**Documentation for Arduino addition to Vacuum Former V0.1**

# Overview

This effort is to computerize a vacuum former with an Arduino microprocessor and relays.

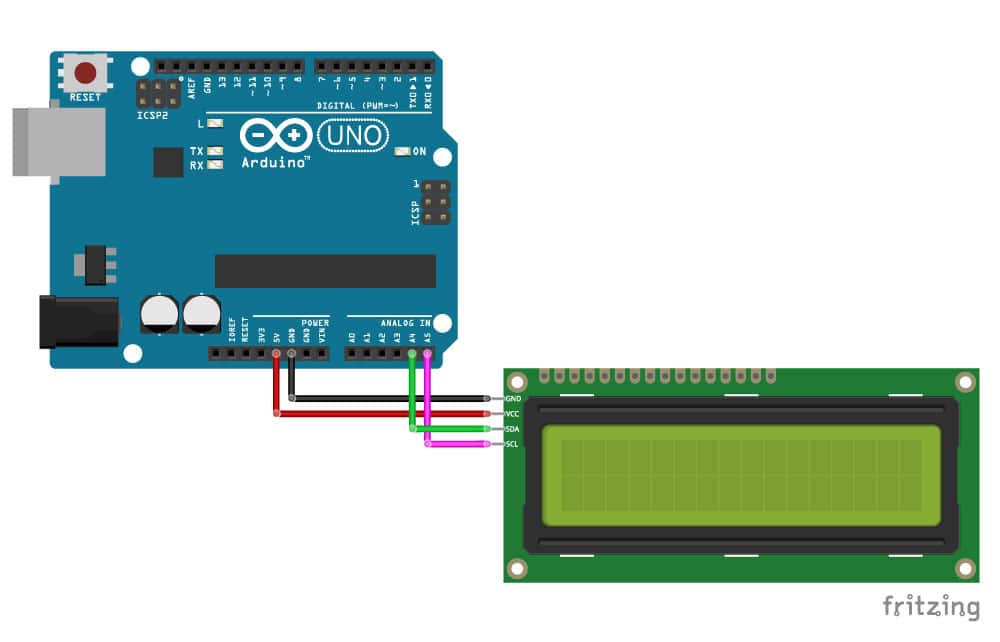
The Flow of the software is as follows:

* 1. Initialize
     1. Set all relays to safe state. Try and read limit switches.
     2. Configure
        1. Manual **Y**/N [save branch for later]
        2. Heater Timer (s)
        3. Power % 0-100 in 10% steps
        4. Vacuum Hi/Low
        5. Vacuum time in (s)
        6. Use Blower **Y**/N or **[timer?]**
  2. Operate
     1. Load Plastic.
        1. **Blower on?**
        2. Lower stage to low limit
        3. Low Limit switch
        4. Press Ok to Proceed
     2. Frame Up
        1. Raise stage to top limit
        2. **[Press or Hold?]**
        3. High Limit switch
        4. Frame motor off
     3. Set mold
        1. Press Ok to Proceed
     4. Cycle start
        1. Blower off
        2. **Automated**: Run heater for required time **[do we want to do math for time and power?]**
        3. Manual: Run until Press Ok to Proceed
        4. Display elapsed time (s)
     5. Heater off
        1. Blower on or delayed?
     6. Frame Down
        1. Frame motor on
        2. Lower frame (no user interaction?)
     7. Touch lower limit switch
        1. Low Limit switch
        2. Frame motor off
        3. Vacuum on at Hi/Low [timer?]
        4. Press Ok to Proceed **[or stop blower?]**
     8. Unload
        1. Press Ok to Proceed
  3. Controls and Relays

|  |  |  |
| --- | --- | --- |
| Function | Type | Notes |
| Blower Fan | Relay |  |
| Heater | Relay |  |
| Frame up | Relay |  |
| Frame down | Relay |  |
| Vac (high) | Relay |  |
| Vac (low) | Relay |  |
| Power | Switch |  |
| Push to advance button | Button |  |
| Navigation config menus | ?? |  |
| Software pause, kill, interrupt? | ?? |  |

# Hardware

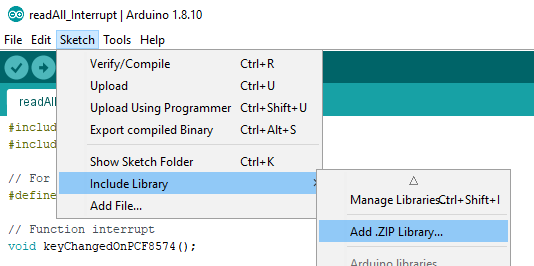
* 1. I2C Serial:

A makerspace 16x2LCD was converted to I2C with a soldered backpack.

The library for the I2C Serial LCD Display is named “LiquidCrystal\_I2C-master.zip” and can be found online at multiple locations.

<https://www.makerguides.com/character-i2c-lcd-arduino-tutorial/>

Use the Include Library menu to add .zip library.



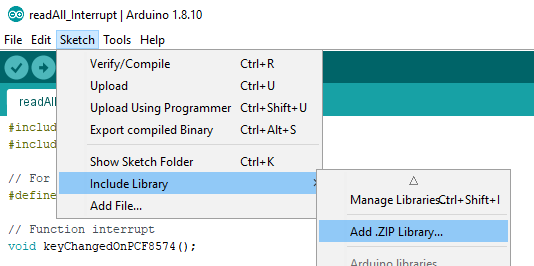
The “SerialDisplay.ino” included in the folder is useful in identifying the address. The display is “0x3F”. Adjusting the contrast of the LCD

**After you have wired up the LCD, you will need to adjust the contrast of the display.** On the I2C module, you will find a potentiometer that you can turn with a small screwdriver.

* 1. PCF8574 8 channel Digital IO for relay:

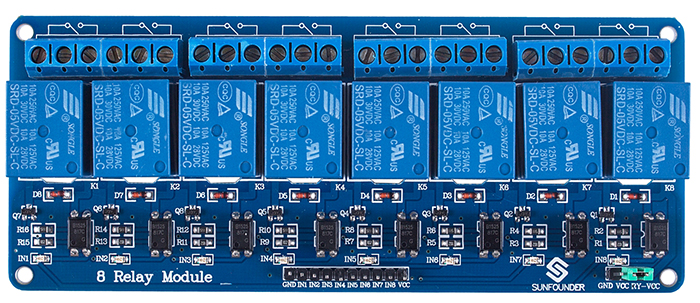
The library for the I2C DIO is named “PCF8574\_library-master.zip” and can be found online at multiple locations.

Use the Include Library menu to add .zip library.



The “SerialDisplay.ino” included in the folder is useful in identifying the address. The DIO is “0x39”.

* 1. 8 channel relays:



Support control of 10A 30V DC and 10A 250V AC signals

5V 8-Channel Relay interface board

**LOW level trigger,** equipped with indicator easy to recognize the working status

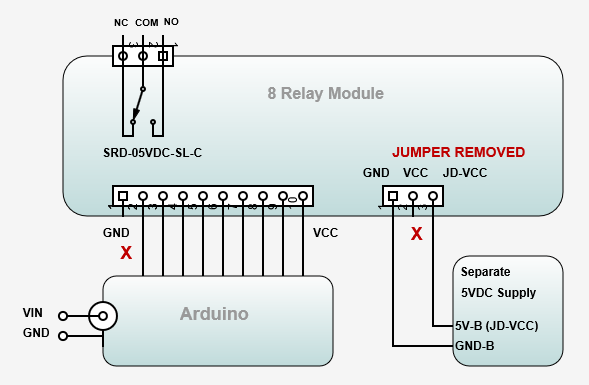
Each relay has NO and NC ports, easier to connect and control the connected devices

* IN1-IN8 wired to P0-P7 in DIO above.
* GND wired to connector next to IN1
* +5V VCC wired to connector next to IN8

Setting all relays on (Low) dims the LCD below readable. Let’s add a better source or use the jumper on the lower right.

Looks like a known condition: This example shows how to make use of the optic isolation feature. <https://forum.arduino.cc/index.php?topic=372547.0>

1. Remove blue jumper
2. Connect JD-VCC and GND from the Relay Module to a Separate 5VDC Supply (1Amp or greater)
3. Connect VCC from the Relay Module to 5V on your Arduino
4. Connect required control signals as shown
5. Note: DO NOT connect GND from the Relay Module to your Arduino



* 1. Button:

Current just SW to Ground to have an input.



A light up arcade button would be cooler



# Software

* 1. I2C\_Scanner:
     1. Utility to find attached things on I2C (Helpful to run serial Monitor) Note **“Include Wire.h”**

/\*I2C\_scanner

This sketch tests standard 7-bit addresses.

Devices with higher bit address might not be seen properly. \*/

#include <Wire.h>

**void** setup() {

Wire.begin();

Serial.begin(9600);

**while** (!Serial);

Serial.println("\nI2C Scanner");

}

**void** loop() {

byte error, address;

**int** nDevices;

Serial.println("Scanning...");

nDevices = 0;

**for** (address = 1; address < 127; address++ ) {

Wire.beginTransmission(address);

error = Wire.endTransmission();

**if** (error == 0) {

Serial.print("I2C device found at address 0x");

**if** (address < 16)

Serial.print("0");

Serial.print(address, HEX);

Serial.println(" !");

nDevices++;

}

**else** **if** (error == 4) {

Serial.print("Unknown error at address 0x");

**if** (address < 16)

Serial.print("0");

Serial.println(address, HEX);

}

}

**if** (nDevices == 0)

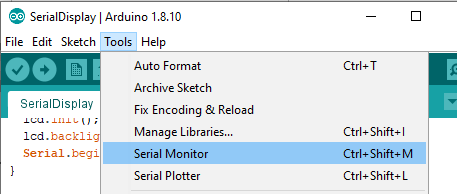
Serial.println("No I2C devices found\n");

**else**

Serial.println("done\n");

delay(5000);

}



[code]

#include <Wire.h>

//Compatible with the Arduino IDE 1.0

//Library version:1.1

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x3F,16,2); // set the LCD address to 0x27 for a 16 chars and 2 line display

void setup()

{

lcd.init(); // initialize the lcd

lcd.backlight(); // Print a message to the LCD.

lcd.setCursor(0,0);

lcd.print(" I2C lcd");

lcd.setCursor(0,1);

lcd.print(" (0x3F,16,2)");

}

[/code]

* 1. PCF8574 8 channel Digital IO for relay:

[code]

//Compatible with the Arduino IDE 1.0

//Library version:1.1

#include <Wire.h>

#include "Arduino.h"

#include "PCF8574.h"

void setup()

{

pcf8574.pinMode(P0, OUTPUT); // Set pinMode to OUTPUT

pcf8574.pinMode(P1, OUTPUT);

pcf8574.pinMode(P2, OUTPUT);

pcf8574.pinMode(P3, OUTPUT);

}

void loop()

{

pcf8574.digitalWrite(P3, HIGH);

delay(1000);

pcf8574.digitalWrite(P3, LOW);

delay(1000);

}

[/code]

* 1. 8 channel relays:
     1. No code at this time Just note P0 = IN1
  2. Button:
     1. An effort to have an interactive button.

[code]

//Compatible with the Arduino IDE 1.0

//Library version:1.1

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include "Arduino.h"

#include "PCF8574.h"

float pressLength\_milliSeconds = 0;//This value will be recorded in seconds

int buttonPin = 2; //The Pin your button is attached to

int pinstart = 0;

int pinend = 0;

void setup()

{

pinMode(buttonPin, INPUT\_PULLUP); // Keep in mind, when pin 2 has ground voltage applied, we know the button is being pressed

}

void loop()

{

while (digitalRead(buttonPin) == LOW ){ //Record \*roughly\* the tenths of seconds the button is pressed

delay(100); //if you want more resolution, lower this number

pressLength\_milliSeconds = pressLength\_milliSeconds + 100;

//display how long button is has been held

lcd.clear();

lcd.print("ms = ");

lcd.println(pressLength\_milliSeconds);

}

}

[/code]