PICAXE-08M2 SERVO DRIVER (AXE024)

Description:

The PICAXE-08M2 servo driver board is primarily designed to allow the PICAXE-08M2 chip to control up to four radio control style servos for robot models. It is also designed to be compatible with a switch on input C.3.

Assembly Overview:

The servo driver board is a high quality plated through PCB and is therefore relatively straight forward to assemble. However a number of the electronic components are polarised, so please ensure these components are fitted the correct way around before soldering (see table below).

Tools required (not supplied):

- Soldering iron and solder
- Side Cutters
- Small pair of pliers

Soldering experience is assumed.

Contents:

- PCB   AXE024   PICAXE-08M2 Servo Driver PCB v2
- IC1   ICH008   8 pin IC socket
- IC1   AXE007M2   PICAXE-08M2 microcontroller
- H1-4   CON037   10 way headers (snap into 4x3 way)
- R1   RES22K   22k resistor (red red orange gold)
- R2, 7   RES10K   10k resistor (brown black orange gold)
- R3-6   RES330   330 resistor (orange orange brown gold)
- C1   CAP012   33uF tantalum capacitor   *** +, - marked on PCB
- C3   CAP001   100nF (104) polyester capacitor
- D1   RES041   1N4001 diode   *** bar end marked on PCB
- CN1   CON039   3.5mm stereo socket
- CN2   CON005   2 pin screw terminal block
- BAT   BAT016   battery clip
- BAT   BAT011   4 x AA battery box

(*** denotes components which must be soldered the correct way around. See notes above).

Circuit Diagram:
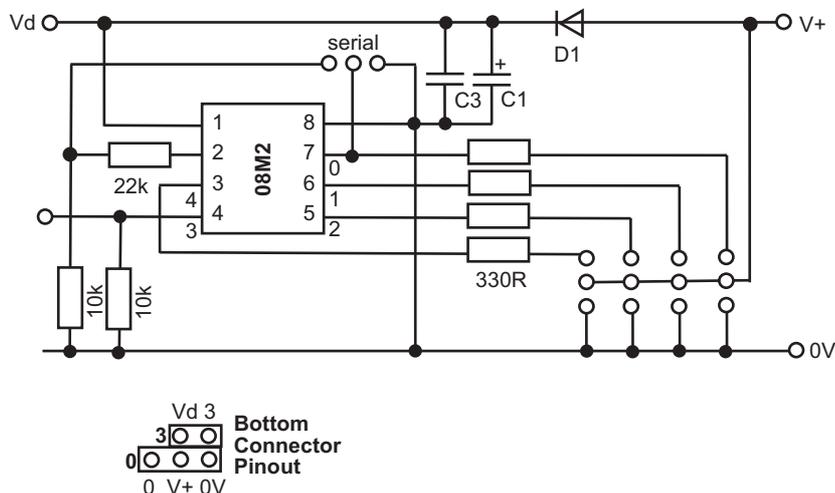
Assembly Instructions:

1. Solder the resistors and diode in position, ensuring correct polarity of the diode.
2. Solder the stereo download socket sockets in position. Ensure the stereo socket 'clicks' into position flat on the PCB prior to soldering (it does not matter if solder joins the two 'pairs' of contacts each side).
3. Solder the 8 pin IC socket in position. Note that due to an unavoidable change in brand of the stereo socket, clearance between the IC and stereo socket is now very tight. Ensure all 8 pins of the IC socket can be soldered by pushing the socket in place at a slight angle.
4. Solder the capacitors in position. Ensure correct polarity of the two tantalum capacitors (long leg is +ve).
5. Solder the 4x 3-pin headers (snapped to size from 10 way header) and the 2 way terminal block in position.
6. Insert the PICAXE-08M2 microcontroller supplied into the 8 pin socket, pin1 to the left.

PICAXE-08M2 Input/Output Pin Connections:

Outputs 0, 1, 2 and 4 are connected to the three pin servo headers. The threes pins provide 0V, V+ and signal. Although these pins are normally used as outputs, they may also be used as inputs if desired.

Remember that this output 0 is also used as the serial output pin and so will 'pulse' on and off during a new PICAXE program download.

Input 3 is already connected to a 10k pull-down resistor and so a switch may be directly connected between the two pads at the bottom of the board.

Connecting Servos:

Each servo should be connected to the three pin header (output 0, 1, 2 or 4). The black wire of the servo should be connected to the inner pin (yellow/white wire on outside of PCB).

Power Supply:

The servo driver board is designed to operate at 6V (4 x alkaline cells) or 4.8V (4 x rechargeable cell). A diode provides a 0.7V voltage drop before the PICAXE chip, which therefore operates at an acceptable maximum of 5.3V. **Do not** use a 7.2V or 9V battery pack without regulating the PICAXE supply to 5V, as this will damge the PICAXE chip. The board will also operate from a regulated 5V or 6V DC supply.
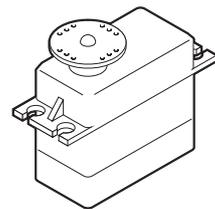
Electrical Noise:

Servos can produce a large amount of electrical noise whilst operating. If your program behaves erratically, particularly resetting often, this can be a symptom of electrical noise on the power rails. Simply remove the servo and test the program again - if normal operation follows removal of the servo this will isolate the problem to noise from the servo. Use further capacitors to suppress the noise on the power rails.

Sample Program:

This sample program will operate 3 servos on the outputs 1, 2 and 4.

```
'*** Program constants
' Delay to slow down servo motion (30ms default)
        symbol servo_delay = 30

'*** Initialisation
init:
' set servos to known position
        servo C.1,120
        servo C.2,120
        servo C.4,120
        pause servo_delay

'*** Move forward
move_forward:
        for b1 = 120 to 180
                servopos C.1,b1
                pause servo_delay
        next b1

        for b1 = 120 to 180
                servopos C.4,b1
                servopos C.2,b1
                pause servo_delay
        next b1

        for b1 = 180 to 120 step -1
                servopos 1,b1
                pause servo_delay
        next b1

        for b1 = 180 to 120 step -1
                servopos C.4,b1
                servopos C.2,b1
                pause servo_delay
        next b1

        goto move_forward
```