# Investigation Report: Software-Controlled LED Using Only USB Port on a Linux Laptop

Aarush Dilawri
Vivek Darji

February 16, 2026

## 1 Objective

The objective of this project was to toggle an external LED connected to a breadboard using only:

- A Linux laptop
- A USB port
- A resistor and LED
- Software commands (no additional hardware components)

The constraint explicitly prohibited the use of any additional active or passive components (no capacitors, diodes, transistors, or microcontrollers).

## 2 Initial Circuit Configuration

The LED was initially connected as:

$$\text{USB 5V} \rightarrow \text{Resistor} \rightarrow \text{LED} \rightarrow \text{GND}$$

Upon plugging in the USB cable, the LED immediately illuminated, confirming that VBUS (5V) was active.

## 3 USB Hub Capability Inspection

The USB topology was examined using:

```
lsusb -t
```

The root hub reported:

**No power switching**

This indicates that the USB controller does not support per-port software-controlled power gating.

## 4 Sysfs Power Control Attempts

The following sysfs paths were inspected:

```
/sys/bus/usb/devices/
/sys/bus/usb/devices/usb1/
/sys/bus/usb/devices/usb2/
```

Attempts were made to use:

- `authorized`
- `power/control`
- `disable`

Example:

```
echo 0 | sudo tee \
/sys/bus/usb/devices/usb2/authorized
```

Result:

- USB data communication stopped.
- The LED remained ON.

Conclusion: These controls affect only data-plane behavior, not the 5V rail.

## 5 USB Controller Unbind Experiment

The PCI USB controller was unbound:

```
echo -n "0000:00:14.0" | sudo tee \
/sys/bus/pci/drivers/xhci_hcd/unbind
```

Observation:

- USB devices stopped functioning.
- Keyboard disconnected.
- LED remained ON.

Conclusion: VBUS is not controlled by the xHCI driver.

## 6 ACPI Power State Investigation

System power states were tested:

- Suspend: `systemctl suspend`
- Hibernate: `systemctl hibernate`
- Shutdown: `systemctl poweroff`

Observation:

- LED remained ON during suspend and hibernate.
- LED turned OFF only during full shutdown (S5 state).

This indicates USB VBUS is tied to global system power state and is only cut during S5.

## 7 BIOS Inspection

BIOS was examined for:

- ErP mode
- USB power in sleep
- Deep sleep control

No USB power gating options were available.
Conclusion: Firmware does not expose independent USB power control.

## 8 Experiment: Using D+ Data Line

To explore alternative behavior, the LED was reconnected:

$$\text{D+} \rightarrow \text{Resistor} \rightarrow \text{LED} \rightarrow \text{5V}$$

### 8.1 Observed Behavior

Upon plugging in the USB cable:

- The LED blinked approximately 5–6 times.
- It then stabilized and glowed steadily.

### 8.2 Electrical Explanation

The LED voltage in this configuration is:

$$V_{LED} = 5V - V_{D+}$$

During USB connection:

- The host performs device detection and enumeration.
- D+ is toggled during reset and detection.
- The LED responded to transient voltage changes.

After repeated failed enumeration attempts:

- The host stopped signaling.
- D+ entered idle state ( 3.3V).
- The LED stabilized.

### 8.3 Analysis

This experiment demonstrated:

- USB physical-layer activity is observable externally.
- The data lines are hardware-controlled by the PHY.
- The `authorized` sysfs entry does not control electrical signaling.

However:

> USB data lines cannot be manually driven via software commands.

The blinking was a side-effect of hardware-level enumeration attempts.

## 9 Evaluation of Other Ports

The following ports were considered:

- Ethernet – transformer isolated
- HDMI – fixed 5V pin
- Audio jack – insufficient voltage without additional components

None provided a software-controllable DC output.

## 10 Final Technical Conclusion

From all experiments:

- USB hub reports no power switching.
- Sysfs control affects only data-plane behavior.
- xHCI unbinding does not cut VBUS.
- Suspend does not cut VBUS.
- Only full shutdown (S5) cuts USB power.

Therefore:

> **The USB 5V rail on this laptop is not software-controllable.**

Under the constraints:

- USB only
- No additional hardware
- External LED

It is physically impossible to toggle the LED via software on this machine.

## 11 Engineering Outcome

This investigation demonstrates:

- Understanding of USB architecture
- Separation of power-plane vs data-plane control
- Analysis of ACPI power states
- Experimental validation of hardware limitations

The limitation arises from motherboard-level electrical design, not from software constraints.

## 12 Recommendation

Successful implementation would require:

- Hardware supporting per-port USB power switching, or
- A programmable USB device (e.g., microcontroller)

Without such hardware capability, the task cannot be completed under the stated constraints.