Battery Programmer Box

1. Overview

The Battery Programmer Box will allow for the programming and reading/verifying of most Zebra batteries. It will also be able to decode the data in most battery packs into human readable form. Versions for use in factory settings to program packs are also planed.

2. Features for initial release

a. Simple serial interface over USB
b. Compatible with Windows, Mac OS, and Linux.
c. Compatible with various battery I2C voltage levels.
    I2C pull up voltage configurable to 5V, 3.3V, and V_{batt}. (V_{batt} can be measured)
    I2C pull ups configurable to 2.3K, 2.9K, 3.4K, 4K, 5.1K, 6.8K, and 10K.
d. Supports standard I2C clock, buffered clock drive, and muxed clock drive.
e. Support for all Zebra EEPROM sizes/addresses
f. Ability to check for a valid authentication IC. (Fully validate starting w/Zebra root key)
g. Support for reading/writing authentication IC EEPROM slot 8.
h. Support for read/write hex files between EEPROM/Authentication IC EEPROM and SD card.
i. Support for "updating" battery data. Do not change serial number, date made, etc.. See section 7.
j. Display some battery data in human readable form.

3. Batteries initially supported

a. MPA2/MPA3 Smart battery
b. MPA3 Gifted battery (MC18, Rogue, Frenzy)
c. Pollux battery
d. Falcon/IronMan Battery
e. Hawkeye Battery
f. Sentry Battery
g. Beast battery

4. Hardware

a. Based on Arduino Mega R3 board.
b. Custom Arduino shield
    Analog mux for pull up voltage selection
    Analog switches for pull up resistor selection
    Analog switches for clock buffer selection
    Clock buffer drive circuit
    SD card interface (Based on standard Arduino SD card interface)
c. 3D printed case.

5. Enhancements after initial release

a. Windows based graphical UI.
b. Read Authentication IC OTP area.
c. Decode all battery data into human readable form. (Only when using Windows UI)
d. Add support for this into Wasath's EEPROM programs.
e. Version for use in pack maker factory.

6. Commands

Cmd	Function	Notes
B	Select battery type	Pick the type of battery, must be done before other operations.
R(E|A)[1]	Read battery data into buffer	Reads entire battery data storage.
W(E|A)[1]	Write battery data from buffer	Overwrites entire battery data storage Verifies after write
U(E|A)[1]	Update battery from buffer	Only overwrites fixed data, does not change values set at manufacture Verifies after write. (See section 7 below)
V(E|A)[1]	Verify battery against buffer	Checks entire battery storage against buffer
E	Erase buffer	Flush buffer to all 0xff's
L	Load hex file into buffer	Loads hex file from com port. Does NOT clear buffer first.
D	Dump hex file from buffer	Dumps a hex file out the serial port from the buffer.
DPP	Display PP Data	Displays PP data in human readable form
DPP+	Display PP+ Data	Displays PP+ data in human readable form
F	Load buffer from file on SD card	Loads a hex file from the SD card. Does NOT clear buffer first.
S	Save buffer to file on SD card	Writes a hex file of the buffer to the SD card.
C	Directory of SD card	Display contents of the SD card.
A	Authenticate Battery	Performs a full authentication on the battery auth chip.
P	Validate Battery	Perform battery data validation (Check checksums, etc..)

1. E or A specifies which device EEPROM/Auth chip to read/write.

7. Update battery function

a. For PP+ data the following is done:

The data blocks 0-2 is copied over from the new data.
The data in blocks 3-7 is left untouched. (Dynamic blocks)
The data in blocks 8-14 is copied from the new data.
The data in blocks 15-16 is left untouched. (Cell identifying data)
The data in blocks 17-31 (25 for auth chip data) is copied from the new data.

b. For Pollux data the following is done:

The format revision byte will be updated, only values 0-3 are supported.
Bytes 2-12 will be left untouched. (Part number, rev, serial number, date made)
Bytes 13-239 will be updated.
The checksum at byte 0 will be updated.
Bytes 240-252 will be left untouched. (aggregate charge)
Bytes 253-295 will be updated.
Bytes 296-323 will be left untouched. (Health, part number)
Bytes 324-511 will be updated.

c. Devices that have both types of data, Falcon, IronMan, etc. will have both sets of changes applied.

d. For Hawkeye data the following is done:

The format revision byte will be updated, only values 0-3 are supported.
Bytes 4-39 will be left untouched. (Part number, rev, serial number, date made)
Bytes 40-199 will be updated.
The checksum at byte 0 will be updated.
Bytes 201-215 will be left untouched. (aggregate charge)
Bytes 216-251 will be updated.
Bytes 252-259 will be left untouched. (Health)
Bytes 260-415 will be updated.