Part 3

lesson 4

Photocell

Overview

In this lesson, you will learn how to measure light intensity using an Analog Input. You will build on lesson 3 of part 3 and use the level of light to control the number of LEDs to be lit.

Component Required:

(1) x Elegoo ESP32

(2) x 400 tie-points breadboard

(8) x leds

(8) x 220 ohm resistors

(1) x 1k ohm resistor

(1) x 74hc595 IC

(1) x Photoresistor (Photocell)

(16) x M-M wires (Male to Male jumper wires)

Component Introduction

PHOTOCELL:

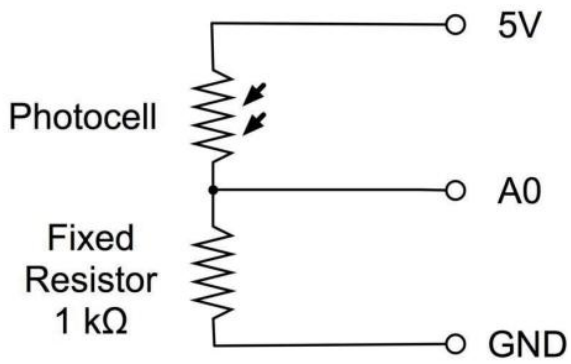
The photocell used, is a type call a light dependant resistor. Also known as an LDR. As the name suggests, these components act just like a resistor, except that the resistance changes in response to how much light is falling on them.

This one has a resistance of about 50 kΩ in near darkness and 500 Ω in bright light.

To convert this varying value of resistance into something we can measure on an ESP32 board's analog input, it needs to be converted into a voltage.

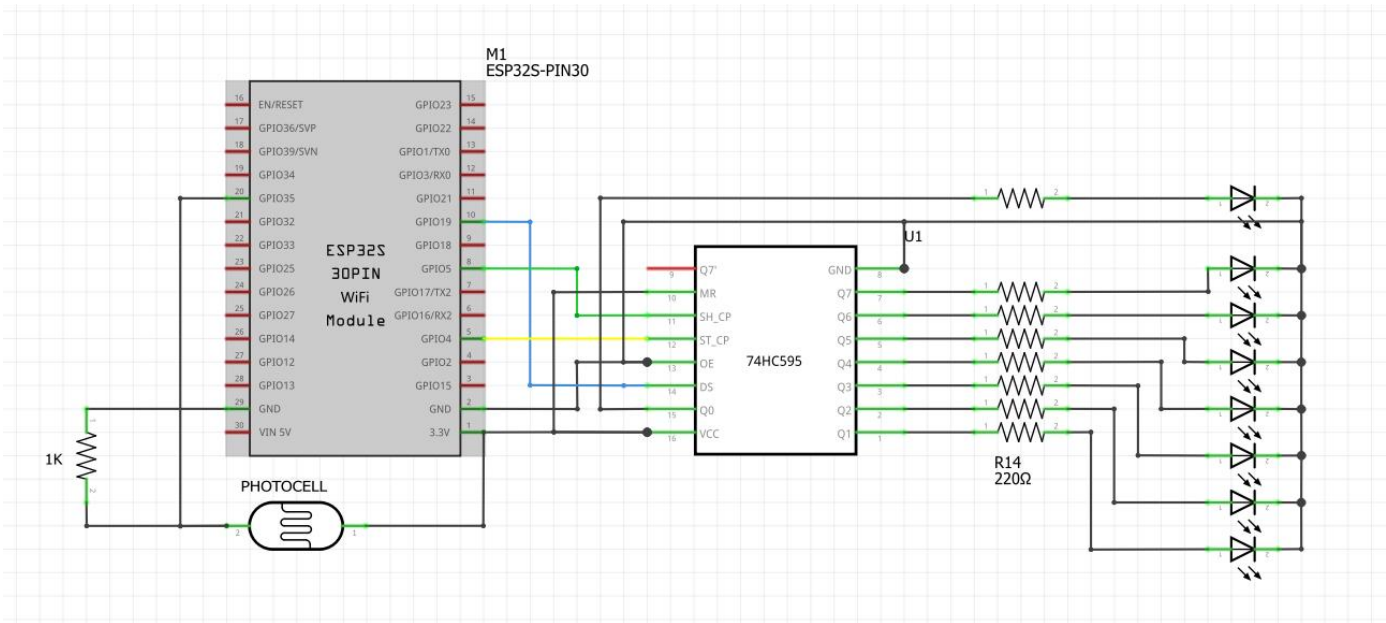
The simplest way to do that is to combine it with a fixed resistor.

The resistor and photocell together behave like a pot. When the light is very bright, then the resistance of the photocell is very low compared with the fixed value resistor, and so it is as if the pot were turned to maximum.

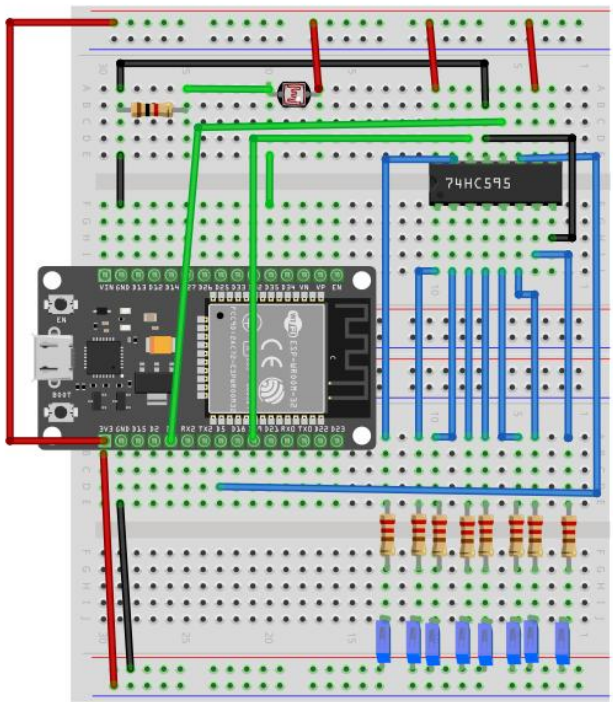


When the photocell is in dull light, the resistance becomes greater than the fixed 1kΩ resistor and it is as if the pot were being turned towards GND.

Load up the sketch given in the next section and try covering the photocell with your finger, and then holding it near a light source.



Connection Schematic



Wiring diagram

Code

After wiring, please open the program in the code folder- **Photocell** and click UPLOAD to upload the program. See Lesson 5 of Part 1 for details about program uploading if there are any errors.

The first thing to note is that we have changed the name of the analog pin to be 'lightPin' rather than 'potPin' since we no longer have a pot connected.

The only other substantial change to the sketch, is to the line that calculates how many of the LEDs are to light.

```
int numLEDSLit = map(reading, 0, 2023, 0, 8);
```

Read Raw Signal: The light sensor converts brightness into an electrical signal. The ESP32 reads this signal via its ADC (Analog-to-Digital Converter), obtaining a value between 0~4095 (ESP32 uses a 12-bit ADC, range 0-4095; for Arduino Uno, it's a 10-bit ADC, range 0-1023).

Proportional Mapping Conversion: Use the map() function to "translate" the sensor's reading range into the range of lit LEDs (0~8).

For example:

Sensor reads 0 (darkest environment) → 0 LEDs lit.

Sensor reads 4095 (brightest environment) → 8 LEDs lit.

Intermediate brightness → Calculate the number of lit LEDs proportionally.

Control LED On/Off: Based on the mapped number, cyclically control the 8 LEDs.

For example, if the mapped result is 3, turn on the first 3 LEDs and turn off the remaining 5.

Tips:

During code tuning, if the LED already reaches full brightness under ambient light, cover the LDR (light-dependent resistor) with your hand and verify that the LEDs dim proportionally. If they respond correctly, gradually reduce the upper bound in the map() function—e.g., from 4095 down to 2000—then re-flash and test again. Iterate until the desired linear dimming range is achieved.