16

CORTEX USER GROUP NEWSLETTER (JAN 1988)

Issue Number 16

CONTENTS

1. Index  
2. Letters  
4. Editorial  
5. Programme (Paint and Dump changes)  
10. Double stepping disk drives  
11. Control keys programme  
12. MDEX information  
13. QBASIC part 2  
15. Adverts etc.

REMEMBER TO SEND IN YOUR ARTICLES FOR THE NEXT NEWSLETTER
LETTERS

A.R.C. Badcock  Hants

Users may like to know that the data separator I.C. the FDC9216B is available from M.S. Components Ltd on 01-670-4466 for £7.15 + VAT. I am interested in C.J. Youngs assembler is it significantly better to use than R.M. Lees. Also is it possible to get a single density version of the MDEX boot track BOOT50 or BOOT55.

C.J. Youngs assembler is written in machine code so it tends to run faster than R.M. Lees, also it will handle very large source files and programmes. It does have the disadvantage however of not producing a list file but this facility is to be added in the future. The MDEX system automatically selects single or double density formats by trying to read a disk and if it gets an error trying again in the other density mode. It should therefore be possible to use the boot track files in either density.

W.R. Bucknall  Sheffield

I enclose copy of data manuals for Cannon MD110-220 drives for you to hold for other members that may require more information. Also I'm looking forward to receiving MDEX Pascal from you as we are studying it here.

If anyone would like a copy of the Cannon drives data manual please send £1.00 to cover photocopying.

W.D. Eaves  Caithness

I have no mains switch for my Cortex Mk II and cannot find one that will fit the cutout, can you supply a suitable switch? What type of connector is required for the E.Bus. I have my Centronics interface connected by vero pins through the main P.C.B. in place of the connector. However I would like to add a backplane to enable the use of the Centronics board and other expansion boards but there is no way to get the E.Bus connector out of the case. Are you supposed to cut a hole? In some past newsletter there has been mentioned a communications package called Commtex is it still available and what hardware is required.

The mains switch for the Mk II Cortex is R.S. Components stock number 337-223 and is available from the user group for £1.00 inc.

The E.Bus connector is a DIN 41612 64 way a/c plug. The best way to connect up a backplane is to use a short length of ribbon cable from a mating socket. This will have to be passed through a hole cut in the side of the case. The Commtex package is now available from the group for £5.00 on disc. It will support all normal modems V21/23/24 etc and uses the serial port of the cortex for connection to the modem.

REMEMBER TO SEND IN YOUR ARTICLES FOR THE NEXT NEWSLETTER
Letters
--------
Prem Holdaway

I have entered both of Mr Rudnicki's programmes, Missile command and Canyon, but have not been able to get any control keys to work even though I have checked my typing several times.

Has anyone else been able to get the listings of the two games to run? If so please write in or even supply a copy of the programmes on disk.

O.C. Walden Milton Keynes

Congratulations on the acquisition of the MDEX software and enhanced coverage in the newsletter. I have been using the core system for some years now in particular for all my assembler programming. Perhaps now with affordable support we shall be able to eradicate some of the resident bugs. As you may know mdex files are not written to consecutive sectors on the disk but use interlaced sectors as defined in the Precession table. I have written a routine to use this table to dump Mdex files in the correct sequence while in native Cortex mode. This obtains the full power of MDEX Editor ASM etc to write files and programmes for either mode. Would this be of interest to others.

I should think a lot of users would be interested please send more details. Also send in more details of any resident bugs you know of in MDEX so that we can try to find a solution.

P.J. Riddle Edinburgh

I am writing for the first time about my Cortex. It has laid dormant for some time but seeing your newsletter has rekindled my interest. Mine is a much modified system but here are the basics. Cortex with 9909 disk controller, 8" disk drives with a basic dos I wrote from scratch, Real time battery backed up clock, Battery backed up static ram, 20 meg hard disk interface (not yet completed), Eprom programmer (not yet completed).

As you can see I have been quite busy. And its a great pity the 99xx range has not prospered as it is a good standard. The reason for writing is ask if you can supply MDEX on 8" Bootable disks.

Yes indeed all MDEX and CDOS software can be supplied on 8" disks but only in single sided. I am sure many members would be interested in more details of your add-ons especially the hard disk interface and eprom programmer. Why not write an article or two for the newsletter.

Nigel Osmond Glostershire

Did MPE release the NOS operating system, which is the big brother of MDEX, to the User Group.

No up to date we do not have NOS for sale.
Letters

-------
Dick Hall  Scotland

I already have MDEX P.D.S. but do not have the source listing of MDEX.REL. I have modified my device drivers to accommodate a parallel printer but would also like to modify the main system programme if source is available.

As far as I know source code was never available for MDEX.REL but it should be possible to re-produce it using Anthony Rowell’s disassembler. If any other users have already done so could you please send a copy in to the user group.

D.L. Wright  Fife

I have a Cortex running MDEX which I use to research into computer security. I am now planning to install a C.D.C. with a 36 Megabyte Winchester and I am faced with drastic operating system changes. As I have 192K of memory available Stephen Pelc at M.P.E. has advised me to install NOS and use this to bring the Winchester on line. I have lost touch with the User Group but now wish to re-join both to take advantage of the software availability and to make contributions to the Newsletter for this excellent machine. I also have a TMS 32010 evaluation module which uses a TMS 9995 to interface to the D.S.P.

At the moment we do not have NOS available from the User Group but if you can obtain it from M.P.E. I’m sure many of our readers would be interested in the results you get from interfacing a Winchester drive. We look forward to hearing more from you in the future.

As you can see there has been quite a lot of interest in MDEX so far. I would like to take time to say thanks to Rex Collins who is trying to answer all queries that we receive from MDEX users, and also to Anthony Rowell who has been of great assistance on the subject. The biggest burden to us with MDEX distribution is photocopying the manuals. Prem Holdaway has volunteered to type the MDEX user guide into a file that can be printed by the user. This would enable us to distribute manuals on disk. If anyone is interested in typing up any more manuals please let us know.

We have details of a few Cortex computers for sale on the second hand market, both Mk I and Mk II. Some have disk drives and other extras fitted and some software is included. Prices range from around £50.00 to £180.00. If anyone would like to get hold of a second machine to use or just to keep for spares please let us know and we will pass on the information.

REMEMBER TO SEND IN YOUR ARTICLES FOR THE NEXT NEWSLETTER
Please find enclosed two listings which will be of use to people who use Centronics 739 printers. The first listing is a variation on the 'PAINT' program in Newsletter 4 and the second is a 'DUMP' program based upon the methods used in 'PAINT'. The listings are the result of an exercise to convert the potentially useful programs into a form which I could use and also in order to learn more about programming in machine code.

PAINT

This program produces the same type of output as Tim Gray's 'PAINT' in Newsletter 4. An A4 size sheet is produced with each pixel mapped onto a 3x3 matrix depending upon its colour. The translation is exactly the same although the data in the table looks different. This is simply due to the way the data is prepared for the printer. The 3 bits representing pixel dots are stored in bits 1-3 of a byte in my program and bits 5-8 in the Epsom printer version. As with the original 'PAINT' the code is entirely relocatable.

```
5E00 020A LI R10,>0458
5E04 068A BL R10
5E06 C288 MOV R11,R10
5E08 022B AI R11,>004A
5E0C CA88 MOV R11,@>000B2(R10)
5E10 CA88 MOV R11,@>00E0(R10)
5E14 101F JMP >5E54
5E16 0007 DATA >0007
5E18 0707 SETO R7
5E1A 0707 SETO R7
5E1C 0705 SETO R5
5E1E 0205<LI R5,>0500
5E22 0507 NEG R7
5E24 0507 NEG R7
5E26 0502 NEG R2
5E28 0507 NEG R7
5E2A 0207 LI R7,>0007
5E2E 0005 DATA >0005
5E30 0205 LI R5,>0500
5E34 0500 NEG R0
5E36 0700 SETO R0
5E38 0002 DATA >0002
5E3A 0007 DATA >0007
5E3C 0207 LI R7,>0502
5E40 0500 NEG R0
5E42 0200 LI R0,>0000
5E46 0000 DATA >0000
5E48 0000 DATA >0000
5E4A 0000 DATA >0000
5E4C 0000 DATA >0000
5E4E 0000 DATA >0000
5E50 0000 DATA >0000
5E52 0000 DATA >0000
5E54 0201 LI R1,>00BF
5E58 CA81 MOV R1,@>0046(R10)
5E5C 1000 NOP
```

16-5
Check if in GRAPH mode

Store old unit flag
Set to unit 4: UNIT-1.

Line feed

Enable graphics routines

Set X=0

Store old cursor position

Clear data store for colour

Set cursor to X,Y

Get colour

Store in R2

If R2 ≠ 0

Then get current background color.

Put data in LSB of R2

Get appropriate pixel data.

Pixel data to MSB of R5, R6, R7:

Cursor = X+1, Y

Same as SEB4 to SECE

Pixel data to top 3 bits of LSB

Same as SECA to SEDA
LEARN FROM DATA FOR PRINTING.

DATA TO PRINTER

\[ y = y - 1 \]

IF \( y \neq -1 \) THEN LOOP.

PRINT LINE FEED & CR.

SET \( y = 191 \)

\[ x = x + 2 \]

IF \( x \leq 255 \) THEN LOOP

DISABLE GRAPHICS ON PRINTER

RESET OLD UNIT FLAG AND CANSO1

EXAMPLE OF DUMP OUTPUT

To BASIC.
DUMP

This program produces a screen dump to printer using foreground & background colours only. This is useful for dumping screens of only two colours though a full coloured screen can be dumped to printer though no representation of colour can be seen. Each pixel is mapped to a 2x2 matrix which is either black or white only. Using a smaller matrix has the advantage of speed where no colour representation is required. Also the printout does not require rotation through 90 degrees which a 3x3 representation does.

The notes on the listings should provide all further information as to how the routines work.

```
5E00 020A LI R10, >0458
5E04 06BA BL R10
5E06 C26E MOV R11,R10
5E08 022E AI R11, >0019
5E0C CABB MOV R11, >0116(R10)
5E10 1007 JMP >5E20
5E12 0000 DATA >0000
5E14 0000 DATA >0000
5E16 0000 DATA >0000
5E18 0000 DATA >0000
5E1A 0000 DATA >0000
5E1C 0000 DATA >0000
5E1E 1000 NOP
5E20 04EA CLR @0010(R10)
5E24 0201 LI R1, >EE95
5E28 C801 MOV R1, @01D12
5E2C C600 MOV @0548, R1
5E30 0241 ANDI R1, @0F00
5E34 C1B1 MOV R1, >0016(R10)
5E38 C600 MOV @0026, R1
5E3C 1602 JNE >5E42
5E3E 2FA0 XOP @0030, 14
5E42 CAA0 MOV @001E, >0000C(R10)
5E48 0201 LI R1, >0008
5E4C C801 MOV R1, >001E
5E50 0201 LI R1, >0A00
5E54 0F01 WRIT R1
5E56 0201 LI R1, >0D00
5E5A 0F01 WRIT R1
5E5C 0201 LI R1, >1B00
5E60 0F01 WRIT R1
5E62 0201 LI R1, >2500
5E66 0F01 WRIT R1
5E68 0201 LI R1, >3000
5E6C 0F01 WRIT R1
5E6E 1000 NOP

RELOCATABLE CODE

VARIABLE STORAGE

SET X = 0.
DISABLE COL CORRECTION
STORE CURRENT BACKGROUND COLOUR
CHECK IF IN GRAPH MODE
SAVE OLD UNIT & SET NEW UNIT

SET UP PRINTER
```

16 - 8
SET Y = 0.
STORE OLD CURSOR

CURSOR = X, Y

GET COLOUR -> R5

CURSOR = X, Y+1
COLOUR -> BELOW MSB

COLOUR -> R5
CURSOR = X, Y+2
COLOUR -> BELOW MSB

COLOUR -> R5

REARRANGE & PRINT DATA.

X = X + 1  IF X <= 255 THEN LOOP

PRINT LF & CR

SET X = 0

Y = Y + 3  IF Y <= 191 THEN LOOP

RESET PRINTER

RESTORE UNIT CURSOR & COLOUR CORRECTION
TO BASIC

SUBROUTINE TO GET COLOUR.

IF COLOUR = BACKGROUND R2 = 0
ELSE R2 = 3
If your disk drives do not have double pulsing for 40/80 trk switching, the circuit below can be used. As can be seen from the timing diagram, the TMS 9909 outputs a squarewave of period 2*STEP. The drive steps on the trailing (rising) edge, hence the second pulse must be generated a given time after the trailing edge when the STEP is inactive. The given time is stated in the manufacturers handbook as minimum time between STEP pulses. For Canon's MDD220 this is 3ms (210, 8110 6ms). Monostable, IC1a triggers on the trailing edge of the STEP pulse for a period of 0.45*R1*C1 (~4.5ms for the given components). The falling edge triggers the second monostable, IC1b which gives the second STEP pulse (~1ms for given components). The total period of the monostables must not be greater than STEP (as set up in CONFIG, 10ms in this case) as can be seen from the timing diagram. This second pulse is gated with TRK00 to prevent head crashing (like head banging but without Marillion or Iron Maiden). It is then mixed in IC2a with the original STEP input to give the double pulse o/p. An open collector NAND is used to reduce chip count, hence the pull up resistors. Normal 80trk operation is achieved by holding the CLR i/p's low, preventing the monostables triggering.
Further to John's article on single keys for control in issue twelve page 13. All ASCII control characters, bar 5 are available as single keys (unaffected by CTRL or SHIFT without diodes) by connecting the new key-switches to the un-used matrix positions as listed below. Also listed are the functions of the other spare matrix positions (affected by CNTRL & SHIFT). Xn & Yn refer to the matrix positions as on the circuit diagram. As can be seen SO, SI, DC2, DC3, & DC4 are not available as single keys.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>ASCII</th>
<th>CONTROL</th>
<th>HEX</th>
<th>Used Positions</th>
</tr>
</thead>
<tbody>
<tr>
<td>X0,Y0</td>
<td>NUL Null</td>
<td>^@</td>
<td>&lt;00&gt;</td>
<td></td>
</tr>
<tr>
<td>X0,Y1</td>
<td>SOH Start of Header</td>
<td>^A</td>
<td>&lt;01&gt;</td>
<td></td>
</tr>
<tr>
<td>X0,Y2</td>
<td>STX Start of Text</td>
<td>^B</td>
<td>&lt;02&gt;</td>
<td></td>
</tr>
<tr>
<td>X0,Y3</td>
<td>ETX End of Text</td>
<td>^C</td>
<td>&lt;03&gt;</td>
<td></td>
</tr>
<tr>
<td>X0,Y4</td>
<td>EOT End of Transmission</td>
<td>^D</td>
<td>&lt;04&gt;</td>
<td></td>
</tr>
<tr>
<td>X0,Y6</td>
<td>ENQ Enquiry</td>
<td>^E</td>
<td>&lt;05&gt;</td>
<td>EDIT key</td>
</tr>
<tr>
<td>X0,Y7</td>
<td>ACK Acknowledge</td>
<td>^F</td>
<td>&lt;06&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BEL Bell</td>
<td>^G</td>
<td>&lt;07&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BS Backspace</td>
<td>^H</td>
<td>&lt;08&gt;</td>
<td>← key</td>
</tr>
<tr>
<td></td>
<td>HT Horizontal Tab</td>
<td>^I</td>
<td>&lt;09&gt;</td>
<td>→ key</td>
</tr>
<tr>
<td></td>
<td>LF Line Feed</td>
<td>^J</td>
<td>&lt;0A&gt;</td>
<td>↓ key</td>
</tr>
<tr>
<td></td>
<td>VT Vertical Tab</td>
<td>^K</td>
<td>&lt;0B&gt;</td>
<td>↑ key</td>
</tr>
<tr>
<td></td>
<td>FF Form Feed</td>
<td>^L</td>
<td>&lt;0C&gt;</td>
<td>CLEAR key</td>
</tr>
<tr>
<td></td>
<td>CR Carriage Return</td>
<td>^M</td>
<td>&lt;0D&gt;</td>
<td>RETURN key</td>
</tr>
<tr>
<td></td>
<td>SO Shift Out</td>
<td>^N</td>
<td>&lt;0E&gt;</td>
<td>no single key</td>
</tr>
<tr>
<td></td>
<td>SI Shift In</td>
<td>^O</td>
<td>&lt;0F&gt;</td>
<td>no single key</td>
</tr>
<tr>
<td>X1,Y0</td>
<td>DLE Data Link Escape</td>
<td>^P</td>
<td>&lt;10&gt;</td>
<td></td>
</tr>
<tr>
<td>X0,Y8</td>
<td>DC1 Device Control 1</td>
<td>^Q</td>
<td>&lt;11&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DC2 Device Control 2</td>
<td>^R</td>
<td>&lt;12&gt;</td>
<td>no single key</td>
</tr>
<tr>
<td></td>
<td>DC3 Device Control 3</td>
<td>^S</td>
<td>&lt;13&gt;</td>
<td>no single key</td>
</tr>
<tr>
<td></td>
<td>DC4 Device Control 4</td>
<td>^T</td>
<td>&lt;14&gt;</td>
<td>no single key</td>
</tr>
<tr>
<td>X1,Y5</td>
<td>NAK Negative Acknowledge</td>
<td>^U</td>
<td>&lt;15&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYN idle Synchronise</td>
<td>^V</td>
<td>&lt;16&gt;</td>
<td>INSERT key</td>
</tr>
<tr>
<td></td>
<td>ETB End of Tx’n Block</td>
<td>^W</td>
<td>&lt;17&gt;</td>
<td>DELETE key</td>
</tr>
<tr>
<td>X1,Y8</td>
<td>CAN Cancel</td>
<td>^X</td>
<td>&lt;18&gt;</td>
<td></td>
</tr>
<tr>
<td>X1,Y9</td>
<td>EM End Medium</td>
<td>^Y</td>
<td>&lt;19&gt;</td>
<td></td>
</tr>
<tr>
<td>X1,Y10</td>
<td>SUB Substitute</td>
<td>^Z</td>
<td>&lt;1A&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ESC Escape</td>
<td>^</td>
<td></td>
<td>&lt;1B&gt;</td>
</tr>
<tr>
<td>X2,Y1</td>
<td>FS Form Seperator</td>
<td>\</td>
<td>&lt;1C&gt;</td>
<td></td>
</tr>
<tr>
<td>X2,Y2</td>
<td>GS Group Seperator</td>
<td>^</td>
<td>&lt;1D&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RS Record Seperator</td>
<td>^</td>
<td>&lt;1E&gt;</td>
<td>HOME key</td>
</tr>
<tr>
<td>X2,Y4</td>
<td>US Unit Seperator</td>
<td>_</td>
<td>&lt;1F&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DEL Delete</td>
<td>&lt;7F&gt;</td>
<td></td>
<td>RUBOUT key</td>
</tr>
</tbody>
</table>

Other unused matrix positions are as below, as normal, shift, control.

| Matrix | | | |
|--------| | | |
| X0,Y9  | P @ DLE |
| X0,Y10 | O SI    |
| X1,Y1  | K [ VT  |
| X1,Y2  | L \ FF  |
| X1,Y3  | N \ SO  |
| X1,Y4  | M ] CR  |
| X2,Y5  | < < NUL |
| X2,Y6  | > > NUL |
| X2,Y7  | . . NUL |
| X2,Y9  | . . NUL |
| X3,Y3  | _ DEL US |
MDEX software for the Cortex.

The article about MDEX software in the last newsletter has caused some mis-understanding. Firstly I did not intend to imply that the software was no longer copy-write protected. The whole of the system is still protected by copywrite of John Walker ex Marinchip Systems and Stephen Pelc of M.P.E. My note about the copy-write was just to set a price assuming any royalties payable were low enough for us to pay without having to adjust the price. In fact we have agreed to pay 20% of the selling price to M.P.E. for distribution to the appropriate writers.

The Forth and Nautilus cross compiler systems have not been released to the user group after all. Apparently they got mixed up the pile of disks collected from M.P.E. by mistake. They have now been withdrawn from our list of items for sale. We hope to have a public domain version of Fig-Forth available to run on CDOS format disks as an alternative in the near future.

MDEX software available is as follows:—

MDEX (Marinchip Disk Executive) is a disk operating system similar in some respects to CPM. It was originally developed by Marinchip in the U.S. for computers using the T.I. TMS9900 processor. It has been modified by M.P.E. in England for use on the Cortex.

MDEX CORE :- with Debug monitor, Text editor, Basic £10.00
ASM & LINK :- Assembler and Linker £10.00
SYSGEN :- System generation Kit £10.00
WORD :- Word processor £10.00
MDEX-PDS :- All of the above systems in one package £30.00
SPL :- System programming language £10.00
META :- Compiler generator £10.00
Q BASIC :- Basic compiler £15.00
PASCAL :- Sequential Pascal £10.00
WINDOW :- Full screen text editor £15.00
SPELL :- Spelling checker £10.00

All the above MDEX software is now available from the Cortex User Group at the normal address. All have good documentation, except Pascal which has very little but many references to published books are given.

16.12
In this issue I will describe some of the other commands used in QBASIC, starting with the 'CHAIN' command. The CHAIN statement allows a QBASIC programme to pass control to another programme, it may be another QBASIC programme or to one of the operating system utility programmes. For example:-

CHAIN "WINDOW\+TEXT.FILE" following this statement the QBASIC programme would call the editor 'Window' and open the file called TEXT.FILE ready for editing.

String handling commands are plenty, with commands such as OVERLAY$ which will put part or all of one string into another. Another is the "*" command: i.e. A$="*9" which will put nine blanks into the string A$.

As well as the single line function declaration QBASIC supports a multiple line function, this means that after the function declaration on the first line any number of QBASIC statements may make up the function body. The function is ended by the FEND statement. Below is an extract from the QBASIC manual, the function takes two string arguments, LINE$ & WORD$, & returns an integer equal to the number of occurrences of WORD$ in LINE$.

```qbasic
DEF FN.WORD.COUNT%(LINE$, WORD$)
  I%=0  (Occurrences found)
  K%=1  (Offset into string for search)
  WHILE 1
    J%=MATCH(UCASE$(WORD$)UCASE$(LINE$), K%)
    EXIT IF J%=0
    I%=I%+1
    K%=J%+LEN(WORD$)
  WEND
  FN.WORD.COUNT=I%
FEND
```

The MATCH statement searches LINE$ for the pattern WORD$, UCASE$ converts all lower case characters in a string into upper case.

The above can be used as a subroutine or more important the function can be a subprogramme which would be compiled separately, then linked to the main programme and called simply by the statement:

DUMMERY$=FN.WORD.COUNT%, but more of this later, I would like to finish with the file input/output statements.

OPEN statement

OPEN <expression> [RECL <expression>] AS <expression> [BUFF <exp>]

The OPEN <exp>, exp=the file name
RECL <exp>, exp=record length, if used the file is random access
BUFF <exp>, exp=buffer size, if used the buffer space the file uses can be controlled, if a large buffer is allocated the performance of a programme can be improved by reducing the number of disc accesses,

AS <exp>, exp=file number 1 to 20 Also more then one file may be opened with one OPEN statement. eg:-

16 . 13
OPEN "Z/MYFILE" AS 1, "Z/YOURFILE" AS 2
OPEN "Z/MYFILE" AS 1 BUFF 10
OPEN "Z/MYFILE" RECL 145 AS 1 ranom access

CREATE statement
CREATE "Z/MYFILE,130" AS 1 Will create a file 130*128 bytes

GETFILE statement
Will open the named file if it exists if it does not it will automatically create it safer to use than CREATE which would destroy a previously created file.

READ statement; will read one or more variables from a sequential file
PRINT and PRINT USING statement; will write to a file
IF END test end of file
GET & PUT similar to PRINT & READ but faster

A file may be opened and read then written to at any point in a sequential file or test for the end of a file then add to it. With the NOS operating system (MDEX's big brother-is it available?) records can be locked.

The CALL & ADRS statement is used to envoke a assembly language module, values can be passed to the module from the main pgm and back again.

ie:- CALL SRC(ADR$VALUE%),COUNT%)
The assembly pgm SRC will be called & COUNT% & VALUE% will be passed.
VALUE% it's address will be passed to QBASIC's stack r10 this allows a value to be returned.
COUNT% the value in count% will be put on the stack.

Assembly language example from the QBASIC manual:-

```
 idt "SRC"
dstk r10

src* data regg,src1 BLWP vector for entry

src1 mov 20(r13),r10 load caller's stack pointer
pop r0 pop value of COUNT% into r0
pop r1 pop address of VALUE% into r1
mov *r1,r2 load argument value
src r2 shift r2 by count in r0
mov r2,*r1 store back in VALUE% address
mov r10,20(r13) update QBASIC's stack pointer
rtwp . return to QBASIC

regg bss 32 register workspace

end
```

Next month linking with assembly pgms & linking QBASIC Modules.