DEMON

MACHINE CODE MONITOR

COMPUSENSE

Software for Dragon/Tandy

DEMON - a Machine Code Monitor For the DRAGON Computer.

Version 1.1. October 1982

Introduction.

A machine code monitor is designed to allow you, the programmer, to gain control of the essential functions of your computer.

Before using this program you should first read about each command and its function and also make yourself familiar with the 6809 instruction set which is supplied as a reference card with DEMON.

If you are a total beginner to machine code and assembly programming then a good book on the subject of 6809 assembly programming will help you to understand the techniques used in programming the 6809 microprocessor.

This program will not damage your computer. At worst you may have to reset the machine in the event that your machine code programs overwrite system areas by mistake.

To be totally compatible with BASIC, DEMON allocates its own stack and variable areas.

Using DEMON you can write machine code programs for your own use, and even create entire programs, like games, in a form which is difficult for other people to copy.

When you have mastered the intricacies of machine code programming you will also have mastered your computer.

Starting Up

The DEMON Cartridge should be plugged into the expansion port on your DRAGON computer (the socket on the right hand side, facing the keyboard). **CAUTION** - Remember to **power off** the computer when inserting a cartridge!

Power your computer on and the Microsoft BASIC heading and prompt will appear normally.

Now enter the following statement :-

```
EXEC &HC000
```

The DEMON menu will be displayed. You can return to BASIC by typing "Z" If you want to reenter DEMON, simply type EXEC.

Now try the following program :-

```
10 EXEC &HC000
20 PRINT "I'M BACK IN BASIC"
30 INPUT "DO IT AGAIN";Y
40 IF Y > 0 THEN 10
```

Type RUN, and you find that it is possible to successively reenter DEMON from BASIC and return to the calling program above. While in DEMON you can experiment with the commands which are described more fully on another page but remember that if you alter any memory locations in the reserved areas of the computer then the "Z" command may not function correctly. If this does happen, then simply reset the computer and BASIC will recover normally.

Remember that DEMON is a powerful tool and should be treated carefully until you are fully conversant with its features.

DEMON Command Summary

Introduction.

DEMON contains mechanisms for changing memory locations and the register contents, and for execution of programs from a given address. In order to set up register contents, DEMON maintains a set of pseudo-register locations which are loaded into the real registers when the programmer starts to execute a program. In this way it is possible to preset values before program execution begins.

A command - Alter the 6809 Registers

This command allows the pseudo register set to be altered to any values. These registers are loaded into the real 6809 registers when a Jump or Go command is entered.

B Command - Set a Breakpoint

Up to 12 breakpoints may be set. The monitor will replace the byte at the address entered with a SWI instruction and will save the byte in a table. When the program encounters a SWI instruction control is passed to the software interrupt routine within DEMON which will print the current register information. (See X command).

E command - Examine memory contents.

An address is requested . When entered the screen will display memory from that address.

Subsidiary commands :-

Up arrow	- move to previous page.
Down arrow	- move to next page
clear	- toggle between Hex and Ascii display mode. The printable characters are displayed in mode. To revert to Hex display, press clear again.
Break	- return to Monitor menu.

F command - Fill memory Contents.

A start and end address is requested and then a single byte is requested. This byte is used to fill the memory between the two addresses.

G command - Go command.

The pseudo register locations are loaded into the microprocessor and execution begins at the address stored in the program counter. After a software interrupt the program counter points to the next program instruction.

J command - Jump to address.

This command is a convenient way of beginning execution of a program without having to preset the program counter (P register). The effect is to set the P register to the value given and then to begin executing the program at that address.

M command - Modify memory contents.

An address is requested. A screen is then displayed and the cursor is positioned over the byte in question.

Subsidiary commands :-

up arrow	- move cursor up one line
down arrow	- move cursor down one line
Return	- move cursor to next page
clear	- move cursor to previous page
left arrow	- move cursor to next byte
right arrow	- move cursor to previous byte
break	- return to monitor
Any character	- If the character is a valid Hexadecimal

constant (0-9, A-F) it will overwrite the value under the cursor. A non hex value moves the cursor to the next part of the byte.

Q command - Test memory.

This command performs a memeory test between the two specified addresses. Memory tests should not be performed on memory below hex address 0800.

R command - Register dump command.

A screen is displayed which shows the current video page, the breakpoints set, the stack pointer, and the contents of the pseudo register set.

V command - Video page.

The V command allows you to change the text video page to any 200 (hex) boundary throughout memory. However - do not forget that certain areas are either reserved by BASIC or DEMON and other areas may not contain RAM. If you find that the screen displays garbage when you use this command, do not panic, just enter Z and the monitor will return you to BASIC as long as the return vectors have not been overwritten. Study the memory map supplied to see where video pages can be located. The default video page address is 0600 (hex) and the default BASIC video page address is 0400 (hex).

X command - Clear Breakpoints.

This command will clear the existing breakpoints and will also restore the bytes at the breakpoint addresses to the values that they held initially. It is very important to remember to clear breakpoints, the program contents cannot be guaranteed if they are not cleared properly.

Z command - Return to BASIC

The Z command restores the BASIC video page and returns to BASIC having restored the stack pointer which is used by BASIC. This command will fail if the BASIC stack area has been corrupted by any means.

User accessible Subroutines in DEMON

DEMON contains a table of addresses pointing to useful subroutines which are documented here and can be used as part of your own programs. The addresses should not be used directly as they may alter in subsequent versions of this monitor, but the table location should be used as an indirect pointer to the subroutine.The programming examples show how to do this.

1. START location C002 hex.

This location contains the address of the restart point of DEMON which will reenter the program from scratch. If DEMON is reentered at this point then all stack and register information is reset and the video page is initialised to hex $0600. CAUTION - The BASIC reentry vector may be corrupted by use of this vector. In most cases it is better to use the NXTCMD entry point.

2. NXTCMD location C004 hex.

This location contains the address of the command processor loop within DEMON. It should be used in general when returning from a program as it will preserve the BASIC reentry point.

3. INCH location C006 hex.

This location contains the address of the input character routine in DEMON. The subroutine will return the ASCII value of the key struck in the A register. All other registers are preserved.

4. INCHE location C008 hex.

This location contains the address of the input with echo subroutine. The subroutine will accept a keyboard input, display it at the next cursor address and returns the value in the A register. All other registers are preserved.

5. INCHEK location C00A hex.

This location contains the address of the keyboard scanning routine. The subroutine checks to see if a key is depressed and will return the ASCII value in the A register. If no key was pressed then the A register will be zero.

6. OUTCH location C00C hex.

This location contains the address of the character output routine in DEMON. This routine will display the character whose ASCII value is in the A register at the cursor address.

7. PDATA location COOE hex.

This location contains the address of a subroutine that will print a text string at the cursor address. In use the X register must be set to the address of the beginning of the string, and characters will be displayed until an EOT character (hex 04) is encountered.

8. PSTRNG location C010 hex.

This subroutine is the same as PDATA except that the output is preceded by an initial carriage return and line feed to begin at the next line.

9. VIDINT location C012 hex.

This subroutine initialises a text screen. The X register is set to the text screen address, and should be a multiple of 0200 (hex). This subroutine will set the cursor to point to the last line of the new text screen and will clear the screen.

10. SETVID location C014 hex.

This location contains the address of a subroutine which will initialise a video page but will not destroy the contents of the screen. The cursor position is set to the last line of the screen. The X register should be set to the video page start address.

11. CLEAR location C016 hex.

This location contains the address of a subroutine which will clear the current video page to spaces without altering the cursor position. This subroutine must be only be called after VIDINT. All registers are preserved.

12. SWI location C018 hex.

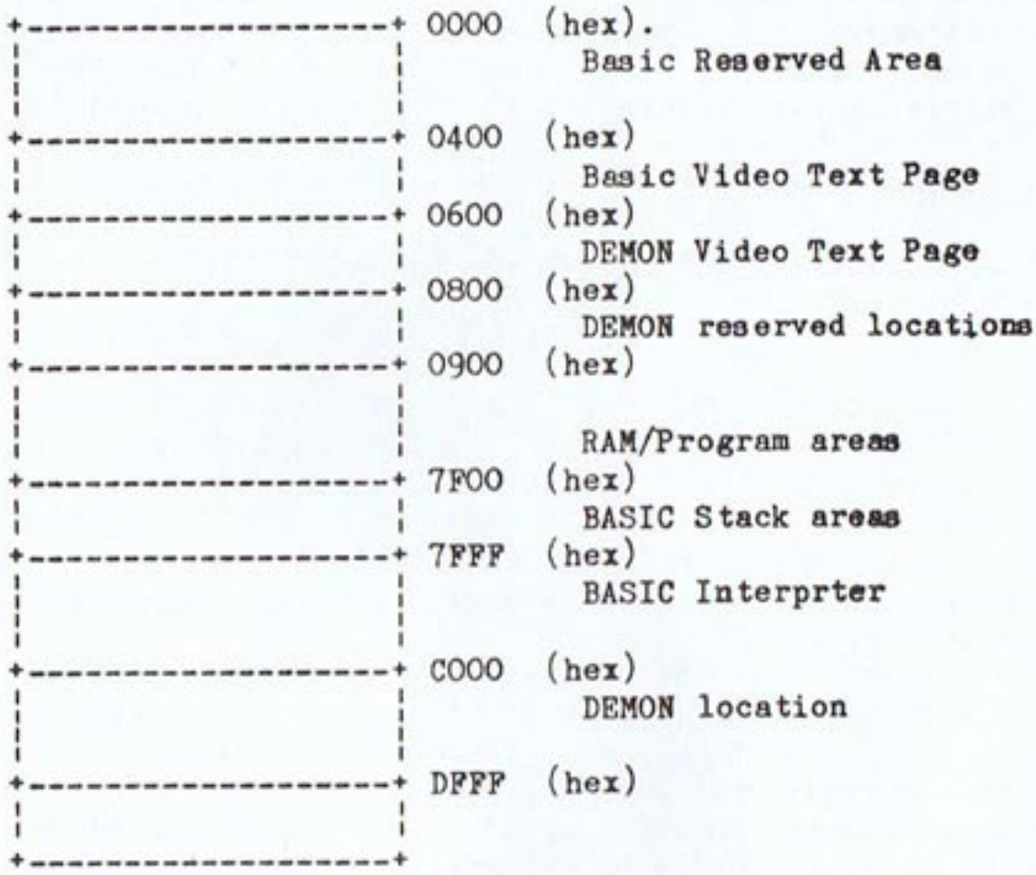
This location contains the address of the Software interrupt service routine in DEMON. This routine will reset the video text page, display register information and return to the DEMON main command loop. It is documented for the assistance of advanced programmers.

13. BASIC location C01A hex.

This location contains the address of the reentry routine which returns control to the the BASIC interpreter.

Memory Map and Variable descriptions

DEMON Memory Map :-



Data location descriptions

1. CHARIN location 0800 (hex) last character accepted from INCH
2. ECHO location 0801 (hex) If non zero, input characters are not echoed
3. SHIFT location 0802 (hex) If non zero, shift lock is on
4. CURSOR location 0803 (hex) Current cursor location
5. PAGE location 0804 (hex) Current text screen page address
6. KEYROW location 0813 (hex) Keyboard Row
7. KEYCOL location 0814 (hex) Keyboard Column
8. SHIFT1 location 0815 (hex) Non zero if shift is pressed
9. NUMUP location C01B (hex) Table of shifted key values

10. SPEC location C03A (hex) Table of special key values

DRAGON Computer Locations

1. PIA1AD location $FF00 (hex) Keyboard PIA Data Register A

2. PIA1AC location $FF01 (hex) Keyboard PIA Control Register A

3. PIA1BD location $FF02 (hex) Keyboard PIA Data Register B

4. PIA1BC location $FF03 (hex) Keyboard PIA Control Register B

5. SAM location $FFC0 (hex) SAM contoller chip location.

Learning About your Computer.

Enter DEMON and use the E command to examine Hex address 0100 . (The command is E 0100)

A screen will be displayed with addresses on the left hand side and data filling the screen. You will notice that the two bytes 0112 and 0113 (hex) are constantly increasing ... this is the location of the BASIC timer.

Now examine address 8000 (hex). Select the Alphanumeric mode by depressing the CLEAR key. You will be able to see the BASIC jump vectors each one of three bytes in the format 7E xxxx and further on there are the BASIC keywords and the related addresses.

Now examine address 0800. This area contains data locations for DEMON. Try depressing some keys and see how the data locations mentioned above are altered. It would be easy to write your own program to combine the row and column values to give different meanings to the keys.

Finally examine address 0880 - the changing area is DEMON's stack area

Next select the Memory change function for address 0400 (hex). (The command is M 0400). Change some bytes at random and then return to BASIC using the Z command. You will find that the text screen has changed to reflect the values you typed in using DEMON.

Loading and saving Tapes

To make DEMON as compatible as possible with the DRAGON computer, no intrinsic tape save and load routines are included with DEMON. It is assumed that the BASIC LOADM and SAVEM commands will be used to load

and save machine code programs which are being created by DEMON. The procedure to load a machine code program is to use the LOADM command in BASIC and then to enter DEMON when the program can be modified and executed using the G command. When testing is complete then DEMON need not be used at all except for its subroutines which are always available even within BASIC programs.

It would be quite easy to write your own control program around DEMON with tape load and restore facilities in BASIC, followed by an EXEC of DEMON.

Using Breakpoints.

One of the most important features of DEMON is its ability to respond to software interrupt (SWI) instructions. Using the breakpoint command it is possible to interrupt a program and examine the registers and memory contents at that time, and then restart the program. Without this ability it is very difficult to write complex programs in machine code or assembly language.

Breakpoints can be used to trap rogue programs before they overwrite sensitive areas of memory. Using the Fill command areas can be protected by writing 3F into them (3F is the code for SWI). If a program jumps into any part of such an area then a software interrupt will result and it will be possible to trace back using the stack and register contents.

After a Breakpoint has occured, a register dump is displayed with the legend "* SOFTWARE INTERRUPT *". At this point it is possible to resume the program by simply typing G, or DEMON commands may be used to modify memory contents, set new breakpoints etc.

Example Programs.

All the example programs are also shown in DASM format. The DASM two pass assembler is the sister cartridge to DEMON.

Example 1. Loading and storing registers

```
                      ORG     $5000   program begins at $5000
86 00                 LDA     #$00    Load A register with #$00
4C           LOOP     INCA            Increment A register
C6 02                 LDB     #$02    Load B register with #$02
8E 1234               LDX     #$1234  Load X register with #$1234
10 8E 5678            LDY     #$5678  Load Y register with #$5678
3F                    SWI             Return to DEMON
7E 5002               JMP     LOOP    Repeat the sequence
```

DASM format.

```
10   CLEAR &H1000,&H4FFF
20   EXEC &HCFFA
30   @START LDA #$00
40   @LOOP  INCA
50   LDB #$02 : LDX #$1234 : LDY #$5678
60   SWI
70   JMP @LOOP
80   END @START
```

This simple program is meant to demonstrate the use of DEMON commands.

First enter the memory examine and change feature by typing M 5000 and then enter the hexadecimal equivalents of the instructions, shown on the left hand side of each instruction, until the entire program is entered.

Now begin running this program by entering the Jump command - J 5000. Demon will respond instantaneously with a software interrupt. If you look at the register contents they will contain the values set above, and the program counter will be set to the location after the SWI instruction.

Now enter the G command to continue the program at the point where it was interrupted. The program will loop to the beginning and again a softare interrupt will occur.

You will notice that the value of the A register has been incremented by one which is what is expected.

If you have the DASM program then the BASIC program shown can be entered and RUN to create this example.

Example 2. Display a message and accept a response.

```
                        ORG     $5000       PROGRAM STARTS AT $5000
AD 9F C016  EX2         JSR     [CLEAR]     CLEAR THE SCREEN
8E 1014     LOOP        LDX     #MSG        X REG POINTS TO THE MESSAGE
AD 9F C010              JSR     [PSTRNG]    PRINT THE MESSAGE
AD 9F C008              JSR     [INCHE]     ACCEPT A CHARACTER
81 59                   CMPA    #'Y         IS IT A Y ?
26 F1                   BNE     LOOP        NO .. TRY AGAIN
3F                      SWI                 YES ... RETURN TO DEMON
54 59 50 45 MSG         FCC     "TYPE Y TO STOP : "
20 59 20 54
4F 20 53 54
4F 50 20 3A
20
04                        FCB     4           TERMINATOR CHARACTER
```

DASM format

```
10 CLEAR &H1000,&H4FFF
20 EXEC &HCFFA
21 @CLEAR  EQU $C016
22 @PSTRNG EQU $C010
23 @INCHE  EQU $C008
30 @EX2 JSR (@CLEAR)
40 @LOOP LDX #MSG : JSR (@PSTRNG) : JSR (@INCHE)
50 CMPA #"Y : BNE @LOOP
60 SWI
70 @MSG FCC "TYPE Y TO STOP"
80 FCB 4
90 END @EX2
```

This example program will print the message and wait for a response. If the character pressed is not 'Y' then the program loops, otherwise it returns to DEMON via the software interrupt.

Note the way that the vectored jumps within DEMON are used by the program. The square bracket notation means that indirect addressing is to be used in standard 6809 assembler notation. That is the address used is that address found at the address specified in the instruction. Indirect addressing is one of the powerful features of the 6809 microprocessor.

Example 3. Display the contents of memory directly on the screen.

```
                    ORG   $5000    PROGRAM STARTS AT ADDRESS 5000
8E   0400 EX3      LDX   #$400     SET X REG. TO START PAGE
AD9F C014 LOOP     JSR   [SETVID]  SWITCH TO THIS PAGE
108E 2000          LDY   #$2000    SET Y REG TO DELAY COUNT
8D0A               BSR   DELAY     DELAY A WHILE
3089 0200          LEAX  $200,X    SET X REGISTER TO NEXT PAGE
8C   4000          CMPX  #$4000    END OF SCREEN SPACE?
26   ED            BNE   LOOP      NO
3F                 SWI             RETURN TO DEMON
313F       DELAY   LEAY  -1,Y      DECREMENT Y REGISTER
26 FC              BNE   DELAY     UNTIL ZERO
39                 RTS             RETURN FROM SUBROUTINE
```

Dasm Format

```
10 CLEAR &H1000,&H4FFF
20 EXEC  &HCFFA
25 @SETVID EQU $C014
30 @EX3  LDY #$0400
40 @LOOP JSR (@SETVID) : LDY #$2000 : BSR @DELAY
50 LEAX $200,X : CMPX #$4000 : BNE @LOOP
60 SWI
70 @DELAY LEAY -1,Y : BNE @DELAY : RTS
80 END @EX3
```

This example shows how to switch text pages without altering the contents. The first page displayed is the BASIC text page, the next page is the DEMON default page and after that each 200 (hex) block of memory is displayed until hex address $4000.

It would be quite feasible to set up an animator program using this sort of technique, and about twenty frames could be displayed in sequence.

Note : On the earlier DRAGON32 computers there is a hardware design fault such that the video display page cannot be assigned above $1FFF. In this case the limit on the X register should be set to $2000.
i.e. CMPX #$2000 in the examples above not CMPX #$4000.