How to interface to the VK5DJ Rotator Controller

Introduction

The controller brings in data from antenna encoders of various types and translates this into azimuth and elevation readings for display on the top line of the controller's LCD. These readings are compared with either internally generated data on the position of the moon or sun or compared with data coming in from a computer via a serial port connection. This can be azimuth and elevation data of any computed object such as a satellite or a celestial body.

The controller may then automatically operate mechanical rotators to bring the antenna to the position of the station/object being tracked or an operator may use manual operation.

To enable the use of satellite or logging programs on a computer, the controller will accept information through a serial port. The essential information is azimuth and elevation of the object in degrees, the controller will compare this with antenna AZ/EL and move the rotators. The azimuth and elevation information is sent to the Rotator Controller as a Word variable (the azimuth and elevation are multiplied by 100 and converted to an integer).

A secondary function in the tracking part of the controller is a "remote mode" where the computer does the tracking AND provides information on which way to turn the rotators. This mode is not normally implemented in programs as the controller tracks very nicely based only on azimuth and elevation data from the computer. I do not recommend using this mode unless there is a special reason, but I have provided information in case a program writer has this need, as was the case for VK3UM's Autotrack.

The controller will also accept an optional time calibration string from the computer to set the internal clock of the controller.

To reduce serial communication time overheads, the strings to and from the controller are kept short and multipurpose. This can make the computer's communication a little more complex if time setting or remote tracking is required.

Part 1: Basic Interfacing

Receiving antenna and elevation data from the controller into the computer via serial port for display on the computer

All actions are initiated by the controller. When the controller is ready to send data it raises the CTS line. Immediately after the string is sent the CTS is lowered and the controller waits for a response from the computer.

If you do not wish to use the data from the rotator controller you should still read the string and then discard it to avoid any overflows in the computer buffers.

After raising the CTS the controller sends strings of ASCII values with the following format:

<A><space><azimuth with 2 decimal places><space><elevation with 2 decimal places><space><beam instruction><CR>

The length of the string varies depending on the size of the azimuth and elevation and whether a minus sign is included with the elevation. The "." (dot) character is used to indicate a decimal point.

Typical values may look like this (for clarity in this document I have used the underline character "_" to indicate a "space")

A_	header from the controller
123.45_	azimuth of 123.45 degrees
43.20_	elevation of 43.20 degrees
0	value of movebeam variable (zero except in "remote mode")
13	carriage return indicating end of string

Note there is no space after the movebeam byte variable.

The string is sent as the following ASCII characters:
65 32 49 50 51 46 52 53 32 52 51 46 50 48 32 48 13

Here's another sample this time with a negative value for elevation.

A_	header from the controller
35.99_	azimuth of 35.99 degrees
-5.40_	elevation of -5.40 degrees
0	value of movebeam variable (see later)
13	carriage return indicating end of string

Is sent as the following ASCII characters:
65 32 51 53 46 57 57 32 45 53 46 52 48 32 48 13
Representing
C Sp 3 5 . 9 9 Sp - 5 . 4 0 Sp 0 <CR>

To read the string you need to test for a CTS active and bring in the string at 9600baud, 8 data bits, 1 stop bit, no parity.

Your interface will:
Test for an <A space> and look for a CR at the end. Then string slice to extract the azimuth, elevation and Movebeam variable.
Azimuth and elevation should be converted to a float or extended variable while Movebeam is always a byte. Note Movebeam is discarded by your program if Basic Interface is used.

You should then be able to manipulate the printing and use of the azimuth and elevation. If you do not wish to use the data you should at least read the string and discard it to prevent buffer overflow. Data should not be sent to the controller until the string has been fully received due to the limitations of a two byte buffer in the USART of the PIC.

After sending the antenna information, the controller then waits for the computer to immediately send data to it.

Sending data to the controller from the computer

This is moderately more complex because the controller has less capacity to work with strings. Integers also provide more accuracy.
The controller always expects the following string irrespective of the data being sent, but the information in the data is manipulated to achieve a number of functions. In this section of Basic Interfacing the structure is straight forward and the byte variable at the end is set to 0.

Again, I use a "_" to make a space visible in this document.
A standard string is of this format:
<C_><word variable in range 0-65535><_><second word variable 0-65535><_>
<byte variable 0-255><_><CR>

Note the space after the last byte variable as the PIC looks for spaces to know when it has finished reading a numeric value.

All characters go out from the computer as ASCII
67 32 50 49 51 52 32 55 49 50 56 32 48 32 13
That is:
C Sp 2 1 3 4 Sp 7 1 2 8 Sp 0 Sp CR

This represents:

C_	the header from the computer
2134_	the value of the azimuth multiplied by 100
7128_	the value of the elevation multiplied by 100 (except negatives)
0_	a movebeam command of 0 (normally 0 for non remote mode)
13	carriage return indicating end of string

The azimuth and elevation values are sent as a word variable for the sake of the PIC in the controller. The float value of the azimuth (and elevation) is multiplied by 100 and then converted to an integer. A value of 12.34 becomes 1234 in a word variable.

Because the PIC is reading a word from 0-65535 I use a special strategy to show negative values, it may not be the best one but it works. Your interface should use something like:

```
azout:= trunc(100*MoonAZ);  {convert float to integer for transmission}
elout:= trunc(abs(100*MoonEL));
```

```
if MoonEL <0 then
    begin
     elout:=36000+elout;
    end;
```

so in the computer a value of -5.43 degrees elevation is multiplied by 100, stripped of its sign, added to 36000 to become 36543 and output as the elevation. The controller subtracts 36000 (if the incoming elevation number is greater than 36000) then divides by 100 to change it to a float.

Here is a line of Delphi Pascal code used to send the data to the controller.

```
BufferOut:=comport1.WriteStr(inttostr(flagout)+' '+inttostr(azout)+' '+inttostr(elout)+'
'+inttostr(mode)+' '+#13);
```

Flagout contains a value of 67 for Basic Interfacing (see later for other options)
Azout is the value of azimuth multiplied by 100 and converted to an integer
Elout is the value of elevation multiplied by 100 and converted to integer
Mode is the byte containing movement or time information (zero for basic interfacing).

The program then waits for another CTS signal.

The above strategies enable basic operation between the controller and the computer program. When the controller is in External Mode with a switch on the front panel it will display the computer generated 'object's Az and El on the bottom line of the LCD and do all the arithmetic for tracking.

The following are optional items that provide either time setting or implementing the Remote Mode (not recommended for most situations)

Part 2: High Level Interfacing

Most programs will not include the features that follow. They are Optional Extras.

(A) Set Remote Mode and move beam from the computer (Optional)

If you have chosen to control the beam rotators directly from the computer program (not usually recommended) it can be done using this function. The string is sent just once.

Before the controller will accept movement commands from the computer the controller must have its Ext/Int switch in the external position. Then the computer must switch on Remote mode by sending the appropriate code via the serial port in use. A successful switch to remote mode is shown on the controller LCD as a "*" character after the AZ and EL readings coming from the computer.

This is how you turn on Remote Mode.

```
BufferOut:=comport1.WriteStr(inttostr(68)+' '+inttostr(azout)+' '+inttostr(elout)+' '+inttostr(68)+'
'+#13);
```

Effectively you send a character 68 as the first item and a character 68 in the last item. Azout and Elout should be normal computed values for azimuth and elevation (although could be any integer value).

After sending the above, ensure that the mode variable is zeroed to prevent initial movement of the roators. The controller is now ready to receive information on rotator movement contained within the last byte before the <CR>.

(B) Sending the movement information to the controller (Optional)

Assign a value to Mode (the last byte before the <CR> as follows:

if Dn then mode:=Mode + 16;	Make bit 4 of byte variable Mode=1
if Up then mode:=Mode + 8;	Make bit 3 of byte variable Mode = 1
if Lt then mode:=Mode + 4;	Make bit 2 of byte variable Mode = 1
if Rt then mode:=Mode + 2;	Make bit 1 of byte variable Mode = 1

Assign your values to flagout and azout and elout
Flagout = “C”
AZ and EL multiplied by 100 and converted to the word variables “ azout” and “elout” as per above.

```
BufferOut:=comport1.WriteStr(inttostr(flagout)+' '+inttostr(azout)+'
'+inttostr(elout)+' '+inttostr(mode)+' '+#13);
```

```
The controller will move the beam in accordance with these instructions if
the controller switch is on “Auto”.
```

(C) Closing Remote Mode (Optional)

When shutting down, or if you wish to stop controlling the beam through the controller, remote operation is cancelled by sending the following string.

```
BufferOut:=comport1.WriteStr(inttostr(69)+' '+inttostr(azout)+'
'+inttostr(elout)+' '+inttostr(69)+' '+#13);
```

The star after the object’s data in the controller’s LCD second line will disappear and the computer will no longer be in control of the device although antenna and object position will continue to be exchanged.

You may choose to send the time and the remote mode commands when first opening the connection to the VK5DJ controller. Otherwise it could be done by the operator.

When closing the connection it may be wise to print a message screen that says:

“Connection with VK5DJ Controller is about to close, ensure Auto switch is off”

If the operator had failed to turn the Auto/Manual switch to manual on the controller when the computer parts with control the controller switches to its own information which will cause the rotators to turn to the position computed in the controller.

(D) Setting the time on the rotator controller (Optional)

The same string out of the computer is able to activate/use other options.

```
BufferOut:=comport1.WriteStr(inttostr(flagout)+' '+inttostr(azout)+'
'+inttostr(elout)+' '+inttostr(mode)+' '+#13);
```

Using the same string as for normal communication with the controller the time is encoded as follows:
Firstly obtain the UTC time using TimeZoneBias and GetDateTime (see sample).

Flagout = UTCsecs + 192	add 192 to UTC seconds (used as a flag in the controller)
Azout = UTCmin * 256	shift the UTC minutes into the high byte of the word variable
Azout = Azout + UTChr	add the UTC hours into the low byte of the word variable
Elout = UTCday * 256	shift the UTC day into the high byte of the variable
Elout = Elout + UTCmth	add the UTC month into the low byte of the word variable
Mode = UTCyr – 2000	we can only fit the last two digits in a byte variable

Now send the string:
BufferOut:=comport1.WriteStr(inttostr(flagout)+' '+inttostr(azout)+' '+inttostr(elout)+'
'+inttostr(mode)+' '+#13);

Wait to receive the acknowledgment back with the next packet from the controller – it will echo the mode value. If the two match (movebeam and mode) then reset Flagout to “C” (character 67) and reset Mode to zero to ensure no beam movement takes place. The program should then return to normal operation.

Attachment

I have also attached a sample interface written in Delphi. It forms the basis of a working system.

General

The initial value in the string sent BY the computer has the following meanings:

“C” or character 67 is the normal flag and indicates normal data from the computer. For Basic operation this flag will always be “C”

“D” or character 68 is the flag value to switch the controller into remote mode (sent once)

“E” or character 69 is the flag value to switch the controller out of remote mode (sent once)

Flagout will have an indeterminate value when setting the time although bits 6,7 will be set high and the remaining bits will code the value of the seconds.

The value of the first character sent by the controller will always be a “A” or character 65 followed by a space. This is used to signify the data has come from the controller and is used by my computer programs as a start sequence for an incoming packet. In accessing the serial stream I look for “A “ at the beginning of the string and the <CR> at the end.

I would be pleased to test any programs. Just send an email to jdrew@seol.net.au

John Drew VK5DJ, 2nd November 2009