



ສາທາລະນະລັດ ປະຊາທິປະໄຕ ປະຊາຊົນລາວ
ສັນຕິພາບ ເອກະລາດ ປະຊາທິປະໄຕ ເອກະພາບ ວັດທະນາຖາວອນ

ມະຫາວິທະຍາໄລແຫ່ງຊາດ

ຄະນະວິສະວະກຳສາດ

ພາກວິຊາ: ວິສະວະກຳຄອມພິວເຕີ ແລະ ເຕັກໂນໂລຊື້ຂໍ້ມູນຂ່າວສານ



ຮຽບຮຽງໂດຍ: ນາງ ຈິດລັດດາ ພາມີສິດ 225N064722 ຫ້ອງ 3COM1 ລໍາເັບ 75

Email: honey.april04bee@gmail.com

WhatsApp: 02054036881

ສອນໂດຍ: ອຈ ປຕ ລັດທິດາ ຄົມສອນລະສິນ
ວິຊາ ໄມໂຄຣໂປຣເຊັດເຊີ ແລະ ໄມໂຄຣຄອມພິວເຕີ

ສາລະບານ

ບົດນຳ	3
ແນະນຳອຸປະກອນທັງໝົດທີ່ໃຊ້ໃນການທິດລອງ	4
ແນະນຳເຄື່ອງມື SOFTWARE ທີ່ໃຊ້ເຂົ້າໃນການທິດລອງ	34
ບົດທີ່ 1: FIRSTLAB BLINK.....	37
ບົດທີ່ 2: LAB1 LED.....	40
ບົດທີ່ 3: LAB 2 PUSH BUTTON	43
ບົດທີ່ 4: LAB3 RGB	46
ບົດທີ່ 5: LAB4 P&A AND BUZZERS	51
ບົດທີ່ 6: LAB5 POTENTIOMETER.....	55
ບົດທີ່ 7: LAB6 SERVO	58
ບົດທີ່ 8: LAB6 SERVO + POTENTIOMETER (ຕໍ່).....	62
ບົດທີ່ 9: LAB7 STEPPER	65
ບົດທີ່ 10: LAB8: LCD	68
ບົດທີ່ 11: PROJECT (18/12/2025).....	72
ບົດທີ່ 12: MID-TERM PROJECT 1.....	76
ບົດທີ່ 13: MID-TERM PROJECT 2.....	80
ລວມຄັງວິດໄອລິ້ງ QR CODE.....	84

ပိဋကဓာ

ในปัจจุบันไมโครโปรเซสเซอร์ (Microprocessor) ได้เป็นเครื่องมีพื้นฐานที่ใช้ในการควบคุมภาระทางไฟฟ้า เช่น เครื่องปรับอุณหภูมิ ตู้เย็น ไมโครเวฟ และ เครื่องซักผ้า เป็นต้น ด้วยความสามารถในการคำนวณและตัดสินใจอย่างรวดเร็วและการเชื่อมต่ออุปกรณ์ภายนอกได้โดยง่าย ทำให้สามารถประยุกต์ใช้ในหลากหลายอุปกรณ์และระบบได้

ການສຶກສາກ່ຽວກັບໄມໂຄຣໂປຣເຊັດເຊີ້ງ ຈະຊ່ວຍໃຫ້ຜູ້ຮຽນສາມາດເຂົ້າໃຈກິນໄກການເຮັດວຽກພື້ນຖານຂອງຄອມພິວເຕີທັງ Software ກັບ Hardware, ໂຄງສ້າງສະຖາປັດຕະບະກາພາຍໃນ (Computer Architecture) ແລະ ພາສາຄອມພິວເຕີໃນລະດັບຕໍ່າ (Assembly Language) ເຊິ່ງໃນການປະຕິບັດຕົວຈິງ ຈະເຮັດໃຫ້ເຮົາໄດ້ເຫັນພາບຕົວຈິງວ່າຂໍ້ມູນທີ່ເປັນໄຕເລກຖານສອງ ຖືກປະມວນຜົນຜ່ານ Register, ຜ່ານໜ່ວຍຄໍານວນ ແລະ ຕັກກະ (ALU) ແລະ ຖືກຄວບຄຸມໂດຍ Control Unit ແນວດໃດ. ການເຂົ້າໃຈໂຄຣສ້າງພື້ນຖານເຫຼົ່ານີ້ຈະຊ່ວຍໃຫ້ນັກສຶກສາ ສາມາດຕໍ່ວິຈາອນ ແລະ ຂຽນໂປຣແກຣມໄດ້ມີປະສິດທິພາບຫຼາຍຂຶ້ນ ເພະເຮົາຈະໄດ້ຮູ້ວ່າຄໍາສັ່ງແຕ່ລະແຫວທີ່ເຮົາຂຽນ ແລະ ອອກແບບລົງໄປນັ້ນຈະສາມາດຄວບຄຸມການເຮັດວຽກຂອງອຸປະກອນເອເລັກໂຕຣນິກໄດ້ແນວໃດ.

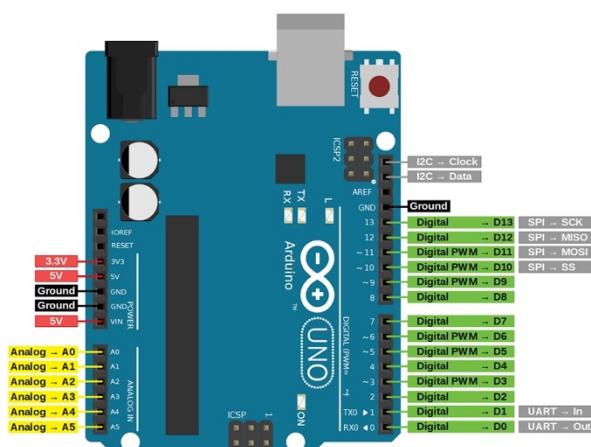
ດັ່ງນັ້ນ, ໄມໂຄນໄປຮອເຊັດເຊີ່ງຈຶ່ງເປັນວິຊາສໍາຄັນໃນການຮຽນຮູ້ກ່ຽວກັບຂັ້ນຕອນໃນການອອກແບບ ແລະ ທີດລອງຮູ່ປະກອນເອເລັກໂຕຣນິກຕ່າງໆໄດ້ ພ້ອມທັງເຮັດໃຫ້ເຮົາສາມາດນຳໄປຕໍ່ຍອດໃນການປະດິດສ້າງ ແລະ ພັດທະນານະວັດທະຍົກໃຫ້ງຈົ່ງທີ່ທັນສະໜັນໄດ້ໃນອະນາຄົດອົງດ້ວຍ.

ແນະນຳອຸປະກອນທັງໝົດທີ່ໃຊ້ໃນການທິດລອງ

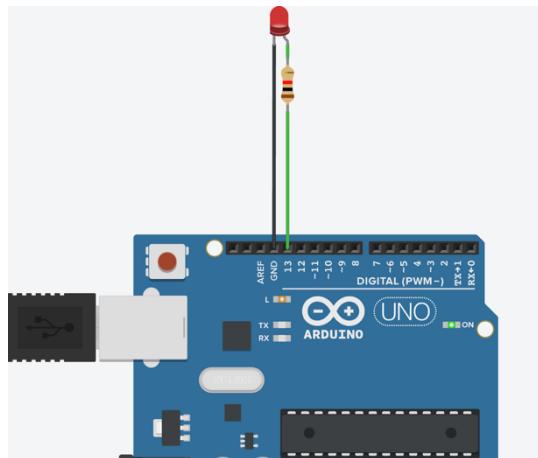
I. Core Boards and Interface

1. Arduino Uno Board

- ແມ່ນ ບອດ Microcontroller ທີ່ໃຊ້ໃນການພັດທະນາໂປຣເຈັກເອເລັກໄຕຣນິກຕ່າງໆເພື່ອຮຽນຮູ້, ການທິດລອງ ແລະ ການສ້າງຕົນແບບນະວັດຕະກຳຕ່າງໆ.
- ການຮັດວຽກຂອງ Arduino ແມ່ນ Microcontroller ຈະຄວບຄຸມການປະມວນຜົນຜ່ານການຮັບຂໍ້ມູນຈາກອຸປະກອນ Input ເຊັ່ນ: ປຸ່ມກິດ, ເຊັ່ນເຊີ, ດອກໄຟ ແລະ ອື່ນໆ ແລະ ໄດ້ຂຽນໂປຣແກຣມດ້ວຍພາສາ C/C++ ຜ່ານໂປຣແກຣມ Arduino IDE ເພື່ອຮັບຄໍາສັ່ງຈາກໂປຣແກຣມໃຫ້ອຸປະກອນທີ່ເຊື່ອມໃນບອດ Arduino ເຮັດວຽກ.

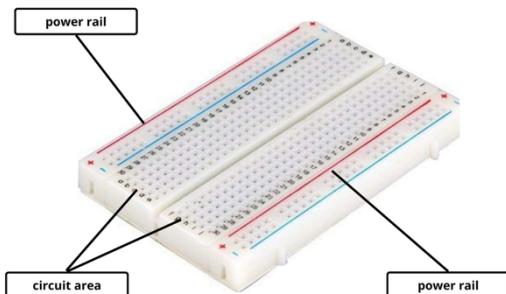


- ຕົວຢ່າງ: ການຕໍ່ດອກໄຟ led ພື້ນຖານດ້ວຍ Arduino
ອີງຕາມຮູບພາບອຸປະກອນປະກອບມີ: Led ສີແດງ, Resistor, Jumper wire ແລະ Arduino Board. ເມື່ອເຮົາຕໍ່ດອກໄຟຂໍ້ວກໄຕຣັດໃສ່ GND ແລະ ຂ້ວອາໂນດເຊື່ອມ resistor ແລ້ວຕໍ່ໃສ່ໃນ Digital Pin 13 ໃນບອດ Arduino ເຊິ່ງຜົນລັບອອກມາແມ່ນດອກໄຟຈະຮູ້ຫັ້ງຈາກເປີດໃຫ້ງນົບອດ Arduino.

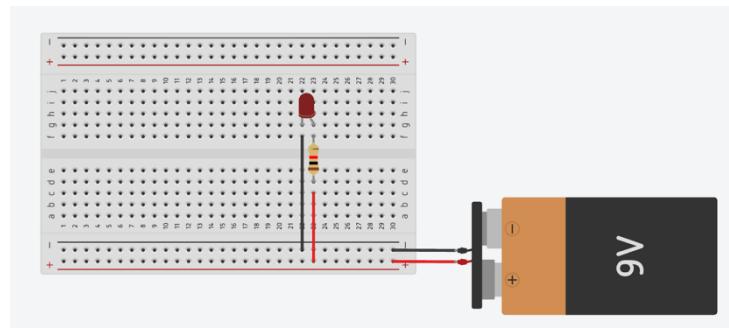


2. Breadboards

- ແມ່ນອຸປະກອນທີ່ໃຊ້ສໍາລັບຕໍ່ວິຈາອນເອເລັກໂຕຣນິກ ເຊິ່ງມັນເປັນຄື່ອງມີທີ່ໃຊ້ໃນການປະກອບວິຈາອນຕ່າງໆເຊັ່ນ: ດອກໄຟ LED, ຕົວຕ້ານຫານ, ເຊັ່ນເຊີ ແລະ ເຊື່ອມຕໍ່ກັບArduino ສະດວກ.
- Breadboard ເຮັດວຽກໂດຍໃຊ້ແຖບໂລຫະ (metal strips) ທີ່ຢູ່ທາງໃນດ້ານລຸ່ມຂອງແຕ່ລະຊ່ອງເພື່ອເຊື່ອມຕໍ່ໄຟຟ້າລະຫວ່າງຮູ້ຕ່າງໆ ເພື່ອໃຫ້ສາມາດສິ່ງກະແສໄຟຟ້າໄດ້.

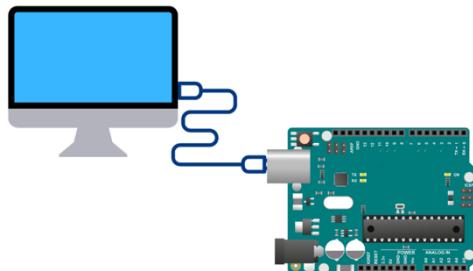


- ຕົວຢ່າງ: ເມື່ອເຮົາເຊື່ອມຕໍ່ດອກໄຟ LED ໃສ່ໃນ Breadboard ເຊິ່ງເຮົາໄດ້ເອົາຂົ້ວກາໂຕດເຊື່ອມໃສ່ຂ້ວລີບ ແລະ ຂຶ້ວອາໄນດເຊື່ອມກັບ ຕົວຕ້ານຫານ ກັບ ຂຶ້ວບວກ ຫຼັງຈາກນັ້ນເຮົາກໍານົດເບີວິວ Battery 9V ເຊື່ອມໃສ່ Breadboard ຫັງຂຶ້ວບວກ ແລະ ຂຶ້ວລີບ ຜົນທີ່ໄດ້ແມ່ນດອກໄຟຈະຮູ້ຂຶ້ນ.



3. USB Cable

- សាយ USB ແມ່ນສາຍທີ່ໃຊ້ເຊື່ອມອຸປະກອນເອເລັກໂຕຣນິກຕ່າງໆເຊົ້າດ້ວຍກັນ ເພື່ອສຶ່ງຂໍ້ມູນ ລະຫວ່າງອຸປະກອນ ແລະ ຈ່າຍໄຟເພື່ອສາກອຸປະກອນ.
- ການຮັດວຽກຂອງອຸປະກອນແມ່ນສຽບສາຍ USB ລະຫວ່າງ 2 ອຸປະກອນເຊັ່ນ: ເຊື່ອມສາຍ Arduino ວັບ Computer ເພື່ອຮັບ ແລະ ສຶ່ງຂໍ້ມູນ.



II. Wiring and Connectors

4. Jumper wires (male-to-male)

- ແມ່ນສາຍໄຟສໍາລັບເຊື່ອມຕໍ່ວົງຈອນໄຟຟ້າ ທີ່ມີປາຍທັງສອງຂ້າງເປັນ ຂາສຽບ (Pin).

5. Jumper wires (female-to-female)

- ແມ່ນສາຍໄຟສໍາລັບເຊື່ອມຕໍ່ວົງຈອນໄຟຟ້າ ທີ່ມີປາຍທັງສອງຂ້າງເປັນ ຊ່ອງສຽບ.

6. Jumper wires (male-to-female)

- ແມ່ນສາຍໄຟສໍາລັບເຊື່ອມຕໍ່ວົງຈອນໄຟຟ້າ ທີ່ມີປາຍຂ້າງໜຶ່ງເປັນຂາສຽບ (male) ສ່ວນອີກປາຍຂ້າງໜຶ່ງເປັນຊ່ອງສຽບ (female).

- ການຮັດວຽກຂອງ Jumper wires (M-M, F-F, M-F) ແມ່ນຮັດໜ້າທີ່ເປັນຕົວນຳໄຟຟ້າສໍາລັບເຊື່ອມຕໍ່ສັນຍານ ຫຼື ກະແສໄຟຟ້າລະຫວ່າງອຸປະກອນເອເລັກໂຕຣນິກຕ່າງໆເຊັ່ນ: Breadboard, Arduino, sensor, module ແລະ ອື່ນໆ.

- ຕົວຢ່າງ: ການນຳໃຊ້ Jumper wire (M-M) ໄປຕໍ່ LED ຢູ່ Breadboard ເຊິ່ງຈະອີກປາຍໜຶ່ງຂອງສາຍສຽບໃສ່ກັບ Resistor ທີ່ເຊື່ອມກັບ LED ຂຶ້ວອາໄນດ ແລ້ວ ອີກປາຍສາຍເຊື່ອມກັບ Pin 13 ຂອງ Arduino ຫລັງຈາກນັ້ນເຮົາກໍເອີ້ນຂ້ອກໄຕດ ຂອງ LED ໄປສຽບກັບ GND ຂອງ Breadboard ກັບ Arduino ຜົນທີ່ໄດ້ອອກໄຟຈະແຈ້ງຂຶ້ນ.



7. 9V Battery Connector

- ແມ່ນສາຍເຊື່ອມຕໍ່ລະຫວ່າງແບກເຕີກ ກັບ ວົງຈອນເອເລັກໂຕຣນິກເຊັ່ນ: Arduino, Breadboard ແລະ ອື່ນໆ ເຊິ່ງລັກສະນະແມ່ນປະກອບມີສາຍສີແດງທີ່ເປັນຂຶ້ວບວກ ແລະ ສາຍສີດຳທີ່ເປັນຂຶ້ວລືບ.
- ການຮັດວຽກແມ່ນກະແສໄຟຟ້າຈະໄຫຼາຈາກຂຶ້ວບວກ ກັບ ຂຶ້ວລືບຂອງແບກເຕີກມານີ້ສາຍໄຟເພື່ອ ຈ່າຍໄຟໃຫ້ແກ່ວົງຈອນທີ່ຕໍ່ກັບ Breadboard ແລະ Arduino.



- ຕົວຢ່າງ: ເອົາ Connector ສົງປິໄສແບກເຕີກແລ້ວເອົາທັງສອງສາຍ ບວກ ກັບ ລົບ ເປັນເຊື່ອມຕໍ່ກັບ VIN ແລະ GND ຂອງ Arduino ຫຼື Breadboard.

III. Basic Components

8. LEDs (Red: 5, Yellow: 5, Blue: 5, RGB: 1)

- ແມ່ນອຸປະກອນເອເລັກໂຕຣນິກທີ່ໃຫ້ແສງສະຫວ່າງເມື່ອມີກະແສໄຟຟ້າຜ່ານໃນທິດທາງທີ່ຖືກຕ້ອງ.
- ການຮັດວຽກຂອງ LED ຈະສະຫວ່າງເມື່ອມີກະແສໄຟຟ້າໄຫຼາຈາກຂຶ້ວອາໄນດ (+) ໄປຫາຂຶ້ວກາໂຕດ (-).
- ຕົວຢ່າງ: ເອົາ LED ສີແດງໜຶ່ງດວງເຊື່ອມຕໍ່ກັບບອດ Arduino ເຊິ່ງຂຶ້ວອາໄນດເຊື່ອມກັບ Resistor ແລະ Pin 10 ຂອງ Arduino ສ່ວນ ຂຶ້ວກາໂຕດເຊື່ອມກັບ GND ຂອງ Arduino.



9. RGB module

- RGB មានចំណាំពីរដែលមានភាពខ្លួនគ្រប់គ្រង់គ្នា ដោយបានចែកចាយជាអ្នកសម្រាប់ការប្រើប្រាស់។
- ការប្រើប្រាស់ RGB LED ត្រូវបានគ្រប់គ្រង់ដោយការប្រើប្រាស់ក្នុងការបង្កើតការងារ។
- Pinout Guide នៃ RGB module:
 - ប្រព័ន្ធឌីជីថល (Common Cathode)

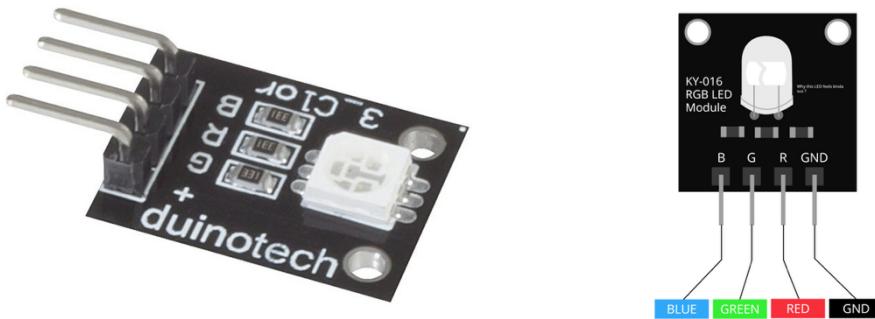
Pin	ខ្លឹម	ប្រភេទ	វិធីការដំឡើង
1	សិរី (Red)	ខ្លឹមគក (+) នៃ LED សិរី	ដំឡើងព័ត៌មាន Output នៃ Microcontroller (ផ្ទាល់ពិនិត្យមានហាម)
2	GND/V	ខាត់គោរ (Common Cathode)	ដំឡើងព័ត៌មានខ្លឹមគក (GND) នៃ Microcontroller (ផ្ទាល់ពិនិត្យមានហាម)
3	សិលិក (Green)	ខ្លឹមគក (+) នៃ LED សិលិក	ដំឡើងព័ត៌មាន Output Pin នៃ Microcontroller (ផ្ទាល់ពិនិត្យមានហាម)
4	សិដ្ឋ (Blue)	ខ្លឹមគក (+) នៃ LED សិដ្ឋ	ដំឡើងព័ត៌មាន Output Pin នៃ Microcontroller (ផ្ទាល់ពិនិត្យមានហាម)

- ប្រព័ន្ធឌីជីថល (Common Anode)

Pin	ខ្លឹម	ប្រភេទ	វិធីការដំឡើង
1	សិរី (Red)	ខ្លឹមលិប (-) នៃ LED សិរី	ដំឡើងព័ត៌មាន Output នៃ Microcontroller (ផ្ទាល់ពិនិត្យមានហាម)
2	GND/V	ខាត់គោរ (Common Cathode)	ដំឡើងព័ត៌មានខ្លឹមគក (GND) នៃ Microcontroller (ផ្ទាល់ពិនិត្យមានហាម)

3	ສີຂຽວ (Green)	ຂຶ້ວລົບ (-) ຂອງ LED ສີຂຽວ	ເຊື່ອມຕໍ່ກັບ Output Pin ຂອງ Microcontroller (ຜ່ານຕົວຕ້ານທານ)
4	ສີພິ້າ (Blue)	ຂຶ້ວລົບ (-) ຂອງ LED ສີພິ້າ	ເຊື່ອມຕໍ່ກັບ Output Pin ຂອງ Microcontroller (ຜ່ານຕົວຕ້ານທານ)

- ຕົວຢ່າງການນຳໃຊ້: ໃຊ້ເຊື່ອມຕໍ່ກັບບອດຄວບຄຸມເຊັ່ນ: ບອດອາດුຍໂນ Arduino ເພື່ອສ້າງໄຟທີ່ປ່ຽນສີໄດ້ຕາມສະຖານະການເຮັດວຽກເຊັ່ນ: ສີແດງ, ສີຂຽວ ແລະ ສີພິ້າ.



10. Resistors (220Ω, 1kΩ, 10kΩ)

- Resistor ແມ່ນຕົວຕ້ານທານທີ່ອອກແບບມາເພື່ອຄວບຄຸມກະແສໄຟຟ້າໃນວິຈອນ ເຊິ່ງຄ່າຄວາມຕ້ານທານມີຫົວໜ່ວຍເປັນ ໂອມ Ω.
- 220 Ω: ມີຄ່າຄວາມຕ້ານທານຕໍ່າ.
- 1kΩ: ມີຄ່າຄວາມຕ້ານທານປານກາງ.
- 10kΩ: ມີຄ່າຄວາມຕ້ານທານສູງ.

ຄ່າຕົວຕ້ານທານ	ແຖບສີ	ການໃຊ້ງານຫຼັກ	ຕົວຢ່າງການນຳໃຊ້
220 Ω	ສີແດງ (2), ສີແດງ (2), ສິນໍ້າຕານ (x10), ສີຄໍາ ($\pm 5\%$)	ຈໍາຮັດກະແສ (Current Limiting)	ຕໍ່ອະນຸກົມກັບ LED ເພື່ອປ້ອງກັນບໍ່ໃຫ້ LED ເສຍຫາຍຍ້ອນກະແສໄຟຟ້າທີ່ຫຼາຍເກີນໄປ
1kΩ	ສິນໍ້າຕານ (1), ສີຄໍາ (0), ສີແດງ (x100), ສີຄໍາ ($\pm 5\%$)	Pull-up/Pull-down, ຕົວແບ່ງແຮງດັນ (Voltage Divider)	ໃຊ້ເປັນຕົວຕ້ານທານ Pull-up/Pull-down ກັບຂາອິນພຸດຂອງ ໄມ ໂຄນຄອນໂທຣເລີ (Microcontroller)

10kΩ	ສິນ້າຕານ (1), ສິດຳ (0), ສີລືມ (x1000), ສິດຳ ($\pm 5\%$)	Pull-up/Pull-down (ຄ່າທີ່ນີ້ຍົມທີ່ສຸດ), ຕົວແບ່ງແຮງດັນ (Voltage Divider)	ໃຊ້ກໍານົດສະຖານະເລີ່ມຕົ້ນ (HIGH/LOW) ທີ່ຊັດເຈນໃຫ້ກັບຂາອິນພຸດຂອງ ໄມໂຄຣຄອນໂທຣເລີ
------	---	---	---

- Pin Guide ຂອງ Resistor: ມັກຈະມີລັກສະນະເປັນຮູບຊີງກະບອກ ແລະ ມີຂາໄລຫະທັງສອງຂ້າງ ເຊິ່ງສ່ວນປະກອບຂອງມັນຈະບໍ່ມີຂໍ້ວ ແຕ່ເຮົາສາມາດຕໍ່ຕົວຕ້ານທານໃນວົງຈອນໄຟຟ້າໄດ້ໂດຍບໍ່ສິ່ງຜົນກະທົບຕໍ່ການຮັດວຽກ.



Resistor 220 ohm



Resistor 1k ohm



Resistor 10k ohm

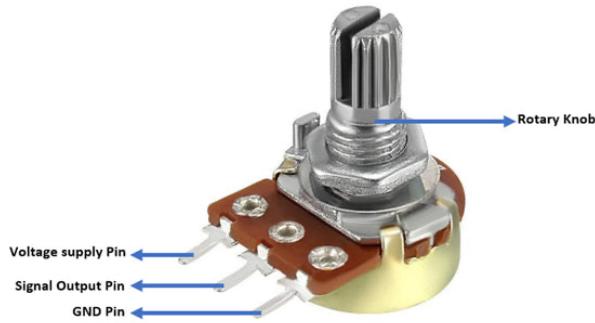
11. Push Buttons (x4 Lids)

- Push Buttons with 4 Lids ແມ່ນສະວິດປຸ່ມກົດແບບຊື່ວຄາວ ຈຳນວນ 4 ອັນ ເຊິ່ງມີຝາປິດ ຫຼື ຫົວປຸ່ມທີ່ເປັນສີຕ່າງໆເຊັ່ນ: ສີຂຽວ, ສີແແງ, ສີເຫຼືອງ ແລະ ສິດຳ ເພື່ອສະດວກໃນການໃຊ້ງານ.
- ການຮັດວຽກຂອງສະວິດແມ່ນຮັດໜ້າທີ່ໃຊ້ເປັນອຸປະກອນຮັບຄ່າເຂົ້າ (Input) ຫຼັກໃນວົງຈອນເອ ເລັກໂຕຣນິກເພື່ອສິ່ງສັນຍານດິຈິຕອນ.
- Pin Guide: ສະວິດ 4 ຂາຈະມີການເຊື່ອມຕໍ່ພາຍໃນເປັນ 2 ຄູ່ທີ່ຕິດກັນ ເຊິ່ງຂາຄູ່ທີ່ຂອງສະວິດແມ່ນ Input Pin ສ່ວນອີກຂາໜຶ່ງແມ່ນ GND.
- ຕົວຢ່າງ: ສະວິດ 4 ຂາສາມາດນາໄປໃຊ້ງານເພື່ອເປີດ-ປິດຂອງດອກໄຟ ຫຼື ໃຊ້ກົດປຸ່ມເພື່ອປ່ຽນ ໂຄມການຮັດວຽກຂອງລະບົບ.



12. Potentiometer (5kΩ)

- Potentiometer ($5k\Omega$) ແມ່ນຕົວຕ້ານທານທີ່ສາມາດປັບຄ່າໄດ້ຂະໜາດ $5k\Omega$ ແມ່ນອຸປະກອນເອົລັກໂຕຣນິກ 3 ຂາທີ່ໃຊ້ໃນການຄວບຄຸມຄວາມດັນໄຟຟ້າ ຫຼື ຄວາມຕ້ານທານໃນວິຈອນໂດຍມີຄ່າຄວາມຕ້ານທານສູງສຸດແມ່ນ 5000 ໂອມ.
- ການຮັດວຽກຂອງມັນມື້ນໍ້າທີ່ສ້າງຄວາມດັນໄຟຟ້າອອກ (Output Voltage) ທີ່ສາມາດປັບປຸງໄດ້ຕໍ່ເນື່ອງຕາມຫຼັກການ ການແບ່ງແຮງດັນ (Voltage Divider).
- Pin Guide: ຂາ 1 ຕໍ່ກັບ GND (0V), ຂາ 3 ຕໍ່ກັບ VCC (ແຮງດັນໄຟຟ້າເຊັ່ນ: 5V), ຂາ 2 (Wiper) ແມ່ນແຮງດັນ output.
- ຕົວຢ່າງການໃຊ້ງານ: ຕໍ່ກັບ Microcontroller ເພື່ອປຸງການໝູນໃຫ້ເປັນສັນຍານບ້ອນເຂົ້າແບບອະນາລັອກ (Analog Input) ເຊັ່ນ: ປັບລະດັບສຽງ, ຄວບຄຸມແສງສະຫວ່າງຂອງດອກໄຟ, ຄວບຄຸມຄວາມໄວຂອງມຳຕີ ແລະ ອື່ນງ.



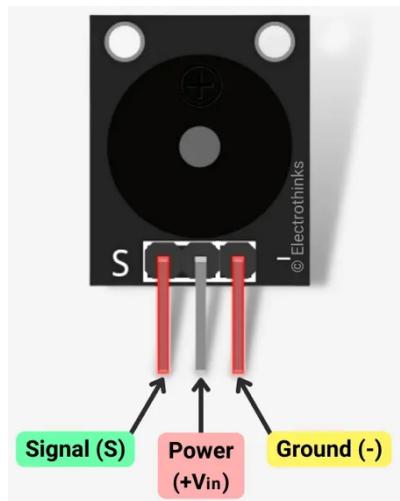
13. Active Buzzer

- Active Buzzer ແມ່ນອຸປະກອນສ້າງສຽງຂະໜາດນ້ອຍຊະນິດທີ່ທີ່ນີ້ຍົມໃຊ້ໃນວິຈອນເອົລັກໂຕຣນິກ ເຊິ່ງໜັ້ນທີ່ຫຼັກຂອງມັນແມ່ນປຸງສັນຍານໄຟຟ້າໃຫ້ເປັນສຽງຕ່າງໆເຊັ່ນ: ເພື່ອນຳໃຊ້ເປັນສັນຍານເຕືອນໄຟ, ສຽງຢືນຢັນ ຫຼື ສຽງແຈ້ງເຕືອນ.
- ຫຼັກການຮັດວຽກຂອງ Active Buzzer ແມ່ນຈະປຸງພະລັງງານໄຟຟ້າ DC ໃຫ້ກາຍເປັນສຽງທັນທີ ເມື່ອໄດ້ຮັບແຮງດັນໄຟຟ້າທີ່ຖືກຕ້ອງ.
- Pin Guide: Active Buzzer ຈະມີ 2 ຂາດັ່ງລຸ່ມນີ້:
 - ຂຶ້ວບວກ VCC: ຕໍ່ເຂົ້າກັບແຫຼ່ງຈ່າຍໄຟ DC (5V)
 - ຂຶ້ວລົບ GND: ຕໍ່ເຂົ້າກັບ GND
- ຕົວຢ່າງການນຳໃຊ້: ນຳເອົາ Active Buzzer ເພື່ອສື່ງສຽງແຈ້ງເຕືອນຈັບເວລາ ເມື່ອເຮົາຈັບເວລາໜີດແລ້ວ Buzzer ຈະມີສຽງແຈ້ງເຕືອນ “ປຶ້ບ” ດັ່ງຂຶ້ນມາ.



14. Passive Buzzer

- Passive Buzzer ແມ່ນອຸປະກອນທີ່ເຮັດໜ້າທີ່ສ້າງສຽງສັນຍານໂດຍປ່ຽນສັນຍານໄຟຟ້າໃຫ້ເປັນສັນຍານສຽງທີ່ມີລັກສະນະຄ້າຢົກບໍລິພົງຂະໜາດນ້ອຍ ແຕ່ຖືກອອກແບບມາເພື່ອສ້າງສຽງທີ່ມີຄວາມທີ່ສູງ.
- ການຮັດວຽກຂອງ Passive Buzzer ຈະຮັດວຽກໂດຍອ້າໄສຫຼັກການຂອງການສັນສະເໜືອນຂອງວັດຖຸທີ່ສ້າງສຽງໂດຍມີສ່ວນປະກອບທີ່ສ້າຄັນຄືແຜ່ນ Piezo ແລະ ສາຍທອງແດງ coil.
- Pin Guide ຂອງ Passive Buzzer ຈະມີ 3 ຂາດັ່ງລຸ່ມນີ້:
 - VCC ເປັນເໝັ້ງຈ່າຍໄຟເພື່ອຮັບແຮງເກີນໄຟຟ້າໃຊ້ງານ (3.3V ຫຼື 5V).
 - GND ເປັນຂີ້ວລີບຕໍ່ກັບຂາ GND ຂອງບອດ.
 - Input/Output ຮັບສັນຍານຄວບຄຸມຄວາມຖືຂອງສຽງ ເຊິ່ງໃຊ້ຕໍ່ກັບຂາ Digital Pin.
- ຕົວຢ່າງ: Passive Buzzer ສາມາດໃຊ້ໃນການຄວບຄຸມຄວາມຖື ແລະ ໄລຍະເວລາເພື່ອສ້າງໂທນ ສຽງທີ່ແຕກຕ່າງກັນເຊັ່ນ: ການຫຼັ້ນເພິງ.

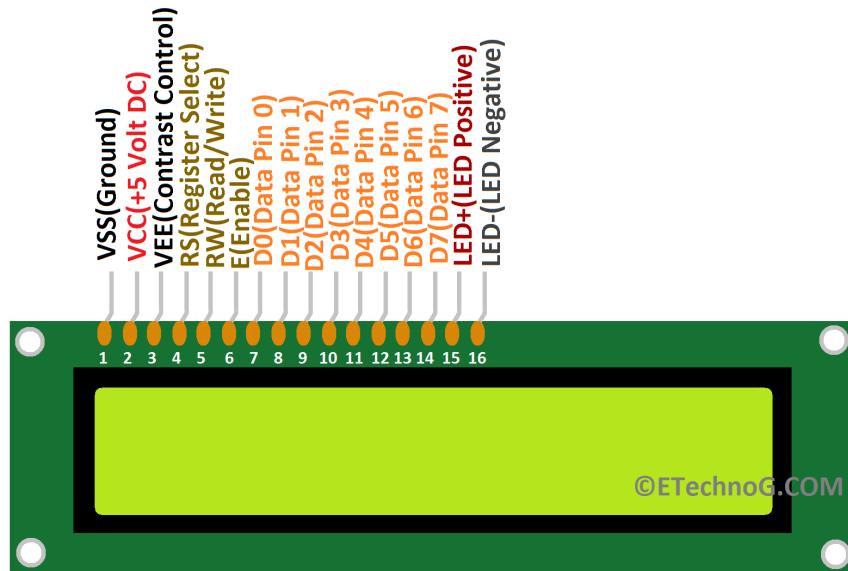


IV. Displays and Output

15. 16x2 LCD Display

- 16x2 LCD Display ແມ່ນຈະສະແດງຜົນທີ່ສາມາດສະແດງຕົວອັກສອນໄດ້ 16 ໂຕອັກສອນຕໍ່ເຖວ ແລະ ມີຫັງໝົດ 2 ແຖວ ລວມເປັນ 32 ແຖວ.
- ຫຼັກການເຮັດວຽກຂອງ 16x2 LCD Display ແມ່ນໃຊ້ Liquid Crystal ໃນການຄວບຄຸມທີ່ສ່ອງ ຜ່ານດ້ານຫຼັງຈຳ.
- Pin Guide ຂອງຈຳ LCD ແມ່ນຈະມີຫັງໝົດ 16 ຂາດັ່ງລຸ່ມນີ້:

ຂາທີ່	ຂື້ Pin	ໜ້າທີ່
1	VSS (GND)	GND (0V)
2	VDD (VCC)	ໄຟລັງ (ມັກເປັນ +5V)
3	V0/VEE	ປັບຄວາມເຂັ້ມຂົງໄຕອັກສອນ (Contrast) ນິຍົມຕໍ່ກັບຕົວຕ້ານທານປັບຕ່າດ (Potentiometer)
4	RS (Register Select)	ກຳນົດຊະນິດຂອງຂໍ້ມູນທີ່ສີ່ງ: High ສໍາລັບ Data, Low ສໍາລັບ Command
5	RW (Read/Write)	ກຳນົດທິດທາງການສົ່ງຂໍ້ມູນ: High ສໍາລັບ Read, Low ສໍາລັບ Write (ສ່ວນໃຫຍ່ ຈະຕໍ່ກັບ GND ເພື່ອໃຊ້ງານໃນໂສ່ມ Write)
6	E (Enable)	ໃຊ້ສົ່ງສັນຍານບອກໃຫ້ LCD ຮັບຂໍ້ມູນທີ່ຂາ Data
7-14	D0-D7	ຂໍ້ມູນ (Data) ແລະ ຄໍາສົ່ງ (Command) ສາມາດເຊື່ອມຕໍ່ໄດ້ 2 ອຸບແບບ: 4-bit mode (ໃຊ້ D4-D7) ຫຼື 8-bit mode (ໃຊ້ D0-D7)
15	A (LED+)	ຂ້ອບວກຂອງໄຟ Backlight (ອາດຕໍ່ຜ່ານຕົວຕ້ານທານຈໍາກັດກະແສ)
16	K (LED-)	ຂ້ອລືບຂອງໄຟ Backlight (ຕໍ່ກັບ GND)

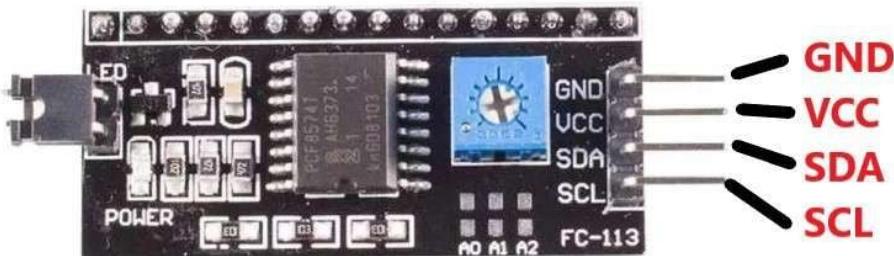


16. 12C Serial Adapter board module

- ແມ່ນໂມຄຸນຂະໜາດນ້ອຍທີ່ຖືກອອກແບບມາເພື່ອຫຼຸດຈຳນວນສາຍໄຟທີ່ຈໍາເປັນໃນການເຊື່ອມຕໍ່ຈະສະແດງຜົນ LCD (Liquid Crystal Display) ມາດຕະຖານ (ເຊັ່ນ 16x2 ຫຼື 20x4) ເຊົ້າກັບໄມໂຄຄອນໂທນເລີ ເຊັ່ນ Arduino ຫຼື ESP32.
 - ການເຮັດວຽກຂອງ 12C Serial Adapter ໃຊ້ຊີບຂະຫຍາຍ I/O ເຊັ່ນ: PCF8574 ເພື່ອເຮັດຫົ້າທີ່ເປັນ ຕົວແປງສັນຍານ ທີ່ຮັບຄໍາສັ່ງອະນຸກົມ I2C ຈາກໄມໂຄຄອນໂທນເລີ (ຜ່ານສາຍ SDA/SCL) ແລ້ວປ່ຽນມັນໃຫ້ເປັນສັນຍານຄວບຄຸມແບບຂະໜານທີ່ຈະ LCD ເຊົ້າໃຈ, ເຊິ່ງເປັນການຫຼຸດຈຳນວນສາຍໄຟທີ່ຕ້ອງການຈາກ 12-16 ສາຍ ໃຫ້ເຫຼືອພຽງ 4 ສາຍທີ່ນັ້ນ ເພື່ອ ປະຢັດຂາ I/O ຂອງໄມໂຄຄອນໂທນເລີໄດ້ຢ່າງມີປະສິດທິພາບ.
 - Pin Guide ຂອງ 12C Serial Adapter board module ມີ 4 ຂາດໍາລົ່ມນີ້:

ຊື່	ປະເພດ	ໜ້າທີ່	ເຊື່ອມຕໍ່ກັບ
GND	ໄຟຟ້າ	ຂາ GND (0V)	ຂາ GND ເຖິງMicrocontroller
VCC	ໄຟຟ້າ	ຂາແຫຼງຈ່າຍໄຟ (5V)	ຂາ 5V ເຖິງ Microcontroller
SDA	12C	Serial Data (ສາຍຂຶ້ນມູນອະນຸກົມ)	ຂາ SDA ເຖິງMicrocontroller
SCL	12C	Serial Clock (ສາຍສັນຍານໂມງອະນຸກົມ)	ຂາ SCL ເຖິງMicrocontroller

- ติวຢາງການໃຊ້ງານ: I2C Adapter ຊ່ວຍໃຫ້ທ່ານເຊື່ອມຕໍ່ LCD ກັບ Arduino ໂດຍໃຊ້ພຽງ 4 ສາຍ. ຫຼັງຈາກນັ້ນ, ເຮົາສາມາດຂຽນໂຄດສິນໆເພື່ອສັ່ງໃຫ້ຊີບ PCF8574 ແປດໍາສັ່ງ ແລະ ສະແດງ ຂໍຄວາມ ຫຼື ຂຶ້ມູນເຊັ່ນຊີ (ເຊັ່ນ: ອຸນຫະພູມ) ອອກເທິງຈໍ LCD ໄດ້ຢາງງ່າຍຕາຍ.

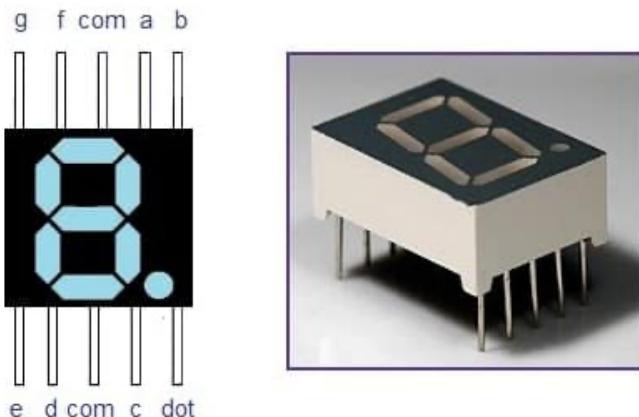


17. 7-segment display (Common Cathode +)

- 7-Segment Display ແມ່ນອຸປະກອນສະແດງຜົນເອເລັກໂຕ້ງນິກທີ່ໃຊ້ໃນການສະແດງຕົວເລກເລກຖານສືບ (0 ຫາ 9) ແລະຕົວອັກສອນບາງຕົວ. ມັນປະກອບດ້ວຍ LED (Light-Emitting Diode) ຈຳນວນເຈັດຫຸ່ວຍ (ເອີ້ນວ່າ " segments") ຈັດລຽງກັນໃນຮູບຊີ່ງເລກ "8", ບວກກັບຈຸດນັ້ນອຍໜຶ່ງ (DP/Dot Point) ສໍາລັບຈຸດທີ່ດສະນິຍົມ.
 - ການຮັດວຽກ: ຂາ Common (Cathode) ຖືກຕໍ່ກັບ GND (0V) ຕະຫຼອດເວລາ; ເພື່ອເປີດ Segment ໄດໜຶ່ງ, ຕ້ອງປ້ອນສັນຍານ HIGH (+VCC) ໄປຫາຂາ Anode ຂອງ Segment ນັ້ນ; ແລະປ້ອນ LOW (GND) ເພື່ອປິດມັນ.
 - Pin Guide:

ชื่อ segment	หมายความ	Pin ที่ใช้หัวไป
COM	Common Cathode (ข้อลิบร์วม)	3, 8 (ต่อกับ GND)
a, b, c, d, e, f, g	Segments (LEDs)	1, 2, 4, 5, 6, 7, 9, 10 (ต่อกับตีก็อกควบคุม)
DP (h)	Decimal Point (จุดขีดสะสม)	คาดจะเป็น Pin 1 หรือ 5

- **ពិរម្ភោះ:** សាមាតុដើម្បីបំពេញមូលកិច្ចទូនដើម្បីខ្លួនត្រង់នៅលើវគ្គភាពខ្លួន: ជីវមុំ, នាយី និង វិនាយី ឬ សាមាតសេដ្ឋកែវតាមការងារប៉ុណ្ណោះ។

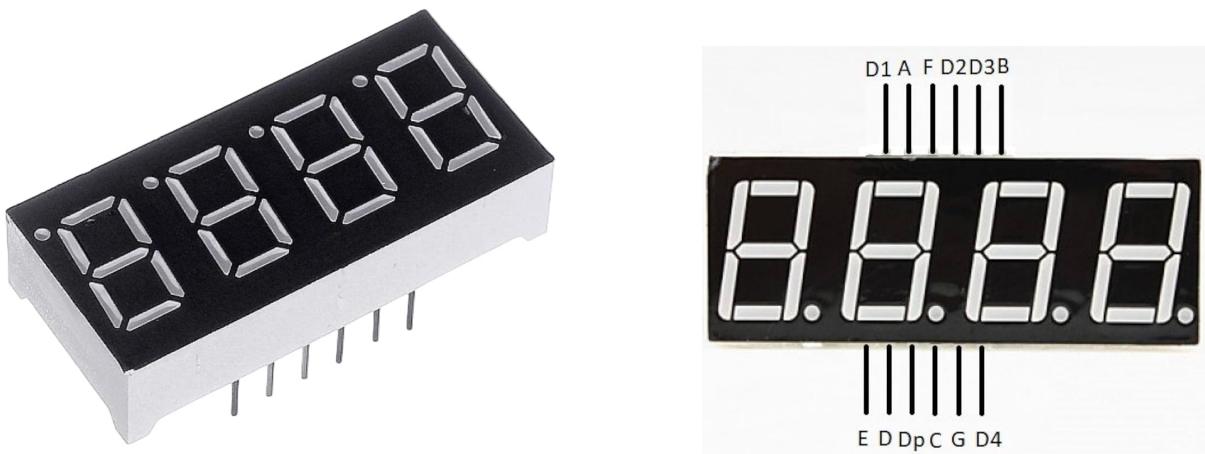


18. 4-Digit 7-Segment Display

- ແມ່ນຈະສະແດງຜົນທີ່ປະກອບດ້ວຍ ສື່ ຕົວເລກແຍກວັນ ທີ່ຖືກລວມເຂົ້າໃນຊຸດດຽວ. ແຕ່ລະຕົວເລກ ແມ່ນປະກອບດ້ວຍ ເຈັດສ່ວນ (segments) ຂອງໄຟ LED ທີ່ຈັດລຽງເປັນຮູບແບບເລກ 8 (ປຶກກະຕິແລ້ວມີສ່ວນທີ່ 8 ເປັນຈຸດທິດສະນິຍົມ, ເຊັ່ນວ່າ DP).
- ການເຮັດວຽກຂອງ 4-Digit 7-Segment Display ສາມາດຫຼຸດການໃຊ້ Pins ຈາກ 32 Pins ມາ ເປັນ 12 Pins ໂດຍການເຊື່ອມຕໍ່ Segments ທີ່ຄີກັນເຂົ້າກັນ (8 Pins) ແລະ ຄວບຄຸມການເປີດ-ປິດ ແຕ່ລະຫຼັກ (Digit) ແຍກັນ (4 Pins), ຈາກນັ້ນສະຫຼັບການສະແດງຜົນຢ່າງໄວວ່າ ເພື່ອສ້າງພາບ ໃຫ້ ເຫັນທັງ 4 ຕົວເລກສະແດງພ້ອມກັນ.
- Pin Guide ຂອງ 4-Digit 7-Segment Display ປະກອບມີ 8 Segment Pins + 4 Digit Pins ດັ່ງລຸ່ມນີ້:

Pin type	Pin	ໜ້າທີ່	Anode	Cathode
Segments	A, B, C, D, E, F, G	ຄວບຄຸມການເປີດ/ປິດ ຂອງແຕ່ລະສ່ວນ (LED).	LOW = ON, HIGH = OFF	HIGH = ON, LOW = OFF
Segments	DP (Decimal Point)	ຄວບຄຸມຈຸດທິດສະນິຍົມ.	LOW = ON, HIGH = OFF	HIGH = ON, LOW = OFF
Digits	D1, D2, D3, D4	ເປີດໃຊ້ງານ (Enable) ຕົວເລກແຕ່ລະໂຕ.	LOW = ON, HIGH = OFF	HIGH = ON, LOW = OFF

ຕົວຢ່າງ: ໂມງດິຈິຕອລ (Digital Clocks): ໃຊ້ສະແດງຊ່ວໂມງແລະນາທີ (ເຊັ່ນ: 10:30). ຈຸດທິດສະນິຍົມ (DP) ມັກຈະຖືກນຳໃຊ້ເປັນສອງຈຸດ (Colon) ລະຫວ່າງຊ່ວໂມງແລະນາທີ.



19. 8x8 Dot Matrix display

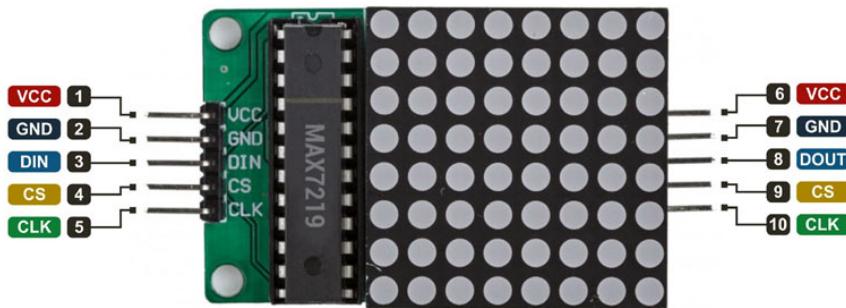
- 8x8 Dot Matrix Display ແມ່ນຈະສະແດງຜົນຂະໜາດນົມໝຍ້ທີ່ປະກອບດ້ວຍໄຟ LED (Light-Emitting Diode) ຈໍານວນ 64 ດອກ ທີ່ຖືກຈັດລຽງເປັນຕາຕະລາງຂະໜາດ 8 ແຖວ (Rows) ແລະ 8 ຊັ້ນ (Columns). ແຕ່ລະດອກ LED ໃນຕາຕະລາງມີເອີ້ນວ່າ "ຈຸດ" ຫຼື "Dot".
- 8x8 Dot Matrix Display ເຮັດວຽກໂດຍການນຳໃຊ້ຫຼັກການ Multiplexing (ການສະແດນ) ເພື່ອ ຄວບຄຸມໄຟ LED 64 ດອກດ້ວຍຈໍານວນຂໍທີ່ຈໍາກັດ (16 ຂໍ້ 3 ຂໍ້ 1 ເມື່ອໃຊ້ IC Driver): ລະບົບ

ຈະ ເປີດໄຟ LED ເປັນແຖວງຢ່າງວ່ອງໄວສະຫຼັບກັນ (ເຊັ່ນ: ເປີດແຖວທີ 1, ປິດ, ເປີດແຖວທີ 2, ປິດ), ໂດຍສາຍຕາຂອງຄືນເຮົາຈະເຫັນວ່າໄຟທັງໝົດທີ່ຕ້ອງການສະຫວ່າງນັ້ນ ຕິດຢູ່ຕະຫຼອດເວລາ ຍ້ອນຄວາມໄວໃນການສະຫຼັບທີ່ສູງ.

- Pin Guide ຂອງ Dot Matrix ລວມມີ 16 ຂາ ແລ້ວ 8 ຂາສໍາລັບແຖວ ສ່ວນອີກ 8 ຂາສໍາລັບຖຸນ.

ຂາ	ໜັກທີ່	ເຊື່ອມຕໍ່ກັບ
VCC	ແຮງດັນໄຟຟ້າບວກ (ປີກກະຕິແມ່ນ 5V)	5V
GND	ສາຍດິນ (Ground)	GND
DIN (Data In)	ຂາຮັບຂໍ້ມູນເຂົ້າ	ຂາ Digital (ເຊັ່ນ: D11)
CS (Chip Select)	ຂາເລືອກ Chip (ເປີດ/ປິດການສິ່ງຂໍ້ມູນ)	ຂາ Digital (ເຊັ່ນ: D10)
CLK (Clock)	ຂາສັນຍານ Clock (ກຳນົດຈັງຫວະຂໍ້ມູນ)	ຂາ Digital (ເຊັ່ນ: D13)

- ຕົວຢ່າງການນຳໃຊ້: ການສະແດງຂໍ້ຄວາມ ເຊິ່ງໃຊ້ສໍາລັບສະແດງຂໍ້ຄວາມຂະໜາດຍາວ ໂດຍໃຫ້ຕົວ ອັກສອນເລື່ອນຜ່ານຈໍສະແດງຜົນ, ໜ້າສໍາລັບປ້າຍໂຄສະນາຂະໜາດນົມ້ອຍ ຫຼື ການແຈ້ງເຕືອນ.



V. Sensors and Input Modules

20. Temperature and humidity sensor (DHT11)

- DHT11 (Digital Humidity and Temperature Sensor) ແມ່ນເຊັ່ນເຊີແບບດິຈິຕອລ (Digital) ລາຄາຖືກ ທີ່ສາມາດວັດແທກຄ່າ ຄວາມຊຸ່ມຊຶ່ນສຳພັດ (Relative Humidity - RH) ແລະ ອຸນຫະພູມ (Temperature) ຂອງອາກາດໄດ້.
- ການຮັດວຽກຂອງເຊັ່ນເຊີ DHT11 ແມ່ນອີງໃສ່ການວັດແທກຄ່າຄວາມຊຸ່ມຊຶ່ນ ແລະ ອຸນຫະພູມ ແບບອະນາລັອກ (Analog) ພາຍໃນ, ແລ້ວປັບປຸງຄ່າເຫຼົ່ານັ້ນໃຫ້ເປັນຂໍ້ມູນແບບດິຈິຕອລ (Digital) ເພື່ອສົ່ງໃຫ້ໄມໂຄຣຄອນໂທນເລີຫຼັກ (ເຊັ່ນ: Arduino) ນໍາໄປໃຊ້.
- Pin Guide ຂອງ DHT11 ມີສອງຮູບແບບຫຼັກ: ແບບເຊັ່ນເຊີດ່ວວ (4 Pins) ແລະ ແບບໂມດຸນ (3 Pins, ໂດຍປີກກະຕິມີ Pull-up Resistor ມາພ້ອມ).

Pin	ໜັກທີ່	ເຊື່ອມຕໍ່ກັບ
VCC (Pin 1)	ແຮງດັນໄຟຟ້າລົງ	ເຊື່ອມຕໍ່ກັບ 3.3V ຫຼື 5V ຂອງ Microcontroller
Data (Pin 2)	ສາຍຂໍ້ມູນແບບສາຍດ່ວວ	ເຊື່ອມຕໍ່ກັບ Digital I/O Pin ຂອງ Microcontroller
NC (Pin 3)	ບໍ່ໄດ້ເຊື່ອມຕໍ່	ປະໄວ້
GND (Pin 4)	0V	ເຊື່ອມຕໍ່ກັບ GND ຂອງ Microcontroller

- ពិវឌ្ឍការណែនាំធ្វើឡើង: DHT11 ត្រូវបានដោះស្រាយដើម្បីរកចុះតម្លៃទូទៅនៃការងារ និង គេងការងារ។

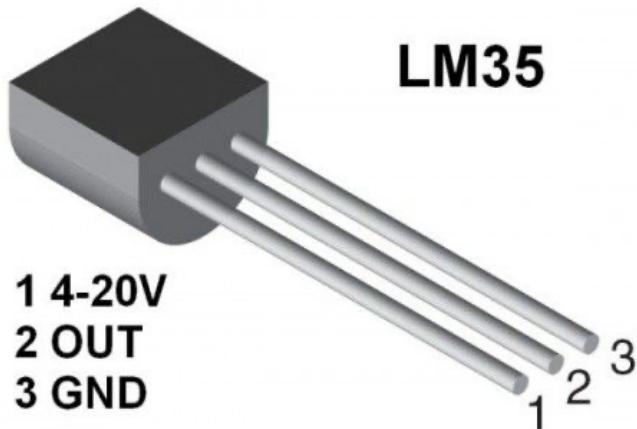


21. LM35 Temperature Sensor

- LM35 Temperature Sensor ແມ່ນ ເຊັນເຊີວັດອຸນຫະພູມແບບວິຈອນລວມ (Integrated-Circuit Temperature Sensor) ທີ່ມີຄວາມແມ່ນຍໍາສູງ ເຊິ່ງໃຫ້ຜົນອອກເປັນສັນຍານແຮງດັນອະນາລັອກທີ່ເປັນເສັ້ນຊື່ (Linear Output) ໂດຍກົງກັບອຸນຫະພູມໃນ ອົງສາເຊັນຊຽວສ ($^{\circ}\text{C}$).
 - LM35 ເຮັດວຽກໂດຍອີງໃສ່ຫຼັກການຂອງ ເຊັນເຊີອຸນຫະພູມແບບແຖບພະລັງງານຊີລິຄອນ (Silicon Bandgap Temperature Sensor). ໂດຍພື້ນຖານແລ້ວ, ມັນໃຊ້ຄຸນສົມບັດທີ່ການຕົກລົງຂອງແຮງດັນໄຟຟ້າຂ້າມຮອຍຕໍ່ຂອງສິ່ງທີ່ນຳໄຟຟ້າ (ເຊັ່ນ: ໄດໂອດ ຫຼື ທອນາຊີສເຕີ) ຈະບ່ຽນແປງໃນລັກສະນະທີ່ສາມາດຄາດຕິໄດ້ເມື່ອອຸນຫະພູມປ່ຽນແປງ.
 - Pin Guide ຂອງ LM35 ຈະມີ 3 ຂາດັ່ງລົມນີ້:

Pin	ໜ້າທີ່	ການເຊື້ອມຕໍ່
VCC	ແຮງດັນໄຟ	5V
Vout	ສັນຍາແຮງດັນ output	ຕໍ່ໃສ່ Pin ເຊັ່ນ: A0
GND	ສາຍດິນ	GND

- ตัวอย่างงานใช้งาน: LM35 สามารถเป็นอุปกรณ์ที่สามารถอ่านอุณหภูมิในสิ่งแวดล้อม และนำไปใช้ในระบบอัตโนมัติ เช่น: อุณหภูมิในห้อง, สถานที่ที่เรียกว่าเป็นตัว。

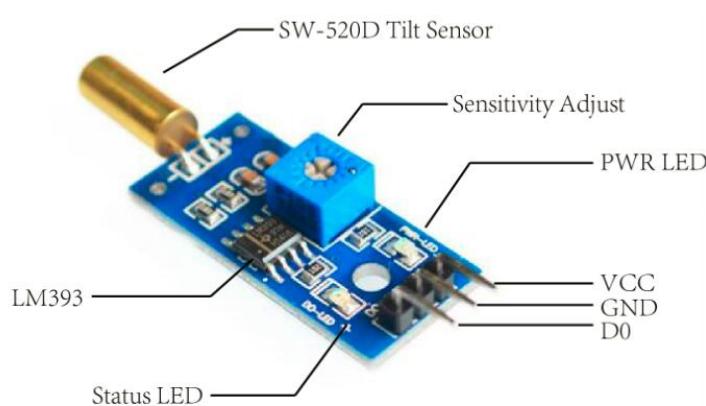


22. Tilt sensor (x2)

- Tilt sensor ແມ່ນອຸປະກອນທີ່ສາມາດກວດຈັບການວາງທິດຫາງ ຫຼື ມຸມອຽງຂອງວັດຖຸ.
- Tilt Sensor ເຮັດວຽກຄືກັບສະວິດທີ່ຄວບຄຸມດ້ວຍແຮງໂນມ໌ທ່ວງ: ພາຍໃນມີ ລູກກົມໄລໜະ ທີ່ເຊື່ອມຕໍ່ສອງຂົ້ວໄຟຟ້າ; ເມື່ອເຊັນເຊີ ຕັ້ງຊື່ (ປົກກະຕິ) ລູກກົມຈະເຊື່ອມຕໍ່ວົງຈອນ ແລະສິ່ງສັນຍານ LOW (0); ແຕ່ເມື່ອມັນ ຖືກອຽງ, ລູກກົມຈະກັ້ງອອກ, ເຮັດໃຫ້ວົງຈອນ ຕັດ ແລະສິ່ງສັນຍານ HIGH (1), ເຊິ່ງບໍ່ມີບອກເຕິງການປ່ຽນແປງຕຳແໜ່ງ.
- Pin Guide ຂອງ Tilt Sensor:

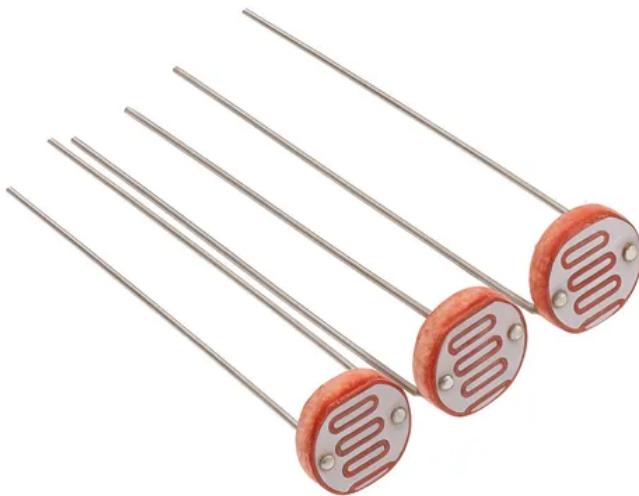
Pin	ເຊື່ອມຕໍ່	ພັງຂຶ້ນ
VCC	5V	ແຫຼ່ງຈ່າຍໄຟ
GND	GND	ສາຍດິນ
DO	Digital Pin	ສິ່ງສັນຍານດິຈິຕອນອອກ

- ຕົວຢ່າງການນຳໃຊ້: Tilt sensor ສາມາດນຳໃຊ້ເປັນສັນຍານເຕືອນກວດຈັບການລົ້ມສໍາລັບຜູ້ສູງອາບຸ ເມື່ອຜູ້ສູງອາບຸລົ້ມ ແຊັນເຊີຈະມີສັນຍານເຕືອນອອກມາ.



23. Photoresistor (LDRs x3)

- Photoresistor ແມ່ນອຸປະກອນທີ່ຮັດຈັກກັນໃນຊື່ LDR (Light Dependent Resistor) ຫຼື ຕົວຕ້ານທານທີ່ຂຶ້ນກັບແສງ ຫຼື ແມ່ນອຸປະກອນອີເລັກໂທຣນິກທີ່ເປັນຕົວຕ້ານທານແບບພື້ນ.
- ຫຼັກການຮັດວຽກຂອງ Photoresistor ແມ່ນຮັດວຽກໂດຍມີການປ່ຽນແປງຄວາມຕ້ານທານ: ເມື່ອບ່ອນມີດ ຄວາມຕ້ານທານຈະຂຶ້ນຫຼາຍ ແລະ ເມື່ອມີແສງກະທົບໃລ່ ຄວາມຕ້ານທານຈະຫຼຸດລົງເປັນຕົ້ນ.
- Photoresistor ບໍ່ມີ Pin ທີ່ກໍານົດ ເພະມັນເປັນອຸປະກອນທີ່ບໍ່ມີຂຶ້ວ ເຊິ່ງພວກເຮົາຈະສາມາດຕໍ່ຂາ ໄດ້ຂາໜຶ່ງເຂົ້າກັບແຫຼ່ງໄຟຟ້າ ແລະ ອີກາຫຶ່ງເຂົ້າກັບວິງຈອນໄດ້.
- ຕົວຢ່າງການນຳໃຊ້: ໃຊ້ເປັນລະບົບເປີດ-ປິດໄຟອັດຕະໂນມັດ: ເມື່ອທ້ອງຟ້າມີດໄຟຈະເປີດ ຖ້າທ້ອງຟ້າແຈ້ງໄຟຈະປິດ.

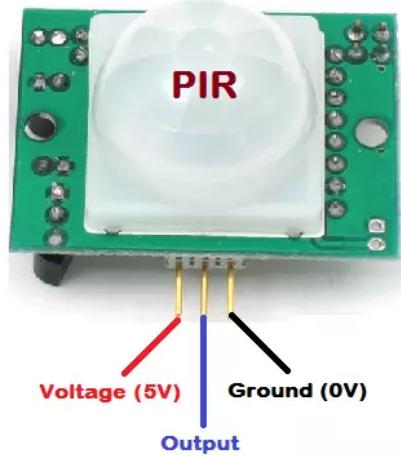


24. PIR Sensor

- PIR Sensor ແມ່ນເຊັນເຊີເອເລັກໂຕຣນິກທີ່ໃຊ້ໃນການ ກວດຈັບການເຄື່ອນໄຫວ ໂດຍການວັດແທກແສງອິນຟຣາເລດ (ຄວາມຮ້ອນ) ທີ່ປ່ອຍອອກມາຈາກວັດຖຸໃນພື້ນທີ່ທີ່ມັນກວດຈັບ.
- PIR Sensor ເຮັດວຽກໂດຍການກວດຈັບ ການປ່ຽນແປງ ຂອງແສງອິນຟຣາເລດ (ຄວາມຮ້ອນ) ທີ່ເຄື່ອນທີ່ຜ່ານພື້ນທີ່ກວດຈັບ, ໂດຍໃຊ້ pyroelectric sensor ທີ່ມີສອງສ່ວນຮັບຮູ້ທີ່ຕໍ່ກັນແບບກົງກັນຂ້າມ; ໃນສະພາບປົກກະຕິສອງສ່ວນຈະຮັກສາຄວາມສືມດຸນກັນ (Output LOW), ແຕ່ເມື່ອຮ້າງກາຍທີ່ອີບອຸ່ນເຄື່ອນຍ້າຍເຂົ້າໄປ, ມັນຈະຮັດໃຫ້ ຄວາມບໍ່ສືມດຸນຂອງຄວາມຮ້ອນ IR ຕີກໃສ່ສອງສ່ວນ, ເຊິ່ງກໍໃຫ້ເກີດການປ່ຽນແປງຂອງສັນຍາໄຟຟ້າທີ່ເປັນເຫດໃຫ້ Output ປ່ຽນເປັນ HIGH (ກວດພື້ນການເຄື່ອນໄຫວ).
- PIR Sensor ຈະມີ 3 ຂາຫຼັກດັ່ງນີ້:

Pin	ໜ້າທີ່	ລາຍລະອຽດ
VCC	ແຫຼ່ງຈ່າຍໄຟ	ເຊື່ອມຕໍ່ກັບ 5V
GND	ສາຍດິນ	ເຊື່ອມກັບຂົ້ວລົບ
Output	ສັນຍານຂາອອກ	ໃຫ້ສັນຍານດິຈິຕອນອອກມາ

- ຕົວຢ່າງ: ສາມາດໃຊ້ເປັນລະບົບຄວາມປອດໄຟເຊັ່ນ: ສັນຍານເຕືອນການບຸກລຸກໃນການກວດຈັບການເຄື່ອນໄຫວຂອງລົນໃນເຮືອນ ຫຼື ຫ້ອງການ ເພື່ອສິ່ງສຽງແຈ້ງເຕືອນ.



25. Ultrasonic Module

- Ultrasonic Module ແມ່ນອຸປະກອນອີເລັກໂທນິກທີ່ໃຊ້ຄື່ນສຽງຄວາມໃໝ່ສູງ (Ultrasonic Sound Waves) ເພື່ອ ວັດແທກໄລຍະຫ່າງ ລະຫວ່າງຕົວເຊັນເຊີກັບວັດຖຸເປົ້າໝາຍ.
- ຫຼັກການຮັດວຽກຂອງ Ultrasonic ວັດແທກໄລຍະຫ່າງໂດຍການໃຊ້ຫຼັກການ Time-of-Flight (ເວລາເດີນຫາງ) ຄື: ບ່ອຍຄື່ນສຽງຄວາມໃໝ່ສູງອອກໄປ, ວັດແທກ ເວລາ (t) ທີ່ຄື່ນສຽງໃຊ້ໃນການເດີນຫາງໄປແລະສະຫອນກັບມາ (Echo), ແລ້ວນຳໃຊ້ ຄວາມໄວຂອງສຽງ ເພື່ອຄຳນວນຫາໄລຍະຫ່າງ.
- Pin Guide ຂອງ Ultrasonic:
 - VCC ຕໍ່ກັບ +5V ຂອງ Microcontroller.
 - GND ຕໍ່ກັບ GND ຂອງ Microcontroller.
 - Trig ຕໍ່ກັບ Digital Output Pin ໃດໜຶ່ງ.
 - Echo ຕໍ່ກັບ Digital Input Pin ໃດໜຶ່ງ.
- ຕົວຢ່າງ: ເຊັນເຊີຈອດລົດເຊິ່ງມັນຈະມີເຊັນເຊີຕິດຢູ່ດ້ານຫລັງຂອງລົດພ້ອມກ້ອງ ເມື່ອເວລາເຮົາ ຖອຍຫລັງເຊັນເຊິ່ງມັນຈະບອກໄລຍະຫ່າງລະຫວ່າງລົດກັບກຳເພິ່ງ.

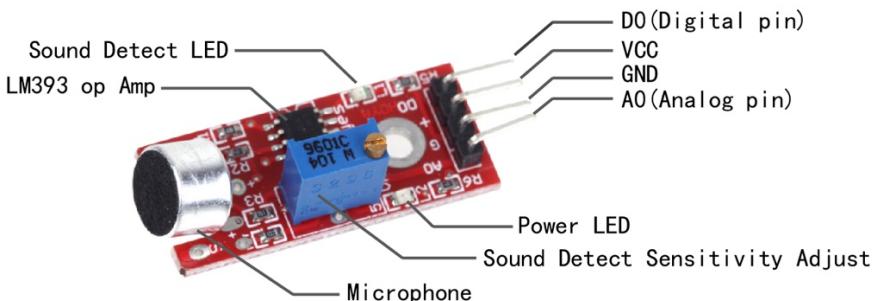


26. Sound Sensor

- Sound Sensor ແມ່ນອຸປະກອນເອເລັກໂຕຣນິກທີ່ຖືກອອກແບບມາເພື່ອກວດຈັບຄົ້ນສຽງ (ການສັນສະເໜີອນ) ແລະປ່ຽນຄົ້ນສຽງເຫຼົ່ານັ້ນໃຫ້ເປັນ ສັນຍານໄຟຟ້າ ທີ່ສາມາດວັດແທກໄດ້.
- ເຊັ່ນເຊີສຽງເຮັດວຽກໂດຍການທີ່ໄມໂຄຣໂຟນຈະ ຮັບຄົ້ນສຽງ ແລ້ວປ່ຽນການສັນສະເໜີອນນັ້ນໃຫ້ເປັນ ສັນຍານໄຟຟ້າອະນາລັອກທີ່ອ່ອນ, ຈາກນັ້ນວົງຈອນຈະ ຂະຫຍາຍສັນຍານ ດັ່ງກ່າວໃຫ້ແຮງຂຶ້ນ.
- Pin Guide ຂອງ Sound Sensor ຈະມີ 4 ຂາດັ່ງນີ້:

Pin	ໜ້າທີ່	ເຊື່ອມຕໍ່
VCC	ແຫຼ່ງຈ່າຍໄຟ	5V or 3.3V
GND	ສາຍດິນ	GND
D0	ສັນຍານດິຈິຕອນ	Digital Pin
A0	ສັນຍານອະນາລັອກ	Digital Pin

- ຕົວຢ່າງ: ເປີດ-ປິດໂດຍຜ່ານການຄວບຄຸມສຽງຕົບມື ເມື່ອເຮົາຕົບມືເຫື່ອທຳອິດດອກໄຟຈະເປີດ ຖ້າຕົບອິກເຫື່ອດອກໄຟຈະດັບ.

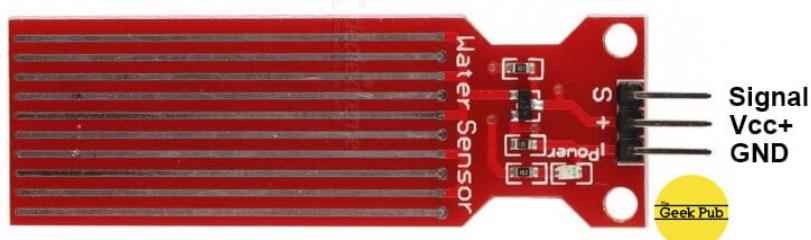


27. Water Sensor

- Water Sensor ແມ່ນອຸປະກອນອີເລັກໂທຣນິກທີ່ໃຊ້ໃນການກວດຈັບການມີປູ້ຂອງນ້ຳ, ລະດັບນ້ຳ ຫຼື ຜິນ. ມັນສາມາດປ່ຽນຄ່າການວັດແທກເຫຼົ່ານັ້ນໃຫ້ເປັນສັນຍານໄຟຟ້າ (ເຊັ່ນ: ລະດັບແຮງດັນໄຟຟ້າ) ເພື່ອໃຫ້ໄມໂຄຣຄອນໂທຣເລີ (ເຊັ່ນ: Arduino) ສາມາດອ່ານ ແລະ ປະມວນຜົນໄດ້.
- Water Sensor ແມ່ນເຮັດວຽກໂດຍອີງໃສ່ຫຼັກການປ່ຽນແປງຄ່າ ຄວາມຕ້ານທານ ຂອງວົງຈອນໄຟຟ້າ ເມື່ອມີນ້າສໍາຜັດ.
- Pin Guide ຂອງ Water Sensor ຈະມີ 3 ຂາດັ່ງລຸ່ມນີ້:

Pin	ໜ້າທີ່	ເຊື່ອມຕໍ່
VCC	ແຫຼ່ງຈ່າຍໄຟ	5V or 3.3V
GND	ສາຍດິນ	GND
S	ຂາສັນຍານອອກ	Analog Pin

- ຕົວຢ່າງ: ໃຊ້ວັດລະດັບນ້ຳໃນຖຸງ ຫຼື ອ່າງນ້ຳ ໂດຍວາງເຊັ່ນເຊີໄວ້ໃນຖຸງນ້ຳເພື່ອກວດສອບວ່ານ້ຳຢູ່ໃນລະດັບຕໍ່ ຫຼື ສູງເກີນໄປ.



28. Flame Sensor

- Flame Sensor ແມ່ນອຸປະກອນທີ່ຖືກອອກແບບມາເພື່ອກວດຈັບການມີຢູ່ຂອງແປວໄຟ ຫຼື ໄຟ. ມັນ ຖືກນຳໃຊ້ຢ່າງກວ້າງຂວາງໃນລະບົບປ້ອງກັນຄວາມປອດໄພ, ລະບົບເຕືອນໄຟໄຟໃໝ່, ແລະ ລະບົບ ອັດຕະໂນມັດອຸດສາຫະກຳ ເພື່ອຕິດຕາມການເຜົາໃໝ່.
- Flame Sensor ເຮັດວຽກໂດຍການກວດຈັບແສງສະຫວ່າງ ຫຼື ລັງສີສະເພາະທີ່ປ່ອຍອອກມາຈາກ ແປວໄຟໃນລະຫວ່າງການເຜົາໃໝ່. ໂດຍທົ່ວໄປ, ມັນຈະກວດຈັບ ລັງສີອິນຟຣາເຣດ (IR) ຫຼື ລັງສີ ຂັ້ນຕາວີໂອເລດ (UV).
- Pin Guide ຂອງ Flame Sensor ຈະມີ 4 ຂາດັ່ງລຸ່ມນີ້:

Pin	ໜ້າທີ່	ເຊື່ອມຕໍ່
VCC	ແຫຼ່ງຈ່າຍໄຟ	5V or 3.3V
GND	ສາຍດິນ	GND
D0	ສັນຍານດິຈິຕອນ (HIGH ບໍ່ມີແປວໄຟ ສ່ວນ LOW ກວດຝຶບໄຟ)	Digital Pin
A0	ສັນຍານອະນາລັອກ (ຄ່າຄວາມເຂັ້ມຂັ້ນຂອງແປວໄຟ)	Digital Pin

- ຕົວຢ່າງ: ໃຊ້ເປັນລະບົບເຕືອນໄຟໄຟໃໝ່ ເຊິ່ງເຮືອໃດຕິດຕັ້ງໃສ່ໃນອາຄານເພື່ອກວດຈັບໄຟໄຟໃນໄລຍະ ເບື້ອງຕົ້ນ ແລະ ເປີດໃຊ້ງານເຕືອນໄຟ.



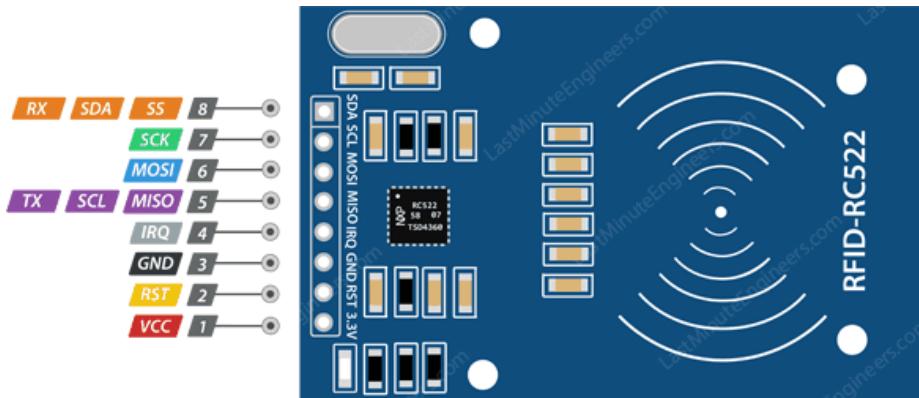
29. RFID Module

- ໂມຄຸນ RFID (RFID module) ແມ່ນສ່ວນປະກອບທາງເອເລັກໂຕຣນິກທີ່ມີຊີບອ່ານ (Reader chip) ແລະ ເສີ້ອາກາດ (Antenna), ເຊິ່ງມີໜ້າທີ່ສ້າງສະໜາມແມ່ນເຫຼັກໄຟຟ້າ (Electromagnetic field) ເພື່ອຕິດຕໍ່ສິ່ສານກັບປ້າຍ (Tags) ຫຼື ບັດ RFID (Cards) ທີ່ຢູ່ໃນໄລຍະໃກ້, ແລະ ອ່ານຂໍ້ມູນທີ່ເກັບໄວ້ ໃນປ້າຍນີ້.
- RFID Module ເຮັດວຽກໂດຍການທີ່ ເຄື່ອງອ່ານ ສ້າງສະໜາມແມ່ນເຫຼັກໄຟຟ້າເພື່ອໃຫ້ພະລັງງານແກ່ ປ້າຍ Passive (ທີ່ບໍ່ມີແບດຕີຣີ), ຈາກນັ້ນ ປ້າຍ ຈະສົ່ງຂໍ້ມູນສະເພາະຕົວ (ID) ກັບຄືນໄປຫາ ເຄື່ອງ ອ່ານ ເຊິ່ງຈະຖອດລະຫັດ ແລະ ສົ່ງຕໍ່ໃຫ້ລະບົບຄວບຄຸມເພື່ອປະມວນຜົນ (ເຊັ່ນ: ການອະນຸຍາດເຂົ້າ-ອອກ).
- Pin Guide:

Pin	ໜ້າທີ່
VCC	ແຫຼ່ງຈ່າຍໄຟ 3.3V
RST	ໃຊ້ເພື່ອກໍານົດການຮືເຊັດ ຫຼື ເປີດປິດການເຮັດວຽກ. ຕໍ່ກັບຂາ Digital ຂອງ MCU.
GND	ຕໍ່ສາຍດິນ
IRQ	ຂາລົບການ, ສາມາດໃຊ້ເພື່ອເຈັ້ງໃຫ້ MCU ຮູ່ວ່າກວດຝຶບປ້າຍແລ້ວ.

MISO	ສໍາລັບການສື່ສານ SPI: ສິ່ງຂໍ້ມູນອອກຈາກໂມດຸນໄປຫາ MCU.
MOSI	ສໍາລັບການສື່ສານ SPI: ສິ່ງຂໍ້ມູນເຂົ້າໂມດຸນຈາກ MCU.
SCK	ສໍາລັບການສື່ສານ SPI: ສັນຍານໂມງ.
SS/SDA	ສໍາລັບການສື່ສານ SPI: ເລືອກໂມດຸນ. ຕ່ັກບໍາ Digital ຂອງ MCU (ມັກຈະແມ່ນຂາ 10).

- ຕົວຢ່າງ: ສາມາດນຳໄປໃຊ້ເປັນໂມດຸນການຄວບຄຸມການເຂົ້າ-ອອກເຊັ່ນ: ບັດແປສໍາລັບເຂົ້າຫ້ອງ ຫຼື ເຂົ້າຫ້ອງການແທນກະແຈ.



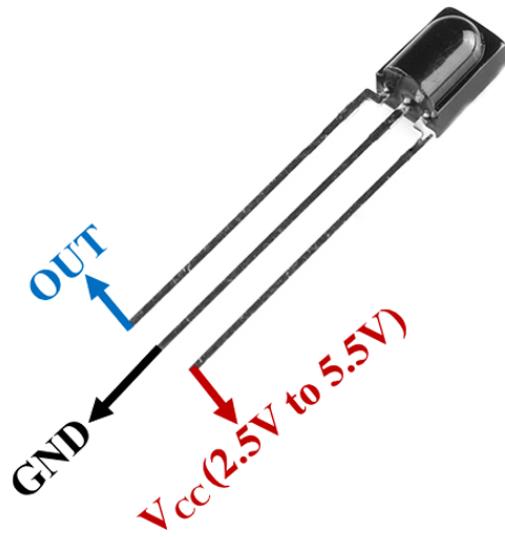
30. RFID tag

- RFID Tag ແມ່ນອຸປະກອນຂະໜາດນົ້ອຍທີ່ໃຊ້ສໍາລັບເກັບຂໍ້ມູນ ແລະ ສິ່ງສັນຍານຜ່ານຄົ້ນວິທະຍຸ ເພື່ອລະບຸຕົວຕືນຂອງວັດຖຸ, ສັດ ຫຼື ບຸກຄົນ ໂດຍບໍ່ຈໍາເປັນຕ້ອງມີການສໍາຜັດ ຫຼື ຢູ່ໃນລະດັບສາຍຕາ ຄືກັບ ບາໂຄດ Barcode.
- ໄລງສ້າງພາຍໃນຂອງ RFID Tag ຈະປະກອບມີ 2 ສ່ວນຫຼັກຄື:
 - Microchip: ໃຊ້ເກັບຂໍ້ມູນ (ເຊັ່ນ: ລະຫັດ ID).
 - Antenna (ສາຍອາກາດ): ເປັນຂີດລວດທີ່ໃຊ້ຮັບ-ສິ່ງສັນຍານຄົ້ນວິທະຍຸ.
- RFID Tag ປະກອບມີ 2 ປະເພດຄື:
 - Passive Tag: ບໍ່ມີແບດຕີ້ໃນຕົວ ເຊິ່ງມັນຈະຮັດດວງກໄດ້ກໍາທຳເມື່ອໄດ້ຮັບພະລັງງານຈາກເຕື່ອງອ່ານ (RFID Reader) ທີ່ສິ່ງຄົ້ນວິທະຍຸມາກະທິບ. ມັນມີລົາຄາຖືກ, ຂະໜາດນົ້ອຍ ແລະ ອາຍຸການໃຊ້ງານຍາວນານ.
 - Active Tag: ມີແບດຕີ້ໃນຕົວ ເຮັດໃຫ້ສາມາດສິ່ງສັນຍານໄດ້ໄກກວ່າ ແລະ ເກັບຂໍ້ມູນໄດ້ຫຼາຍກວ່າ ແຕ່ມີລົາຄາສູງ ແລະ ຂະໜາດໃຫຍ່ກວ່າ.
- ຮູບແບບຕ່າງໆທີ່ເຮົາສາມາດພືບໄດ້ມີ:
 - ແບບບັດ (Card): ບັດພະນັກງານ, ບັດເຂົ້າ-ອອກໂຮງແຮມ.
 - ແບບຫຼຽນ: ໃຊ້ສໍາລັບລະບົບ Keyless entry.
 - ແບບສະຕິກາເກີ: ໃຊ້ຕິດສິນຄ້າເພື່ອປ້ອງກັນການລັກຂະໂມຍ ຫຼື ຈັດການສະຕຼອກ.



31. Infrared (IR) Receiver

- ແມ່ນອຸປະກອນເອເລັກໂຕຣນິກທີ່ເຮັດໜ້າທີ່ຮັບສັນຍານແສງອິນຟາເຣດ (ເຊິ່ງເຮົາບໍ່ສາມາດເບິ່ງເຫັນ ດ້ວຍຕາເບິ່ງ) ຈາກຕົວສິ່ງສັນຍານເຊັ່ນ: ລິໂມດໂທລະຫັດ ແລ້ວປ່ຽນເປັນສັນຍານໄຟຟ້າເພື່ອໃຫ້ Microcontroller ເຊັ່ນ: Arduino ສາມາດນຳໄປປະມວນຜົນຕໍ່ໄດ້.
- ໂດຍປຶກກະຕິແລ້ວ ວານເຮັດວຽກຂອງ IR Receiver, ສັນຍານຈະຖືກສິ່ງອອກມາໃນຮູບແບບຂອງ ວານ “ກະພິບ” ດ້ວຍຄວາມຖື່ທີ່ແມ່ນອນ (ສ່ວນຫຼາຍແມ່ນ 38kHz) ເພື່ອປ້ອງກັນການລົບກວນຈາກ ແສງແດດ ຫຼື ດອກໄຟພາຍໃນຮີອິນ. ຕົວຮັບ IR ຈະມີວິຈາອນກອງສັນຍານ (Demodulator) ຢູ່ພາຍ ໃນເພື່ອແຍກເອົາແຕ່ຂຶ້ມູນທີ່ຖືກສິ່ງມາແທ້ງ.
- Pin Guide: ສໍາລັບ IR Receiver ຈະມີ 3 ຂາດັ່ງນີ້:
 - ຂາ OUT ຕໍ່ເຂົ້າ Digital Pin
 - ຂາ GND ຕໍ່ເຂົ້າ GND
 - ຂາ VCC ຕໍ່ເຂົ້າໄຟລ້ວງ 5V
- ຕົວຢ່າງການໃຊ້ງານ: ໃຊ້ເປັນລິໂມດເປີດ-ປິດໄຟ, ເປັນລິໂມດຄວບຄຸມຂອງມຳຕີເປັນຕົ້ນ.

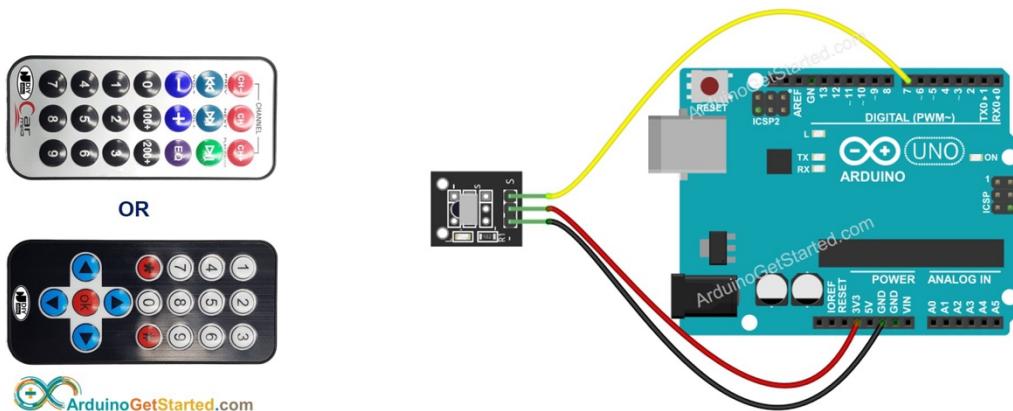


VI. Remote and Control

32. Infrared remote control

- Infrared (IR) Remote Control ແມ່ນອຸປະກອນທີ່ໃຊ້ຄືນເສົງໃນຄວາມທີ່ເພື່ອສິ່ງສັນຍານຄວບ ຄຸມອຸປະກອນໄຟຟ້າຕ່າງໆເຊັ່ນ: ໂທລະຫັດ, ເຄື່ອງປັບອາກາດ ຫຼື ໂປຣເຈັກເຕີ.
- ການເຮັດວຽກຂອງ IR Remote ຈະເປັງອອກເປັນ 2 ພາກສ່ວນຫຼັກດັ່ງນີ້:

- Transmitter (ຕົວສື່ງ): ຢູ່ທີ່ຕົວລືໂມດເຊິ່ງມີ IR LED ເຮັດຫັ້າທີ່ຍິງແສງອິນຟຣາເຮດອອກມາເປັນ ກໍາມະຈອນ (Pulses). ເມື່ອເຮົາກິດບຸ່ມ Microcontroller ພາຍໃນຈະປ່ຽນລະຫັດບຸ່ມນັ້ນໃຫ້ເປັນ ສັນຍານ Binary (0 ແລະ 1).
- Receiver (ຕົວຮັບ): ຢູ່ທີ່ຕົວອຸປະກອນເຊັ່ນໂທລະຫັດ ເຊິ່ງມີ IR Receiver Module ເຮັດຫັ້າທີ່ ກວດຈັບແສງທີ່ສື່ງມາ, ຖອດລະຫັດຄືນເປັນຄາສົ່ງເພື່ອໃຫ້ອຸປະກອນສາມາດເຮັດວຽກຕາມທີ່ ຕ້ອງການ.
- IR Receiver Pin Guide:
- Pin1: OUT ຕໍ່ເຂົ້າ Digital Pin.
- Pin2: GND ຕໍ່ເຂົ້າ GND.
- Pin3: VCC ຕໍ່ເຂົ້າໄຟລ໌ງງ 5V.
- ຕົວຢ່າງ: ໃຊ້ງານເປັນລະບົບຄວບຄຸມໄຟຟ້າ ແລະ ພັດລິມລິໂມດ.

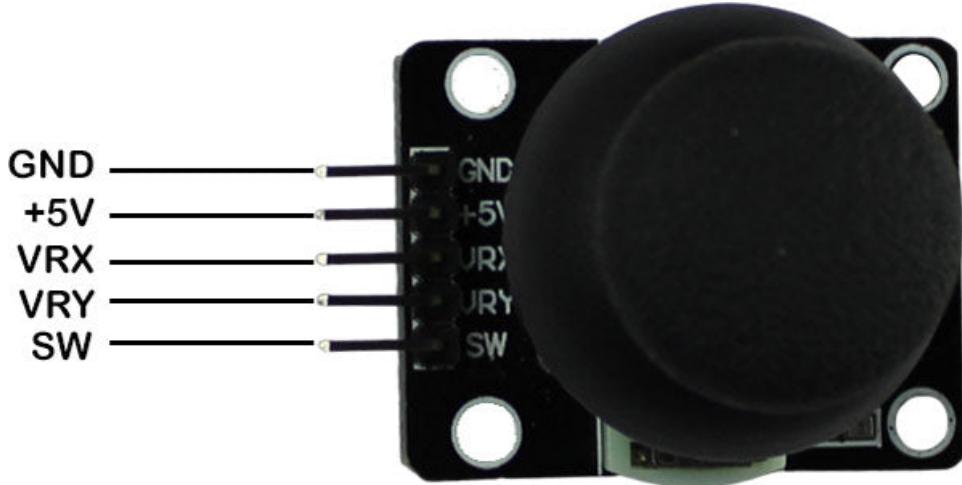


33. Joystick module

- ແມ່ນອຸປະກອນ Input ທີ່ປ່ຽນການເຄື່ອນທີ່ທາງກາຍະພາບ (ການຢູ່ຄັນບັງຄັບ) ໃຫ້ເປັນສັນຍານ ໄຟຟ້າ. ມັນປະກອບດ້ວຍ Potentiometer (ຕົວຕ້ານທານທີ່ປັບຄ່າໄດ້) 2 ຕົວ ທີ່ວາງຕັ້ງສາກັນ ເພື່ອວັກແທກແກ່ນ X (ຊ້າຍ ຫາ ຂວາ). ນອກຈາກນີ້, ຍັງມີບຸ່ມກິດສະວິດຢູ່ທາງກາງອີກດ້ວຍ.
- ຫຼັກການເຮັດວຽກ:
 - ແກ່ນ X ແລະ Y: ເມື່ອເຮົາຢູ່ບັງຄັບ, ມັນຈະໝູນໄປຄ່າຄວາມຕ້ານທານພາຍໃນ Potentiometer ເຮັດໃຫ້ແຮງດັນໄຟຟ້າທີ່ສື່ງອອກມາປ່ຽນແປງ. ໂດຍທົ່ວໄປຈະໃຫ້ຄ່າເປັນ Analog (0-1023).
 - ບຸ່ມກິດ (Z-axis): ເມື່ອເຮົາກິດຄັນບັງຄັບລົງຊື່ງ, ມັນຈະເຮັດວຽກຄືກັບບຸ່ມກິດທົ່ວໄປ (Push Button) ສິ່ງສັນຍານເປັນ Digital (0 ຫຼື 1).
- Pin Guide ຂອງ Joystick ຈະມີ 5 ຂາດັ່ງນີ້:

Pin	ຕໍ່ໃສ່ຂາ
GND	ຕໍ່ໃສ່ GND
VCC	ຕໍ່ໃສ່ 5V
VRx	ຕໍ່ໃສ່ຂາອະນາລັອກ ເຊັ່ນ (A0)
VRy	ຕໍ່ໃສ່ຂາອະນາລັອກເຊັ່ນ (A1)
SW	ຕໍ່ໃສ່ຂາ Digital

- **ពិវប្ប័យ:** ឱ្យបែនកៅំរូមិនការស្វោះរោគកៅំនាំង, លើកប៉ុងកាប និង ឱ្យបែនកៅំរូមិតាមបុណ្យគុណភាព។

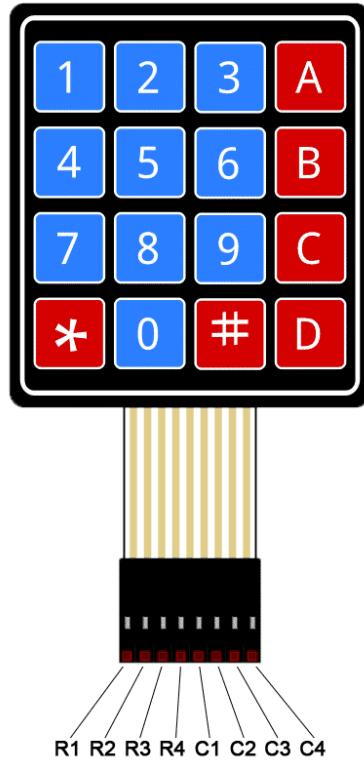


34. 4x4 Matrix Keyboard Module

- ແມ່ນອຸປະກອນປ້ອນຂໍ້ມູນ (Input Device) ທີ່ນີ້ຢືມໃຊ້ໃນວຽກງານ ເອເລັກໂຕຣນິກ ແລະ Microcontroller. ມັນຈະປະກອບດ້ວຍປຸ່ມທັງໝົດ 16 ປຸ່ມທີ່ຈະລຽງກັນໃນຮູບແບບ 4 ແຖວນອນ ແລະ 4 ແຖວຕັ້ງ.
 - ການເຮັດວຽກຂອງມັນຈະອາໄສການກວດສອບ “ຈຸດຕັດ” ລະຫວ່າງ Row ແລະ Column:
 - Scanning: Microcontroller ຈະສື່ງສັນຍານ (Logic LOW or HIGH) ໄປທີ່ແຖວນອນ ເທື່ອລະແຖວ.
 - Detection: ໃນຂະນະທີ່ສື່ງສັນຍານໄປແຖວນອນ, ມັນຈະຄອຍ “ອ່ານ” ດໍາຈາກແຖວຕັ້ງ.
 - Mapping: ຖ້າປຸ່ມໄດ້ໜຶ່ງຖືກກົດ, ວິຈອນລະຫວ່າງ Row ກັບ Column ມັນຈະເຊື່ອມຕໍ່ກັນ. ເມື່ອ Microcontroller ຮູ້ວ່າ Row ໄດ້ກຳລັງເຮັດວຽກ ແລະ Column ໄດ້ມີສັນຍານເຂົ້າມາ, ມັນກຳຈະລະບຸໄດ້ທັນທີວ່າແມ່ນປຸ່ມໃດ.
 - Pin Guide ຈະປະກອບມີ 8 ຂາດັ່ງນີ້:

Pin	Function	ລາຍລະອຽດ
1	R1	Row 1 (ແຖວນອນທີ 1)
2	R2	Row 2 (ແຖວນອນທີ 2)
3	R3	Row 3 (ແຖວນອນທີ 3)
4	R4	Row 4 (ແຖວນອນທີ 4)
5	C1	Column 1 (ແຖວຕັ້ງທີ 1)
6	C2	Column 2 (ແຖວຕັ້ງທີ 2)
7	C3	Column 3 (ແຖວຕັ້ງທີ 3)
8	C4	Column 4 (ແຖວຕັ້ງທີ 4)

- ติวปีภาษาไทย: ละบีบภิດลหัดผ่าน, เศียร์กิดเล็ก และ ภาษาตี้ๆถ่ำเม้มในจ. LCD.



35. Relay Module

- ແມ່ນອຸປະກອນສະວິດໄຟຟ້າທີ່ໃຊ້ສັນຍາໄຟຟ້າກະແສງົງທີ່ມີແຮງດັນຕໍ່າ ເພື່ອໄປ ຄວບຄຸມການປິດ-ເປີດ ຂອງວິງຈອນໄຟຟ້າທີ່ມີກຳລັງສູງ ຫຼື ແຮງດັນສູງ (ເຊັ່ນ: ໄຟບ້ານ 220V AC). ມັນເຮັດໜ້າທີ່ຄືກັບ “ຂົວຕໍ່” ທີ່ແບກສ່ວນຂອງການຄວບຄຸມອອກຈາກສ່ວນທີ່ມີກຳລັງໄຟສູງເພື່ອຄວາມປອດໄພຂອງອຸປະກອນໄຟຟ້າ.
- ຫຼັກການເຮັດວຽກ: ພາຍໃນ Relay ຈະມີກິດລວດພັນຢູ່ອ້ອມແກນເຫຼັກ. ເມື່ອມີກະແສໄຟຟ້າໃຫ້ ຜ່ານກິດລວດ ມັນຈະສ້າງສະໜາມແມ່ເຫຼັກຂຶ້ນມາເພື່ອດຶງແຜ່ນໜ້າສໍາຜັດ (Contact) ໃຫ້ປ່ຽນຕໍ່າ ແລ້ວ:

 - ສະພາວະປີກະຕິ: ໜ້າສໍາຜັດຈະແຕະຢູ່ທີ່ຂາ NC (Normally Closed).
 - ເມື່ອປ້ອນໄຟ: ແມ່ເຫຼັກຈະດຶງໜ້າສໍາຜັດໃຫ້ຢ້າຍໄປແຕະທີ່ຂາ NO (Normally Open) ແກ່ນເຮັດໃຫ້ວິງຈອນເຮັດວຽກ.

- Pin Guide ສ່ວນຫຼາຍຈະແບ່ງຂາອອກເປັນສອງຝ່າງຄື:

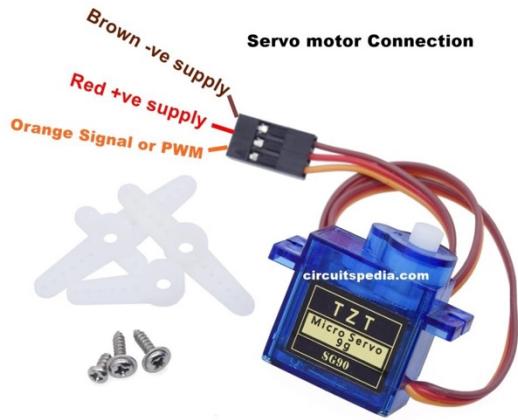
 - 1) ຜົງ Input (ຕໍ່ກັບ Microcontroller)
 - VCC: ຕໍ່ກັບ 5V
 - GND: ຕໍ່ກັບ GND
 - IN/Signal: ຂາຮັບສັນຍາຈາກ Arduino (HIGH or LOW) ເພື່ອສັ່ງໃຫ້ Relay ເຮັດວຽກ.
 - 2) ຜົງ Output (ຕໍ່ກັບອຸປະກອນໄຟຟ້າ)
 - COM (Common): ຂາກາງທີ່ຕ້ອງຕໍ່ໄຟລ້ຽງ (Line) ຂອງອຸປະກອນທີ່ຈະຄວບຄຸມເຂົ້າມາ.
 - NO (Normally Open): ໃນສະພາວະປີກະຕິ ວິງຈອນຈະ “ເປີດ” (ໄຟບໍ່ໃຫ້). ໄຟຈະໃຫ້ກຳຕໍ່ເມື່ອເຮັດສັ່ງ Trigger ເທົ່ານັ້ນ.

- NC (Normally Closed): ในสภาวะปีกจะติดไว้จนจะ “ปิด” (ໄຟໄຫຼຕະຫຼອດ). ໄຟຈະຕັດກຳຕໍ່ເມື່ອເຮົາສັ່ງ Trigger.
- ຕົວຢ່າງການນຳໃຊ້: ລະບົບເປີດ-ປິດໄຟອັດສະລິຍະ, ລະບົບທຶນນໍ້າອັດສະລິຍະ.



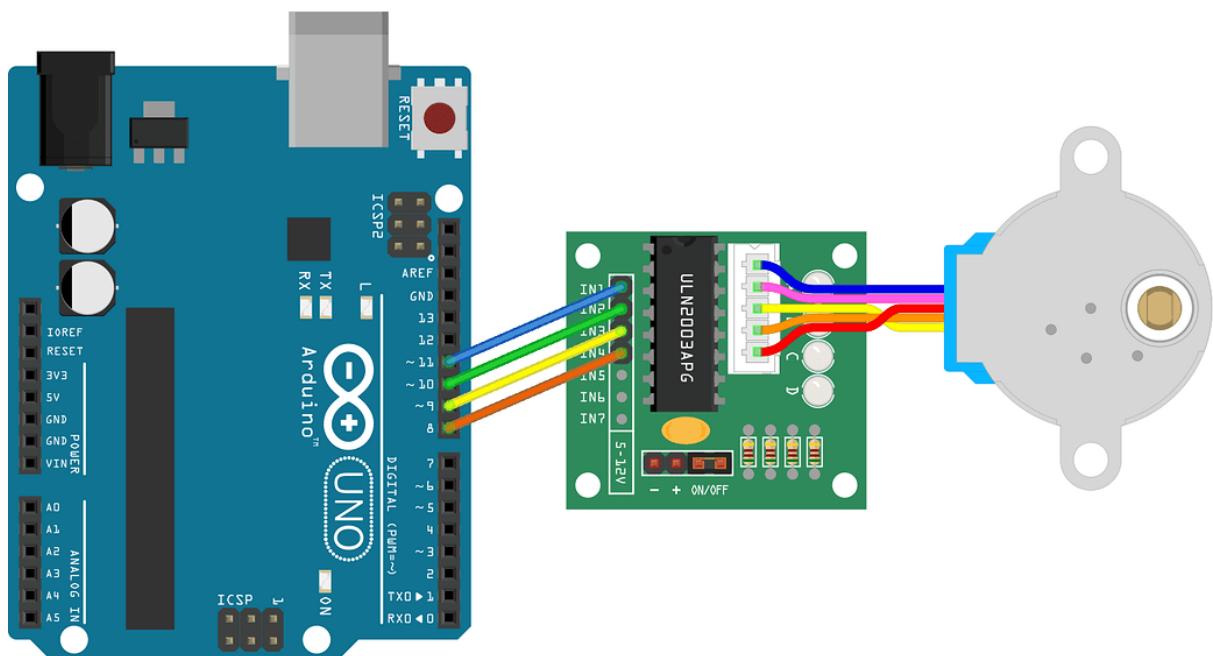
36. Servo motor

- ແມ່ນອຸປະກອນໄຟຟ້າຊະນິດນີ້ທີ່ສາມາດຄວບຄຸມຕໍ່າແໜ່ງ (Angle), ຄວາມໄວ (Speed) ແລະ ຄວາມຊັດເຈນ (Precision) ໄດ້ຢ່າງຖືກຕ້ອງ. ມັນແຕກຕ່າງຈາກມີເຕີທີ່ວ່າໄປທີ່ປິ່ນໄປເລື້ອຍໆ ເພະ Servo ສາມາດສັ່ງໃຫ້ຢູ່ໄຍ້ທີ່ອີງສາທີ່ເຮົາຕ້ອງການໄດ້ (ສ່ວນຫຼາຍແມ່ນ 0-180ອີງສາ).
- ການເຮັດວຽກຂອງ Servo Motor ອາໄສລະບົບ Closed-loop System (ລະບົບຄວບຄຸມແບບປັບປຸງ)
- ເຊິ່ງປະກອບມີ 4 ສ່ວນຫຼັກ:
 - 1) DC Motor: ຕົວຕົ້ນກຳນົດປິ່ນ.
 - 2) Gear System: ຊຸດເຟືອງເພື່ອເພີ່ມແຮງບົດ (Torque) ແລະ ຫຼຸດຄວາມໄວ.
 - 3) Potentiometer: ຕົວຕ້ານຫານທີ່ປັບຄ່າໄດ້ ເພື່ອເຮັດໜ້າທີ່ເປັນ Sensor ບອກຕໍ່າແໜ່ງຂອງແກ່ນມີເຕີ.
 - 4) Control Circuit: ວິຈອນຄວບຄຸມທີ່ຮັບສັນຍານຈາກພາຍນອກມາປຽບທຽບກັບຕໍ່າແໜ່ງປັດຈຸບັນ.
- ການຄວບຄຸມດ້ວຍສັນຍານ PWM: ເຮົາຄວບຄຸມ Servo ຜ່ານສັນຍານ PWM (Pulse Width Modulation). ຄວາມກວ້າງຂອງກຳມະຈອນໄຟຟ້າ (Pulse Width) ຈະເປັນຕົວກຳນົດອີງສາ:
 - 1ms pulse: ມີເຕີໄປທີ່ 0 ອີງສາ.
 - 1.5ms pulse: ມີເຕີຈະໄປທີ່ 90 ອີງສາ.
 - 2ms pulse: ມີເຕີຈະໄປທີ່ 180 ອີງສາ.
- Pin Guide ຈະມີສາຍໄຟ 3 ເສັ້ນດັ່ງນີ້:
 - ສິນ້າຕາມ: GND
 - ສີແດງ: VCC
 - ສີສັ້ນ: Signal
- ຕົວຢ່າງການໃຊ້ງານ: ລະບົບເປີດປະຕຸອັດຕະຖານມັດ, ລະບົບເປີດ-ປິດຝາ.



37. Stepper motor

- ແມ່ນມີເຕີໄຟຟ້າຊະນິດໜຶ່ງທີ່ປ່ຽນສັນຍານໄຟຟ້າໃຫ້ເປັນເຄື່ອນທີ່ໃນລັດສະນະເປັນ “ກ້າວ”. ຕ່າງຈາກ ມີເຕີທີ່ວ່າປະທິ່ນມູນແບບຕໍ່ເນື້ອງ, Stepper Motor ຈະໝູນໄປເຫຼືອລະມຸມທີ່ແມ່ນອນຮັດໃຫ້ເຮົາ ສາມາດຄວບຄຸມຕຳແໜ່ງ ແລະ ຄວາມໄວໄດ້ຢ່າງຊັດເຈນ.
- ການຮັດວຽກຂອງ Stepper Motor ຮັດວຽກໂດຍໃຊ້ຫຼັກການຂອງແມ່ເຫຼັກໄຟຟ້າ. ພາຍໃນ ປະກອບດ້ວຍ: Rotor (ສ່ວນໝູນ) ແລະ Stator (ສ່ວນຢູ່ກັບທີ່).
- Pin Guide:
 - 1) Stepper Motor Driver (ULN2003)
 - IN1, IN2, IN3, IN4: ຕໍ່ເຂົ້າກັບ Digital Pins ຂອງ Arduino.
 - VCC: ຕໍ່ກັບ 5V
 - GND: ຕໍ່ກັບ GND
 - White Connector: ສຽບສາຍຈາກ Stepper Motor ເຂົ້າໄດ້.
- ຕົວຢ່າງການນຳໃຊ້ງານ: ເຄື່ອງພິມ 3D, ແຂນກົມ Robot.



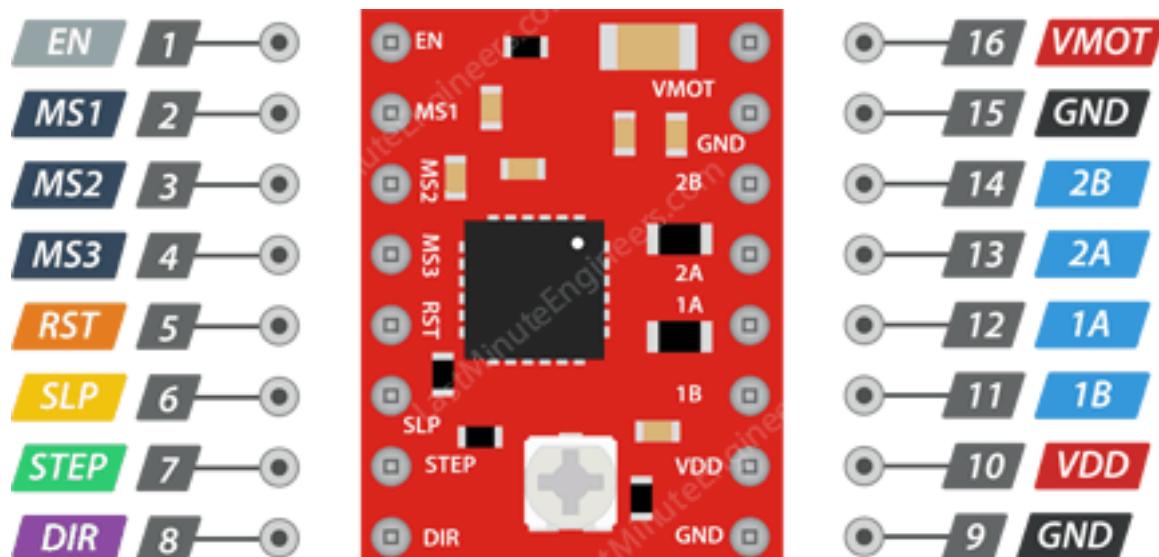
38. Stepper motor driver board

- ແມ່ນອຸປະກອນເອລັກໂຕຣນິກທີ່ເຮັດໜ້າທີ່ເປັນຂົວຕໍ່ລະຫວ່າງ Controller ກັບ ຕົວ Motor. ເນື່ອງຈາກ Controller ບໍ່ສາມາດຈ່າຍກະແສໄຟຟ້າໄດ້ພຽງພໍທີ່ຈະໝູນ Motor ໂດຍກິງໄດ້, Driver ຈຶ່ງຮັບໜ້າທີ່ສັນຍານຄໍາສັ່ງ (Logic Signal) ແລ້ວປ່ຽນເປັນພະລັງງານໄຟຟ້າທີ່ມີກາລັງສູງເພື່ອຂັບ ເຄືອນ Stepper motor ໃຫ້ໝູນຕາມຈັງຫວະທີ່ຕ້ອງການ.
- ການຮັດວຽກ:
 - ຮັບສັນຍານ: Driver ຈະຮັບສັນຍານ Pulse (STEP) ແລະ ທິດທາງ (DIR) ຈາກ Microcontroller.
 - ແປງສັນຍານ: ວົງຈອນພາຍໃນຈະຄິດໄລ່ວ່າຕ້ອງຈ່າຍໃຫ້ກັບຂົດລວດໄດ້ຂອງ Motor ວ່າໃນ ລໍາດັບໃດ.
 - ຂັບເຄືອນ: Driver ຈະປ່ອຍກະແສໄຟຟ້າຈາກເຫຼົ່ງຈ່າຍໄຟເຂົ້າສູ່ ມີເຕີເພື່ອສ້າງສະໜາມແມ່ ເຫຼັກເຮັດໃຫ້ແກ່ນມີເຕີໝູນເປັນບາດ.

- Pin Guide:

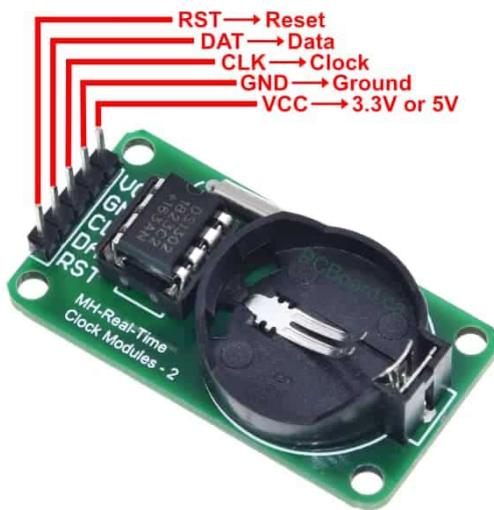
ປະເພດ	Pin	ລາຍລະອຽດ
1	STEP	ຮັບສັນຍານ Pulse
2	DIR	ກຳນົດທິດທາງການໝູນ
3	VCC	ໄຟຟ້າສຳລັບ Chip
4	GND	ສາຍຕົນ ຫຼື ຂົວລືບ
5	VMOT	ໄຟຟ້າ Motor
6	GND	ສາຍຕົນ ຫຼື ຂົວລືບ
7	1A, 1B, 2A, 2B	ຂາຕໍ່ເຂົ້າກັບສາຍຂອງ Stepper Motor
8	EN/Enable	ໃຊ້ສັ່ງເປີດ ຫຼື ປິດການຮັດວຽກຂອງ Driver
9	MS1, MS2, MS3	ໃຊ້ຕັ້ງຄ່າ Micro stepping

- ຕົວຢ່າງການນຳໃຊ້: ເຄືອງພິມ 3D, ແຂນກົນ Robot.



39. Real-time Clock Module DS1302

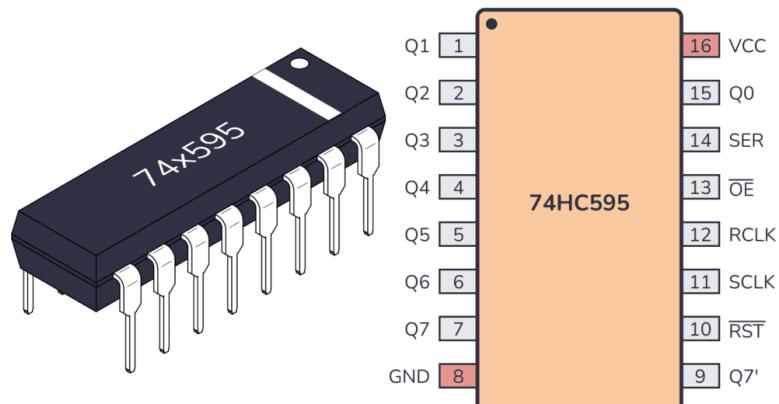
- ແມ່ນໂມດູນໂມງເວລາຈິງ (Real-Time Clock ຫຼື RTC) ທີ່ໄດ້ຮັບຄວາມນິຍົມໃນການໃຊ້ງານ ຮ່ວມກັບ Arduino ແລະ Microcontroller ອື່ນໆ. ມັນເຮັດຫັ້ນທີ່ນັບເວລາໃຫ້ເປັນປັດຈຸບັນຢ່ສະເໜີ ເຖິງແມ່ນວ່າພວກເຮົາຈະປິດໄຟລ່ຽງເຄື່ອງຄວບຄຸມຫຼັກໄປແລ້ວກຳຕາມ. DS1302 ແມ່ນ IC ທີ່ ບັນຈຸໂມງ ແລະ ປະຕິທິນພ້ອມກັບ RAM ຂະໜາດ 31 Bytes. ມັນສາມາດບອກເວລາໄດ້ທັງວິນາທີ, ຊົ່ວໂມງ, ວັນທີ, ເຕືອນ, ມື້ໄດ້ອາຫິດ ແລະ ປີ.
- ການຮັດວຽກຂອງ DS1302 ມັນຮັດວຽກໂດຍໃຊ້ຄິດຕັນ (Crystal Oscillator) ຄວາມຖື 32.768 kHz ເປັນຕົວສັນຍານຈັງຫວາະ. ການສື່ສານກັບ Microcontroller ຈະໃຊ້ລະບົບ Serial Communication ແບບ 3 ສາຍ (CE, I/O, SCLK) ເຊິ່ງບໍ່ແມ່ນ 12C ແຕ່ມີລັກສະນະຄ້າຍກັນ.
- Pin Guide ຂອງໂມດູນ DS1302 ຈະມີ 5 ຂາດັ່ງນີ້:
 - VCC: ຕໍ່ໃສ່ 5V
 - GND: ຕໍ່ໃສ່ GND
 - CLK: Serial Clock ໃຫ້ຈັງຫວາະໃນການສົ່ງຂໍ້ມູນ.
 - DAT (I/O): Data Input/Output ສາຍສົ່ງ ແລະ ຮັບຂໍ້ມູນເວລາ.
 - RST (CE): Reset ເປີດໃຊ້ງານການສື່ສານ.
- ຕົວຢ່າງການນຳໃຊ້ງານ: ລະບົບຕັ້ງເວລາ, ໂມງຕັ້ງໂຕະ ແລະ ລະບົບຈັບເວລາ.



40. 74HC595 Chip

- ແມ່ນຊືບ 8-bit Serial-in, Parallel-out Shift Register ທີ່ນິຍົມໃຊ້ໃນເອລັກໂຕຣນິກ ເພື່ອ ຂະຫຍາຍຂາຂອງ Output ຂອງ Microcontroller.
- ຫຼັກການຮັດວຽກຂອງມັນແມ່ນຮັບຂໍ້ມູນແບບ ລຽນແຖວ (Serial) ເຊິ່ງມາເທື່ອລະບົດແລ້ວປ່ຽນ ໃຫ້ອອກໄປແບບ ຂະໜານ ພ້ອມກັນ 8 Bit.
 - 1) Data Input: ສົ່ງຂໍ້ມູນ 0 ຫຼື 1 ເຊິ່ງໄປທີ່ຂາ Data.

- 2) Shifting: เมื่อมีสัญญาณ Clock สิ่งมา, ข้อมูลนั้นจะถูกย้ายเข้าไปเก็บไว้ใน Register ที่
 - ถ้ามีข้อมูลใหม่เข้ามาอีก, ข้อมูลเดิมจะถูกย้ายไป Register ต่อไป (ถ้าหากไม่มีแล้ว).
 - 3) Latching: เมื่อเรียกสิ่งข้อมูลที่บีบ 8 bit แล้ว, เรียกใช้สัญญาณไปที่ขา Latch เพื่อส่งให้ข้อมูลทั้งหมดที่อยู่ใน แทนที่ขา ออกไปเป็นแบบผ่านทาง Output พร้อมกัน.
- Pin Guide:
- | Pin | ຊື່ຂາ | ລາຍລະອຽດ |
|---------|--------------------|---|
| 1-7, 15 | Q0-Q7 | ຂາສິ່ງສັນຍານອອກໄປທາອຸປະກອນ |
| 8 | GND | ຕົ້ນໃສ່ຂຶ້ວລົບ |
| 9 | Q7' | ໃຊ້ສໍາລັບຕໍ່ໄປຫາຊີບ 74HC595 ໂຕຕັດໄປ |
| 10 | MR (MASTER RESET) | ຖ້າຕໍ່ໃສ່ LOW ຈະເປັນການ Reset ຂໍ້ມູນທັງໝົດ |
| 11 | SHCP (CLOCK) | ຂາຈັງຫວະ Clock ເພື່ອດຶງຂໍ້ມູນເຂົ້າ Shift Register |
| 12 | STCP (LATCH) | ຂາສັ່ງໃຫ້ຂໍ້ມູນອອກໄປທີ່ Output |
| 13 | OE (OUTPUT ENABLE) | ໃຊ້ເປີດ-ປິດ Output ຫ້າໝືດ |
| 14 | DS (DATA) | ຂາຮັບຂໍ້ມູນ Serial ຈາກ Microcontroller |
| 16 | VCC | ຕົ້ນໃສ່ໄຟລ້ວງ 5V |
- ຕົວຢ່າງການນຳໃຊ້: ການຮັດບ້າຍໄຟປະດັບ, ຄວບຄຸມ Relay Module ແລະ ໃຊ້ກັບຈຳ LCD.



ແນະນຳເຄື່ອງມື Software ທີ່ໃຊ້ເຂົ້າໃນການທິດລອງ

ເປັນຫຍຸງເຮົາຈຶ່ງຈໍາເປັນໃຊ້ເຄື່ອງມື Software?

ເພາະວ່າເຄື່ອງມືຊອບແວດເປັນເຄື່ອງມືທີ່ສໍາຄັນຊ່ວຍໃຫ້ເຮົາສາມາດອອກແບບວົງຈອນເອເລັກໂຕຣນິກ ແລະ ຂຽນຄໍາສັ່ງ Coding ເພື່ອໃຫ້ອຸປະກອນສາມາດຮັດວຽກໄດ້ກ່ອນລົງຮັດຕົວຈີງ.

1. TinkerCAD

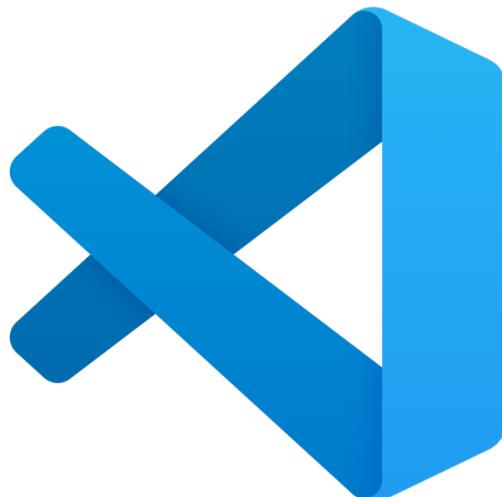
ແມ່ນໂປຣແກຣມທີ່ໃຊ້ອອກແບບ 3D Model ແລະ ຈໍາລອງວົງຈອນເອເລັກໂຕຣນິກແບບອອນລາຍ ແຊ່ງພັດທະນາໂດຍ Autodesk. ມັນຖືກອອກແບບມາເພື່ອໃຫ້ໃຊ້ງານຈ່າຍ, ເໝາະກັບຜູ້ເລີ່ມຕົ້ນຮຽນ, ນັກຮຽນ ແລະ ນັກພັດທະນາທີ່ຕ້ອງການຮັດໂປຣເຈັກ.



2. Visual Studio Code

ແມ່ນ Integrated Development Environment (IDE) ຈາກ Microsoft ເຊິ່ງເປັນເຄື່ອງມືຂຽນໂປຣແກຣມທີ່ຄືບວົງຈອນ ແລະ ຊ່ວຍອໍານວຍຄວາມສະດວກໃຫ້ແກ່ນັກພັດທະນາ (Developer) ໃນການຂຽນ Code, ກວດສອບຂໍຜິດພາດ (Debug) ແລະ ສ້າງ Software ຕ່າງໆ.

Visual Studio ສາມາດໃຊ້ເພື່ອພັດທະນາຫຼາຍຢ່າງເຊັ່ນ: Desktop App, Web Application, Mobile App, System Programming ແລະ Game Development ເປັນຕົ້ນ.



3. Arduino IDE

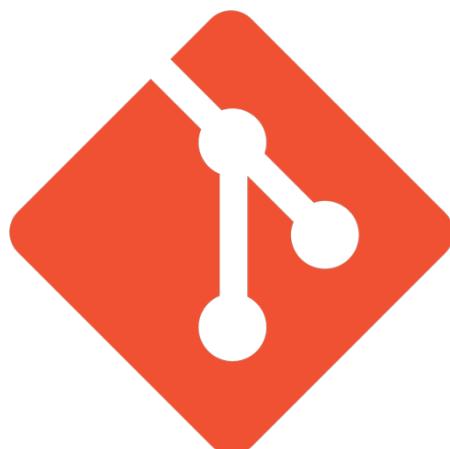
ແມ່ນຊອບແວດັ່ງນີ້ພື້ນຖານທີ່ນັກພັດທະນາ ແລະ ວິສະວະກອນໃຊ້ເພື່ອຂຽນ Code ແລະ Upload ຄໍາສົ່ງ ລົງໄປໃນບອດ Arduino ຫຼື Microcontroller ຕ່າງໆ.



4. Git

ແມ່ນລະບົບທີ່ໃຊ້ສໍາລັບ Version Control (ການຈັດການເວີຊັນຂອງ File) ທີ່ເປັນແບບ Distributed ເຊິ່ງກີກສ້າງຂຶ້ນໂດຍ Linus Torvalds. Git ມີປະໂຫຍດຫຼາຍຢ່າງດັ່ງນີ້:

- 1) ບັນທຶກປະຫວັດການຮັດວຽກ: ມັນຈະຮູ້ວ່າໃຜເປັນຄົນແກ້ໄຂ Code, ແກ້ໄຂແຖວໃດ ແລະ ແກ້ໄຂ ເມື່ອໃດ.
- 2) ຍ້ອນກັບໄປເວີຊັ້ນເກົ່າ: ຖ້າຂຽນ Code ແລ້ວ Program ເພື່ອບັນທຶກ (Bug) ຫຼາຍເກີນໄປ, ເຮົາ ສາມາດຍ້ອນກັບໄປຫາເວີຊັ້ນເກົ່າທີ່ເຕີຍໃຊ້ງານໄດ້ດີຢູ່ໄດ້.
- 3) ເຮັດວຽກຮ່ວມກັນເປັນທີມ: ນັກພັດທະນາຫຼາຍຄົນສາມາດຂຽນ Code ໃນໂປຣເຈັກດຽວກັນໄດ້ ໂດຍບໍ່ຕ້ອງກັງວົງວ່າຈະໄປທັບ Code ຂອງໜີ່.
- 4) ການແຍກສາຍພັດທະນາ: ເຮົາສາມາດແຍກກຳ (Branch) ອອກມາເພື່ອທິດລອງສ້າງ Feature ໃໝ່ງໂດຍບໍ່ໃຫ້ກະທົບກັບ Code ຫຼັກທີ່ໃຊ້ງານປຸ່ງ.



5. GitHub

ແມ່ນເຄື່ອງມີສໍາລັບນັກພັດທະນາ Software ໃຊ້ເພື່ອເກັບຮັກສາ ແລະ ແບ່ງປັນ Source Code ໂດຍ ໃຊ້ລະບົບ Git ໃນການຈັດການ. GitHub ໃຊ້ເພື່ອເປັນບ່ອນ Backup ຂໍ້ມູນ Code, ການເຮັດວຽກ ຮ່ວມກັບນັກພັດທະນາຫຼາຍຄົນ ແລະ ສ້າງ Portfolio ເພື່ອເກັບຜົນງານ.



6. Obsidian

ແມ່ນແຮ້ບົບພິເຄຊັ້ນສໍາລັບຈົດບັນທຶກ ແລະ ຈັດການຄວາມຮູ້ ແຕ່ Obsidian ບໍ່ໄດ້ເປັນພຽງແຮ້ບົບຈົດ ບັນທຶກເທົ່ານັ້ນ ແຕ່ຍັງສາມາດເປັນເຄື່ອງມີສໍາລັບນັກພັດທະນາ ເຊິ່ງສາມາດຈັດການ Code Snippets, ການເຮັດ Document Project ແລະ ຮອງຮັບ Plugins ໃນການຂຽນໂປຣແກຣມ.



7. Platform IO

ແມ່ນ Extension ທີ່ຢູ່ໃນ Visual Studio Code ເພື່ອໃຫ້ຂຽນ Code ແລະ ເຊື່ອມຕໍ່ກັບ ບອດ Microcontroller ໄດ້ສະດວກຂຶ້ນ.



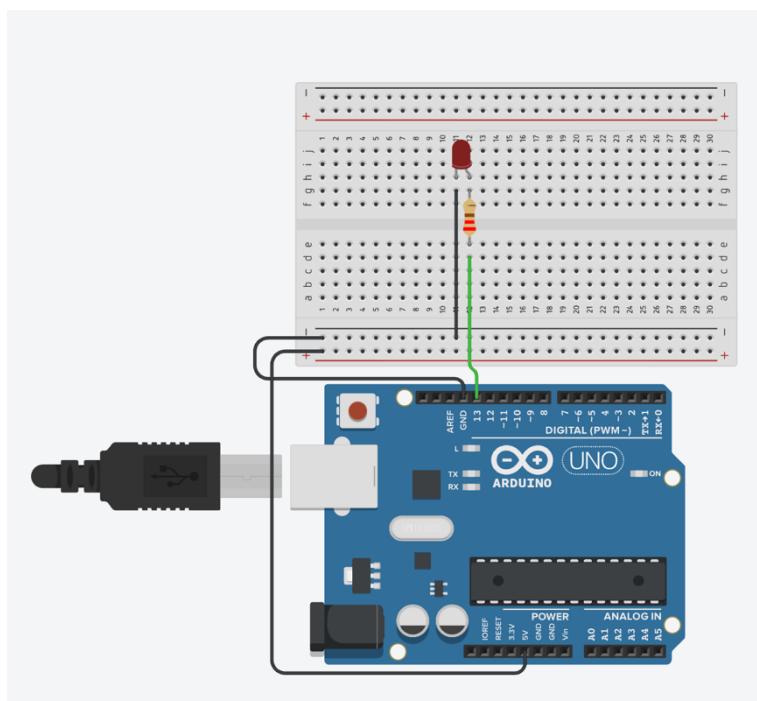
ပိုင်စီး 1: FirstLab Blink

I. ພິດນໍາ

II. ອຸປະກອນ

LED	1
Resistor 220ohm	1
Arduino R3	1
Breadboard	1

III. งานเขื้องตั่งวิจัย

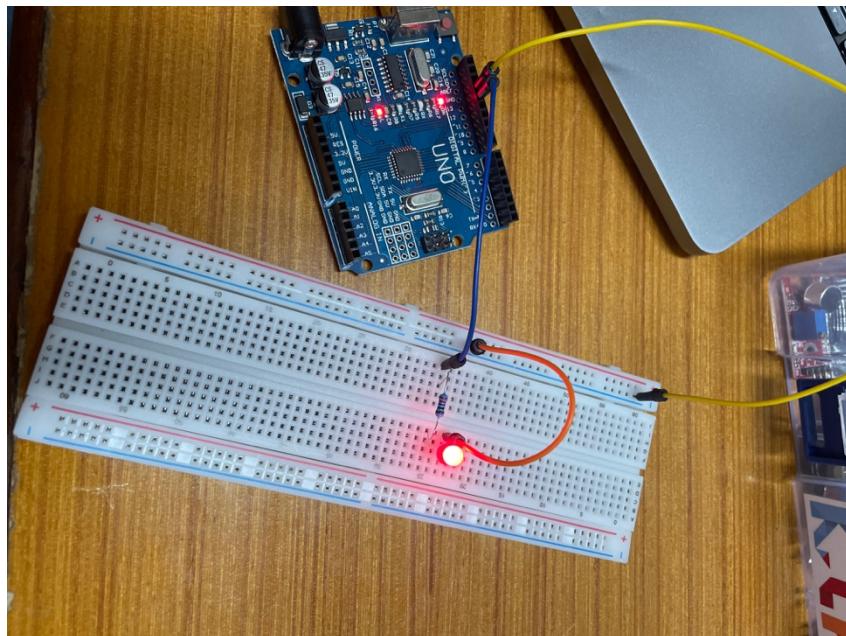


IV. Coding

```
void setup() {  
    // put your setup code here, to run once:  
  
    pinMode(LED_BUILTIN, OUTPUT); // PIN 13  
  
}  
  
void loop() {  
    // put your main code here, to run repeatedly:  
  
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(1000); // Wait for 1000 millisecond(s) = 1 second  
  
    digitalWrite(LED_BUILTIN, LOW);  
    delay(1000); // Wait for 1000 millisecond(s) = 1 second  
}
```

V. ការរៀបគ្មានវិវឌ្ឍន៍

- 1) ខ្ចាំពួនហំខីតិ៍ Arduino ការមិនិតិ៍ Pin 13 បែងខាងក្រោម OUTPUT.
- 2) ចុះថវិកទូទៅ: Arduino ស្តីពី 5V និង Pin 13 (HIGH), ឬផ្សេងៗពីរដាក់បានឡើង។
 ហើយ LED នៅលើក្នុងខ្សែភ្លើង។
- 3) ឬផ្សេងៗមិនិតិ៍ Arduino ឲ្យត្រួតពី 0V និង LOW, នៅលើ LED តុបតុប។
- 4) ការរៀបគ្មានមួយនេះនឹងរួចរាល់ខ្លួនខ្លួនទៅ 2 ឬ 3 ខ្លួន។



VI. QR Code

FirstLab Blink



ບົດທີ 2: Lab1 LED

I. ບິດນໍາ

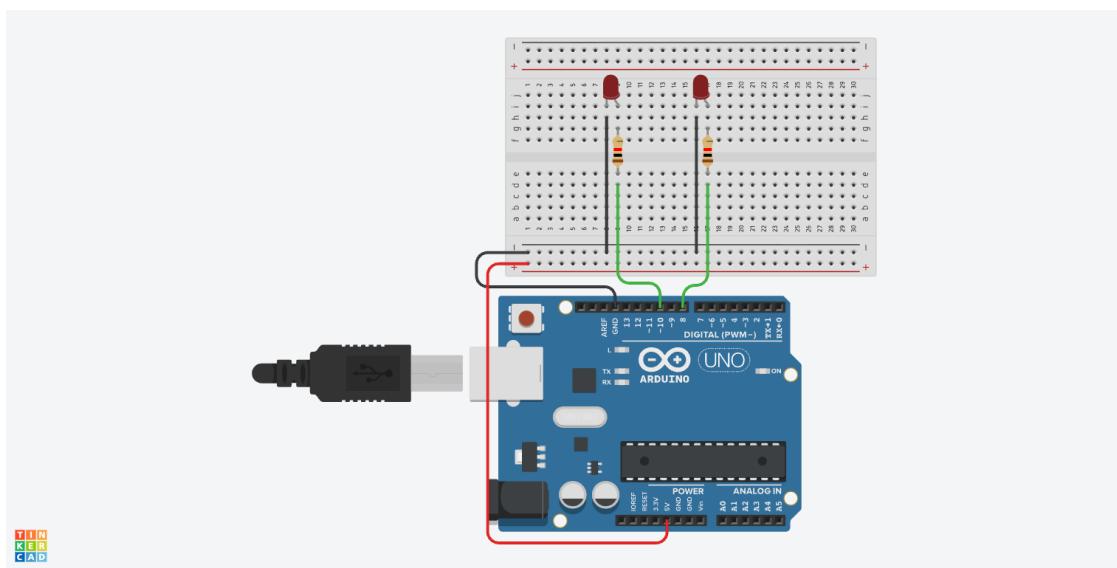
ໃນບົດທິດລອງທໍາອິດເຮົາຈະສຶກສາທີ່ວ່ອຂໍ້ງວຽກກັບ LED ເພື່ອສຶກສາ ແລະ ເຂົ້າໃຈວິທີການນຳໃຊ້ການເຮັດວຽກຂອງດອກໄຟ LED ເຊິ່ງມັນຈະປ່ຽນພະລັງງານໄຟຟ້າໃຫ້ເປັນແສງສະຫວ່າງ. ນອກຈາກນີ້, ເຮົາຢັ້ງໄດ້ຮຽນຮູ້ກ່ຽວກັບຂຶ້ວບວກ (Anode) ແລະ ຂຶ້ວລົບ (Cathode), ການນຳໃຊ້ຕົວຕ້ານທານ Resistor ເພື່ອຄວບຄຸມແສງສະຫວ່າງ ແລະ ການນຳເອົາ LED ໄປປະຍຸກນຳໃຊ້ເຊັ່ນ: ໄຟສັນຍານ ແລະ ອຸປະກອນສະແດງຜົນເປັນຕົ້ນ.

II. ອຸປະກອນ

LED	2
Resistor 220 ohm	2
Arduino	1
Breadboard	1

III. ការង្វិតអំពីរបាយការណ៍

- 1) ស្វែប LED រាប Resistor ទាំង 2 គីឡូ នៃ Breadboard តាម Resistor ដែលធ្វើឱមព័ត៌មាន នូវបន្ទុក ឬ ខ្លួន យាយខ្លួន LED.
 - 2) ចូលចុចត់ Resistor រាប LED1 ដោយខ្លួនបន្ទុកលេខ 8 នៃលាមពិភោគ Digital Pin 8 និងបណ្តុះបណ្តាល Arduino.
 - 3) ចូលចុចត់ Resistor រាប LED2 ដោយខ្លួនបន្ទុកលេខ 10 នៃលាមពិភោគ Digital Pin 10 និងបណ្តុះបណ្តាល Arduino.
 - 4) ខ្លួនខ្លួន LED ទាំង 2 តាមដៃនៃខ្លួនលិបិនី Breadboard ចូលចុចត់នូវលាមពិភោគ នៃខ្លួន ឬ លិបិនី ឬ លាមពិភោគ GND និងបណ្តុះបណ្តាល Arduino.



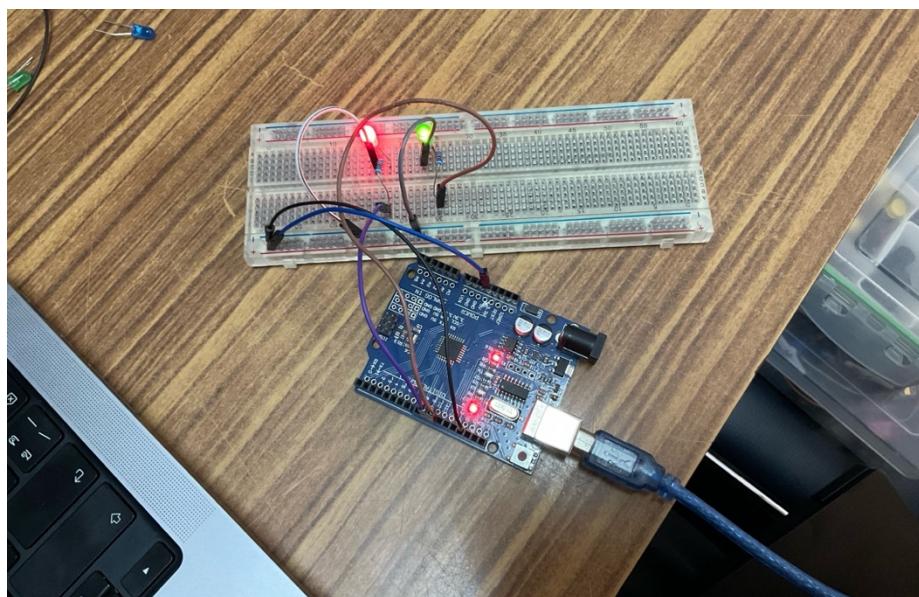
IV. Coding

```
int led1 = 10;  
int led2 = 8;  
void setup()  
{  
    pinMode(led1, OUTPUT);  
    pinMode(led2, OUTPUT);  
}  
  
void loop()  
{  
    digitalWrite(led1, HIGH);  
    digitalWrite(led2, HIGH);  
    delay(1000); // Wait for 1000 millisecond(s)  
    digitalWrite(led1, LOW);  
    digitalWrite(led2, LOW);  
    delay(1000); // Wait for 1000 millisecond(s)  
}
```

V. ການເຮັດວຽກຂອງວິຈາໂຄນ

ວິຈາໂຄນນີ້ຈະຮັດໃຫ້ໄຟ LED ຫ້າງສອງດອກ ຮຸ່ງພ້ອມກັນ 1 ວິນາທີ ແລະ ມອດພ້ອມກັນ 1 ວິນາທີ ສະລັບກັນໄປເລື້ອຍໆ. ການເຮັດວຽກຂອງມັນມີຂັ້ນຕອນຕໍ່ນີ້:

- 1) digitalWrite(led1, HIGH) ແລະ led2, HIGH ຈະສັ່ງໃຫ້ໄຟຫ້າງສອງດອກເປີດພ້ອມກັນ
- 2) delay(1000) ມັນຈະຄ້າງໄວ້ 1 ວິນາທີ.
- 3) digitalWrite(led1, LOW) ແລະ led2, LOW ມັນຈະສັ່ງໃຫ້ໄຟຫ້າງສອງດອກປິດພ້ອມກັນ
- 4) delay(1000) ຄ້າງໄວ້ 1 ວິນາທີ ກ່ອນຈະກັບໄປເລີ່ມຕົ້ນຂັ້ນຕອນທີ່ 1 ໃໜ່.



VI. QR Code

Lab1: LED



បុណ្យទី 3: Lab 2 Push Button

I. ពិណីតា

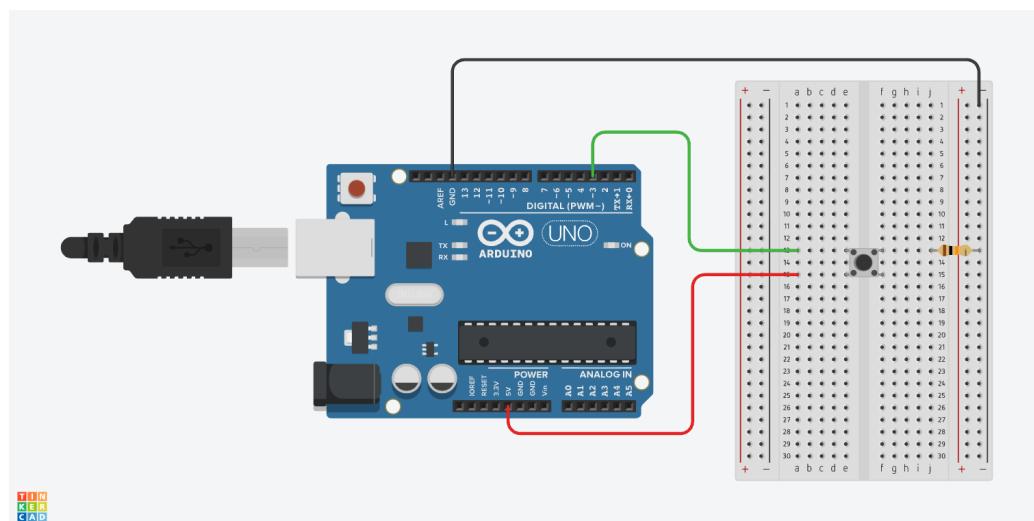
ໃນ Lab ទី 2 នេះវាគ្មានការងាររវាងការប្រើប្រាស់ Push Button ដើម្បីបង្កើតសម្រាប់គ្រប់គ្រងការងារ។ ក្នុងការងារនេះ យើងត្រូវប្រើប្រាស់ការងារខាងក្រោមដើម្បីធ្វើការងារ។

II. អ៊ូប្រភេទ

Button	1
Resistor 10kilo ohm	1
Arduino R3	1
Breadboard	1

III. ការងារដើម្បីបង្កើត

- ស្វែប Button លើក្នុង Breadboard.
- ស្វែប Resistor លើក្នុង Breadboard និង តែងតាំងខ្លួនលើបុរាណ ដើម្បីជួយចូលរួមការងារនេះ។
- ចូលរួមការងារនេះ ដើម្បីបង្កើតការងារ។
- ចូលរួមការងារនេះ ដើម្បីបង្កើតការងារ។
- ចូលរួមការងារនេះ ដើម្បីបង្កើតការងារ។



IV. Coding

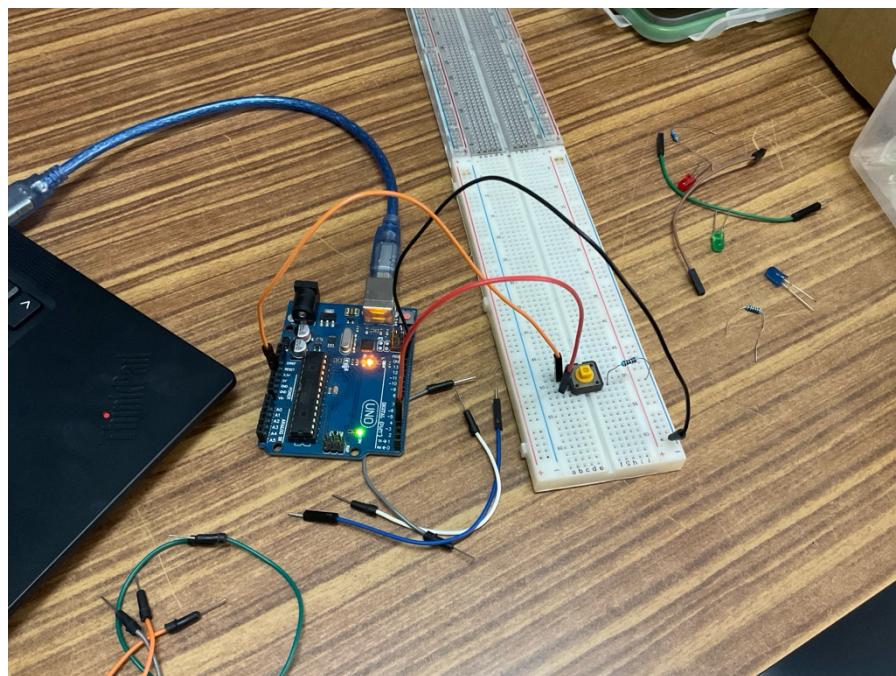
```
int buttonState = 0;

void setup()
{
    pinMode(3, INPUT);
    pinMode(LED_BUILTIN, OUTPUT);
    Serial.begin(9600);
}

void loop()
{
    buttonState = digitalRead(3);
    if (buttonState == HIGH) {
        Serial.println("Button HIGH");
        digitalWrite(LED_BUILTIN, HIGH);
    } else {
        Serial.println("Button LOW");
        digitalWrite(LED_BUILTIN, LOW);
    }
    delay(10);
}
```

V. ການເຮັດວຽກຂອງວິຈອນ

- 1) ເມື່ອເຮົາກິດປຸ່ມຄ້າງໄວ້ ມັນຈະຂຶ້ນ Button HIGH ໜີ້ໜ້າຈຳ Serial Monitor.
- 2) ເມື່ອເຮົາປ່ອຍປຸ່ມມັນຈະຂຶ້ນຄໍາວ່າ Button LOW ຕະຫຼອດເວລາ.



VI. QR Code

Lab2 Push Button



ပິດທີ 4: Lab3 RGB

I. ປິດນຳ

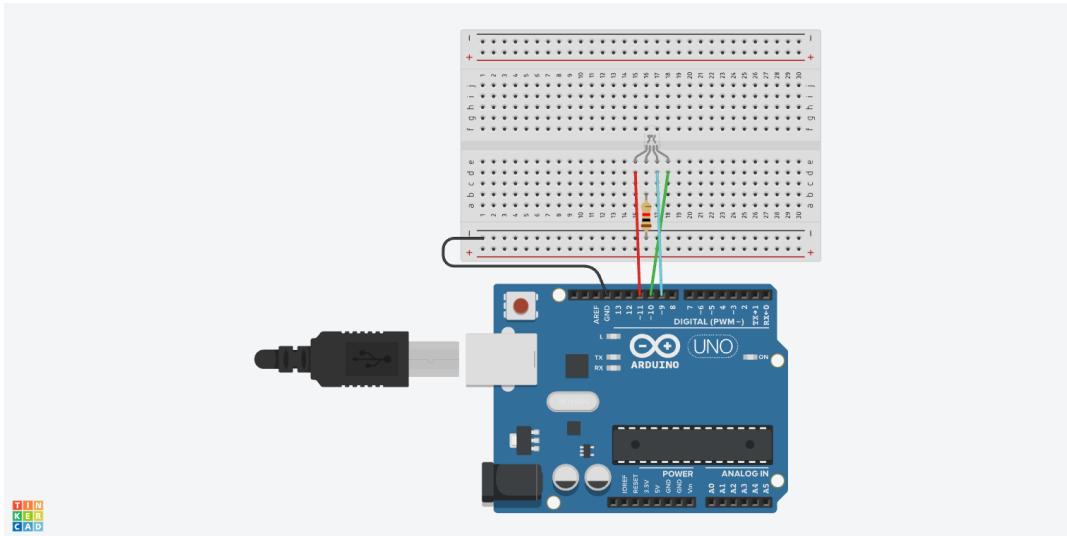
ໃນ Lab ທີ່ 3 ເຮົາຈະໄດ້ຮູມຮັການນຳໃຊ້ດອກໄຟ RGB ເຊິ່ງເປັນດອກໄຟ 3 ດວງ (ແດງ, ຂວວ, ພໍາ) ໄວ້ໃນດອກດຽວກັນ ເຊິ່ງມັນຈະການຄວບຄຸມຄວາມເຂັ້ມຂອງແສງສະຫວ່າງຂອງແຕ່ລະສີໃຫ້ແຕກຕ່າງກັນ, ເຮົາສາມາດປະສົມກັນອອກມາໄດ້ຫຼາຍທີ່ແຕກຕ່າງໄດ້.

II. ອຸປະກອນ

RGB	1
Resistor 1kilo ohm	1
Arduino R3	1
Breadboard	1
ຫຼື RGB Module	1

III. ການເຊື່ອມຕໍ່ວົງຈອນ

- 1) ສູບ RGB ລົງໃນ Breadboard
- 2) ເຊື່ອມຕໍ່ Resistor ກັບຂາຂຶ້ວລົບຂອງ RGB
- 3) ຂາສີແດງຂອງ RGB ເຊື່ອມຕໍ່ໄປຫາ digital pin 11 ຂອງບອດ Arduino.
- 4) ຂາສີຂວາງຂອງ RGB ເຊື່ອມຕໍ່ໄປຫາ digital pin 10 ຂອງບອດ Arduino.
- 5) ຂາສີຟ້າຂອງ RGB ເຊື່ອມຕໍ່ໄປຫາ digital pin 9 ຂອງບອດ Arduino.
- 6) ຂັ້ນຕອນສຸດທ້າຍແມ່ນຕໍ່ສາຍຈາກຂຶ້ວລົບຂອງ Breadboard ເຊື່ອມຕໍ່ໄປຫາ GND ຂອງ Arduino.



IV. Coding

```
const int ledPins[] = {11, 10, 9}; //RGB

void setup() {
    // put your setup code here, to run once:
    Serial.begin(9600);
    for (int pin : ledPins){
        pinMode(pin, OUTPUT);
    }

}

void loop() {
    // put your main code here, to run repeatedly:

    while (Serial.available() > 0) {
        // read input from serial monitor
        String input = Serial.readStringUntil('\n'); // Read until newline
        for (int pin : ledPins){
            digitalWrite(pin, LOW);
        }
        delay(10);

        if (input == "on" || input == "rgb" || input == "rbg" || input == "grb" || input == "gbr" || input == "brg" || input == "bgr" ) {
            // turn all LEDs on
            for (int pin : ledPins){
                digitalWrite(pin, HIGH);
            }
        }

        else if (input == "off") {
            // turn all LEDs off
            for (int pin : ledPins){
                digitalWrite(pin, LOW);
            }
        }

        else if (input == "r") {
            digitalWrite(ledPins[0], HIGH);
        }

        else if (input == "g") {
            digitalWrite(ledPins[1], HIGH);
        }

        else if (input == "b") {
```

```

        digitalWrite(ledPins[2], HIGH);
    }

    else if (input == "rg" or input == "gr") {
        digitalWrite(ledPins[0], HIGH);
        digitalWrite(ledPins[1], HIGH);
    }
    else if (input == "rb" or input == "br") {
        digitalWrite(ledPins[0], HIGH);
        digitalWrite(ledPins[2], HIGH);
    }
    else if (input == "gb" or input == "bg") {
        digitalWrite(ledPins[1], HIGH);
        digitalWrite(ledPins[2], HIGH);
    }
    else {
        allColors();
    }
}
}

void allColors(){
//WHITE
for(int pin : ledPins){
    digitalWrite(pin, HIGH);
}
delay(100);
for(int pin : ledPins){
    digitalWrite(pin, LOW);
}
delay(100);

//RED, GREEN, BLUE
for(int pin : ledPins){
    digitalWrite(pin, HIGH);
    delay(100);
    digitalWrite(pin, LOW);
    delay(100);
}

//RED + GREEN index 0 and index 1
for(int i = 0; i < 2; i++){
    digitalWrite(ledPins[i], HIGH);
}
delay(100);
for(int i = 0; i < 2; i++){
    digitalWrite(ledPins[i], LOW);
}
}

```

```

delay(100);

//RED BLUE index 0 and index 2
for(int i = 0; i < 3; i += 2){
    digitalWrite(ledPins[i], HIGH);
}
delay(100);
for(int i = 0; i < 3; i += 2){
    digitalWrite(ledPins[i], LOW);
}
delay(100);

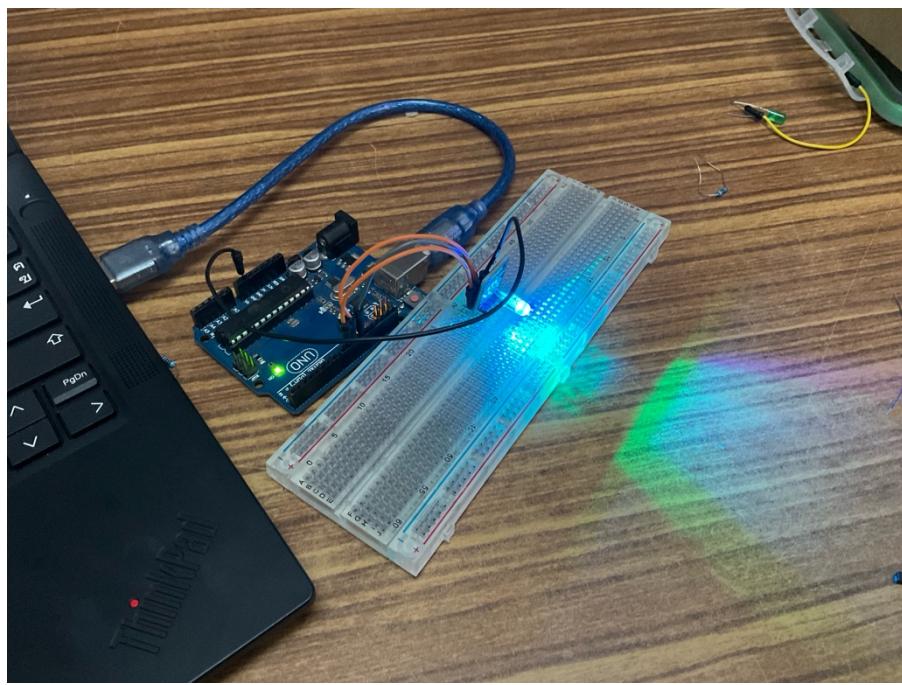
// GREEN + BLUE index 1 and index 2
for(int i = 1; i < 3; i++){
    digitalWrite(ledPins[i], HIGH);
}
delay(100);

for(int i = 1; i < 3; i++){
    digitalWrite(ledPins[i], LOW);
}
delay(100);
}

```

V. ການຮັດວຽກຂອງວິງຈອນ

- 1) ກໍານົດໃຫ້ Pin 9, 10, 11 ເປັນຂາອອກ Output.
- 2) ອັບຄໍາສັ່ງສັ່ງ (Loop) ເຊັ່ນ: rgb, r, g, b, rg, gr, rb, br, gb, bg.
- 3) ເມື່ອເຮົາພິມ rgb ມັນຈະສັ່ງຫັງ 3 ຂາເປັນ HIGH ພ້ອມກັນເທົ່າຈະໄດ້ໄຟສີປະສິມເປັນສີຂາວ.
- 4) ເມື່ອເຮົາພິມ r ໄຟຈະເປັນສີແດງ.
- 5) ເມື່ອເຮົາພິມ g ໄຟຈະເປັນສີຂຽວ.
- 6) ເມື່ອເຮົາພິມ b ໄຟຈະເປັນສີຟ້າ.
- 7) ເມື່ອເຮົາພິມ rg ຫຼື gr ໄຟຈະເປັນສີເຫຼືອງ.
- 8) ເມື່ອເຮົາພິມ rb ຫຼື br ໄຟຈະເປັນສີບິວ.
- 9) ເມື່ອເຮົາພິມ gb ຫຼື bg ໄຟຈະເປັນສີຟ້າອ່ອນ.



VI. QR Code

Lab 3 RGB



ပိုင်ချို့ 5: Lab4 P&A Buzzers

I. ປິດນໍາ

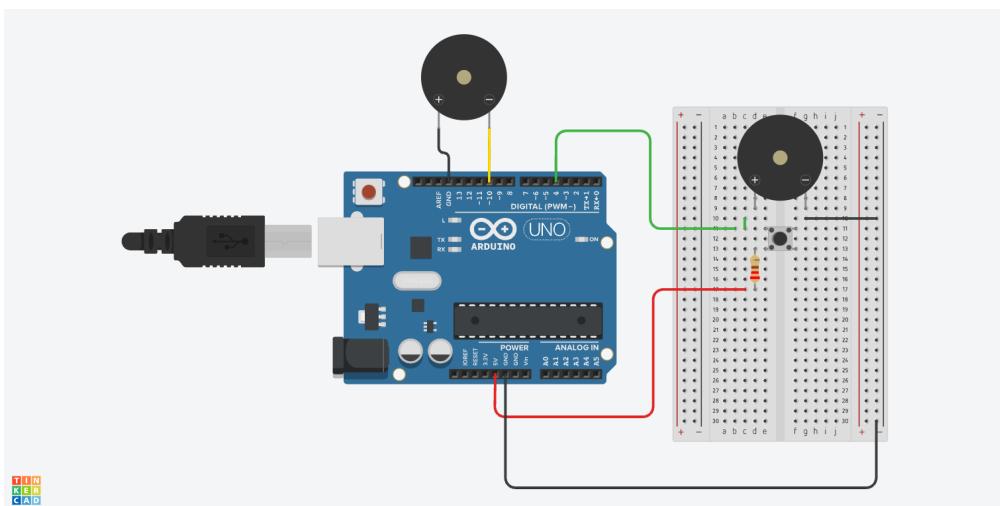
ໃນ Lab ທີ່ 4 ເຮົາຈະໄດ້ຮູມຮູກ່ຽວກັບ Buzzer ເຊິ່ງແມ່ນອຸປະກອນທີ່ປ່ຽນສັນຍານໄຟຟ້າໃຫ້ເປັນສຽງ, ມັນມີ 2 ຊະນິດທີ່ແຕກຕ່າງກັນຄື: Active Buzzer ແມ່ນ buzzer ທີ່ມີວິຈອນກຳເນີດຄວາມທີ່ຢູ່ພາຍໃນ ແລະ Passive Buzzer ແມ່ນບັດເຊີ່ທີ່ບໍ່ມີວິຈອນກຳເນີດສຽງໃນຕົວ. ໃນບົດທິດລອງນີ້ເຮົາຈະໄດ້ເຂົ້າໃຈການຄວບຄຸມຄວາມທີ່ຂອງສຽງ ແລະ ສັນຍານບອກສະຖານະດ້ວຍສຽງທີ່ແຕກຕ່າງກັນລະຫວ່າງ Active Buzzer ກັບ Passive Buzzer.

II. ອຸປະກອນ

Button	1
Resistor 220ohm	1
Passive Buzzer	1
Active Buzzer	1
Arduino	1
Breadboard	1

III. งานเขี๙มตํวิชาชອน

- 1) ស្រប Button လើងនៃ Breadboard.
 - 2) ស្រប Resistor ដើមព័ត៌មាននៃការបង្កើតនៃការងារ។ ត្រូវបង្កើតនៅលើតុលាការនៃការងារ។ ត្រូវបង្កើតនៅលើតុលាការនៃការងារ។
 - 3) ស្រប Active Buzzer លើងនៃ Breadboard.
 - 4) ខ្លួនបង្កើតនៃការងារ។ ត្រូវបង្កើតនៅលើតុលាការនៃការងារ។
 - 5) ខ្លួនបង្កើតនៃការងារ។ ត្រូវបង្កើតនៅលើតុលាការនៃការងារ។
 - 6) Passive Buzzer ខ្លួនបង្កើតនៃការងារ។ ត្រូវបង្កើតនៅលើតុលាការនៃការងារ។
 - 7) Passive Buzzer ខ្លួនបង្កើតនៃការងារ។ ត្រូវបង្កើតនៅលើតុលាការនៃការងារ។



IV. Coding

```
#define SPEAKER 10
#define BTN 4
#define FREQ 50

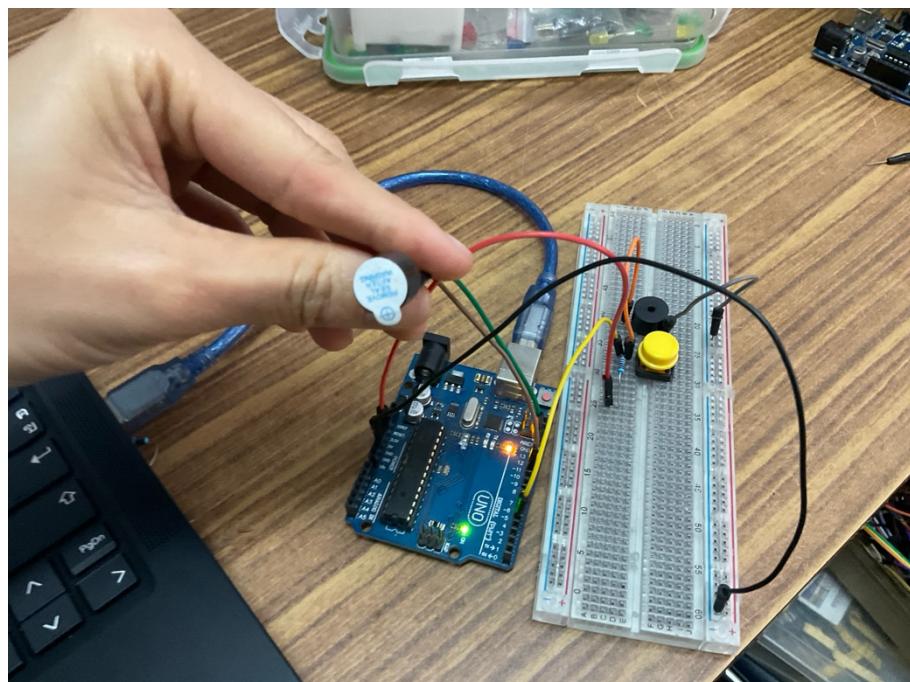
int startFreq = FREQ;
int endFreq = 4000;
int btnState = 0;

void setup()
{
    Serial.begin(9600);
    pinMode(SPEAKER, OUTPUT);
    pinMode(LED_BUILTIN, OUTPUT);
    pinMode(BTN, INPUT);
}

void loop()
{
    btnState = digitalRead(BTN);
    Serial.println(btnState);
    if(btnState == HIGH) {
        delay(500);
        digitalWrite(LED_BUILTIN, HIGH);
        for(startFreq; startFreq < endFreq; startFreq += 100){
            tone(SPEAKER, startFreq);
            delay(100);
        }
        digitalWrite(LED_BUILTIN, LOW);
        delay(100);
    }
    if (startFreq > endFreq) {
        noTone(SPEAKER);
        startFreq = FREQ;
    }
}
```

V. ການເຮັດວຽກຂອງວິຈຈອນ

- 1) ການກວດສອບ Input: ໂປຣແກຣມຈະກວດສອບສະຖານະຂອງບຸ່ມກິດ (btnState) ຢູ່ຕະຫຼອດເວລາ.
- 2) ເມື່ອບຸ່ມຖືກກິດ ສັນຍານໄຟ LED ຈະເຮັດວຽກ ແລະ ລະບົບຈະເລີ່ມໃຊ້ຄໍາສັ່ງ for loop ເພື່ອເພີ່ມຄວາມຖີ່ຂອງສຽງຈາກ 50Hz ໄປຫາ 4000Hz ໂດຍຈະເພີ່ມຂຶ້ນທີ່ອລະ 100Hz ໃນທຸກໆກາງ 0.1 ວິນາທີ (delay(100)).
- 3) ເມື່ອສຽງດັງກອດ 4000Hz ແລ້ວ, ໄຟຈະມອດລົງ.
- 4) ຄໍາສັ່ງ noTone(SPEAKER) ຈະຖືກເອີ້ນໃຊ້ເພື່ອປິດສຽງ.
- 5) ຫຼັງຈາກນັ້ນຄ່າຄວາມຖີ່ຈະຖືກ reset ກັບໄປທີ່ 50Hz ເພື່ອລຳທັການກິດບຸ່ມຄັ້ງຕໍ່ໄປ.



VI. QR Code

Lab 4 P&A Buzzer



ပိုင်စီး ၆: Lab5 Potentiometer

I. ບິດນໍາ

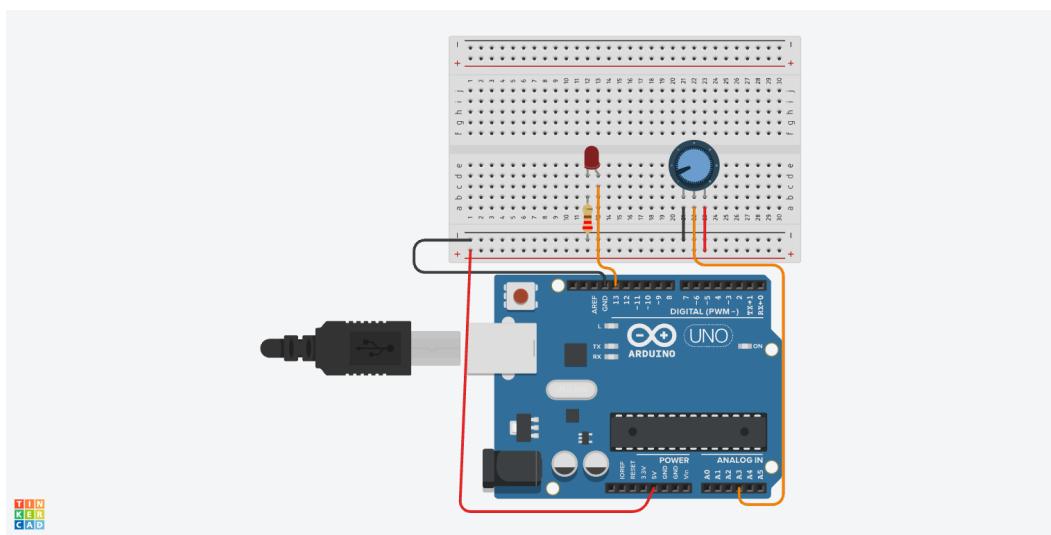
ใน Lab ที่ 4 เรขาจะได้รูปธุรกิจกับ Potentiometer ซึ่งมีตัวถ่วงที่สามารถหมุนเพื่อเปลี่ยนค่าความต้านทานพายในอิฐของเรามาใช้ในการบันทึกค่าความต้านทานของ microcontroller.

II. ອຸປະກອນ

LED	1
Resistor 220-ohm	1
Potentiometer	1
Arduino	1
Breadboard	1

III. งานเขี๊ยะมติวิชาจอน

- 1) ស្វែរ LED លើក្នុង Breadboard.
 - 2) ស្វែរ Resistor លើក្នុង Breadboard ដូចមានក្នុងក្រុមហ៊ុនខាងក្រោម (Cathode) ខាង LED និង ខ្លួនខាងក្រោម (Anode).
 - 3) ខ្លួនបន្ទាត់ (Anode) ខាង LED ត្រូវមានតំណាងជាបន្ទាត់ digital pin 13 ខាងក្រោម Arduino.
 - 4) ស្វែរ Potentiometer លើក្នុង Breadboard.
 - 5) ខ្លួនលើប (GND) ខាង Potentiometer ត្រូវមានតំណាងជាបន្ទាត់ Analog pin A3 ឲ្យបន្ទាត់ Arduino.
 - 6) ខាងក្រោមក្នុង Wiper ខាង Potentiometer ត្រូវមានតំណាងជាបន្ទាត់ VCC ឲ្យបន្ទាត់ Breadboard.
 - 7) ខ្លួនបន្ទាត់ VCC ខាង Potentiometer ត្រូវមានតំណាងជាបន្ទាត់ ខ្លួនបន្ទាត់ Breadboard.



IV. Coding

```
int analogValue = 0;
int ledValue = 0;
void setup()
{
    Serial.begin(9600);
    pinMode(13, OUTPUT);
}

void loop()
{
    analogValue = analogRead(A3);
    Serial.println(analogValue);

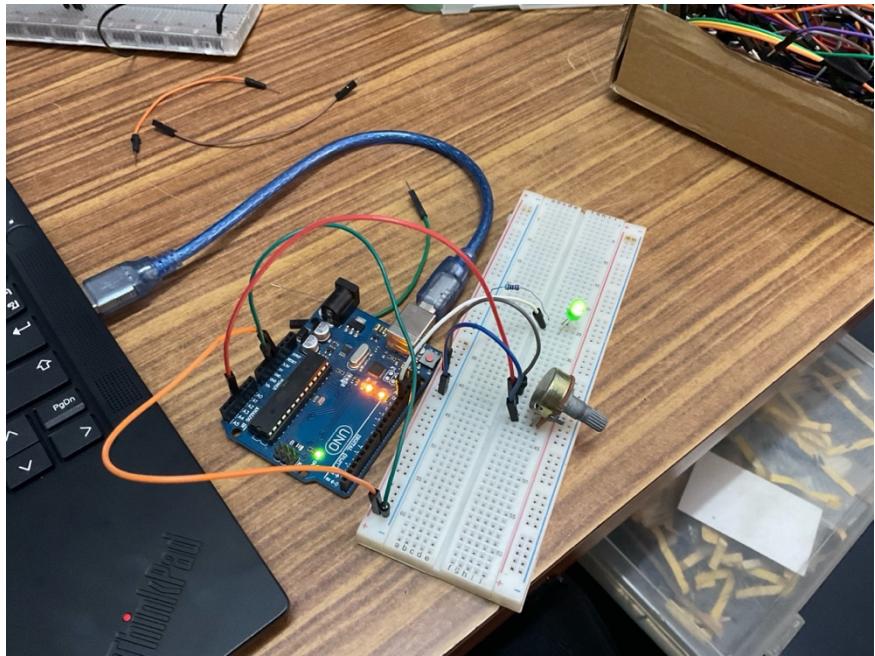
    ledValue = map(analogValue, 0, 1023, 0, 255);
    Serial.println(ledValue);

    analogWrite(13, ledValue);
    delay(10);
}
```

V. ការនេះវិភាគខ្លួនឱ្យចូល

ໃນວິຈາອນນີ້ Potentiometer ແມ່ນເພື່ອຄວບຄຸມຄວາມສະຫວັງຂອງດອກໄຟ LED ໂດຍຜ່ານບອດ Arduino ເຊິ່ງມັນເປັນການປ່ຽນສັນຍາ Analog (ຈາກການໝູນປຸ່ມ) ໃຫ້ເປັນສັນຍາ PWM ເພື່ອເຮັດໃຫ້ LED ແຈ້າ ທີ່ ມີວ.

- 1) Potentiometer จะอ่านค่าจาก analogValue = analogRead(A3); แล้ว Arduino จะอ่านค่าจากขา A3. ค่าที่ได้จะอยู่ระหว่าง 0-1023 (ขึ้นกับว่าเริ่มบันทุณี่ไปทางใด).
 - 2) ภาระแบบ Mapping: ledValue = map(analogValue, 0, 1023, 0, 255); เมื่อจากค่าที่อ่านได้แม่น 0-1023 แต่ภาระสั้นใช้ LED แล้ว (PWM) รับค่าได้พวง 0-255. คำสั่ง map จึงแสดงค่าที่ทຽบหักตากลับไว้โดยหักตากะในมัด.
 - 3) digitalWrite(13, ledValue); สั่งค่าที่แบบแล้วออกไปที่ขา 13. ถ้าค่าฐานะไฟจะแจ้ง, ถ้าค่าน้อยไฟจะมืด.



VI. QR Code

Lab5 Potentiometer



ບົດທີ 7: Lab6 Servo

I. ປິດນໍາ

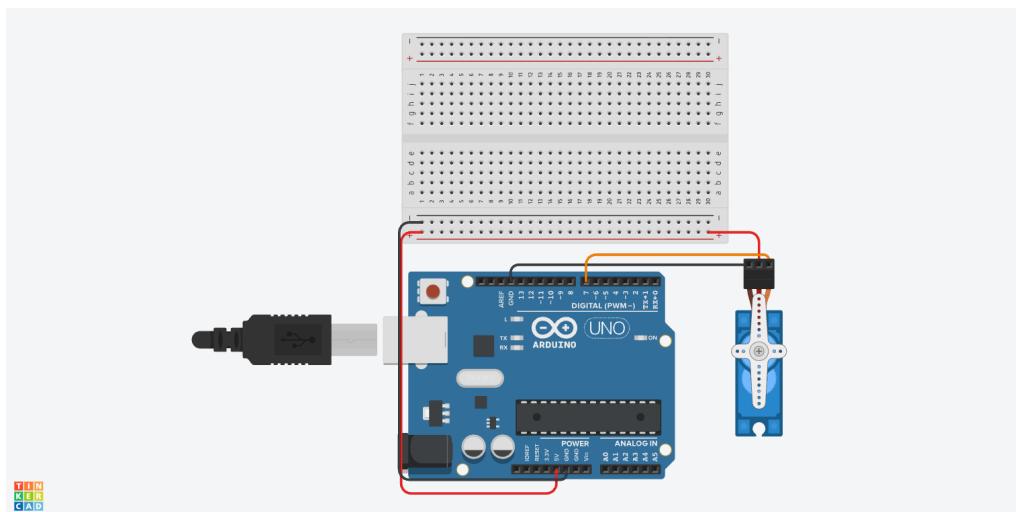
ໃນ Lab ທີ່ 6 ນີ້ເຮົາຈະໄດ້ຮູມຮັກງວກັບ Servo ເຊິ່ງເປັນອຸປະກອນຄວບຄຸມຕຳແໜ່ງ, ຄວາມໄວ ແລະ ຄວາມເລື່ງ. ໃນການທຶດລອງນີ້ເຮົາຈະໄດ້ຮູມຮັກຄວບຄຸມການເຄື່ອນທີ່ໃນມູນທີ່ແມ່ນອນ ແລະ ການກຳນົດອີງສາຂອງມຳເຕີ.

II. ອຸປະກອນ

Servo	1
Breadboard	1
Arduino	1

III. ການເຊື້ອມຕໍ່ວົງຈອນ

- 1) ឧប្បគលិបខទេសervo ដើមតែសាយសិកា ដោយ GND ខណ្ឌបន្ថយ Arduino.
 - 2) ឧប្បគលិបខទេសervo ដើមតែសាយសិកាបានចូលរួមនៅក្នុង Breadboard.
 - 3) ឧប្បគលិបខទេសervo ដើមតែសាយសិកា ដោយ digital pin 7 នៃបន្ថយ Arduino.
 - 4) ខ្លួនិយបន្ថយ Breadboard សាយសិកា ដើមតែ ដោយ GND ខណ្ឌបន្ថយ Arduino.
 - 5) ខ្លួនិយបន្ថយ Breadboard សាយសិកាបានចូលរួមនៅក្នុង 5V ខណ្ឌបន្ថយ Arduino.



IV. Coding

```
#include <Servo.h>

Servo myservo; //create servo object to
void setup()
{
    Serial.begin(9600);
    myservo.attach(7);
}

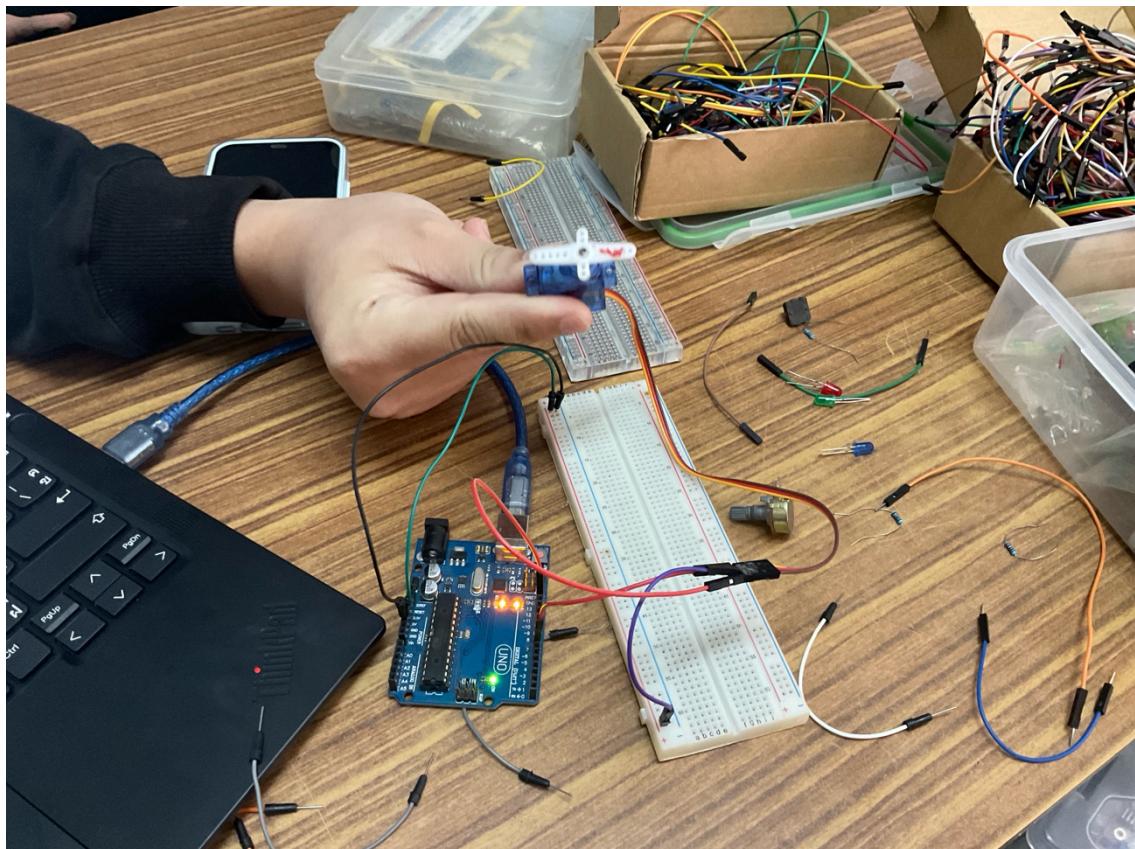
void loop()
{
    for(int i = 0; i <= 180; i++){
        myservo.write(i);

        Serial.println(i);
        delay(15);
    }
    delay(500);
    for (int i = 180; i >= 0; i--){
        myservo.write(i);

        Serial.println(i);
        delay(15);
    }
    delay(500);
}
```

V. ການເຮັດວຽກຂອງວົງຈອນ

- 1) เรียกใช้ library `#include <Servo.h>` เพื่อช่วยในการถูกติดต่อสัญญาณที่สั่งงาน Servo.
 - 2) กำหนดขา 7 เป็นขาสั่งสัญญาณผ่านคำสั่ง `myservo.attach(7)`.
 - 3) งานขุมนับไป 0-180 องศา จะใช้ `loop` for ด้วยเพิ่มค่าต่อปีกิ : เช่น 1 องศา. ในแต่ละ ขั้นตอน, มีการซัพพอร์ตตามค่า i และ ล็อก 15ms เพื่อให้มีเวลาเพียงพอที่ได้รับ.
 - 4) เมื่อขุมนับไป 180 องศาแล้ว, มันจะปุ่ดล็อก 500ms.
 - 5) งานขุมนับกลับ 180-0 องศา โปรดแก้ไขตามมาตรฐานค่า i ล่วงๆ เช่น 1 องศา จึงกับมารอดจุดเลื่อนต้น (0 องศา).
 - 6) เมื่อขุมนับกลับ 0 องศาแล้ว, มันจะปุ่ดล็อก 0.5s ก่อนเลื่อนรอบใหม่.



VI. QR Code

Lab 6 Servo



ບົດທີ່ 8: Lab6 Servo + Potentiometer (ຕໍ່)

I. ປິດນໍາ

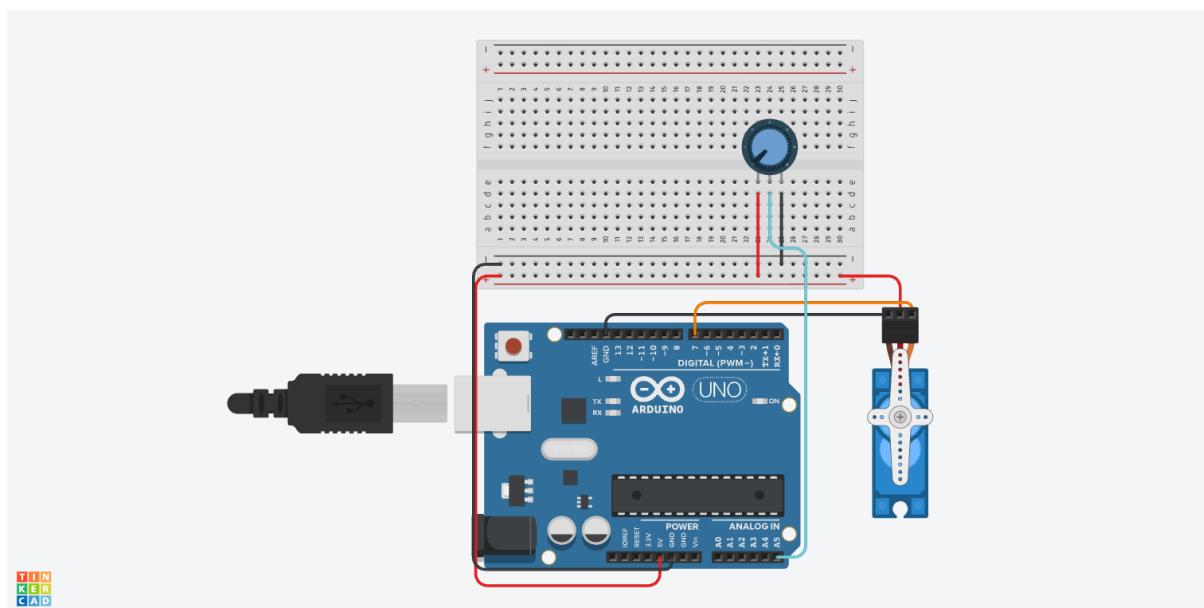
ใน Lab ที่ 6 นี้จะได้รูปแบบในการใช้งานเซอร์โวมอเตอร์ทั่วไป Servo กับ Potentiometer โดยมันจะควบคุมการป้อนข้อมูลด้วยภาษา Python ผ่าน Port COM ของคอมพิวเตอร์ เพื่อวัดมุมติดตามของเซอร์โว Servo.

II. ອຸປະກອນ

Potentiometer	1
Servo	1
Arduino	1
Breadboard	1

III. งานเขี๊ยวต์วิจัย

- 1) ស្រួល Potentiometer តិចនៅ Breadboard.
 - 2) ខាបវកខង់ Potentiometer សាយសិរីណែងផ្ទើមពាំន័យបន្ទាន់ខ្លួន Breadboard.
 - 3) ខាតាការការ Wiper នៃ Potentiometer ផ្ទើមពាំន័យ analog A5 នៃ Arduino.
 - 4) ខាលិបខង់ Potentiometer សាយសិរីតាំផ្ទើមពាំន័យខ្លួន Breadboard.
 - 5) ខាផីតិចខ្លួន Servo ផ្ទើមពាំន័យសិរីតាំដែលត្រូវបន្ទាន់ខ្លួន GND នៃ Arduino.
 - 6) ខាផីតិចខ្លួន Servo ផ្ទើមពាំន័យសិរីណែងដែលត្រូវបន្ទាន់ខ្លួន Breadboard.
 - 7) ខាងក្រោម Control នៃ Servo ផ្ទើមពាំន័យ digital pin 7 នៃ Arduino.
 - 8) ខ្លួនខ្លួន Breadboard សាយសិរីតាំផ្ទើមពាំន័យ GND នៃ Arduino.
 - 9) ខ្លួនខ្លួន Breadboard សាយសិរីណែងផ្ទើមពាំន័យ 5V នៃ Arduino.



IV. Coding

```
#include <Servo.h>

Servo myservo; //create servo object to control a servo

int potpin = A5;
int val;

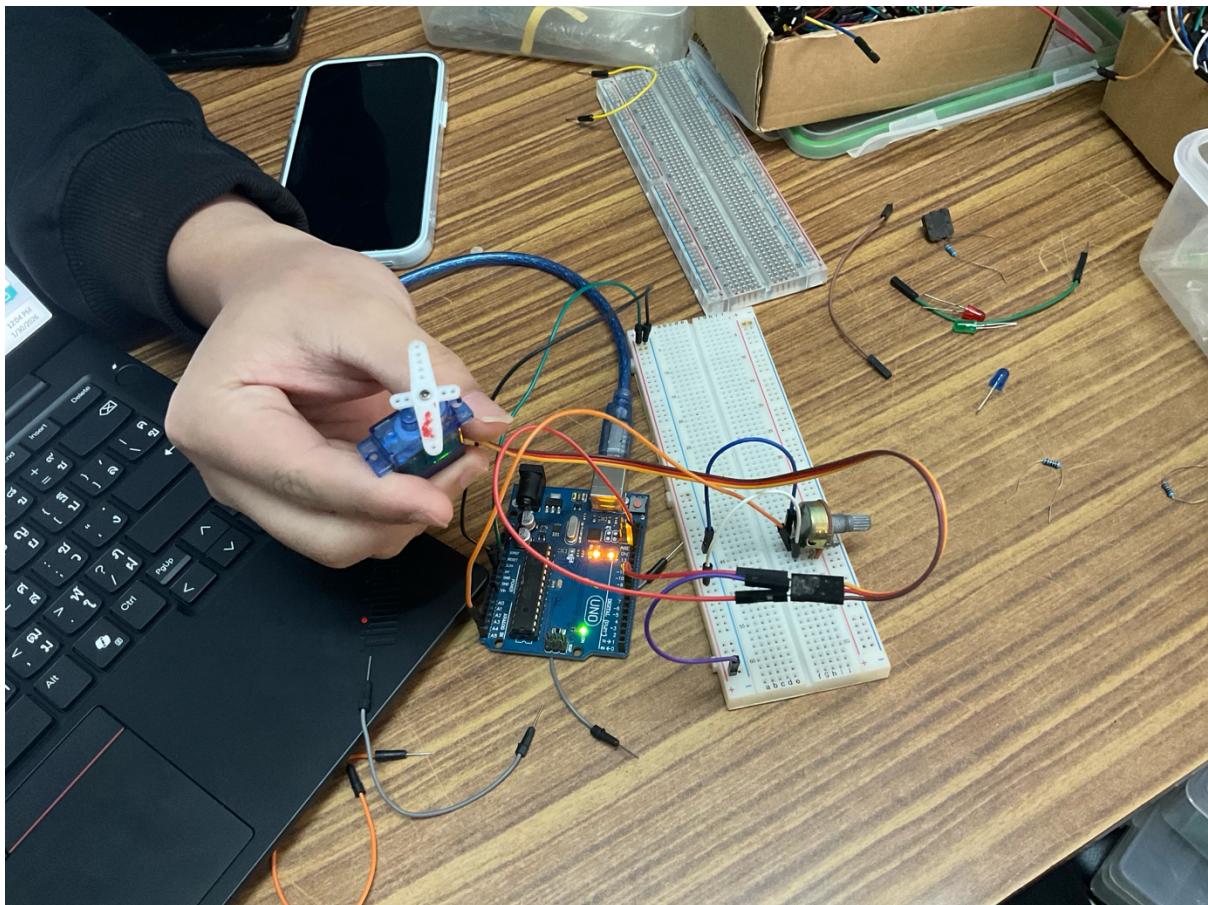
void setup()
{
    Serial.begin(9600);
    myservo.attach(7);
}

void loop()
{
    val = analogRead(potpin);
    Serial.print("pot = ");
    Serial.print(val);
    Serial.print(", servo = ");
    val = map(val, 0, 1023, 0, 180);

    Serial.println(val);
    myservo.write(val);
    delay(15);
}
```

V. ការងាររបស់កុម្ភយោង

- 1) មានតម្លៃទូទៅថា Reading ត្រួតពិនិត្យដោយលើកសំណង់ analogRead(potpin) ដើម្បីទូទៅតម្លៃទូទៅ។
- 2) ការរោចរាប់ថា Mapping: នឹងទូទៅថា Servo Motor ត្រូវតម្លៃណាមួយដែលត្រូវតម្លៃទូទៅ 0 ទៅ 180 ឯងសារ។ រាប់ថាទីត្រូវតម្លៃណាមួយដែលត្រូវតម្លៃទូទៅ 0-1023. តើតើអ្វីដឹងទូទៅថា: val = map(val, 0, 1023, 0, 180);
- 3) ឱ្យកុម្ភយោង myservo.write(val) ដើម្បីបញ្ចប់ការងារនៃកុម្ភយោង។
- 4) មិនត្រូវបានបញ្ជូនការងារ 15ms (delay(15)) ដើម្បីបញ្ចប់ការងារ។
- 5) មិនត្រូវបានបញ្ជូនការងារ 15ms (delay(15)) ដើម្បីបញ្ចប់ការងារ។



VI. QR Code

Lab6 Servo + Potentiometer



ပိုင်ဆို ၁၁: Lab7 Stepper

I. ປິດນໍາ

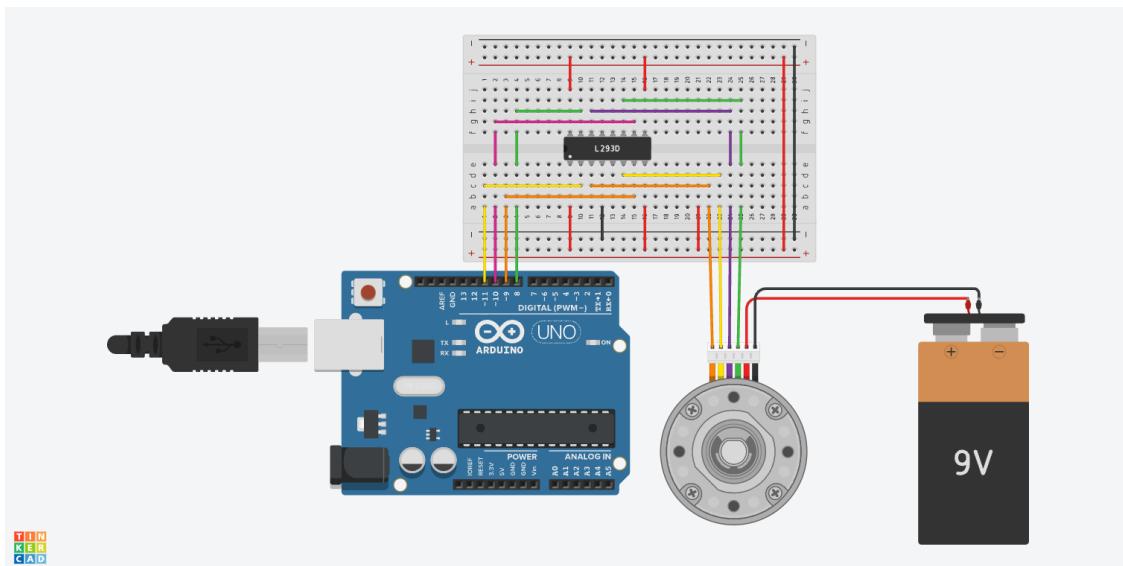
ໃນ Lab ທີ່ 7 ເຮົາຈະໄດ້ຮູບຮັກງວກັບ Stepper ເຊິ່ງແມ່ນມີເຕີທີ່ປ່ຽນສັນຍາໃຫ້ເປັນການເຄື່ອນທີ່
ທາງກົນຈັກແບບເປັນຈັງຫວະ. ໃນການທຶດລອງນີ້ເຮົາຈະໄດ້ເຂົ້າໃຈການຄວບຄຸມຕຳແໜ່ງ ແລະ ຄວາມໄວ
ຂອງ Stepper.

II. ອຸປະກອນ

Stepper Motor	1
Motor Driver ULN2003	1
Arduino	1
Breadboard	1

III. ການເຊື້ອມຕໍ່ອປະກອນ

- 1) ເອີ້າ Stepper Motor ສຽບໃສ່ຊ່ອງ pin ສີຂາວທັງ 5 ຊ່ອງໃສ່ໃນ ULN2003.
 - 2) ຫຼັງຈາກມິ້ນສຽບສາຍຈາກ IN1, IN2, IN3, IN4 ໄປສຽບໃສ່ຊ່ອງ digital pin 8, 9, 10, 11 ຂອງ Arduino.
 - 3) ສຽບຂໍ້ວບກະຂອງ ULN2003 ໄປຫາ 5V ຂອງArduino.
 - 4) ສຽບຂໍ້ວລືບຂອງ ULN2003 ໄປຫາ GND ຂອງArduino.



IV. Coding

```
#define STEPROTATION 2048

#include "Stepper.h"

Stepper Steppermotor(STEPROTATION , 8, 10, 9, 11); ///(1)

void setup() { ///(2)
}

void loop() {
    Steppermotor.setSpeed(5);
    Steppermotor.step(STEPROTATION );
    delay(500);

    Steppermotor.setSpeed(1);
    Steppermotor.step(-STEPROTATION /4);
    delay(500);
}
```

V. ការរៀបគ្រែកម្រិតទូទៅ

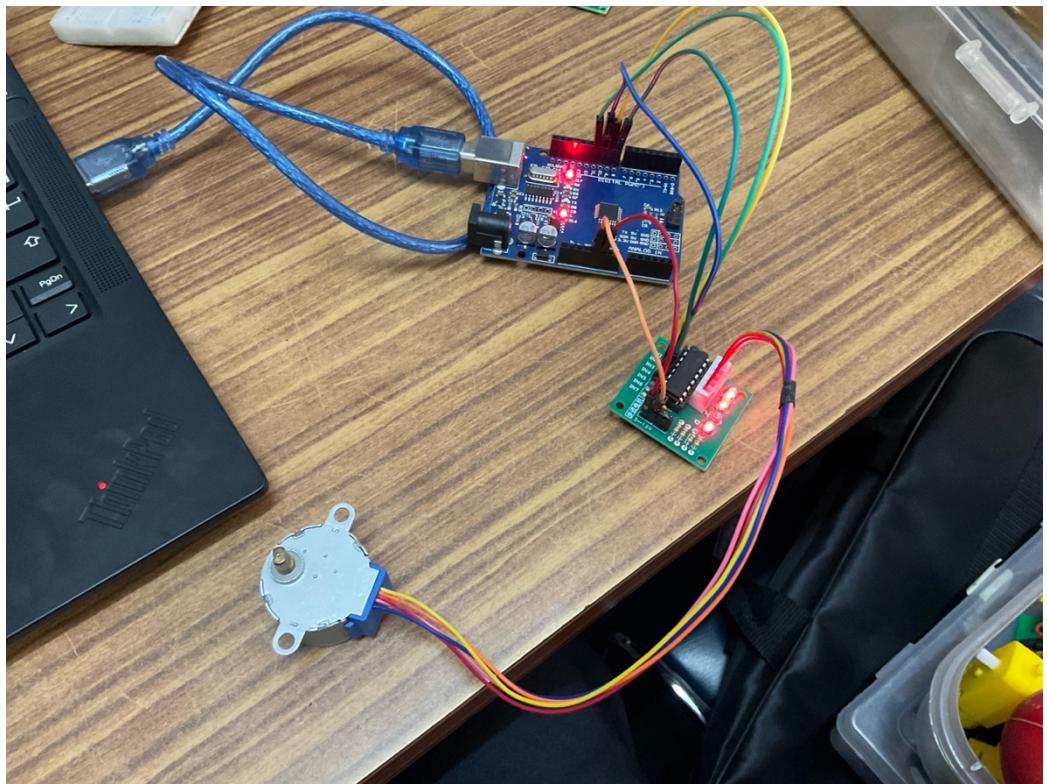
ការរៀបគ្រែកម្រិតទូទៅនេះមានចំណាំខាងក្រោម:

1) ខ្លួនទី 1:

- setSpeed(5): ព័ត៌មានថាអង្គភាពរៀបចំ 5 រ៉ូបតំបន់។
- Step(STEPROTATION): សំឡើង motor មូលដ្ឋាន 2048 រ៉ាវ ដើម្បីរៀបចំ 1 រ៉ូប។
- Delay(500): រៀបចំពេលវេលាដែលការរៀបគ្រែកត្រូវបានរាយការណ៍ 0.5 វិនាទិ។

2) ខ្លួនទី 2:

- setSpeed(1): ព័ត៌មានថាអង្គភាពរៀបចំ 1 រ៉ូបតំបន់។
- Step(-STEPROTATION / 4): សំឡើង motor មូលដ្ឋាន 2048/4 = 512 រ៉ាវ ដើម្បីរៀបចំ 90 ីរីសាត្រ ឬ 1/4 ខែករៀប។
- Delay(500): រៀបចំពេលវេលាដែលការរៀបគ្រែកត្រូវបានរាយការណ៍ 0.5 វិនាទិ។



VI. QR Code

Lab 7 Stepper



ບົດທີ່ 10: Lab8 LCD

I. ປິດນໍາ

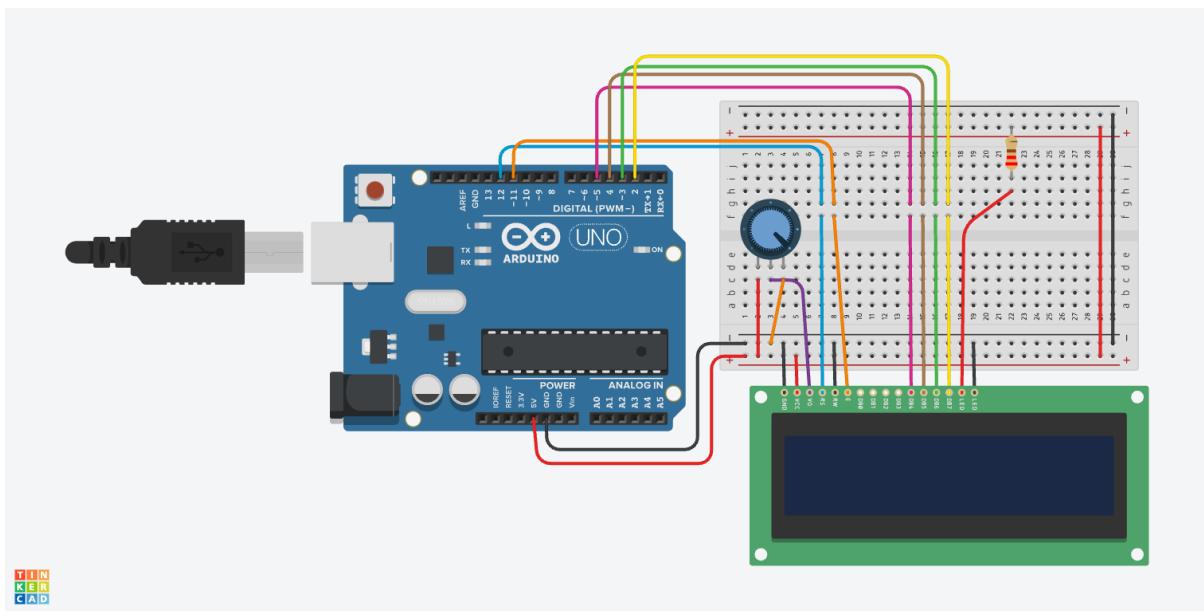
ໃນ Lab ທີ 8 ເຮົາຈະໄດ້ຮູ່ກ່ຽວກັບ ຈຳ LCD ເຊິ່ງແມ່ນອຸປະກອນສະແດງຜົນເປັນຕົວອັກສອນ ຫຼື ຮູບພາບທີ່ໃຊ້ກັບບອດ Microcontroller. ໃນການທຶດລອງນີ້ເຮົາຈະໄດ້ຮູ່ໃນການເຊື່ອມຕໍ່ຂາ ສັນຍານຕ່າງໆເຊັ່ນ: RS, EN ແລະ ຂາໜໍ້ມູນ Do-D7. ນອກຈາກນີ້, ຍັງໄດ້ປະຢຸກນຳໃຊ້ໃນການສະແດງ ຜົນເຊັ່ນເຊີ້ ແລະ ສ້າງຕົວອັກສອນ ຫຼື ໂຕເລກຕ່າງໆພ້ອໃຫ້ສະແດງຂຶ້ນທີ່ຈະ.

II. ອຸປະກອນ

Potentiometer	1
LCD	1
Resistor 220-ohm	1
Arduino	1
Breadboard	1

III. งานเข้มต่อปะกอน

- 1) ស្រួល Potentiometer តិចនៅ Breadboard.
 - 2) ខាងក្រោមនេះ potentiometer ត្រូវមកតាមរបៀបខាងក្រោមនេះ Breadboard សំរាប់ខ្លួនខាងក្រោមនេះ ត្រូវមកដើម្បី ខ្លួនបង្ហាញនៅ Breadboard.
 - 3) LCD Pin1 (VSS) និង Pin 16 តែងតាំងនៅ GND នៃ Breadboard.
 - 4) LCD Pin2 (VDD) តែងតាំងនៅខ្លួនបង្ហាញ នៃ Breadboard.
 - 5) LCD Pin3 (VO) ត្រូវមកតែងតាំងនៅ ខាងក្រោម Wiper នៃ Potentiometer.
 - 6) LCD Pin4 (RS) តែងតាំងនៅ digital pin 12 នៃ Arduino.
 - 7) LCD Pin5 (RW) តែងតាំងនៅ GND នៃ Breadboard.
 - 8) LCD Pin6 (E) តែងតាំងនៅ digital pin 11 នៃ Arduino.
 - 9) LCD Pin 11 (D4) តែងតាំងនៅ digital pin 5.
 - 10) LCD Pin 12 (D5) តែងតាំងនៅ digital pin 4.
 - 11) LCD Pin 13 (D6) តែងតាំងនៅ digital pin 3.
 - 12) LCD Pin 14 (D7) តែងតាំងនៅ digital pin 2.
 - 13) LCD Pin 15 (LED+) តែងតាំងនៅ Resistor 220-ohm សំរាប់ខ្លួនខាងក្រោមនេះ resistor តែងតាំងនៅខ្លួនបង្ហាញនៃ Breadboard.
 - 14) ស្នើសុំការបង្កើតខ្លួនបង្ហាញនៅ Breadboard ដោយ 5V នៃ Arduino សំរាប់ខ្លួនខាងក្រោមនេះ ត្រូវបង្ហាញនៅ GND.



IV. Coding

```
// C++ code
//
/*
LiquidCrystal Library - Hello World
```

Demonstrates the use of a 16x2 LCD display.
The LiquidCrystal library works with all LCD displays that are compatible with the Hitachi HD44780 driver. There are many of them out there, and you can usually tell them by the 16-pin interface.

This sketch prints "Hello World!" to the LCD and shows the time.

The circuit:

- * LCD RS pin to digital pin 12
- * LCD Enable pin to digital pin 11
- * LCD D4 pin to digital pin 5
- * LCD D5 pin to digital pin 4
- * LCD D6 pin to digital pin 3
- * LCD D7 pin to digital pin 2
- * LCD R/W pin to ground
- * LCD VSS pin to ground
- * LCD VCC pin to 5V
- * 10K resistor:
 - * ends to +5V and ground
 - * wiper to LCD VO pin (pin 3)

Library originally added 18 Apr 2008 by David A. Mellis
library modified 5 Jul 2009 by Limor Fried (<http://www.ladyada.net>)
example added 9 Jul 2009 by Tom Igoe
modified 22 Nov 2010 by Tom Igoe

This example code is in the public domain.

```
http://www.arduino.cc/en/Tutorial/LiquidCrystal
*/
#include <LiquidCrystal.h>

int seconds = 0;

LiquidCrystal lcd_1(12, 11, 5, 4, 3, 2);

void setup()
{
  lcd_1.begin(16, 2); // Set up the number of columns and rows on the LCD.

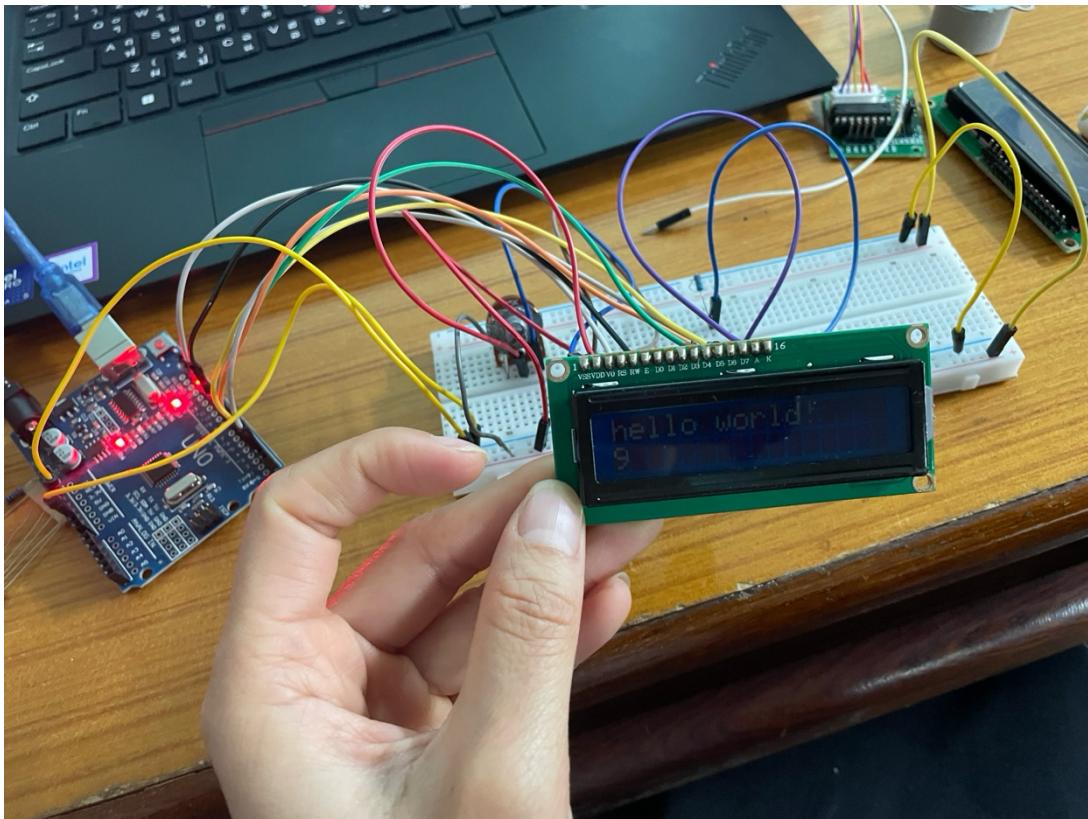
  // Print a message to the LCD.
  lcd_1.print("hello world!");
}

void loop()
{
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting
  // begins with 0):
  lcd_1.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd_1.print(seconds);
  delay(1000); // Wait for 1000 millisecond(s)
  seconds += 1;
}
```

V. ການຮັດວຽກຂອງວິງຈອນ

ເມື່ອເຮົາເລີ່ມກິດ Simulation ມັນຈະເລີ່ມຮັດວຽກດັ່ງນີ້:

- 1) ເລີ່ມຫຼຸນ Potentiometer ເພື່ອເປີດຕົວອັກສອນໃນຈຳ.
- 2) ຫັ້ນຈຳຈະຂຶ້ນແຖວຄໍາວ່າ Hello World.
- 3) ຫັ້ນແຖວທາງລຸ່ມຈະເລີ່ມນັບເລກ 0, 1, 2, 3.....ໄປເລື້ອຍໆ.



VI. QR Code

Lab 8 LCD



ບົດທີ່ 11: Project (18/12/2025)

I. ບິດນໍາ

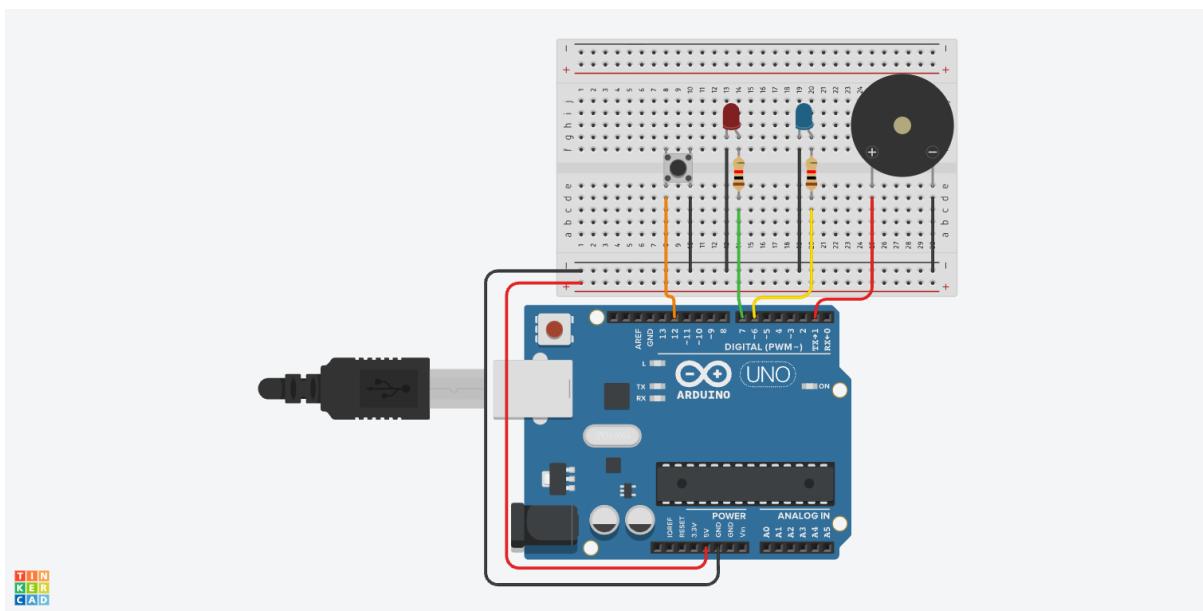
ໃນບົດທິດລອງນີ້ເປັນໂປຣເຈັກ Mid-term ເຊິ່ງໃນຫົວຂໍທິດລອງແມ່ນກ່ຽວກັບ ການເປີດ-ປິດໄຟຟ້າ ສະຫຼັບໂດຍການກົດປຸ່ມສະວິດ ພ້ອມມີສຽງ Buzzer ອອກມາ.

II. ອຸປະກອນ

LED	2
Resistor 1kilo-ohm	2
Button	1
Buzzer	1
Arduino	1
Breadboard	1

III. งานเขี๊ยะมตໍ່ອປະກອນ

- 1) ស្វែប Button តុលាកិច្ចកម្មនៃ Breadboard ឡើងតំខាងក្រោមថា ត្រូវបង្ហាញលើ digital pin 12 នៃ Arduino ដែលត្រូវបង្ហាញលើ GND នៃ Breadboard.
 - 2) ស្វែប LED តុលាកិច្ចកម្មនៃ Breadboard ឡើងតំខាងក្រោមថា ត្រូវបង្ហាញលើ digital pin 6 ឬ 7 នៃ Arduino ដែលត្រូវបង្ហាញលើ Anode នៃ LED និងត្រូវបង្ហាញលើ digital pin 5 ឬ 6 នៃ Arduino ដែលត្រូវបង្ហាញលើ Cathode នៃ LED.
 - 3) ស្វែប Buzzer តុលាកិច្ចកម្មនៃ Breadboard ខាបកវិញថា ត្រូវបង្ហាញលើ digital pin 4 នៃ Arduino ដែលត្រូវបង្ហាញលើ GND នៃ Breadboard.
 - 4) ត្រូវបង្ហាញលើគ្រប់គ្រងបន្ថែមនៃ Breadboard ដែលត្រូវបង្ហាញលើ 5V នៃ Arduino ដែលត្រូវបង្ហាញលើ GND នៃ Arduino.



IV. Coding

```
const int buttonPin = 12;  
const int buzzerPin = 4;
```

```

const int blueLED = 6;
const int greenLED = 7;

bool state = false;
bool lastButton = HIGH;

void setup()
{
    pinMode(buttonPin, INPUT_PULLUP);
    pinMode(buzzerPin, OUTPUT);
    pinMode(blueLED, OUTPUT);
    pinMode(redLED, OUTPUT);

    digitalWrite(redLED, HIGH);
}

void loop()
{
    bool currentButton = digitalRead(buttonPin);

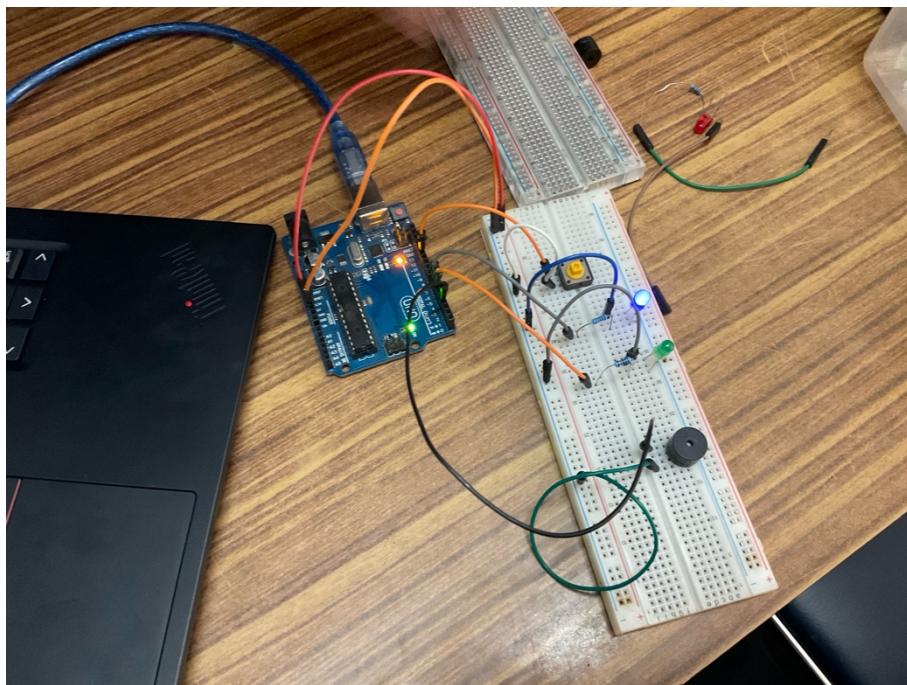
    if (lastButton == HIGH && currentButton == LOW) {
        state = !state;

        if (state) {
            digitalWrite(buzzerPin, HIGH);
            digitalWrite(blueLED, HIGH);
            digitalWrite(redLED, LOW);
        } else {
            digitalWrite(buzzerPin, LOW);
            digitalWrite(blueLED, LOW);
            digitalWrite(redLED, HIGH);
        }
        delay(200);
    }
    lastButton = currentButton;
}

```

V. ການເຮັດວຽກຂອງວິຈອນ

- 1) ເມື່ອເຮົາເລີ່ມຕົ້ນ Simulation ດອກໄຟສີຟ້າຈະຮູ່ຄ້າງໄວ້.
- 2) ເມື່ອເຮົາກິດປຸ່ມ ລະບົບຈະສະຫຼັບການເຮັດທັນທີໂດຍໄຟສີຟ້າຈະມອດ ສ່ວນໄຟສີຂຽວຮູ່ຈຶ່ນ ແລະ ມີສຽງ Buzzer ດັງຈຶ່ນມາ.
- 3) ເມື່ອເຮົາກິດປຸ່ມອີກຄັ້ງມັນຈະສະຫຼັບກັບສະພາບເດີມ ແຊ່ງໄຟສີຟ້າຈະຮູ່ ສ່ວນ ໄຟສີຂຽວ ກັບ ສຽງ ພຽບ.



VI. QR Code

Project 1



ပိုင်ခိုင်မှု 12: Mid-term Project 1

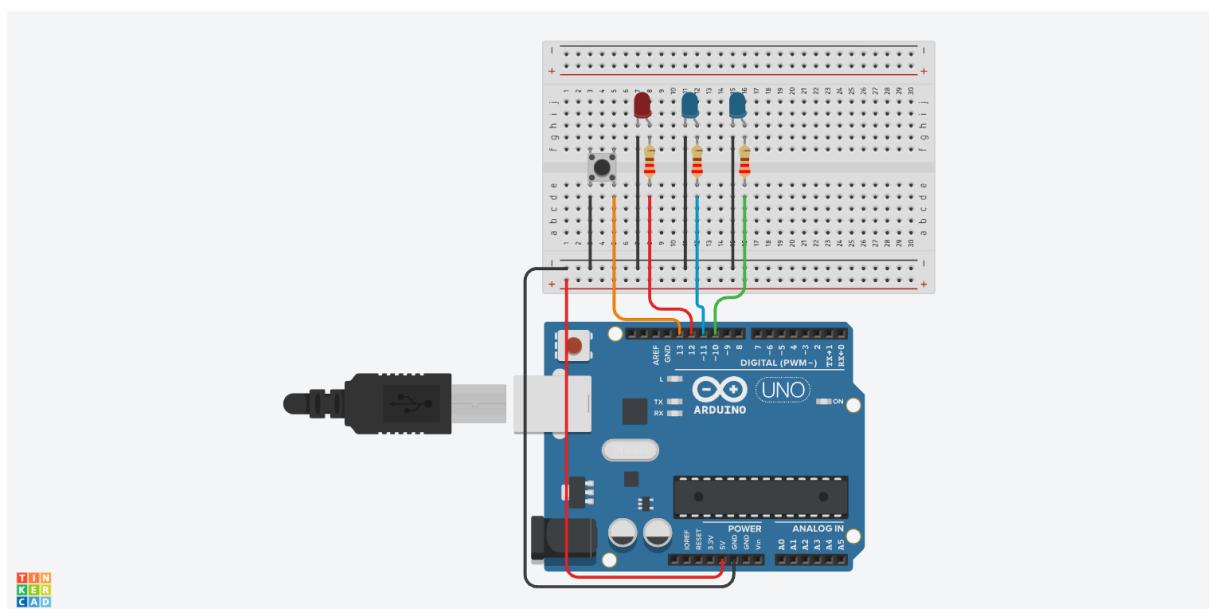
I. ພິດນໍາ

II. ទូរសព្ទ

LED	3
Resistor 220-ohm	3
Button	1
Arduino	1
Breadboard	1

III. งานเขี๊ยมต่อปะกอน

- 1) ស្របតាមរាង LED 3 ពិសេសនៃ Breadboard.
 - 2) ស្រប Resistor 3 ពិសេសនៃ Breadboard ដើម្បីខ្លួនត្រូវតាមរាងតាមការចូលតុលាការ Anode នៃ LED 3 ពិសេស។ ចូលតុលាការតាមលក្ខណៈ digital pin 10, 11, 12 នៃបណ្តុះ Arduino.
 - 3) ទាញយក Cathode នៃ LED ដើម្បីខ្លួនត្រូវតាមរាងតាមការចូលតុលាការ GND នៃ Breadboard.
 - 4) ស្រប Button 3 ពិសេសនៃ Breadboard ដើម្បីខ្លួនត្រូវតាមរាងតាមការចូលតុលាការ digital pin 13 នៃ Arduino សែនីជាន់។



IV. Coding

```
const int buttonPin = 13;  
const int led1 = 10;
```

```

const int led2 = 11;
const int led3 = 12;

int mode = 0;
int lastButtonState = HIGH;

void setup() {
    pinMode(buttonPin, INPUT_PULLUP);
    pinMode(led1, OUTPUT);
    pinMode(led2, OUTPUT);
    pinMode(led3, OUTPUT);
}

void loop() {
    int buttonState = digitalRead(buttonPin);

    // Button Debounce & Mode Switching
    if (buttonState == LOW && lastButtonState == HIGH) {
        mode++;
        if (mode > 3) mode = 0;
        delay(200); // Slight delay to prevent double-triggering
    }
    lastButtonState = buttonState;

    // --- Mode 0: Sequential (1 -> 2 -> 3) ---
    if (mode == 0) {
        int sequence[] = {led1, led2, led3};
        for(int i = 0; i < 3; i++) {
            digitalWrite(sequence[i], HIGH);
            delay(300);
            digitalWrite(sequence[i], LOW);
            // Check button during sequence for better responsiveness
            if(digitalRead(buttonPin) == LOW) break;
        }
    }

    // Mode 1:
    else if (mode == 1) {
        digitalWrite(led1, HIGH);
        digitalWrite(led2, HIGH);
        digitalWrite(led3, HIGH);
        delay(300);
        digitalWrite(led1, LOW);
        digitalWrite(led2, LOW);
        digitalWrite(led3, LOW);
        delay(300);
    }
}

```

```

// Mode 2:
else if (mode == 2) {
    int reverseSequence[] = {led3, led2, led1};
    for(int i = 0; i < 3; i++) {
        digitalWrite(reverseSequence[i], HIGH);
        delay(300);
        digitalWrite(reverseSequence[i], LOW);
        if(digitalRead(buttonPin) == LOW) break;
    }
}

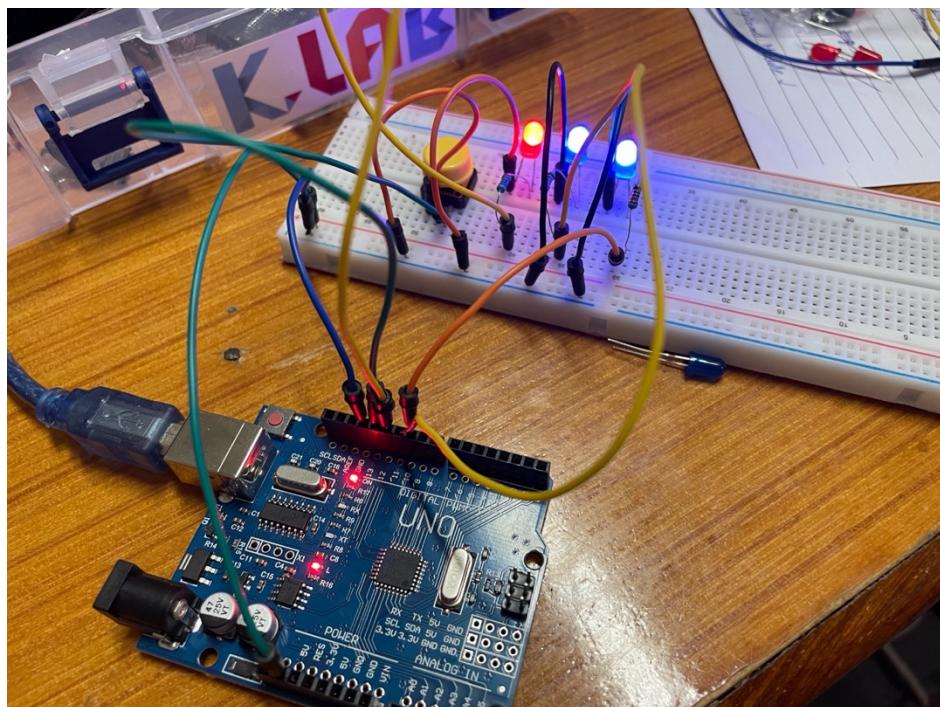
// Mode 3: All Off
else if (mode == 3) {
    digitalWrite(led1, LOW);
    digitalWrite(led2, LOW);
    digitalWrite(led3, LOW);
}

```

V. ការង្គេតរូបភាពនៃការងារ

ម៉ោងនៅក្នុងបុង្ឋាយ 3 គីឡូ និងការងារដែលបានបង្កើតឡើង:

- 1) ឈ្មោះការងារទី 1: ឯកសារការងារដែលត្រួវបានបង្កើតឡើង។
- 2) ឈ្មោះការងារទី 2: ឯកសារការងារដែលត្រួវបានបង្កើតឡើង។
- 3) ឈ្មោះការងារទី 3: ឯកសារការងារដែលត្រួវបានបង្កើតឡើង។
- 4) ឈ្មោះការងារទី 4: ឯកសារការងារដែលត្រួវបានបង្កើតឡើង។



VI. QR Code

Mid-term Project 1



ပິດທີ 13: Mid-term Project 2

I. ປິດນຳ

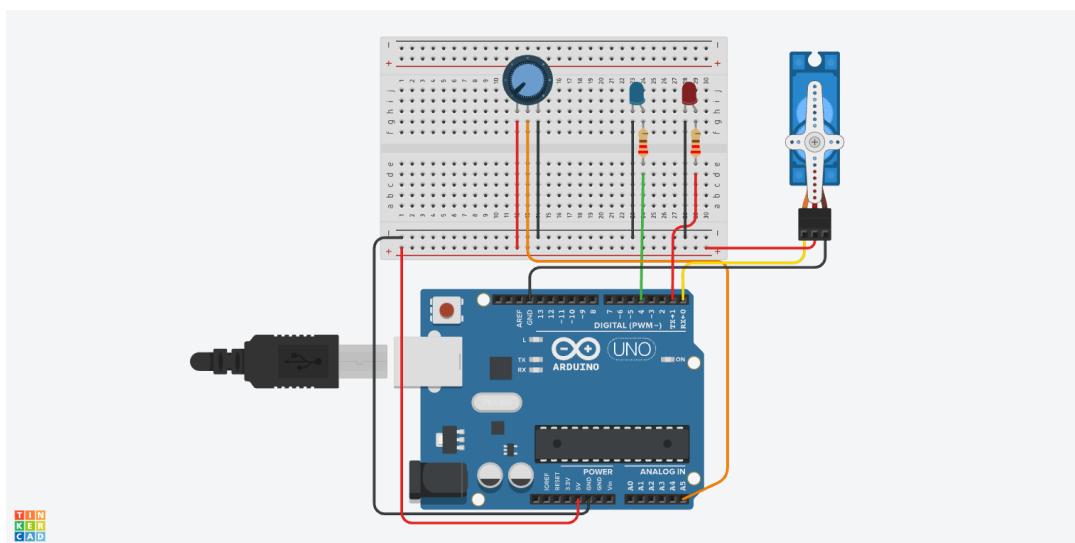
ໃນການທິດລອງນີ້ແມ່ນໂປຣເຈົກ Mid-term ທີ 2 ເຊິ່ງຫົວຂໍຈະແມ່ນກ່ຽວກັບໂດຍມີ Potentiometer ເປັນຕົວຫຼັກໃນການຄວບຄຸມເພື່ອສ້າງອຸປະກອນ ດອກໄຟ LED ແລະ Servo.

II. ອຸປະກອນ

LED	2
Potentiometer	1
Resistor 220-ohm	2
Servo	1
Arduino	1
Breadboard	1

III. ການເຊື່ອມຕໍ່ວົງຈອນ

- ສຽບ LED ຫັງສອງລົງໃນ Breadboard ແລ້ວເອົາ Resistor ຫັງສອງເຊື່ອມຕໍ່ກັບຂຶ້ວ Anode ຂອງ LED ຫຼັງຈາກນັ້ນເຊື່ອມຕໍ່ໃສ່ digital pin 9, 10 ໃນບອດ Arduino ສ່ວນຂ້ວລົບແມ່ນຕໍ່ໃສ່ GND ຂອງ Breadboard.
- ສຽບ Potentiometer ລົງໃນ Breadboard ເຊິ່ງ Terminal 1 ເຊື່ອມຕໍ່ສາຍໃສ່ ຂົ້ວບວກຂອງ Breadboard, Wiper ຂາກາງ ເຊື່ອມໃສ່ Analog A5 ຂອງ Arduino ແລະ Terminal 2 ເຊື່ອມຕໍ່ສາຍໃສ່ຂົ້ວລົບຂອງ Breadboard.
- ຂາ Signal ຂອງ Servo ເຊື່ອມໃສ່ digital pin 3 ຂອງ Arduino.
- ຂາ Power ຂອງ Servo ເຊື່ອມໃສ່ ຂົ້ວບວກຂອງ Breadboard.
- ຂາ GND ຂອງ Servo ເຊື່ອມໃສ່ GND ຂອງ Arduino.
- ຂົ້ວລົບຂອງ Breadboard ເຊື່ອມຕໍ່ສາຍໄປຫາ GND Arduino ສ່ວນ ຂົ້ວບວກໄປຫາ 5V Arduino.



IV. Coding

```
#include <Servo.h>

Servo myServo;
int potPin = A5;
int led1 = 9;
int led2 = 10;
int potVal;
int servoPos;

void setup() {
    myServo.attach(3);
    pinMode(led1, OUTPUT);
    pinMode(led2, OUTPUT);
}

void loop() {
    // read potentiometer (0-1023) and map to servo angle (0-180)
    potVal = analogRead(potPin);
    servoPos = map(potVal, 0, 1023, 0, 180);

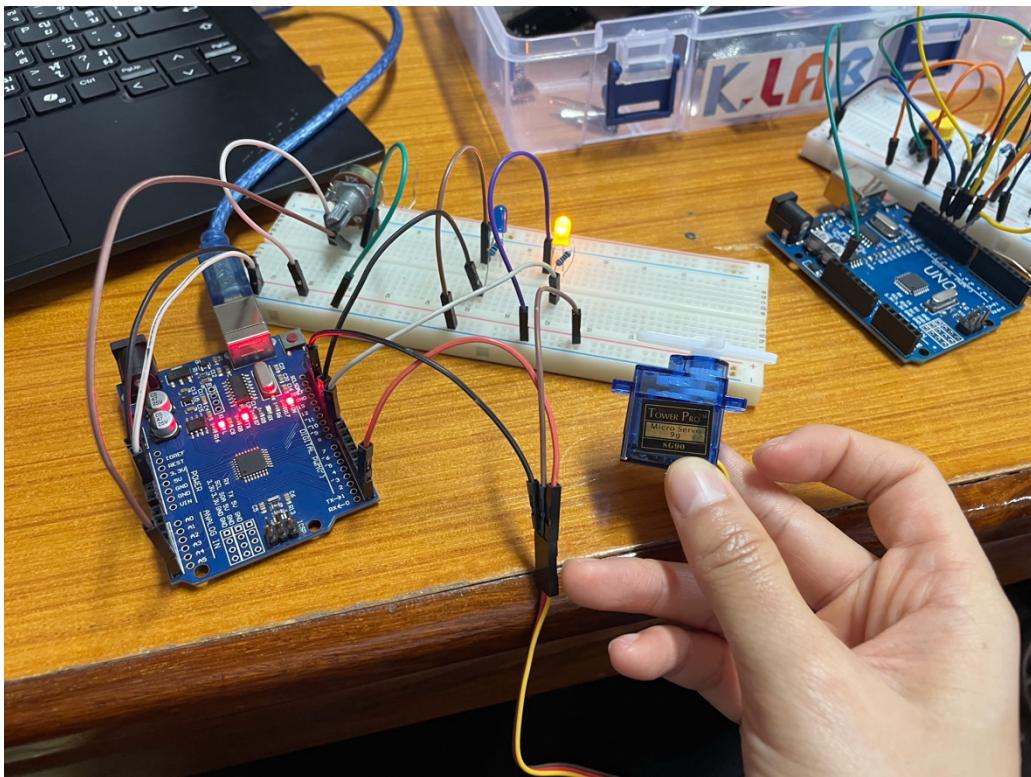
    // move servo
    myServo.write(servoPos);

    // control LEDs
    if(servoPos > 90){
        digitalWrite(led1, HIGH);
        digitalWrite(led2, LOW);
    } else {
        digitalWrite(led1, LOW);
        digitalWrite(led2, HIGH);
    }

    delay(15); // small delay for smooth servo movement
}
```

V. ការង្គេចវិញ្ញាយការងារ

- 1) ការប័បតាំ: ម៉ោងទឹកមុនបូម Potentiometer, មីនិត្យសំណើនាការមុនខែវេរួយ Servo និង ភាព-បិណ្ណភាព។
- 2) ផ្ទាត់ទឹកមុនបូមទៅ 90 ឯកសាត់: ឈឺដែលភាពទី 1 ត្រូវបានសែន ឈឺដែលភាពទី 2 ត្រូវបានកូល។
- 3) ផ្ទាត់ទឹកមុនរាប 0 ឯកសាត់: ឈឺដែលភាពទី 2 ត្រូវបានកូល ឈឺដែលភាពទី 1 ត្រូវបានកូល។



VI. QR Code

Mid-term Project 2



ລວມຄັງວິດີໂອລື້ງ QR Code ຂອງການທິດລອງຕົວຈິງ

			
ບົດທີ່ 1: FirstLab	ບົດທີ່ 2: LED	ບົດທີ່ 3: Push Button	ບົດທີ່ 4: RGB
			
ບົດທີ່ 5: P&A Buzzer	ບົດທີ່ 6: Potentiometer	ບົດທີ່ 7: Servo	ບົດທີ່ 8: Servo+Potentiometer
			
ບົດທີ່ 9: Stepper	ບົດທີ່ 10: LCD	ບົດທີ່ 11: Project	ບົດທີ່ 12: Mid-term P1
			
ບົດທີ່ 13: Mid-term P2			

Google Drive:

<https://drive.google.com/drive/folders/1QVWqpWoxT1KjXFqVjbOGhLBPYhTyweBF?usp=sharing>