COMPLEMENT AND ADD

NO. 4 AUGUST-SEPTEMBER 1979

Hello again, the last two months have been rushing past and it is again time for me to put the pieces together for another 8 page wonder.

Before I go too much further, lets have some good news and some bad news.

Good News is that there are 111 folk interested in the MK14 User Group from all corners of G.B. and beyond.

Bad News is that I am forced to charge a small fee for the newsletter, £1 for the next 6 newsletters, which should not lighten your wallets too much! This is just to cover the photocopying & postage costs. Of course the basic idea of circulating newsletters is the same - if anyone wants their own copy to keep, the charge is 70p including P/P per newsletter (50p for issues 1,2,3)
Also if someone would like a newsletter to borrow for one week (and copy if desired) then the cost is 30p.
Of course those helping with copying etc are not subject to this cost.
In time I shall get a Scotch copier and this will allow the newsletters to get fatter.

It would be faster if the fee could be sent directly to me rather than in a newsletter.

Perhaps some of you may like to get together, recieving one newsletter between you.

The SC/MP members list has been updated to three pages and should help you meet each other, perhaps while on holiday!

Since Last Newsletter

One or two things have been published since June, my Silly Basic for example, and some mentions of the club by CT, PC.

Also, since no rules were published about Maze in CT, here is a brief summary.

When a question is asked, e.g. 'Ford lft?' you must reply 0 for the first option, 1 for the second option. So in this example, 1 would mean that you wanted to go left whereas in the question 'Left-rt?' it would mean go right. The game ends when 'END' is displayed. Comments are displayed E.g. Bench that require no answer.

National Semiconductor have kindly sent some details on the 70 series family, and this material has confirmed my suspicions that the devices are based on the SC/MP 11. A full report later in this issue!

Also in this issue is a program written by Clive Isbell; incidentally

if you want something put in this newsletter, as part of the newsletter not just a circulating note of interest, just send it to me directly. Another excerpt is from Mr. A.J.Whitehead.
Obviously I cannot put all your interesting mail in the newsletter, except in different circulations.

To brighten up this newsletter, is anyone of artistic circles, it would be nice to have some interesting photos, drawings or cartoons or excuses from S. of C. (we're sorry your VDU is delayed, but all our 74LS00's were struck by lightening)

The September issue of Practical Electronics is worth buying - there is a good SC/MP puzzle to work out, the prize being a S. of C. VDU.

It is amazing what Sneeze 5 MK14's have got built on to them in comparison to the Sneeze 2's. For a start all the 1½K wasted by the monitor in the no. 2's is decoded out, enabling you to easily use the area for extra ram. There is also edge connections underneath on the far side. The manual has been vastly improved, no programs added though!
(sneeze is a pun on issue - what do you mean I should get a new script-writer?!)

Extra digits

I am informed by Paul from Kenton, who in fact sent me two tips - one for next issue!, that there is a way of getting the extra digit available on some machines to do something.
1. On the underside of the board, solder a wire to pin 10 of IC13, after removing the device if you desire.
2. On the underside again, with the keyboard towards you, locate the displa connections. Solder the other end of the wire to the second connector from the right.
3. The new digit is location 0D08, or 0F08 if using the monitor. Unfortunately, the monitor does not zero this byte, so you must do it yourself.

Last months puzzle

I get the feeling that no-one is doing my puzzles, but you're going to get the answer anyway!
The problem was to find as many ways of loading zero into the A register using two bytes.
1) C4 00 2) C0 00 3) D4 00 4) D0 00 5) 40 60

Here is a quickie that must be solved in one minute flat:
How do you shift left one bit a number in the extension register?
You have a shift right instruction, but you would need 7 shift rights for one shift left. Only 3 bytes are needed!

Here's that contribution from Mr. A. Whitehead:

I have written a program for the MK 14/Tape module to record and play-back up to 64K bytes with recorder motor on/off & 10 second pause at start & end of recording.

Program listing is as follows, this can conveniently placed in the RAM. I/O memory i.e. 0880 to 08FF :-

```
0880 C4 82 C8 7A 36 C8 78 32
0888 C8 76 90 04 C4 02 90 F2
0890 9C 02 B8 6B C0 68 07 94
0898 1D C4 08 C8 60 06 D4 20
08A0 98 FB 8F 1C 19 8F 1C B8
```

```
08A8 54 9C F2 40 CD 01 B8 50
08B0 9C E7 B8 4B 94 E3 C4 4C
08B8 C8 43 8F FF B8 3F 9C FA
08C0 C0 3C 94 04 C4 00 07 3F
08C8 C5 01 01 C4 01 C8 2F C4
08D0 03 07 8F 08 C0 28 50 98
08D8 07 8F 18 C4 02 07 90 05
08E0 C4 02 07 8F 18 8F 20 C0
08E8 15 F0 13 9C E0 B8 11 9C
08F0 D7 B8 0C 94 D3 90 BF 08
08F8 08 08 08 08
```

Memories:-	08FC	Delay loop
	08FD	Pointer & Mask
	08FE	Program length M.S.B.
	08FF	" " L.S.B.
Enter:-	P1H	Program address M.S.B.
	P1L	" " L.S.B.
	P2H	" length M.S.B.
	P2L	" " L.S.B.
Start:-	0880	Tape → Memory
	088C	Memory → Tape

Note:- 088C was selected for quick keying on my MK 14 which has a separate differently laid out key-board.

Review of the SC/MP 111 (My terminology not National's)

The 70 series family, that is the 8070 64 bytes of ram.
8072 64 bytes of ram 2.5K rom.
8074 64 bytes of ram 4K rom.

The similarities to SC/MP are apparant from the cover, signals like NHOLD,NENIN and NENCUT.
The on board ram is at FFC0-FFFF and can be used as a program area or a stack area.
The registers available are similiar to the 8060, with the exception of:
P1 is replaced by a Stack pointer register.
T register is a new 16 bit register.
The Accumulator and Extension Register can be used as a 16 bit register for exchanging with the T register etc.
One of the amazing things about this device is the on-chip multiply and divide! The multiply uses 16 bit operands and produces a 32 bit answer.
The divide gives a 16 bit quotient and 16 bit remainder from 16 bit numbers.
Nat Semi have been very sneaky and called Complement and add 'subtract'!
We may have to change the title of this newsletter to follow suit.
What else? Oh yes how about the search and skip instruction, which looks through 256 memory locations specified by P2 or P3. If a match occurs

then the follwing two bytes of the program are skipped, leaving the pointer 1 greater than the address of the matching byte.
Maybe you think that having lost P1, leaving P2 and P3, that there is lack of pointers. Not so! Assuming the Stack pointer to be set up (or as it was on switch on) you can pop/push any register to and from the stack with just one byte!
Then there is the testto branch if a digit in the accumulator is not a valid numeric ascii code, and convert it to bcd if it is!
Branch facilities are similiar, with an absolute version of unconditional jump, letting you jump anywhere in your 64K.
Also using the stack idea, there is a 3 byte subroutine call, and you don't upset P3 in the process! (A one byte return)
A lot of codes (e.g. Add is F0) are unchanged from the old SC/MP, but there are enough changes to stop you plugging an 8070 into your SC/MP socket! Of course the pinouts are different, but perhaps what could happen in the future is an 8072 in a socket wired up in some way to the SC/MP socket in the MK14, and of course a new monitor (with extra goodies) to handle the new op-codes.

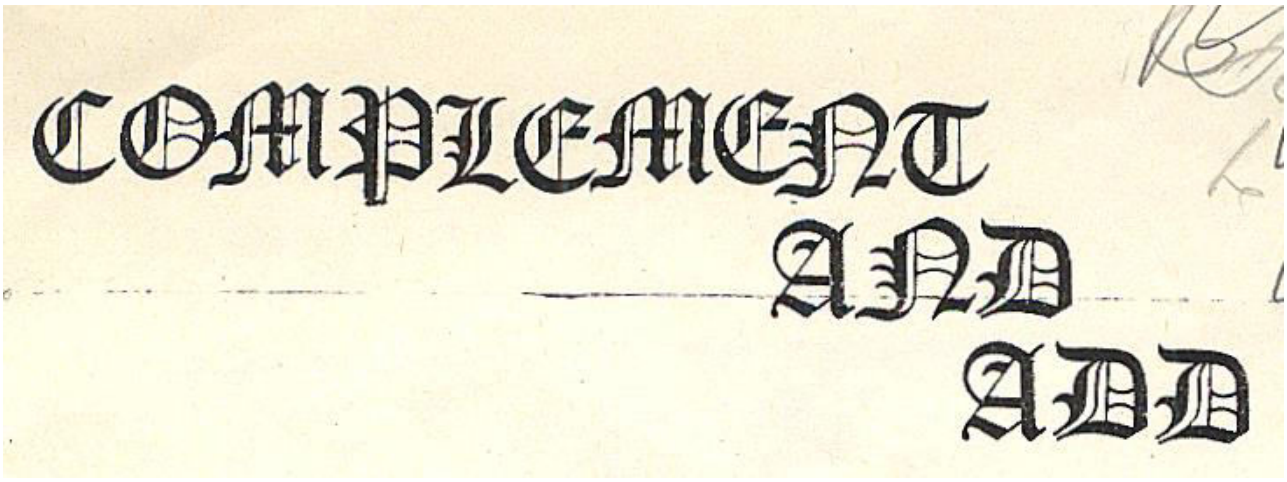
Horse Race Program

If I didn't keep getting such interesting letters from SC/MP owners, I could get down to some serious program writing! As it is, you will have to put up with my old horse race program.
All you have to do is to decide which 'horse' you think will win.
A horse is represented by either '‾' or - or '_', top middle and bottom segments. They move from left to right until at least one drops of the edge, winning the race. The display will freeze until 0 is pressed for 3 seconds and a new game will begin.
No program mnemonics given, I have not the time or energy to work them out!

```
0F1C C0 FE F0 FB C8 F9          Random numbers
0F22 1E F0 F7 C8 F5 08
0F28 08 08 C0 EF D4 01          Heads or tails?
0F2E 9C 06 B8 E4 94 02          Move a horse!
0F34 A8 DF                      Set a flag if won.
0F36 C0 E3 D4 02 9C 06          Move another horse!
0F3C B8 D9 94 02 A8 D3          If won set a flag.
0F42 C0 D7 D4 04 9C 06          Toss coin for last horse.
0F48 B8 CE 94 02 A8 C7          Move horse, set flag if won.
0F4E C0 C6 01                   Position of horse into e-reg.
0F51 C4 0D 37 C4 00 33          P3 contains 0D00 for display.
0F57 C4 23 CB 80                Move top horse.
0F5B 8F 00 C0 B8 01             Delay and move 2nd horse to e-reg.
0F60 C4 40 CB 80 8F 00          Move horse to display.
0F66 C0 B0 01                   Last horse into e-reg.
0F69 C4 1C CB 80 8F 00          Move out & delay.
0F6F B8 A2 9C DB                Show display for  16 times.
0F73 C4 10 C8 9C                Outside loop for display.
0F77 B8 9B 9C D3                Loop 256 times.
0F7B C4 FF C8 95 C0 94 98 99    Test winning flag to see if race ended
0F83 AB 00 98 C7                Test if key pressed, if not them show display
0F87 C4 00 C8 8A C4 09 C8 87 C8 86 C8 85 90 EA
                                Reset initial conditions for new game.
```

Execute program from 0F87, when storing on cassette, store only 0F1C.... to ensure random numbers are random!

If you like, each segment position represents one furlong!!



NO. 4
AUGUST-SEPTEMBER 1979

Hello again, the last two months have been rushing past and it is again time for me to put the pieces together for another 8 page wonder.

Before I go too much further, lets have some good news and some bad news.

Good News is that there are 111 folk interested in the MK14 User Group from all corners of G.B. and beyond.
Bad News is that I am forced to charge a small fee for the newsletter, £1 for the next 6 newsletters, which should not lighten your wallets too much! This is just to cover the photocopying & postage costs. Of course the basic idea of circulating newsletters is the same - if anyone wants their own copy to keep, the charge is 70p including P/P per newsletter (50p for issues 1,2,3) Also if someone would like a newsletter to borrow for one week and copy if desired) then the cost is 30p. Of course those helping with copying etc are not subject to this cost. In time I shall get a Scotch copier and this will allow the newsletters to set fatter.
It would be faster if the fee could be sent directly to me rather than in a newsletter
Perhaps some of you may like to get together, receiving one newsletter between you.
The SC/MP members list has been updated to three pages and should help you meet each other, perhaps while on holiday!

Since last Newsletter
one or two things have been published since June, my Silly Basic for example, and some mentions of the club by CT, PC,
Also, since no rules were published about Maze in CT, here is a brief summary When a question is asked, e.g. 'Ford lft?' you must reply 0 for the first option, 1 for the second option. So in this example, 1 would mean that you wanted to go left whereas in the question 'Left-rt?' it would mean go right. The game ends when "END" is displayed. Comments are displayed E.g. 'Bench' that require no answer.

National Semiconductor have kindly sent some details on the 70 series family, and this material has confirmed my suspicions that the devices are based on the SC/MP 11. A full report later in this issue!
Also in this issue is a program written by Clive Isbell; incidentally

If you want something put in this newsletter, as part of the newsletter not just a circulating note of interest, just send it to me directly. Another excerpt is from Mr. A.J. Whitehead. Obviously I cannot put all your interesting mail in the newsletter, except in different circulations.

To brighten up this newsletter, is anyone of artistic circles, it would be nice to have some interesting photos, drawings or cartoons or excuses from S. of C. (we're sorry your VDU is delayed, but all our 74LSOO's were struck by lightening)
The September issue of Practical Electronics is worth buying - there is a good SC/MP puzzle to work out, the prize being a S. of C. VDU.

It is amazing what Sneeze 5 MK14's have got built on to them in comparison to the Sneeze 2's. For a start all the 12K wasted by the monitor in the no. 2's is decoded out, enabling you to easily use the area for extra ram. There is also edge connections underneath on the far side. The manual has been vastly improved, no programs added though! (sneeze is a pun on issue - what do you mean I should get a new scriptwriter?!)

Extra digits
I am informed by Paul from Kenton, who in fact sent me two tips - one for next issue!, that there is a way of getting the extra digit available on some machines to do something.
1. On the underside of the board, solder a wire to pin 10 of IC13
after removing the device if you desire.
2. On the underside again, with the keyboard towards you, locate the display connections. Solder the other end of the wire to the second connector from the right..
3. The new digit is location OD08, or OF08 if using the monitor.
Unfortunately, the monitor does not zero this byte, so you must do it it yourself.

Last months puzzle
I get the feeling that no-one is doing my puzzles, but you're going to get the answer anyway! The problem was to find as many ways of loading zero into the A register using two bytes.
   1) C4 00 2) CO 00 3) D4 00 4) DO 00 5) 40 60
   2)
Here is a quickie that must be solved in one minute flat: How do you shift left one bit a number in the extension register? You have a shift right instruction, but you would need 7 shift rights for one shift left. Only 3 bytes are needed!

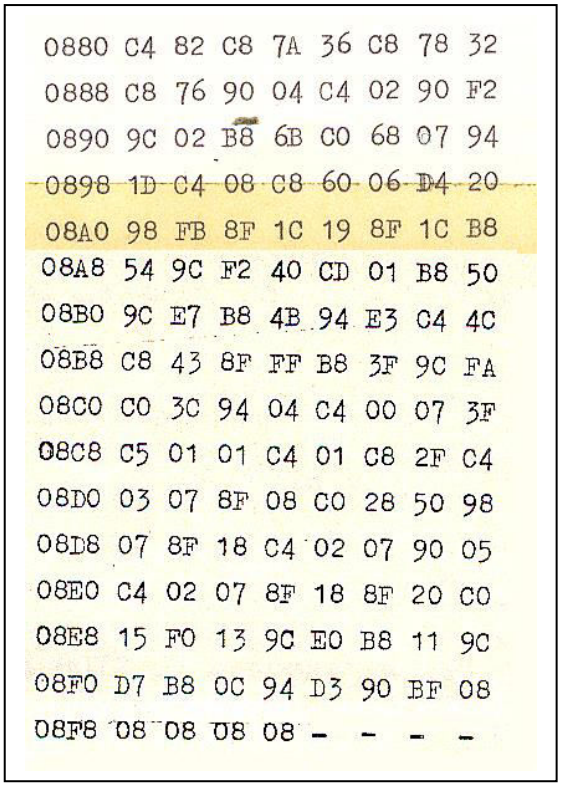
Here's that contribution from Mr. A. Whitehead:
I have written a program for the MK 14/Tape module to record and play-back up to 64K bytes with recorder motor on/off & 10 second pause at start & end of recording.
Program listing is as follows, this can conveniently placed in the RAM. I/O memory i.e. 0880 to 08FF :

```
0880  C4 82 C8 7A 36 C8 78 32
0888  C8 76 90 04 C4 02 90 F2
0890  9C 02 B8 6B CO 68 07 94
0898  1D C4 08 C8 60 06 D4 20
08A0  98 FB 8F 1C 19 8F 10 B8
```

```
08A8  54 9C F2 40 CD 01 B8 50
08B0  9C E7 B8 4B 94 E3 C4 4C
08B8  C8 43 8F FF B8 3F 9C FA
08CO  CO 3C 94 04 C4 00 07 3F
08C8  C5 01 01 C4 01 C8 2F C4
08D0  03 07 8F 08 C0 28 50 98
08D8  07 8F 18 C4 02 07 90 05
OBE0  C4 02 07 8F 18 8F 20 CO
08E8  15 FO 13 9C E0 B8 11 9C
08F0  D7 B8 0C 94 D3 90 BF 08
08F8  08 08 08 08 - - - -

Memories:- 08FC delay loop
           08FD Pointer and mask
           08FE Program length M.S.B.
           08FF  "          "   L.S.B


Enter:
P1H Program address M.S.B.
P1L                 L.S.B.
P2H length M.S.B.
P21        L.S.B.
0880 Tape -> Memory
088C Memory -> Tape
```



088C was selected for quick keying on my MK 14 which has a separate differently laid out keyboard.

Review of the SC/MP III (My terminology not Nationals) The 70 series family, that is the
8070 64 bytes of ram.
8072 64 bites of ram 2.5K rom.
8074 64 bytes of ram 4K rom. The similarities to SC/MP are apparent from the cover, signals like NHOLD, NENIN and NANCUT. The on board ram is at FFCO-FFFF and can be used as a program area or a stack area. The registers available are similar to the 8060, with the exception of: Pl is replaced by a Stack pointer register. T register is a new 16 bit register. The Accumulator and Extension Register can be used as a 16 bit register for exchanging with the T register etc. One of the amazing things about this device is the on-chip multiply and divide! The multiply uses 2 16 bit operands and produces a 32 bit answer. the divide gives a 16 bit quotient and 16 bit remainder from 16 bit number's. Nat Semi have been very sneaky and called Complement and add 'subtract!!' We may have to change the title of this newsletter to follow suit. What else? Oh yes how about the search and skip instruction, which looks through 256 memory locations specified by P2 or P3. If a match occurs

then the following two bytes of the program are skipped, leaving the pointer 1 greater than the address of the matching byte. Maybe you think that having lost Pl, leaving P2 and P3, that there is lack of pointers. Not so! Assuming the Stack pointer to be set up (or as it was on switch on you can pop/push any register to and from the stack with just one byte! Then there is the test to branch if a digit in the accumulator is not a valid numeric ascii code, and convert it to bed if it is! Branch facilities are similar, with an absolute version of unconditional jump, letting you jump anywhere in your 64K. Also using the stack idea, there is a 3 byte subroutine call, and you don't upset P3 in the process! (A one byte return) A lot of codes (e.. Add is FO) are unchanged from the old SC/MP, but there are enough changes to stop you plugging an 8070 into your SC/MP socket! Of course the pinouts are different, but perhaps what could happen in the future is an 8072 in a socket wired up in some way to the SC/MP Socket in the MK14, and of course a new monitor (with extra goodies) to handle the new op-codes.

Horse Race Program.

If I didn't keep getting such interesting letters from SC/MP owners, I could get down to some serious program writing!
As it is, you will have to put up with my old horse race program. All you have to do is to decide which 'horse' you think will win. A horse is represented by either or - or ', top middle and bottom segments. They move from left to right until at least one crops of the edge, winning the race. The display will freeze until 0 is pressed for 3 seconds and a new cane will begin. No program mnemonics given, I have not the time or energy to work them out!

```
OF1C  CO FE FO FB C8 F9  Random numbers
OF22  1E FO F7 C8 F5 08
OF28  08 08 CO EF D4 01 Heads or tails?
OF2E  9C 06 B8 E4 94 02 Move a horse!
OF34  A8 DF             Set a ilag if won.
OF36  CO E3 D4 02 9C 06 Move another horse!
OF3C  B8 D9 94 02 A8 D3 If won set a flag.
OF42  CO D7 D4 04 9C 06 Toss coin for last horse.
OF48  B8 CE 94 02 A8 C7 Move horse, set flag if won.
OF4E  CO C6 01          Position of horse into e-reg.
OF51  C4 OD 37 C4 00 33 P3 contains ODOO for display.
OF57  C4 23 CB 80       Move top horse.
OF5B  8F 00 CO B8 01    Delay and move 2nd horse to e-reg.
OF60  C4 40 CB 80 8F 00 Move horse to display.
OF66  CO BO 01          Last horse into e-reg.
OF69  C4 1C CB 80 8F 00 Move out & delay.
OF6F  B8 A2 9C DB       Show display for 16 times.
OF73  C4 10 C8 9C       Outside loop for display.
OF77  B8 9B 9C D3       Loop 256 times.
OF7B  C4 FF C8 95 CO 94 98 99 Test winning flag to see if race ended
OF83  AB 00 98 C7       Test if key pressed, if not them show display
OF87  C4 00 C8 8A C4 09 C8 87 C8 86 C8 85 90 EA Reset initial conditions for new
game.
```

Execute program from OF87, when storing on cassette, store only OFIC.... to ensure random numbers are random! If you like, each segment position represents one furlong!!

PROGRAM TO CONVERT °Centigrade TO °Farenheit, from 1°C to 28°C.

; Written as a training exercise to discover how the SC/MP instructions work.
; Only simple "one byte" maths is used and each sub-program is kept separate without attempting to reduce the length of the program.
; °C are keyed in, in decimal, the 10s as units in OF12 and the units in OF13.
; °F displayed in data section in decimal, preceded by FFFF in address section.
; Pointer 2 set by program and utilized throughout.
; Pressing reset button clears display, and allows the next data entry.

OF12 = Tens as units° C ends with decimal °F
OF13 = Units °C
OF14 = Temporary store
OF15 = " "

; Program commences at OF19.

```
OF19  C40F   LDI  OF      ┐
OF1B  36     XPAH 2       │ Set pointer 2
OF1C  C400   LDI  00      │ to OF00
OF1E  32     XPAL 2       │
OF1F  08     NOP          ┘
OF20  C212   LD  OF12     ┐
OF22  CA14   ST  OF14     │
OF24  C409   LDI 9        │
OF26  CA15   ST  OF15     │ Converting
OF28  C214   LD  OF14     │ decimal
OF2A  02     CCL          │ to
OF2B  F212   ADD OF12     │ hexadecimal
OF2D  CA14   ST  OF14     │ TENS in as
OF2F  BA15   DLD OF15     │ units X 10
OF31  9CF5   JNZ to OF28  │ plus UNITS
OF33  C214   LD  OF14     │
OF35  F213   ADD OF13     │
OF37  CA12   ST  OF12     │
OF39  08     NOP          ┘
OF3A  C212   LD  OF12     ┐
OF3C  CA14   ST  OF14     │
OF3E  C408   LDI 8        │
OF40  CA13   ST  OF13     │ Multiply
OF42  C214   LD  OF14     │ by
OF44  02     CCL          │ nine
OF45  F212   ADD OF12     │ by
OF47  CA14   ST  OF14     │ repeated
OF49  BA13   DLD OF13     │ addition
OF4B  9CF5   JNZ OF22     │
OF4D  08     NOP          ┘
OF4F  C400   LDI 0        ┐ Clear
OF50  CA12   ST  OF12     │ temp
OF52  CA13   ST  OF13     ┘ store
OF54  AA12   ILD OF12     ┐
OF56  03     SCL          │
OF57  C214   LD  OF14     │
OF59  FC05   CAI 5        │
OF5B  9813   JZ  OF70     │
OF5D  CA14   ST  OF14     │
OF5F  03     SCL          │ DIVIDE
OF60  FC05   CAI 5        │ by
OF62  CA15   ST  OF15     │ five
OF64  1D     SRL          │ by
OF65  9402   JP  OF69     │ repeated
OF67  90EB   JMP OF54     │ subtraction
OF69  C215   LD  OF15     │
OF6B  02     CCL          │
OF6C  F405   ADI 5        │
OF6E  CA13   ST  OF13     │
OF70  08     NOP          ┘
OF71  02     CCL          ┐
OF72  C212   LD  OF12     │ Add
OF74  F420   ADI 32(10)   │ 32
OF76  CA12   ST  OF12     │
OF78  08     NOP          ┘
OF79  C400   LDI 0        ┐
OF7B  CA13   ST  OF13     │
OF7D  CA14   ST  OF14     │
OF7F  AA13   ILD OF13     │ Convert
OF81  03     SCL          │ Hexadecimal
OF82  C212   LD  OF12     │ to
OF84  FC0A   CAI 10(10)   │ decimal
OF86  9813   JZ  OF9B     │
OF88  CA12   ST  OF12     │
OF8A  03     SCL          │
OF8B  FC0A   CAI 10(10)   ┘
```

```
OF8D  CA15    ST OF15
OF8F  1D      SRL
OF90  9402    JP to OF94      Hex+ 10
OF92  90EB    JMP to OF7F     put result
OF94  C215    LD OF15         in top
OF96  02      CCL             4 bits
OF97  F409    ADI 9           put
OF99  CA14    ST OF14         remainder
OF9B  C213    LD OF13         in lower
OF9D  1E      RR              4 bits
OF9E  1E      RR
OF9F  1E      RR              works
OFA0  1E      RR              15 hex to
OFA1  F214    ADD OF14        63 hex.
OFA3  CA12    ST OF12
OFA5  08      NOP
OFA6  C4FF    LDI FF
OFA8  CA0E    ST AdH OF0E
OFAA  C4FF    LDI FF          Display
OFAC  CA0C    ST AdL OF0C     routine
OFAE  C212    LD OF12         via
OFB0  CA0D    ST Word OF0D     monitor
OFB2  C401    LDI 1 H(DispD)
OFB4  37      XPAH 3
OFB5  C43F    LDI 3F L(DispD)
OFB7  33      XPAL 3
OFB8  3F      XPPC 3 END.
```

Interesting correspondence will be gratefully received, I am always willing to learn.

C. R. ISBELL

FELSTED
ESSEX.

PROGRAM TO CONVERT °Centigrade TO °Farenheit, from 1°C to 28°C.

```
; Written as a training exercise to discover how the SC/MP instructions
  work.
; Only simple "one byte" maths is used and each sub-program is kept
  separate without attempting to reduce the length of the program.
; °C are keyed in, in decimal, the 10s as units in OF12 and the
  units in OF13.
; °F displayed in data section in decimal, preceded by FFFF in address
  section.
; Pointer 2 set by program and utilized throughout.
; Pressing reset button clears display, and allows the next data entry.

      OF12    =  Tens as units° C     ends with decimal °F
      OF13    =  Units °C
      OF14    =  Temporary store
      OF15    =     "        "

; Program commences at OF19.

OF19  C40F     LDI  OF      ---+
OF1B  36       XPAH 2          |  Set pointer 2
OF1C  C400     LDI  00         |  to OF00
OF1E  32       XPAL 2          |
OF1F  08       NOP          ---+
OF20  C212     LD OF12      ---+
OF22  CA14     ST OF14         |
OF24  C409     LDI 9           |  Converting
OF26  CA15     ST OF15         |  decimal
OF28  C214     LD OF14         |  to
OF2A  02       CCL             |  hexadecimal
OF2B  F212     ADD OF12        |  TENS in as
OF2D  CA14     ST OF14         |  units X 10
OF2F  BA15     DLD OF15        |  plus UNITS
OF31  9CF5     JNZ to OF28     |
OF33  C214     LD OF14         |
OF35  F213     ADD OF13        |
OF37  CA12     ST OF12         |
OF39  08       NOP          ---+
OF3A  C212     LD OF12      ---+
OF3C  CA14     ST OF14         |
OF3E  C408     LDI 8           |  Multiply
OF40  CA13     ST OF13         |  by
OF42  C214     LD OF14         |  nine
OF44  02       CCL             |  by
OF45  F212     ADD OF12        |  repeated
OF47  CA14     ST OF14         |  addition
OF49  BA13     DLD OF13        |
OF4B  9CF5     JNZ OF22        |
OF4D  08       NOP          ---+
OF4F  C400     LDI 0        ---+  Clear
OF50  CA12     ST OF12         |  temp
OF52  CA13     ST OF13      ---+  store

OF54  AA12     ILD OF12     ---+
OF56  03       SCL             |
OF57  C214     LD OF14         |
OF59  FC05     CAI 5           |
OF5B  9813     JZ OF70         |
OF5D  CA14     ST OF14         |
OF5F  03       SCL             |  DIVIDE
OF60  FC05     CAI 5           |  by
OF62  CA15     ST OF15         |  five
OF64  1D       SRL             |  by
OF65  9402     JP OF69         |  repeated
OF67  90EB     JMP OF54        |  subtraction
OF69  C215     LD OF15      ---+
OF6B  02       CCL             |
OF6C  F405     ADI 5           |
OF6E  CA13     ST OF13      ---+
OF70  08       NOP          ---+
OF71  02       CCL             |
OF72  C212     LD OF12         |  Add
OF74  F420     ADI 32(10)      |  32
OF76  CA12     ST OF12         |
OF78  08       NOP          ---+
OF79  C400     LDI 0        ---+
OF7B  CA13     ST OF13         |
OF7D  CA14     ST OF14         |
OF7F  AA13     ILD OF13        |  Convert
OF81  03       SCL             |  Hexadecimal
OF82  C212     LD OF12         |  to
OF84  FC0A     CAI 10(10)      |  decimal
OF86  9813     JZ OF9B         |
OF88  CA12     ST OF12         |
OF8A  03       SCL             |
OF8B  FC0A     CAI 10(10)   ---+
```

```
0F8D  CA15    ST 0F15
0F8F  1D      SRL
0F90  9402    JP to 0F94      Hex+ 10
0F92  90EB    JMP to 0F7F     put result
0F94  C215    LD 0F15         in top
0F96  02      CCL             4 bits
0F97  F409    ADI 9           put
0F99  CA14    ST 0F14         remainder
0F9B  C213    LD 0F13         in lower
0F9D  1E      RR              4 bits
0F9E  1E      RR
0F9F  1E      RR              works
0FA0  1E      RR              15 hex to
0FA1  F214    ADD 0F14        63 hex.
0FA3  CA12    ST 0F12
0FA5  08      NOP
0FA6  C4FF    LDI FF
0FA8  CA0E    ST AdH 0F0E
0FAA  C4FF    LDI FF          Display
0FAC  CA0C    ST AdL 0F0C     routine
0FAE  C212    LD 0F12         via
0FB0  CA0D    ST Word 0F0D     monitor
0FB2  C401    LDI 1 H(DispD)
0FB4  37      XPAH 3
0FB5  C43F    LDI 3F L(DispD)
0FB7  33      XPAL 3
0FB8  3F      XPPC 3 END.
```

Interesting correspondence will be gratefully received, I am always willing to learn.

C. R. ISBELL

FELSTED
ESSEX.