(running from OF1F this time)

But while the computer will happily add 4F and 9E together, we prefer to add numbers like 79 and 68.

Decimal Add

This function is the same as Add but bases its calculations on numbers 0-9 (like us).

So replace F4 with EC in our examples and you will find 3 plus 3 still equals 6 (or 7 if you haven't set the carry) but 80 plus 80 will now equal 60 (?). The 1 in the hundreds column has not been lost because there is now a 1 set in the carry link (test location OFFF which is the Status register as it came out of your program).

Thus you can add 2 6BCD numbers (or any number) just by letting the carry influence each add.

More Logic Contd...

Complement and Add

The complement and add function is the equivelent of subtract when used properly.

When used in one byte arithmetic, to subtract one byte from another, use the complement and add having set the carry to 1 with an SCL (03) command. E.g. 0F20 03 C4 09 Load 9
0F23 FC 5 C&A 5
C8 02 3F

The display should show 4 on termination.

You can also chain subtracts together as with adds.
First of all set the carry link to 1, then deal with each byte from the least significant to the most significant.

The C/L is acting as a borrow, set to 0 when it has borrowed.

There is no decimal subtract for SC/MP, but for single 2 digit numbers, just complement & add the number from 9A (with carry set) to get the decimal complement, and then add the result to the first number. (decimal add with Carry reset).

Next time I shall go into the principles involved in multiply and divide

Also : How to program a board game

MOVING DASH PROGRAM (or spot the looney)

This program is intended to be part of a routine to display some sort of movement on your 7 segment display.
By using tables the program is incredibly compact and each change of direction requires only 3 bytes.
No program description is given to conserve space.

```
0F20 C4 0D 35 C4 00 31 C4 0B 37 C4 40 33 C4 FF 01 C7 01 E4 88 98 EB
0F35 C7 02 C8 E7 40 02 F3 FD D4 07 01 C3 FF C9 80 8F 00 B8 D7 9C FA
0F4A B8 D4 9C EB 90 DF
```

table at 0B40 (could be relocated) example: 01 08 01 00 01 02 00 01 04 FF 07 08 00 01 10 00 01 20 88

The table consists of groups of 3 bytes, terminated by 88.
The first byte gives the 'velocity', 1 for right to left, FF for left to right and 0 for no movement.

MOVING DASH PROG. contd

The 2nd byte gives the number of times the movement is to take place. The last byte is the character to be displayed in this position.

To speed up/slow down the display, adjust the 8F 00 accordingly. Using the example given, the display should show a mad electron endlessly flowing anti-clockwise.
To turn the program into a random display program, make the following changes: 0f27 00

0F45 11 08 08 08 08 08

This will now use the monitor as the table and make the movements very rapidly.

You may also like to try changing 0F3B (add) to some other functi such as OR and see what the effects are.

Cassette Recovery

Some of you with a cassette interface must have at some time found you had recorded a program perhaps too near the beginning (on to the leader tape) or the switch on click.

As a result you have a 1 in 8 chance of having part of the program intact. But most of the time you will get a program completely displaced by say 1 bit.

This routine will shuffle down your program by 1 bit and thus may be needed to be run 7 times.
The program deals with 256 bytes at a time, and the pointer must be set up to contain the highest point in the program.

```
880 (or wherever convenient) C4 00 36  1 greater than top address.
                             C4 00 32  Address low
                             C6 FF 1F  Load @(2) and shift right w.1
                             CA 00     Store back
                             B8 04
                             9C F7 3F 00
```

The program listing would work for any program in 0F00-FF.

JUNE-JULY 1979 Issue 3.

Welcome to the third issue of C& A.

GENERAL

I am indebted for all of you who added a contribution to the first newsletters, I shall try to put as many of your letters as possible into other circulations. Incidentally, there are now over 40 members of the MUC (MK Fourteen Users Club) two of which are living abroad. One is from Denmark, he actually 'phoned me! The other is from France, writing after seeing a letter in Personal Computer World. I am grateful to Steve Birch for photocopying one newsletter and starting its distribution.
This issue includes a list of all members except the following late arrivals: [Addresses withheld]
Now you have got no excuse for not meeting the folk in your area. (Not that I can talk - my recent attempt to meet Mike Gundy ended up with me just missing him by a few minutes)
I managed to get another letter published in PCW recently, and also my Maze Program in Computing Today. (They haven't paid me yet).

2)

I am hoping to contribute to a SC/MP column in Practical Computing, and possibly some program for PE'S VDU via Mr. Berk who designed the system.
Coming back to toe subject of meetings, would anyone like to arrange some meeting place and time in London? (I will appoint them M.O. (meeting Organiser)) I am too busy at the moment getting newsletters out and answering letters.

1979 COMPUTER SHOW

From 5-7th July, there is a Micro-Computer Exhibition being held. I certainly intend to go, probably on the 6th (it will be crowded on the Saturday) to see the exhibition and mess about on the PETs (and. Mk14s if any)

For more information consult Practical Computing.

Cassette Interface

Originally (many months ago) I built a Cassette Interface module and found to my disappointment that it didn't work. So I (hopefully) sent it off to S of C. and forgot all about it.
Since then I purchased another from them (via Greenbank) and got it working. They have now returned the old one (Free of charge!) which is now lying around doing nothing.

Would anyone who would like this C.I. (I haven't tested it) please send me a stamped addressed jiffy bag(!!) and 2 books of 9p stamps so that I can regain some of my costs.

If you 'phone me up first you can determine whether someone else has nabbed it before you!
4K NIBL BASIC
In the most recent issue of Elektor there was a listing of a Basic Interpreter for SC/MP which might help anyone who is trying to fit a mini Basic into 512 bytes.
This magazine has quite a lot of SC/ME based products, they don't seem to like other micros.

3)

PE VDU
Since the newsletter no. 2, I have managed to connect up a Practical Electronics VDU with 95 per cent success.
The VDU provides 16 lines of 64 characters mapped onto 1K of memory. These characters can be either white on black background or vice-versa. All upper case ASCII characters are provided, 64 in all, but I have added a lower case character generator which provides some peculiar graphics too!
The VDU is much easier to use than the MK14 display as you do not have to multiplex the output. and you are not limited to 7 segments.
Most of the software I am writing, or have written should easily be adapted for S. of C's own VDU as it is also based on the memory mapped principle. (Paul from Kenton tells me they are going to charge £30)
However I have the problem that writing to a location in the TDU'S ram occasionally causes a corruption in another location.
I have rigged it so that I can either have reliable memory or DU. (The or is exclusive-or i.e. E4).
If anyone thinks they have the solution, please let me know.

So far Mark Hyllie is the only one with a solution to the insert delete routines, His solutions were correct but only work for 256 bytes at a time, basically the programs are the same. I think I gave the solution in issue 2 but here they are anyway:
INSERT P2 points to start of where insertion is needed, extension reg. contains byte. enter these at OFFB, C for address of insertion and OFFE for the byte. Program (relocatable): C6 01 01 CA FF E4 88 9C 37 3F. The program must be terminated by 88. In the same way:
DELETE routine: C6 O1 CA FE E4 88 9C F8 3F, again, P2 points to start, and the program finishes with 88.

PROBLEM FOR THE MONTH
The problem is a bit more tricky this time. Find out how many ways there are to load zero into the accumulator using up to 2 bytes and making no assumptions about the current status of other registers.
I will start you of: with Load Immediate 00! I can think of at least 4 others.

4)

BAD MEMORY?
For those without the revised monitors, kindly leave the paragraph. Try the following
OF20 C4 00 37 c4 00 33 3F Execute the prog from 0F20, and the display will show OF26 3F. So far so good? Right, now enter OF20 GO, that is run it again. The display will there show OF27 nn. Why? Well examine OF25 and you will discover that the 33 has turned into a 22!!
This did not happen with the old monitor and an examination of 0001 will reveal all! 0001 CF FF will load whatever is in the accumulator into (P3) - 1. Now P3 will contain OF 26 and the 33 instruction will leave the old value of P3 low in the accumulator.
So the moral of the tale is that whenever you want to abort after having used pointer 3, don't jump to 0001, jump to 0003 otherwise you will find yourself with a nasty program bug.

BAD MEMORY - 2
There is a quick and easy way of finding out any flaws in your memory and will test any 18 consecutive bytes. The way is to persuade your obliging monitor to give up its OFOO area of ram and move home to somewhere else, e.g. OB00-12.
You see the monitor routine needs these bytes so that it knows what it is displaying, and needs many a loop counter for its housekeeping.

So to test e.g. OB00-12, enter OFFB OB OFFC 12 and run from 0040. Hopefully you should notice no difference, but if you get stutterings on the display or anything that is abnormal, you know that you have got gammy memory in the area OB00-12.

BACK ISSUES
I have currently got 2 copies of newsletter no. 1, and probably in a month will have newsletter no.2. .
If anyone would like to see them, send a stamped address envelope to me and return as soon as poss.
The rest of the newsletter will be individual items, so good luck with whatever you are doing and I hope the list of members will help some of you get in touch with each other.

5)
More on Logic
Last issue looked at Boolean Algebra, that is the AND OR type functions.
This month I will go over some arithmetic operations.

Add
This function adds a number to the accumulator totally in Binary. Try the following: (press reset first)
OF 20 04 03 (load 3)
F4 03 (add 3)
C8 02 (store after the 3F)
37 (return) Run from OF 20, the answer 6 will appear on the right. Note that we are not talking about two's complement add any more, if you add 80 and 80 you will get 00 but you have now left a time bomb! So enter 80 into locations OF21 and OF23 and run again.
The display will show 00. Now put the 3s back where they were! Run again and you will be surprised. Well it has always been maintained that Computers cannot make decisions! Our display shows that 3 plus 3 is 7!! Run again and our computer gets its sums right this time.
The time bomb I mentioned is called the Carry link and enables you to add up numbers of any length.
When an add occurs, the carry link is also added. This is simply a one digit binary number and is stored in the most significant position in the Status Register.
So, to start off an add, clear this carry first with the 02 CCL instruction. Place an 02 at OF1F and retry the above examples.(running from OFIF this time) But while the computer will happily add 4F and 9E together, we prefer to add numbers like 79 and 68.

Decimal Add
This function is the same as Add but bases its calculations on numbers 0-9 (like us).
So replace F4 with EC in our examples and you will find 3 plus 3 still equals 6 (or 7 if you haven't set the carry) but 80 plus 80 will now equal 60 (?). The 1 in the hundreds column has not been lost because there is now a 1 set in the carry link (test location OFFF which is the Status register as it came out of your program).
Thus you can add 2 BCD numbers [of any length] just by letting the carry influence each add.

6)

More Logic Contd...

Complement and Add

The complement and add function is the equivalent of subtract when used properly.
When used in one byte arithmetic, to subtract one byte from another, use the complement and add having set the carry to 1 with an SCL (03) command.
E.g.
OF20 03 04 09 Load 9
OF23 FC 5 C&A 5
C8 02 3F The display should show 4 on termination.

You can also chain subtracts together as with adds. First of all set the carry link to 1, then deal with each byte from the least significant to the most significant.
The c/L is acting as a borrow, set to 0 when it has borrowed.
There is no decimal subtract for SC/MP, but for single 2 digit numbers, just complement & add the number from 9A (with carry set) to get the decimal complement, and then add the result to the first number. (decimal add with Carry reset).

Next time I shall go into the principles involved in multiply and divide.

Also : How to program a board game MOVING DASH PROGRAM (or spot the looney)

This program is intended to be part of a routine to display some sort of movement on your 7 segment display. By using tables the program is incredibly compact and each change of direction requires only 3 bytes. No. program description is given to conserve space.
OF20 C4 OD 35 C4 00 31 C4 OB 37 C4 40 33 C4 FF 01 C7 01 E4 88 98 EB
OF35 C7 02 C8 E7 40 02 F3 FD D4 07 01 C3 FF C9 80 8F 00 B8 D7 9C FA
OF4A B8 D4 9C EB 90 DF
table at 0B40 (could be relocated) example: 01 08 01 00 01 02 00 01 04 FF 07 08 00 01 10 00 01 20 88
The table consists of groups of 3 bytes, terminated by 88. The first byte gives the 'velocity', 1 for right to left, FF for left to right and 0 for no movement.

MOVING DASH PROG. contd.

The 2nd byte gives the number of times the movement is to take place. The last byte is the character to be displayed in this position.

To speed up/slow down the display, adjust the 8F 00 accordingly. Using the example given, the display should show a mad electron endlessly flowing anti-clockwise. To turn the program into a random display program, make the following changes :
0F27 00
OF45 11 08 08 08 08 This will now use the monitor as the table and make the movements very rapidly.
You may also like to try changing OF3B (add) to some other function such as OR and see what the effects are.

Cassette Recovery

Some of you with a cassette interface must have at some time found you had recorded a program perhaps too near the beginning (on to the leader tape) or the switch on click.
As a result you have a 1 in 8 chance of having part of the program intact. But most of the time you will get a program completely displaced by say 1 bit.

This routine will shuffle down your program by 1 bit and thus may be needed to be run 7 times. The program deals with 256 bytes at a time, and the pointer must be set up to contain the highest point in the program. 880 (or wherever convenient)

```
C4 00 36   1 greater than top address.
C4 00 32   Address low
C6 FF IF   Load @(2) and shift right w.l
CA 00      Store back
B8 04
9C F7 3F 00 The program listing would work for any program in OF00-FF.
```

(end of Issue 3)