Personality Editor User's Guide

1 Overview

The Teknovus Personality Editor is a program that allows users to easily create and edit personality files for the TK3700 series of EPON chips. The program has three modes of operation: user mode, author mode, and command-line mode.

User mode allows users to define values for personality fields, based upon a personality profile which defines the layout of the personality region for different chips. Engineers designing with the TK3700 series would use this mode to define their desired personality values for the OLT and ONU products that they design.

Author mode allows the user to edit the profiles which define the personality layout. Author mode can be used to create new profiles for future firmware releases, change default values, or edit text describing the personality fields. Teknovus supplies personality profiles to match the personality format for Teknovus firmware releases, as documented in the Host Interface Specifications for the various products in the EPON chip family.

Command-line mode is intended for use by automated scripts. CLI mode allows changing a few fields within an existing personality file. This feature could be used, for instance, to serialize MAC address for a series of ONUs on a manufacturing line, starting from a base or template personality file and customizing that file for each ONU produced.

Once you have edited the file to suit your needs, you can upload the profile to a TK37XX device using either the serial port or through an OLT via the Host Interface. The Personality Editor can interface to firmware both from the CLI port or an Ethernet-based Host Interface on the OLT.

Alternatively, a completed personality file can be saved as a binary image, which you can then edit with the command-line automation included in this program. This binary image can then be loaded back into the GUI program, or it can be applied to the device in a manufacturer-specific manner.

2 Installation

2.1 System Requirements

This program requires a PC running Microsoft Windows. The PC should optionally have a serial port to connect to the ONU or OLT CLI, and/or an Ethernet port to connect to the Host Interface of an OLT.

Depending on the Personality Editor distribution you're using, it may also be required that the PC you're using have the Microsoft .NET runtime environment version 1.1 installed. See "readme.txt" for more details.

The Microsoft .NET runtime is available for download at no charge from www.microsoft.com.

http://msdn.microsoft.com/netframework/downloads/framework1_1/

You can also download the file using Windows Update.

2.2 Procedure

1. Just copy the distribution folder to a location on your hard drive, then run the appropriate executable.
2. See "readme.txt" for more details on the distribution modes / appropriate executable.

3 User Mode

User mode is used to create a personality file. All personality files are based upon a "profile", which defines the layout of all of the fields in the personality region.

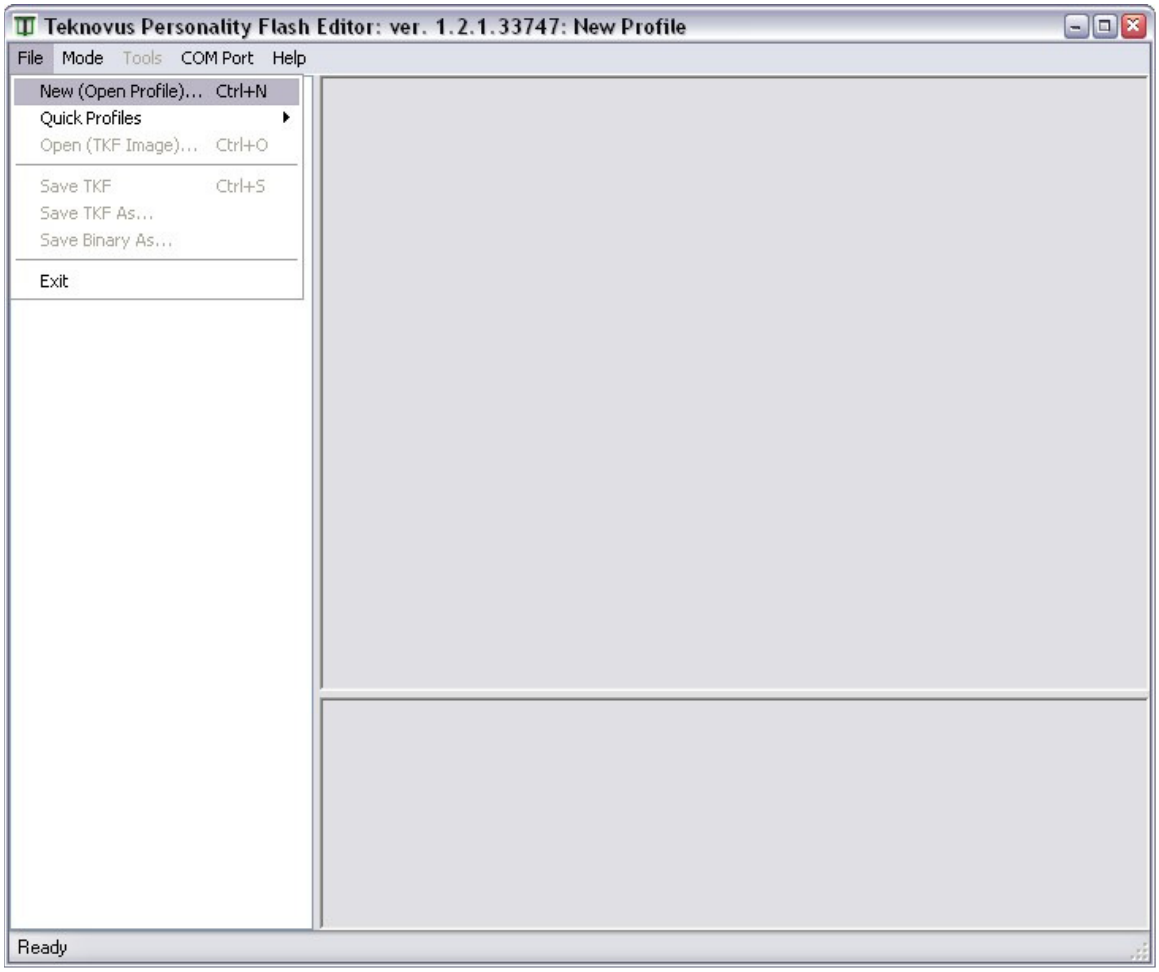
3.1 User Mode vs. Expert Mode

There are two flavors of "user mode" available. The first, which is called simply *user mode*, allows the user to change any of the personality fields which are generally safe to change. In general, this is the mode of choice for most users. In this mode, it is difficult to make a mistake which could have a severe impact on the device.

On the other hand, *expert mode* gives the user access to each and every personality field. If some of these fields are changed, it could have disastrous effects on the device. Use this mode with caution.

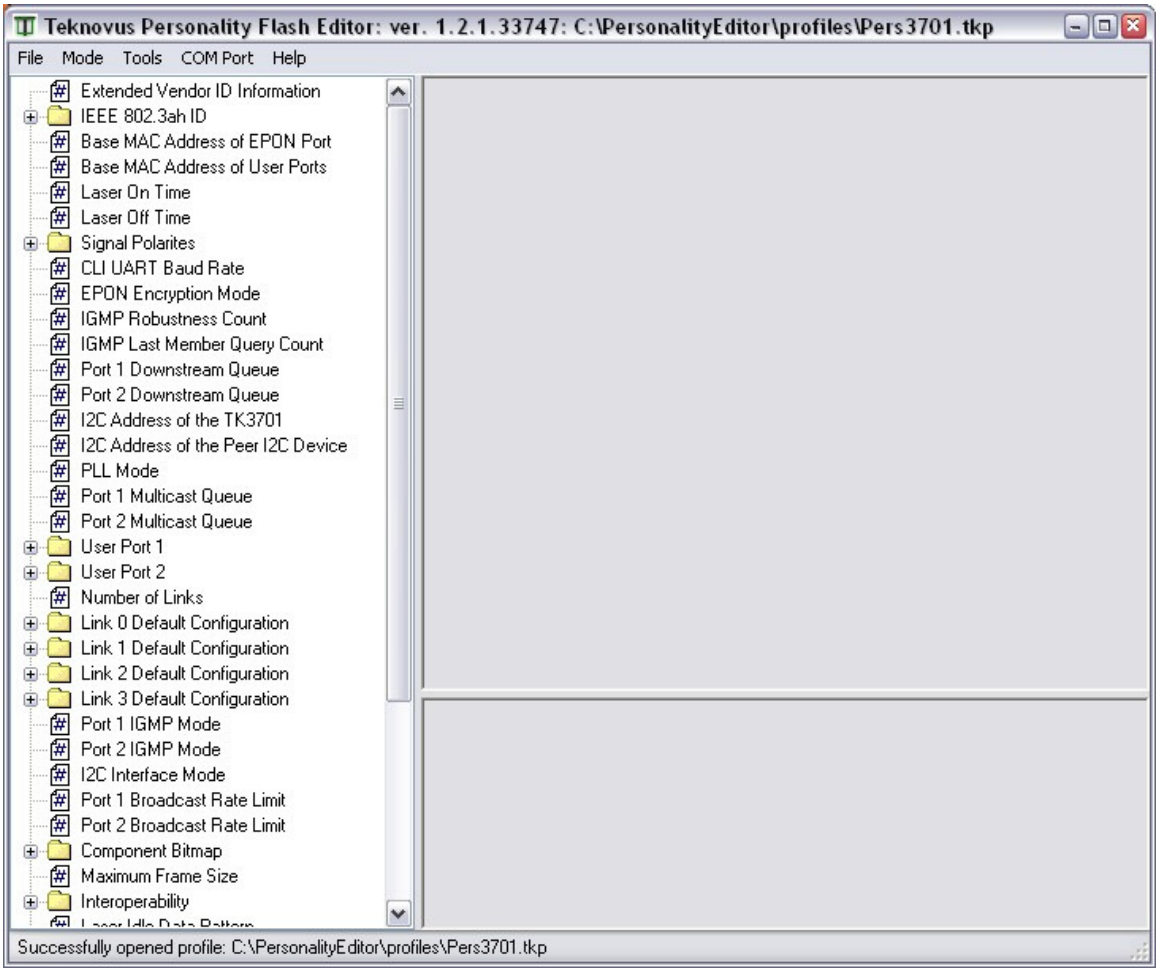
3.2 Creating Personality Files

To create a new personality file, first choose the profile that matches the chip number and personality version desired.



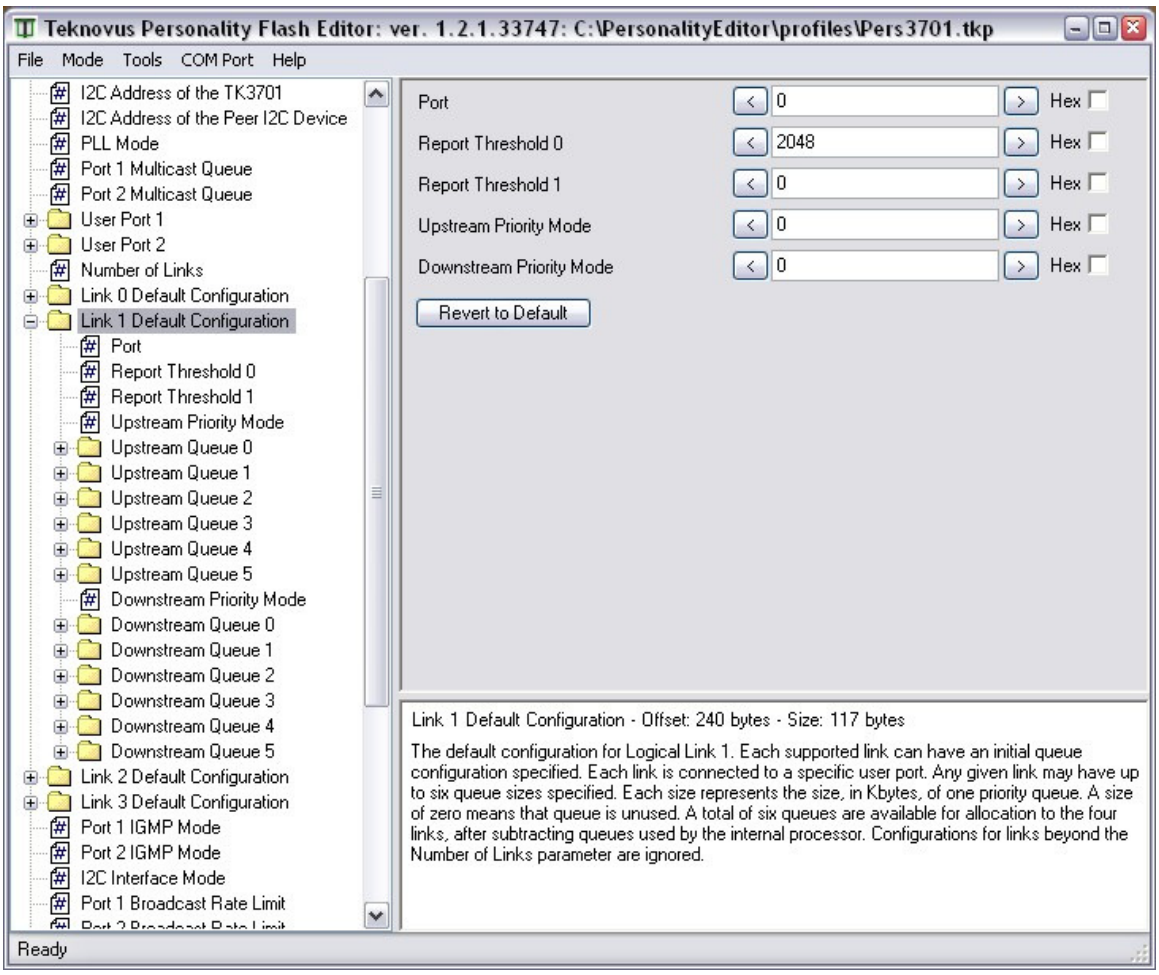
Personality profiles have the extension ".tkp", and have a name similar to "Pers3701_R110.tkp".

Once the profile has been selected, the left-most window will be populated with a list of fields in the personality region. Fields represent individual values in the personality file.



When a field is selected, the upper-right panel will display details of that particular field. This panel can be used to change the value of the field. The lower-right panel will display a description of the field and any special meanings for values of that field.

Fields with the # icon are *scalar* fields, which have just one value. Fields with the folder icon are *structured* fields, which are a collection of related values. Structured fields may contain scalar fields or other structured fields. Selecting a structured field will display a list of all scalar fields in the structure in the upper right window. If any of the fields are structured, the "+" to the left of the folder icon can be used to expand those structured fields to gain access to the scalar components of the structure.

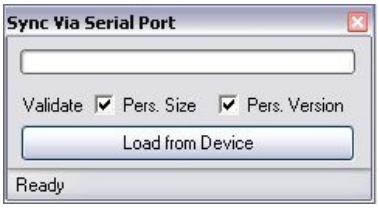


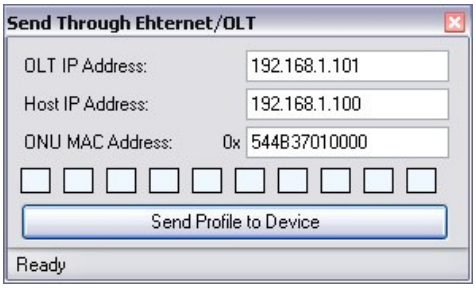
The “Revert To Defaults” option in the Tools menu can be used to set all fields in the personality file to the default values defined by the profile in use.

Once all of the fields in the personality file have been set as desired, the file can be saved using options in the File menu. Files can be saved either in the Teknovus .TKF format, or as raw binary images of the personality region.

3.3 Connecting to an OLT or ONU

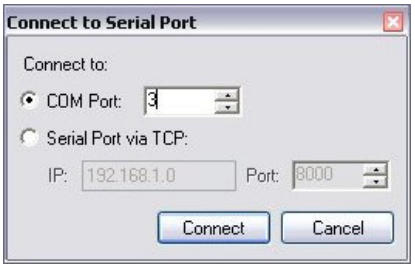
The Teknovus Personality Editor can also be used to read and write personality files to a TK3700 series device via a serial connection to the firmware CLI, or through an OLT using the Ethernet Host Interface. The Tools menu contains options to read the personality from the flash memory of a device, or to write the personality file to the flash on a device.

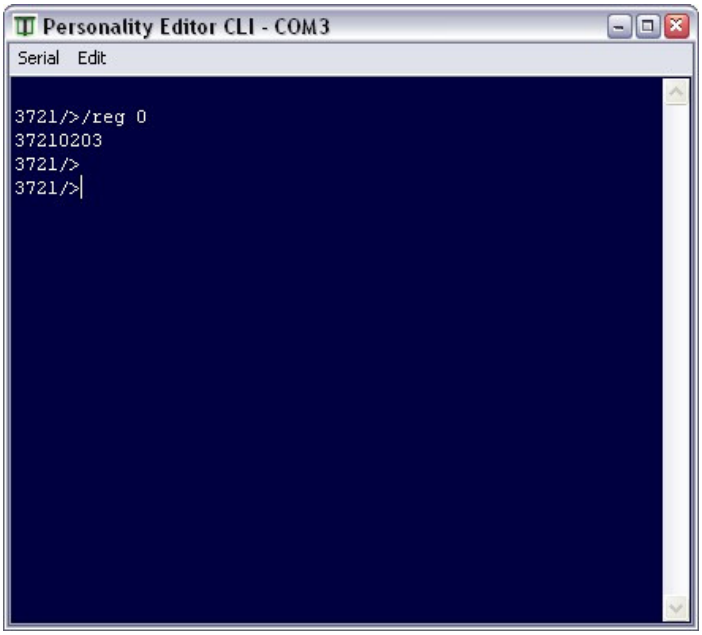




3.3.1 Using the Built-In CLI

For ease of use, the Teknovus Personality Editor has a built-in function to connect to an OLT/ONU using a serial port. With this feature, you can stay connected to your device while uploading/downloading personality files. The CLI is easy to use, simply click on "Show CLI…" under the "Serial Port" menu. It will bring up a dialog which allows you to choose which serial port to use. You have the option of communicating with a built-in serial port (COM Port), or a serial port via TCP/Telnet. Once you have chosen the serial port that is connected to the device, it will bring up a window which you can use to communicate with the device.





3.4 Menu Options

- File → New (Open Profile): Opens a profile for editing.
- File → Quick Profiles: Displays a menu containing all profiles under the folder where you last opened a profile. Note that this option is not available if you have no .tkp files in the main folder.
- File → Open (TKF Image): Loads the values from a previously saved TKF file. This only works if the profile you are currently working on is the same profile the TKF was created with.
- File → Save TKF/Save TKF As: This saves the values to a TKF file (the format actually stored in the device). This file can then be reloaded at a later date, or saved for use by another program.
- File → Save Binary As: This saves the values to a raw binary file with no TKF header.
- File → Migrate TKF Images: This is a function which allows you to take TKF files created with old profiles, and update them so they can be used by different, newer profiles.
- File → Recent Files: Below "Save Binary As" in the options menu, there are a number of recently opened profiles. Once you open a profile, it is saved in this menu. There are up to 4 recently opened profiles stored in this menu.
- File → Export Table: Exports the entire personality file as a human-readable HTML table. This can be used to generate documentation, or as a quick snapshot of the personality values which could be sent over email or by any other means.
- Mode → Chip Options: Here you can view all of the options related to the current device (such as chip ID, personality version, etc).
- Mode → Author Mode/Expert Mode/User Mode: Use this to switch between author mode and user mode/expert mode. You should probably stay in user mode unless you have to.

- Tools → CLI Read/Write: This will open a window which allows you to read or write the current personality profile to a device through the serial port command line interface. Note that you can only send a personality profile through the CLI if the device's CLI includes the /load/RxPers command.
- Tools → OLT Host Read/Write: This will open a window which allows you to read or write the current personality profile to a device through the OLT using host interface messages. Note that, as of this time, there is no host interface message to read the personality from a device.
- Tools → Reset ONU/OLT: This will simply reset the ONU/OLT device.
- Tools → Reset to Defaults: This will reset all values to their defaults according to the profile.
- Tools → Increment MACs: A tool that gives you a quick way to increment one or more of the MAC address fields in the personality file.
- Tools → Diff/Merge: A utility to examine/repair the differences between two personality files. This will take two inputs, which may be two separate TKF files or may reference the current editor defaults/values. As with other visual diff/merge tools, it will then allow you to view the line-by-line (or, in our case, node by node) differences between the two personality files, and optionally replace the current editor values with the merged result.

4 Author Mode

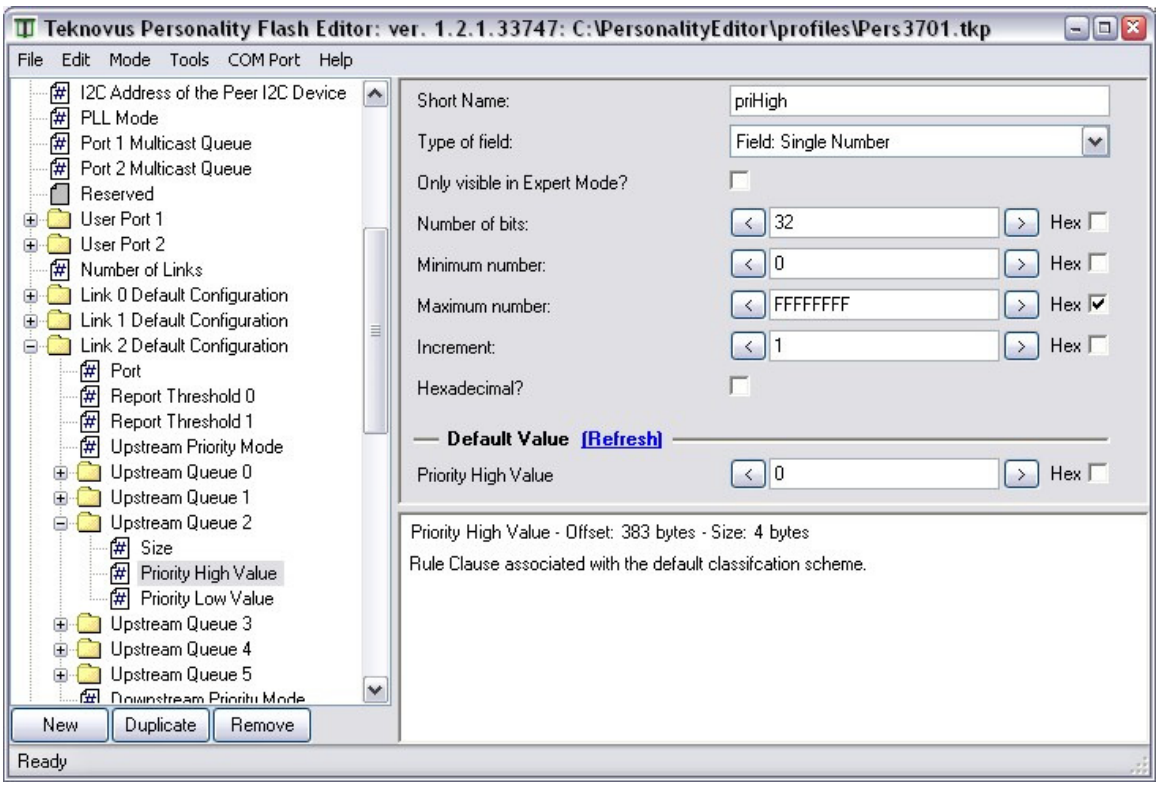
4.1 Creating Personality Profiles

Author mode is used to create the personality profiles which define the personality files used in user mode to control creation of the personality file. A profile is a description of a binary personality file which defines the locations of fields, type of input for those fields, the default value, and description of the field. This information is stored in a profile. User mode uses these to ensure validity of the format and values of the personality file being edited.

The layout of author mode is similar to that of User mode. The difference is that Author mode edits the profile, while User mode edits the personality file itself. Underneath the field list window, there are "Add", "Delete", and "Duplicate" buttons to add, delete and duplicate fields, respectively. The add button will add a new field to the personality profile before the selected field. Delete will remove the currently selected field. Duplicate will create another copy of the current field.

In author mode, the field value window can be used to define the type of input the user can provide, as well as the size of the field and the default value. Text in the description window can also be edited in author mode. The "short name" field in the field value window defines a shorter name for the selected field. These shorter names are used by the command-line mode (see 6).

In author mode, it is possible to rearrange the personality tree in the left-hand panel. Just click and drag a personality field (structure or scalar) to another position. Be careful only to rearrange the structure of the personality file when it is necessary. You can also rearrange the fields by holding down the shift key and pressing the arrow keys.



Note: Fields which are named “reserved” or “padding” are only visible in author mode, you cannot see them in user mode. “Reserved” fields are undefined fields that may be utilized at a later date, and “padding” fields are just fields that were added to the profile due to the way the device stores its values in memory, caused by the processor and data bus size. These fields are meaningless to User mode.

4.1.1 Field Types

There are a number of types for the fields you create in author mode. With any type, you can set whether or not the field must be viewed in expert mode. Also, with all of the scalar field types, you can set how many bits wide the field is. In addition to those options, there are custom options per field type. Here is a brief description of the types:

1. **Structure**: A structured personality field, which can contain other personality fields (structure or scalar). The only extra option for this type is whether or not to show all of the subfields when this field is selected in user mode.
2. **Checkbox**: A field that appears as a checkbox. You can set the numerical values for the checked and unchecked states.
3. **Dropdown List**: A field that appears as a drop-down list. You can set the number of elements in the list, and what number each list element stands for.
4. **List of Numbers**: A field that appears as a number of numeric-input boxes, laid out in a row. You can choose how many elements the list contains, the bit width per list item, and whether or not the input should be in hexadecimal.
5. **Single Number**: A field that appears as a numeric input field, with up and down buttons. You can choose the minimum and maximum numbers, as well as the increment when clicking the up/down buttons. Also, you can choose whether or not the input is in hexadecimal by default, even though the user can specify this by clicking the “hex” checkbox beside the field.
6. **Raw Input**: A simple raw hexadecimal field, padded with 0s if the input is too short.
7. **Hex/ASCII Input**: A field which is displayed as raw hexadecimal bytes, and also the ASCII equivalent of those bytes. You can edit either field, and the other field is updated accordingly. You can select the number of bytes you wish the field to occupy.

8. **Linked Field**: A field which takes its view settings from another field. You can set which field's settings you'd like to base the current field on. Note that this is only for view settings; it does not link the values or defaults for these fields. This is convenient when you have set up a complex field, such as a large dropdown list, and there are many identical fields which should have the same list. In this case, you can create the list once, then make all of the identical fields link to the first field, instead of recreating the list repeatedly.

4.2 Spell-Check

In author mode, the Personality Editor has the ability to spell-check all of the fields in the personality tree. In specific, the "description" field in each node is spell-checked by default when in author mode. You will notice as you type that incorrectly spelled items will turn red and become underlined. You can right-click these misspelled words to replace them with suggested alternatives, add the word to your dictionary or ignore the misspelling for this Personality Editor session.

In order to provide this feature, the Personality Editor utilizes "GNU Aspell" (aspell.net). The default distribution of the Personality Editor should come with a top-level subdirectory named "Aspell". If this directory is not included, or is incorrectly formed, the spell-check function will become disabled and effectively invisible to the user.

4.3 Menu Options

In addition to some of the menu options available in user mode, there are a number of menu options that are only available in author mode:

- File → New Profile: This creates a new, blank personality profile.
- File → Save Profile/Save Profile As: This saves the profile (including both structure and values) in XML or compressed XML format.
- Options → Chip Options: In author mode, this lets you edit all of the chip parameters for this profile (chip ID, personality version, etc).
- Tools → Values->Defaults: This will change all of the default values in the profile to match their current values.
- Tools → Find Duplicate Paths: This tool allows you to search for multiple nodes that share the same "short name" path string. This path string should be a globally unique identifier, used to specify one node alone. This is especially important for use with command-line mode (see 6).
- Tools → Spell Check: This allows you to spell-check every field in the entire structure. Unlike the inline spell-check (see 4.2), this will also spell check field names and any other appropriate metadata.

5 Important Notes

5.1 Endianness of Sub-Fields

Fields may be one or more bits wide. The list of fields may have several adjacent fields that are smaller than one byte (8 bits) wide. Within a byte, the subfields are defined from the most significant bit to the least significant. As an example, consider a structure that has sub-fields A, B and C, in that order, where A is 3 bits wide, B is 8 bits, and C is 5 bits. Thus , these three fields occupy two bytes in the personality file. Numbering the bits in this 16-bit portion of the file from 15 as the most significant to 0 as the least significant, the fields would be stored in the personality file as follows:

```
[ --  A -- ] [ -- -- --  B -- -- -- -- ] [ -- --  C -- -- ]
  15 14 13     12 11 10  9  8  7  6  5      4  3  2  1  0
```

6 Command-Line Mode

User mode allows an engineer to create a personality file for a particular product. Since the personality flash contains values that are unique to individual instances of a product, such as the MAC address, or serial number information in the vendor extended region, it is often convenient to be able to change just a few fields in an existing personality image. User mode could be used to create a "base" or "template" personality file containing most of the desired values. The command-line mode could then be used to alter just the fields that change, and produce a new personality field customized for one particular instance of a product.

Command-line mode would typically be called by some sort of automated script. This script would consult whatever files or databases are necessary to issue values for MAC addresses, serial numbers, and so on.

When Command-line mode needs a numeric input, numbers can be input in either decimal, or in C-style hexadecimal (base 16) designated by a 0x prefix. Hexadecimal values can be of any length, even longer than 32 or 64 bits. The storage for all values will be rounded to the next byte. For example, the value 0x1517, while it only requires 13 bits, would be considered a 16 bit (2 byte) value. Also, to enter numbers which are not an even number of bytes long (such as a 1 or 2 bit value), you can use syntax like '010. That is, an apostrophe character followed by a binary number composed of only 1s and 0s. This will allow you to set such fields as a single value within a compact bitmap.

6.1 Important Notes

When running the Teknovus Personality Editor in command-line mode, you must run the executable file which does not optionally install the .NET framework. In the default distribution package, you'll see two executable programs. "tkPersEditor.exe" is the main executable, and will optionally install the .NET framework if it is not detected on the current system. The second executable, "perseditor_exec.exe", will only run if the .NET framework is present. In order to use the command-line mode of the Teknovus Personality Editor, you must feed your command-line arguments to "perseditor_exec.exe". See the "readme.txt" file that comes with the distribution for more details.

6.2 Command Line Arguments

Arguments to the personality editor are in the format below.

perseditor_exec [[<opts>] <tkfFile> <profile> <idValuePair>*]

<tkfFile> is the personality file you'd like to modify. This file must be in .tkf format. Note that this file will have various fields changed (including the CRC automatically), and then will be overwritten by the program.

<profile> is a personality profile, as created using the GUI in user mode (see section 3, User Mode). This file must be in .tkp format. This file will be used as the reference file for the structure of the personality region. Endian-ness is also assumed based on this file.

<idValuePair> is a list of zero or more pairs values within the personality file you'd like to change. It is in the format: (id val) (id val) (id val) … (id val).

> **id** is an alphanumeric identifier as specified in the personality profile.
> **val** is a number representing the new value.
>
> **id** must correspond to a "short name" of one of the fields in the personality profile. Short names can be changed for every field of the personality profile in user mode (see section 3, User Mode). This value will be used to calculate the position in memory of the field, as well as the width of the field. Note that this will not work properly if either the bit offset or the number of bits in the field do not lie on byte boundaries.

Important: if the the "short name" you are looking for within the profile is buried within one or more structure fields, you must prefix the name with the names of those structures, separated by periods. For example, say you want to change the value of the "Size" field within the "Upstream Queue 0" structure, which is within the "Link 1 Default Configuration" structure. Assuming these fields have the short names "size", "upQ0", and "link1", respectively, your **`id`** value should be "link1.upQ0.size".

Note: if you specify the `-d`, `-upload` or `-verify` flags, it is acceptable to omit these fields.

<opts> can be any combination of different option flags. The different flags you can specify are:

Flag	Effect
`-r`	This will reverse the position of the 0 padding in long data fields. By default, in big-endian mode, values smaller than the field size are padded with 0s in the most significant bytes. This option pads the value in the least significant bytes, with the value stored in the most significant bytes. For example, without this flag, in big-endian, if you stored 0x1517 in a 4 byte field, the value stored would be 0x00001517. If you use this flag, the value would be 0x15170000. In little-endian mode, the padding is in the least significant bytes by default, and this flag causes the value to be padded in the most significant bytes.
`-d`	Opens the supplied profile and creates a .tkf file with all of the default values. If a TKF filename is specified, any existing TKF file with that name will be overwritten. Otherwise, the generated .tkf file will have the same filename as the profile, but with .tkp replaced with .tkf. If this flag is used, the <idValuePair> will have no effect and should be omitted.
`-readtkf <fields>`	This will cause the Personality Editor to read the specified fields from a TKF file, printing the values to standard out. This field must be followed by the desired fields to read, <**`fields`**>, which is a comma-delimited list of absolute field identifiers (AKA "short names").
`-upload`	This can be used to upload the new file directly to a device as soon as the values have been changed. If this flag is used, the flag `-serial` must also be supplied, with an appropriate serial port.
`-verify`	This causes, once other operations have been completed, the personality to be read back from a device and checked against the current file. The return code from the program indicates if this comparison was successful (see 6.3). This flag can be used with or without additionally specifying `-upload`. If this flag is used, the flag `-serial` must also be supplied, with an appropriate serial port.
`-download`	This can be used to download the personality file from a device and store it in the given TKF file. The TKP file supplied will be used as a basis for the download. If this flag is specified, the given TKF file need not already exist, and if it does it will be written over with the personality file from the device. If this flag is used, the flag `-serial` must also be supplied, with an appropriate serial port.
`-nosave`	This will cause the changes to not be committed to the TKF file. This only makes sense for use with `-upload` and `-verify`, where the changes will be committed to the device, but not saved to the TKF file. This flag is designed to facilitate production lines, where one can have a master TKF file, and change a few values before sending it to a device, without altering the master TKF file.
`-serial <port>`	This specifies a serial port to communicate with for the `-upload` and `-verify` operations. This field must be followed by a port specifer, <**`port`**>. For serial ports attached to the local PC, <**`port`**> must be in the format "*COM#*", where # is the number of the COM port. To access a serial port via Telnet, use the format "*X.X.X.X*:*P*", where *X.X.X.X* is an IPv4 address and *P* is the port number.

6.3 Return Codes

When run from the command-line, the Teknovus Personality editor returns a non-zero error code if an error occurred or zero if the change/upload/verification was successful.

Error Code	Meaning
0	**OK**: no problems encountered.
1	**Unspecified Error**
2	**Format Error**: incorrectly formatted command-line arguments.
3	**I/O Error**: such as opening and closing files.
4	**TKP/TKF Mismatch**: the TKF file could not be applied to the TKP file specified.
5	**ID/Value Pair Error**: one of the (id val) pairs are incorrect.
6	**Upload Error**: error uploading TKF to device with `-upload`.
7	**Verification Error**: personality on target device does not match current file (`-verify`).
8	**Download Error**: error downloading TKF from device with `-download`.

6.4 Examples

```
perseditor_exec Pers3701.tkf Pers3701.tkp (eponMac 0x544B37010033)
  (userMac 0x544B37010033) (extId 0xFFFFFFFFFFFFFFFF0CFFFFFFFFFFFFFF)
```

This command will open the personality file Pers3701.tkf, and change the Base EPON MAC Address, Base User MAC Address, and Vendor Extended ID to the values specified. Note that this assumes “eponMac”, “userMac” and “extId” are the “short names” given to the appropriate fields in the personality profile.

```
Perseditor_exec -serial COM3 -upload -verify -nosave Pers3711.tkf
  Pers3711.tkp (eponMac 0x544B37110033) (userMac 0x544B37110033)
```

This command will open the personality file Pers3711.tkf, change the Base EPON MAC Address and the Base User MAC Address to the values specified, and send the result to a TK3711 ONU connected to the local PC’s COM3 port. The personality will then be downloaded from the ONU to make sure they have taken place. These changes will not be saved to Pers3711.tkf.

6.5 Other Uses

The command-line has a few more uses than the one mentioned above.

Command	Action
`tkPersEditor`	Opens the GUI Personality Editor.
`tkPersEditor <profile>`	Opens the GUI Personality Editor with the supplied profile.

7 Strategies and Tips

One possible strategy for programming flash is to use User mode to create a template personality file that contains most of the values for a particular product. Command-line mode would then be used to alter that template for each individual instance of that product.

Note that command-line mode overwrites the file supplied on the command line. With a template scheme, the caller should first copy the template to a working file specific to an individual instance, and then invoke the command-line argument to alter the specific file. Otherwise, the template itself would be altered.

If all instances are going to modify the same fields, and every one of those fields is set for each instance, then it may be acceptable to simply use one file and modify it repeatedly.