```
'{$STAMP BS2}
'{$PBASIC 2.5}
'This program is to be used with the BStamp, two 180 degree
'servos to make up one 360 AZ servo, and a 180 degree servo
'to make a satellite antenna tracking demonstration system.
'The software in the Stamp is designed to work with NOVA or
'SatPC32 Satellite tracking programs using the EASYCOMM1 protocol.
'The Stamp software takes the commanded antenna azimuth and elevation
'degrees and drives the attached servos to the appropriate relative
'positions.  The servos need to be manually positioned to
'0 degrees (North) and 0 degrees elevation before the start
'of the satellite pass.  This is a demonstration system only,
'the step resolution of the servos is not sufficient for use
'as an actually satellite tracking system, therefore the pointing
'angles of the system will only be approximate.

'Sending a pulse with of .500 mS will position the rotors to 180
'degrees, sending a pulse with of 2.30 mS will position the
'rotors to 0 degrees.  These values were determined experimentally
'and are approximate.  Since the standard servos will only turn 180
'degrees, two servos, one mounted on top of the other and turned by
'the bottom servos shaft are required TO make up the AZ rotor.  The
'bottom servo will turn the antenna from 0 to 180 degrees AZ, the top
'servo then takes over to turn the antenna from 181 to 359 degrees.
'Only one EL servo is required since elevation is limited to 180 degrees.

'BY MARK SPENCER, WA8SME.  Author can be contact at:
'mspencer@arrl.org.



'working variables
new_az      VAR     Word           'place holder for desired azimuth
new_el      VAR     Word           'place holder for desired elevation
i            VAR    Byte
'BStamp relay direction control and communication constants
portin      CON     16            'Stamp to computer I/O pin

'main program

new_az = 0                        'stating position 0/0 degres
new_el = 0

main:                             'beginning of main program loop


                                   'elevation change

GOSUB  get_new_az_el              'get position data from computer
                                  'if auto tracking parameters for elevation are set
                                  'to -3 degrees and 183 degrees, then tracking program will
                                  'return negative elevation values when the satellite
                                  'is below the horizon.  The next line will ignore
                                  'negative elevations.  If the parameters are set as
                                  'default 0 and 180 degrees, then the interface will
                                  'track the satellite even if it is below the horizon
                                  'because the tracking program NOVA will send 0 degrees
                                  'when the satellite is below the horizon.


 IF new_el < 0 OR new_el > 180 THEN main
                                  'because the BStamp uses fuzzy integer math during
                                  'comparisons of negative numbers, this line avoids
                                  'problems by checking for the desired range of positive
                                  'elevations (0 to 180 degrees).
GOSUB move_servos


GOTO main

'subroutines

move_servos:                      'subroutine that causes the servos to move to the new
                                  'commanded position.  remember the pulse width parameter
                                  'in the PULSOUT command is in counts of 2 uS and the
                                  'servos deal with pulse widths between .5 and 2 mS'
                                  'so the parameter to send a pulse width of .5 mS would
                                  'be 250 uS * 2 = 500 uS = .5 mS.  to send a pulse width
                                  'of 2.3 mS the parameter would be 1150 uS *2 = 2300 uS =
                                  '2.3 mS. some math discussion is also needed here for
                                  'scaling.  if 0 degrees results from a .5 mS pulse width
```

```
                                'and 180 degrees results from a 2.3 mS pulse with, then
                                'there are 10 uS pulse width differences per degree.
                                'because the Stamp uses 2 uS wide counts to determine
                                'pulse widths, there are actually 5 2 uS counts per degree.
                                'then to calculate the number of 2 uS time intervals to move
                                'the number of degrees change: degree*5.  you need to think
                                'about this to be ready to instruct it.

  IF (new_az<181) THEN          'here we check if the AZ is less than 180 degrees, if so
  FOR i = 0 TO 100              'the bottom AZ servo is turned.
    PULSOUT 13, 1150-(new_el*5) 'move the EL servo first with a single pulse
    PULSOUT 15, 1150-(new_az*5) 'move the bottom AZ servo with a single pulse
    PULSOUT 14, 1150            'make sure the top AZ servoe is at the 0 degree (relative)
    PAUSE 20                    'position, and pause for 20 mS required by the servo electronics
  NEXT                          'the 100 pulses should be more than enough time to move the full
                                'travel


ELSEIF (new_az>180) THEN        'here we elseif the AZ is greater than 180 (it is either or)
  FOR i = 0 TO 100
    PULSOUT 13, 1150-(new_el*5) 'move the EL rotor like above, have to duplicate this
    PULSOUT 14, 250+((360-new_az)*5)'move the top AZ servo for 181-359 degrees
    PULSOUT 15, 250             'make sure bottom servo is at 180 degrees
    PAUSE 20
  NEXT

ENDIF

RETURN  'move_servos


get_new_az_el:                  'get new position from tracking program
    SERIN portin, 84,2000, get_new_az_el, '9600 baud
      [WAIT ("AZ"), DEC new_az, WAIT ("."), WAIT ("EL"), SDEC new_el, WAIT (".")]

RETURN  'get_new_az_el


END
```