Application Program Interface (API) for CPS Model 2600 Power Supplies

Introduction

CPS provides a free software development kit (SDK) for CPS model 2600 power supplies. The SDK includes an API that allows application software to communicate with model 2600 devices. The API is compatible with all variants of model 2600 and supports multiple devices connected to a single computer. This document describes API functions and operation, and is intended for software developers who create application programs for model 2600 devices.

Windows SDK

The Windows SDK includes the following files:

- `c2600.dll` – API executable. This must reside in the application program's directory or in the system's DLL search path. Note that this is a 32-bit DLL, which means it can be used on both 32- and 64-bit machines, but only with 32-bit applications.
- `c2600.lib` – Linker library for the API executable. This is used by the object file linker when building application programs.
- `c2600api.h` – Header file containing API constants and function declarations. This must be included in C/C+ source code files that call API functions.
- `example.c` – A simple application example, written in C, which shows how to call API functions.
- `c2600config.exe` – Configuration utility; see below for details.
- A demo program, `2600demo.exe`, and its source files: `2600demo.c`, `2600demo.h`, `2600demo.rc`, and `c2600.h`.

Device number

Most of the API functions use a device identifier argument (`devnum`) to specify a particular power supply. Each device must be assigned a unique `devnum` value in the range 0 to 7. The device identifier is assigned to a device when the `C2600_OpenDevice` function is called.

Error codes

Unless otherwise noted, the value returned by an API function is an error code. Zero is returned if a function executes without error, otherwise a non-zero value is returned. When a non-zero value is returned, it may be one of the error codes listed in c2600api.h or it may be a system error value specific to the operating system.

Configuration program

The `c2600config.exe` program is provided for users who are upgrading from an earlier API version. You must run this program if, while running your application software, the `C2600_OpenDevice` function returns `C2600_ERR_INVALID_RANGE`.

The program will ask you to enter two values, `vrange` and `irange`, which specify the power supply's full-scale output voltage (in Volts) and current (in microamperes), respectively.

The entered values must match the specifications listed on the power supply data sheet. To ensure this is done correctly, first verify that the data sheet and power supply have matching serial numbers. Then obtain the full-scale voltage from the data sheet's "output voltage swing" specification and the full-scale current from the "I Limit" specification.

For example, if the data sheet "output voltage swing" is "0 to 20 kV" and "I Limit" is "1 mA" then you would set `vrange=20000` and `irange=1000`.

API functions

C2600_OpenDevice

```
int C2600_OpenDevice(int devnum, int port);
```

This function must be called once for each device to assign it a device identifier and to enable subsequent communication with the device. No communications are possible with a device until it has been opened by this function.

`devnum` is the device identifier that will be assigned to the device. After calling this function, the assigned `devnum` value will be valid for any other API functions that accept an `devnum` argument.

`port` specifies the serial COM port to which the device is connected. In Windows systems, the value of this argument equals the COM port number minus one. For example, COM1=0, COM2=1, etc.

If the function returns `C2600_ERR_INVALID_RANGE` then you must run the c2600config.exe program.

Example:

```
  C2600_OpenDevice(0, 0);  // open device on COM1 and assign it devnum 0
```

C2600_CloseDevice

```
int C2600_CloseDevice(int devnum);
```

This function closes a device; it must be called once for each open device before closing the application program. After calling this, further communication with the device is prohibited.

C2600_SetVoltage

```
int C2600_SetVoltage(int devnum, double *volts);
```

This function programs the voltage at the HV output. The volts argument specifies the desired output voltage in Volts.

C2600_ReadMeter

```
int C2600_ReadMeter(int devnum, int meter, double *data);
```

This function measures one of the internal device meters and returns the meter value in `data`. The `meter` argument specifies the meter to read and also determines the units of the value returned in `data`:

`meter`	`data` units	Description
`C2600_METER_VOLTAGE`	Volts	HV output voltage
`C2600_METER_CURRENT`	Microamps	HV output current
`C2600_METER_TEMPERATURE`	Degrees C	Internal device temperature
`C2600_METER_SETPOINT`	Volts	HV control voltage

C2600_SetLimits

```
int C2600_SetLimits(int devnum, int vlimit, int ilimit);
```

This function establishes HV output voltage and current alarm limits. The limit values are specified as a percentage of the corresponding full-scale output value, with a value of 0 to 100. `vlimit` specifies the voltage limit as a percent of full-scale output voltage; `ilimit` specifies the current limit as a percent of full-scale output current.

C2600_GetRanges

`const char * C2600_GetRanges(int *vrange, int *irange);`

This function reads the HV output's voltage and current ranges. The full-scale output voltage is returned in `vrange` in Volts. The full-scale output current is returned in `irange` in microamps.

C2600_GetApiVersion

`const char * C2600_GetApiVersion(void);`

This function returns a pointer to the API version string.

C2600_GetFwVersion

`int C2600_GetFwVersion(int devnum, int *ver);`

This function reads the firmware version number of a device and returns it in `ver`.

C2600_GetStatusFlags

`int C2600_GetStatusFlags(int devnum, int *flags);`

This function reads status flags from a device and returns them in `flags`. See the header file for a list of status flags.

C2600_GetFaultFlags

`int C2600_GetFaultFlags(int devnum, int *flags);`

This function reads fault flags from a device and then clears the `C2600_STATUS_RESET` and `C2600_STATUS_COMERR` flags. The fault flags are returned in `flags`. See the header file for a list of fault flags.

C2600_SetWD

`int C2600_SetWD(int devnum, unsigned int msec);`

This function configures a device's watchdog timer. The `msec` argument specifies the timer interval in milliseconds. The maximum allowed value is 2550, which corresponds to a 2.55 second interval. Specify `msec=0` to disable the watchdog timer.

C2600_Reset

`int C2600_Reset(int devnum, int hardreset);`

This function sets the power supply output voltage to 0 V and clears all communication buffers. Set `type=0` for soft reset, or type=1 for hard reset.

C2600_GetErrString

`const char * C2600_GetErrString(int errcode);`

This function maps an API error code to an associated descriptive string. The return value is a pointer to the descriptive string associated with the errcode argument.

Example

```
int errcode = C2600_OpenDevice(0, 0);
if (errcode != C2600_ERR_NONE)
    printf("C2600_OpenDevice failed: %s\n", C2600_ErrorString(errcode));
```