

```

1  =====
2  |
3  |   SERVO CONTROLLER FOR R/C MODELS
4  |
5  |
6  |   VERSION 1.0 30th October 2008           © A.F.Bond 2008
7  |
8  |=====
9  | This module was developed for the control of guns and cranes mounted on the
10 | decks of model boats, but aircraft, military and robotics modellers etc will
11 | doubtless find other applications for it. It combines the functionality of an
12 | R/C switch, a servo-slower and a servo stretcher in a unit whose footprint is
13 | not much larger than a standard servo.
14 |
15 | The R/C switch function detects joystick positions either side of a centre
16 | deadband and sets the servo moving in the appropriate direction. Releasing the
17 | stick (to centre) at any time stops the movement and holds the position. The
18 | traverse rate of the servo can be set to take up to approximately 10
19 | seconds from end-to-end. The end of travel limits for both clockwise and
20 | anticlockwise can be set by the user. Also, if the servo can cope with being
21 | (mechanically) over-driven, the usual +/-45 degrees swing can be extended up
22 | to a full +/-90 degrees.
23 |
24 | NOTE: In the description below be aware that PIN1, PIN2 etc are virtual
25 | pins and not to be confused with the device's physical pins, which in PIC
26 | parlance are referred to as 'legs'
27 |
28 |=====
29 |
30 | HARDWARE DEFINITIONS
31 | symbol jumper_link=pin4    'exactly what it says!
32 | symbol push_button=pin2    'exactly what it says!
33 | symbol servoh = 0 'pin0    'note spelling 'servo' is a reserved word
34 |
35 |
36 | VARIABLE DEFINITIONS
37 | symbol signal_input = w0    'from r/c receiver
38 | symbol servo_position=w1    'exactly what it says!
39 | symbol delta=b4            'no. of 5uSec steps the servo moves by each time
40 | symbol loops=b9            'no. of times round main loop before servo moves
41 | symbol kount=b5            'for general purpose counting
42 | symbol anticlockwise=b6    'movement direction flag
43 | symbol clockwise=b7        'movement direction flag
44 | symbol rate=b8             'traverse rate set by potentiometer
45 | symbol anticlockwise_limit=w5 'end of servo angular travel
46 | symbol clockwise_limit=w6   'end of servo angular travel
47 |
48 |
49 | CONSTANT DEFINITIONS      'USER MAY ADJUST DEADZONE & HYSTERSIS VALUES
50 | symbol deadzone=20        'region in which joystick movement has no effect
51 | symbol hysteresis=2       'prevents chatter at switching point
52 |
53 |
54 | DERIVED CONSTANTS        'BUT MUST LEAVE THESE ALONE !
55 | symbol anticlockwise_trip=300+deadzone    '300=stick centre
56 | symbol anticlockwise_release=anticlockwise_trip-hysteresis
57 | symbol clockwise_trip=300-deadzone        '300=stick centre
58 | symbol clockwise_release=clockwise_trip+hysteresis
59 |
60 |
61 |
62 | CHIP PROGRAMMING DIRECTIVE (only executed when program is loaded into chip)

```

```

63 'Pre-load non-zero values into EEPROM storage locations 0 and 1
64 data 0,(100) 'set clockwise limit of travel to 45 degrees
65 data 1,(100) 'set anticlockwise limit of travel to 45 degrees
66
67
68
69
70
71
72
73
74 '===== PROGRAM STARTS HERE =====
75
76 'INITIALISATION
77 setfreq m8 'set 8Mhz operation to yield 5uSec resolution
78
79
80 '=====
81 ' POWER UP CHECK
82 '=====
83 if jumper_link=0 then 'is jumper fitted?
84 goto setup_clockwise_limit 'YES - enter set-up mode
85 else
86 read 0, delta 'NO - retrieve user settings from
87 clockwise_limit=300-delta 'non-volatile memory and enter main program
88 read 1, delta
89 anticlockwise_limit=delta+300
90 servo_position=300 'initialise servo to centre position
91 goto main
92 end if
93 '=====
94
95
96
97 '=====
98 ' SET UP
99 '=====
100 'MOVE SERVO DIRECTLY WITH POT POSITION (clockwise from centre only)
101 setup_clockwise_limit:
102 readadc 1,clockwise_limit 'read pot on PIN1's ADC
103 clockwise_limit=clockwise_limit*20/25 'scale pot value 0 to 200
104 delta=clockwise_limit 'save amount prior to offsetting
105 clockwise_limit=300-clockwise_limit 'limit now in range 100 to 300
106 pulsout servoh,clockwise_limit 'move servo to that position
107 pause 40 '20 mSec frame rate
108 if push_button=1 then 'is this setup finished?
109 goto setup_clockwise_limit 'NO - keep looping round
110 else
111 do 'YES
112 if push_button=1 then 'has button been released yet?
113 write 0,delta 'YES - store limit in NV memory
114 goto setup_anticlockwise_limit 'exit this loop to next setup
115 end if
116 loop 'NO - wait until button released
117 end if
118
119
120 'MOVE SERVO DIRECTLY WITH POT POSITION (anticlockwise from centre only)
121 setup_anticlockwise_limit:
122 readadc 1,anticlockwise_limit 'read pot on PIN1's ADC
123 anticlockwise_limit=anticlockwise_limit*20/25 'scale pot value 0 to 200
124 delta=anticlockwise_limit 'save amount prior to offsetting

```

```

125 anticlockwise_limit=anticlockwise_limit+300 'limit now in range 300 to 500
126 pulsout servoh,anticlockwise_limit      'move servo to that position
127 pause 40                                '20 mSec frame rate
128 if push_button=1 then                   'is this setup finished?
129     goto setup_anticlockwise_limit      'NO - keep looping round
130 else
131     do                                   'YES
132         if push_button=1 then           'has button been released yet?
133             write 1, delta               'store limit in NV memory
134             servo_position=anticlockwise_limit 'set position variable to match
135                                     'where servo now is
136             anticlockwise=1             'and set direction flag too
137             goto main                   'exit this loop to next setup
138         end if
139     loop                                 'NO - wait until button released
140 end if
141 '=====

142
143
144
145
146
147
148
149
150
151
152 '=====
153 ' MAIN PROGRAM LOOP STARTS HERE
154 '=====
155 main:
156 readadc 1,rate                           'read pot on PIN1's ADC
157 rate=rate/25                             'scale to 10 distinct values
158
159 'USER MAY ADJUST VALUES IN THE LOOKUP TABLE BELOW TO CUSTOMISE TRAVERSE RATES
160 'get traverse time data related to pot position
161 lookup rate,(2,1,2,1,2,1,2,1,2,4),delta
162 lookup rate,(9,4,7,3,5,2,3,1,1,1),loops
163
164 pulsins 3,0,signal_input                  'check receiver input
165
166 'CHECK IF JOYSTICK POSITION EXCEEDS THRESHOLDS AND SET MOVEMENT FLAGS TO SUIT
167 if signal_input>anticlockwise_trip then
168     anticlockwise=1                       'set to move
169 else
170     if anticlockwise=1 then
171         if signal_input<anticlockwise_release then 'hysteresis zone exceeded?
172             anticlockwise=0                 'YES - cancel movement
173         end if
174     end if
175 end if
176
177 if signal_input<clockwise_trip then
178     clockwise=1                           'set to move
179 else
180     if clockwise=1 then
181         if signal_input>clockwise_release then 'hysteresis zone exceeded?
182             clockwise=0                     'YES - cancel movement
183         end if
184     end if
185 end if

```

```

186
187
188
189
190 'DETERMINE WHEN TO MOVE SERVO POSITION
191 if anticlockwise=1 then           'ANTICLOCKWISE MOVEMENT
192   if servo_position>clockwise_limit then   'reached the limit yet?
193     inc kount                       'NO - increment loop counter
194     if kount>=loops then           'loop counter reached target?
195       servo_position=servo_position-delta 'YES - alter servo position
196       kount=0                      'then reset loop counter
197     end if
198   end if
199 end if
200
201 if clockwise=1 then              'CLOCKWISE MOVEMENT
202   if servo_position<anticlockwise_limit then 'reached the limit yet?
203     inc kount                       'NO - increment loop counter
204     if kount>=loops then           'loop counter reached target?
205       servo_position=servo_position+delta 'YES - alter servo position
206       kount=0                      'then reset loop counter
207     end if
208   end if
209 end if
210
211                               'NO MOVEMENT REQUIRED
212                               'if both flags zero, program
213                               'drops through to this point
214
215
216
217 'REFRESH SERVO POSITION (whether moved or not)
218 pulsout servoh,servo_position
219
220
221
222 goto main                       'loop round again
223
224 =====

```