POWERplate Command Reference

This reference can also be accessed using the `Help()` command from the POWERplate Python module. For example:

```
import piplates.POWERplate as POW
POW.help()
```

Note that the `addr` argument for each command is always set to 0 (zero)

General

help() - this will display a summary of all of the commands associated with the POWERplate

getID(0) - returns the string "Pi-Plates POWERplate"

Input Power Monitor

getVin(0) - returns the measured input voltage with an accuracy of ±2%

getIn(0) - returns the amount of current flowing through the power plate with an accuracy of ±10%. This includes the currents flowing to:

- the stack (RPi and all other Pi-Plates)
- the output connectors on the front edge of the board
- the electronics and fan on POWERplate

getCPUtemp() - this function allows you to monitor the CPU temperature on the Raspberry Pi. This a system function and not specific to the POWERplate so no address argument is required.

Wake Scheduler

setRTC(0,zone) - by default, the POWERplate mirrors the real time clock using the local time zone. This local clock gets set whenever the POWERplate module is loaded and does not require any user intervention. However, it it possible to override the time zone with one of two values:

'L' - use local time zone

'G' - use Greenwich meantime also known as "zulu" time.

setWAKE(0,hour,minute,second) - sets the wakeup time on the POWERplate in 24 hour format. The ranges for each argument are:

hour should be between 0 and 23

minute should be between 0 and 59

second should be between 0 and 59

enableWAKE(0) - this function enables the wakeup mode. It should be preceded with the setWAKE call and followed by the powerOFF command.

disableWAKE(0) - this is a complementary function to enableWAKE - it's primarily useful during programming and troubleshooting.

getWAKESOURCE(0) - this function allows the bootup script to interrogate what caused the POWERplate to apply power to the stack. For example, if the POWERplate powers up the stack because of a scheduled wake then the bootup script might make a series of measurements, update a database, then shutdown. But, if the pushbutton causes the powerup, then the script might exit immediately and not shutdown. This function will return the following values:

0 - initial power up

1 - power applied with pushbutton

2 - power applied with scheduled wakeup

powerOFF(0) - simulates the pushbutton and performs the powerdown sequence. This function only works correctly if the line `dtoverlay=gpio-shutdown,gpio_pin=24` has been added to the end of the /boot/config.txt file.

LED Control

setLED(0,led) - this function allows manual control of the bicolor LED on the facing edge of the board. The led argument is a string with one of the following case independent values:

'RED'

'GREEN' or 'GRN'

'YELLOW' or 'YEL'

'OFF'

ledMODE(0,mode) - the default state of the LED is to be on at all times. But, to reduce power consumption, it can be programmed to blink at a 1 second rate with a 12.5% duty cycle or stay off. The mode arguments to do this are shown below. Note that this setting is saved on the POWERplate and will be retained between power cycles. Holding the pushbutton down for 10 seconds will reset this setting to mode 0.

mode 0: auto: always off

mode 1: auto: blink

mode 2: auto: always on (default)

mode 3: manual - set automatically when the setLED command is issued.

Fan Control

fanON(0) - sets the cooling fan to the ON state. This setting is saved on the POWERplate and will be retained between power cycles.

fanOFF(0) - sets the cooling fan to the OFF state. This setting is saved on the POWERplate and will be retained between power cycles. Holding the pushbutton down for 10 seconds will reset this setting to the ON state.

fanSTATE(0) - returns a 1 if the fan is on and a 0 if it is off.

Pushbutton / Power Control

getSWstate(0) - this function provides a programmatic method of reading the status of the button. Returns a 1 if the button is pressed and a 0 otherwise.

enablePOWERSW(0) - this tells the POWERplate to initiate the power control functions when the pushbutton is held down for 3 seconds. This setting is saved on the POWERplate and will be retained between power cycles. This function only works correctly if the line `dtoverlay=gpio-shutdown,gpio_pin=24` has been added to the end of the /boot/config.txt file.

disablePOWERSW(0) - this disables pushbutton control of power. This setting is saved on the POWERplate and will be retained between power cycles.

setSHUTDOWNdelay(0,delay) - by default, after the POWERplate instructs the Raspberry Pi to power down, it delays for 20 seconds to allow the OS to complete the shutdown function. This delay can be modified to be in the range of 10 to 240 seconds. Avoid setting this value for anything less than 20-30 seconds with older versions of the RPi. Note that this value is saved on the POWERplate and will be retained between power cycles. Holding the pushbutton down for 10 seconds will rest this value to 20 seconds.