CoboltMonitor 4.x.x.x for 06-81

Design Description

Table of Contents

1. Introduction 3

1.1 Revision History 3

2. System Overview 3

3. Description of classes 4

3.1 Laser 4

3.1.1 SerialHandler 4

3.1.2 LaserCommon 5

3.1.3 Gen0681Laser 5

3.1.4 CoboltMonitor 5

3.1.5 Tec 5

3.1.6 CommandResponse 5

3.2 GUI 5

3.2.1 MainForm 5

3.2.2 AboutDlg 6

3.2.3 IPLog2 6

3.2.4 LogInDlg 7

3.2.5 LogTerm 7

3.2.6 OptionsDlg 8

3.2.7 ParLog 8

3.2.8 PIDMonitor 8

# Introduction

This is a design description of the software for CoboltMonitor by Cobolt AB. CoboltMonitor 4.x.x.x is used to monitor and control Cobolt lasers of generation 06-81.

## Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Description | Author |
| A | 2013-04-18 | First Revision | Anders Rönnholm |
| B | 2013-04-23 | Changed title | Anders Rönnholm |

# System Overview

The software consists of mainly two parts, the GUI and the laser functionality.

The GUI handles the interaction with the user, responds to actions made by the user and events from laser. The GUI consists of a main window and multiple dialogs for displaying and logging various laser data. Each dialog that communicates with a laser gets a reference to it to be able to send commands and receive events when the laser is updated.

The laser part is separated in three modules each responsible for a specific task.

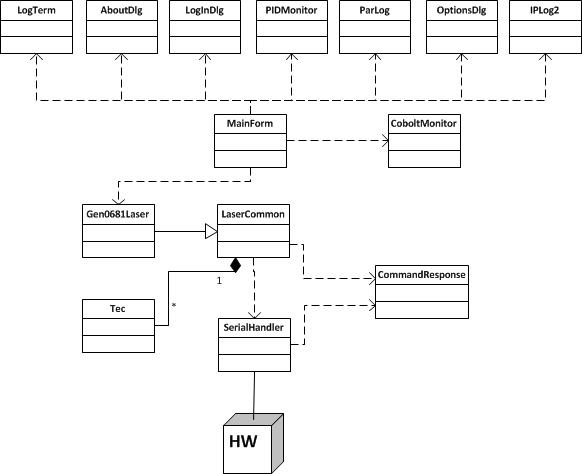
• Specialized laser  
Handles all specialized laser functionality.

• Common laser  
Handles all common functionality for all supported lasers.

• Serial handler  
Handles all serial communication with a laser.

This design makes it easy to support future laser generations. By having a common laser class and a class handling all serial communication a future laser would only need to implement the new functionality in its specialized laser class while reusing the common modules. With this design it would also be possible to make CoboltMonitor able to connect to multiple lasers as each laser object has its own serial communication object.

Below is a class diagram of the whole system.



The following sections describe these classes in more detail.

# Description of classes

## Laser

This section describes the laser related classes.

### SerialHandler

This class responsibility is to handle all serial communication with the laser. It has a queue with commands ready to be sent to the laser which is populated by objects using the SerialHandler. Once a response from the laser has arrived it is parsed and sent with an ResponseReceived event, this tells listeners that a response to a command has been received from the laser.

If a response to a sent command is not received within a given time the command is resent until a response is received.

### LaserCommon

This is an abstract class, this forces specialized laser classes that want to use this to inherit it. LaserCommon keeps all general functionality and data for supported lasers.

All commands for all supported lasers have a public method in LaserCommon to be able to send that command to the laser and a property to be able to read the laser value.

LaserCommon listens on ResponseReceived events sent from SerialHandler. When such an event is received, LaserCommon checks what command the response is for and if it is interested in that command it handles it. LaserCommon then generates an event LaserUpdatedEvent which others can chose to listen on. When this event is raised it tells listeners that the laser has been updated and for what command the laser has been updated. It is then up to the listeners what to do with it.

LaserCommon periodically polls the laser to get current values. This is handled by a timer that periodically triggers an event.

### Gen0681Laser

This class inherits LaserCommon to get all common functionality. It also adds 0681 specific functionality.

As it has unique functionality it overrides ProcessResponse, SaveCommand and the timer event from LaserCommon to add logic for the unique functionality.

### CoboltMonitor

The CoboltMonitor class is not bound to a laser but to the application.

It keeps track on whether the user has logged in or not and when connecting to a laser is it this class responsibility to find out what laser that is connected and create correct laser class.

### Tec

Tec is a small class that only contains properties of a laser tec such as temp, drive and state. Each laser class has a tec array of variable size depending on how many tecs the laser has.

### CommandResponse

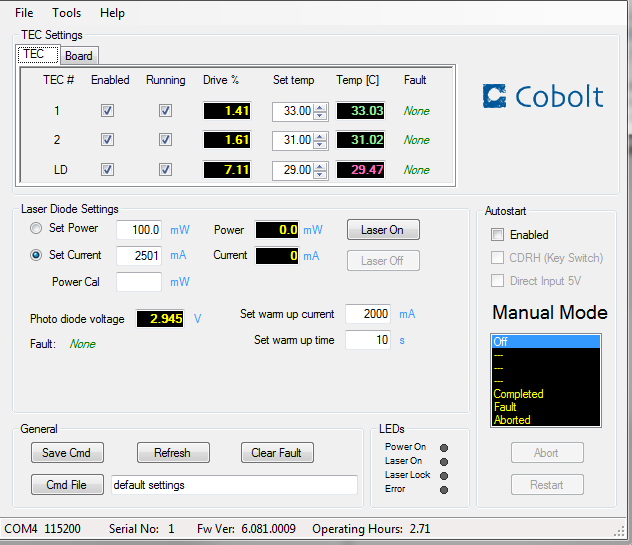
CommandResponse contains data for each command that is sent to the laser and also the response.

## GUI

This section describes the GUI related classes.

### MainForm

This is the main form, it displays laser and tec values and gives the user the ability to control the laser by enabling/disabling tecs, set power and current on the laser.



When connecting to a laser a call to CoboltMonitor is made to get a reference to the currently connected laser.

MainForm adds itself to listen on a LaserUpdatedEvent event triggered by the laser class. This tells MainForm that it is time to update displayed laser values.

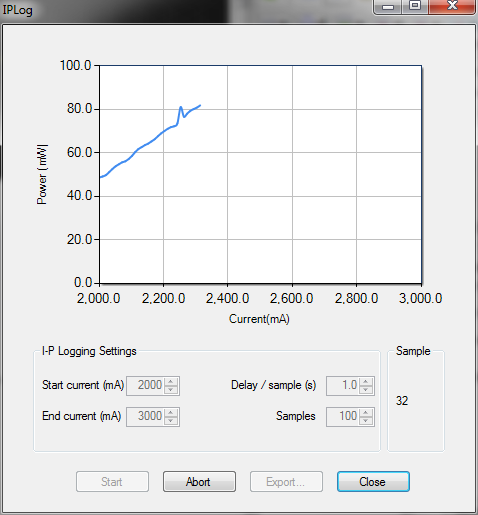
### AboutDlg

This dialog shows information such as revision and owner.

### IPLog2

When creating the IPLog2 object a reference to a laser is sent as an argument.

This dialog receives new current and power values from the laser by listening on the same event from the laser as MainForm, LaserUpdatedEvent. When new current and power values are received IPLog2 updates its I-P log graph.



### LogInDlg

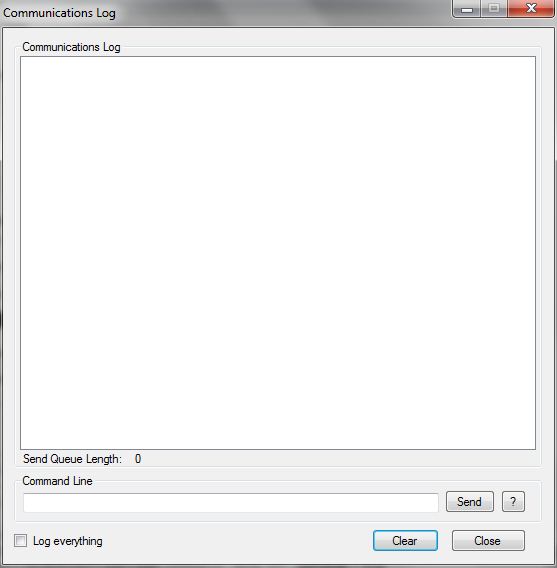
This is a small dialog giving the user the ability to enter a password.

### LogTerm

LogTerm logs the laser communication and lets the user to send raw laser commands. There is no validation of the command sent from LogTerm which makes it possible to send new laser commands when the laser firmware is updated without having to update CoboltMonitor.

When creating a LogTerm object a reference to a laser is sent as an argument. It then listens on the lasers LaserUpdatedEvent event and displays the response in the log.

If log everything is selected all communication is logged and not only commands sent from LogTerm.



### OptionsDlg

This is a small dialog giving the user the ability to select baud rate and com port for connecting to a laser.

### ParLog

This dialog logs drive and temp data from the tecs and the Laser Current, Power, Diode Volt, RMS %, RMS Volt, Photo Diode Voltage to a file.

When creating a ParLog object a reference to a laser is sent as an argument. It then listens on the lasers LaserUpdatedEvent event to see if log data has been updated in the laser.

### PIDMonitor

This dialog shows the PID in a graph for selected tec.

When creating the PIDMonitor object a reference to a laser is sent as an argument. It then listens on the lasers LaserUpdatedEvent event to see if new pid data has been received. When new data is received the chart gets updated.

