SPEEDSET GS

Let Speedset change the speed of your IIGS for you. This short machine-language utility will set the speed to slow or fast, without the Control Panel.

If you use the Apple IIGS, you know that the new 65816 processor has brought a welcomed speed boost to much of your software. Unfortunately, not all the software can take advantage of this speed. Most of these programs are arcade games.

Since I prefer to leave the default speed at fast, it must be reset whenever I play a game. However, I can never remember to do that until I see the graphic shapes zoom across the screen. If the software disables interrupts, the only way to call up the Control Panel is to reboot the system. And since my preschooler also plays some of these games when I'm not around, I wanted a method of slowing down the processor automatically without changing the default.

Speedset is such a routine — in fact, it's two assembler subroutines, one to set the speed to normal and the other reset it to the fast speed. Each routine is small and needs no input. To combine them into one subroutine would require making the routine more complex than it needs to be. Each routine assumes that the IIGS is running in 6502 emulation; otherwise, native-mode software can set the speed on its own.

USING THE PROGRAM

Speedset is compatible with all Apple II operating systems; however, there should be no need to use it with ProDOS 16. It is also relocatable, and can be added to your collection of assembler routines. Speedset can be called from the HELLO program. Inserting the following command:

```
PRINT CHR$(4);"BRUN SPEEDSET,
A768"
```

into any BASIC program not only loads the program but executes the subroutine SLOW to place the processor into the slower speed. The faster speed can be activated by calling the load address plus 17. Using the above example as a load address, the call would look like this:

```
CALL 768+17
```

The demonstration (**Listing 1**) shows how you can incorporate Speedset into a program and how different processor speeds affect an Applesoft program. Since there may be advantages to switching the speeds within a program, a possible enhancement would be to install Speedset as a ProDOS command. Also, Speedset can be called from other assembler code. You may, however, wish to make some modifications based on your application.

ENTERING THE PROGRAM

I used the ORCA/M assembler because it can assemble 65816 code. It's also compatible with the Apple Programmers Workshop (APW). If you use APW, the 65816 ON at **line 2** is not needed. Also, since APW generates Object Module Format (OMF), the utility Macbin must be run to convert the output to a binary load file if Speedset is to be used under ProDOS 8 or DOS 3.3.

Type in the Applesoft demonstration program in **Listing 1** and save it to disk with the command:

```
SAVE SPEEDSET.DEMO
```

If you have an ORCA/M or APW assembler, type in the source code from **Listing 2** and assemble it, saving the object code as SPEEDSET. If you don't have one of those assemblers, type in the machine code from **Listing 3** and save it to disk with the command:

```
BSAVE SPEEDSET,A$300,L$22
```

HOW THE PROGRAM WORKS

Speedset's mechanics are simple. First, the program must switch between emulation mode and native mode by setting or clearing the Emulation bit. Unfortunately, the only way to gain access to the Emulation bit is through the Carry bit in the status register. The program first clears the Carry bit with a CLC and then transfers the bit to the Emulation bit with the XCE command. By clearing the Emulation bit, we ensure that the processor will be running in native mode. Native mode is necessary to guarantee access to the configuration register, address $C036 in bank $E0.

The next step is to make sure the length of the accumulator is 8 bits. This is done by issuing a SEP with a bit string of 00100000, $20, setting the M bit in the status register. The reason for this becomes obvious when we manipulate the high bit of the byte at $E0:C036 (i.e., location $C036 of bank $E0). Failure to do this could change the byte at $E0:C037.

The LDA <$E0C036 forces long addressing to ensure access to the byte in the correct bank. To set normal speed, set the high bit to zero; conversely, a high bit set to 1 sets the speed to fast. Next, the byte is saved back to the configuration register.

The last thing to do before returning to the caller is to return to emulation mode. This is accomplished by setting the Carry bit and transferring it to the Emulation bit.

Listings for Speed Set GS

LISTING 1: SPEEDSET.DEMO

```
37 | 10  REM  ********************
C0 | 20  REM  * SPEEDSET.DEMO      *
B9 | 30  REM  * BY E. OMICCIOLI   *
AE | 40  REM  * COPYRIGHT(C) 1988 *
CB | 50  REM  * MICROSPARC, INC   *
24 | 60  REM  * CONCORD, MA 01742 *
45 | 70  REM  ********************
47 | 80  HOME : PRINT  CHR$ (4)"BRUN SPEEDSET,A768":
          S = 0
54 | 90  FOR X = 0 TO 3
8E | 100  IF  NOT S THEN  VTAB 23: HTAB 10: PRINT "S
          peed is set to slow": CALL 768
93 | 110  IF S THEN  VTAB 23: HTAB 10: PRINT "Speed
          is set to fast": CALL 768 + 17
7F | 120  GOSUB 150:S =  NOT S
87 | 130  NEXT
7E | 140  END
B5 | 150  FOR I = 1 TO 39: FOR J = 0 TO 100: NEXT
B8 | 160  HTAB I: VTAB 12: PRINT " A";: NEXT
B7 | 170  FOR I = 39 TO 1 STEP  - 1: FOR J = 0 TO 10
          0: NEXT
0C | 180  HTAB I: VTAB 12: PRINT "A ";: NEXT
18 | 190  HTAB 0: VTAB 12: PRINT " ": RETURN
```

TOTAL: C435

END OF LISTING 1

LISTING 2: SPEED.SRC

```
         keep  speedset
         65816 on
*********************************************************** ****
*
*  SPEEDSET.SRC
*
*  By Edward D. Omiccioli, Jr
*  Copyright (c) 1988
*  MicroSPARC, Inc
*  Concord, MA 01742
*
*  ORCA/M Macro Assembler
*
*********************************************************** ****
*
slow     start

;        switch from 6502 emulation to native mode

         clc                  clear carry flag
         xce                  exchange carry with E bit (clear E bit)

         sep   #$20           set 8 bit accumulator
         longa off            tell assembler about it

         lda   >$e0c036       get the configuration register in page E0
         and   #$7f           erase high bit
         sta   >$e0c036       save byte back in configuration register

;        return to 6502 emulation mode

         sec                  set carry flag
         xce                  exchange carry with E bit (set E bit)
;
         rts

;        switch from 6502 emulation to native mode

fast     clc                  clear carry flag
         xce                  exchange carry with E bit (clear E bit)

         sep   #$20           set 8 bit accumulator
         longa off            tell assembler about it

         lda   >$e0c036       get the configuration register in page E0
         ora   #$80           set high bit
         sta   >$e0c036       save byte back in configuration register

;        return to 6502 emulation mode

         sec                  set carry flag
         xce                  exchange carry with E bit (set E bit)
;
         rts
         end
```

END OF LISTING 2

LISTING 3: SPEEDSET

```
Start: 2000  Length: 22

54| 2000:18 FB E2 20 AF 36 C0 E0
2C| 2008:29 7F 8F 36 C0 E0 38 FB
D4| 2010:60 18 FB E2 20 AF 36 C0
A1| 2018:E0 09 80 8F 36 C0 E0 38
36| 2020:FB 60
```

TOTAL: 0DCA

END OF LISTING 3