

Implementing a Simple Serial Mouse Controller

INTRODUCTION

The mouse is becoming increasingly popular as a standard pointing data entry device. It is no doubt that the demand of the mouse is increasing. Various kinds of mice can be found in the market, including optical mouse, opto-mechanical mouse, and its close relative, trackball. The mouse interfaces to the host via an RS-232 port or a dedicated interface card. Their mechanisms are very similar. The major electrical components of a mouse are:

- Microcontroller
- Photo-transistors
- Infrared emitting diode
- Voltage conversion circuit

The intelligence of the mouse is provided by the microcontroller, hence the features and performance of a mouse is greatly related to the microcontroller used.

This application note describes the implementation of a serial mouse using the PIC16C54. The PIC16C54 is a high speed 8-bit CMOS microcontroller offered by Microchip Technology Inc. It is an ideal candidate for a mouse controller.

THEORY OF OPERATION

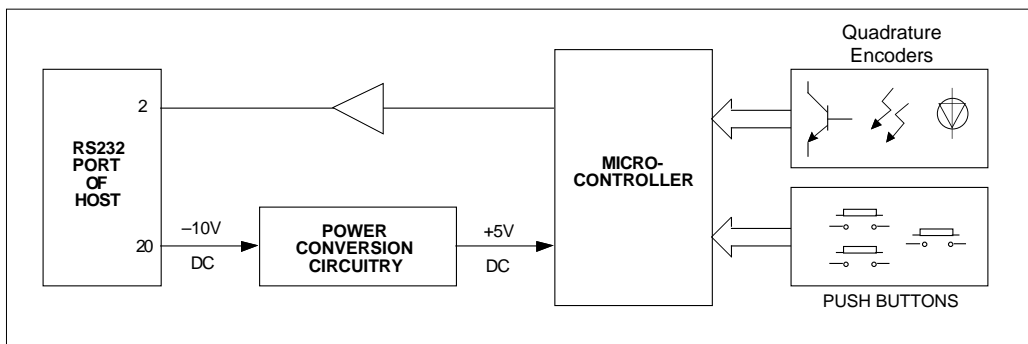
A mouse can be divided into several functional blocks:

- Microcontroller
- Button detection
- Motion detection
- RS-232 signal generation
- 5V DC power supply unit

A typical functional block diagram is shown in Figure 1.

In Figure 2, three push buttons are connected to the input ports of the PIC16C54. When a switch opening or closure is detected, a message is formatted and sent to the host. The X and Y movements are measured by counting the pulses generated by the photo-couplers. In the case of an opto-mechanical mouse, the infrared light emitted by the rotating wheel, so that the pulses are generated on the photo-transistor side. In case of an optical mouse, the infrared light emitted by the infrared diode is reflected off the reflective pad patterned with vertical and horizontal grid lines. It is then received by the photo-transistor in the mouse. When any X or Y movement is detected, a message is formatted and sent to the host.

FIGURE 1 - FUNCTIONAL BLOCKS OF A SERIAL MOUSE



Implementing a Simple Serial Mouse Controller

The Microsoft® Mouse System and the Mouse Systems® device both use serial input techniques. The Mouse System protocol format contains five bytes of data. One byte describes the status of three push buttons, two bytes for the relative X movements and two bytes for the relative Y movements. The Microsoft protocol format contains three bytes of data describing the status of two push buttons and the relative X and Y movements. The details of these protocols are given in Table 1.

Three lines are connected to the host via the RS-232 port:

- Signal Ground
- Received Data
- Request to Send

“Received Data” carries the message sent by the mouse. While “Request to Send” provides a -10V DC for voltage conversion circuitry. A voltage of +5V DC is required for electronic components inside the mouse, however, +5V DC is not part of an RS-232 port, so voltage conversion circuitry is required. This circuit is typically composed of a 555 timer, Zener diodes, and capacitors. An example circuit is shown in Figure 3. Since the current supplied through the RS-232 port is limited to 10 mA, the mouse cannot be designed to consume more than 10 mA current unless an external power supply is provided. The PIC16C54, running at 4 MHz (1 μ s instruction cycle) can provide a very high tracking speed. An 8 MHz version of PIC16C54 is also available if higher performance is desired.

FIGURE 2 - PIC16C54 PIN ASSIGNMENT

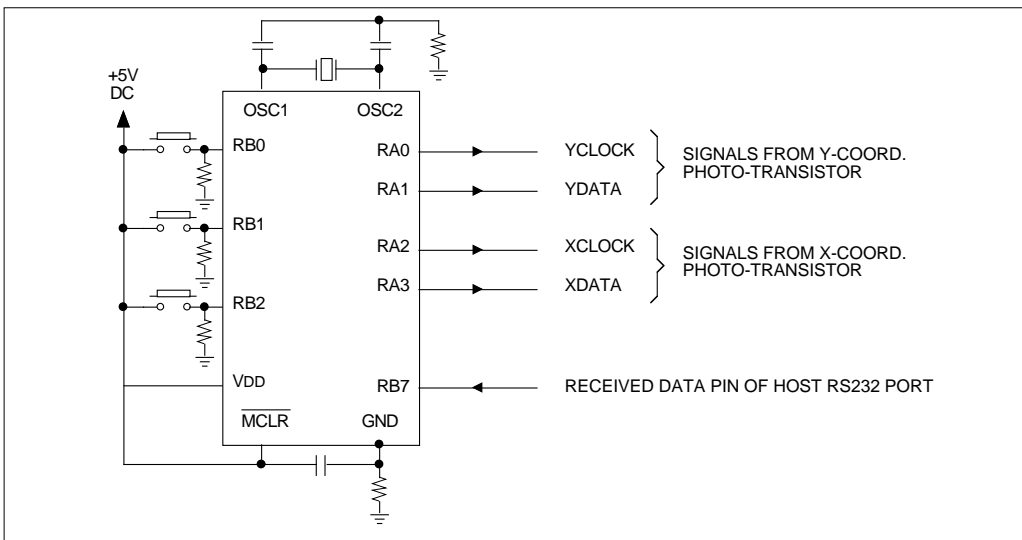
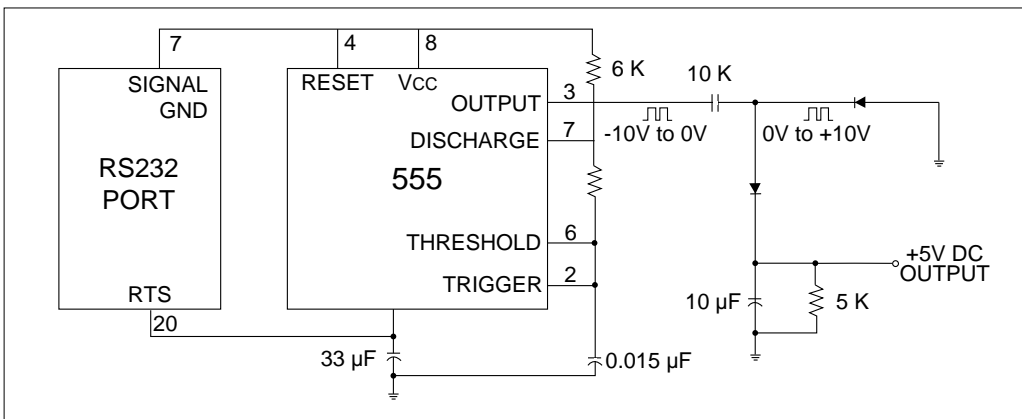


FIGURE 3 - VOLTAGE CONVERSION CIRCUITRY



Implementing a Simple Serial Mouse Controller

ABOUT THE SOFTWARE

The major tasks performed by the software are button scanning, X and Y motion scanning, formatting and sending serial data to the host. These tasks need to be performed in parallel in order to gain better tracking speed. The pulses generated by the photo-couplers are counted while transmitting the serial signals to the RS-232 port. The number of pulses reflects the speed of the movement. The more number of pulses, the faster the movement is.

The directions of the movement are determined by the last states and the present states of the outputs of the photo-transistors. In Figure 4, XCLOCK and XDATA are outputs from the photo-transistors corresponding to the

X-axis movement. XDATA is read when a rising or a falling edge of XCLOCK is detected. For right movement, XDATA is either LOW at the rising edge of XCLOCK or HIGH at the falling edge of XCLOCK. The up and down movement detections follow the same logic. In Table 1, X7:X0 are data for relative movement. If X is positive, it implies that the mouse is moving to the right. If X is negative, it implies a movement to the left. Similarly, if Y is positive, it indicates that the mouse is moving down and if Y is negative, it indicates that the mouse is moving up. The pulses generated by the photo-couplers are checked before every bit is sent. A bit takes 1/1200 second to send, if the distance between the grid lines is 1 mm, the tracking speed will be up to 1200 mm/second.

FIGURE 4 - VOLTAGE CONVERSION CIRCUITRY

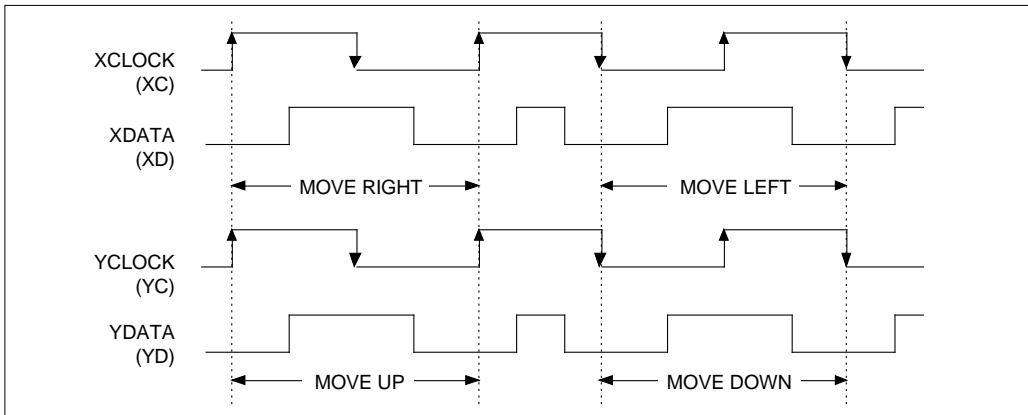


TABLE 1 - MOUSE SYSTEM AND MICROSOFT PROTOCOLS

Bit Position	Mouse System Format*								Microsoft Format*							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Byte 1	1	0	0	0	0	L	M	R	1	1	L	R	Y7	Y6	X7	X6
Byte 2	X7	X6	X5	X4	X3	X2	X1	X0	0	0	X5	X4	X3	X2	X1	X0
Byte 3	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0	0	0	Y5	Y4	Y3	Y2	Y1	Y0
Byte 4	X7	X6	X5	X4	X3	X2	X1	X0								
Byte 5	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0								

* L = Left Key Status 1 = Pressed X7-X0 = X-Axis Movement Data
M = Middle Key Status 0 = Released Y7-Y0 = Y-Axis Movement Data
R = Right Key Status

Implementing a Simple Serial Mouse Controller

The buttons are scanned after a message is sent and the time used to send the message is used as the debouncing time. The message is in an RS-232 format with 1200 baud, eight data bits, no parity, and two stop bits.

The flow charts of the main program, subroutine BYTE and subroutine BIT are shown in Figures 5, 6, and 7. Figure 5 shows the Trigger Flag is set when any change of button status or X/Y movement is detected. Subroutine BYTE is called in the main program five times to send five bytes of information. Subroutine BYTE controls the status of the "Received Data" (RD) pin. If Trigger Flag is clear, RD will always be HIGH. Hence, no message will be sent even when subroutine BYTE is called. Figure 7 shows that subroutine BIT counts the number of pulses from the outputs of the photo-transistors, determines the directions, and generates 1/1200 second delay to get 1200 baud timing.

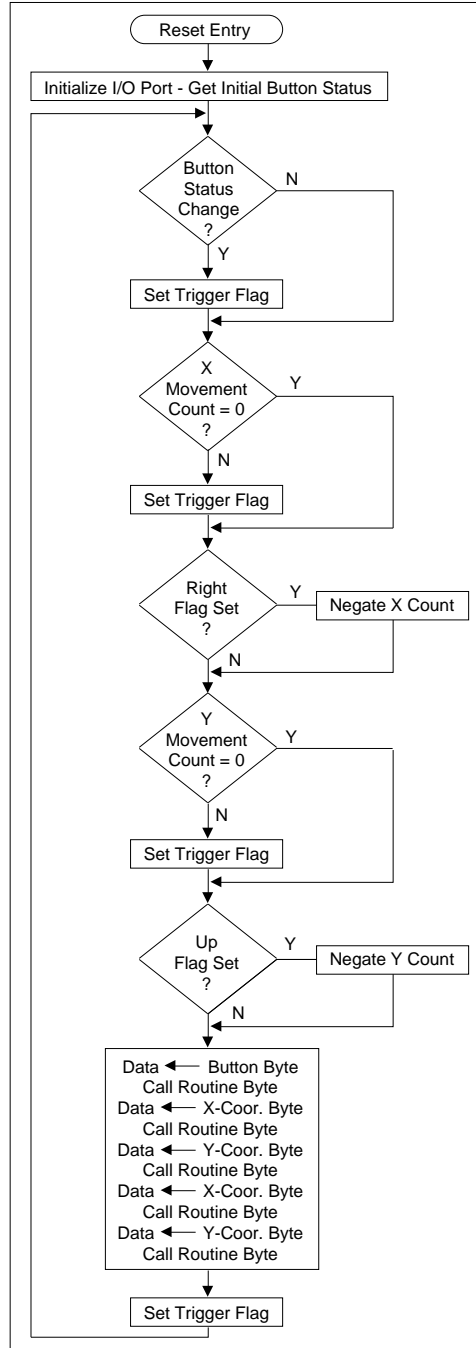
The mouse has been tested in Mouse System Mode and is functioning properly. A completed listing of the source program is given in Appendix A.

SUMMARY

The PIC16C54 from Microchip Technology Inc. provides a very cost-effective, high performance mouse implementation. Its low power (typically < 2 mA at 1 μs instruction cycle), small package (18-pin) and high reliability (on-chip watchdog timer to prevent software hang-ups) are among several reasons why the PIC16C54 is uniquely suitable for mouse applications.

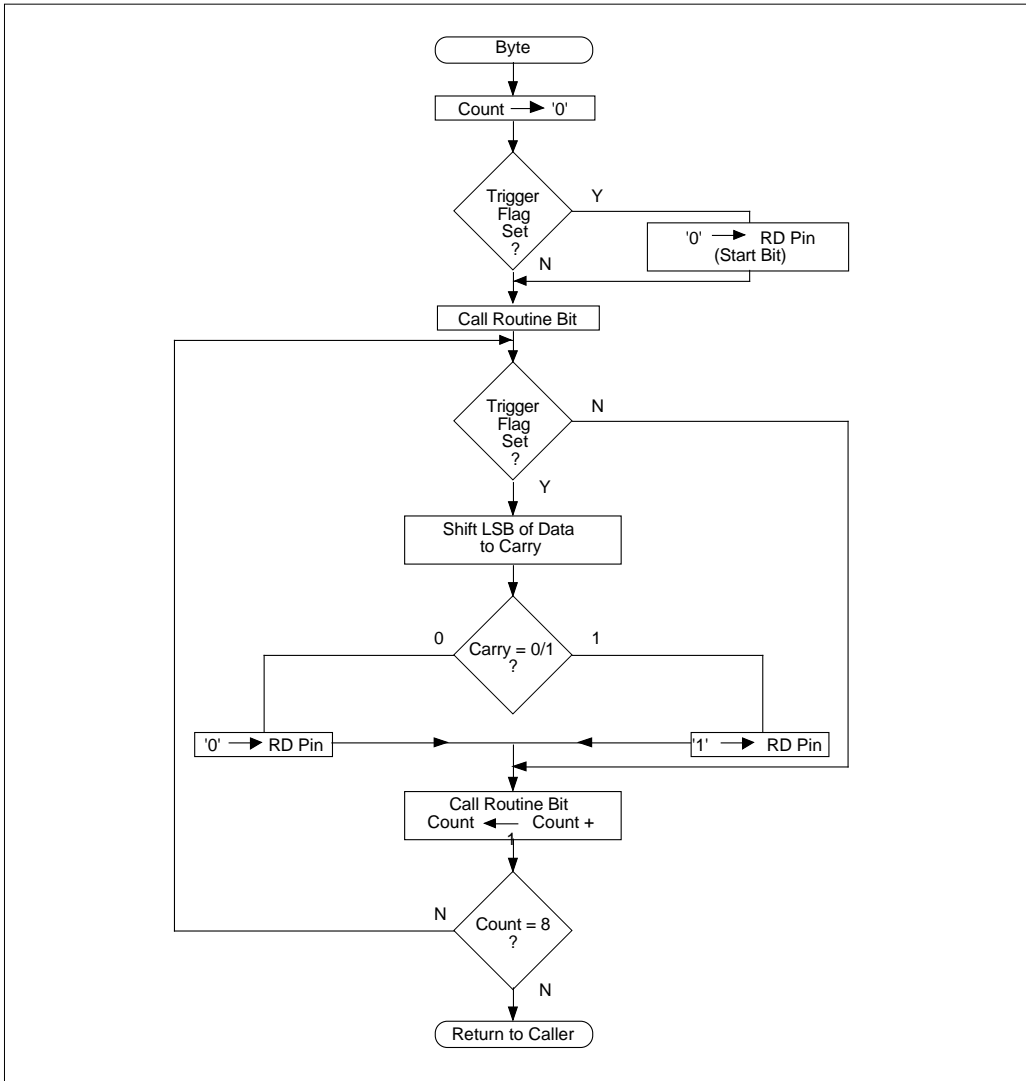
This application note provides the user with a simple, fully functional serial mouse implementation. The user may use this as a starting point for a more comprehensive design. For fully implemented and compliant mouse products see Microchip's ASSP device family (MTA41XXX).

FIGURE 5 - FLOW CHART OF THE MAIN PROGRAM



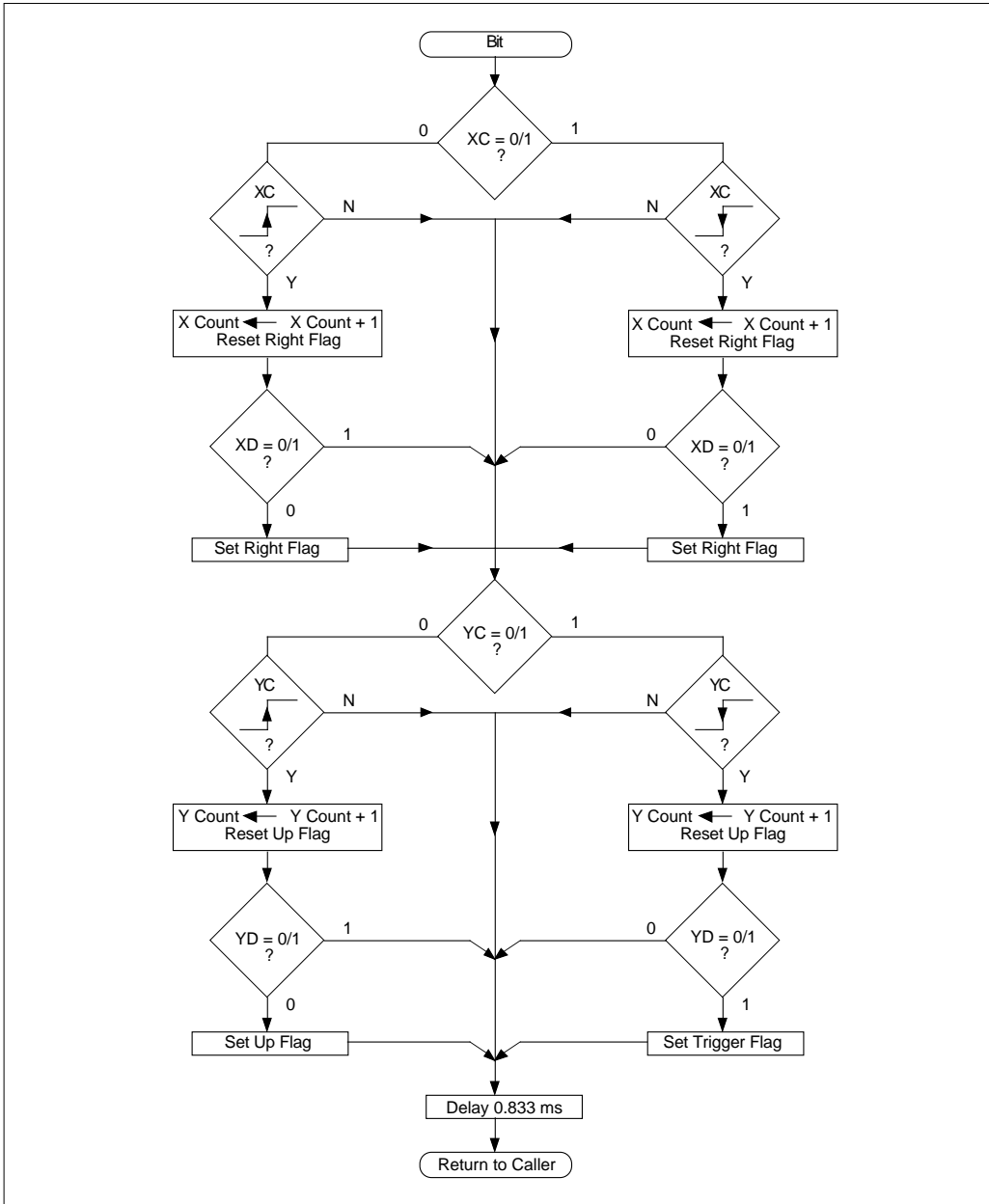
Implementing a Simple Serial Mouse Controller

FIGURE 6 - FLOW CHART OF ROUTINE BYTE



Implementing a Simple Serial Mouse Controller

FIGURE 7 - FLOW CHART OF ROUTINE BIT



Implementing a Simple Serial Mouse Controller

APPENDIX A:

MPASM B0.54
MOUSE

PAGE 1

```
TITLE " MOUSE "
LIST P=16C54,R=0
;
;*****
;*
;* MOUSE CONTROLLER
;*
;* VERSION : 25 APRIL, 1990
;*
;* MODE = PIC16C54XT CLK=4.0MHZ
;*****
;
;-----
; FILES ASSIGNMENT
;-----
;
0003 STATUS EQU 3 ;STATUS REGISTER
0005 RA EQU 5 ;I/O PORT A
0006 RB EQU 6 ;I/O PORT B
0008 TIMER1 EQU 10 ;COUNTER FOR DELAY
000C CSTAT EQU 14 ;CO-ORDINATE STATUS
000D BSTAT EQU 15 ;BUTTON STATUS
000E DATA0 EQU 16 ;
000F DATA1 EQU 17 ;
0010 DATA2 EQU 20 ;5 BYTE RS232 DATA
0011 DATA3 EQU 21 ;
0012 DATA4 EQU 22 ;
0013 FLAGA EQU 23 ;GENERAL PURPOSE FLAG
0014 XCOUNT EQU 24 ;X-MOVEMENT COUNTER
0015 YCOUNT EQU 25 ;Y-MOVEMENT COUNTER
0016 FLAGB EQU 26 ;GENERAL PURPOSE FLAG
0018 COUNT EQU 30 ;GENERAL PURPOSE COUNTER
0019 DATA_AREA EQU 31 ;FOR TEMP. STORAGE
;
;-----
; BIT ASSIGNMENT
;-----
;
0000 YC EQU 0 ;Y-CLOCK PIN
0001 YD EQU 1 ;Y-DATA PIN
0001 UP EQU 1 ;MOVING UP FLAG
0002 XC EQU 2 ;X-CLOCK PIN
0003 XD EQU 3 ;X-DATA PIN
0003 RI EQU 3 ;MOVING RIGHT FLAG
0000 BU1 EQU 0 ;BUTTON #1 PIN
0002 BU2 EQU 2 ;BUTTON #2 PIN
0000 CA EQU 0 ;CARRY FLAG
0007 RD EQU 7 ;RECEIVED DATA PIN TO RS232
0002 ZERO_AREA EQU 2 ;ZERO FLAG
0002 TR EQU 2 ;TIGGER FLAG
;
;=====
; SUBROUTINES
;=====
;
;*****
ORG 0
;*****
;
;=====
; DELAY A BIT TIME AND CHECK XC & YC STATUS
;=====
```

2

Implementing a Simple Serial Mouse Controller

```

BIT
0000 0745      BTFSS   RA,XC          ;XC = 1 ?
0001 0A0A      GOTO    BIT0
0002 064C      BTFSS   CSTAT,XC       ;(XC=1)
0003 0A11      GOTO    BITY          ;(XC ALWAYS = 1)
0004 02B4      INCF    XCOUNT        ;(XC -|_)
0005 0476      BCF     FLAGB,RI        ;DEFAULT LEFT
0006 0765      BTFSS   RA,XD          ;LEFT / RIGHT ?
0007 0A11      GOTO    BITY
0008 0576      BSF     FLAGB,RI
0009 0A11      GOTO    BITY

BIT0
000A 074C      BTFSS   CSTAT,XC       ;(XC=0)
000B 0A11      GOTO    BITY          ;(XC ALWAYS = 0)
000C 02B4      INCF    XCOUNT        ;(XC _|_)
000D 0476      BCF     FLAGB,RI        ;DEFAULT LEFT
000E 0665      BTFSS   RA,XD          ;LEFT / RIGHT ?
000F 0A11      GOTO    BITY
0010 0576      BSF     FLAGB,RI

BITY
0011 0705      BTFSS   RA,YC          ;YC = 1 ?
0012 0A1B      GOTO    BITY0
0013 060C      BTFSS   CSTAT,YC       ;(YC=1)
0014 0A22      GOTO    BITDY          ;(YC ALWAYS = 1)
0015 02B5      INCF    YCOUNT        ;(YC -|_)
0016 0436      BCF     FLAGB,UP        ;DEFAULT DOWN
0017 0725      BTFSS   RA,YD          ;DOWN / UP ?
0018 0A22      GOTO    BITDY
0019 0536      BSF     FLAGB,UP
001A 0A22      GOTO    BITDY

BITY0
001B 070C      BTFSS   CSTAT,YC       ;(YC=0)
001C 0A22      GOTO    BITDY          ;(YC ALWAYS = 0)
001D 02B5      INCF   YCOUNT        ;(YC _|_)
001E 0436      BCF     FLAGB,UP        ;DEFAULT DOWN
001F 0625      BTFSS   RA,YD          ;DOWN / UP ?
0020 0A22      GOTO    BITDY
0021 0536      BSF     FLAGB,UP

BITDY
0022 0205      MOVF    RA,W           ;SAVE COOR. STATUS
0023 002C      MOVWF   CSTAT
0024 0CC1      MOVLW  193D          ;0.833 MS DELAY
0025 0028      MOVWF   TIMER1

BITD0
0026 0000      NOP
0027 02E8      DECFSZ  TIMER1
0028 0A26      GOTO    BITD0
0029 0800      RETLW  0
;
;=====
;
;*****
;*      SUBROUTINE TO SEND A BYTE          *
;*      AS RS232C FORMAT 8,N,1          *
;*****
;
BYTE
002A 0078      CLRFSZ  COUNT          ;RESET 8 BIT COUNT
002B 0753      BTFSS   FLAGA,TR       ;ANY TRIGGER
002C 0A2E      GOTO    BYTE0
002D 04E6      BCF     RB,RD          ;LOW RD FOR START BIT

BYTE0
002E 0900      CALL   BIT

BYTE1
002F 0753      BTFSS   FLAGA,TR       ;ANY TRIGGER ?
0030 0A37      GOTO    BYTE3
0031 0339      RRF     DATA_AREA     ;SHIFT DATA TO CARRY
0032 0703      BTFSS   STATUS,CA      ;0 / 1 ?
0033 0A36      GOTO    BYTE2

```


Implementing a Simple Serial Mouse Controller

```
0034 05E6          BSF    RB,RD          ;SEND A 1
0035 0A37          GOTO   BYTE2
                                BYTE2
0036 04E6          BCF    RB,RD          ;SEND A 0
                                BYTE3
0037 0900          CALL   BIT
0038 02B8          INCF   COUNT
0039 0778          BTFSS  COUNT,3        ;COUNT = 8 ?
003A 0A2F          GOTO   BYTE1
003B 0753          BTFSS  FLAGA,TR       ;ANY TRIGGER ?
003C 0A42          GOTO   BYTE4
003D 04E6          BCF    RB,RD          ;SEND SENT BIT
003E 0900          CALL   BIT
003F 05E6          BSF    RB,RD
0040 0900          CALL   BIT
0041 0A44          GOTO   BYTE5
                                BYTE4
0042 0900          CALL   BIT
0043 0900          CALL   BIT
                                BYTE5
0044 0800          RETLW  0
;
;=====
;          RESET ENTRY
;=====
;
INIT
0045 0CC1          MOVLW  B'11000001'    ;DISABLE WATCH DOG
0046 0002          OPTION
0047 0C0F          MOVLW  B'00001111'    ;INIT RB0~3 BE INPUTS
0048 0006          TRIS  RB             ;RB4~7 BE OUTPUTS
0049 0CFE          MOVLW  B'11111111'    ;INIT RA0~3 BE INPUTS
004A 0005          TRIS  RA
004B 05E6          BSF    RB,RD          ;HIGH RD PIN
004C 0246          COMF   RB,W          ;GET INIT BUTTON INPUTS
004D 0E05          ANDLW  B'00000101'
004E 0D80          IORLW  B'10000000'
004F 002D          MOVWF  BSTAT
0050 002E          MOVWF  DATA0
0051 0205          MOVF   RA,W
0052 002C          MOVWF  CSTAT
0053 0073          CLRF  FLAGA          ;CLEAR TR FLAG
0054 0074          CLRF  XCOUNT       ;RESET XCOUNT & YCOUNT
0055 0075          CLRF  YCOUNT
                                SCAN
0056 006F          CLRF  DATA1          ;UPDATE X,Y MOVEMENT DATA
0057 0070          CLRF  DATA2
0058 0071          CLRF  DATA3
0059 0072          CLRF  DATA4
005A 0214          MOVF   XCOUNT,W        ;XCOUNT = 0 ?
005B 0743          BTFSS  STATUS,ZERO_AREA
005C 0A80          GOTO   WRITX
                                SCANA
005D 0215          MOVF  Y COUNT,W        ;YCOUNT = 0 ?
005E 0743          BTFSS  STATUS,ZERO_AREA
005F 0A92          GOTO   WRITY
                                SCANB
0060 0246          COMF   RB,W          ;BUTTON STATUS CHANGE ?
0061 0E05          ANDLW  B'00000101'
0062 0D80          IORLW  B'10000000'
0063 00AD          SUBWF  BSTAT
0064 0643          BTFSC  STATUS,ZERO_AREA  ;IF CHANGE THEN TRIGGER
0065 0A6B          GOTO   SCANC          ;(NO CHANGE)
0066 0553          BSF    FLAGA,TR       ;(CHANGE) SET TRIGGER FLAG
0067 0246          COMF   RB,W          ;FORMAT BUTTON STATUS DATA
0068 0E05          ANDLW  B'00000101'
0069 0D80          IORLW  B'10000000'
```

Implementing a Simple Serial Mouse Controller

```
006A 002E          MOVWF  DATA0
                   SCANC
006B 0246          COMF   RB,W
006C 0E05          ANDLW  B'00000101'
006D 0D8D          IORLW  B'10000000'
006E 002D          MOVWF  BSTAT
006F 020E          MOVF   DATA0,W           ;SEND DATA0,1,2,3,4 TO HOST
0070 0039          MOVWF  DATA_AREA
0071 092A          CALL  BYTE
0072 020F          MOVF   DATA1,W
0073 0039          MOVWF  DATA_AREA
0074 092A          CALL  BYTE
0075 0210          MOVF   DATA2,W
0076 0039          MOVWF  DATA_AREA
0077 092A          CALL  BYTE
0078 0211          MOVF   DATA3,W
0079 0039          MOVWF  DATA_AREA
007A 092A          CALL  BYTE
007B 0212          MOVF   DATA4,W
007C 0039          MOVWF  DATA_AREA
007D 092A          CALL  BYTE
007E 0453          BCF   FLAGA,TR           ;CLEAR TRIGGER FLAG
007F 0A56          GOTO  SCAN
                   ;
                   WRITX
0080 0553          BSF   FLAGA,TR           ;SET TRIGGER FLAG
0081 0C40          MOVLW 40H               ;IF XCOUNT > 64 THEN XCOUNT <-64
0082 0094          SUBWF XCOUNT,W
0083 0603          BTFSC STATUS,CA
0084 0A8D          GOTO  WRITR
                   WRITS
0085 0776          BTFSS FLAGB,RI           ;LEFT / RIGHT ?
0086 0A90          GOTO  WRITL
0087 0274          COMF  XCOUNT           ;(RIGHT) NEG XCOUNT
0088 0294          INCF  XCOUNT,W
                   WRITA
0089 002F          MOVWF  DATA1
008A 0031          MOVWF  DATA3
008B 0074          CLRFB XCOUNT           ;RESET XCOUNT
008C 0A5D          GOTO  SCANA
                   ;
                   WRITR
008D 0C40          MOVLW 40H               ;XCOUNT <- 64
008E 0034          MOVWF  XCOUNT
008F 0A85          GOTO  WRITS
                   ;
                   WRITL
0090 0214          MOVF   XCOUNT,W           ;(LEFT)
0091 0A89          GOTO  WRITA
                   ;
                   WRITY
0092 0553          BSF   FLAGA,TR           ;SET TRIGGER FLAG
0093 0C40          MOVLW 40H               ;IF YCOUNT > 64 THEN YCOUNT <-64
0094 0095          SUBWF YCOUNT,W
0095 0603          BTFSC STATUS,CA
0096 0A9F          GOTO  WRITV
                   WRITW
0097 0736          BTFSS FLAGB,UP           ;DOWN / UP ?
0098 0AA2          GOTO  WRITD
0099 0275          COMF  YCOUNT           ;(UP) NEG YCOUNT
009A 0295          INCF  YCOUNT,W
                   WRITB
009B 0030          MOVWF  DATA2
009C 0032          MOVWF  DATA4
009D 0075          CLRFB YCOUNT           ;RESET YCOUNT
009E 0A60          GOTO  SCANB
                   ;
```

Implementing a Simple Serial Mouse Controller

```

                                WRITV
009F 0C40                MOVLW  40H                ;YCOUNT <- 64
00A0 0035                MOVWF  YCOUNT
00A1 0A97                GOTO   WRITW
;
WRITD
00A2 0215                MOVF  YCOUNT,W                ; (DOWN)
00A3 0A9B                GOTO  WRITB
;
;=====
;          RESET ENTRY
;=====
;
ORG 777
01FF 0A45                GOTO  INIT                ;JUMP TO PROGRAM STARTING
;
END
;
;*****
```

```
Errors   :    0
Warnings :    0
```

Implementing a Simple Serial Mouse Controller

NOTES:

WORLDWIDE SALES & SERVICE

AMERICAS

Corporate Office

Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602 786-7200 Fax: 602 786-7277
Technical Support: 602 786-7627
Web: <http://www.mchip.com/microhip>

Atlanta

Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770 640-0034 Fax: 770 640-0307

Boston

Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508 480-9990 Fax: 508 480-8575

Chicago

Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 708 285-0071 Fax: 708 285-0075

Dallas

Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton

Microchip Technology Inc.
35 Rockridge Road
Englewood, OH 45322
Tel: 513 832-2543 Fax: 513 832-2841

Los Angeles

Microchip Technology Inc.
18201 Von Karman, Suite 455
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

New York

Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516 273-5305 Fax: 516 273-5335

AMERICAS (continued)

San Jose

Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408 436-7950 Fax: 408 436-7955

ASIA/PACIFIC

Hong Kong

Microchip Technology
Unit No. 3002-3004, Tower 1
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

Korea

Microchip Technology
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku,
Seoul, Korea
Tel: 82 2 554 7200 Fax: 82 2 558 5934

Singapore

Microchip Technology
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65 334 8870 Fax: 65 334 8850

Taiwan

Microchip Technology
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886 2 717 7175 Fax: 886 2 545 0139

EUROPE

United Kingdom

Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44 0 1628 851077 Fax: 44 0 1628 850259

France

Arizona Microchip Technology SARL
2 Rue du Buisson aux Fraises
91300 Massy - France
Tel: 33 1 69 53 63 20 Fax: 33 1 69 30 90 79

Germany

Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 Muenchen, Germany
Tel: 49 89 627 144 0 Fax: 49 89 627 144 44

Italy

Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 039 689 9939 Fax: 39 039 689 9883

JAPAN

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81 45 471 6166 Fax: 81 45 471 6122

9/22/95

All rights reserved. © 1995, Microchip Technology Incorporated, USA.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.
