# Project Overview

## Introduction

The project selected for this study is the Control Panel hardware design contained in the bobc_hardware GitHub repository. It serves as an example of an open-source electronics project in which design files are made available to engineers, students, and electronics enthusiasts for review, analysis, and further development.

The project is particularly suitable for educational purposes because it provides not only a finished representation of the circuit, but also the underlying design files. The Control Panel folder includes a KiCad PCB design file, a schematic file, netlist files, component-related files, as well as PDF and PNG representations of the design.

This enables the project to be studied at multiple levels, including the electrical circuit, the PCB layout, the components used, and the relationship between the schematic and the physical printed circuit board.

## The GitHub Repository

The project is hosted in the bobc_hardware repository belonging to the GitHub user bobc. GitHub provides a convenient platform for storing and sharing electronic design files alongside conventional software source code.

The repository is now archived and was made read-only by its owner on October 14, 2018.

The Control Panel directory contains the following key files:

| File | Purpose |
|---|---|
| control_panel.sch | Original electronic schematic |
| control_panel.kicad_pcb | PCB layout/design file |
| Control Panel-sch.pdf | PDF representation of the schematic |
| Control Panel-PCB.pdf | PDF representation of the PCB |
| control_panel.png | Image representation of the design |
| control_panel.csv | Component-related data |
| control_panel.net | Electrical netlist |
| control_panel.cmp | Component information |
| control_panel.pro | KiCad project configuration file |
| control_panel-cache.lib | KiCad component library data |

These files demonstrate that the project extends beyond a simple circuit diagram. It represents a complete electronic design package that can be examined using appropriate engineering tools.

Modern PCB design workflows, including KiCad, can export manufacturing files such as Gerber data, drill files, and pick-and-place files directly from the .kicad_pcb project. Production-quality outputs are normally generated using KiCad itself or dedicated EDA software.

## Purpose of the Control Panel

A control panel is an electronic interface through which a user interacts with a larger electronic system. It typically includes physical controls such as switches, push buttons, indicators, connectors, or other user-interface elements.

The key engineering concept is that the Control Panel functions as an interface between the human operator and the electronic system.

When an operator activates a control, an electrical signal is generated or modified. This signal is then transmitted to another part of the system, where it is interpreted and used to perform the required operation.

**Human operator → Control Panel → Electrical signals → Main electronic system**

**Main electronic system → Electrical signals → Control Panel → Indicator → Human operator**

This bidirectional interaction is fundamental to many electronic systems.

## Major Parts of the Project

For the purpose of this study, the Control Panel can be examined in four main areas.

## User Interface

The first area is the physical interface presented to the user. This includes the components through which the operator interacts with the electronic system.

Examples include switches, buttons, indicators, and connectors.

The design of such an interface must consider both electrical functionality and physical usability.

## Electronic Circuit

Behind the physical controls lies the electronic circuitry that interconnects the components.

The schematic describes these electrical relationships. It defines which components are connected and specifies the electrical nets between them.

The schematic therefore represents the logical design of the circuit.

## Printed Circuit Board

The PCB provides the physical implementation of the electronic circuit.

The control_panel.kicad_pcb file contains the PCB design data. It defines the physical placement of components and the routing of copper tracks between them.

**Schematic = electrical design**

**PCB layout = physical implementation**

Both must correspond accurately for the manufactured circuit board to function as intended.

## Documentation and Manufacturing Data

The repository also includes supporting files that enable the design to be documented, reviewed, and potentially manufactured.

PDF versions of both the schematic and PCB are provided for reference. CSV, netlist, and component-related files are also included. In typical PCB workflows, the .kicad_pcb file can be used to generate Gerber files, drill files, and other manufacturing outputs.

This reflects a key principle of professional electronics engineering: a hardware project should be documented in sufficient detail to allow another engineer to understand, reproduce, and modify the design.

## Overall System Architecture

**Operator**

↓

**Physical Controls**

↓

**Control Panel Electronics**

↓

**Connectors / Signal Interface**

↓

**External Electronic System**

The external system can also provide feedback through the interface:

**External Electronic System**

↓

**Signal Interface**

↓

**Control Panel Electronics**

↓

**Indicators / User Interface**

↓

**Operator**

This architecture separates the human interface from the electronic system being controlled.

Such separation is beneficial because the control panel can be optimized for human interaction, while the main electronic system focuses on processing and control functions.

## Why the Project Is Suitable for Study

The project is particularly suitable for a school-level electronics report because it allows multiple engineering concepts to be explored using a single practical example.

A student can begin with the schematic to understand how the electrical circuit is structured. The PCB can then be examined to observe how the same circuit is translated into a physical board layout.

The project also provides opportunities to study:

- Electronic components
- Electrical interconnections
- Circuit schematics
- PCB layout design
- Component placement
- Copper track routing
- Netlists
- Technical documentation
- KiCad project structure
- Open-source hardware development
- Gerber file generation and manufacturing outputs

Rather than studying electronics purely in theory, the student is able to analyse a real-world hardware design.

## Relationship Between the Files

One of the key insights from this project is that the various design files are closely interconnected.

Schematic → Defines components and electrical connections

↓

Netlist → Describes electrical connectivity

↓

PCB Layout → Physically places components and routes connections

↓

PCB Manufacturing → Produces the physical circuit board, typically using Gerber and drill files

This workflow is representative of a standard PCB development process.

### Educational Objective

The primary objective of studying this project is not only to understand the function of the Control Panel, but more importantly to understand how an electronic product is digitally represented and transformed from a conceptual circuit into a physical PCB design.

By examining the project, a student can learn how engineers structure electronic designs and how schematic information is translated into PCB layouts.

The project also demonstrates the educational value of open-source hardware. Instead of treating a circuit board as a closed commercial product, students are able to examine the actual design files and gain insight into the engineering decisions made by the designer.

### Summary

The Control Panel project provides a practical example of an open-source PCB design. Its GitHub repository contains the schematic, PCB layout, and several supporting design and documentation files.

The project therefore serves as a useful foundation for studying the complete electronics design workflow.

In the following chapters, the Control Panel will be examined in greater technical detail, beginning with its schematic and individual electronic components, followed by the PCB layout, component placement, routing, and manufacturing considerations.

Yes. I have now examined the **actual uploaded Control Panel-sch.pdf**, including the schematic image. The PDF is a one-page KiCad schematic dated **20 May 2012**, and we can use it as the authoritative source for this part of your paper.

The most useful way to understand it is to divide the signals into **LCD signals, push-button signals, LED signals, and power/ground**.

# Understanding the Schematic Signals

## VCC — +5 V supply

The schematic explicitly labels:

**VCC = 5 V**

This is the main positive supply for the Control Panel.

It supplies:

- LCD circuitry

- LED circuitry
- Pull-up resistors for the push buttons
- LCD contrast circuit

The other important supply connection is **GND (ground)**, which provides the return path for the electrical circuits.

## LCD signals

The Control Panel has an interface connector labelled **J1**, which is a 16-pin LCD connector.

The schematic identifies the following LCD signals:

| Signal | Function |
|---|---|
| LCD_CONTRAST | LCD contrast adjustment |
| LCD_RS | LCD Register Select |
| LCD_RW | LCD Read/Write |
| LCD_ENABLE | LCD enable/control |
| LCD_D4 | LCD data bit 4 |
| LCD_D5 | LCD data bit 5 |
| LCD_D6 | LCD data bit 6 |
| LCD_D7 | LCD data bit 7 |
| LCD_BACKLIGHT+ | LCD backlight positive |
| LCD_BACKLIGHT- | LCD backlight negative |

The schematic uses a **4-bit LCD data interface**, because only D4–D7 are brought out.

That is a very important observation for your paper.

Instead of using eight data lines:

D0 D1 D2 D3 D4 D5 D6 D7

the design uses:

D4 D5 D6 D7

This reduces the number of connections required between the LCD and the controlling electronics.

## LCD_RS

LCD_RS means **Register Select**.

The signal is connected to:

**P1 pin 1**

and also appears at the LCD connector J1.

Conceptually, this signal tells the LCD controller what type of information is being transferred.

Typically:

- RS = 0 → command/instruction
- RS = 1 → display data

So the external controller can use LCD_RS to distinguish between an instruction and a character/data byte.

## LCD_RW

LCD_RW means **Read/Write**.

At J1, it is connected to:

**J1 pin 5**

The schematic shows this signal connected to ground.

Therefore, in this particular design, LCD_RW is held at the ground level rather than being brought out as an independently controlled signal.

That is consistent with a design where the LCD is intended primarily for **writing/displaying information** rather than reading information back from the LCD.

## LCD_ENABLE

LCD_ENABLE appears at:

**P1 pin 4**

and

**J1 pin 6**

An interesting feature of this particular schematic is that LCD_ENABLE is shown connected to ground.

Therefore, **we should document what the schematic actually shows rather than assume that the signal is controlled elsewhere**.

This is also a good example of why examining the actual schematic is important: a signal name alone does not tell us how the designer electrically implemented it.

## LCD_D4 through LCD_D7

These are the four LCD data signals.

The P1 connector assigns:

| P1 pin | Signal |
|---|---|
| 8 | LCD_4 |
| 9 | LCD_5 |
| 10 | LCD_6 |
| 11 | LCD_7 |

The LCD connector J1 assigns:

| J1 pin | Signal |
|---|---|
| 11 | LCD_4 |
| 12 | LCD_5 |
| 13 | LCD_6 |
| 14 | LCD_7 |

Thus, the Control Panel provides a **four-bit parallel LCD data interface**.

This is one of the most important signal groups in the schematic.

## LCD_CONTRAST

The schematic contains a potentiometer labelled:

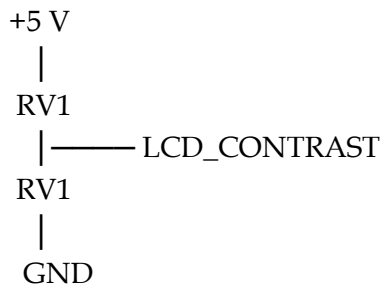
**RV1**

and the associated signal:

**LCD_CONTRAST**

The potentiometer is connected to the 5 V supply and ground, with its wiper providing the contrast voltage.

Conceptually:




The purpose is to allow the LCD's character visibility/contrast to be adjusted.

This is a **variable analog voltage**, unlike the digital control signals such as LCD_D4.

## Push-button signals

This is perhaps the most interesting part of the schematic.

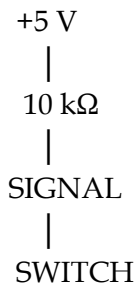
There are **nine push buttons**:

- B_X-
- B_Y-
- B_ZERO
- B_X+
- B_Y+
- B_Z+
- B_CANCEL
- B_Z-
- B_OK

Each has a **10 kΩ pull-up resistor** connected to VCC.

The switch then connects the signal to ground when pressed.

The basic circuit is:



|
GND

This is a classic **active-low switch input arrangement**.

## When the button is NOT pressed

The 10 kΩ resistor pulls the signal toward +5 V:

SIGNAL ≈ HIGH

## When the button IS pressed

The switch connects the signal to ground:

SIGNAL ≈ LOW

Therefore:

**Button released → HIGH**

**Button pressed → LOW**

This is an excellent circuit example to discuss in your school paper.

## The nine button signals

The schematic associates the following buttons with P1:

| P1 pin | Signal |
|---|---|
| 3 | B_OK |
| 6 | B_CANCL |
| 12 | B_X+ |
| 13 | B_X- |
| 14 | B_Y+ |
| 15 | B_Y- |
| 16 | B_Z+ |
| 17 | B_Z- |
| 18 | B_ZERO |

These names strongly indicate the intended operator functions.

For example:

- B_X+ → X positive direction
- B_X- → X negative direction
- B_Y+ → Y positive direction
- B_Y- → Y negative direction
- B_Z+ → Z positive direction
- B_Z- → Z negative direction
- B_ZERO → zero/reference operation
- B_OK → accept/confirm
- B_CANCEL → cancel

The schematic itself gives us the signal names and wiring; the exact software interpretation of each button is not specified on this page, so we should not claim more than that without examining the main controller/software.

## LED signals

There are three indicator signals:

LED1
LED2
LED3

The schematic shows an LED and a **1 kΩ resistor** associated with each indicator.

For example:

LED1
 |
LED
 |
1kΩ
 |
GND

The same arrangement exists for LED2 and LED3.

The three LED signals are also connected to P1:

| P1 pin | Signal |
|---|---|
| 5 | LED1 |
| 19 | LED2 |
| 20 | LED3 |

The 1 kΩ resistors limit current through the LEDs.

## P1 is the important system interface

The **P1 20-pin connector** is essentially the main electrical interface between the Control Panel and the rest of the system.

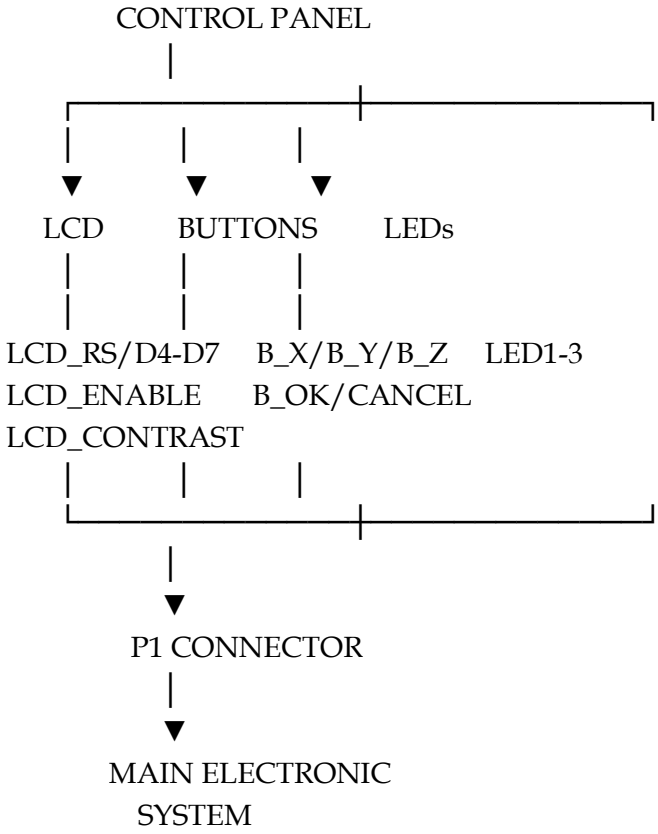
Its signals are:

| Pin | Signal |
|---|---|
| 1 | LCD_RS |
| 2 | GND |
| 3 | B_OK |
| 4 | LCD_ENABLE |
| 5 | LED1 |
| 6 | B_CANCEL |
| 7 | LCD_D?* |
| 8 | LCD_D4 |
| 9 | LCD_D5 |
| 10 | LCD_D6 |
| 11 | LCD_D7 |
| 12 | B_X+ |
| 13 | B_X- |
| 14 | B_Y+ |
| 15 | B_Y- |
| 16 | B_Z+ |
| 17 | B_Z- |
| 18 | B_ZERO |
| 19 | LED2 |
| 20 | LED3 |

**Correction:** On close inspection of the uploaded schematic, **P1 pin 7 is labelled LCD_D1**, not an unspecified LCD signal. This is interesting because the LCD connector itself only brings out D4–D7. We should preserve this observation in our analysis rather than silently "correcting" it.

## The most important electrical concept

The schematic can be understood as four functional blocks:




This makes the purpose of the PCB much clearer.

It is **not a processor board**.

It is primarily a **human-machine interface (HMI)** board.

The Control Panel:

- accepts commands from the operator through buttons;
- provides visual feedback through LEDs;
- communicates with an LCD;
- receives/transmits these signals through P1.

# Interfacing the Control Panel with a PC/Laptop Through USB

# Introduction

The Control Panel is designed to operate as an interface between a user and a larger electronic system. The schematic shows physical push buttons, an LCD interface, indicator LEDs, a contrast control, power connections, and a 20-pin connector designated P1. The connector is a 10×2 pin header and provides the electrical interface between the Control Panel and the external system. (GitHub)

For testing and educational purposes, it is desirable to operate the Control Panel from a modern laptop or desktop computer. A conventional computer, however, does not provide direct access to arbitrary digital signals through its USB connector. Therefore, a USB interface circuit is required between the computer and the Control Panel.

The proposed arrangement is:

**PC/Laptop → USB → USB-to-GPIO Interface → P1 → Control Panel**

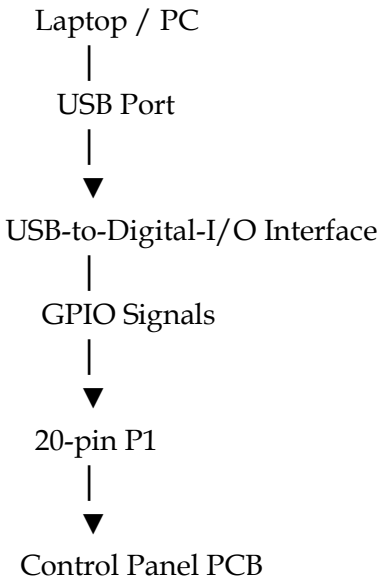
This approach allows the original Control Panel PCB to remain unchanged.

## Why USB Cannot Be Connected Directly to P1

USB is a serial communication interface with its own electrical and communication protocol. The Control Panel connector P1, on the other hand, carries individual control and display signals.

Therefore, a simple USB cable cannot be connected directly between the computer and P1.

An intermediate interface is required:



The interface converts commands from the computer into digital signals and can also read digital signals generated by the Control Panel.

## Signals Available at the Control Panel

The original schematic identifies the connector as:

**P1 – CONN_10X2**

Thus P1 provides 20 connections. (GitHub)

The schematic identifies the following important Control Panel signals.

## Push-button signals

The Control Panel contains nine push buttons:

| Function | Signal |
|---|---|
| X− | B_X− |
| Y− | B_Y− |
| ZERO | B_ZERO |
| X+ | B_X+ |
| Y+ | B_Y+ |
| Z+ | B_Z+ |
| CANCEL | B_CANCEL |
| Z− | B_Z− |
| OK | B_OK |

The schematic shows these button signals together with 10 kΩ resistors. (GitHub)

These switches can therefore be tested electronically by an external controller that monitors the corresponding signals.

## LCD Interface

The schematic also provides a parallel interface for the LCD.

The identified LCD signals include:

- LCD_RS
- LCD_ENABLE
- LCD_D4
- LCD_D5
- LCD_D6
- LCD_D7

This is a six-signal LCD interface. The schematic also includes a 10 kΩ potentiometer designated RV1 for LCD contrast control. (GitHub)

The LCD signals are important because the Control Panel is not merely an input device. The external controller can send information to the LCD so that the user receives feedback from the system.

## LED Indicators

Three indicator signals are identified:

- LED1
- LED2
- LED3

The schematic shows the corresponding LEDs and 1 kΩ resistors. (GitHub)

These signals can be controlled by the external interface. Consequently, a computer program could turn individual indicators on and off during testing.

For example:

PC command
↓
USB interface
↓
LED1 signal
↓
Control Panel LED1
↓
Visual indication

## Power Supply

An important feature of the original schematic is that the Control Panel uses:

**VCC = 5 V**

This is explicitly noted in the schematic. (GitHub)

This has to be considered carefully when selecting the USB interface.

A USB port normally provides a 5 V supply, but the USB port should **not automatically be assumed to be a suitable power source for the complete Control Panel**. The current requirement of the assembled PCB should first be measured or confirmed.

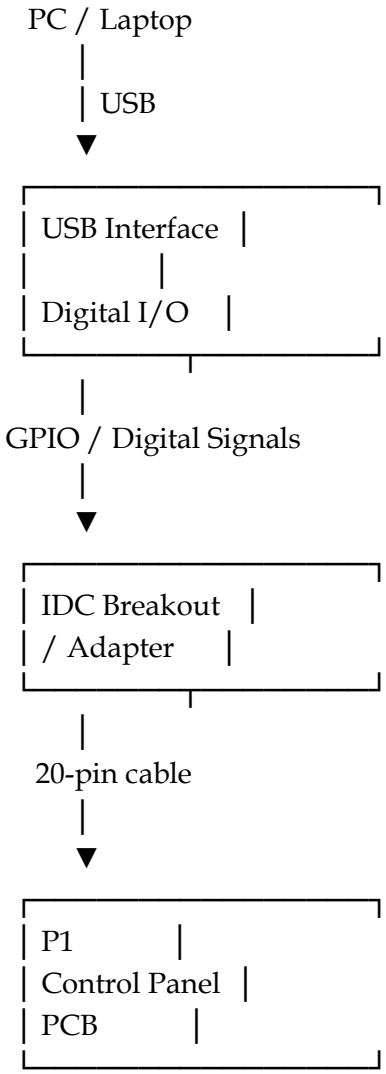
For initial testing, a separate regulated 5 V supply may therefore be preferable.

The USB interface and Control Panel must also have a **common ground** when digital signals are exchanged.

## Proposed USB Test Interface

A suitable experimental arrangement would use a USB-capable microcontroller or USB-to-GPIO interface board.

The proposed system is:



This arrangement provides a convenient way of connecting the already-designed PCB to a computer without changing the PCB layout.

## Number of Digital I/O Signals

From the schematic, the principal externally useful signals consist of:

- 9 push-button signals
- 6 LCD signals
- 3 LED signals

This gives:

**9 + 6 + 3 = 18 digital signals**

In addition, the interface requires appropriate power and ground connections.

This is an important design consideration because a simple USB-to-GPIO device with only 8 or 10 GPIO pins would not be sufficient to control and monitor every function simultaneously.

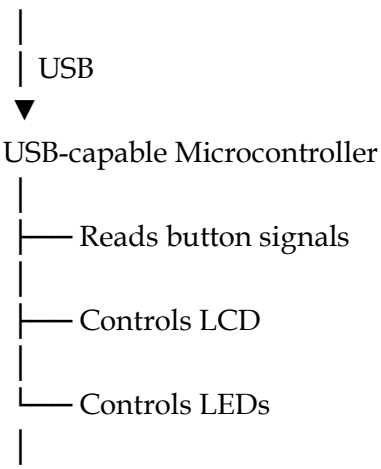
A USB interface providing approximately 18 or more suitable GPIO lines would therefore be desirable for a complete test system.

Alternatively, multiple GPIO expanders could be used.

## USB Microcontroller Alternative

A particularly flexible solution is to use a small microcontroller development board with native USB capability.

The architecture would be:

PC


▼
Control Panel P1

The PC communicates with the microcontroller through USB. A program running on the PC can then send commands such as:

LED1 ON
LED2 OFF
LCD DISPLAY "TEST"

The microcontroller converts these commands into the appropriate electrical signals.

Similarly, when a user presses a button on the Control Panel, the microcontroller can detect the corresponding input and report it to the PC.

For example:

User presses ZERO
↓
B_ZERO changes state
↓
Microcontroller detects it
↓
USB communication
↓
PC receives "ZERO pressed"

This creates a complete two-way test system.

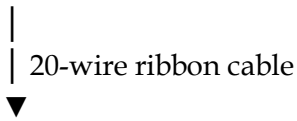
## Why an IDC Breakout Is Useful

The Control Panel uses a 20-pin connector designated P1. (GitHub)

Instead of permanently wiring individual connections to the PCB, a suitable 20-pin IDC breakout board can be used.

The arrangement would be:

USB Interface
|
▼
IDC Breakout Board




This makes the connection effectively plug-and-play and avoids soldering wires directly to the fabricated Control Panel.

It also makes it easier to disconnect the test equipment when the PCB is being used in its normal application.

## Electrical Safety

The USB interface must not be connected to the Control Panel until the electrical characteristics of every signal have been verified.

In particular, the following must be checked:

1. Logic-high voltage.
2. Logic-low voltage.
3. Maximum permitted current.
4. Direction of each signal.
5. Whether any signal is bidirectional.
6. Power-supply requirements.
7. Common-ground requirements.
8. Whether the USB interface requires level shifting.

This is especially important because the schematic identifies the Control Panel supply as 5 V. (GitHub)

A USB GPIO interface operating at a different logic voltage should therefore not be connected directly without confirming compatibility.

## Recommended Testing Procedure

The proposed testing procedure is:

## Stage 1 – Visual inspection

Inspect the fabricated PCB for:

- Correct components

- Solder bridges
- Missing components
- Damaged tracks
- Connector orientation

## Stage 2 – Power test

Apply the appropriate regulated supply and verify the 5 V rail.

## Stage 3 – Button test

Connect the USB interface and monitor:

- B_X−
- B_Y−
- B_ZERO
- B_X+
- B_Y+
- B_Z+
- B_CANCEL
- B_Z−
- B_OK

Each button should produce the expected change in its corresponding signal.

## Stage 4 – LED test

The computer can command the interface to activate:

- LED1
- LED2
- LED3

The corresponding LEDs should illuminate.

## Stage 5 – LCD test

The interface can send appropriate control and data signals through:

- LCD_RS
- LCD_ENABLE
- LCD_D4
- LCD_D5

- LCD_D6
- LCD_D7

The LCD can then be tested for initialization and display operation.

## Stage 6 – Complete functional test

Finally, a computer program can simulate the operation of the external system.

The complete test cycle becomes:

Computer
↓
USB Interface
↓
Control Panel
↓
Buttons / LCD / LEDs
↓
USB Interface
↓
Computer

## Advantages of the Proposed Interface

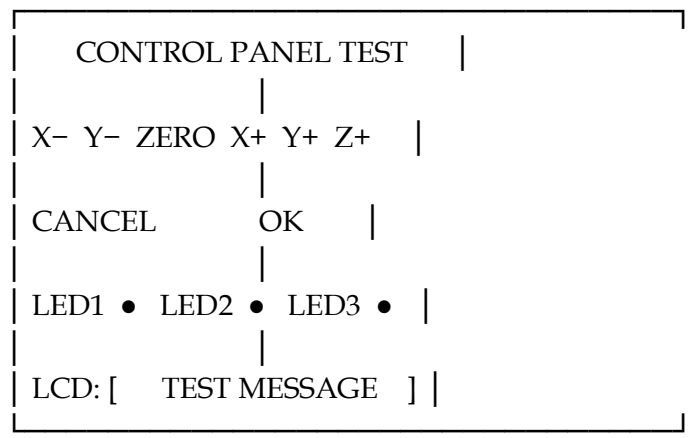
The USB test interface provides several advantages:

- No modification to the existing PCB is required.
- The same fabricated PCB can be tested repeatedly.
- A laptop can be used as the test controller.
- Button operation can be recorded automatically.
- LEDs can be controlled by software.
- LCD operation can be tested.
- Automated test programs can be developed.
- The arrangement is useful for educational demonstrations.
- The same interface can potentially be reused for future versions of the Control Panel.

## Future Development

Once the basic USB interface has been demonstrated, a PC-based test application could be developed.

The application could display a virtual representation of the Control Panel:




The software could show which physical button is pressed and could command the LEDs and LCD.

This would turn the Control Panel into a complete **PC-controlled hardware test platform**.

## Conclusion

The original Control Panel design already provides a suitable external interface through its 20-pin P1 connector. The schematic identifies nine button signals, six LCD interface signals, three LED signals, power and ground connections, and specifies a 5 V VCC supply. (GitHub)

A direct connection between the PC USB port and P1 is not recommended. Instead, a USB-to-GPIO interface should be placed between the computer and the Control Panel.

The proposed solution is:

**Laptop/PC → USB → USB-capable interface → IDC breakout → P1 → Control Panel**

This approach preserves the original PCB design while providing a convenient method for testing the fabricated prototype. It also provides a foundation for developing automated hardware testing software and makes the project more suitable for an educational demonstration.

The Control Panel project therefore extends beyond PCB fabrication: it can become a complete experimental system involving **hardware design, USB communication, digital I/O, embedded programming, PC software, and automated testing**.

# Appendix A – Sources for USB Interface and Control Panel Test Hardware

## A.1 Purpose

The Control Panel PCB can be tested from a laptop or desktop computer by using an intermediate USB-to-digital-interface board. The original PCB does not need to be modified for this purpose.

The proposed test arrangement is:

**Laptop/PC → USB Interface → GPIO → IDC Breakout → 20-pin P1 → Control Panel**

The following components and interface boards are potential sources for constructing this test arrangement.

## A.2 USB-to-GPIO Interface Boards

### Option 1 – Adafruit MCP2221A USB-to-GPIO

The MCP2221A is a USB-to-GPIO interface that also provides I²C and ADC functions. It is available from Indian electronics suppliers and is a convenient option for PC-controlled hardware experiments.

A current product listing shows the Adafruit MCP2221A breakout at approximately ₹824. It has a high user rating on the product listing.

**Important limitation:** The MCP2221A does not provide enough GPIO lines to directly control all of the Control Panel's button, LCD and LED signals simultaneously. Additional GPIO expanders or another interface would therefore be required for a complete implementation.

**Adafruit MCP2221A Breakout General Purpose USB to GPIO**

*₹824.27*

### Option 2 – FT232H USB-to-GPIO Interface

An FT232H-based interface is a more flexible alternative. Current Indian listings show FT232H boards providing USB connectivity together with GPIO, SPI, I²C and UART functions.

One Indian listing for a SmartElex FT232H USB-C breakout describes GPIO as well as SPI, I²C, UART and JTAG capability.

This type of board is particularly interesting for the Control Panel project because the PC can communicate with the interface through USB while the board provides programmable digital signals.

**SmartElex FT232H USB-to-GPIO / SPI / I²C Breakout**

*₹1,162*

The FT232H board operates at 3.3 V, so its electrical compatibility with the Control Panel must be checked before connection.

### Option 3 – CH341 USB Interface

A CH341-based multifunction interface is another possible option. An Indian supplier currently lists a CH341 board providing USB, UART, I²C, SPI and parallel-interface functions, with selectable 3.3 V/5 V operation, at approximately ₹767 including GST.

CH341 USB-to-UART/I²C/SPI/Parallel Interface – DNA Tech India

Its selectable voltage makes it potentially interesting for this project, but the exact GPIO capabilities and software support must be checked before purchase.

## A.3 20-Pin IDC Breakout

The Control Panel uses a 20-pin, 2×10 connector. A 20-pin IDC breakout allows the signals to be brought out to individual terminals without soldering wires directly onto the Control Panel.

A suitable 2×10 IDC breakout is therefore recommended as the mechanical interface between the Control Panel and the USB/GPIO electronics.

**IDC20 Breakout Board 2×10 20-Pin**

*₹3,483*

For a simpler and cheaper arrangement, individual 20-pin IDC connectors and ribbon cable can also be used.

A current Indian listing shows a 20-pin IDC male header with 2.54-mm pitch at approximately ₹17.

**20-Pin IDC Male Header – 2.54 mm**

*₹16.80*

## A.4 Ribbon Cable

A 20-conductor IDC ribbon cable should be used between the breakout board and the Control Panel.

The recommended arrangement is:

**USB Interface → jumper wires → IDC breakout → 20-conductor ribbon cable → P1**

This arrangement is preferable to individually soldering wires to the PCB because the Control Panel remains unchanged.

The cable should have a clearly identified Pin-1 conductor so that the connector orientation cannot be accidentally reversed.

## A.5 USB Cable

The USB cable should match the connector fitted to the selected interface board.

For example:

- USB-C interface → USB-C cable
- Micro-USB interface → Micro-USB cable
- USB-B interface → USB-A-to-USB-B cable

Simple USB breakout boards are also available from Indian suppliers. For example, SunRobotics lists a USB Type-B female breakout board with VCC, GND, D+ and D− connections.

SunRobotics USB Type-B Breakout Board

However, a USB breakout board **by itself is not a USB-to-GPIO interface**. It only exposes the USB electrical connections. Therefore it should not be confused with the FT232H, MCP2221A or microcontroller interface boards described above.

## A.6 Microcontroller Alternative

Instead of a dedicated USB-to-GPIO interface, a USB-capable microcontroller development board can be used.

Possible examples include Arduino-compatible or STM32 development boards.

A low-cost STM32F103C8T6 development board is currently available in India at approximately ₹179 in one product listing.

**STM32F103C8T6 Development Board**

*₹179*

A microcontroller solution has an important advantage: the complete Control Panel test procedure can be programmed into the microcontroller.

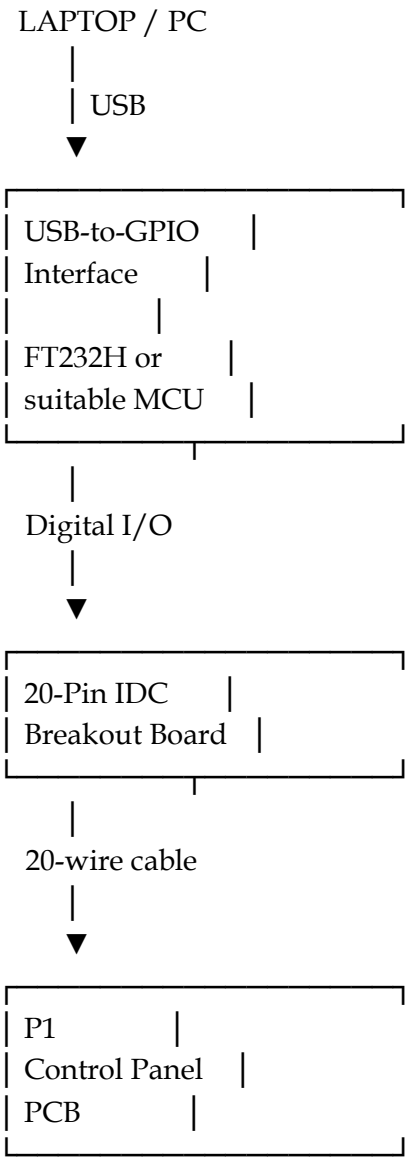
For example:

PC
 │
 │ USB
 ▼
Microcontroller
 │
 ├── Read push buttons
 ├── Control LEDs
 └── Drive LCD
 │
 ▼
20-pin IDC
 │
 ▼
Control Panel

The PC can then run a test program while the microcontroller performs the real-time digital I/O.

## A.7 Recommended Configuration

For the first prototype test, the following arrangement is recommended:




The exact USB interface should be selected only after confirming the required number of GPIO lines and the electrical characteristics of P1.

## A.8 Preliminary Shopping List

| Item | Purpose | Approximate current price |
|---|---|---|
| FT232H USB/GPIO board | Main PC interface | ~₹1,160 |
| MCP2221A | Alternative USB/GPIO interface | ~₹824 |
| CH341 multifunction interface | Alternative interface | ~₹767 |
| 20-pin IDC breakout | Connection to P1 | ~₹10–₹100 depending on type |
| 20-pin IDC connector | PCB/cable connection | ~₹17 |
| 20-conductor ribbon cable | Connect breakout to P1 | Depends on length |
| USB cable | PC-to-interface connection | Depends on type |
| Optional microcontroller | Alternative to USB-GPIO board | ~₹180–₹500+ |

Prices are indicative and can change with supplier, stock and shipping.

## A.9 Important Electrical Precautions

The Control Panel must not be connected to a USB/GPIO board until the P1 pinout has been verified.

Before connecting the prototype, confirm:

1. P1 pin numbering.
2. Signal name for every P1 pin.
3. Which pins are inputs and outputs.
4. Control Panel supply voltage.
5. Logic-high and logic-low voltage levels.
6. Maximum permitted current.
7. Ground connections.
8. Whether any signal is bidirectional.
9. Whether level shifting is necessary.
10. Whether the USB interface can safely drive the required loads.

This is particularly important because the Control Panel is a working PCB and the purpose of the interface is to test it without damaging it.

## A.10 Recommended Purchase Strategy

We should **not purchase several different interface boards**.

The preferred approach is:

**Step 1:** Verify the complete P1 pinout from the original schematic.

**Step 2:** Determine the number of GPIO inputs and outputs actually required.

**Step 3:** Confirm the voltage levels.

**Step 4:** Select the USB/GPIO interface.

**Step 5:** Purchase one 20-pin IDC breakout and the appropriate ribbon cable.

**Step 6:** Test the interface independently before connecting the fabricated Control Panel.

This minimizes cost and, more importantly, reduces the risk of damaging the prototype.

## A.11 Indian Suppliers

Potential Indian sources identified during the research include:

- MakerBazar – USB/GPIO and FT232H interface boards.
- IndustryBuying – Adafruit USB-to-GPIO products.
- DNA Tech India – USB interface and development boards.
- Robu / Robocraze – development boards and connectors.
- SunRobotics – USB breakout boards.
- Sharvi Electronics – IDC connectors and PCB hardware.

Availability and prices should be checked again immediately before ordering because electronic-component stock and prices change frequently.

## A.12 Final Recommendation

For this project, the **FT232H USB-to-GPIO approach is the most interesting starting point for investigation**, while a USB-capable microcontroller is an excellent alternative if we want to develop a dedicated automated test program.

The 20-pin IDC breakout and ribbon cable are relatively simple mechanical accessories and can remain separate from the Control Panel PCB.

The final system should therefore aim for:

**PC USB → Programmable USB/GPIO Interface → 20-pin IDC Breakout → P1 → Control Panel**

This provides a reusable laboratory test interface while keeping the original Control Panel PCB unchanged.

The key point is that **we should verify the P1 pinout before buying the expensive interface board**. The FT232H is attractive, but because the Control Panel has multiple button, LCD and LED signals, we should calculate the exact I/O requirement first.

The Control Panel PCB Fabrication Files can be downloaded from the following location

https://doi.org/10.5281/zenodo.21892765