

# ZX-PSX

## Play Station Controller Interface board

---

### Technical features

- The male Play Station controller connector for supporting Play Station controller.
- Works with Play Station 1 and 2 controller
- Din, Dout, CS and CLK pin for interfacing with any microcontroller
- All signal pin use 3-pin JST connector for connecting INEX microcontroller board
- Optional 100-mil connector for supporting all signal pins on the PSX Bus.
- Supply voltage +5V from JST connector or PSX bus connector.
- Size is 3 x 4 cm.

### Kit contents

- ZX-PSX board x 1
- JST3AA-8 cable x 4
- Documentation

---

More the article about How to interfacing PSX controller with BASIC Stamp microcontroller available for free-download at

[http://www.parallax.com/html\\_pages/downloads/nvcolumns/Nuts\\_Volts\\_Downloads\\_V4.asp](http://www.parallax.com/html_pages/downloads/nvcolumns/Nuts_Volts_Downloads_V4.asp).

Select the Column #101.

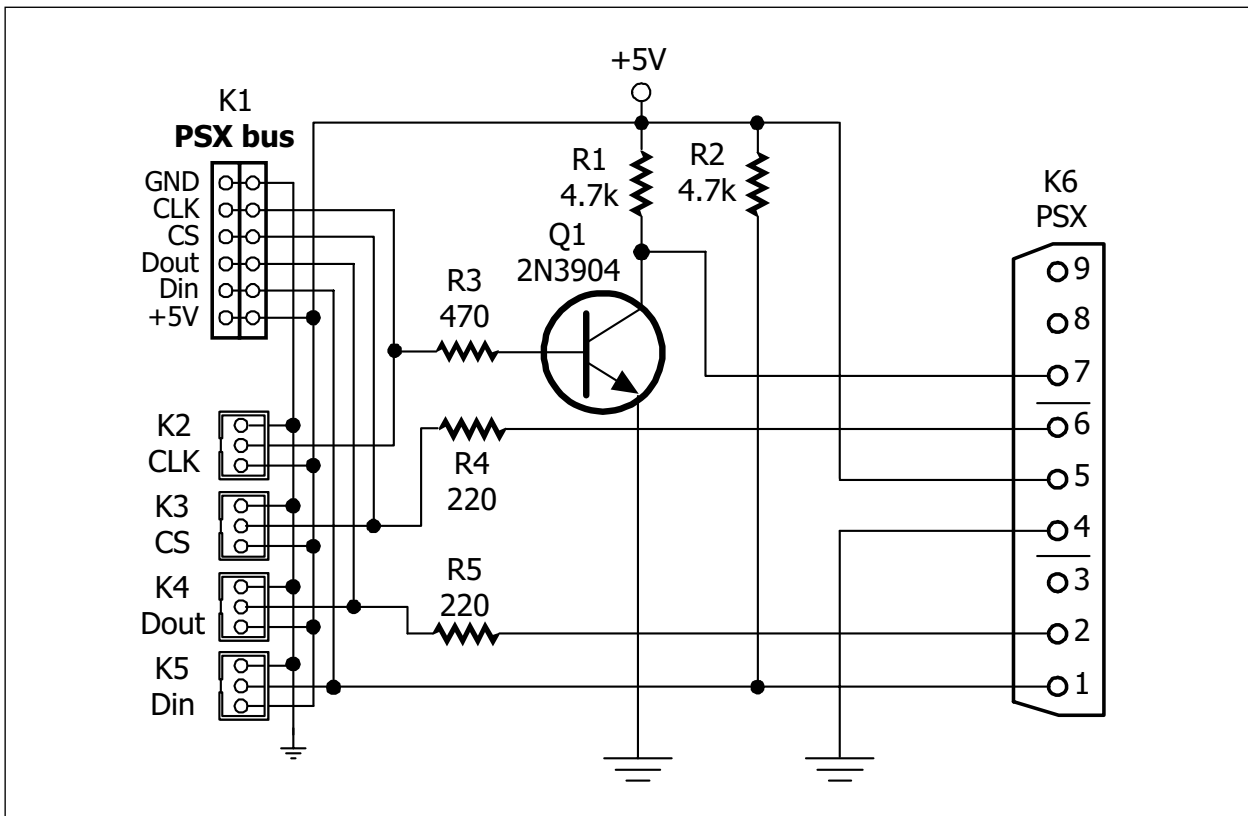


Figure 1 - The ZX-PSX schematic diagram

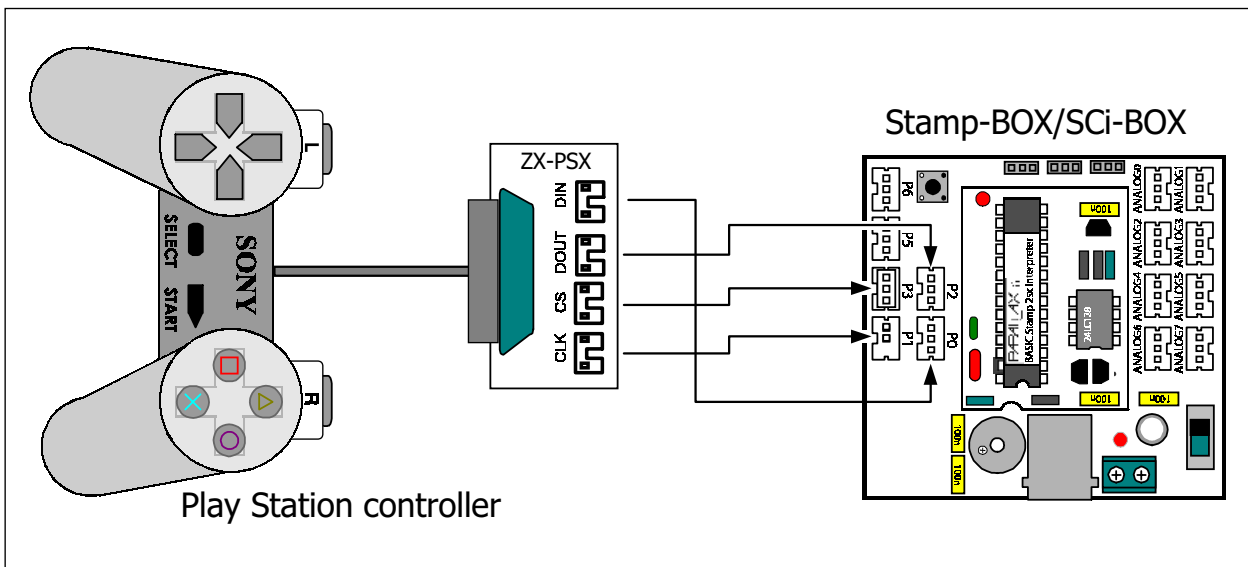


Figure 2 - Interfacing diagram of the ZX-PSX and Stamp-BOX (INEX's BASIC Stamp2SX robotics controller board) or Sci-BX (INEX's BASIC Stamp2SX Science controller board) : *DIN-P0, CLK -P1, DOUT-P2 and CS-P3*

## Example code

```
' This example code is demonstration of reading the PSX controller data to
' showing on the Debug Terminal of BASIC Stamp Editor

' {$STAMP BS2sx}
' {$PBASIC 2.5}
' {$PORT COM1}
PsxAtt      PIN3      ' PSX joystick interface
PsxClk      PIN1
PsxCmd      PIN2
PsxDat      PIN0

' [ Constants ]
Inverted    CON1      ' inverted clock signal
Direct      CON0      ' no inverter in clock line
ClockMode   CONInverted

' [ Variables ]
idx         VARNib    ' loop counter
psxOut      VARByte   ' byte to controller
psxIn       VARByte   ' byte from controller
' joystick packet
psxID       VARByte   ' controller ID
psxThumbL   VARByte   ' left thumb buttons
psxThumbR   VARByte   ' right thumb buttons
psxStatus   VARByte   ' status ($5A)
psxJoyRX    VARByte   ' r joystick - X axis
psxJoyRY    VARByte   ' r joystick - Y axis
psxJoyLX    VARByte   ' l joystick - X axis
psxJoyLY    VARByte   ' l joystick - Y axis

' [Initialization]
Setup:
  HIGH PsxAtt          ' deselect PSX controller
  OUTPUT PsxCmd
  PsxCmd = ~ClockMode  ' release clock
  OUTPUT PsxClk        ' make clock an output

' [ Program Code ]
Main:
DO
  GOSUB Get_PSX_Packet_Fast  ' type and packet
  DEBUG HOME, "Type = "
  IF (psxStatus = $5A) THEN
    DEBUG IHEX2 psxID, " (", IHEX2 psxStatus, ")", CLREOL, CR, CR, BIN8 psxThumbL,
    " ", BIN8 psxThumbR, " "
    IF (psxID <> $41) THEN
      DEBUG DEC3 psxJoyLX, " ", DEC3 psxJoyLY, " ", DEC3 psxJoyRX, " ", DEC3
      psxJoyRY
    ELSE
      DEBUG CLREOL
    ENDIF
  ELSE
    DEBUG "Unknown. No response.", CR, CLRDN
    PAUSE 1000
  ENDIF
  PAUSE 100
LOOP
END
```

to be continue on next pages >>>

#### 4 ● ZX-PSX : Play Station Controller Interface board

```

' [ Routines ]
' This routine REQUIRES inverted clock signal from
' Stamp to PSX controller
Get_PSX_Buttons:
    IF (ClockMode = Direct) THEN Get_PSX_Packet ' redirect if not inverted
    LOW PsxAtt
        SHIFTOUT PsxCmd, PsxClk, LSBFIRST, [$01, $42]
        SHIFTIN PsxDat, PsxClk, LSBPOST, [psxThumbL, psxThumbL, psxThumbR]
        psxId = $41
    HIGH PsxAtt
RETURN
'-----
' This routine manually creates the clock signal,
' so it can be used with a direct (via 220 ohm resistor)
' connection to the clock input

' Execution time on BS2 is ~145 ms.
Get_PSX_Packet:
    LOW PsxAtt ' select controller
        psxOut = $01 : GOSUB PSX_TxRx          ' send "start"
        psxOut = $42 : GOSUB PSX_TxRx          ' send "get data"
        psxId = psxIn ' save controller type
        psxOut = $00 : GOSUB PSX_TxRx
        psxStatus = psxIn ' should be $5A ("ready")
        GOSUB PSX_TxRx : psxThumbL = psxIn      ' get PSX data
        GOSUB PSX_TxRx : psxThumbR = psxIn
        GOSUB PSX_TxRx : psxJoyRX = psxIn
        GOSUB PSX_TxRx : psxJoyRY = psxIn
        GOSUB PSX_TxRx : psxJoyLX = psxIn
        GOSUB PSX_TxRx : psxJoyLY = psxIn
    HIGH PsxAtt ' deselect controller
RETURN
'-----
' Transmit psxOut to, and receive psxIn from the
' PSX controller
PSX_TxRx:
    FOR idx = 0 TO 7
        PsxCmd = psxOut.LOWBIT(idx)          ' setup command bit
        PsxClk = ClockMode                    ' clock the bit
        psxIn.LOWBIT(idx) = PsxDat            ' get data bit
        PsxClk = ~ClockMode                   ' release clock
    NEXT
RETURN
'-----
' This routine combines manual and built-in shifting
' routines to get the best speed and all valid data.
'
' Execution time on BS2 is ~40 ms.
Get_PSX_Packet_Fast:
    IF (ClockMode = Direct) THEN Get_PSX_Packet ' redirect if not inverted
    LOW PsxAtt ' select controller
        SHIFTOUT PsxCmd, PsxClk, LSBFIRST, [$01] ' send "start"
        psxOut = $42 : GOSUB PSX_TxRx ' send "get data"
        psxId = psxIn ' save controller type
        SHIFTIN PsxDat, PsxClk, LSBPOST, [psxStatus] ' should be $5A ("ready")
        SHIFTIN PsxDat, PsxClk, LSBPOST, [psxThumbL]
        SHIFTIN PsxDat, PsxClk, LSBPOST, [psxThumbR]
        SHIFTIN PsxDat, PsxClk, LSBPOST, [psxJoyRX]
        SHIFTIN PsxDat, PsxClk, LSBPOST, [psxJoyRY]
        SHIFTIN PsxDat, PsxClk, LSBPOST, [psxJoyLX]
        GOSUB PSX_TxRx : psxJoyLY = psxIn
    HIGH PsxAtt ' deselect controller
RETURN

```