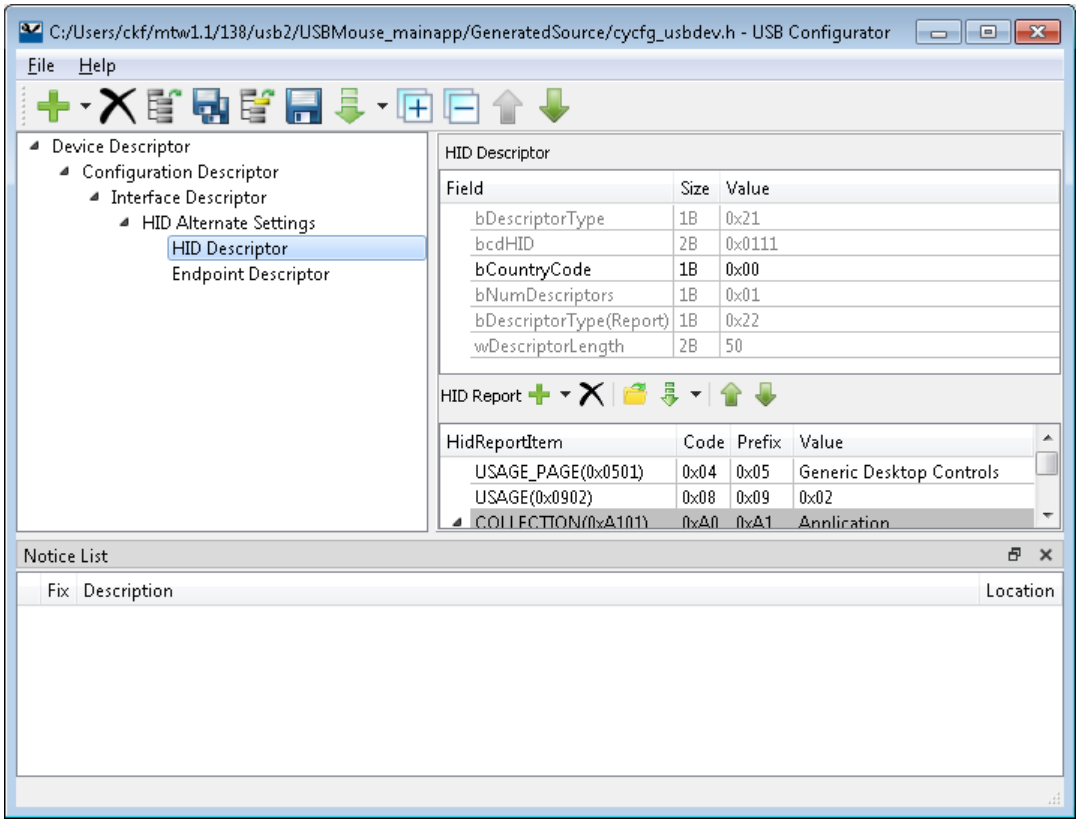
ModusToolbox™ USB Configurator Guide

Version 1.1

Overview

The USB Configurator is a configuration tool included in the ModusToolbox software. Use the USB Configurator to configure USB Device descriptors. The list of supported USB descriptors is provided in the Supported Descriptors section. After configuring and saving a USB Device descriptors, the USB Configurator generates header (.h) and source (.c) files that store USB Device descriptors and other information (see Code Generation) which is used by USB Device middleware configuration and operation.



Launch the USB Configurator

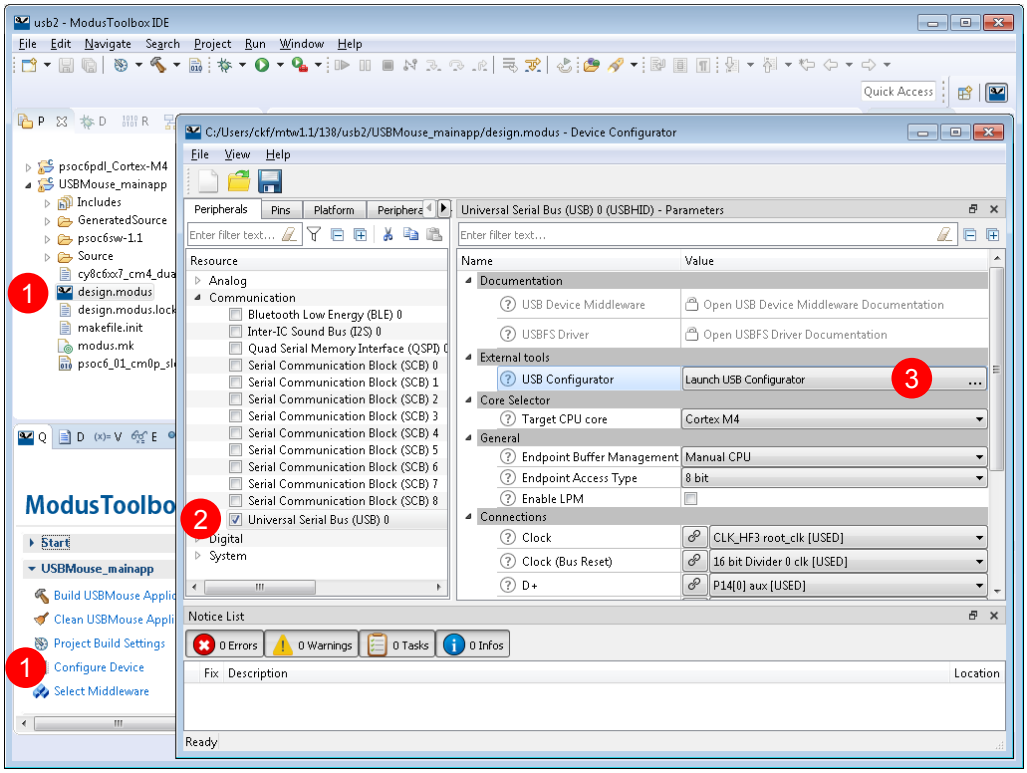
The USB Configurator is a stand-alone tool that contains menus, toolbars, and two panes used to configure device descriptors. You can run the configurator from, and use it with, a ModusToolbox IDE application. You can also run it independently of the ModusToolbox IDE. You can use the generated source with a ModusToolbox IDE application, or in any software environment you choose.

From a ModusToolbox IDE Application

1. Launch the Device Configurator from an application within the ModusToolbox IDE using the *design.modus* file.
2. Open the **Peripherals** tab and enable **Universal Serial Bus (USB)** peripheral by clicking the enable check box.

Note After enabling USB peripheral the unresolved the tasks appear in the Notice List Pane. These tasks can be left unresolved. Refer to the *Device Configurator Guide* section Notice List Pane for more information.

3. On the USB peripheral **Parameters** pane, click the **Launch USB Configurator**



This method of launching the USB Configurator passes the *design.modus* file path and provides access to the USB peripheral configuration and connection information. When you save changes in the USB Configurator, it updates the USB peripheral parameters that depend on USB Descriptors. It then generates/updates the *cycfg_usbdev.c* and *cycfg_usbdev.h* files in the ModusToolbox IDE application's "GeneratedSouce" folder.

Note When the USB Configurator saves the *design.modus* file, a notice may display indicating errors, warnings, tasks, or notes available in the Device Configurator Notice List Pane. In such cases, close this notification and the USB Configurator. Then, use Notice List Pane to address notices.

Independent of the ModusToolbox IDE

To launch the USB Configurator independently, navigate to the install location and run the executable. The default install location for the USB Configurator on Windows is:

<user_home>\ModusToolbox_<version>\tools\usbdev-configurator-<version>

For other operating systems, the installation directory will vary, based on how the software was installed.

When run independently, the configurator opens without a configuration file. Load the configuration file using the **Open** menu. If the file does not exist, then it will be created when clicking the **Save** button.

This command opens the GUI with the specified configuration file. If the file exists, the configurator loads the existing configuration. If the file does not exist, the configurator creates a new file when clicking the **Save** button.

From the Command Line

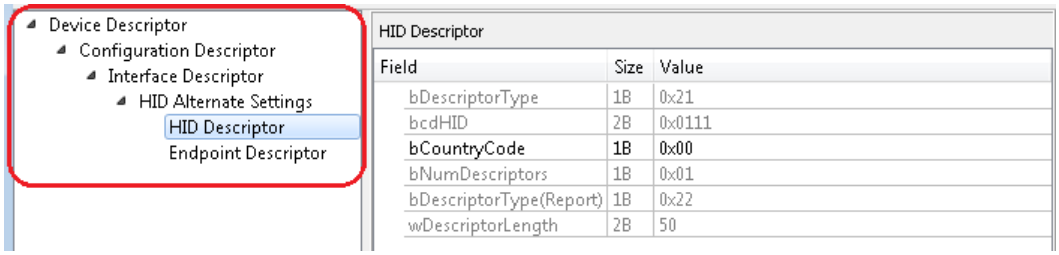
You can run the configurator from the command line. However, there are only a few reasons to do this in practice. The primary use case would be to re-generate source code based on the latest configuration settings. This would often be part of an overall build script for the entire application.

For information about command line options, run the configurator using the `-h` option.

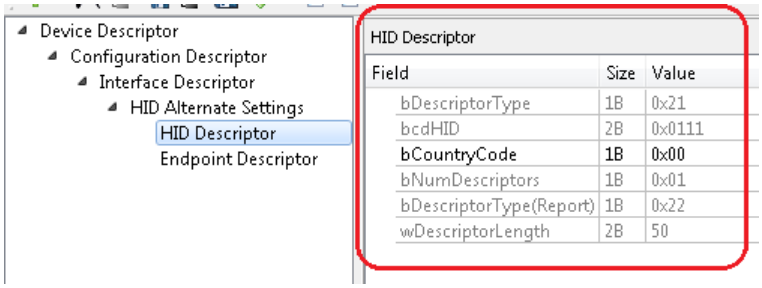
Quick Start

This section provides a simple workflow for how to use the USB Configurator.

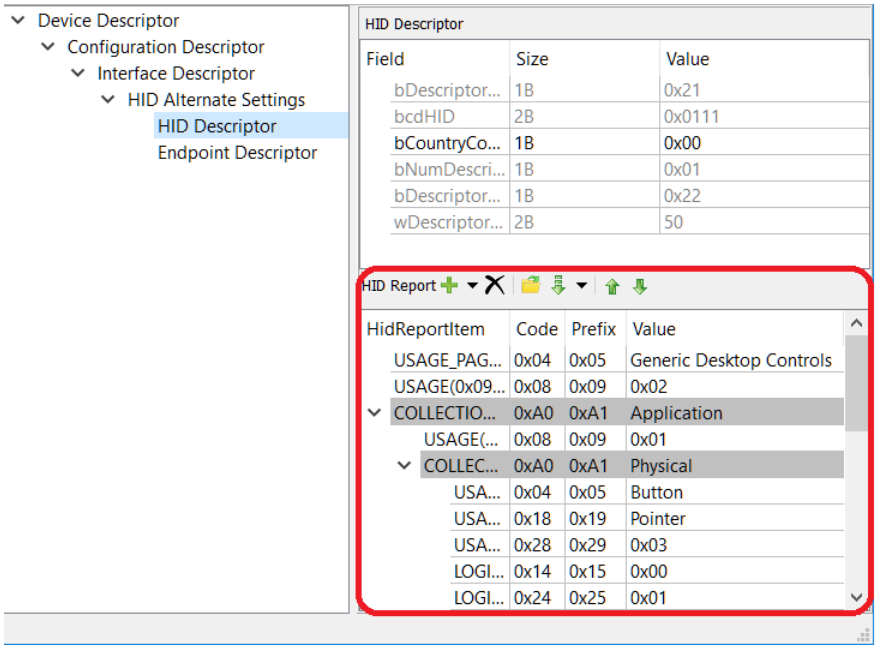
1. Launch the configurator.
2. Configure the device descriptors hierarchy in the **Device Descriptor** pane. See Descriptors section.



3. Configure device descriptor parameters in the **Parameter** pane.



4. The **Parameter** pane contains a sub-pane for **HID descriptor** (**HID report pane**).



5. Save the configuration.

   The USB Configurator generates code into the “GeneratedSource” directory in your ModusToolbox IDE application, or in the location you specified for a non-ModusToolbox IDE application. That directory contains the source (.c) and header (.h) files with relevant firmware which is used by USB Device middleware configuration and operation.

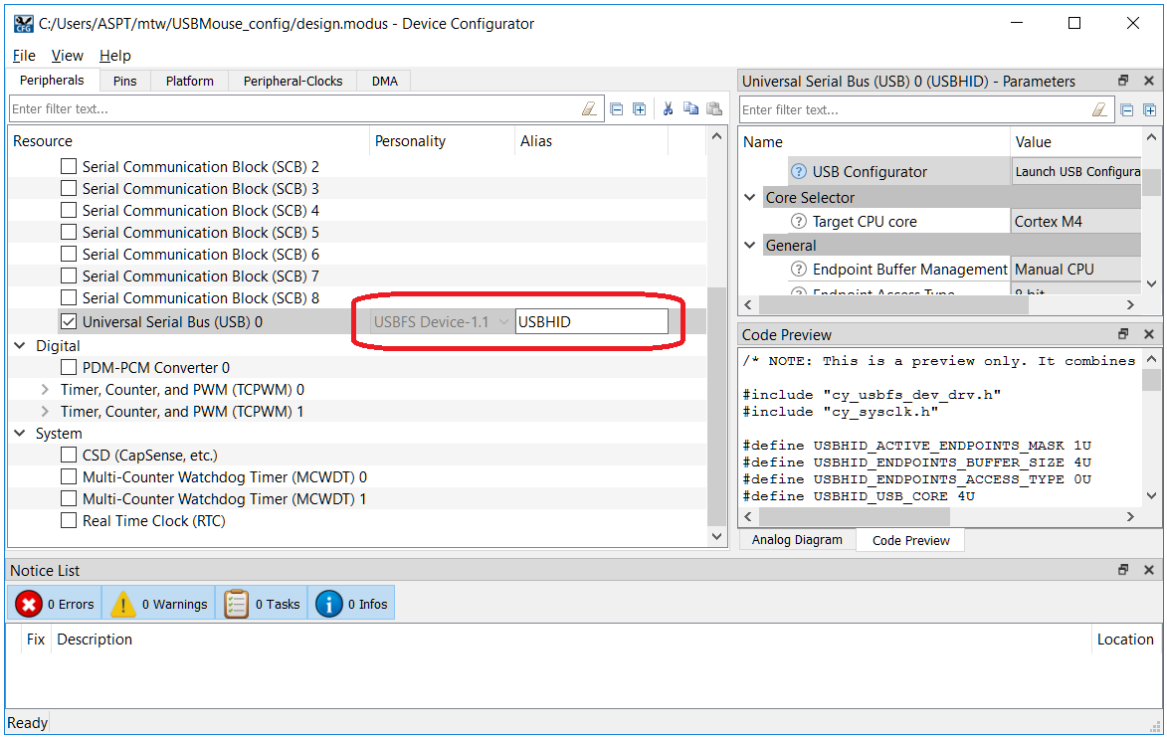
6. Use the generated structures as input parameters for functions in your application.

Code Generation

The USB Configurator generates header (.h) and source (.c) files that contain relevant firmware which is used by USB Device middleware configuration and operation. It contains arrays to store USB Device descriptors, structures that help middleware to access descriptors, middleware and classes configuration structures and set of defines. When the USB Configurator is launched from a ModusToolbox IDE application the generated files *cycfg_usbdev.h* and *cycfg_usbdev.c* are located in the *GeneratedSource* folder next to the **.modus* file. However, when tool is launched independently, the files names and location can be changed. Refer to the USB Device Middleware Library for more information about this code. There is a link to the API documentation from the Device Configurator.

Device Alias

A device alias can be used to generate device-specific code (defines and struct names, comments, errors, etc.).



For a stand-alone flow, the device Alias can be passed via command parameter **-n** or **–name**. If the device alias is not specified, then default device alias will be used: **usb_0**.

For example, using **USBHID** as the device alias will generate the following code:

```
#if !defined(CYCFG_USBDEV_H)
#define CYCFG_USBDEV_H

#include <stddef.h>
#include "cy_device_headers.h"

#if defined(CY_PSOC_CREATOR_USED)
    #include "USBHID.h"

    /* Enable code for both cores */
    #define USBHID_USB_CORE __CORTEX_M

    /* Endpoint Buffer Size */
    #define USBHID_ENDPOINTS_BUFFER_SIZE 4U

    #if (USBHID_ACTIVE_ENDPOINTS_MASK != 0x1)
    #error The USB Device peripheral parameter Endpoint Mask expected value is 0x1.
Change Endpoint Mask to this value or run the USB Configurator using PSoC Creator and
adjust the number of active endpoints.
    #endif
…
#if defined(USBHID_USB_CORE) && (USBHID_USB_CORE == __CORTEX_M)

#if defined(__cplusplus)
extern "C" {
#endif

#include "cy_usb_dev.h"
```

```
#include "cy_usb_dev_descr.h"
#include "cy_usb_dev_hid.h"
#include "cy_usb_dev_cdc.h"

/* Number of USB Device */
#define USBHID_NUM_DEVICES           1

/* Class specific defines */
#define USBHID_AUDIO_CLASS_SUPPORTED 0U
#define USBHID_CDC_CLASS_SUPPORTED   0U
#define USBHID_HID_CLASS_SUPPORTED   1U


/* Array of USB Devices */
extern const cy_stc_usb_dev_device_t USBHID_devices[USBHID_NUM_DEVICES];
```

Note The same configuration can generate different code depending on the device alias for different flows. For example, code generated in standalone flow without specified Device Alias will be with default Device Alias. Code generated from ModusToolbx with specified Device Alias will be different. In this case user must open The USB Configurator and save configuration one more time. Generated header contains a set of pragma errors to worn user that generated code and configuration is out of synchronization.

Menus

The menus include:

- **File** – Provides basic commands to open, close, and save files, as well as exit the configurator.
  - **Open (Ctrl+O)** – Open an existing configuration header (.h) file. The current file, if any, will be closed.
  - **Save (Ctrl+S)** – Save the current file and generate code for the configured device descriptors.
  - **Exit –** Close the configurator.
- **View** – Provides options to show/hide the Notice List and Toolbar.
- **Help** – Provides access to this document and an About box.

Toolbars

The toolbars include:

- **Descriptor Toolbar**: Provides basic commands to open and save files and to configure descriptors hierarchy.

  - **Add descriptor –** Create a new descriptor under the selected one.
  - **Delete descriptor –** Delete a selected descriptor.
  - **Open (Ctrl+O)** – See Menus.
  - **Save (Ctrl+S)** – See Menus.
  - **Load Descriptor –** Used to load a descriptor from a file.
  - **Save Descriptor –** Used to save a descriptor to a file.
  - **Import Descriptor –** Select a descriptor from the pull-down menu to import.
  - **Expand all –** Expand all items in the descriptor tree.
  - **Collapse all –** Collapse all items in the descriptor tree.
  - **Move Up/Down –** Used to arrange the order of items in the tree.
- **HID Report Toolbar**: Provides basic commands to configure the HID descriptor report.

  - **Add report item –** Create a new report item for the current HID descriptor.
  - **Delete report item –** Delete the selected report item for the current HID descriptor.
  - **Open –** Open a report file generated by HID Descriptor Tool.
  - **Import report item –** Import a report item for the current HID descriptor.
  - **Move Up/Down –** Used to arrange the order of items in the tree.

Panes

The USB Configurator contains two panes that display information about descriptors and their parameters:

Descriptors

This pane shows the descriptors hierarchy.

Note All descriptors have their own hierarchy, and they can be created when their parent is selected. However, the **Device Descriptor** is a root descriptor and it has no parent. The **Device Descriptor** can be created from any descriptor and will be added to the end of the tree.

To add specific descriptors such as **CDC Descriptor** or **HID Descriptor** special parent descriptor should be added:

- CDC Interface Descriptor should be added for CDC Descriptor
- HID Alternate Settings should be added for HID Descriptor.

Parameters

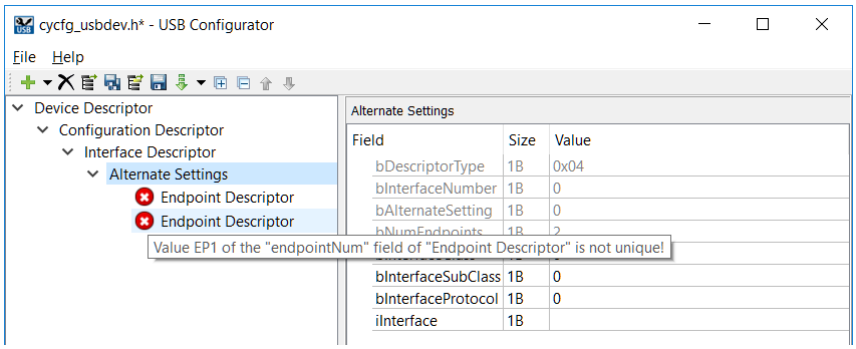
This pane shows configuration information for the selected descriptor.

Note The Parameter pane has different controls to edit different parameters (text box combo box, or multi-line text box). Most parameters have a text box as the editing control.

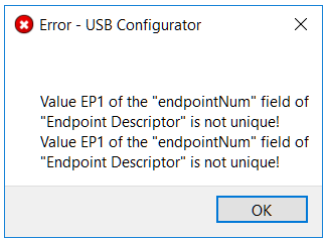
- **Vendor-defined items** – Some string parameters or **HID report items**, such as **iSerialNumber**, have a combo box with a list of predefined items. The “Empty” value is selected by default. To specify a value not on the list, select “Vendor-defined.” The combo box will change to a text box to type an appropriate value. To return to a combo box, erase the value and leave the control.
- **String pool** – Parameters such as **iChannelNames** in **AC Processing Unit** of **Audio Interface Descriptor 1.0** support string pool and require a multi-line text box. To insert several strings, use an end line separator.
- **Read only parameters** – There are two types of read-only parameters: predefined **bDescriptorType** or auto-calculated **bConfigurationValue**.
- **Array parameters** – Parameters such as **bSubordinateInterface** in union with **Communication Alternate Settings** is an array. Use “;” to separate elements.
- **Hexadecimal/Decimal** – When a value starts from “0x,” it is parsed as a hexadecimal value. In other cases, it is parsed as a decimal.
- **Map and bit fields** – Some parameters, such as **bmAttributes**, are read-only by themselves. However, they can be inserted using the related bit fields below them in the parameters list. **Bit fields** have a size 0B. Their name starts from the field name of which they are a part, plus bits for which they are responsible. For example, **bmAttributes(1-0): Transfer Type**.

Errors

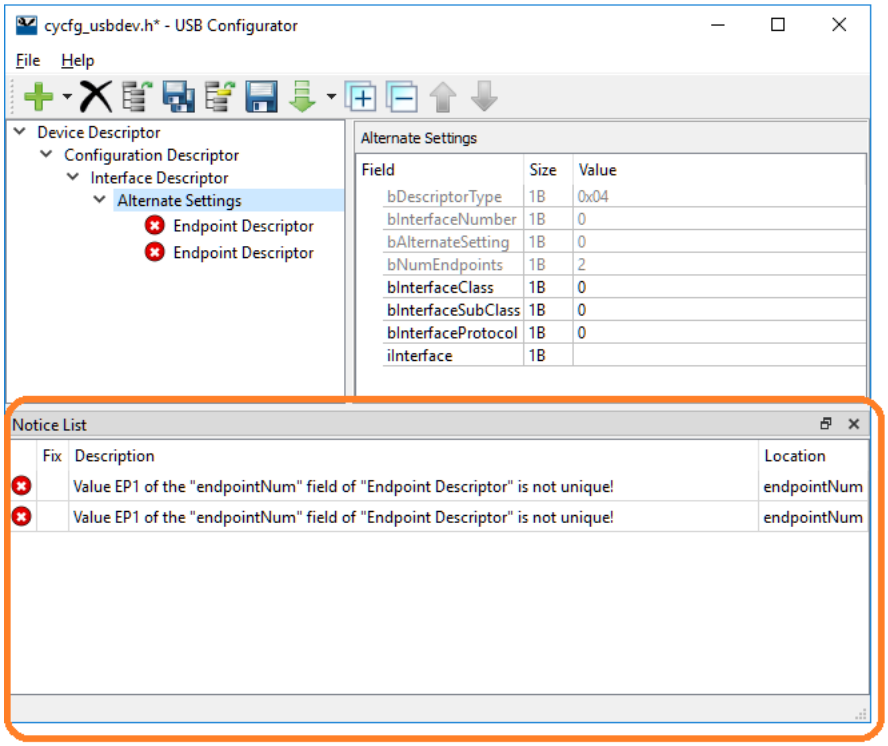
The USB Configurator has a validation system which can identify errors. Notification is made via error icons with error message in tooltips.



When you try to save the configuration with errors, a message displays indicating the problem that needs to be fixed.



All errors are shown in the Notice List:



Supported Descriptors

All supported descriptors are described by the USB Implementers Forum, Inc. You can find more details at http://www.usb.org.

List of supported descriptors:

- Device, Configuration, IAD (Interface Association Descriptor), Interface, Endpoint
- Audio v1.0 and v2.0 descriptors
- CDC Communication descriptors:
  - Header Functional descriptor
  - Union Functional descriptor
  - Country Selection Functional descriptor
  - Call Management Functional descriptor (PSTN)
  - Abstract Control Management Functional descriptor (PSTN).
- HID descriptor and HID Report descriptor
- BOS descriptors (including Container ID and USB 2.0 Extension descriptors)
- Microsoft OS descriptors v1.0

Note The Interface descriptor is represented by two items to build tree structure: the Interface descriptor with empty parameters pane and alternate settings that contains all Interface descriptor parameters.

Note Class-Specific AS Encoder/Decoder Descriptor from USB Audio v2.0 do not supported.

Known Issues, Limitations, and Workarounds

The USB Configurator supports import from the HID Descriptor Tool. Current version 2.4 contains an error related to strings. Per spec HID1.11 String items should have such values:

- String Index 0111 10 nn
- String Minimum 1000 10 nn
- String Maximum 1001 10 nn

But the HID Descriptor Tool generates:

- String Index 0110 10 nn
- String Minimum 0111 10 nn
- String Maximum 1000 10 nn

Before import, these items should be manually fixed in the file generated with the HID Descriptor Tool.

There are several possible flows to run the USB Configurator. To migrate configurations between flows, you must regenerate code for each flow.

References

Refer to the following documents for more information, as needed:

- Device Configurator Guide
- ModusToolbox IDE User Guide
- Cypress USBFS Device Middleware Library Documentation
- http://www.usb.org

Version Changes

This section lists and describes the changes for each version of this tool.

Version	Change Descriptions
1.0	New tool.
1.1	Updated the icons to be standard. Added Notice List.