International Beacon Program – V2

Guide for configuring a controller with the operating parameters and loading new software.

# Introduction

The version 2 rev B controller has several parameters which can be programmed into local memory. Among these parameters is the slot number the unit should operate in. This guide provides instructions for programming those parameters.

The parameters are stored in EEPROM in the Atmel microcontroller on the Arduino Leonardo board. The beacon software will use some of the parameters to operate the beacon. Others are stored for reference. Access for programming the parameters is thru a serial USB cable connecting to the Leonardo board, and the front panel menu button.

Table 1 lists the parameters stored in EEPROM, and their usage.

# Revision history

|  |  |  |
| --- | --- | --- |
| When | Who | What |
| 2/15/2017 | K6TD | Initial release |
| 2/18/2017 | K6TD | Updated delay to be 16 int, little endian, added software loading procedure |
|  |  |  |

# Connecting and putting the controller in EEPROM programming mode

## Connecting the USB cable

1. Remove power from the controller by turning off the radio.
2. Unscrew and remove the cover. There are tow screws in the top cover.
3. Locate the USB connector. It’s a micro USB connector, located on the lower PCB (the Leonardo board) right below the SMA connector
4. Connect a USB cable to this connector.
   1. A right angle style cable is needed, due to limited clearance
   2. The connector must be threaded past the coax and the various control cables.
   3. Do not put lateral stress or pressure on the SMA connector. It will come un-soldered.
   4. Push down gently on the top of the SMA connector to stabilize the unit, and then insert the USB connector.
5. The other end of the USB cable should be connected to a computer USB port – PC, MAC, or a raspberry Pi.

## Putting the controller in programming mode

1. Apply power to the radio and controller.
2. The unit will go thru a sequence of screens as various stages of init are completed.
   1. The controller must be connected to a radio and a GPS antenna before it will enter programming mode
3. As shown in Figure nn, the LCD display will eventually turn Blue and the lower left operating mode will say MENU.
4. When MENU is displayed – Press the MENU button for about 2 seconds.
   1. The controller software will only wait about 8 seconds for a MENU button press before moving on to OPER mode.
5. The LCD should now display “CONS connected”.
   1. If the LCD displays SKIP, the controller didn’t find a serial USB connected to a computer.

## Programming Commands

The controller accepts two command: R for reading a byte from EEPROM, and W for writing a byte to EEPROM. Each command is followed by a carriage return (**↵).**

The R command syntax is:

R <address> Where address is a decimal string of digits representing the address of the byte to read from the EEPROM. The command display’s the contents of the byte addressed, as hexadecimal digits.

Example:

R 0**↵**

R:0:f

R 4**↵**

R:4:9 (for slot 90 seconds after the start of the 3 minute epoch).

The W command syntax is:

W <address> <value> - address is the same as for read command. Value is a string of decimal digits representing the value to write into the EEPROM at the specified address.

Example:

W 0 22**↵** (write the decimal value 22 into the EEPROM byte at address 0)

(no response is given)

# Returning the controller to OPER mode

When EEPROM programming is complete, return the controller to operating mode.

To exit programming mode, press the RESET button on the rear of the controller. Verify the unit operates correctly, with the parameters programmed.

1. Power of the unit, by powering off the radio.
2. Press down gently on the SMA connector top.
3. Remove the USB connector cable from the controller
4. Reinstall the top cover, and tighten the screws
5. Re-connect any cables and power the radio on.
6. VERIFY the unit operates normally and with the programmed parameters

This completes the parameter programming process.

# EEPROM layout and parameters

The EEPROM is byte accessible with addresses from 0 thru 0xff. Not all of the EEPROM is used by the controller software.

Table

|  |  |  |
| --- | --- | --- |
| Address | Name | Usage |
| 00 | Layout version | 1 |
| 01 | Bytes used | 8 |
| 02 | CRC | Calculated – used for error detection |
| 03 | debug | 1= operator controller in debug mode |
| 04 | slotID | Timing slot – see table 2 |
| 05 | sernum | Serial number of unit |
| 06 | HWver | Presently - 2 |
| 07 | delayL | Value to use at slot loop start – see table 3 |
| 08 | delayH | High byte of delay variable |

The layout is defined in file ‘beacon.h’.

Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| slotID | Timing slot | Callsign | Country | Operator |
| 0 | 00 | 4U1UN | United Nations | UNRC |
| 1 | 10 | VE8AT | Canada | RAC/NARC |
| 2 | 20 | W6WX | United States | NCDXF |
| 3 | 30 | KH6RS | Hawaii | MARC |
| 4 | 40 | ZL6B | New Zealand | NZART |
| 5 | 50 | VK6RBP | Australia | WIA |
| 6 | 60 | JA2IGY | Japan | JARL |
| 7 | 70 | RR9O | Russia | SRR |
| 8 | 80 | VR2B | Hong Kong | HARTS |
| 9 | 90 | 4S7B | Sri Lanka | RSSL |
| 10 | 100 | ZS6DN | South Africa | ZS6DN |
| 11 | 110 | 5Z4B | Kenya | ARSK |
| 12 | 120 | 4X6TU | Israel | IARC |
| 13 | 130 | OH2B | Finland | SRAL |
| 14 | 140 | CS3B | Maderia | ARRM |
| 15 | 150 | LU4AA | Argentina | RCA |
| 16 | 160 | OA4B | Peru | RCP |
| 17 | 170 | YV5B | Venezuela | RCV |

The table values are defined in the file ‘stations.h’.

Table 2 are the values in a C language ‘struct’ array. The EEPROM slotID value is used as an index in this table to obtain the slot timing value and the callsign, used during beaconing.

Table

|  |  |  |
| --- | --- | --- |
| Value | Who | Date |
| 248 | K6TD | 2/15/2017 – V2.7f  07 = 248  08 = 0 |
|  |  |  |

The delay parameter is a loop start delay value used in the beacon scheduling code. It’s value is determined by operating a beacon, and adjusting the value based on the observations of delay by FAROS.

# Loading software into the controller

## Introduction

Loading software into the controller is done thru the Arduino utilities executing on a Raspberry Pi with Raspbian Jessie release and USB connection to the controller.

## RPi prep

### Generic or common to standing up a raspberry Pi

Prepare an SD card with an the Raspbian Jessie image from the raspberrypi web site in the downloads section.

1. Install the SD card in a Raspberry Pi and apply power
2. Either connect to the unit via SSH, or a local KVM
3. Login with the userid pi, password raspberry.
4. Update the software
   1. sudo apt-get update
   2. sudo apt-get upgrade
   3. sudo rpi-update (updates firmware
5. start raspi-config
   1. set the time zone to the local time - Pacific new
   2. pick a hostname
   3. expand the file system
6. Reboot

A unique userid can be added if desired.

### Beacon controller specific configuration

Thjs procedure involves adding packages needed for direct programming of the controller, using the Arduino tools, and a USB port

1. sudo apt-get install arduino
2. sudo apt-get install microcom
3. cd ~
4. mkdir beacon
5. git clone <https://github.com/vanbwodonk/leonardoUploader.git>
6. cd leonardoUploader
7. make
8. sudo make install

Plug in the USB cable to a port on the raspberry Pi. Look to see if it was found by the OS – ls –laF /dev/ttyACM\*

kevin@estespark ~ $ ls -laF /dev/ttyACM\*

crw-rw---T 1 root dialout 166, 0 Feb 17 19:57 /dev/ttyACM0

kevin@estespark ~ $

This confirms the USB port has been found by the OS.

## Loading software on the controller

This procedure uses the RPi prepared and configured in the previous section.

Decide on the beacon hex file to be loaded, and copy it to the RPI ‘beacon’ folder. This can be done several ways.

One technique is to directly clone the GIT repository onto the raspberry PI.

1. cd ~/beacon
2. git clone <https://github.com/IronHeartConsulting/IBPV2.git> .
3. git checkout VE8AT\_1

The hex files can be found in directory: ~/beacon/release

ls –laF \*.hex

K6TD-15in-MBP:release kevin$ pwd

/Users/kevin/working/Ham Radio/IBPV2\_VE8AT\_version/release

K6TD-15in-MBP:release kevin$ ll

total 448

drwxr-xr-x+ 8 kevin staff 272 Feb 16 19:59 ./

drwxr-xr-x+ 14 kevin staff 476 Jan 28 09:31 ../

-rw-r--r--+ 1 kevin staff 66315 Feb 14 15:14 beacon\_2\_7f.hex

-rw-r--r--+ 1 kevin staff 70017 Feb 16 10:34 beacon\_2\_8a.hex

-rw-r--r--+ 1 kevin staff 70017 Feb 16 19:59 beacon\_2\_8b.hex

-rw-r--r--+ 1 kevin staff 33 Feb 14 15:15 md5sum\_2\_7f

-rw-r--r--+ 1 kevin staff 33 Feb 16 10:34 md5sum\_2\_8a

-rw-r--r--+ 1 kevin staff 33 Feb 16 19:59 md5sum\_2\_8b

K6TD-15in-MBP:release kevin$

Now to directly load the file:

1. cd ~beacon/release
2. arduinoUploader /dev/ttyACM0 beacon\_<release>.hex

This should complete automatically and terminate.

The LCD display on the controller should show the same version number as the hex file loaded.

To update the files from git:

1. cd ~/beacon
2. git pull origin VE8AT\_1

# Using the an the RPI to access to console for configuration

This procedure describes connecting to the controller thru the RPi configured above as a console, for parameter configuration.

1. Connect the USB cable between the controller and the RPi
2. Verify USB port is present – ls –laF /dev/ttyACM0
3. Start the terminal program – microcom –p /dev/ttyACM0 –speed 115200
4. Let microcom sit for 3 minutes. You should see a $GPSRMC message at least once.
5. Press the reset button on the controller. Microcom will exit.
6. Restart microcom –p /dev/ttyACM0 –speed 115200 (usually up arrow will get you this command back).
7. When the controller LCD display turns blue and shows “MENU” in the lower left corner, press and release the “MENU” button.

Proceed to program the operating and configuration parameters per first segment of this guide.

When complete, press the reset button on the controller.