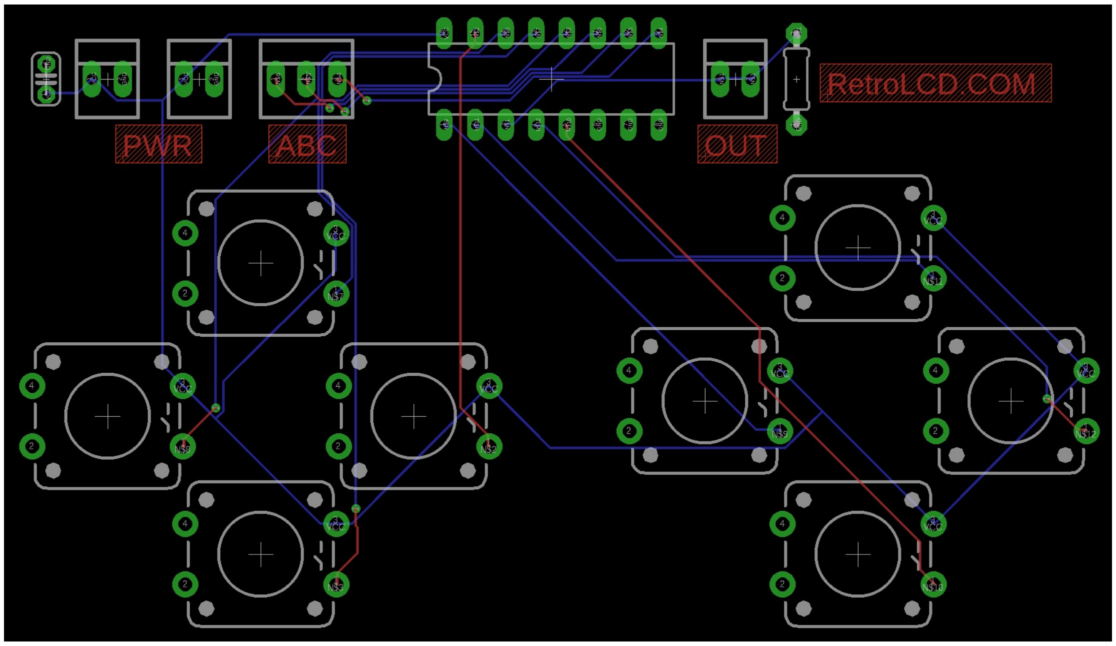
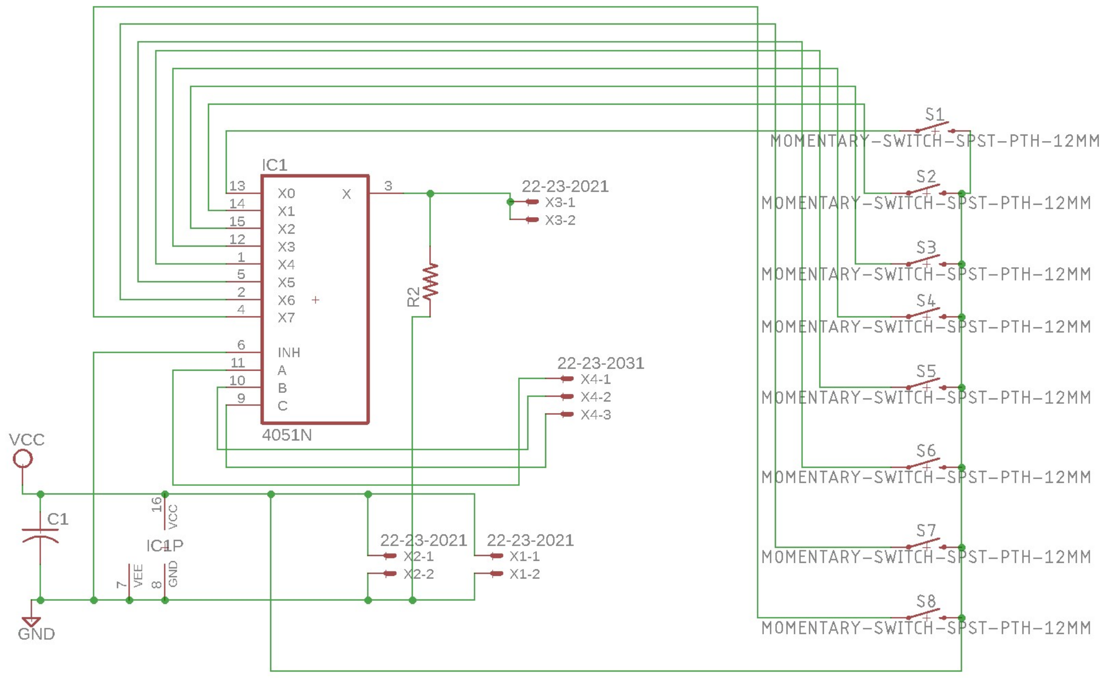
RetroLCD

Key-In Code: The Controller

RetroLCD.com

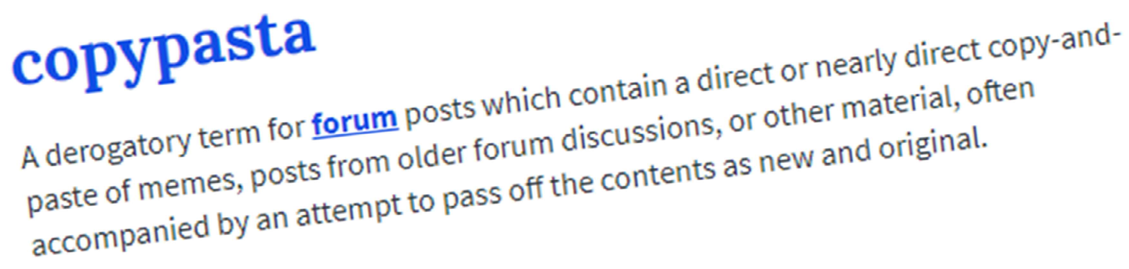
This code is used as the driver for the RetroLCD.com controller board which makes use of an 8bit Multiplexer and up to 8 pushbutton switches.





RetroLCD.com

Urban Dictionary


copypasta

A derogatory term for forum posts which contain a direct or nearly direct copy-and-paste of memes, posts from older forum discussions, or other material, often accompanied by an attempt to pass off the contents as new and original.

Don’t be a Copypasta.

An important part of the learning process is typing in code. This forces you to read every line of code, digest it, and will give you ample opportunity to practice and improve your typing skills.

Most code provided by RetroLCD.com will be provided in a way which discourages copying and pasting.

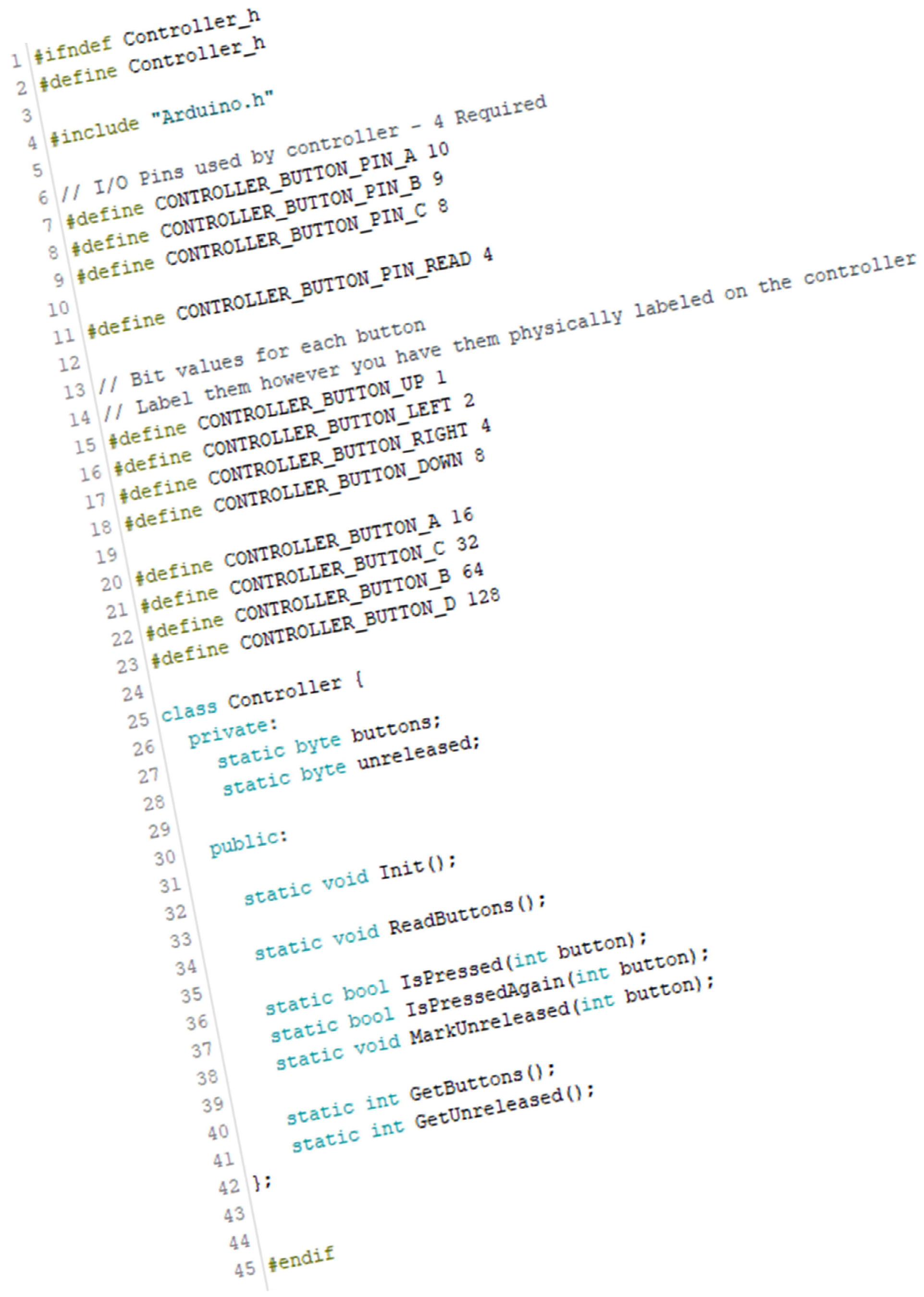
In fact, as projects advance, a lot of code won’t even be provided. Programming is about understanding a problem and figuring out how you would go about solving it. As you get better, your solutions will be better.

Provided code will focus on foundational knowledge like the alphabet, words and sentence structure. But; the idea is not to tell you how to write your book.

Print these Key-In Codes, trim and rotate the sheets to a comfortable angle and type them in. Keep a notebook handy so you can write down notes about what you learn.

RetroLCD.com

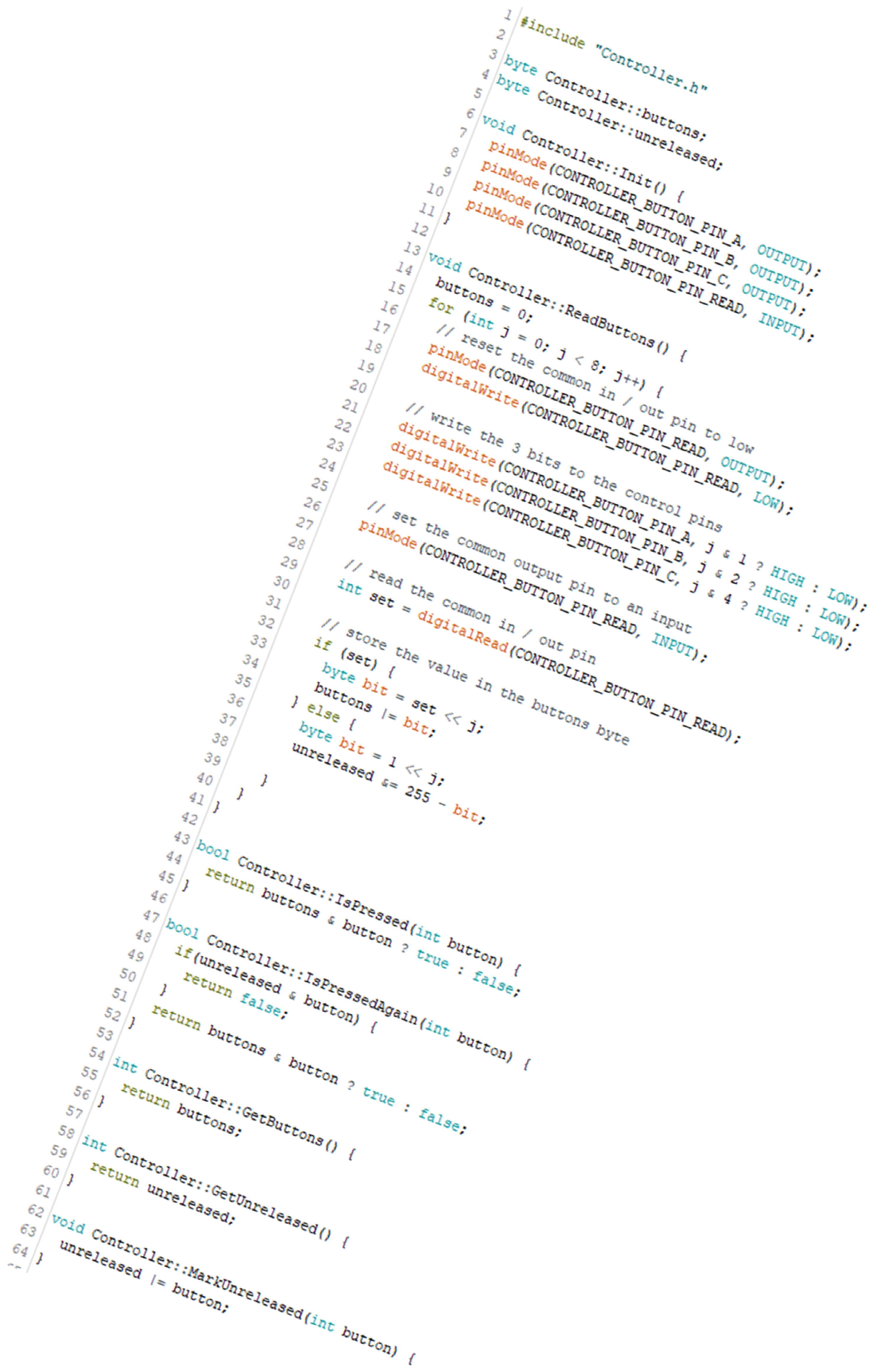
Controller.h



```
#ifndef Controller_h
#define Controller_h

#include "Arduino.h"

// I/O Pins used by controller - 4 Required
#define CONTROLLER_BUTTON_PIN_A 10
#define CONTROLLER_BUTTON_PIN_B 9
#define CONTROLLER_BUTTON_PIN_C 8

#define CONTROLLER_BUTTON_PIN_READ 4

// Bit values for each button
// Label them however you have them physically labeled on the controller
#define CONTROLLER_BUTTON_UP 1
#define CONTROLLER_BUTTON_LEFT 2
#define CONTROLLER_BUTTON_RIGHT 4
#define CONTROLLER_BUTTON_DOWN 8

#define CONTROLLER_BUTTON_A 16
#define CONTROLLER_BUTTON_C 32
#define CONTROLLER_BUTTON_B 64
#define CONTROLLER_BUTTON_D 128

class Controller {
  private:
    static byte buttons;
    static byte unreleased;

  public:

    static void Init();

    static void ReadButtons();

    static bool IsPressed(int button);
    static bool IsPressedAgain(int button);
    static void MarkUnreleased(int button);

    static int GetButtons();
    static int GetUnreleased();
};


#endif
```

RetroLCD.com

Controller.cpp



```cpp
#include "Controller.h"

byte Controller::buttons;
byte Controller::unreleased;

void Controller::Init() {
  pinMode(CONTROLLER_BUTTON_PIN_A, OUTPUT);
  pinMode(CONTROLLER_BUTTON_PIN_B, OUTPUT);
  pinMode(CONTROLLER_BUTTON_PIN_C, OUTPUT);
  pinMode(CONTROLLER_BUTTON_PIN_READ, INPUT);
}

void Controller::ReadButtons() {
  buttons = 0;
  for (int j = 0; j < 8; j++) {
    // reset the common in / out pin to low
    pinMode(CONTROLLER_BUTTON_PIN_READ, OUTPUT);
    digitalWrite(CONTROLLER_BUTTON_PIN_READ, LOW);

    // write the 3 bits to the control pins
    digitalWrite(CONTROLLER_BUTTON_PIN_A, j & 1 ? HIGH : LOW);
    digitalWrite(CONTROLLER_BUTTON_PIN_B, j & 2 ? HIGH : LOW);
    digitalWrite(CONTROLLER_BUTTON_PIN_C, j & 4 ? HIGH : LOW);

    // set the common output pin to an input
    pinMode(CONTROLLER_BUTTON_PIN_READ, INPUT);

    // read the common in / out pin
    int set = digitalRead(CONTROLLER_BUTTON_PIN_READ);

    // store the value in the buttons byte
    if (set) {
      byte bit = set << j;
      buttons |= bit;
    } else {
      byte bit = 1 << j;
      unreleased &= 255 - bit;
    }
  }
}

bool Controller::IsPressed(int button) {
  return buttons & button ? true : false;
}

bool Controller::IsPressedAgain(int button) {
  if(unreleased & button) {
    return false;
  }
  return buttons & button ? true : false;
}

int Controller::GetButtons() {
  return buttons;
}

int Controller::GetUnreleased() {
  return unreleased;
}

void Controller::MarkUnreleased(int button) {
  unreleased |= button;
}
```