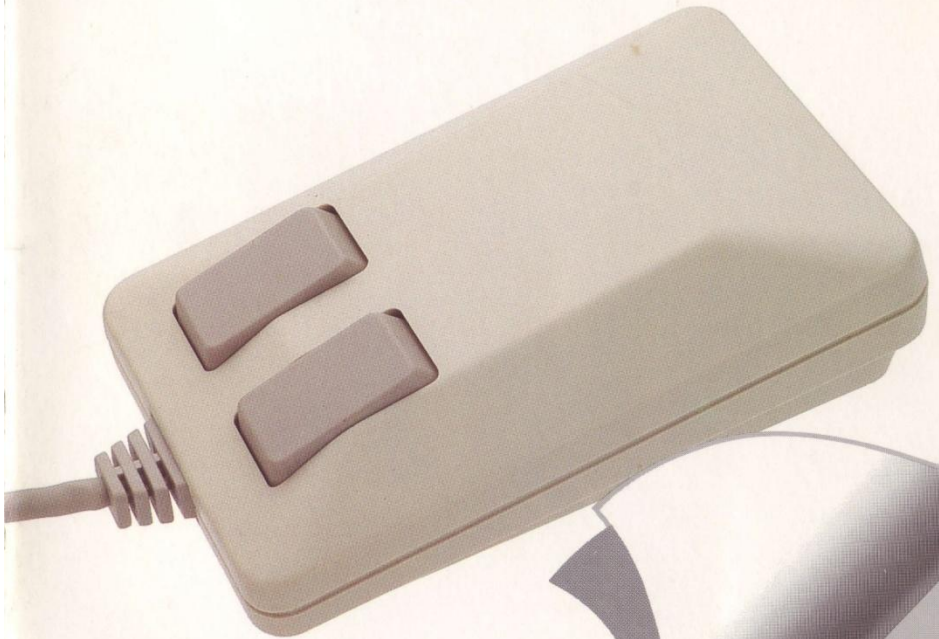
COMMODORE® 1351 MOUSE

user's guide



For Use With Commodore C64,® 64C,™ C128™ Computers

1351 MOUSE USER'S GUIDE

USER'S MANUAL STATEMENT

WARNING:
This equipment has been certified to comply with the limits for a Class B computing device, pursuant to subpart J of Part 15 of the Federal Communications Commission's rules, which are designed to provide reasonable protection against radio and television interference in a residential installation. If not installed properly, in strict accordance with the manufacturer's instructions, it may cause such interference. If you suspect interference, you can test this equipment by turning it off and on. If this equipment does cause interference, correct it by doing any of the following:

- Reorient the receiving antenna or AC plug.
- Change the relative positions of the computer and the receiver.
- Plug the computer into a different outlet so the computer and receiver are on different circuits.

CAUTION: Only peripherals with shield-grounded cables (computer input-output devices, terminals, printers, etc.), certified to comply with Class B limits, can be attached to this computer. Operation with non-certified peripherals is likely to result in communications interference.

Your house AC wall receptacle must be a three-pronged type (AC ground). If not, contact an electrician to install the proper receptacle. If a multi-connector box is used to connect the computer and peripherals to AC, the ground must be common to all units.

If necessary, consult your Commodore dealer or an experienced radio-television technician for additional suggestions. You may find the following FCC booklet helpful: "How to Identify and Resolve Radio-TV Interference Problems." The booklet is available from the U.S. Government Printing Office, Washington, D.C. 20402, stock no. 004-000-00345-4.

First Printing, September 1986

Copyright © 1986 by Commodore Electronics Limited

All rights reserved

ABOUT THIS MANUAL

Basically, this manual is divided into two parts. The first part includes the introduction, mouse cleaning, and tips for general care of the mouse. That part is for the user with mouse-compatible software, who wants simply to plug in the mouse and begin using it. The second part of the manual contains information needed by those who wish to develop software for the mouse.

INTRODUCTION

The Commodore 1351 Mouse™ is a controller designed for use with the Commodore 64® or Commodore 128™ computers. It features two buttons on the top, and a ball on the underside that is rolled upon a flat surface to manipulate onscreen activity.

The mouse has two modes of operation—joystick mode and proportional mode.

In joystick mode, the mouse emulates a joystick and can be used with all joystick-compatible software. In this mode, the left button is the fire button and the right button is usually ignored.

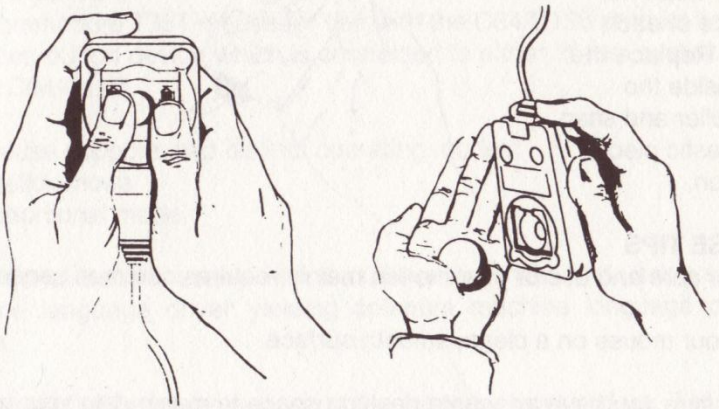
In proportional mode, the mouse uses a new technique to communicate mouse movement to the controlling application software. That requires the software to know the mouse is there and how to read it. For example, the GEOS™ operating system can use many different input drivers. One of them is the Commodore Mouse driver, which can be used with the 1351 in proportional mode.

The 1351 provides proportional mode so that applications can have a fast, responsive pointer that moves easily on the screen. Joystick mode acts as a fallback for those applications that don't have installable device drivers. Therefore, you can use the mouse as a joystick for older software, and take advantage of the benefits provided by proportional mode with newer applications.

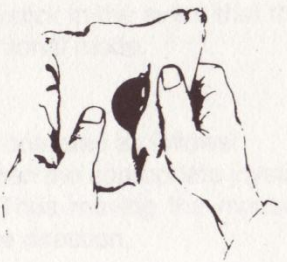
The mouse automatically powers up in proportional mode. To choose joystick mode, plug the mouse into either joystick port on the side of the computer and hold down the right button as the computer is powered up.

MOUSE CLEANING

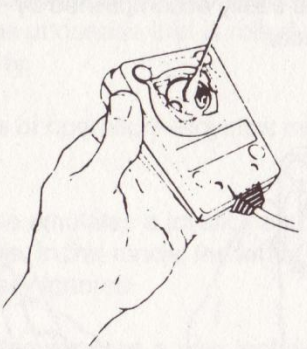
Since the ball of your mouse must roll freely to accurately manipulate the cursor (or whatever) on the screen, it's important that the ball remain free of dirt or debris. This is easily accomplished by sliding out the plastic piece holding the ball in place.



Remove the ball and wipe it off with a soft cloth, such as a handkerchief.



To remove any dirt or dust from the ball area, just blow gently into the opening. Around the top of the opening, there are three metal rollers. To clean these, take a cotton-tipped swab, moistened with head cleaning fluid or alcohol, and gently clean the surface of each roller. Replace the ball inside the controller and snap the plastic piece back on.



MOUSE TIPS

Proper care and use of your mouse mainly requires common sense.

Use your mouse on a clean, smooth surface.

Make sure you have adequate desktop space to manipulate your mouse, so you don't have to constantly pick up and reposition it.

Don't hold the mouse by its cord, or let the body of the mouse hang off the table.

PROPORTIONAL MOUSE DEVELOPER'S GUIDE

This section explains the theory of operation of the Commodore 1351 mouse and suggests software strategies for interfacing to it.

INTRODUCTION

The Commodore 1351 mouse for use with the C64/C128 product line is a small two-button device which is connected to either of the joystick ports on the C64/C128.

The mouse supports two distinct operating modes:

1) Joystick mode.
2) Proportional mode.

Proportional mode is usable with the C64 or the C128, and uses a special machine language driver yielding optimum machine language performance.

Mode selection is determined when the mouse is powered up. If the user depresses the right mouse button when the device is powered up, then the mouse will be in joystick mode.

If the user does not depress the right mouse button when the device is powered up, then the mouse will default to proportional mode.

It is the intent of joystick mode to provide a mode of operation where the mouse can be used as a joystick in the event that the software being run does not support the proportional mode.

JOYSTICK MODE

In joystick mode the mouse operates as follows:

1) If the mouse is moved, then the appropriate joystick lines are activated for a period of 20 ms. Thus moving the mouse is like pushing the joystick in the appropriate direction.
2) The left mouse button is mapped to what would be the fire button on a joystick.
3) The right mouse button is mapped into the SID POTX register. If the button is depressed then the SID POTX register will contain a number <$80. If the button is not depressed then SID POTX will contain a number >=$80.
4) See the section on SID REGISTER CAUTIONS.

Software interface:
For most applications, the interface for joystick mode of operation shall be just as any joystick driver, and the right button shall be ignored.

PROPORTIONAL MODE
In proportional mode the mouse operates as follows:

1) Mouse movement is tracked internally to the mouse. The position of the mouse MOD 64 is transmitted to the SID POTX and POTY registers every 512 us., requiring no software intervention.

   The POTX register is used to read the X position of the mouse and the POTY register is used to read the Y position of the mouse.

   The register contents are as follows:

Bit Position	7	6	5	4	3	2	1	0
POT Register	X	P5	P4	P3	P2	P1	P0	N

   where:

   X . . . . . is a don't care bit.
   P5-P0 . . . . . is the mouse position MOD 64.
   N . . . . . is a special (noise) bit (keep reading . . .).

2) The left mouse button is mapped to what would be the fire button on a joystick.
3) The right mouse button is mapped to what would be the UP direction on a joystick.

Software interface:

1) Because the left and right buttons appear as joystick lines, reading them from software is a trivial exercise in polling.

   Note that as with a joystick, the buttons will interfere with the keyboard map, and software should make some effort to distinguish between a point short in the keyboard matrix (i.e., a key being depressed), and a whole row or column being grounded (i.e., a joystick type of signal).
2) The position information is not difficult to handle. It fits ideally in the 60 hz interrupt routine (preferably at the beginning—see the section on SID REGISTER CAUTIONS).

   The strategy is as follows:
   1) Read the mouse position MOD 64.
   2) Determine if the mouse has moved by comparing the current position with a saved copy of the previous position.
   3) If the mouse has moved, then modify your pointer position appropriately.

The mouse makes an effort to transmit a position to the SID register. Unfortunately, there is a single bit of noise in the transmission.

For example, even if the mouse is still, it is possible for the POT register to vacillate between $80 and $7F. This would result in the mouse position as jittering between two points.

It is therefore necessary to consider the low order bit of the POT register before making any decision as to whether the mouse has moved.

All of this can be seen in the supplied mouse driver code.

SID REGISTER CAUTIONS:

In the C64 & C128, the SID pot lines are connected to both joystick ports. A 4066 analog switch is used to switch the POT lines between the two ports based on one of the keyboard scan lines. The means that the normal keyscan interrupt temporarily affects the values returned in the POT registers. Therefore, in order to perform reliable conversions, the POT lines must be connected to the mouse for a period of >1.6 ms before the value returned in the POT register is valid.

The best way to insure this is to wedge the mouse driver software into the IRQ handler prior to the polled keyscan. This more-or-less assures that the keyscan lines have been sufficiently stable before the POT register is read by the mouse drivers.

BASIC AND MACHINE LANGUAGE PROGRAMS FOR 1351 MOUSE AND C64

```
100 GOSUB140:GOSUB330
110 V=13*4096:POKEV+21,1:POKEV+39,1:POKEV+
    0,100:POKEV+1,100:POKEV+16,0
120 POKE2040,56:SYS12*4096+256
130 END
140 FORX=0TO129:READA$:GOSUB430:POKE49408+X,Y:NEXTX:
    RETURN
150 DATAAD,15,03,C9,C1,F0,19,08
160 DATA78,AD,14,03,8D,00,C0,AD
170 DATA15,03,8D,01,C0,A9,21,8D
180 DATA14,03,A9,C1,8D,15,03,28
190 DATA60,D8,AD,19,D4,AC,02,C0
200 DATA20,58,C1,8C,02,C0,18,6D
210 DATA00,D0,8D,00,D0,8A,69,00
220 DATA29,01,4D,10,D0,8D,10,D0
230 DATAAD,1A,D4,AC,03,C0,20,58
240 DATAC1,8C,03,C0,38,49,FF,6D
250 DATA01,D0,8D,01,D0,6C,00,C0
260 DATA8C,05,C0,8D,04,C0,A2,00
270 DATA38,ED,05,C0,29,7F,C9,40
```

```
280 DATAB0,07,4A,F0,12,AC,04,C0
290 DATA60,09,C0,C9,FF,F0,08,38
300 DATA6A,A2,FF,AC,04,C0,60,A9
310 DATA00,60
320 REM----------
330 FORX=0TO63:READA$:GOSUB430:POKE3584+X,Y:NEXTX:
    RETURN
340 DATAF8,00,00,90,00,00,B8,00
350 DATA00,DC,00,00,8E,00,00,07
360 DATA00,00,02,00,00,00,00,00
370 DATA00,00,00,00,00,00,00,00
380 DATA00,00,00,00,00,00,00,00
390 DATA00,00,00,00,00,00,00,00
400 DATA00,00,00,00,00,00,00,00
410 DATA00,00,00,00,00,00,00,00
420 REM----------
430 Y=1:Y1=0
440 IFLEFT$(A$,1)<>MID$("0123456789ABCDEF",Y,1)
    THENY=Y+1:GOTO440
450 Y1=(Y-1)*16:Y=1
460 IFRIGHT$(A$,1)<>MID$("0123456789ABCDEF",Y,1)
    THENY=Y+1:GOTO460
470 Y=Y1+Y-1:RETURN

READY.
```

```
                        1   ;
                        2   ;       C/64 mouse driver for BASIC 2.0 applications
                        3   ;
                        4
        =0314           5   iirq    = $0314
        =D000           6   vic     = $d000
        =D400           7   sid     = $d400
        =D419           8   potx    = sid+$19
        =D41A           9   poty    = sid+$1a
                       10
        =D000          11   vicdata = $d000          ; vic registers
        =D000          12   xpos    = vicdata+$00   ; low order x position
        =D001          13   ypos    = vicdata+$01   ; y position
        =D010          14   xposmsb = vicdata+$10   ; bit 0 is high order x position
                       15
        =C000          16                 *=$c000
                       17
C000    =C002          18   iirq2          *=*+2
C002    =C003          19   opotx          *=*+1
C003    =C004          20   opoty          *=*+1
C004    =C005          21   newvalue       *=*+1
C005    =C006          22   oldvalue       *=*+1
                       23
        =C100          24            * = $c100
                       25
C100    AD 0315        26   install lda iirq+1
C103    C9 C1          27           cmp #>mirq
C105    F0 19          28           beq 90$
C107    08             29           php
C108    78             30           sei
C109    AD 0314        31           lda iirq
C10C    8D C000        32           sta iirq2
C10F    AD 0315        33           lda iirq+1
C112    8D C001        34           sta iirq2+1
                       35
C115    A9 21          36           lda #<mirq
C117    8D 0314        37           sta iirq
C11A    A9 C1          38           lda #>mirq
C11C    8D 0315        39           sta iirq+1
                       40
C11F    28             41           plp
C120    60             42   90$     rts
```

```
                        43
C121   D8               44   mirq    cld              ; just in case.....
C122   AD D419          45            lda potx         ; get delta values for x
C125   AC C002          46            ldy opotx
C128   20 C158          47            jsr movchk
C12B   8C C002          48            sty opotx
                        49
C12E   18               50            clc              ; modify low order x position
C12F   6D D000          51            adc xpos
C132   8D D000          52            sta xpos
C135   8A               53            txa
C136   69 00            54            adc #$00
C138   29 01            55            and #%00000001
C13A   4D D010          56            eor xposmsb
C13D   8D D010          57            sta xposmsb
                        58
C140   AD D41A          59            lda poty         ; get delta value for y
C143   AC C003          60            ldy opoty
C146   20 C158          61            jsr movchk
C149   8C C003          62            sty opoty
                        63
C14C   38               64            sec              ; modify y position ( decrease y for increase in pot )
C14D   49 FF            65            eor #$ff
C14F   6D D001          66            adc ypos
C152   8D D001          67            sta ypos
                        68
C155   6C C000          69   90$      jmp (iirq2)      ; continue w/ irq operation
```

```
C158   8C C005      70
C15B   8D C004      71   ; movchk
C15E   A2 00        72   ;       entry   y = old value of pot register
                    73   ;               a = currrent value of pot register
                    74   ;       exit    y = value to use for old value
                    75   ;               x,a = delta value for position
                    76
C160   38           77   movchk  sty oldvalue     save old &  new values
C161   ED C005      78           sta newvalue
C164   29 7F        79           ldx #0           preload x w/ 0
C166   C9 40        80
C168   B0 07        81           sec              a <=  mod64( new-old )
C16A   4A           82           sbc oldvalue
C16B   F0 12        83           and #%01111111
C16D   AC C004      84           cmp #%01000000   if      > 0
C170   60           85           bcs 50$
                    86           lsr a                    a <= a/2
                    87           beq 80$                  if      <> 0
                    88           ldy newvalue                     y <= newvalue
                    89           rts                              return
C171   09 C0        90
C173   C9 FF        91   50$     ora #%11000000   else    or in high order bits
C175   F0 08        92           cmp #$ff                 if      <> -1
C177   38           93           beq 80$
C178   6A           94           sec                              a <= a/2
C179   A2 FF        95           ror a
C17B   AC C004      96           ldx #$ff                         x <= -1
C17E   60           97           ldy newvalue                     y <= newvalue
                    98           rts                              return
C17F   A9 00        99
C181   60           100  80$     lda #0           a <= 0
                    101          rts              return w/ y = old value
                    102
```

```
                    * * Cross Reference * *

Reference flags  (# = Definition, $ = Write, <BLANK> = Read)

Symbol     Value          References

IIRQ      =0314           5#      26      31      33      37$     39$
IIRQ2      C000           18#     32$     34$     69
INSTALL    C100           26#
MIRQ       C121           27      36      38      44#
MOVCHK     C158           47      61      77#
NEWVALUE   C004           21#     78$     88      97
OLDVALUE   C005           22#     77$     82
OPOTX      C002           19#     46      48$
OPOTY      C003           20#     60      62$
POTX      =D419           8#      45
POTY      =D41A           9#      59
SID       =D400           7#      8       9
VIC       =D000           6#
VICDATA   =D000           11#     12      13      14
XPOS      =D000           12#     51      52$
XPOSMSB   =D010           14#     56      57$
YPOS      =D001           13#     66      67$
```

BASIC AND MACHINE LANGUAGE PROGRAMS FOR 1351 MOUSE AND C128

```
100 GOSUB230:GOSUB420:SYS6144
120 BA=DEC("0A04"):POKE BA,1ORPEEK(BA)
130 SPRITE 1,1,2:MOVSPR 1,100,100
140 GRAPHIC1,1:CHAR 1,8,1,"1351 MOUSE PAINT"
150 DO:IF (JOY(1) AND 128) THEN GOSUB 180
160 IF (JOY(1) AND 1) THEN GRAPHIC 1,1:CHAR 1,8,1, "1351 MOUSE PAINT"
170 LOOP
180 X=RSPPOS(1,0)-25:Y=RSPPOS(1,1)-51:X=-X*(X>0):Y=-Y*(Y>0)
190 LOCATE X,Y: C=1-RDOT(2):DRAW C,X,Y
200 DO:X=RSPPOS(1,0)-25:Y=RSPPOS(1,1)-51:
    X=-X*(X>0):Y=-Y(Y>0)
210 DRAW C TO X,Y:LOOP WHILE JOY(1) AND 128 : RETURN
220 REM----------------------
230 FORX=0TO135:READA$:POKE6144+X,DEC(A$):NEXTX:
    RETURN
240 DATAAD,15,03,C9,18,F0,19,08
250 DATA78,AD,14,03,8D,F0,18,AD
260 DATA15,03,8D,F1,18,A9,21,8D
270 DATA14,03,A9,18,8D,15,03,28
```

```
280 DATA60,D8,AD,7E,11,D0,33,AD
290 DATA19,D4,AC,F2,18,20,5D,18
300 DATA8C,F2,18,18,6D,D6,11,8D
310 DATAD6,11,8A,69,00,29,01,4D
320 DATAE6,11,8D,E6,11,AD,1A,D4
330 DATAAC,F3,18,20,5D,18,8C,F3
340 DATA18,38,49,FF,6D,D7,11,8D
350 DATAD7,11,6C,F0,18,8C,F5,18
360 DATA8D,F4,18,A2,00,38,ED,F5
370 DATA18,29,7F,C9,40,B0,07,4A
380 DATAF0,12,AC,F4,18,60,09,C0
390 DATAC9,FF,F0,08,38,6A,A2,FF
400 DATAAC,F4,18,60,A9,00,60,00
410 REM----------
420 FORX=0TO63:READA$:POKEDEC("0E00")+X,DEC(A$):NEXTX:
    RETURN
430 DATAF8,00,00,90,00,00,B8,00
440 DATA00,DC,00,00,8E,00,00,07
450 DATA00,00,02,00,00,00,00,00
460 DATA00,00,00,00,00,00,00,00
470 DATA00,00,00,00,00,00,00,00
480 DATA00,00,00,00,00,00,00,00
490 DATA00,00,00,00,00,00,00,00
500 DATA00,00,00,00,00,00,00,00
```

```
                        1   ;
                        2   ;       C/128 mouse driver for BASIC 7.0 applications
                        3   ;
                        4
        =0314           5   iirq    = $0314
        =D000           6   vic     = $d000
        =D400           7   sid     = $d400
        =D419           8   potx    = sid+$19
        =D41A           9   poty    = sid+$1a
                       10
        =117E          11   active  = $117e          ; if zero, then move sprite
                       12
        =11D6          13   vicdata = $11d6          ; basic's copy of vic register image
        =11D6          14   xpos    = vicdata+$00    ; low order x position
        =11D7          15   ypos    = vicdata+$01    ; y position
        =11E6          16   xposmsb = vicdata+$10    ; bit 0 is high order x position
                       17
        =18F0          18                 *=$18f0
                       19
18F0    =18F2          20   iirq2          *=*+2
18F2    =18F3          21   opotx          *=*+1
18F3    =18F4          22   opoty          *=*+1
18F4    =18F5          23   newvalue       *=*+1
18F5    =18F6          24   oldvalue       *=*+1
                       25
        =1800          26            * = $1800
                       27
1800    AD 0315        28   install lda iirq+1
1803    C9 18          29           cmp #>mirq
1805    F0 19          30           beq 90$
1807    08             31           php
1808    78             32           sei
1809    AD 0314        33           lda iirq
180C    8D 18F0        34           sta iirq2
180F    AD 0315        35           lda iirq+1
1812    8D 18F1        36           sta iirq2+1
                       37
1815    A9 21          38           lda #<mirq
1817    8D 0314        39           sta iirq
181A    A9 18          40           lda #>mirq
181C    8D 0315        41           sta iirq+1
                       42
181F    28             43           plp
1820    60             44   90$     rts
```

```
                        45
1821   D8               46   mirq    cld              ; just in case.....
1822   AD 117E          47           lda active       ; if    basic is moving sprite
1825   D0 33            48           bne 90$          ;       let basic have it (why not ?)
1827   AD D419          49           lda potx         ; get delta values for x
182A   AC 18F2          50           ldy opotx
182D   20 185D          51           jsr movchk
1830   8C 18F2          52           sty opotx
                        53
1833   18               54           clc              ; modify low order x position
1834   6D 11D6          55           adc xpos
1837   8D 11D6          56           sta xpos
183A   8A               57           txa
183B   69 00            58           adc #$00
183D   29 01            59           and #%00000001
183F   4D 11E6          60           eor xposmsb
1842   8D 11E6          61           sta xposmsb
                        62
1845   AD D41A          63           lda poty         ; get delta value for y
1848   AC 18F3          64           ldy opoty
184B   20 185D          65           jsr movchk
184E   8C 18F3          66           sty opoty
                        67
1851   38               68           sec              ; modify y position ( decrease y for increase in pot )
1852   49 FF            69           eor #$ff
1854   6D 11D7          70           adc ypos
1857   8D 11D7          71           sta ypos
                        72
185A   6C 18F0          73   90$     jmp (iirq2)      ; continue w/ irq operation
```

```
185D   8C 18F5
1860   8D 18F4
1863   A2 00

1865   38
1866   ED 18F5
1869   29 7F
186B   C9 40
186D   B0 07
186F   4A
1870   F0 12
1872   AC 18F4
1875   60

1876   09 C0
1878   C9 FF
187A   F0 08
187C   38
187D   6A
187E   A2 FF
1880   AC 18F4
1883   60

1884   A9 00
1886   60
```

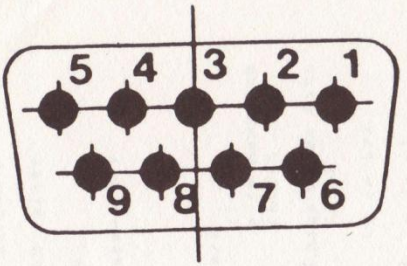
```
74
75   ; movchk
76   ;       entry   y = old value of pot register
77   ;               a = currrent value of pot register
78   ;       exit    y = value to use for old value
79   ;               x,a = delta value for position
80
81   movchk  sty oldvalue    save old &  new values
82           sta newvalue
83           ldx #0         preload x w/ 0
84
85           sec            a <=  mod64( new-old )
86           sbc oldvalue
87           and #%01111111
88           cmp #%01000000 if      > 0
89           bcs 50$
90           lsr a                     a <= a/2
91           beq 80$                   if      <> 0
92           ldy newvalue                      y <= newvalue
93           rts                               return
94
95   50$     ora #%11000000 else    or in high order bits
96           cmp #$ff               if      <> -1
97           beq 80$
98           sec                               a <= a/2
99           ror a
100          ldx #$ff                          x <= -1
101          ldy newvalue                      y <= newvalue
102          rts                               return
103
104  80$     lda #0         a <= 0
105          rts            return w/ y = old value
106
```

* * Cross Reference * *

Reference flags (# = Definition, $ = Write, <BLANK> = Read)

Symbol	Value	References					
ACTIVE	=117E	11#	47				
IIRQ	=0314	5#	28	33	35	39$	41$
IIRQ2	18F0	20#	34$	36$	73		
INSTALL	1800	28#					
MIRQ	1821	29	38	40	46#		
MOVCHK	185D	51	65	81#			
NEWVALUE	18F4	23#	82$	92	101		
OLDVALUE	18F5	24#	81$	86			
OPOTX	18F2	21#	50	52$			
OPOTY	18F3	22#	64	66$			
POTX	=D419	8#	49				
POTY	=D41A	9#	63				
SID	=D400	7#	8	9			
VIC	=D000	6#					
VICDATA	=11D6	13#	14	15	16		
XPOS	=11D6	14#	55	56$			
XPOSMSB	=11E6	16#	60	61$			
YPOS	=11D7	15#	70	71$			

1350 MOUSE PIN-OUT



CONNECTION TABLE		
	FUNCTION	
PIN NO.	JOYSTICK MODE	PROPORTIONAL MODE
1	UP	RIGHT BUTTON
2	DOWN	UNUSED
3	LEFT	UNUSED
4	RIGHT	UNUSED
5	UNUSED	Y-POSITION
6	LEFT BUTTON	LEFT BUTTON
7	+5V	+5V
8	GND	GND
9	RIGHT BUTTON	X-POSITION

Commodore Business Machines, Inc.
1200 Wilson Drive • West Chester, PA 19380

Commodore Business Machines, Limited
3470 Pharmacy Avenue • Agincourt, Ontario, M1W 3G3

312066-01

PRINTED IN USA