ZX-SERVO16
16-Ch. Serial Servo Controller

Features :

l Runtime Selectable Baud rate. 2400 to 38k4 Baud.

l 16 Servos. All servos driven simultaneously all of the time.

l 180 degrees of rotation.

l Servo Ramping. 63 ramp rates (0.75 - 60 seconds) allows the user to set the speed of each servo on a per-move basis. You may choose one of 63 ramp rates for each servo. The ramping function allows you to individually set the speed of each servo. With ramping, you can tell the servo where to go, and just how fast to get there. The result is true, "set-it and forget-it" functionality.

l Speed range for 0 - 180 degrees of rotation is 0.75 second to 60 seconds.

l Position Reporting. User may request position of an individual servo at any time. This command allows you to request the position of a servo channel, be it stationary or on the move.

l Network Ready. Two modules may be linked together to drive 32 servos at the same time. It is possible to network two ZX-SERVO16 boards together to control up to 32 servos. Simply use a second 3-Conductor cable to daisy chain two ZX-SERVO16 boards together as shown in the documentation. The presence of a shunt differentiates between Unit 0 (channels 0-15) and Unit 1 (channels 16-31).

l Enhanced Resolution. Use of 16-bit PWM timers enables 0 - 180 degree servo rotation at 2 microseconds per step.

l Serial Command Format. ZX-SERVO16 supports several commands that are sent to it via RS-232 serial protocol. The voltage swing of this serial line is 0-5 VDC (TTL level). Each serial command must be preceded with an exclamation point, "!", and the pair of letters, "SC".

Packing List

l ZX-SERVO16 board

l Documentation and CD-ROM

l CX-4 custom serial port cable

l 2 of PCB3A-8 interface cable

Before You Begin

The processor used on the ZX-SERVO16 requires a supply +5 Vdc. Use a separate power supply for your servos. In general, servos require 4-7.5 VDC. Be sure that the servo power source can supply ample current at the proper voltage and will not damage the servos.

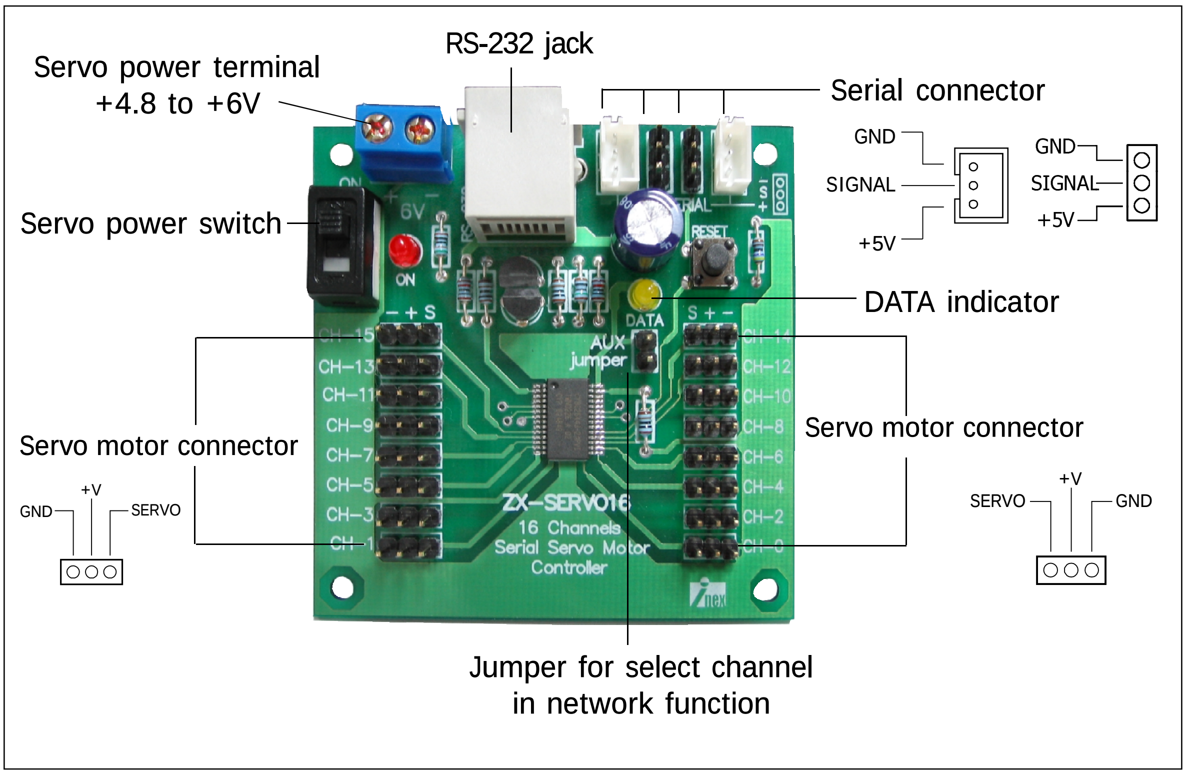


Figure 1 Layout of ZX-SERVO16 board

1. Step of using

(1) Turn-off Servo power switch

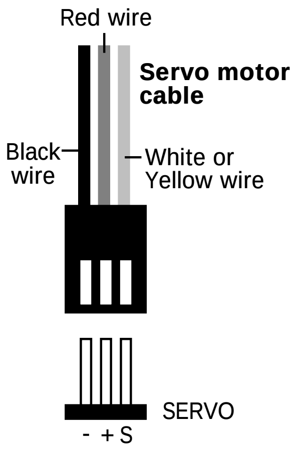
(2) Connect Servo power supply +4.8 to +6V to Servo power terminal. Becareful the supply polarity nust to correct connection.

(3) Plug servo cable to Servo motor connector on ZX-SERVO16 board. Must to plug in correct direction. See the figure below.

Black wire is Negative supply (-) or Ground

Red wire is Positive supply (+)

White or Yellow wire is signal (S)



(4) Connect the serial cable from any microcontroller board to Serial connector on the ZX-SERVO16 board. In ZX-SERVO16 board package bundle the custom PCB3A-8 cable. User can use this cable to connect INEX robot controller board.

(5) Apply the supply voltage +5V to the microcontrolelr board that control the ZX-SERVO16. The +5V will sent to the ZX-SERVO16 board too. "ON" LED on ZX-SERVO16 board turn-on

(6) Write the program to interface and control servos.

Note : In case control via computer, see the Computer control topic in next.

2. Interface programming with ZX-SERVO16 board

2.1 Serial Command Format

The ZX-SERVO16 supports several commands that are sent to it via RS-232 serial protocol. The voltage swing of this serial line is 0-5 VDC (TTL level). Each serial command must be preceded with an exclamation point, "!", and the pair of letters, "SC".

The ZX-SERVO16 does not support Auto-Baud. When your ZX-SERVO16 starts up, the default baudrate is 2400. The "SC" portion is an identifier that pertains to the ZX-SERVO16 board. Together, the "!" and the "SC" form a preamble, "!SC". The preamble serves to distinguish commands for the ZX-SERVO16 from other messages on the serial I/O line. After the preamble is sent, the command and associated parameters are sent. The eighth and final character sent is a $0D, (CR), used to terminate the string. If the command causes the ZX-SERVO16 to reply, a three-byte reply is sent after a 1.5 mS delay.

2.2 Baudrate setting

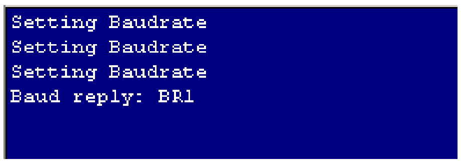
ZX-SERVO16 board can select 2 baudrates 2400 and 38400 bit per second. the default baudrate is 2400. User can change and check the current baudrate bt sending "!SCSBR" command. See sample program-1 below (This program is written to compattible with BASIC Stamp2SX or i-Stamp)

Baud value in Sample program-1 for **SEROUT** *command is 1021. It means 2,400 baud. If need 38400 baud must change to 45*. However in setting baud value of **SEROUT** command must add $8000 to select communication mode to Open-Collector

In Sample program-1 send "!SCSBR" ,1,CR command to change the current baudrate from 2400 to 38400. After that, baudrate interface must use 38400. The **SERIN** comamand in next, must use baud value to 45 instead. SERIN command is used to read the current baudrate to store in variable, BR1. It is 38400.

The method to change baudrate back to 2400 again is *RESET*. User can press RESET switch on ZX-SERVO16 board or send the command "!SCSBR" ,0,CR.

```
'{$PBASIC 2.5}
Sdat PIN   9      ' Serial Data I/O pin
Baud CON   1021     ' Constant for 2400 baud
buff VAR   Byte(3) ' temporary variable
SetBaud:
DEBUG "Setting Baudrate", CR
SEROUT Sdat, Baud+$8000, ["!SCSBR",1,CR]
SERIN Sdat, 45,500, SetBaud, [STR buff\3]
DEBUG "Baud reply: ", buff(0), buff(1), DEC1 buff(2), CR
STOP
```



```
Setting Baudrate
Setting Baudrate
Setting Baudrate
Baud reply: BR1
```

Sample program-1 PBASIC code for select and read baudrate of ZX-SERVO16 board

2.3 VER? Command - Identify Firmware Version Number

Syntax: **"!SCVER?" $0D**

Reply: **"1.3"**

The **VER?** command causes the ZX-SERVO16 to reply with its firmware version number. A string terminator, $0D, must follow each command; note that you may use the constant CR instead. The version number divulged pertains to the version of firmware contained within the IC. The following code snippet can be used to find and identify your ZX-SERVO16. See the Sample program-2 for example.

```
'{$STAMP BS2sx}
'{$PBASIC 2.5}
Sdat     PIN   9                  ' Serial Data I/O pin
Baud     CON   1021               ' Constant for 2400 baud
buff     VAR   BYTE(3)            ' temporary variable
FindPSC:                          ' Find and get the version
DEBUG "Finding PSC", CR           ' number of the PSC.
SEROUT Sdat, Baud+$8000, ["!SCVER?",CR]
SERIN Sdat, Baud, 500, FindPSC, [STR buff\3]
DEBUG "PSC ver: ", buff(0), buff(1), buff(2), CR
STOP
```

Sample program-2 PBASIC code for reading version of ZX-SERVO16

This program select P9 of Stamp to interface with ZX-SERVO16. Baud value set to 1021 for 2400 baud. Reserve 3 byte memory for collect version data response.

Within the DEBUG window, you will see a series of "Finding PSC" messages. The Stamp should find the unit immediately. This is evidenced by the ZX-SERVO16 replying with a number like "1.3", which is it's firmware version number. If after 3 or 4 messages the ZX-SERVO16 has not been found, you should check that all the connections are proper and that power is applied. If the ZX-SERVO16 fails to respond, you may momentarily push the Reset button on the ZX-SERVO16. If the ZX-SERVO16 fails to respond, contact INEX's Technical Support at tech@inex.co.th.

2.4 Position Command - Set the Position of a Servo Channel

Syntax: **"!SC" C R pw.LOWBYTE, pw.HIGHBYTE, $0D**

To control a servo, you must write a position command to the ZX-SERVO16. Each position command is comprised of a header, three parameters: C, R, and PW, and a command terminator.

l "!SC" is the header. The header signifies to all devices on the same wire that this is a command for a Servo Controller.

l C parameter is a binary number 0-31 corresponding to the servo channel number. The servo channel number should be 0-15 with no jumper present on the ZX-SERVO16, or 16-31 with the jumper present.

With a jumper present on the ZX-SERVO16,

No connect AUX jumper	servo channel 0 to 15,
Connect AUX jumper	servo channel 16 to 31

- **R** parameter is a binary number 0 - 63 that controls the ramp function for each channel.

  If the ramp parameter is set to 0,

  ramping is disabled and the pulse width will be set to the P parameter sent immediately.

  If the Ramp values of 1-63

  correspond to speeds from ¾ of a second (0.75 second) up to 60 seconds for a full 500µSec to 2.50 mSec excursion.

- P parameter is a 16-bit Word that corresponds to the desired servo position. The range, (250-1250), corresponds to 0 to 180 degrees of servo rotation with each step equaling 2 µSec.

- The command terminator, $0D, (CR), must not be omitted.

The following code in Sample program-3 demonstrates the Position Command by ramping a servo channel 11 from 0 to 250 at ramp rate 7.

```
'{$STAMP BS2sx}
'{$PBASIC 2.5}
ch   VAR BYTE
pw   VAR WORD
ra   VAR BYTE
Sdat CON 9            ' Serial Data IN/OUT
baud CON 1021         ' Baudrate 2400
ra = 7                '
ch = 11               ' Control servo Motor on CH11

DO
     pw = 1250
     SEROUT Sdat, Baud+$8000,["!SC", ch, ra, pw.LOWBYTE, pw.HIGHBYTE, CR]
     PAUSE 1000
     pw = 250
     SEROUT Sdat, Baud+$8000,["!SC", ch, ra, pw.LOWBYTE, pw.HIGHBYTE, CR]
     PAUSE 1000
LOOP
```

Sample program-3 PBASIC code for setting servo position

Note: Not all servos are exactly alike. If your servos appear to strain when commanded to position 250, you should increase the 250 up to 260 (or so) to prevent the servo from straining itself. Similarly, if your servo strains when commanded to go to position 1250, you should decrease the 1250 to 1240 or so to prevent the servo from straining itself.

2.5 RSP Command - Report the Position of a Servo Channel

Syntax: **"!SCRSP" ch $0D**

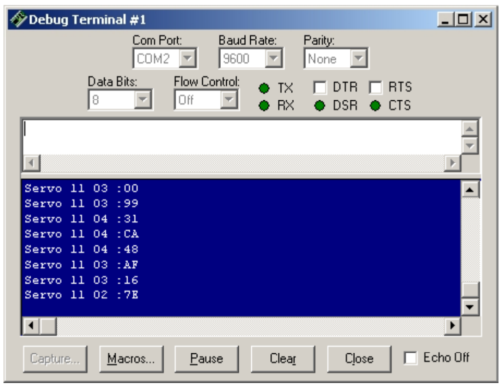
Reply: **ch position1 position2**

Sometimes it is necessary to know the current servo position. The RSP command returns the value of the specified servo channel. If the servo is still moving as a result of a ramp command, the position may still be read.

The following code in Sample program-4 demonstrates how to use this command. Within the **DO-LOOP**, this program sets the pulse width (**pw**) to one extreme or the other and writes this value to the ZX-SERVO16. The ramp value (**ra**) was set to give the program time to poll the servo position several times. Within the WRservo subroutine, the new **pw** is sent, and the position is polled five times, once each second. A DEBUG command is used to format the reply and post it to a window for your review.

```
'{$STAMP BS2sx}
'{$PBASIC 2.5}
ch VAR BYTE
pw VAR WORD
ra VAR BYTE
x VAR BYTE
Buff VAR BYTE(3)
Sdat CON 9
baud CON 1021
Init:
ra = 15: ch = 11
DO
     pw = 1240: GOSUB WRservo
     pw = 240: GOSUB WRservo
LOOP

WRservo:
     SEROUT Sdat, Baud+$8000,["!SC", ch, ra, pw.LOWBYTE, pw.HIGHBYTE, CR]
     FOR x = 0 TO 4
     PAUSE 1000
     SEROUT Sdat, Baud+$8000, ["!SCRSP", ch, CR]
     SERIN Sdat, Baud, 1000, Init,[STR Buff\3]
     DEBUG "Servo ", DEC buff(0), " ", HEX2 buff(1), " :", HEX2 buff(2), CR
     NEXT
RETURN
```



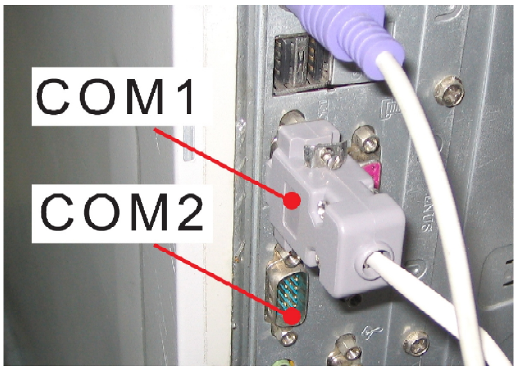
```
Servo 11 03 :00
Servo 11 03 :99
Servo 11 04 :31
Servo 11 04 :CA
Servo 11 04 :48
Servo 11 03 :AF
Servo 11 03 :16
Servo 11 02 :7E
```

Sample program-4 PBASIC code for reading servo position from ZX-SERVO16 board

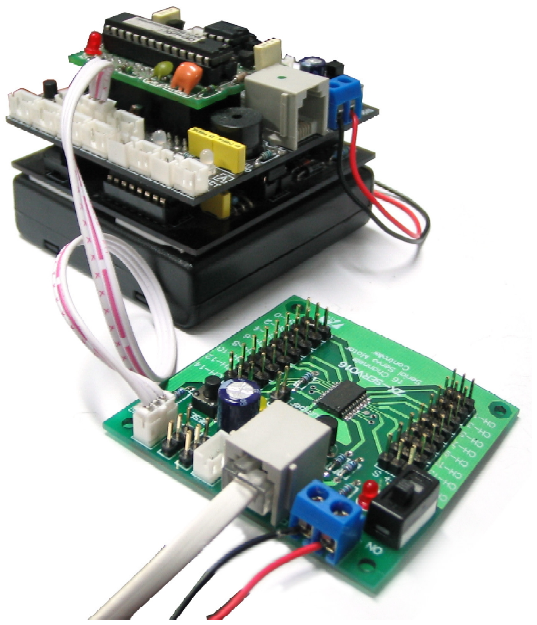
3. ZX-SERVO16 with Computer control

Interfaceing between ZX-SERVO16 baord with computer will be done via RS-232 serial port. See the figure 4 to method of interfacing.User must apply +5V supply to ZX-SERVO16 board separate Servo supply voltage. In figure 4, use +5V supply from Stamp-BOX (Robot controller baord with i-Stamp). User can apply +5V at +5V pin of Serial connector.

The controlled software is PSCI or Parallax Servo Controller Interface.Visit at www.parallax.com and search for PSCI. or download at www.inex.co.th. Once the program is downloaded, double-click on it and follow the installation instructions.



Connect the CX-4 custom RS-232 cable at DB-9 side to PC's serial port at COM1 or COM2



Connect CX-4 at modular end to RS-232 jack on ZX-SERVO16 board. Apply +5V supply voltage from Stamp-BOX board to ZX-SERVO16 processor. Do not forget to appply +6V to Servo power terminal for supply servo motors.

Figure 4 How to interface ZX-SERVO16 with computer via RS-232 serial port

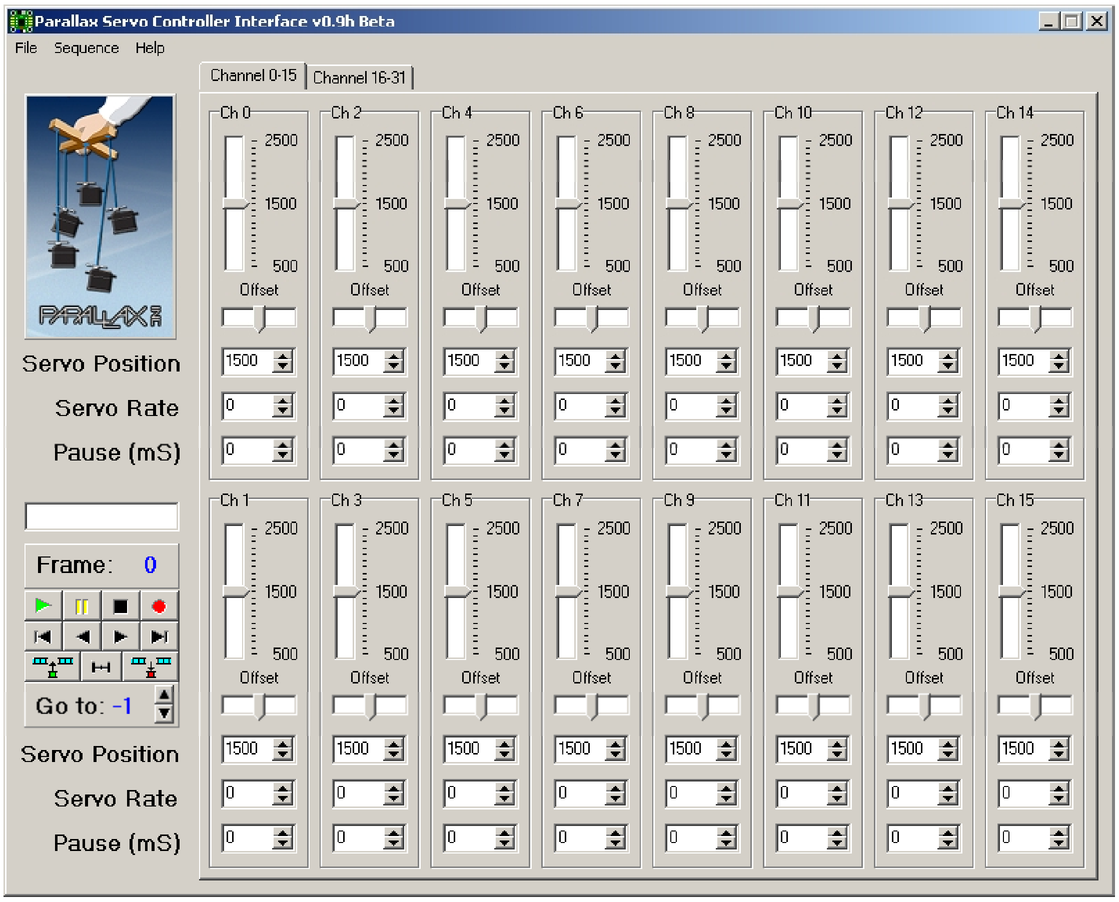


Figure 5 PSCI : Parallax Servo Controller Interface software main window

3.1 Lauch the PSCI software

Once installed, launch the PSCI software. It should start immediately and the following screen will be displayed the figure 5

3.2 Get version of ZX-SERVO16 board

To checking the communication between ZX-SERVO16 board and PC, select COM port and select Get PSC Version command in File menu to read version data of ZX-SERVO16 board

If correct : *The version dialog box will appear and show message in figure 6*

If cannot connect : Software will show *PSC Not Found* message instead. Check the interface again included the supply voltage too.

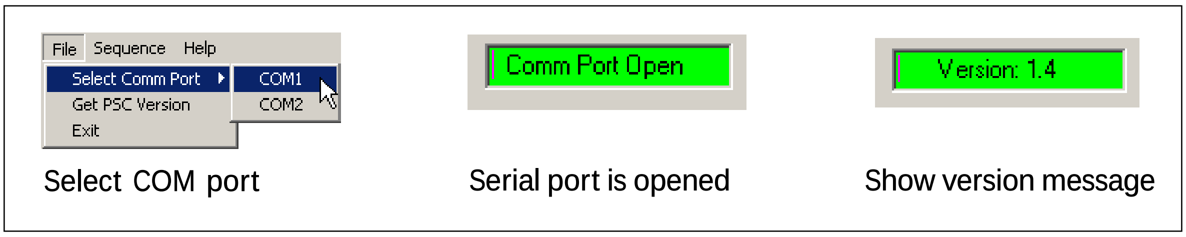
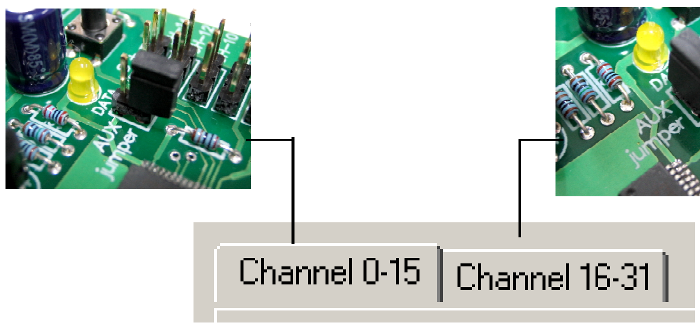


Figure 6 Version checking of ZX-SERVO16 board with PSCI software



Fugure 7 Show the servo channel jumper selection at AUX jumper on ZX-SERVO16 board

3.3 Controls Description

3.3.1 Real-time mode

PSCI can control servo in 2 modes, Real-time and Animate mode. All modes can control all 32 servos.In case control all 32 servos, user must connect tow of ZX-SERVO16 board together with serial cable at Serial connector. One board does not fit AUX jumper to define servo channel 0 to 15. Another must fit AUX jumper to define servo channel 16 to 31 (see figure 7).

To control the servos in real-time mode, connect your servos to the ZX-SERVO16 and simply slide the power switch on the ZX-SERVO16 to the ON position. Now, moving the corresponding servo position slide bar in the PSCI software positions the servo (figure 8).

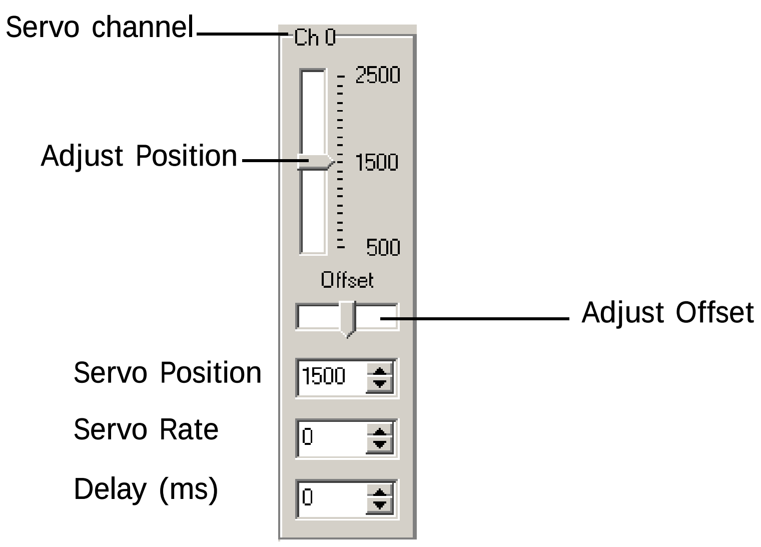


Figure 8 Real-time servo slide control in Channel 0

The description about some control parameter as :

Offset : Not all servos are alike so the offset control allows you to vary the center point of each servo. This may not matter too much for standard servos, but is very helpful when controlling servos modified to rotate 360 degrees.

Position : You may use either the slide control or the up/down arrows, or enter the servo position numerically to set the servo position. Please note that by clicking on the numbers 2500, 1500, or 750, the servo position will be set to that number. These numbers correspond to the width of the servo command pulse, in microseconds.

Rate : Each servo may be set to a rate of rotation. Furthermore, each servo may have a different speed for each frame. A servo rate of 0 will cause the servo to move as quickly as possible; it's fastest rate. A servo position of 63 will cause the servo to move to it's destination position at its slowest rate. At speed 63, the servo takes about 45 seconds to complete a 180 degree motion.

Delay : *When a frame is executed, as in animate mode*, a servo command is sent to the ZX-SERVO16 for each servo in the order 0,1,2,3... After each servo command is sent, the delay time is observed before sending the next servo command. *The Delay parameter has no effect when operating in real-time mode.*

3.3.2 Amimate control

Controls are provided to facilitate the creation, controlling, and editing of animation sequences. By hovering the mouse over each control button (without clicking) a hint will pop up to help clarify the purpose of the button. Additional information is available by clicking on Help->Help in the PSCI software.

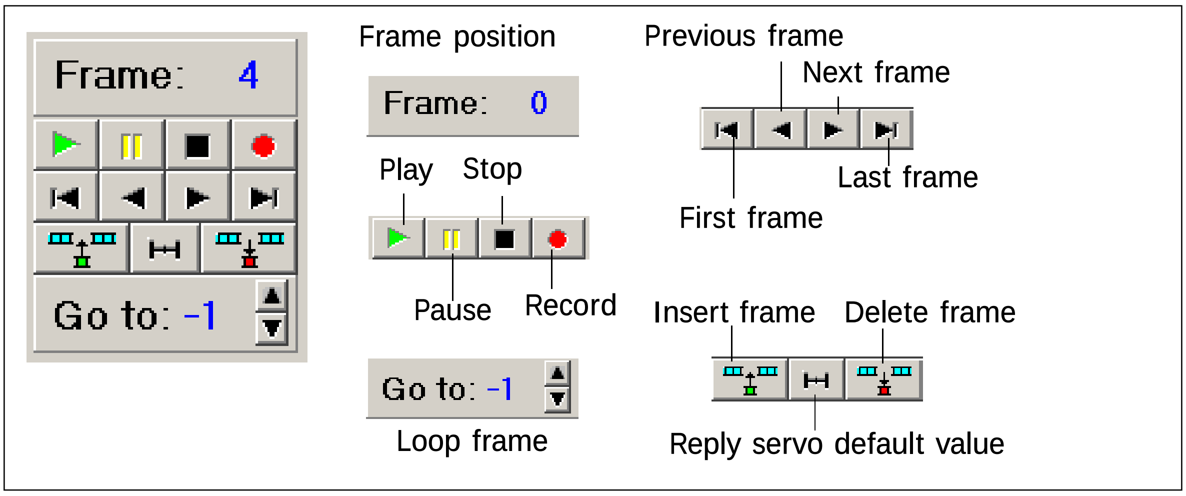


Figure 9 Command button in Animate control of PSCI software

The description about some control parameter as :

Frame : The number to the right of the word “Frame” indicates the current frame.

Go to : The number to the right of the phrase “Go to” shows the frame number that will be jumped to when all servo commands have been sent and all delays have been observed for the current frame. Note: no servo commands are sent for servos that appear to be unused.

3.3.2.1 Creating a Sequence

Creating a sequences is easy. Maximum frame can contain in a sequence is 128 (0 to 127). The following example shows you how to create a simple sequence.

(1) Select servo channel, adjust Position, select Servo Rate and Delay

(2) Click RECORD button ● for recoreding the servo adjustment value.

(3) Select another channel or still adjust same channel. Adjust, set the value and record value by click RECORD button. Do it until the last frame depend on your demand (max 128 frame).

(4) If need to looping, set the number of frame that go back into Goto box. If you need to re-start at first frame, select Goto 0 etc.

3.3.2.2 Playback the sequence

Click PLAY button ▶ . Program will re-start at first frame and play it. You can brake temporaly by click PAUSE button ⏸ and replay again by click PLAY button.

If need to stop program, click STOP button ■ to stop operation.

You can save the sequence into disk. Enter to Sequence **à** Save . The saving file is .PSC extension. If need open the file, to menu Sequence **à** Open

3.3.2.3 Sample animate control

In ths sample owrk with servo channel 0, select to looping operation and delay 0.5 second.

(1) Click on the 500 position for servo channel 0 and edit the delay for that channel to 500mS as shown.

(2) Click the red circle ● to record the frame.

Note: clicking the record button on the last frame of a sequence caused the current frame to be copied to the next frame (this helps with animation sequences). If you were to click the record button after editing an existing frame, the current frame would be updated, and the next frame would not be disturbed.

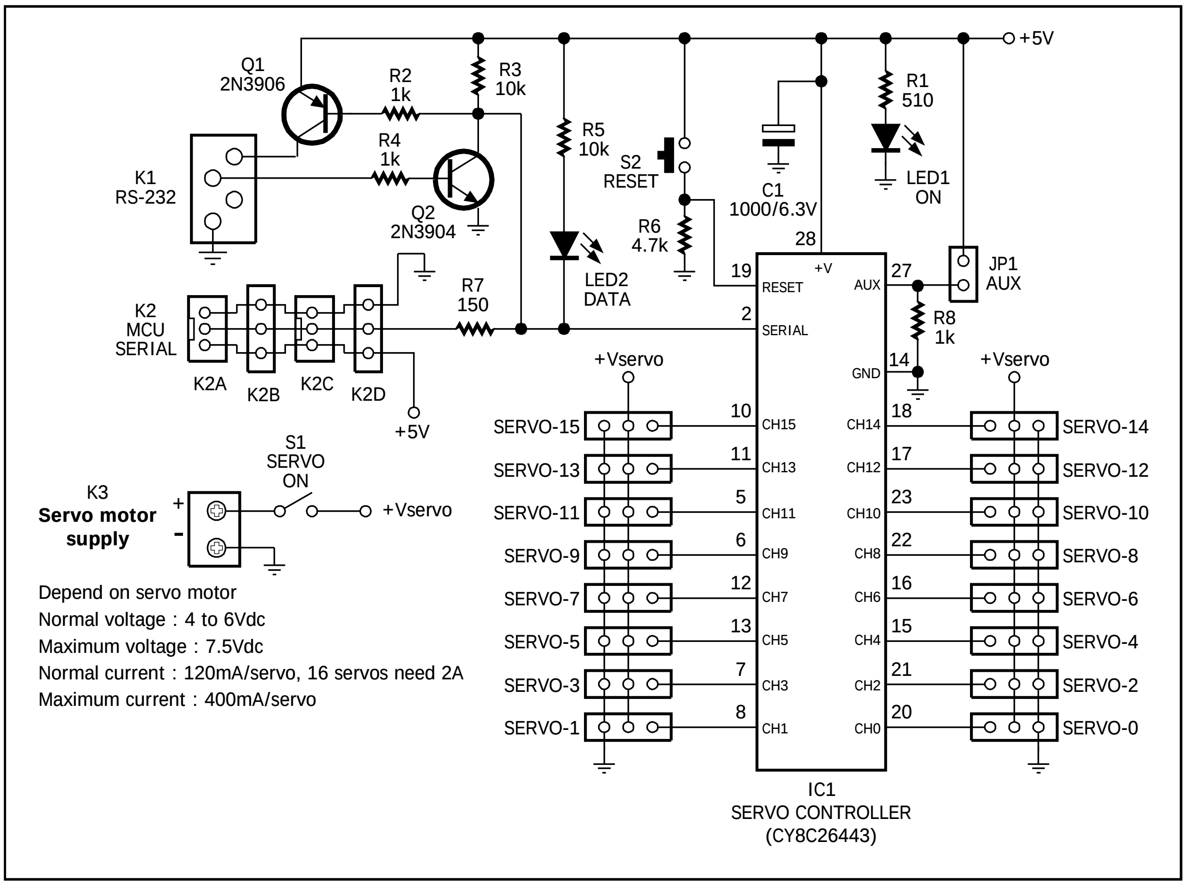
(3) Click the number 2500 for channel 0 and click the record button again to record first frame.

(4) Click the number 500 for channel 0 and click the record button again to record the second frame.

(5) Click the number 1500 for channel 0, increment the Go to number to 0 and click the record button to record the third frame. That completes one simple animation sequence. Now you may use the CD-like control buttons to run, stop, and pause your animation sequence.

(6) To save your sequence, simply click on Sequence **à** Save and specify a file name.

(7) To retrieve your saved sequence, simply click on Sequence **à** Open and specify the filename.



ZX-SERVO16 schematic

INNOVATIVE EXPERIMENT