Introduction

This document guides a designer through converting a configuration file to a Zilker Labs Hex Line Delimited (ZLHLD) file, using the ZLHLD Generator tool. The ZLHLD file format is intended for loading configuration data, via SMBus, into a Zilker Labs Digital-DC™ device using an industry-standard high volume programmer.

Design-to-Production with ZLHLD Generator

Figure 1 illustrates the design-to-production flow of device configuration data. All steps shown are expected to be taken by the designer. The ZLHLD Generator converts a finalized device configuration from a human-readable format to machine-readable format, as shown in Figure 2. To use the ZLHLD Generator, you will need to provide a configuration file written to production guidelines, and a PC running Windows XP or Windows Vista.

The rest of this application note will detail steps of the Design-to-Production Flow of Device Configuration Data. These steps will also contain directions on using the ZLHLD Generator.

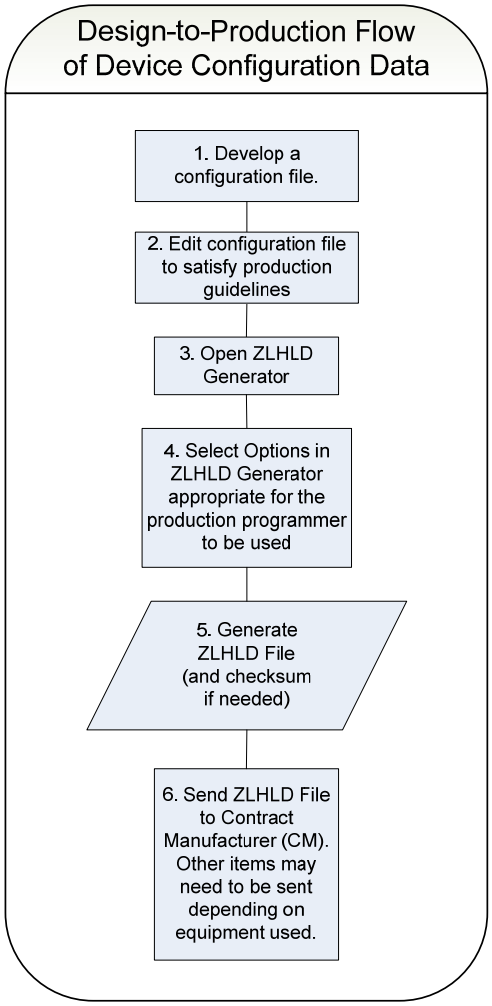

Figure 1. Design-to-Production Flow of Device Configuration

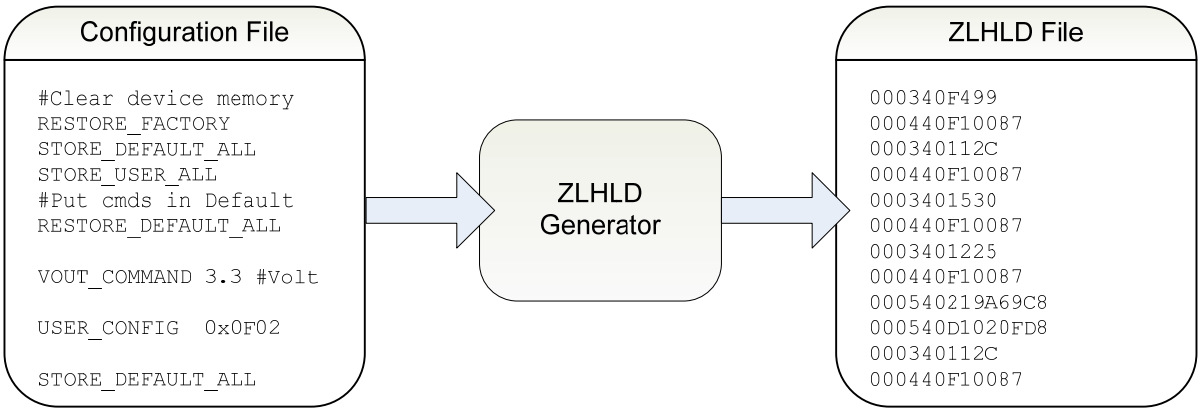

Figure 2. Conversion From Human-Readable Configuration File to Machine-Readable ZLHLD File

Step 1: Develop a Configuration File

First, you will need a configuration file that you have tested in a development environment, and are satisfied that the configuration satisfies your design requirements, and has been tested. The PowerNavigator™ evaluation software can be used to create a configuration file, as well as manipulate an active configuration for testing and development

Step 2: Check That Configuration File Satisfies Production Guidelines

After developing a configuration file that is ready for production, you will need to make sure it can be loaded repetitively onto devices in production. To satisfy this condition, one should make sure of the following:

- Commands written into the configuration file will get stored into non-volatile memory, as the device will be power-cycled during production. Please consult AN31 for information on how to do this.
- Command values derived from pin-strap settings should be accounted for. If the production board has a different pin-strap configuration than the design prototype, you may need to edit the configuration file to add command values originally defaulted as pin-strap values.

Additionally, if your high-volume programmer is serializing devices via the MFR_SERIAL command, you will need to place the serial command data in a fixed location that the programmer can locate. See Appendix I for an example of where serial command data is placed when used with a BPMicro programmer.

Step 3: Open ZLHLD Generator

Now that your configuration file meets the production guidelines, you are ready to launch the ZLHLD Generator. The program should appear as shown in Figure 3 below.

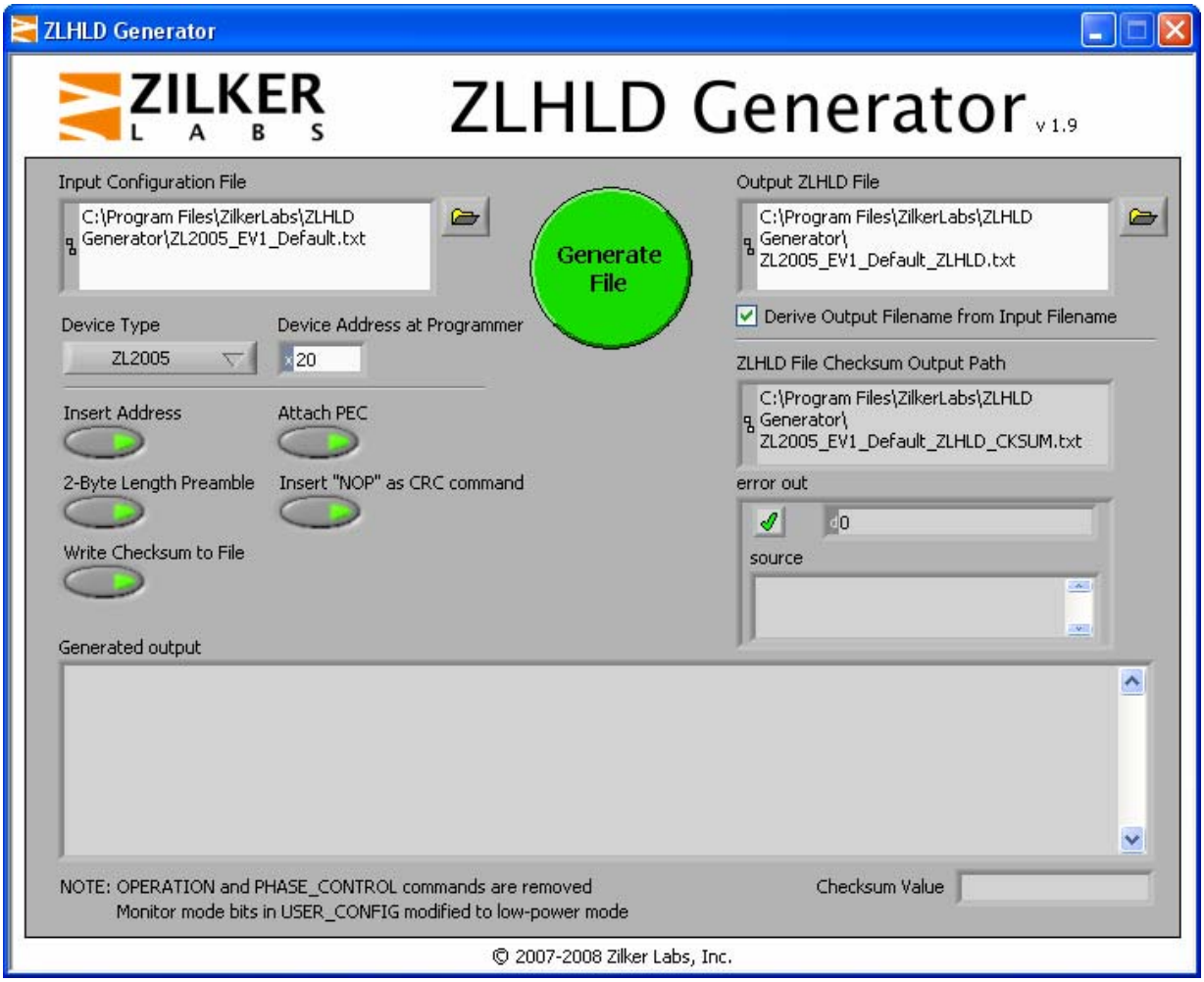

Figure 3. ZLHLD Generator

As shown in the above screenshot, there are many user inputs and options. Here is a detailed feature listing:

Program Inputs:

Input Configuration File: This is the path of the finalized configuration file.

Device Type: The product name (ZL2005, ZL2006, etc.) that the configuration will be programmed into. Setting the correct product type also helps the ZLHLD Generator check for possible errors in your configuration file.

Device Address at Programmer: Enter in the device address that the device will be when being programmed by the high-volume production programmer. In some cases, the device address will differ from when it is on the programmer versus when it is being used in its final application. The address is used when either the *Insert Address* or *Attach PEC* options are selected.

Output ZLHLD File: This is the path of the ZLHLD file to be generated.

Derive Output Filename from Input Filename: This option, selected by default, changes the Output ZLHLD Filename and the ZLHLD Checksum Filename to a name similar to the Input Configuration File, but with _ZLHLD and _ZLHLD_CKSUM suffixes, respectively.

ZLHLD File Checksum Output Path: This is the path of the ZLHLD file checksum. This field cannot be manipulated, instead the output is always the same as your *Output ZLHLD File* but with a "_CKSUM" appended to the pathname. The checksum file is only created when the *Write Checksum to File* option is selected.

Insert Address: This option will insert a left-shifted version of the *Device Address at Programmer* into each line of the ZLHLD file, before any command byte. The address is left-shifted because the address is physically sent out over PMBus left-shifted, along with a read/write bit.

Attach PEC: This option will append a Packet Error Checking (PEC) Byte onto each line of the ZLHLD file. The PEC byte is calculated using a CRC-8 routine, as described in the SMBus specification.

2-Byte Length Preamble: This option inserts a two byte long length on each line of the ZLHLD file. The length represents the number of bytes represented in ASCII-Hex on a given line, excluding the two bytes used by the length preamble itself. This is useful for production programmers in setting a limit in the number of bytes it must write before issuing a stop condition and moving onto the next command.

Insert "NOP" as CRC Command: This option will insert a signal after every STORE/RESTORE command to trigger a delay time to minimize interruption during non-volatile memory operations. The signal used is a the custom CRC command (0xF1) – a benign command which does not affect device configuration, and can either be sent or not sent, depending on how the programmer is setup. See AN30 for more information on required delays during configuration.

Write Checksum to File: This option will write the *Checksum Value* of the created ZLHLD file to a checksum file. The location of the checksum file is as shown by *ZLHLD File Checksum Output Path.*

Program Outputs:

Generated Output: This text windows shows the output contents of the ZLHLD file. Under the *Generated Output* will be some notes on the output. These notes may appear dynamically depending on the *Device Type* chosen.

Checksum Value: This displays the checksum of the *Generated Output*. It can also be written to a file if the option *Write Checksum to File* is selected.

Error Out: This displays whether generating the ZLHLD file was successful or not. If it is successful, it should display a green checkbox and a text stating "OK". Otherwise, a red X will appear along with an error message.

Step 4: Enter ZLHLD Generator Inputs & Select Options

Once you've opened the ZLHLD Generator, you will need to enter in at a minimum the following:

- *Input Configuration File* – Required as it contains your configuration data
- *Device Type* – This is required in order to check for errors, and ensure correct generation of the ZLHLD File. Failure to enter the Device Type in correctly may result corrupted ZLHLD files, and/or unexpected errors in attempts to generate a ZLHLD file.
- *Output ZLHLD File* – Required as the program must know where to put the ZLHLD file.

Aside from the required inputs, you will also need to select other options depending on the high-volume programmer being used. An example of what options may need to be selected can be found on the Submission Checklist for BPMicro Programmers found in Appendix II.

Step 5: Generate the ZLHLD File

After entering the required inputs and selecting programmer-specific options, you are ready to generate the ZLHLD file. Press the green *Generate File* button to create the file. If the *Write Checksum to File* was selected, the checksum file will also be generated at this time.

If successful, the *Error Out* message will have a green checkmark with an OK statement. If not successful, read the error message generated. Typical errors will be due to malformed filepaths, or issues with parsing the inputted configuration file. Issues of the latter will typically have a pop-up error issued, directing to the erroneous line in the configuration file.

Step 6: Submit ZLHLD File to Manufacturer

Now that the ZLHLD file has been created, you must now submit the file to a manufacturer, whether it be a Contract Manufacturer (CM) or an internal manufacturing department. It is expected that the high-volume programmer the manufacturer uses supports Zilker Labs devices. In some cases, additional files such as the ZLHLD checksum, or external serialization programs will also need to be submitted to the manufacturer. Because of this, it is recommended to use a Submission Checklist to ensure that all the necessary data is sent in the correct format to the manufacturer. An example of a Submission Checklist for BPMicro Programmers can be seen in Appendix II.

Appendix I: Writing Configuration Files for Serialization

When creating a configuration intended to be serialized in production, you must write the configuration such that the serial data (six ASCII characters) is at a fixed location. In order to do this, some limitations must be imposed. First, the MFR_SERIAL command must be the fifth line in the configuration file. Second, some high-volume production programmers are unable to detect the serial number as a line in a ZLHLD file, but as a fixed byte location within a concatenated stream of bytes in the ZLHLD file. When this limitation is imposed, one must also ensure that commands 1-4 are always the same for all configurations programmed using that programmer.

As the configuration file example in Figure 4 shows, the first five command lines are always the same. Despite this limitation, one can still configure the device as desired, and take advantage of storing settings to both USER and DEFAULT stores, and even include password protection (see AN31 for details).

If a device that doesn't contain a USER store is used with serialization, the STORE_USER_ALL command can be replaced with a second STORE_DEFAULT_ALL command to remain compatible with the high volume programmer. See Figure 5 for an example of this.

```
   # Clear Device Memory

1  RESTORE_FACTORY      # Clear
2  STORE_DEFAULT_ALL    # Default Store
3  STORE_USER_ALL       # & User Store

   # Perform actions for Default Store

4  RESTORE_DEFAULT_ALL # Prepare Default
                       # for adding cmds

    # Insert configuration data
    # you want in Default Store

   # MFR_SERIAL must be fifth command,
   # and have six characters
5  MFR_SERIAL           INSERT
   MFR_ID               Example OEM
   VOUT_MAX             5.0 #Volts
   VOUT_COMMAND         3.3 #Volts

   STORE_DEFAULT_ALL    # Store Settings

   # Perform actions for User Store

   RESTORE_USER_ALL # Prepare User
                    # for adding cmds

    # Insert configuration data
    # you want in User Store

   VOUT_COMMAND         3.0 #Volts
   PID_TAPS  A=1634, B=-2799, C=1227

   STORE_USER_ALL       # Store Settings
```

```
   # Clear Device Memory

1  RESTORE_FACTORY      # Clear
2  STORE_DEFAULT_ALL    # Default Store
3  STORE_DEFAULT_ALL    #

   #NOTE: Second instance of
   # STORE_DEFAULT_ALL is to put
   # MFR_SERIAL in the right
   # position of the ZLHLD file

   # Perform actions for Default Store

4  RESTORE_DEFAULT_ALL # Prepare Default
                       # for adding cmds

    # Insert configuration data
    # you want in Default Store

   # MFR_SERIAL must be fifth command,
   # and have six characters
5  MFR_SERIAL           INSERT
   MFR_ID               Example OEM
   VOUT_MAX             5.0 #Volts
   VOUT_COMMAND         3.3 #Volts

   STORE_DEFAULT_ALL    # Store Settings
```

Figures 4 & 5. Example production configurations with serialization support, with and without a USER Store

Appendix II: Example Submission Checklist for BPMicro Programmer

The example checklist shown is intended for a designer to go through and verify the ZLHLD file generated is compatible with the programmer used by the manufacturer.

ZLHLD Submission Checklist for BPMicro Programmers

Designer Name: _____ Company: _____ Manufacturer: _____

Zilker Product used: ZL_____ Date: _____ Other: _____

Configuration Verification

1. ☐ Configuration file satisfies project requirements
2. ☐ Configuration file puts the desired commands in to the non-volatile USER or DEFAULT store
3. ☐ Configuration File follows Production Programming Guidelines
4. Am I using Device Serialization? ☐ Yes ☐ No

4a. If Yes is selected to the above question, check that: ☐ The MFR_SERIAL command is the fifth command in a configuration file (i.e. the fifth line excluding comments / whitespace)

ZLHLD Generator: Required Inputs

☐ *Input Configuration File* ☐ *Device Type* ☐ *Output ZLHLD File*

ZLHLD Generator: Programmer-Required Options

Note: For BPMicro Systems Programmers all options listed MUST be selected

☐ *Device Address at Programmer MUST be set to "20"*

☐ *Insert Address*

☐ *Attach PEC*

☐ *2-Byte Length Preamble*

☐ *Insert "NOP" as CRC command*

☐ *Write Checksum to File*

Content Submission

The following must be sent to the Contract Manufacturer:

☐ ZLHLD File

☐ Checksum of ZLHLD file

If using device serialization: ☐ ZL_ESP serialization program (dependent on product being used)

Designer Signature: X_____ Supervisor Signature: X_____

Notes

Revision History

Date	Rev. #	Description
10/28/08	1.9	Software: - Updated parser allows for tabs and space before ASCII data on certain commands - Output Configuration File selection now allows folders or new files to be selected - Output Configuration Files / Checksum Files are defaulted to be derived from the Input Configuration Filename
08/09/08	1.8	Document: Added device address setting to BPMicro Example Checklist Software: - Instances of USER_CONFIG are modified to set monitor mode to low power on devices ZL2005, ZL2005-2, ZL2005P, and ZL2105 - Configuration mode command inserted into hex files on all DDC-enabled devices
06/06/08	1.0	Initial Release
05/01/09	AN2036.0	Assigned file number AN2036 to app note as this will be the first release with an Intersil file number. Replaced header and footer with Intersil header and footer. Updated disclaimer information to read “Intersil and it’s subsidiaries including Zilker Labs, Inc.” No changes to application note content.

Zilker Labs, Inc.
4301 Westbank Drive
Building A-100
Austin, TX 78746

Tel: 512-382-8300
Fax: 512-382-8329
www.zilkerlabs.com