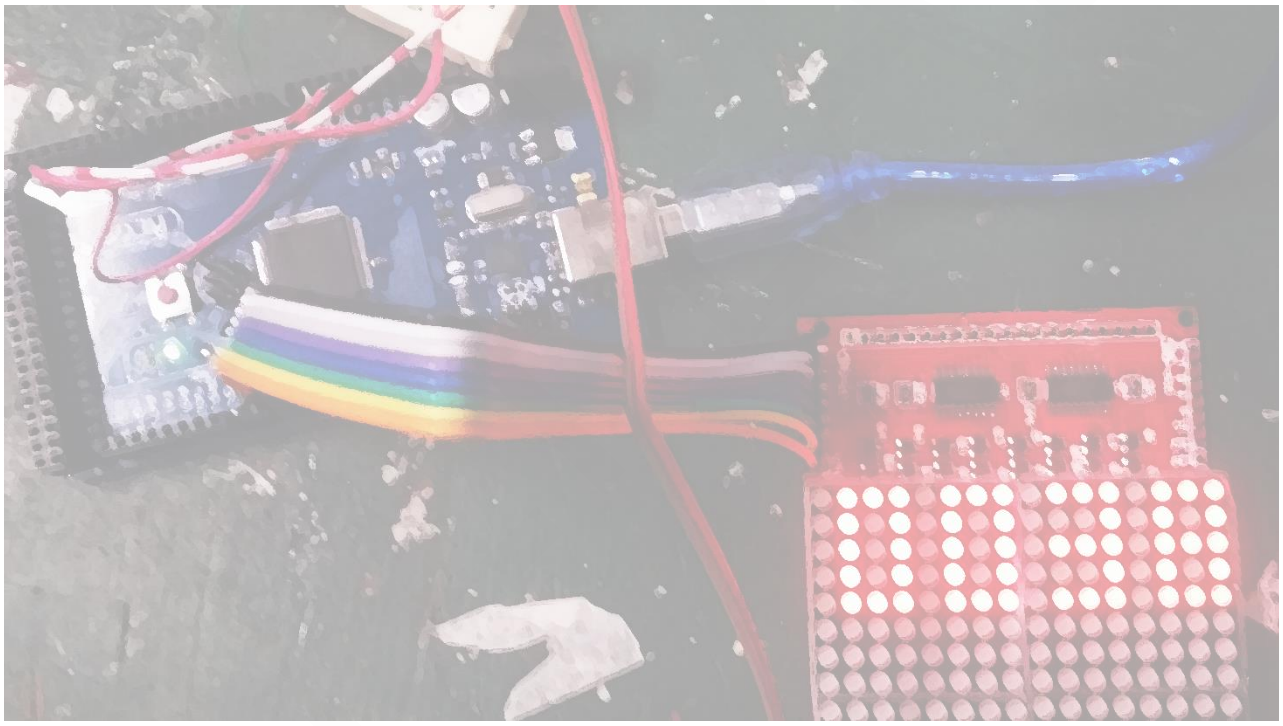
CAN COUNTER

AN ELETRONIC COUNTER FOR THE 10C RECYCLING BIN



Read this manual before attempting to maintain, adjust, update, improve or dismantle this system.

The most updated version of this manual will be located at https://github.com/Teflae/CanCounter/blob/main/Manual.pdf.

CONTENTS

1. DESIGN

1.0. SAFTEY

Voltage over most accessible components should never exceed 12V. Therefore an adapter must be used.

This system will need to operate continuously, and so it will need to be supplied by mains power. The adapter itself needs to be tested and fully insulated.

Tampering could lead to injury.

Since the components have been installed so that they can be reused, they are also easy to access and pull apart. This could pose a serious hazard to individuals. Therefore the system should be kept locked inside the bin at all times.

If this system is malfunctioning, you can get technical support by contacting the author at gpaulcyril@gmail.com.

1.1. CRITERIA

1.1.0. This system relies on an electro-optical (EO) system to detect individual cans as they fall into the bin. The EO system consists of a LASER which produces a narrow beam of light. This beam is reflected back and forth to form a 'net.' A light sensor continuously measures the intensity of that beam. When an item, such as a soft drink can or plastic bottle, breaks or blocks the beam, the drop in intensity is detected by an Arduino. The Arduino then stores and displays this count onto the LED matrix.

1.1.1. It is designed to meet the following criteria:

- **1.1.1.0.** Reliable, low-cost power source.
- **1.1.1.1.** Arduino with plenty of I/O pins.
- **1.1.1.2.** Large display that is legible from a long distance.
- **1.1.1.3.** Accurate timekeeping and flash memory
- **1.1.1.4.** Ability to accurately tally cans.

1.2. PARTS LIST

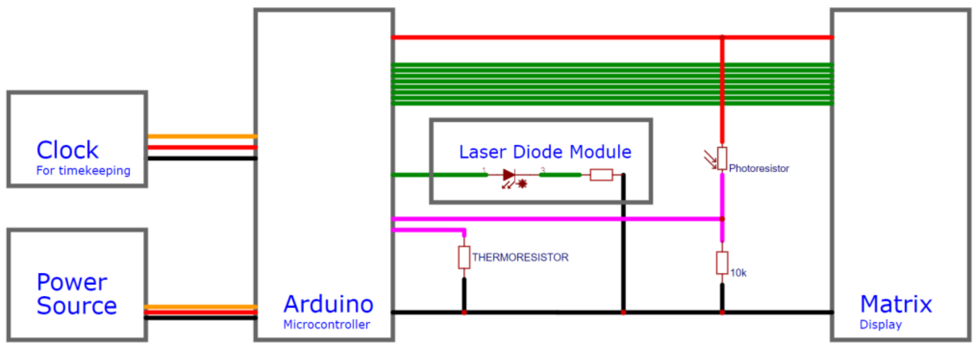
Key	Planned	Under construction	Complete	Optional

Section	Function	Name	Source	Cost ($)[1]
Control	Microcontroller	Arduino Mega	School	49.95
	Clock	Arduino Compatible Real Time Clock Module	School	5.95
	Base & Mount		Built	5.00
	Transformer and Converter	Phone charger	School	10.00
	USB Cable		School	4.88
Sensor	LASER	Red LASER Diode Module	School	4.95

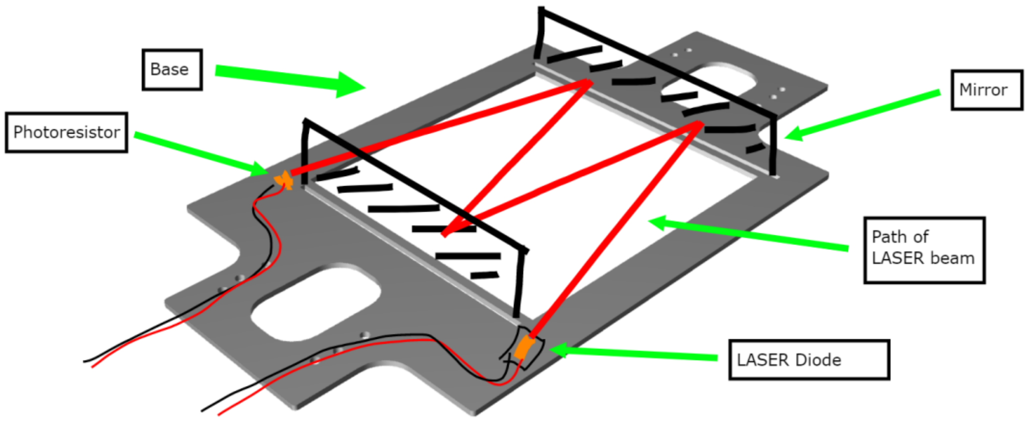
[1] Estimates only.

	Photometer	Photoresistor (LDR)	School	4.00
	Mirrors	Reflective Acrylic	Built	4.00
	Mount	Black Acrylic 20x40cm	Built	13.99
Display	Display	16x16 LED matrix	School	24.95
	Display Protector	Transparent acrylic	Built	4.00
	Cables	Bus Wire		10.00
Misc.	Single-strand wire		School	1.00
	Heat Shrink / Masking Tape		School	1.00
	Thermo-resistor		School	2.00

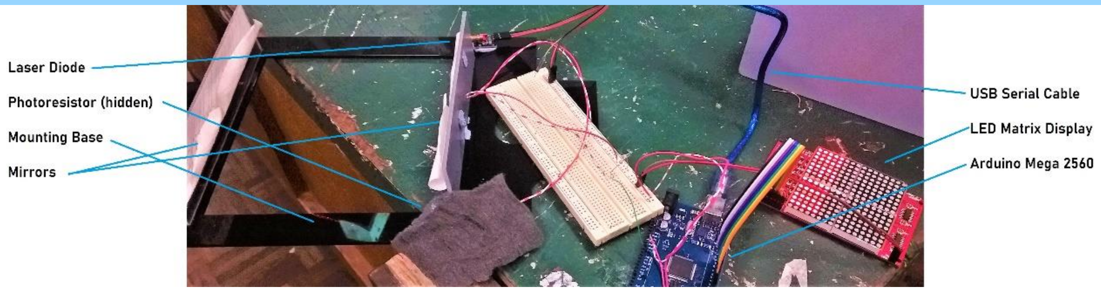
1.3. ELECTRICAL DIAGRAM



1.4. LAYOUT OF EO BASE



1.5. SETUP OF PROTOTYPE



2. SOFTWARE

2.0.0. The source code is available at https://github.com/Teflae/CanCounter.

2.0.1. The Arduino project folder is the 'CC' folder, not the root folder.

2.0.2. Most lines have been carefully commented to make it easy to understand. Read those comments and ensure that you understand the code fully before attempting to modify the code.

2.1. TIMING

2.1.0. The `void loop` function on the Arduino is designed so that it runs at exactly the rate specified by `DELAY_MODE_NORMAL`. This value should be around 20-40ms.

2.1.1. If, due to long pieces of code, the execution of code in the loop function exceeds `DELAY_MODE_NORMAL`, then the built in led will flash to indicate the error.

2.2. MEMORY

2.2.0. The Arduino stores the all-time count inside its built-in EEPROM chip.

2.2.1. The EEPROM is limited to 100,000 write cycles and is written every time a new can is counted. If the all-time count stops working, increase the `AllCountAddress` (located in CC.ino) by 64. This will store the all-time count in a different and hopefully undamaged location on the EEPROM. Use the console command `ee-allcount` to set the all-time count again. (See page 5 below).

3. SERIAL (USB) CONSOLE

3.0. ACCESSING THE CONSOLE

3.0.0. The console can be accessed by connecting the Arduino to a computer running any serial console, including *Arduino Serial Monitor* and *PuTTY*.

- **3.0.0.0.** Check that the port shows '(Arduino/Genuino Mega or Mega 2560)'.
- **3.0.0.1.** Set the baud rate to 57600.

3.0.1. To activate *Debug mode*, send any command to the Arduino.

3.0.2. The Arduino will output a message beginning with `> CanCounter.ino` when ready.

3.0.3. Enter the necessary commands, separated by a `/` or newline character. Individual words, or parameters, should be separated by a whitespace character.

- **3.0.3.0.** Some commands require parameters to work, whilst other commands can optionally require a parameter, like `ee-allcount`.
- **3.0.3.1.** Commands are not case sensitive. Alternatively, the 'index' number of the command can be sent instead.

For example, to find the remainder of 423 divided by 100, send `mod 423 100` or `2 423 100`. The Arduino will return `> mod↲ 23`.

3.0.4. To resume normal processes, send the command `exit`.

3.1. TABLE 1: AVAILABLE COMMANDS

INDEX	COMMAND	DESCRIPTION
1	Echo [string]	Returns a copy of [string]
2	Mod [int a] [int b]	Returns remainder of [a] divided by [b]

3	Exit		Exit debug mode and resume main processes
4	Buffer		Returns the current matrix buffer for display
5	Set [int row] [int val]		Sets row [row] of the buffer to the binary equivalent of [val]
6	printglyph		Deprecated.
7	Test [flag] Available flags:		Set the testing mode. This selects which parameters should be outputted to the Serial for testing and debugging purposes. For example, to check the EO system, send `Test EO`.
	0	Null	Clear tests
	1	EO	Print the following EO data when running: • `rawVal`: The raw `analogRead` () value from the photoresistor • `Reading`: The dampened reading from the photoresistor. • `Average`: The long-term averaged reading from the photoresistor, used as the trigger point to detect a break. • `AllCount`: The all-time count.
	2	Timing	Prints the following timing data, in microseconds: • Program run time, or how long the code took to run. • All time count. • Loop run time, or how long the last loop was.
8	ee-allcount		Returns the All-Time Count, as stored on the EEPROM
8	ee-allcount [int]		Sets the All-Time Count, and writes to EEPROM
9	Breaks		Prints the last 32 breaks (of the laser), with the following format: • `Boolean Counted`, whether the break was actually counted. Breaks won't be counted if they happened `TIME_RESET` close to a previous break (duplicates), or if they ended `TIME_IGNORE` after the break started (external lights turning off). • Timestamp, in ms, when the break occurred. • Duration, in ms, of the break. • `Reading` when the break ended. • `Average` when the break ended.

*4.*DISMANTLING

If this system is malfunctioning, you can get technical support by contacting the author at gpaulcyril@gmail.com.

4.0.0. Many of the components used are not recyclable and contain plastics and other heavy metals. Therefore the components have been installed in such a way that they can be salvaged and re-used. No soldering will be done on the Arduino, breadboard or the matrix, and so these components can be extracted directly and re-used.

4.0.1. Wires may be soldered onto the photoresistor and/or LASER, but these can be cut down if necessary.

4.0.2. The acrylic base should be disposed of in a landfill.

5. APPENDIX

For technical support, you may contact the Author at gpaulcyril@gmail.com.

Notes: