MODAC - MorphOptic Data Acquisition and Control

Proof of Concept (PoC) System - Summer 2020

Documentation Overview

(DRAFT)

June 2020

Developed under contract by Jerry Isdale, Maui Institute of Art and Technology ([jerry@mioat.com](mailto:jerry@mioat.com))

Directed by Joe Ritter of Morphoptic

Table of Contents

[1 System Description](#_7srdt0919xpm)

[1.1 Basic Overview](#_g1dfrn6gq9xn)

[1.2 Online Archive:](#_az535bmii8xh)

[1.3 Suggestions for next gen MODAC](#_vyht94hu1kay)

[2 User Documentation](#_d77jg74h00rc)

[3 Design Documentation](#_4wxvaogxn3oj)

[4 Overview Capabilities](#_74xack9ditgk)

[4.1 MODAC is Client/Server System:](#_u2p3rr8ny1gw)

[4.2 Modular/Adaptive Architecture](#_pgx5quhvw90s)

[4.3 File Artifacts](#_820uhgjdijxm)

[4.4 Hardware:](#_6sk4f3t4gmyj)

# 1 System Description

[modified from [original April2019 SoW](https://drive.google.com/file/d/10HGTnMD_rxB_27FZh-aBQCK1kV3q8tQA/view?usp=sharing)] MorpOptic Data Acquisition and Control (MODAC) Proof Of Concept (PoC) is a small linux based computer, with additional Windows 10 PC python UI software, to read and control a glass slumping kiln. This PoC system will be integrated with an existing kiln and additional sensors provided by MorphOptic (and their client).

The MorpOptic Data Acquisition and Control (MODAC) project is a proof of concept general purpose Data Acquisition and Control (DAC) system written in Python on a Raspberry Pi (rPi). Its preliminary use is to control a glass slumping kiln with a few extras, eg non-contact (laser) distance sensor, fans, etc. For its generic DAC functionality, it supports binary outputs, 16 & 24 bit resolutions of Analog/Digital (AD) inputs, K-Type thermocouple support, and expandable via I2C devices. A separate Graphical User Interface (GUI) provides monitoring and control through an IP messaging protocol.

## 1.1 Basic Overview

The MODAC system was conceived as a modular system for collecting and controlling laboratory equipment. The Proof of Concept (PoC) project was undertaken to build a basic framework around a small Linux computer with a Graphical User Interface running on either the same machine or a local network attached system.

*Note: Web connection is not desired at this time. System should be air gapped from the Internet, as overall system security of the PoC was not addressed. A wifi connection is desirable for updates, but otherwise should be turned off. The PoC design could be extended to provide web interface in the future.*

One half of the software (MODAC\_Server) controls the hardware directly on a Raspberry Pi, while a second (Client) software package provides the Human Interface. These communicate over an IP Message Passing Protocol (i.e. NNG). That User Interface (Graphical UI Client) application could run on the same rPi, or it could run on a different machine. The message protocol is (very simply) encrypted and uses uncommon ports, providing a modicum of security with hooks to implement more robust protections if later required.

Modules supporting a variety of input/output devices are implemented, as are simple user interfaces for monitoring and controlling those devices. All devices post their data to a common ‘blackboard’ (in memory data) which is shared with the UI Client. Specific application (i.e. Kiln Control) can add their own data, processing and UI.

## 1.2 Online Archive:

Software for MODAC is kept in the Morphoptic GitHub repository. It may be a good idea to fork the MODAC project at conclusion of the PoC project, and start the next phase from a cleaner slate. GitHub provides a number of Issue Tracking features, which can be used for suggesting and tracking implementation of features and bug fixes.

* <https://github.com/morphoptic> (home of morphoptic on gitHub)
* <https://github.com/morphoptic/modac1> (top of the MODAC POC project)

Documentation is (for now) in a shared Google Drive Folder

* <https://docs.google.com/document/d/1vcEaLtuQ_AqeQSqTlI7zB-rP2MYEZJO3FoMQ5v-v96M/edit?usp=sharing>

Photos of the June Installation are in a subfolder

* <https://drive.google.com/drive/folders/1vT3xqBGDffVKJEHyx7Fh8LhRGOuSqF-o?usp=sharing>
* a G.Presentation of these is in <https://docs.google.com/presentation/d/1wCMaHFE69drXNiUvN24ZZsVhTHZMaK_a-4Hsss59AhI/edit?usp=sharing>

## 1.3 Suggestions for next gen MODAC

* Fork Git Repository; use GitHub Issue Tracking
* different/complete touch screen w/mount
  + Purchased touch screen requires careful mounting to avoid breakage
  + Alternate pre-mounted screens are available
* rPi + USB 5vdc power
  + the rPi gets power from a USB-B type cable providing 5vdc, 3A
  + provides only small Amp (0.5?) on USB ports
  + tried using a USB-B cable to connect to power bus but fails
    - Desktop UI has blinking lightning bolt in corner on low power
  + powered USB hub does not have power
* Utilize AC power distribution in cabinet
  + cabinet has 6 AC outlets and 1 breakered input that are unused
* External antenna for BLE/WiFi
  + unknown: the case may block RF sufficiently to make wireless comms unreliable
* New box for Kiln Adjacent Components
* Separate Client System:
  + The Statement of Work specified a Windows 10 for client
  + The old laptop used for the original KISS kiln control application runs on Windows 7 and is inadequate for the MODAC software.
  + The Client runs adequately on the same rPi as the server application, although some screen lockups have been observed. It would be good to have client running on a separate machine
  + A second rPi is available and should fairly easily be setup as Client GUI.
* Kiln Temperature Calibration
  + The k-type sensor chain does not report close values to the KISS temperature
  + We need some independent (certified) temperature sensor
  + Cones are a valid way to provide (rough +-nC?) feedback on temperature reached
  + Calibration needs to cover not just the thermocouple, but the whole sensor chain.
  + Wiring position issues with the Kiln make it difficult to connect both KISS and MODAC to the same thermocouples
  + Detailed calibration is beyond the scope of the MODAC PoC contract

# 2 User Documentation

User Documentation provides instruction on basic hardware setup and GUI use for Kiln Control.

The documentation is in a [google drive subFolder](https://drive.google.com/drive/folders/1_CS7XPMKPJ14HiZDGGAnZuaY4YvucM4z?usp=sharing), starting with the file [MODAC User Documentation](https://docs.google.com/document/d/131gk2v6jvdlfa9OxDu13b9-0aTxPXDuq0zz-3A7oEMg/edit?usp=sharing)

# 3 Design Documentation

Design documentation is intended for support and development engineers. It gives an overview of the architecture, design constraints, and details of the hardware and software included in MODAC PoC, circa June/July 2020.

The [MODAC Design Documentation](https://docs.google.com/document/d/1IHdJdUaz1vRYGcZqRdQVnzF7wiIQA6-rH-l5l5adt78/edit?usp=sharing) is in a [google drive subfolder](https://docs.google.com/document/d/1IHdJdUaz1vRYGcZqRdQVnzF7wiIQA6-rH-l5l5adt78/edit?usp=sharing)

# 4 Overview Capabilities

## 4.1 MODAC is Client/Server System:

MODAC is designed along a Client/Server pattern. The Server provides control of the hardware sensors and effectors. The Client provides the Monitoring/Control Interface - Human and/or scripted. This allows the Server to focus on its tasks, ignoring presentation. Multiple Clients are supported and developed to emphasize different aspects (GUI, message recording, etc). A Brokerless Internet Protocol (IP) is used to communicate between the client and server.

## 4.2 Modular/Adaptive Architecture

MODAC is designed for modularity in both hardware and software architectures. The basic system can add/remove sensors, etc fairly easily. New sensors and control loops can be added on the basic hardware. While this is not a major feature for a simple dedicated kiln controller, it will allow MODAC to be adapted and repurposed for other Data Acquisition and Control applications.

## 4.3 File Artifacts

MODAC has several File Artifacts. These are described in more detail in the MODAC Design Documentation

* Kiln Scripts are used to step the system through a series of actions. Log files collect messages useful for debugging.
* CSV data files record sensor values.
* Configuration files can change how MODAC operates.

## 4.4 Hardware:

(see hardware [spreadsheet](https://drive.google.com/file/d/14kRG222QJ8bDXm9dK72yMRpR9fs0lM_9/view?usp=sharing) for details)

The MODAC PoC server consists of:

* Raspberry Pi 3B+
* HAT (Daughter) boards
  + Pi-EZ Connect Terminal Breakout HAT (screw terminal daughter board)
  + Waveshare 24 bit AD/DA Converter HAT (8 AD, 2 DA) w screw terminals
    - note: the 24 bit Digital/Analog output is not supported in software
    - note: 24bit A/D is not used in latest version as a) its overkill and b) it was noisy, given long wires between kiln and MODAC
    - a previous 24Bit A/D converter board failed and had to be replaced
* I2C Connected:
  + i2c-RJ45 driver pair (run i2c signals over Cat5 cable)
  + 16 bit AD Converter (4 AD, i2c connect)
  + 4 channel k-type thermocouple amplifier
  + i2c Environment Sensor (Temp, Humidity, Pressure)
* Binary Outputs
  + 8 Channel Relay Module
  + 3 Power strips w/relay control (always on, normally on, normally off)
* Leica Disto Blue Tooth laser distance sensor
* Power Supplies:
  + 5v 10A dc power supply + screw connector bus
  + 5v 3A dc -usb for rPi
  + 12vdc power supply (output side of Relay Module)
* UI components
  + powered USB hub (low quality, missing power supply)
  + USB keyboard
  + USB mouse
  + Desktop HD0MI Monitor (ok DVI with dvi-hdmi cable)
* Case: upcycled rack mount from old UPS system
* Unused Components
  + 7” touch screen
  + 1.3” OLED screen (i2c)
  + USB-Serial for Pi
  + 4 port ethernet switch)