ORPP MotionController API Documentation

About this Document

- This document describes the ***orpp.h*** interface in detail.
- The ORPP MotionController is connected to a host PC with a serial cable.
- The documentation of the low level serial interface is contained in serial.txt.
- This file decribes the interface used by programmers that need to communicate with the board using C++.

Table of Contents

Example Program

The best way to understand how the interface works it to go over a few basic examples.

```
// Example 1 - orppex_1.cpp
//
// The MotionController is connected to COM1
// This program turns on an LED, waits for a keypress,
// then turns it off.
//

#include <stdio.h>
#include "orpp.h"               // Interface to the orpp board

int main( void )
{
    printf("ORPP MotionController Example 1\n");
    printf("This program turns the first led on and off\n\n");

    orpp_class orpp("COM1");    // Setup the board for COM1
    orpp.connect();             // Connect the program to the board

    printf("LED 0 on \n");

    orpp.set(orpp_led, 0, 1);   // Turn led 0 on

    getchar();                  // Wait for enter key

    printf("LED 0 off \n");

    orpp.set(orpp_led, 0, 0);   // Turn led 0 off

    getchar();

    return(0);                  // Return no error
}
```

The first interesting thing about the program is this line: orpp_class orpp("COM1"); // Setup the board for COM1

Comprehensive definition of the interface

This is a list of all of the interface methods intended for use by programmers.

Communications

After the object is declared using either:

```
orpp_class name("COMPORT");
```

or

```
orpp_class name("HOSTNAME", PORT);
```

The software needs to be told to connect to the board. This is done so the object can be defined before connection takes place.

For example:

```
orpp_class orpp("COM1");        // Create an orpp on COM1
orpp.connect();                 // Connect to the board
orpp.set(orpp_led, 0, 1);       // Turn led 0 on
```

When the following function is called with one parameter it will put the orpp into serial connetion mode using the selected comm port. This is useful if the program needs to change connections without deleting the object

```
void set_connection(char *serial_port);
```

Example:

```
orpp_class orpp;                // Create an orpp
orpp.set_connection("COM1");    // Setup for serial operation
orpp.connect();                 // Connect to the board
orpp.set(orpp_led, 0, 1);       // Turn led 0 on
```

When this function is called with two parameters it will put the orpp into network communication mode. The first parameter is the hostname, and the second parameter is the port number. This is useful if the program needs to change connections without deleting the object

```
void set_connection(char *host, unsigned int the_port);
```

Example:

```
orpp_class orpp;                                // Create an orpp
orpp.set_connection("192.168.0.1", 1200);       // Setup for network operation
orpp.connect();                                 // Connect to the board
orpp.set(orpp_led, 0, 1);                       // Turn led 0 on
```

Setting Outputs

```
void set(control_type control, byte param1, byte param2)
```

This is the general way to set the state of an output on the board. The first parameter is the control type. Controls are either inputs or outputs, only outputs are meaningfull for this call.

These are the outputs that are available for setting:

orpp_led	Control for led outputs, numbered 0 and 1 Example: orpp.set(led, 0, 1); // This will turn Led 0 on.
orpp_digital_out	Control for digital outputs numbered 0..5

Example: orpp.set(digital_out, 0, 1); // Turn digital output 0 on

orpp_servo Control for servo PWM outputs
Example: orpp.set(orpp_servo, 0, 0x40); // Set servo 0 to position 0x40

orpp_opencollector Control for open collector outputs
These are intended to connect an output directly to ground.
Example: orpp.set(open_collector, 0, 1); // Enable opencollector output 0

orpp_pwm Change the duty cycle of a pwm output
Example: orpp.set(orpp_pwm, 0, 1); // Set PWM 0 to 50 percent

```
void set_next(control_type control, byte param1, byte param2)
```

The method behaves the same as set(...) , but the command to change the output is not sent until set_all() is called.

This is used when the updating for the outputs need to happens very often. The has less over head than setting the outouts one at a time.

Example:

```
orpp.set_next(orpp_servo, 0, 0x30); // Setup to change the positions
orpp.set_next(orpp_servo, 1, 0x50); // of the first four servos
orpp.set_next(orpp_servo, 2, 0x60);
orpp.set_next(orpp_servo, 3, 0x70);

orpp.set_next(orpp_digital, 0, 1);  // Setup to enable digital 0 and 0
orpp.set_next(orpp_digital, 1, 1);

orpp.set_next(orpp_pwm, 0, 0x30);   // Setup to change the PWM outputs
orpp.set_next(orpp_pwm, 1, 0xa0);

orpp.set_all();                     // Set all of the outputs at once
```

The following will send all of the outputs set using set_next(...); to the board

```
void set_all( void );          // Send all of the output values at once
```

Getting Input

```
word get(control_type c, byte num)  // Get control value
```

This will return the value of an input. The inputs that can be used with this function are:

orpp_analog Calling the method with this parameter will store the analog value internally and return a 16 bit number from 0 to 1023 representing the voltage level on the analog input
Example: unsigned int value = orpp.get(orpp_analog, 0); // Get analog 0

orpp_digital_in Passing orpp_digital_in as the first parmeter to the set function will store the current state of a digital in and return it's value
Example: bool digital = orpp.get(orpp_ditigal, 0); // Get digital 0

orpp_radio Sample an RC radio and return an 8 bit unsigned value representing the poisition of the control
Example: unsigned char steer = orpp.get(orpp_radiop, 0); // Get radio 0

```
word get_last(control_type c, byte num) // Return last known value
```

Operates the same as get(...) but will return the last known value

Example:

```
bool digital = orpp.get_last(orpp_ditigal, 0); // Get last known digital 0
```

The following reads all of the inouts from the board at the same time. Used in high speed application to eliminate packet overhead.

```
void get_all( void );
```

The values can be retrived with get_last(...)

Example:

```
get_all();                                  // Return the value of a switch
bool s = get_last(orpp_digital_in, 0);      // connected to digital 0
unsigned char r = get_last(orpp_radio 0);   // Get the radio 0 position
```

Modifying Values

```
void add(control_type c, byte num, int delta) // Add an integer to a control
```

Add ***delta*** to the last known value of an outout. Useful for manipulating servos and pwm outputs. It will clamp the new number to 0..255

Example:

```
orpp.add(orpp_servo, 0, 1);  // Incerment servo 0 by 1
```

To toggle an led or digital outouts:

```
void toggle(control_type c, byte num)
```

Example:

```
orpp.toggle(orpp_led, 0);      // Toggle led 0
```

Other

```
vod clear( void );             // Reset to boot values
```

Forces the board to reset to boot values. Also resets the internally stored values

```
void set_error_function(void (*f_ptr)(char *);
void set_message_function(void (*f_ptr)(char *);
```

These are used if the programs needs to display internal orpp events. Mostly, these are communications errors and notifications.

Example:

```
void my_error(char *error)
{
    printf("Error: %s\n", error);
}

void my_message(char *message)
```

```
{
    printf("Error: %s\n", message);
}


void main( void )
{
    orpp_class orpp("COM1");    // Setup the board for COM1

    orpp.set_error_function(my_error);
    orpp.set_message_function(my_message);


    orpp.connect();
}
```

With this arrangement, my_error will be called when a communication error occors and my_message will be called for notifications.

```
bool get_comm_ok(void) // Returns true if communications is working
```

This function will return false if the communications sytems has encountered an error.

It should be checked perodically and connect() called if false is returned.

Application Notes

Send bug reports to nulluser [at] gmail