Personal computer users often like to try to change the operating system of their machines, however slight the changes may be. This, of course, is a way of personalising the machine and making it more suitable for the user's own particular needs. The modification described here is both elegant and efficient. It improves TM (Tape Monitor) by adding a new function to automatically start programs read from cassette. This function explains the title of this article: 'GET' = load the program, and 'GO' = run it!

P. Barrat

GET & GO

automatic program start for the Junior Computer after loading from cassette by TM

The software given here lets the Junior Computer automatically start programs after transferring them from magnetic tape via the cassette interface and TM to random access memory. The principle is that, during the RDTAPE routine, the return address saved on the stack by the JSR-RDTAPE instruction (executed as soon as the user presses the GET key during TM) is replaced by the *start address* (SA) of the program read from the cassette. After loading, the processor leaves the RDTAPE routine by means of the RTS instruction and finds on the stack, not the address it left in order to execute RDTAPE, but rather the start address of the program it has just read from cassette. Therefore it goes to this address to run the program. This presupposes, of course, that the start address of the block of data transferred to RAM *is also the start address of the program*, and also, that the stack is empty (stack pointer equal to $FF) when the GET key is pressed (executing the RDTAPE routine). This last condition is met when TM is used 'normally' as we will see later.

DUMPB

In order to achieve the desired effect a DUMPB routine has been created. This is simply a modified copy of the DUMP routine of TM and it registers on cassette a heading containing three specific items of data: address $Ø1FE which acts as a load pointer, the start address of the program, which RDTAPE places at addresses $Ø1FE and $Ø1FF – the top of the stack in other words –, and byte $2Ø which RDTAPE will not accept, so it starts RDTAPE anew, normally this time. DUMPB ends by jumping to TM resulting in the DUMP routine being executed normally.
Comparing the listing of table 1 with the listing of DUMP (on page 194 of JC book 4), it is clear that the instructions for initializing CHKL and CHKH, and also for POINT and SA ($ØAØA . . . $ØA19), have been omitted and an instruction to initialize the stack pointer at $Ø73Ø (TXS on the listing in table 1) has been added.
We then see that DUMPB outputs address $Ø1FE to the tape (which RDTAPE considers as a load vector), and then changes the start address of the block of data to be loaded before storing it in its turn on the tape. This correction is needed to ensure that the RTS instruction works properly at the end of RDTAPE. The last character given by DUMPB is $2Ø. The JMP-TM instruction now leads to the normal procedure during which DUMP loads the program from the cassette.

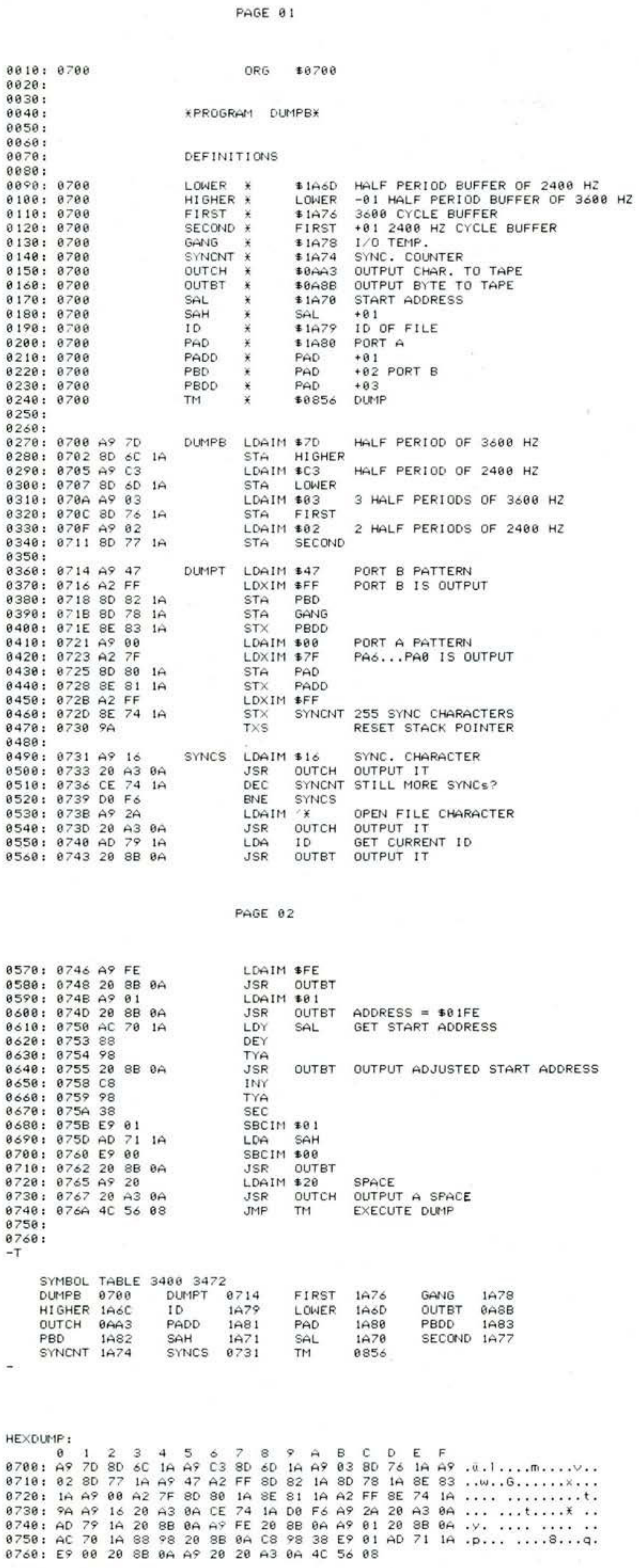
Reading

From the listing of RDTAPE (page 197 of JC book 4) the sequence of operations after loading the heading prepared by DUMPB can easily be followed. Having read the synchronization characters, the start character of the file (*), and the identification number ID, subroutine RDTAPE reads address $Ø1FE as a load vector (POINT). It then immediately loads the two next bytes which it then places in $Ø1FE and $Ø1FF, thus changing its own return address on the stack. The new address is none other than the start address of the program that is to be loaded. The next byte loaded by RDTAPE is the 'space' character ($2Ø). This, however, does not get past the BMI instruction at $ØB73 (page 198 in book 4) so RDTAPE is started again and this time it simply reads the program registered by DUMP after executing DUMPB. At the end of the load, the RTS instruction at $ØB9A leads the processor to look for the return address on the stack.
As we have seen, it finds the start address of the program it has just loaded and it then proceeds to run this.

Using DUMPB

In order to avoid having to modify TM, the author of DUMPB used quite an imaginative solution. The instructions in table 1 should be loaded in memory starting at $Ø7ØØ (or whatever address you like) and the NMI vector ($1A7A, $1A7B) positioned at the start address of DUMPB ($Ø7ØØ in our case).
Then TM is just used normally, except that *the ST/NMI key on the hexadecimal keyboard is now used for the SAVE function with DUMPB.*
Finally we would like to draw your attention to the fact that while using this automatic start, the configuration of the output ports is still that of RDTAPE and not that of the hexadecimal monitor.

Table 1.



```
                                PAGE 01


0010: 0700                    ORG   $0700
0020:
0030:
0040:                 *PROGRAM  DUMPB*
0050:
0060:
0070:                 DEFINITIONS
0080:
0090: 0700            LOWER   *     $1A6D  HALF PERIOD BUFFER OF 2400 HZ
0100: 0700            HIGHER  *     LOWER  -01 HALF PERIOD BUFFER OF 3600 HZ
0110: 0700            FIRST   *     $1A76  3600 CYCLE BUFFER
0120: 0700            SECOND  *     FIRST  +01 2400 HZ CYCLE BUFFER
0130: 0700            GANG    *     $1A78  I/O TEMP.
0140: 0700            SYNCNT  *     $1A74  SYNC. COUNTER
0150: 0700            OUTCH   *     $0AA3  OUTPUT CHAR. TO TAPE
0160: 0700            OUTBT   *     $0A8B  OUTPUT BYTE TO TAPE
0170: 0700            SAL     *     $1A70  START ADDRESS
0180: 0700            SAH     *     SAL    +01
0190: 0700            ID      *     $1A79  ID OF FILE
0200: 0700            PAD     *     $1A80  PORT A
0210: 0700            PADD    *     PAD    +01
0220: 0700            PBD     *     PAD    +02 PORT B
0230: 0700            PBDD    *     PAD    +03
0240: 0700            TM      *     $0856  DUMP
0250:
0260:
0270: 0700 A9 7D      DUMPB   LDAIM $7D    HALF PERIOD OF 3600 HZ
0280: 0702 8D 6C 1A           STA   HIGHER
0290: 0705 A9 C3              LDAIM $C3    HALF PERIOD OF 2400 HZ
0300: 0707 8D 6D 1A           STA   LOWER
0310: 070A A9 03              LDAIM $03    3 HALF PERIODS OF 3600 HZ
0320: 070C 8D 76 1A           STA   FIRST
0330: 070F A9 02              LDAIM $02    2 HALF PERIODS OF 2400 HZ
0340: 0711 8D 77 1A           STA   SECOND
0350:
0360: 0714 A9 47      DUMPT   LDAIM $47    PORT B PATTERN
0370: 0716 A2 FF              LDXIM $FF    PORT B IS OUTPUT
0380: 0718 8D 82 1A           STA   PBD
0390: 071B 8D 78 1A           STA   GANG
0400: 071E 8E 83 1A           STX   PBDD
0410: 0721 A9 00              LDAIM $00    PORT A PATTERN
0420: 0723 A2 7F              LDXIM $7F    PA6...PA0 IS OUTPUT
0430: 0725 8D 80 1A           STA   PAD
0440: 0728 8E 81 1A           STX   PADD
0450: 072B A2 FF              LDXIM $FF
0460: 072D 8E 74 1A           STX   SYNCNT 255 SYNC CHARACTERS
0470: 0730 9A                 TXS          RESET STACK POINTER
0480:
0490: 0731 A9 16      SYNCS   LDAIM $16    SYNC. CHARACTER
0500: 0733 20 A3 0A           JSR   OUTCH  OUTPUT IT
0510: 0736 CE 74 1A           DEC   SYNCNT STILL MORE SYNCs?
0520: 0739 D0 F6              BNE   SYNCS
0530: 073B A9 2A              LDAIM '*     OPEN FILE CHARACTER
0540: 073D 20 A3 0A           JSR   OUTCH  OUTPUT IT
0550: 0740 AD 79 1A           LDA   ID     GET CURRENT ID
0560: 0743 20 8B 0A           JSR   OUTBT  OUTPUT IT


                                PAGE 02


0570: 0746 A9 FE              LDAIM $FE
0580: 0748 20 8B 0A           JSR   OUTBT
0590: 074B A9 01              LDAIM $01
0600: 074D 20 8B 0A           JSR   OUTBT  ADDRESS = $01FE
0610: 0750 AC 70 1A           LDY   SAL    GET START ADDRESS
0620: 0753 88                 DEY
0630: 0754 98                 TYA
0640: 0755 20 8B 0A           JSR   OUTBT  OUTPUT ADJUSTED START ADDRESS
0650: 0758 C8                 INY
0660: 0759 98                 TYA
0670: 075A 38                 SEC
0680: 075B E9 01              SBCIM $01
0690: 075D AD 71 1A           LDA   SAH
0700: 0760 E9 00              SBCIM $00
0710: 0762 20 8B 0A           JSR   OUTBT
0720: 0765 A9 20              LDAIM $20    SPACE
0730: 0767 20 A3 0A           JSR   OUTCH  OUTPUT A SPACE
0740: 076A 4C 56 08           JMP   TM     EXECUTE DUMP
0750:
0760:
-T

   SYMBOL TABLE 3400 3472
   DUMPB  0700     DUMPT  0714     FIRST  1A76     GANG   1A78
   HIGHER 1A6C     ID     1A79     LOWER  1A6D     OUTBT  0A8B
   OUTCH  0AA3     PADD   1A81     PAD    1A80     PBDD   1A83
   PBD    1A82     SAH    1A71     SAL    1A70     SECOND 1A77
   SYNCNT 1A74     SYNCS  0731     TM     0856
-


HEXDUMP:
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
0700: A9 7D 8D 6C 1A A9 C3 8D 6D 1A A9 03 8D 76 1A A9 .ü.l....m....v..
0710: 02 8D 77 1A A9 47 A2 FF 8D 82 1A 8D 78 1A 8E 83 ..w..G.......x...
0720: 1A A9 00 A2 7F 8D 80 1A 8E 81 1A A2 FF 8E 74 1A .... .........t.
0730: 9A A9 16 20 A3 0A CE 74 1A D0 F6 A9 2A 20 A3 0A ... ...t....* ..
0740: AD 79 1A 20 8B 0A A9 FE 20 8B 0A A9 01 20 8B 0A .y. .... .... ..
0750: AC 70 1A 88 98 20 8B 0A C8 98 38 E9 01 AD 71 1A .p... ....8...q.
0760: E9 00 20 8B 0A A9 20 20 A3 0A 4C 56 08
```

Table 1. This short program is all that is needed to make the Junior Computer start programs automatically after loading from cassette by TM (tape monitor).