Name:

`smt` standalone manufacturing tool

Synopsis

```
smt   option1 option1_argument option2 option2_argument…
```

Description

`smt` is a standalone executable tool that provides command line capability to flash Silicon Labs fixed function devices with desired configuration values. The tool takes as input a configuration file in text format that was created using the Xpress Family configuration tools provided in Simplicity Studio. All devices must be of the same device type (e.g. CP2102N). There is a separate SMT exe for each library.

`--help`

Output this page.

`--reset`

Performs soft resets of all connected devices equivalent to a USB disconnect/reconnect.

`--device-count <decimal number>`

Mandatory. Specifies how many devices are connected. Programming process will not start if it finds a different number of devices or fails to open them. Verification process will endlessly retry until it can verify this number of devices.

`--serial-nums {X Y Z…}|GUID`

Used to specify that serial numbers should be written to the devices. Serial numbers are only written if this option is provided.

`{X Y Z}` Specifies list of values to be used as serial numbers to be programmed to the device. The number of serial numbers provided must match the number of devices to be programmed or an error will be reported. If multiple devices are connected, the list must contain the same number of serial numbers as connected devices.

`GUID` SMT will automatically generate a unique serial number to be written to connected devices, using a platform-supplied UUID generation function.

`--set-and-verify-config config_file_name`

Programs and verifies each device using the configuration provided in config_file_name. Prints the list of serial numbers programmed.

`--set-config config_file_name`

Programs each device using the configuration provided in config_file_name. Prints the list of serial numbers programmed.

```
--verify-config config_file_name
```

Verifies each device using the configuration provided in config_file_name. `smt` will report an error if any connected devices fail verification."–serial-nums GUID" can't be used with this option; instead pass the numbers reported earlier by the programming process.

Example Usage

Example 1: The following example assumes multiple same device type devices are connected to USB. The following commands will program and verify the device contents in one invocation of smt. Serial numbers will be automatically created and written to the devices.

```
smt   --device-count 3 --set-and-verify-config my_config.txt –
      serial_nums GUID
```

Example 2: The following example assumes multiple same device type devices are connected to USB. The following commands will write and verify the device contents in three separate invocations of smt. Serial numbers are provided by the user to write to the device and provided again by the user for verification.

```
smt   --device-count 3 --set-config my_config.txt –serial_nums
      {20 21 22 23}
smt   --device-count 3 --reset
smt   --device-count 3 --verify-config my_config.txt –serial_nums
      {20 21 22 23}
```

Configuration lock

All devices support permanent configuration locking. CP210x can only lock all parameters at once, while other devices can lock parameter separately. SMT, however, always locks all parameters at once. Also, the device to be programmed must have all parameters unlocked, otherwise SMT rejects it.

Configuration File

Has the same syntax as in the CFG wrapper, except that names are simpler and there is no device index. If a customization parameter is omitted, SMT will not customize or validate it.

Parameters supported by all devices

`FilterPartNumByte`
`FilterVidPid`
`VidPid`
`MaxPower`
`PowerMode`
`DeviceVersion`

```
ProductStringAscii
```

CP210x-specific parameters and support by device type

```
ManufacturerString<T>         , , , ,8, ,
ProductStringUnicode         2,3,4,5,8,9,2N
InterfaceString<T><N>         , , ,5,8, ,
FlushBufferConfig             , ,4,5,8, ,
DeviceMode                    , , ,5, , ,??
BaudRateConfig               2,3, , , ,9,
PortConfig                    ,3,4, , , ,??
DualPortConfig                , , ,5, , ,
QuadPortConfig                , , , ,8, ,
Config                        , , , , , ,2N
```

CP2110/4-specific parameters and support by device type

```
ManufacturerStringAscii    10,14
FlushBufferConfig          10,14
PinConfig                  10,14
RamConfig                    ,14
OTP                          ,14
```

CP2130-specific parameters

```
ManufacturerStringAscii
TransferPriority
PinConfig
```

Notes

FilterPartNumByte and FilterVidPid must always be specified, in the order shown. These are NOT customization parameters, they are used to select devices to work on. Most manufacturing libraries require VID/PID to open the device. The part number is also used to make sure the configuration file parameters are applicable to the specified device type.

The <T> suffix must be either “Ascii” or “Unicode”.
The <N> suffix must be either “0” or “1” for CP2105 or from “0” to “3” for CP2108.

Sample CP2105 file

FilterPartNumByte { { 05 } }
FilterVidPid { { 10c4 } { ea70 } }
VidPid { { 1234 } { 5678 } }
MaxPower { { 32 } }
PowerMode { { 00 } }

```
ProductStringAscii { { 43 50 32 31 30 35 20 44 75 61 6c 20 55 53 42 20 74 6f 20 55 41 52 54 20 42 72 69
            64 67 65 20 43 6f 6e 74 72 6f 6c 6c 65 72 } }
```