

```

{
    J O Y S T P
}
{-----}
{ Task           : Demonstrates joystick reading through BIOS. }
{-----}
{ Author        : Michael Tischer }
{ Developed on   : 02/25/91 }
{ Last update    : 02/05/92 }
{*****}

program JOYSTP;

uses DOS, CRT;

{== Type declarations =====}

type JSPOS = record { Describes joystick position }
    x,
    y : integer;
end;

{== Constants =====}

const CRLF = #13#10;

{== Global variables =====}

var jsp      : array [1..2] of JSPOS;    { Current joystick position }
    maxx, maxy,
    minx, miny,
    x, y,
    xold, yold : integer;
    curstick   : byte;
    j1but,
    j2but      : array[1..2] of byte;    { Button 1 of joysticks 1 and 2 }
    xfactor,
    yfactor    : real;
    { Button 2 of joysticks 1 and 2 }
    { Coordinate factors for X and Y }

{*****}
* GetJoyButton: Returns joystick button status.
*-----*
* Input: J1B1 = 1 if button 1 (stick 1) depressed, otherwise 0
*         J1B2 = 1 if button 2 (stick 1) depressed, otherwise 0
*         J2B1 = 1 if button 1 (stick 2) depressed, otherwise 0
*         J2B2 = 1 if button 2 (stick 2) depressed, otherwise 0
*-----*
{*****}

procedure GetJoyButton( var j1b1, j1b2, j2b1, j2b2 : byte );

var Regs : Registers;          { Processor registers for interrupt call }

begin
    Regs.ah := $84;              { Function 84H }
    Regs.dx := 0;                { Sub-function 00H }
    intr( $15, Regs );           { Call interrupt 15H }
    j1b1 := ( ( Regs.al and 16 ) shr 4 ) xor 1; { Bit 4 of AL = J1B1 }
    j1b2 := ( ( Regs.al and 32 ) shr 5 ) xor 1; { Bit 5 of AL = J1B2 }
    j2b1 := ( ( regs.al and 64 ) shr 6 ) xor 1; { Bit 6 of AL = J2B1 }
    j2b2 := ( ( regs.al and 128 ) shr 7 ) xor 1; { Bit 7 of AL = J2B2 }
end;

{*****}
* GetJoyPos : Gets positions of both joysticks.
*-----*
* Input      : JS1 = Joystick structure for joystick 1
*              JS2 = Joystick structure for joystick 2
*-----*
{*****}

procedure GetJoyPos( var Js1, Js2 : JSPOS );

var Regs : Registers;          { Processor registers for interrupt call }

begin
    Regs.ah := $84;              { Function 84H }
    Regs.dx := 1;                { Sub-function 01h }
    intr( $15, Regs );           { Call interrupt 15H }
    Js1.x := Regs.ax;             { X-position: Joystick 1 }
    Js1.y := Regs.bx;             { Y-position: Joystick 1 }
    Js2.x := Regs.cx;             { X-position: Joystick 2 }
    Js2.y := Regs.dx;             { Y-position: Joystick 2 }
end;

{*****}
*
* MAIN PROGRAM
*
{*****}

```

```

begin
{-- Get maximum joystick positioning -----}

ClrScr;
writeln( 'JOYSTICK POSITION TEST' );
writeln( 'Push the joystick to the upper right, ' +
        CRLF + 'then press one of the two buttons, ' );

repeat
    { Wait for a joystick button }
    GetJoyButton( j1but[1], j2but[1], j1but[2], j2but[2] );
until ( ( j1but[1] or j2but[1] or j1but[2] or j2but[2] ) <> 0 );

if ( j1but[1] or j2but[1] ) <> 0 then    { Which joystick was that? }
    curstick := 1
else
    curstick := 2;

GetJoyPos( jsp[1], jsp[2] );           { Read position }
maxx := jsp[curstick].x;               { Set position }
miny := jsp[curstick].y;

repeat
    { Wait for release of a joystick button }
    GetJoyButton( j1but[1], j2but[1], j1but[2], j2but[2] );
until ( ( j1but[curstick] or j2but[curstick] ) = 0 );

{-- Get minimum joystick positioning -----}

writeln( CRLF + CRLF + 'Push the joystick to the lower left, ' +
        CRLF + 'then press one of the two buttons, ' );

repeat
    { Wait for a joystick button }
    GetJoyButton( j1but[1], j2but[1], j1but[2], j2but[2] );
until ( ( j1but[curstick] or j2but[curstick] ) <> 0 );

GetJoyPos( jsp[1], jsp[2] );           { Read position }
minx := jsp[curstick].x;               { Set position }
maxy := jsp[curstick].y;

xfactor := 80.0 / ( maxx - minx + 1 ); { Compute coordinate factor }
yfactor := 23.0 / ( maxy - miny + 1 ); { using X-axis and Y-axis }

{-- Read joystick, display position until -----}
{-- the user presses both joystick buttons -----}

ClrScr;
GotoXY( 40, 1 );
write( 'JOYSTP - (c) 1991, 92 by Michael Tischer' );
GotoXY( 1, 25 );
write( 'Press both joystick buttons to end the program' );

xold := 0;                             { Set old position }
yold := 0;

repeat
    GetJoyPos( jsp[1], jsp[2] );         { Read position }

    {-- Compute new X-position of the joystick -----}

    x := round(xfactor * ( jsp[curstick].x - minx + 1 ));
    if ( x < 0 ) then
        x := 0;
    if ( x > 79 ) then
        x := 79;

    {-- Compute new Y-position of the joystick -----}

    y := round(yfactor * ( jsp[curstick].y - miny + 1 ));
    if ( y < 0 ) then
        y := 0;
    if ( y > 22 ) then
        y := 22;

    {-- Display new position if position changes -----}

    if ( x <> xold ) or ( y <> yold ) then
        begin
            GotoXY( xold+1, yold+2 );
            write( ' ' );
            GotoXY( x+1, y+2 );
            write( 'X' );
            xold := x;
            yold := y;
        end;

GotoXY( 1, 1 );
write( '(', jsp[curstick].x:3, '/', jsp[curstick].y:3, ')' );

```

```
    GetJoyButton( j1but[1], j2but[1], j1but[2], j2but[2] );  
until ( j1but[curstick] = 1 ) and ( j2but[curstick] = 1 );  
ClrScr;  
GotoXY( 1, 1 );  
writeln( 'End program.' );  
end.
```