Basic Electronics + Arduino

(Pre-Lab) Workshop for CprE

ห้องปฏิบัติการระบบสมองกลฝังตัว (Embedded System Lab)

ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มจพ.

12 ตุลาคม พ.ศ. 2557

ภาพรวมของกิจกรรม

- ประเด็นและแนวทางการเรียนรู้ด้านฮาร์ดแวร์สำหรับ CprE
- แนะนำ Arduino (ฮาร์ดแวร์และซอฟต์แวร์)
- ทดลองใช้ซอฟต์แวร์ Arduino IDE version 1.0.x ภายใต้ระบบปฏิบัติการ Linux / Ubuntu
- ฝึกเขียนโปรแกรมสำหรับ Arduino เบื้องต้น (ใช้คำสั่งพื้นฐาน)
- เริ่มต้นใช้งานบอร์ด Arduino และต่อวงจรอิเล็กทรอนิกส์บนเบรดบอร์ด
- การใช้โปรแกรม Fritzing สำหรับออกแบบการต่อวงจรบนเบรดบอร์ด

ประเด็น / คำถาม

- การเรียนรู้ด้านฮาร์ดแวร์สำหรับวิศวกรรมคอมพิวเตอร์ (Computer Engineering - CprE) เกี่ยวข้องกับความรู้และทักษะในด้านใดบ้าง
- ในการเรียนรู้ด้านฮาร์ดแวร์นั้น Arduino มีความสำคัญอย่างไร และเกี่ยวข้องอย่างไรกับวิชาในสาขาวิศวกรรมคอมพิวเตอร์
- ต้องใช้ความรู้พื้นฐานการเขียนโปรแกรมหรือไม่ และใช้ภาษาคอมพิวเตอร์ใดบ้าง (เช่น Processing, Java, Python, C, C++)_

หัวข้อในการเรียนรู้พื้นฐานด้านฮาร์ดแวร์สำหรับ CprE

- พื้นฐานทางไฟฟ้า เช่น กฎของโอห์ม ความปลอดภัยสำหรับงานไฟฟ้า
- การวัดปริมาณทางไฟฟ้า และการใช้เครื่องมือวัด เช่น มัลติมิเตอร์ ออสซิลโลสโคป
- ชิ้นส่วนหรือองค์ประกอบในวงจรอิเล็กทรอนิกส์
- การออกแบบ และวิเคราะห์วงจรอิเล็กทรอนิกส์
- การต่อวงจร และการประกอบวงจรอิเล็กทรอนิกส์
- การตรวจสอบ ค้นหาสาเหตุ ป้องกันและแก้ไขข้อผิดพลาดในวงจร
- การประยุกต์ใช้งานวงจรอิเล็กทรอนิกส์ในด้านต่างๆ
- การเขียนโปรแกรมสำหรับระบบสมองกลที่ใช้ไมโครคอนโทรลเลอร์

  ↳ ใช้ Arduino เป็นสื่อการเรียนรู้ และเชื่อมโยงไปสู่วิชาอื่น เช่น Electronics Lab, Embedded System Design

หัวข้อพื้นฐานในการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **พื้นฐานทางไฟฟ้า**
  - หลักการ ทฤษฎีพื้นฐานทางไฟฟ้า เช่น กฎของโอห์ม
  - เครื่องมือช่างพื้นฐาน (ไฟฟ้าและอิเล็กทรอนิกส์)
  - ข้อควรระวังและอันตรายเกี่ยวกับไฟฟ้า
  - ชนิดของสัญญาณไฟฟ้า (แอนะล็อก - ดิจิทัล)
  - ความสำคัญและบทบาทของอุปกรณ์ไฟฟ้าและอิเล็กทรอนิกส์ในชีวิตประจำวัน เช่น ความปลอดภัย การประหยัดพลังงาน

หัวข้อพื้นฐานในการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **การวัดปริมาณทางไฟฟ้าและใช้เครื่องมือวัด**
  - ปริมาณทางไฟฟ้า (ความต้านทาน กระแสไฟฟ้า และแรงดันไฟฟ้า)
  - ความสัมพันธ์ระหว่างปริมาณทางไฟฟ้า
  - การวัดปริมาณทางไฟฟ้าในวงจร โดยใช้เครื่องมือวัด เช่น มัลติมิเตอร์ ได้อย่างถูกต้องและเหมาะสม
  - การทดลองปฏิบัติ การวิเคราะห์และสรุปผลการทดลอง

หัวข้อพื้นฐานในการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **ส่วนประกอบในวงจรอิเล็กทรอนิกส์พื้นฐาน**
  - หลักการทำงานของอุปกรณ์หรือชิ้นส่วนประเภทต่างๆ

    เช่น ตัวต้านทาน ตัวเก็บประจุ ไดโอด ทรานซิสเตอร์ ไอซีหรือวงจรรวมต่างๆ ไอซีลอจิก ไมโครคอนโทรลเลอร์ แหล่งจ่ายไฟเลี้ยงกระแสตรง เซนเซอร์ประเภทต่างๆ ...
  - การสืบค้นข้อมูลจากอินเตอร์เน็ต การทำความเข้าใจ **“เอกสารคู่มือการใช้งาน”** (User’s Manual) หรือ **“ดาต้าชีท”** (Datasheet) ที่เป็น **“ภาษาอังกฤษ”**
  - การเลือกใช้อุปกรณ์อย่างถูกต้องและเหมาะสม

หัวข้อพื้นฐานในการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **การออกแบบและวิเคราะห์วงจรอิเล็กทรอนิกส์**
  - ขั้นตอนและกระบวนการออกแบบวงจรอิเล็กทรอนิกส์
  - การวาดผังวงจรหรือไดอะแกรมสำหรับวงจรอิเล็กทรอนิกส์
  - การใช้สัญลักษณ์ทางไฟฟ้าของอุปกรณ์อิเล็กทรอนิกส์
  - การวิเคราะห์การทำงานวงจรอิเล็กทรอนิกส์ ซึ่งแบ่งได้เป็น
    - ระดับอุปกรณ์ (device level)
    - ระดับวงจร (circuit level) และ
    - ระดับระบบ (system level)
  - การใช้โปรแกรมซอฟต์แวร์สำหรับงานดังกล่าว

หัวข้อพื้นฐานในการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **การต่อวงจรและประกอบวงจรอิเล็กทรอนิกส์**
  - ความรู้และทักษะพื้นฐานในการต่อวงจร
  - การอ่านแบบ (ผังวงจร)
  - การต่อวงจรหรือประกอบวงจรตามแบบหรือผังวงจร
  - การเลือกใช้เครื่องมือและวัสดุอุปกรณ์ให้เหมาะสม
    - เช่น การใช้แผงต่อวงจรอเนกประสงค์ การใช้อุปกรณ์บัดกรี การใช้งานแผ่นวงจรพิมพ์สำเร็จรูป

หัวข้อพื้นฐานในการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **การตรวจสอบ ค้นหาสาเหตุ ป้องกันและแก้ไขข้อผิดพลาดในวงจรอิเล็กทรอนิกส์**
  - การตรวจสอบความถูกต้องของวงจรอิเล็กทรอนิกส์ เพื่อป้องกันความผิดพลาดที่อาจเกิดขึ้นได้
  - การค้นหาสาเหตุและวิธีแก้ไขข้อผิดพลาดในวงจร
  - การซ่อมแซมวงจรอิเล็กทรอนิกส์เบื้องต้น

หัวข้อพื้นฐานในการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **การประยุกต์ใช้งานวงจรอิเล็กทรอนิกส์**
  - การประยุกต์ใช้ความรู้และทักษะในการสร้างวงจร

    เช่น การสร้างหุ่นยนต์ การสร้างสิ่งประดิษฐ์ การทำโครงงานวิทยาศาสตร์ เป็นต้น
  - รวมถึงการนำเสนอผลงาน (Powerpoint Presentation, Video Clip / Youtube) การจัดทำเอกสาร คู่มือการใช้งาน

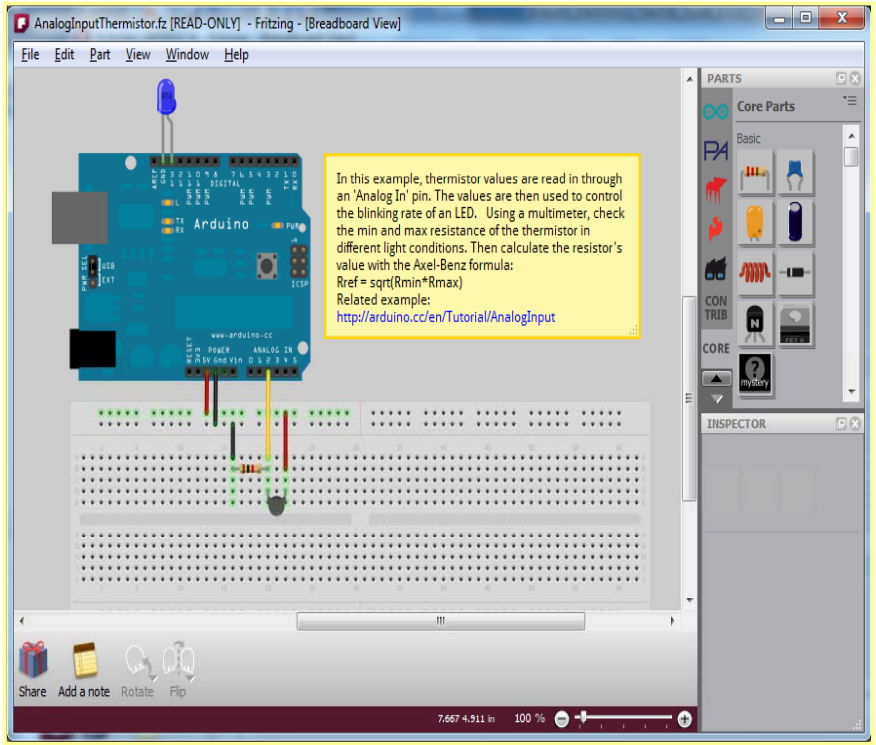
หัวข้อพื้นฐานในการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **การเขียนโปรแกรมสำหรับระบบสมองกลที่ใช้ไมโครคอนโทรลเลอร์**
  - การเขียนโปรแกรมในภาษาคอมพิวเตอร์ระดับสูง
  - การวางแผน กำหนดขั้นตอนหรือเงื่อนไขในการทำงานของโปรแกรมสำหรับไมโครคอนโทรลเลอร์
  - การใช้ซอฟต์แวร์พัฒนาโปรแกรมสำหรับระบบสมองกลฝังตัว

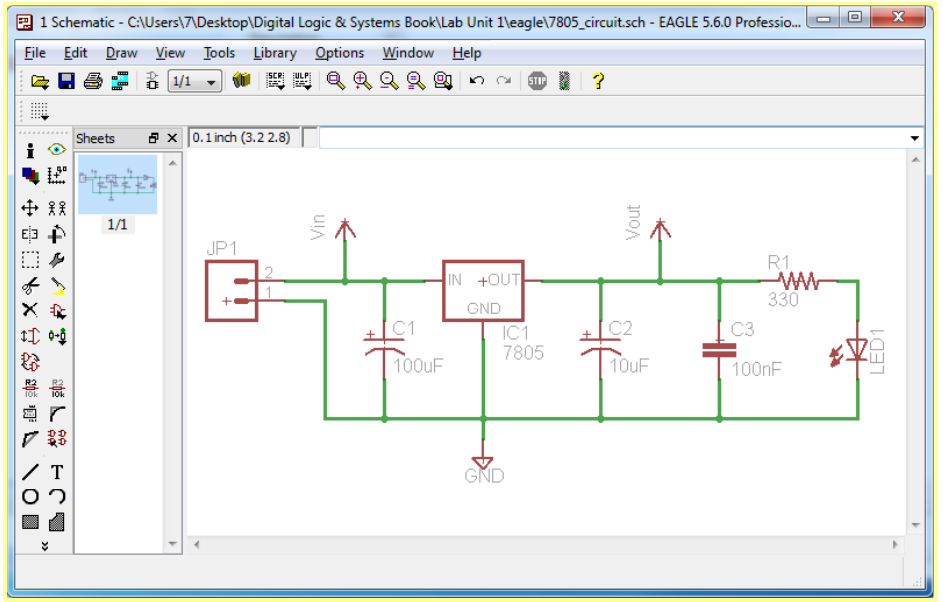
การจำแนกสื่อการเรียนรู้ด้านฮาร์ดแวร์ (อิเล็กทรอนิกส์)

- **จำแนกกิจกรรมการเรียนรู้ตามสื่อการเรียนรู้ที่พบเห็นได้**
  - การออกแบบวงจร และการจำลองการทำงานโดยใช้คอมพิวเตอร์และซอฟต์แวร์
  - การต่อวงจรอิเล็กทรอนิกส์โดยใช้ฮาร์ดแวร์จริง
  - การฝึกประกอบและบัดกรีชิ้นส่วนอิเล็กทรอนิกส์
  - การฝึกต่อวงจรอิเล็กทรอนิกส์โดยใช้แผงวงจรสำเร็จรูป
  - การฝึกเขียนโปรแกรมไมโครคอนโทรลเลอร์

การออกแบบวงจรด้วยซอฟต์แวร์และตัวอย่างซอฟต์แวร์ที่ใช้

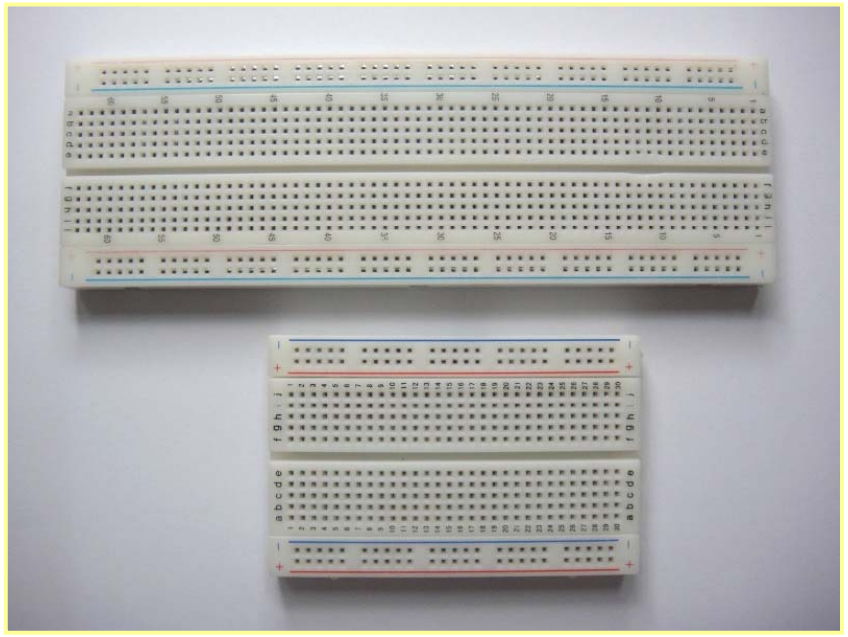


โปรแกรม Fritzing (Open Source)
สำหรับการวาดผังวงจร
และวางแผนการต่อวงจรบนเบรดบอร์ด



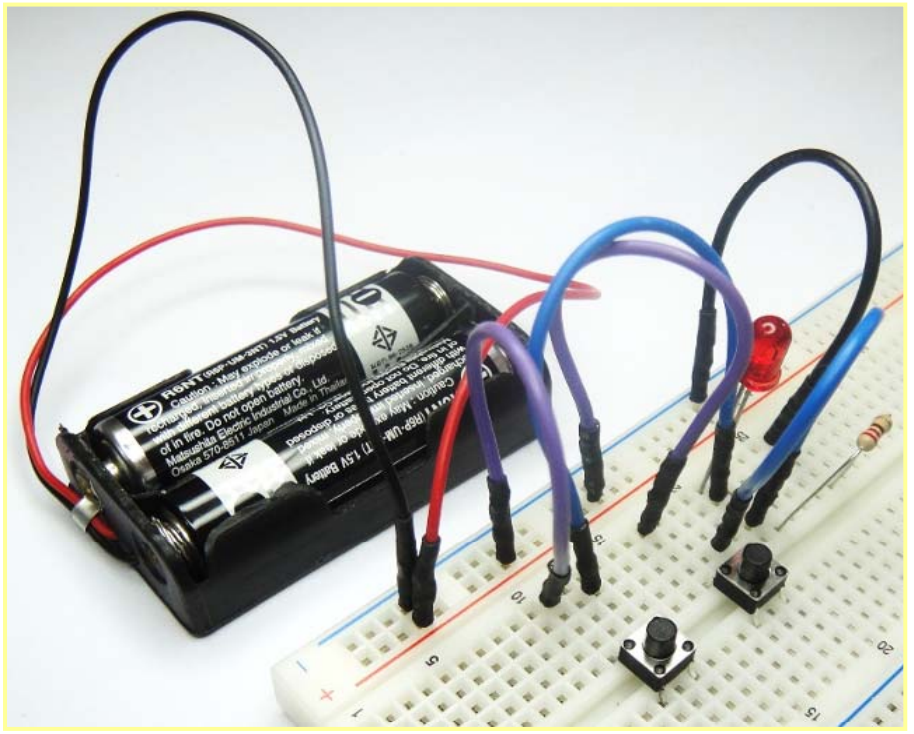
โปรแกรม CadSoft's Eagle (Free)
สำหรับการวาดผังวงจร
และออกแบบลายวงจรพิมพ์ (PCB)

การฝึกต่อวงจรโดยไม่ต้องบัดกรี

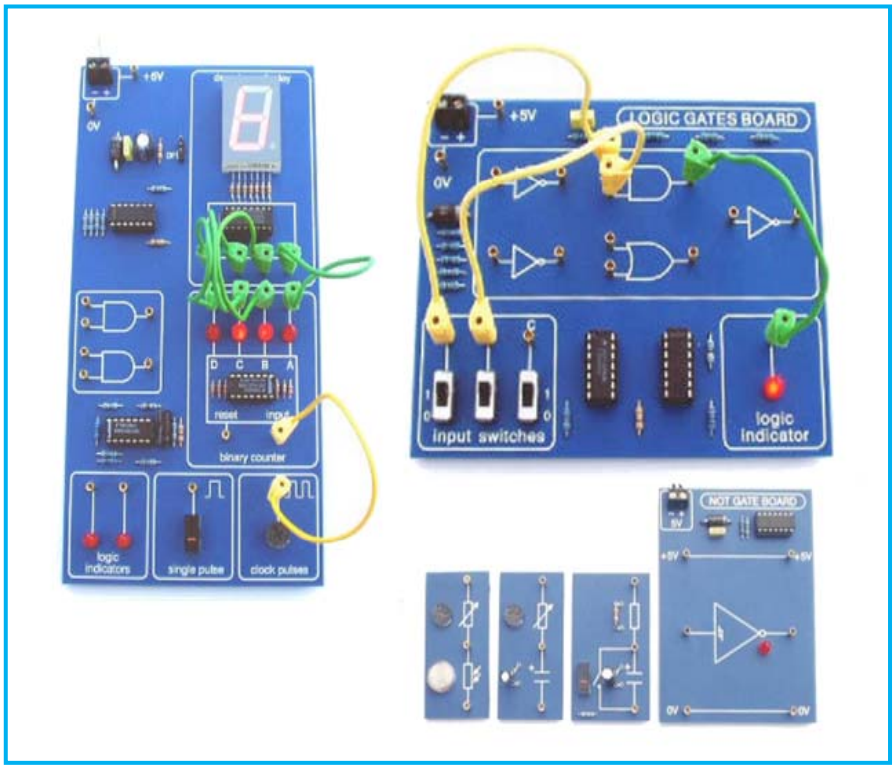


แผงต่อวงจร หรือ เบรดบอร์ด (Breadboard)

ตัวอย่างการต่อวงจรบนเบรดบอร์ด



ตัวอย่างชุดทดลองต่อวงจร

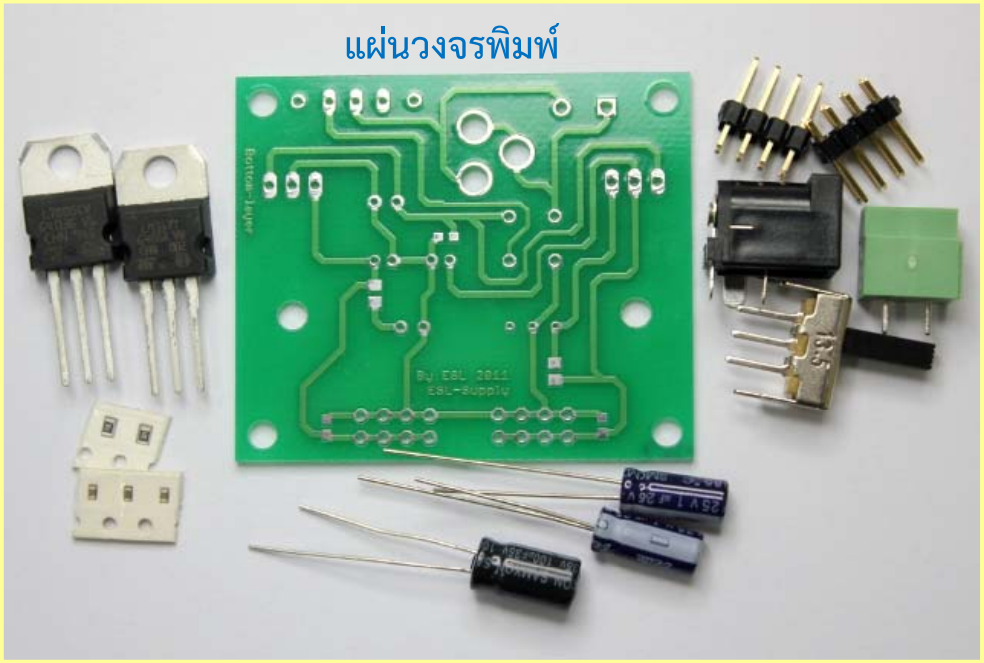


ตัวอย่างชุดทดลองต่อวงจร (จากบริษัทต่างประเทศ)

ตัวอย่างชุดทดลองต่อวงจร (จากบริษัทในประเทศ)

ตัวอย่างวัสดุในการฝึกบัดกรีวงจร

แผ่นวงจรพิมพ์

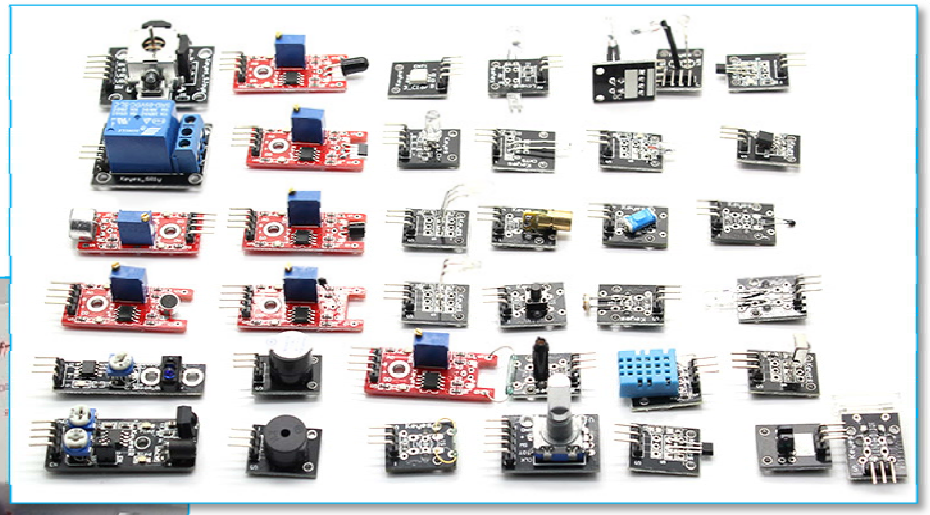


ชุดคิท (Kit) หรือแผงวงจรอิเล็กทรอนิกส์ที่ต้องประกอบเอง
สำหรับฝึกประกอบและบัดกรีชิ้นส่วนอิเล็กทรอนิกส์
ตามวงจรที่ได้ออกแบบไว้แล้ว

อุปกรณ์ในการบัดกรี

ตัวอย่างแผงวงจรสำเร็จรูป และชุดคิท

Fritzing Creator Kit



37-in-1 Sensor Module Kit

ภาพรวมของกิจกรรม

- ประเด็นและแนวทางการเรียนรู้ด้านฮาร์ดแวร์สำหรับ CprE
- แนะนำ Arduino (ฮาร์ดแวร์และซอฟต์แวร์)
- ทดลองใช้ซอฟต์แวร์ Arduino IDE version 1.0.x ภายใต้ระบบปฏิบัติการ Linux / Ubuntu
- ฝึกเขียนโปรแกรมสำหรับ Arduino เบื้องต้น (ใช้คำสั่งพื้นฐาน)
- เริ่มต้นใช้งานบอร์ด Arduino และต่อวงจรอิเล็กทรอนิกส์บนเบรดบอร์ด
- การใช้โปรแกรม Fritzing สำหรับออกแบบการต่อวงจรบนเบรดบอร์ด

แนะนำ Arduino

- Arduino (http://arduino.cc)
  - เป็นแนวคิดในการสร้างสิ่งที่เรียกว่า แพลตฟอร์ม (Platform)
  - ประกอบด้วยทั้งฮาร์ดแวร์ (Hardware) และ ซอฟต์แวร์ (Software)
  - ใช้สำหรับการเชื่อมต่อคอมพิวเตอร์กับสิ่งแวดล้อมทางกายภาพและรวมถึงผู้ใช้ด้วย (หรือที่เรียกว่า Physical Computing)
  - เริ่มต้นเผยแพร่ผลงานในราวปี ค.ศ. 2005
  - มีวัตถุประสงค์คือ การสร้างฮาร์ดแวร์และซอฟต์แวร์ที่เป็น Open Source
    - เปิดเผยโค้ดต้นฉบับ ผังวงจรสำหรับบอร์ดไมโครคอนโทรลเลอร์ และเอกสารรายละเอียดอื่นๆ เช่น คู่มือการใช้งาน โค้ดตัวอย่าง ภายใต้เงื่อนไขการนำไปเผยแพร่และพัฒนาต่อ
  - เหมาะสำหรับผู้ที่ต้องการสร้างอุปกรณ์อิเล็กทรอนิกส์เป็นต้นแบบ (Electronics Prototyping) และนำไปสู่การนำเสนอผลงาน และแลกเปลี่ยนอย่างเสรีในวงกว้าง

ทำไม Arduino จึงได้รับความนิยมและแพร่หลาย

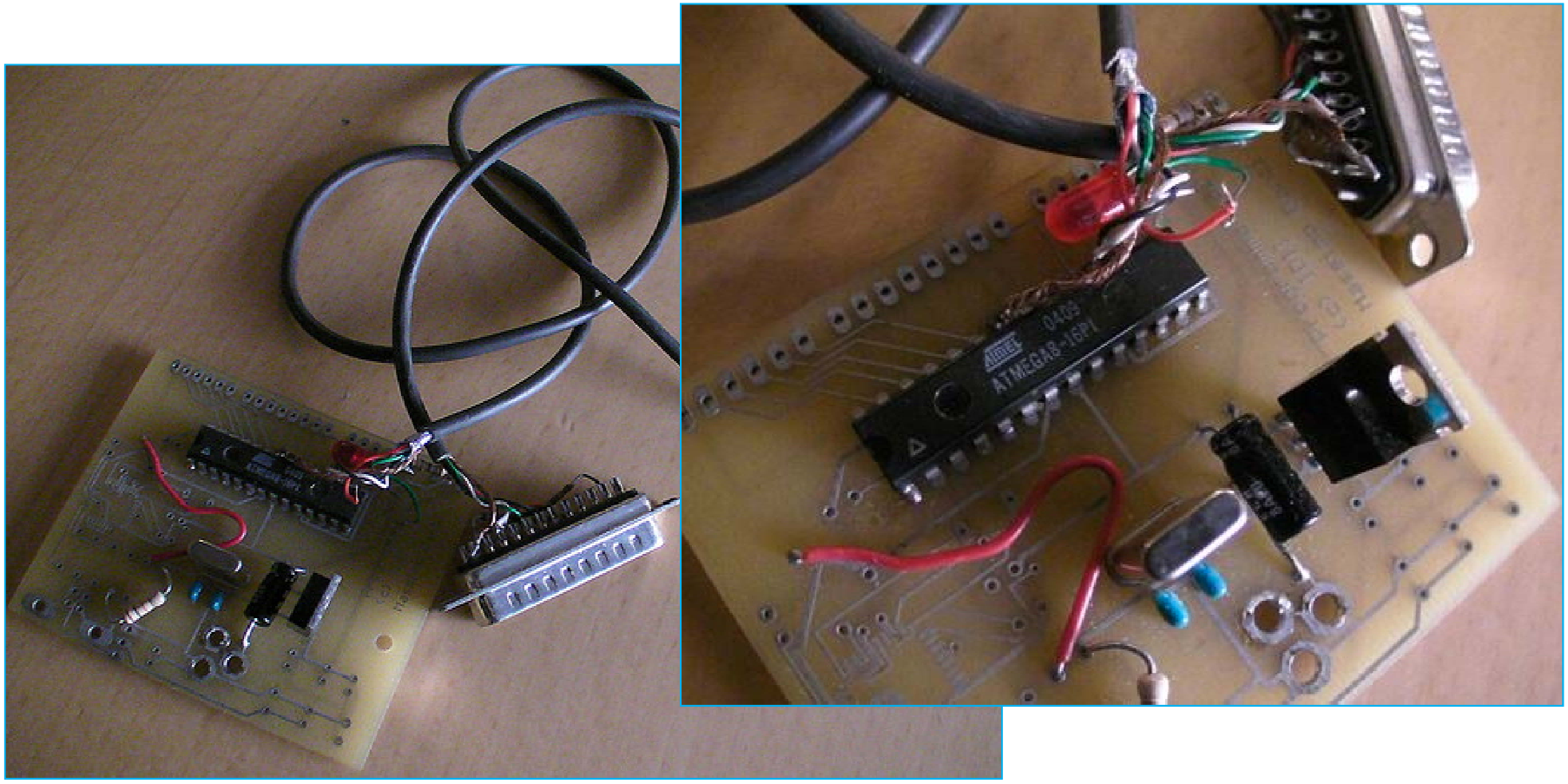
- ใช้งานง่าย
  - เช่น ใช้แหล่งจ่ายไฟเลี้ยงจากพอร์ต USB + ดาวน์โหลดโปรแกรมผ่านสาย USB ได้
- ซอฟต์แวร์ในการพัฒนาโปรแกรมเป็น Opensource
  - ทำงานได้หลายแพลตฟอร์ม (Windows, Mac OS X, Linux)
  - เปิดเผยโค้ดต้นฉบับ (Sourcecode) และนำไปพัฒนาต่อยอดได้
- เปิดเผยวงจร (ไฟล์ผังวงจรและลายวงจรสำหรับ PCB) เป็น OpenHardware
  - เปิดเผยวงจรภายใต้ลิขสิทธิ์ที่ชื่อว่า Creative Commons License
    - http://creativecommons.org/licenses/
  - สามารถนำไปดัดแปลง ต่อขยายและเพิ่มเติมได้ภายใต้เงื่อนไขตามลิขสิทธิ์ดังกล่าว
  - มีความหลากหลายทางฮาร์ดแวร์ Arduino / Arduino Clones / Arduino Shields

Arduino Software

- ใช้ซอฟต์แวร์ Arduino IDE ในการเขียนโปรแกรม
  - มีสองเวอร์ชัน 1.0.5 และ 1.5.x (รองรับฮาร์ดแวร์ใหม่ เช่น Due, Yun)
  - สำหรับระบบปฏิบัติการ Windows, Linux, Mac OS X
  - ใช้ภาษา C/C++ ในการเขียนโปรแกรม (เรียกว่า Arduino Sketch)
    - ดังนั้นสามารถเขียนโปรแกรมเชิงวัตถุได้ (Object-oriented Programming / OOP)
    - คลาส (Class) หรือ ออปเจค (Object) เกี่ยวข้องกับการใช้งานฮาร์ดแวร์ต่างๆ ของตัวประมวลผล หรือที่เรียกว่า ไมโครคอนโทรลเลอร์ (Microcontroller)
- มีชุดคำสั่งสำหรับใช้งานสำหรับ Arduino
  - มีการสร้างฟังก์ชัน (Functions) หรือคำสั่งไว้ให้เรียกใช้งานหลายคำสั่ง
  - จัดทำในรูปของไลบลารี่ในภาษา C++ (C++ Library) ต่างๆ

IDE = Integrated Development Environment

The First Prototype of Arduino Hardware



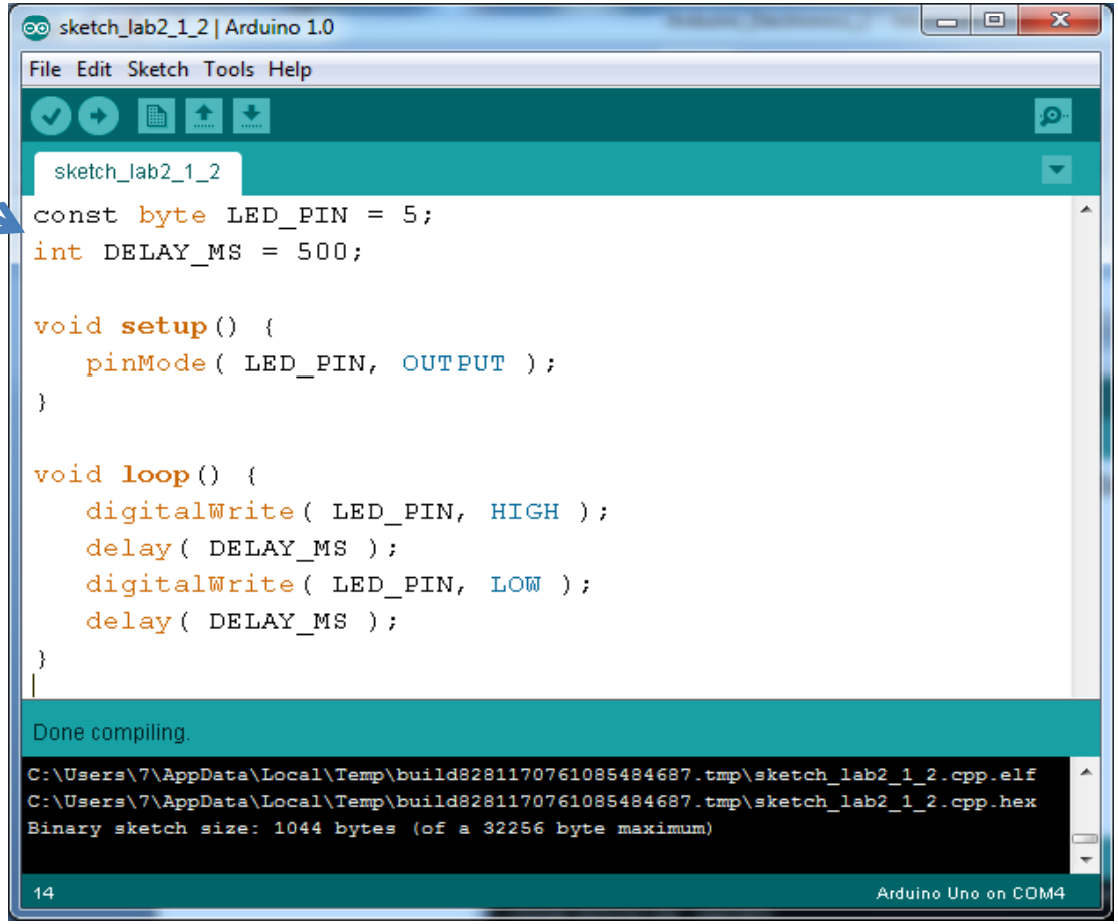
ที่มา: http://makezine.com/2014/03/28/a-look-at-arduinos-origins-the-first-prototype/

หน้าต่างหลักของ Arduino IDE v1.0.x

ประกาศตัวแปรภายนอก (Global Variables หรือ ค่าคงที่ (Constants)

ประกอบด้วยฟังก์ชันที่ต้องมีคือ setup() และ loop()

โครงสร้างของโค้ดคล้ายกับของภาษา Processing



Arduino Hardware

- เป็นบอร์ดไมโครคอนโทรลเลอร์
  - บางทีก็เรียกว่า Arduino I/O Board
  - เป็นส่วนเชื่อมต่อกับวงจรหรืออุปกรณ์อิเล็กทรอนิกส์อื่นๆ
- นิยมใช้ชิปไมโครคอนโทรลเลอร์ ตระกูล ATmega ของบริษัท Atmel
  - เบอร์ที่นิยมใช้คือ ATmega 168 และ 328P (ต่างกัน เช่น ความจุของหน่วยความจำภายในชิป)
    - ความจุของหน่วยความจำสำหรับโปรแกรม (Program Memory) แบบ Flash
    - ความจุของหน่วยความจำสำหรับข้อมูล (Data Memory) แบบ SRAM
  - ยังมีเบอร์อื่นอีก เช่น ATmega 1280 และ 2560 เป็นต้น

ตัวถัง (Package) ของ ATmega 168/328P

ATmegaXX8 Family

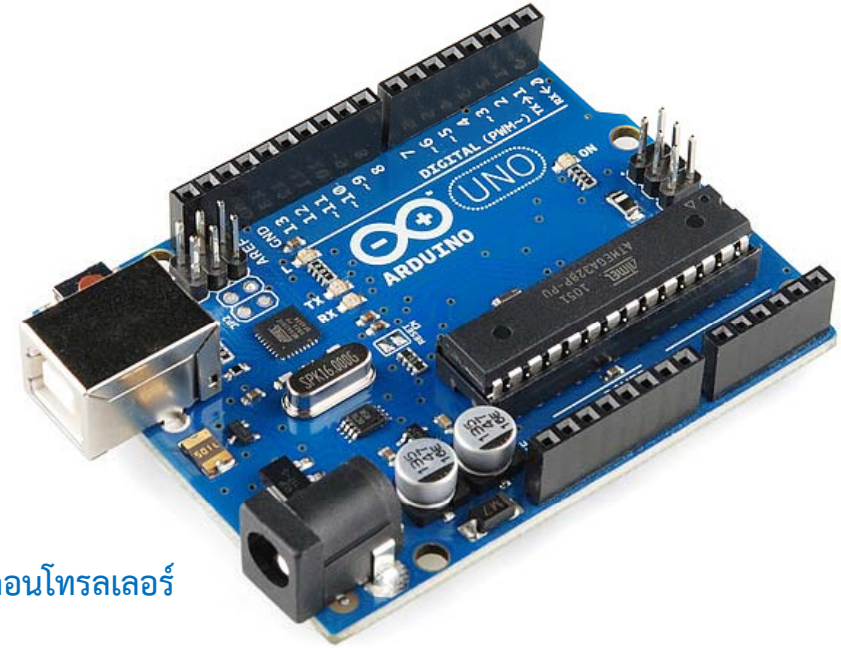
Image	Part # & Datasheet	Flash	SRAM	Distributors	EAGLE library
	Atmega328P-PU (DIP)	32K	2K	Digikey Mouser	avr-6
	Atmega328P-AU (TQFP)	32K	2K	Digikey Mouser Arrow	avr-6
	Atmega168V-PU	16K	1K	Digikey Mouser	avr-6
*	Atmega168-PU	16K	1K	Digikey Mouser	avr-6
*	Atmega48V-PU	4K	512b	Digikey Mouser	avr-6

ที่มา: http://www.adafruit.com/blog/2010/10/03/looking-for-atmega328p-in-tqfp/

ตัวอย่างบอร์ด: Arduino Uno R3

Features:

- ATmega328 microcontroller (5v), 16MHz
- Input voltage: 7..12V
- 14 Digital I/O Pins (6 PWM outputs)
- 6 Analog Inputs
- 32k Flash Memory
- 16Mhz Clock Speed



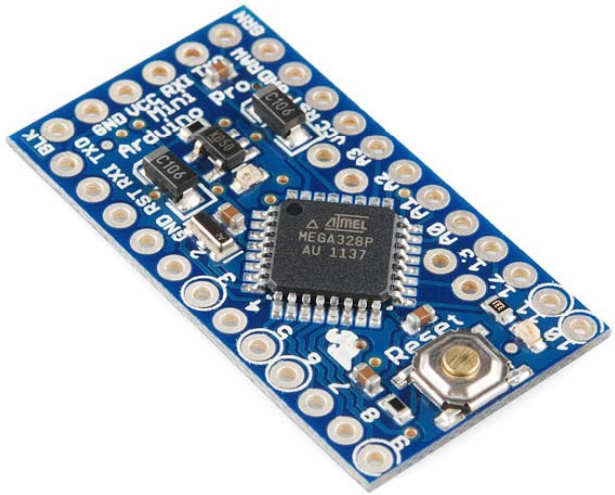
สามารถเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB ได้
เพื่อดาวน์โหลดโปรแกรมสำหรับ Arduino ลงในตัวไมโครคอนโทรลเลอร์

ที่มา: http://www.sparkfun.com/products/11021

ตัวอย่างบอร์ด: Arduino Pro Mini 328 (5V/16MHz)

Features:

- ATmega328 running at 16MHz
- Interfacing circuitry to 5V devices
- USB connection off board
- 5V regulator, Max 150mA output
- Over current protected
- Reverse polarity protected
- DC input 5V up to 12V
- Analog Pins: 8
- Digital I/Os: 14



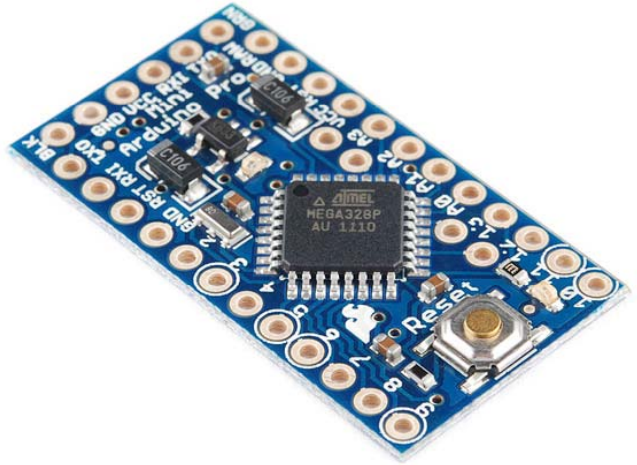
- ไม่สามารถเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB ได้โดยตรง ต้องมีอุปกรณ์เสริม เพื่อใช้ในการเชื่อมต่อระหว่างบอร์ด Arduino นี้กับพอร์ต USB ของคอมพิวเตอร์
- ใช้สำหรับเชื่อมต่อกับวงจรอื่นที่แรงดัน 5V เท่านั้น

ที่มา: http://www.sparkfun.com/products/11113

ตัวอย่างบอร์ด: Arduino Pro Mini 328 (3.3V/8MHz)

Features:

- ATmega328 running at 8MHz
- Interfacing circuitry to 3.3V devices
- USB connection off board
- 3.3V regulator, Max 150mA output
- Over current protected
- Reverse polarity protected
- DC input 3.3V up to 12V
- Analog Pins: 8
- Digital I/Os: 14



- ไม่สามารถเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB ได้โดยตรง ต้องมีอุปกรณ์เสริม เพื่อใช้ในการเชื่อมต่อระหว่างบอร์ด Arduino กับพอร์ต USB ของคอมพิวเตอร์
- ใช้สำหรับเชื่อมต่อกับวงจรอื่นที่แรงดัน 3.3V เท่านั้น

ที่มา: http://www.sparkfun.com/products/11114

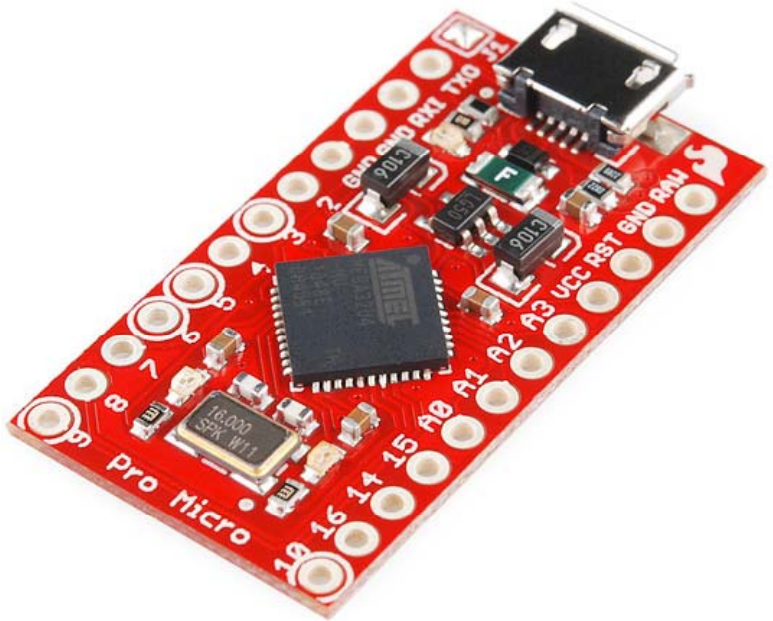
ตัวอย่างบอร์ด: Pro Micro (5V/16MHz)

Features

- ATMega 32U4 (5V/16MHz)
- Supported under Arduino IDE v1.0 ("leonardo")
- On-Board micro-USB connector for programming
- 4 x 10-bit ADC pins
- 12 x Digital I/Os (5 are PWM capable)
- Rx and Tx Hardware Serial Connections

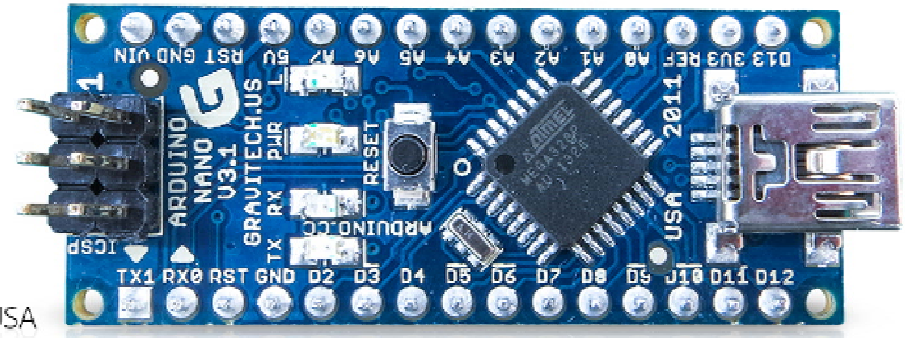
- สามารถเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB ได้โดยตรง
- ใช้สำหรับเชื่อมต่อกับวงจรอื่นที่แรงดัน 5V เท่านั้น

เหมาะสำหรับบัดกรี Pin Header (Male) ให้ขาอยู่ด้านล่าง (ให้บัดกรีด้านบน) เวลาใช้งานเ ให้เสียบขาลงบนเบรดบอร์ด



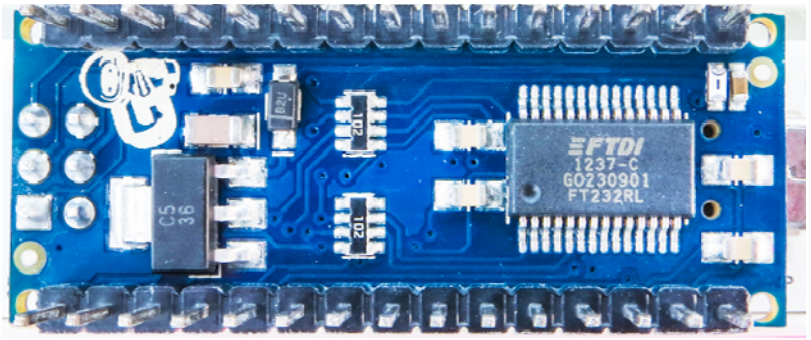
ที่มา: http://www.sparkfun.com/products/11098

ตัวอย่างบอร์ด: Arduino Nano



Made in USA

Arduino Nano 3.1



ที่มา: http://gravitechthai.com/

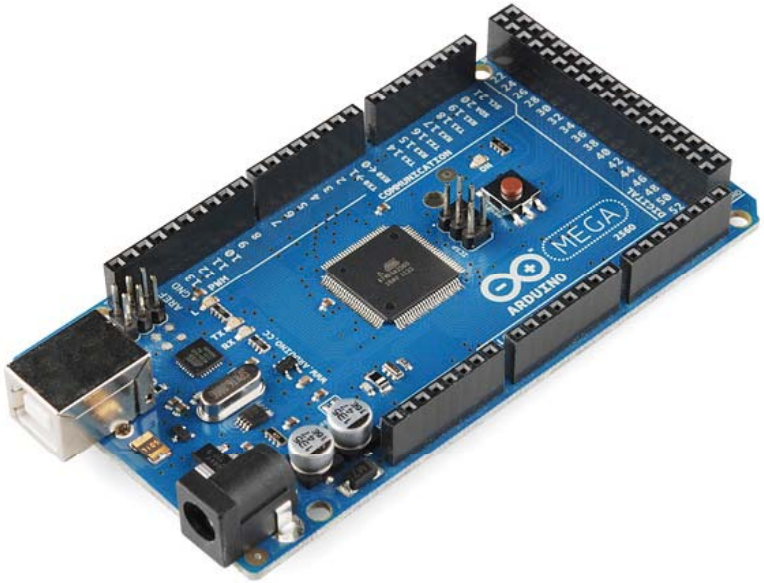
ตัวอย่างบอร์ด: Leonardo (ATmege32u4, 5V/16MHz)

http://arduino.cc/en/Main/arduinoBoardLeonardo

http://arduino.cc/en/Guide/ArduinoLeonardoMicro

ตัวอย่างบอร์ด: Arduino Mega 2560 R3 และ Mega Pro 3.3V

ATmega2560 (5V / 16MHz)

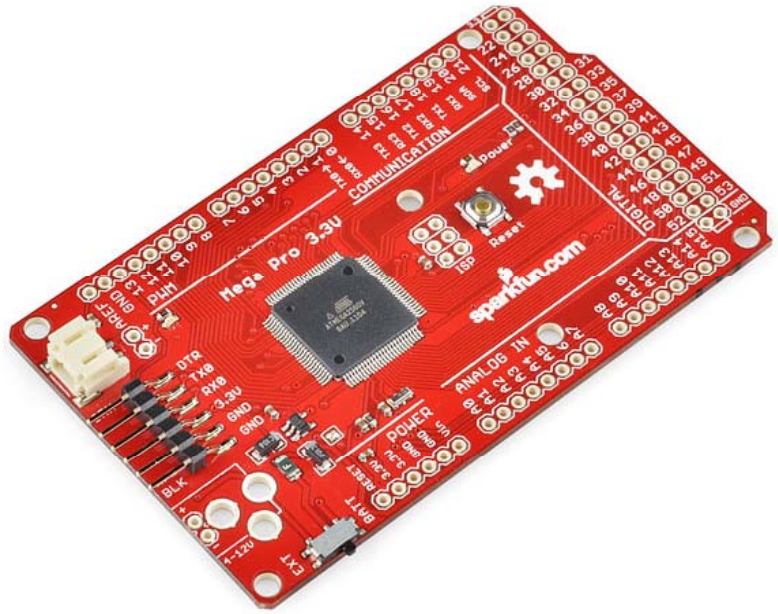


ที่มา: http://www.sparkfun.com/products/11061

- สามารถเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB ได้โดยตรง
- ใช้สำหรับเชื่อมต่อกับวงจรอื่นที่แรงดัน 5V เท่านั้น

ATmega2560 (3.3V / 8MHz)

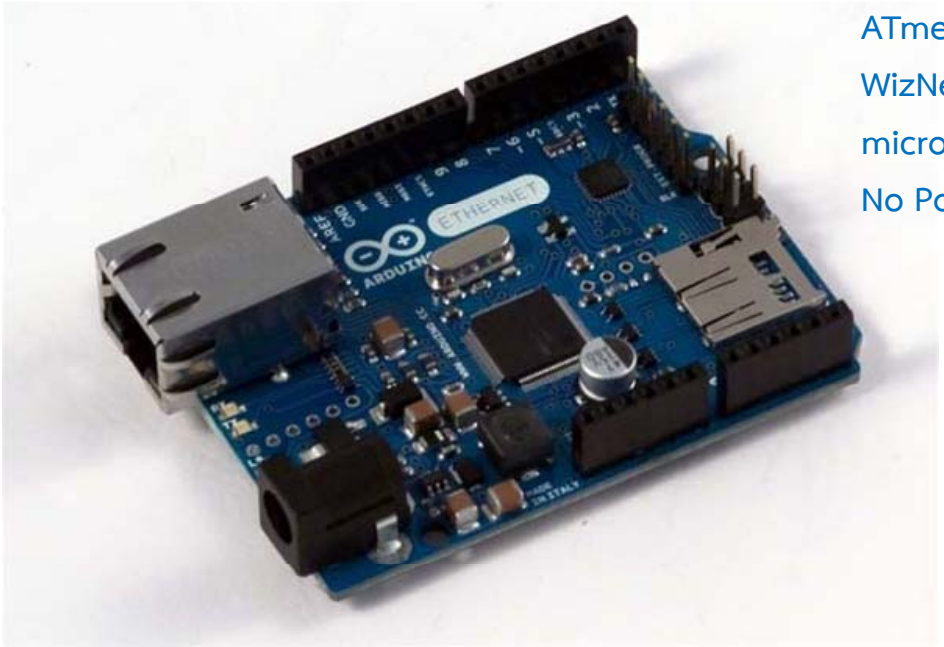
Requires FTDI Basic Breakout board !



ที่มา: http://www.sparkfun.com/products/10744

- ไม่สามารถเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB ได้โดยตรง (ต้องใช้อุปกรณ์อย่างเช่น FTDI Basic Breakout ต่อเพิ่ม)
- ใช้สำหรับเชื่อมต่อวงจรอื่นที่แรงดัน 3.3V เท่านั้น

ตัวอย่างบอร์ด: Arduino Ethernet without PoE



ATmega328, 16MHz / 5V

WizNet W5100 Ethernet Controller

microSD card reader

No Power-over Ethernet (PoE)

The Arduino Ethernet integrates an Arduino Uno with the Ethernet Shield.

The board does not include a POE module.

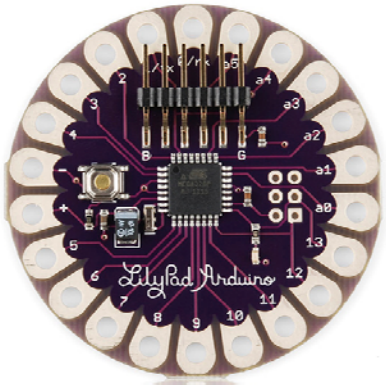
It requires an FTDI breakout board for programming.

ที่มา: http://www.makershed.com/Arduino_Ethernet_WITHOUT_Poe_p/mksp9.htm

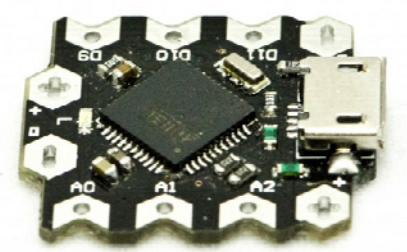
http://arduino.cc/en/Main/ArduinoBoardEthernet

ตัวอย่างฮาร์ดแวร์ที่เกี่ยวข้องกับ Arduino

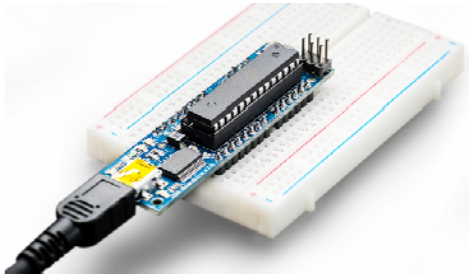
Arduino LilyPad



DFRobot Beetle (ATmega32u4)



USB Boarduino (ATmega328P)

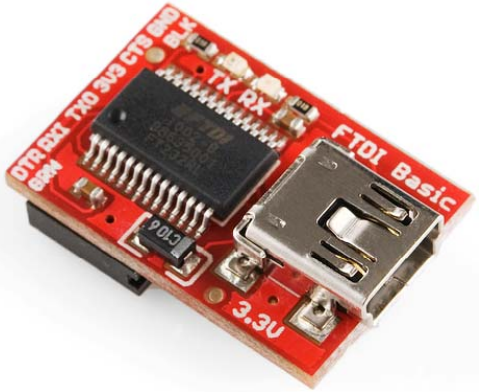


LCD + Keypad Shield

7-segment display Shield

Arduino WiFi Shield

ตัวอย่างอุปกรณ์ USB-to-Serial Converter

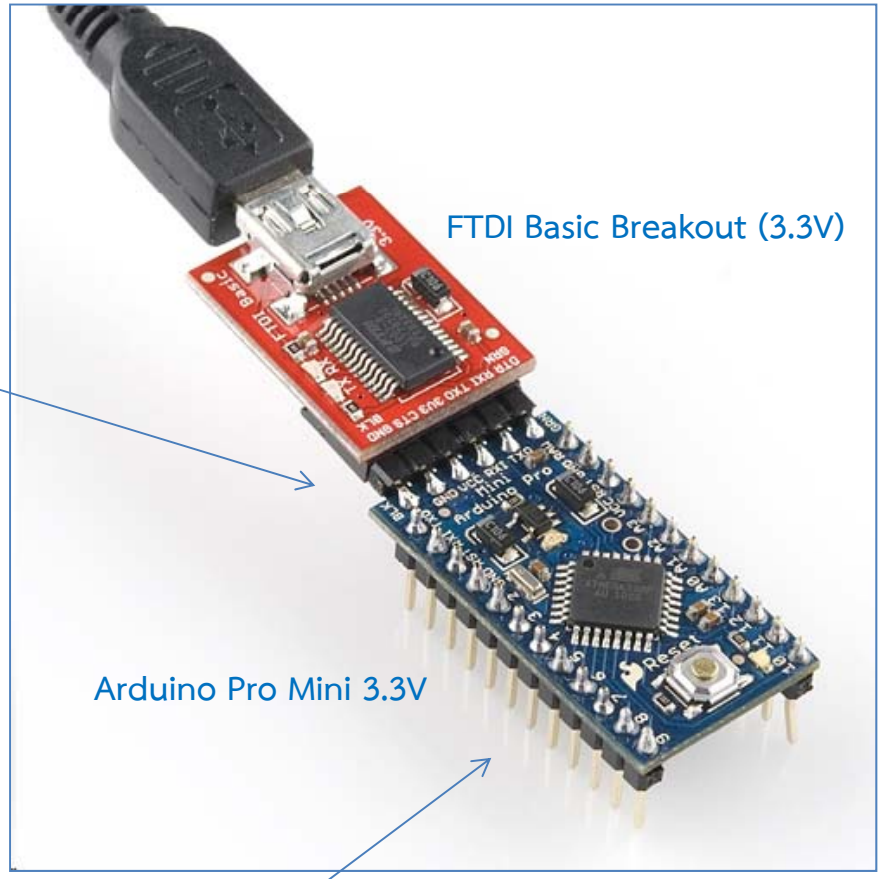


FTDI Basic Breakout 3.3V

FTDI Basic Breakout 5V

ที่มา: http://www.sparkfun.com/tutorials/244

http://www.sparkfun.com/products/9873

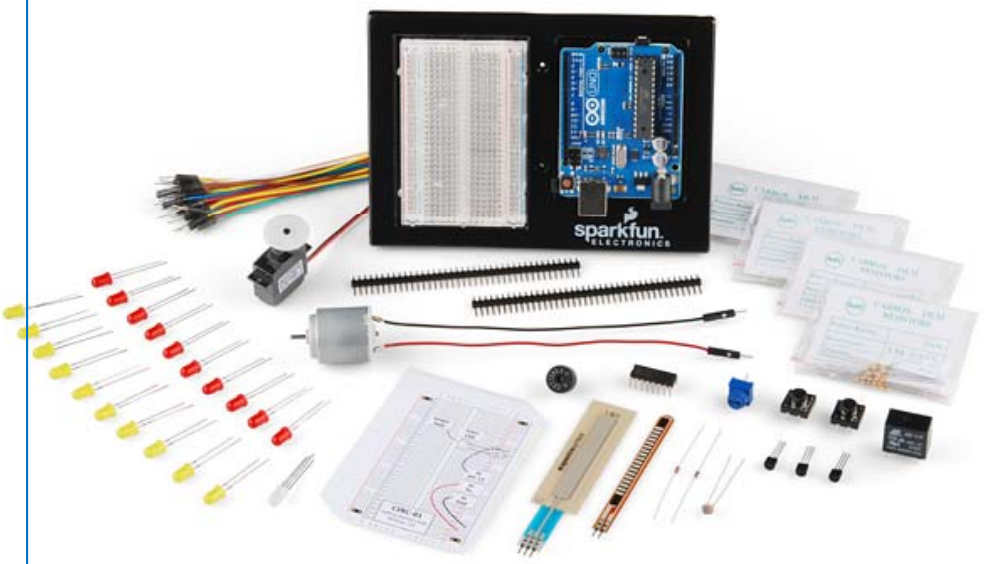


ตัวอย่างชุดคิท: SparkFun Inventor's Kit for Arduino

มีแต่บอร์ด Arduino อาจจะไม่มีประโยชน์ ถ้าไม่มีอุปกรณ์ที่สามารถนำมาต่อใช้งานร่วมกับบอร์ด Arduino

- Arduino Uno + USB cable
- Breadboard + Holder
- 74HC595 Shift Register
- 2N2222 Transistors
- 1N4148 Diodes
- DC Motor with wires
- Small Servo
- 5V Relay
- TMP36 Temp. Sensor

- Flex sensor
- Softpot
- Jumper Wires
- Photocell (LDR)
- Tri-color LED
- Red & Yellow LEDs
- 10K Trimpot
- Piezo Buzzer
- Big 12mm Buttons
- 330 and 10K
- Male Headers



ที่มา: http://www.sparkfun.com/products/10173

ตัวอย่างชุดคิท: Fritzing Starter Kit

Fritzing Starter Kit

ตัวอย่างบอร์ด Arduino ที่จำหน่ายในประเทศไทย

- ผู้นำเข้าบอร์ด Arduino จากต่างประเทศ เช่น
  - http://www.thaieasyelec.com/
  - http://www.warf.com/
- ผู้ผลิตบอร์ด Arduino Clone (Arduino-compatible) ในประเทศไทย เช่น
  - http://www.gravitechthai.com/ บริษัท Gravitech Thailand (บอร์ด Arduino Nano)
  - http://www.etteam.com/ บริษัท ETT เช่น บอร์ด ET-Easy168 STAMP)
  - http://www.inex.co.th บริษัท Innovative Experiment (INEX) เช่น บอร์ด POP-168, Unicon

ตัวอย่างบอร์ด ET-Easy168 Stamp

ET-Easy168 STAMP

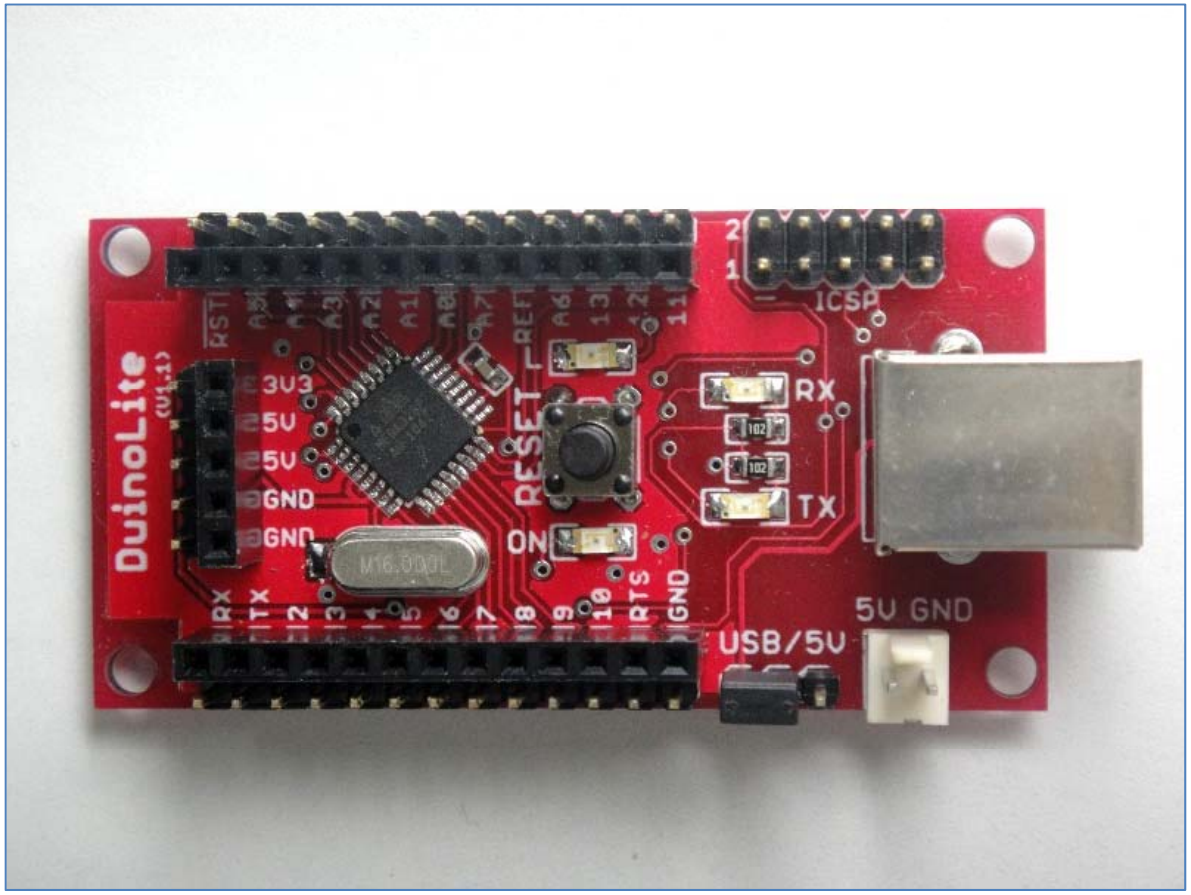
ที่มา: http://www.etteam.com/product2009/ET-AVR/ET-Easy168_STAMP.html

- สามารถเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB ได้โดยตรง
- ใช้สำหรับเชื่อมต่อที่แรงดัน 5V เท่านั้น

ESL DuinoLite v1.1 (Arduino Clone)

ATmega168 / 328P

5V / 16MHz

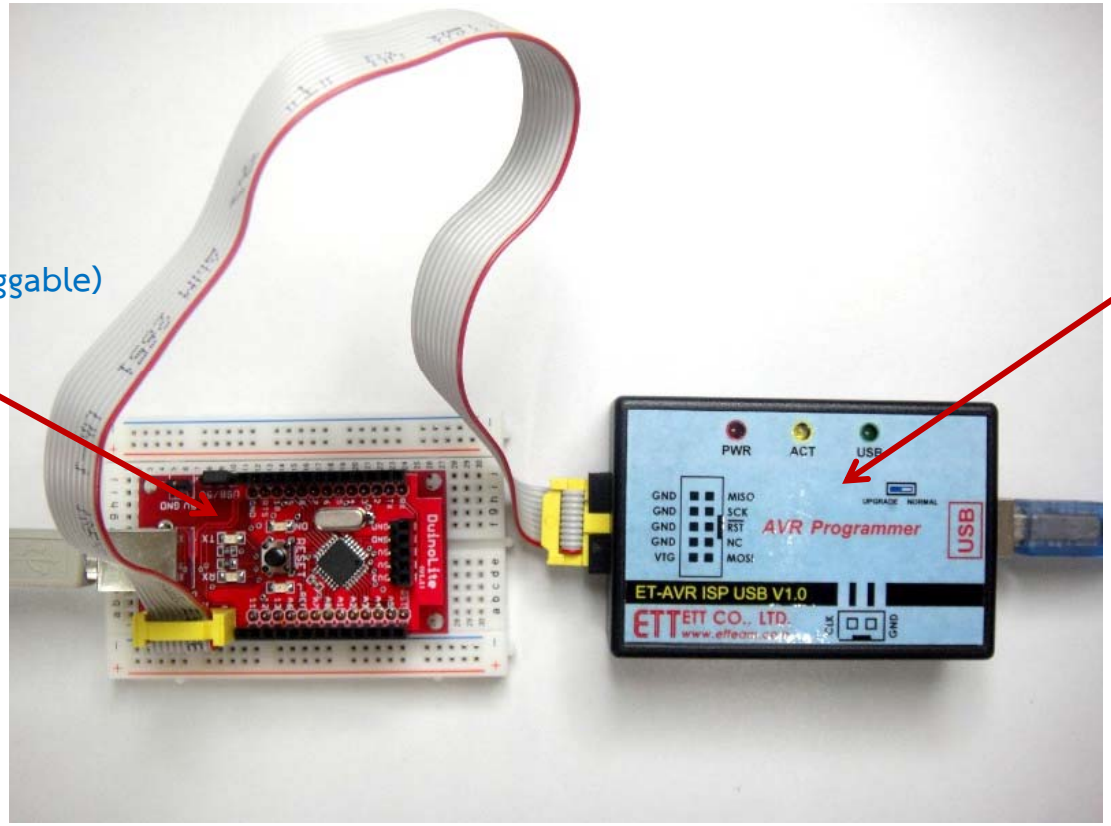


พัฒนาในปี ค.ศ. 2011 โดยห้องปฏิบัติการระบบสมองกลฝังตัว (ESL) เพื่อใช้เป็นสื่อการเรียนการสอน

อุปกรณ์สำหรับดาวน์โหลด Arduino Bootloader

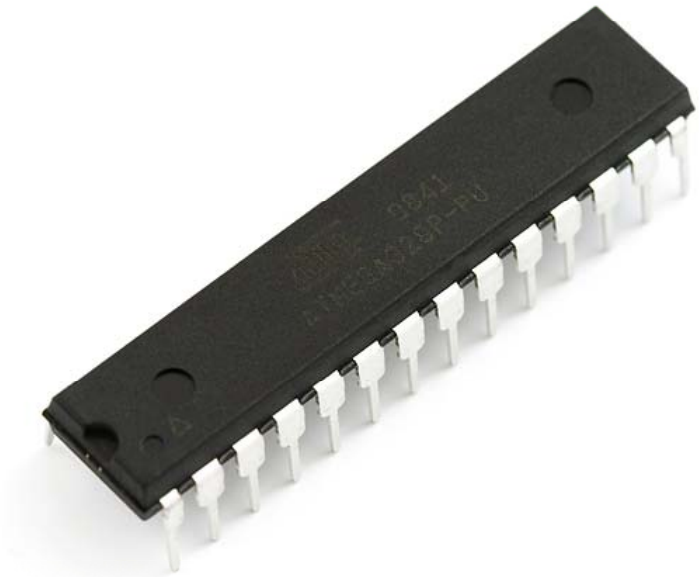
ESL DuinoLite
(Breadboard-Pluggable)

AVR ISP Programmer



บอร์ด Arduino จะใช้งานได้ จะต้องมีการใส่ Bootloader (Firmware) ลงในชิปไมโครคอนโทรลเลอร์ก่อน (ทำเพียงครั้งเดียว)
เมื่อมี Bootloader อยู่ภายในแล้ว จะสามารถใช้สาย USB เชื่อมต่อระหว่าง Arduino Board กับคอมพิวเตอร์ได้

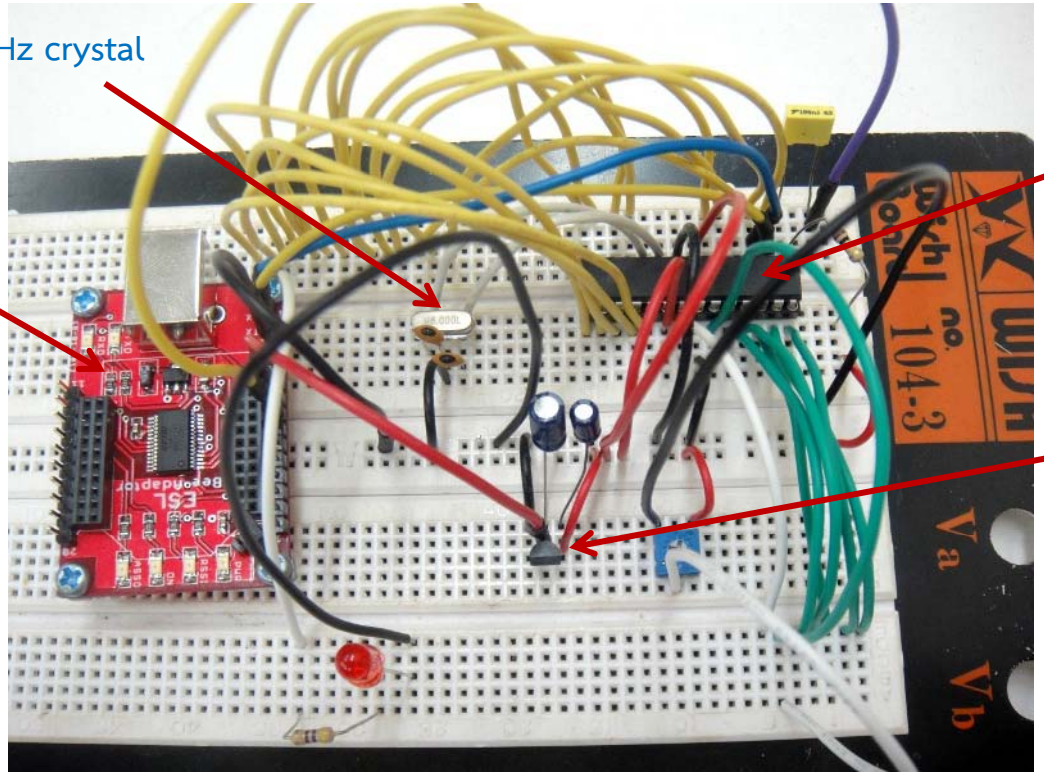
Atmel ATmega168/328P (28-pin DIP)



- Atmel ATMega168 (16KB Program Flash Memory)
- Atmel ATMega328P (32KB Program Flash Memory)
- 8-bit MCU
- 28-pin DIP package
- 23 I/O lines
- 1.8V to 5V operating voltage

ถ้าจะนำชิปนี้ไปต่อวงจรเพื่อใช้งานสำหรับ Arduino จะต้องมี Arduino Bootloader (Firmware) อยู่ภายใน (ต้องใช้ฮาร์ดแวร์ที่เรียกว่า AVR ISP Programmer ในการโปรแกรมข้อมูลลงในชิป)

ตัวอย่างการต่อวงจรบนเบรดบอร์ดเพื่อใช้งานสำหรับ Arduino



ตัวอย่างเครื่องมือสำหรับงานด้านอิเล็กทรอนิกส์

Tool Kit - Deluxe

ตัวอย่างชุดเครื่องมือ-วัสดุอุปกรณ์สำหรับงานอิเล็กทรอนิกส์พื้นฐาน สำหรับการบัดกรีวงจร

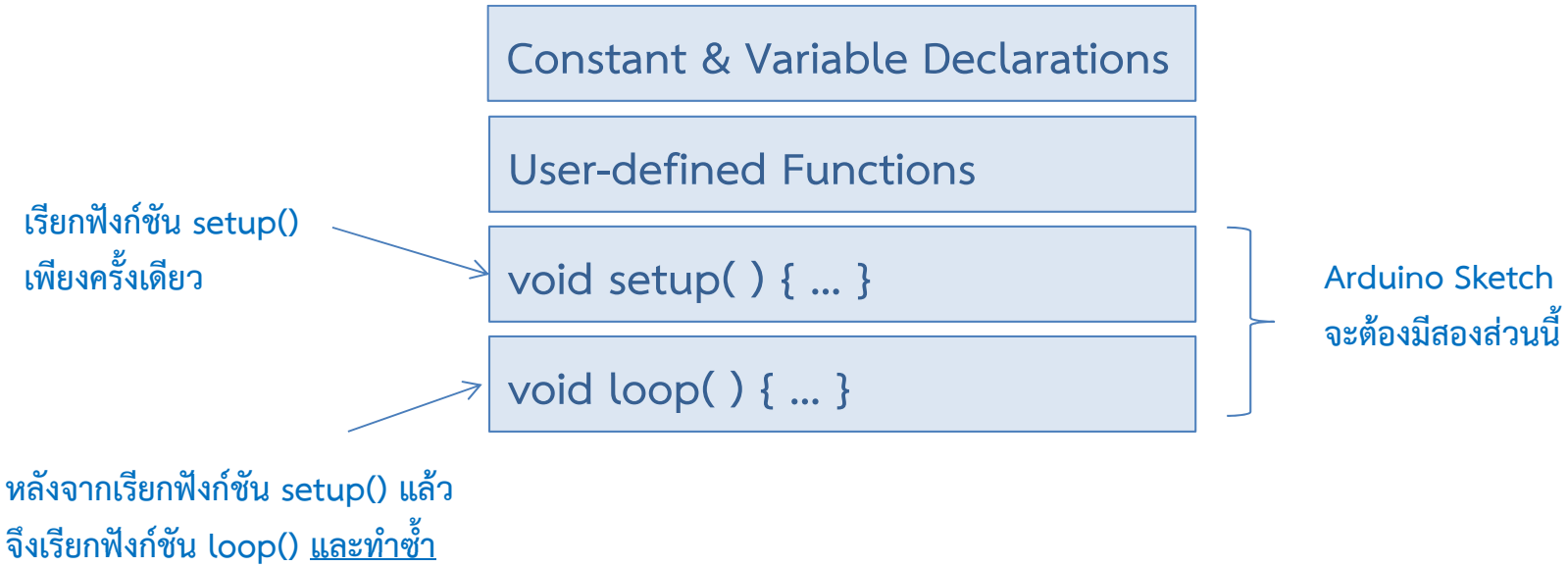
ราคา 7,800 บาท

ที่มา: http://www.warf.com/view.Tool_Kit_Deluxe-4843.html

โครงสร้างของโปรแกรมสำหรับ Arduino

ในการเขียนโค้ด (Sketch) สำหรับ Arduino สามารถแบ่งออกเป็น 4 ส่วนที่สำคัญคือ

1) การประกาศค่าคงที่และตัวแปรภายนอก
2) การสร้างฟังก์ชันขึ้นมาใช้งานใหม่ (เพื่อเรียกใช้งาน)
3) การสร้างฟังก์ชัน setup()
4) การสร้างฟังก์ชัน loop()



ตัวอย่างโค้ด: Blink

จาก Arduino IDE: เมนู File> Examples> Basics> Blink

```
/*
  Blink: Turns on an LED on for one second, then off for one second, repeatedly.
*/

void setup() {
   // initialize the digital pin as an output.
   // Pin 13 has an LED connected on most Arduino boards:
   pinMode( 13, OUTPUT );
}
void loop() {
   digitalWrite( 13, HIGH );         // set the LED on
   delay(1000);                      // wait for a second
   digitalWrite( 13, LOW );          // set the LED off
   delay(1000);                      // wait for a second
}
```

ตัวอย่างโค้ด: Blink (ดัดแปลงแก้ไข)

```
/*
  Blink: Turns on an LED on for one second, then off for one second, repeatedly.
*/
const byte LED_PIN = 13;
const int DELAY_MS = 1000;

void setup() {
   // initialize the digital pin as an output.
   // Pin 13 has an LED connected on most Arduino boards:
  pinMode( LED_PIN, OUTPUT );
}
void loop() {
   digitalWrite( LED_PIN, HIGH );    // set the LED on
   delay( DELAY_MS );                // wait for a second
   digitalWrite( LED_PIN, LOW );     // set the LED off
   delay( DELAY_MS );                // wait for a second
}
```

ชนิดข้อมูลพื้นฐานสำหรับ Arduino C/C++

byte	ใช้สำหรับข้อมูลที่เป็นเลขจำนวนเต็มได้ตั้งแต่ 0 ถึง 255
int	ใช้สำหรับข้อมูลที่เป็นเลขจำนวนเต็มได้ตั้งแต่ -32768 ถึง +32767
long	ใช้สำหรับข้อมูลที่เป็นเลขจำนวนเต็มได้ตั้งแต่ -2,147,483,648 ถึง +2,147,483,647
float	ใช้สำหรับข้อมูลที่เป็นเลขทศนิยม เป็นค่าที่เป็นบวกหรือลบในช่วงที่กว้างกว่าชนิดข้อมูลแบบ byte และ int และเป็นเลขทศนิยมได้ด้วย แต่มีความละเอียดเพียง 6-7 ตำแหน่งหลังจุดทศนิยมในเลขฐานสิบ
boolean	ใช้สำหรับข้อมูลที่เป็นค่าทางลอจิก true (จริง) หรือ false (เท็จ) เท่านั้น

คำสั่งพื้นฐานสำหรับ Arduino ที่ควรรู้ (1)

- **pinMode( )** ใช้กำหนดทิศทางสัญญาณ (I/O direction) ของขาดิจิทัล
  - http://arduino.cc/en/Reference/pinMode
- **digitalRead( )** ใช้อ่านค่าจากขาดิจิทัลที่ถูกกำหนดให้เป็นอินพุต
  - http://arduino.cc/en/Reference/digitalRead
- **digitalWrite( )** ใช้เขียนค่า (LOW หรือ HIGH) ให้ขาดิจิทัลที่ถูกกำหนดให้เป็นเอาต์พุต
  - http://arduino.cc/en/Reference/digitalWrite
- **analogWrite( )** ใช้สร้างสัญญาณ PWM เป็นเอาต์พุต
  - http://arduino.cc/en/Reference/analogWrite
- **analogRead( )** ใช้อ่านค่าจากขาแอนะล็อก-อินพุต
  - http://arduino.cc/en/Reference/analogRead
- **analogReference()** กำหนดระดับแรงดันอ้างอิงสำหรับการอ่านค่าจากขาแอนะล็อก-อินพุต
  - http://arduino.cc/en/Reference/analogReference

คำสั่งพื้นฐานสำหรับ Arduino ที่ควรรู้ (2)

- **delay( )** รอเวลาให้ผ่านไปตามระยะเวลาที่กำหนด (มิลลิวินาที) ก่อนที่จะทำขั้นตอนต่อไป
  - http://arduino.cc/en/Reference/delay
- **delayMicroseconds( )** รอเวลาให้ผ่านไปตามระยะเวลาที่กำหนด (ไมโครวินาที)
  - http://arduino.cc/en/Reference/delayMicroseconds
- **randomSeed( )** กำหนดค่าเริ่มต้นสำหรับการสร้างเลขแบบสุ่มเทียม
  - http://arduino.cc/en/Reference/randomSeed
- **random( )** ให้ค่าเป็นเลขสุ่มเทียม (Pseudo-Random Number)
  - http://arduino.cc/en/Reference/random
- **millis( )** บอกเวลาที่ผ่านไปในหน่วยเป็นมิลลิวินาที นับตั้งแต่โปรแกรมเริ่มต้นทำงาน
  - http://arduino.cc/en/Reference/millis

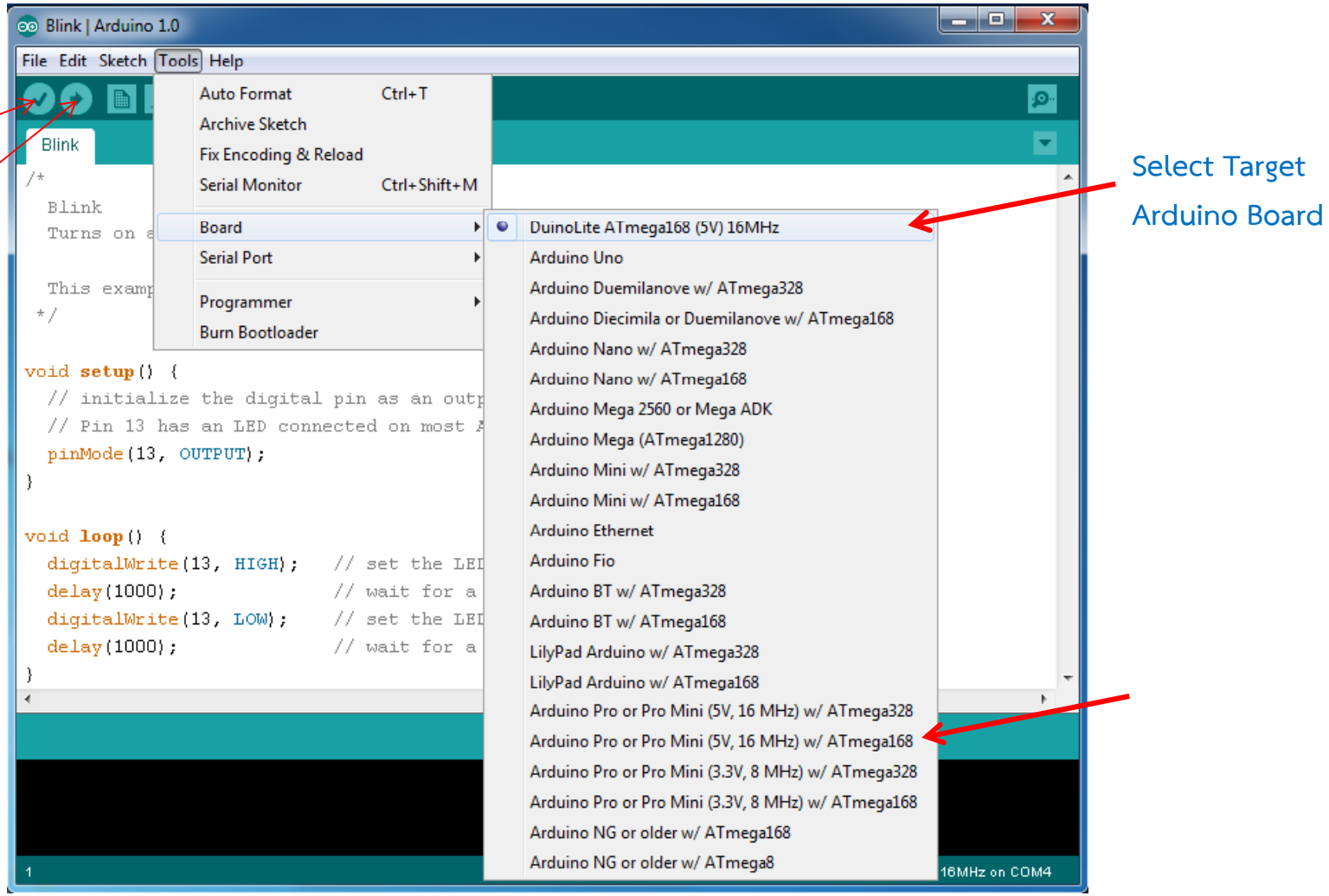
คำสั่งพื้นฐานสำหรับ Arduino ที่ควรรู้ (3)

- min( ) ให้ค่าน้อยที่สุดระหว่างตัวเลขสองค่าที่นำมาเปรียบเทียบกัน
  - http://arduino.cc/en/Reference/min
- max( ) ให้ค่าที่มากที่สุดระหว่างตัวเลขสองค่าที่นำมาเปรียบเทียบกัน
  - http://arduino.cc/en/Reference/max
- abs( ) ให้ค่าสัมบูรณ์ของตัวเลข
  - http://arduino.cc/en/Reference/abs
- constrain( ) ให้ค่าที่ไม่เกินช่วงที่กำหนด
  - http://arduino.cc/en/Reference/constrain
- map( ) ให้ค่าที่ได้จากการย่อหรือขยายเชิงเส้นตามช่วงที่กำหนด
  - http://arduino.cc/en/Reference/map

คำสั่งพื้นฐานสำหรับ Arduino ที่ควรรู้ (4)

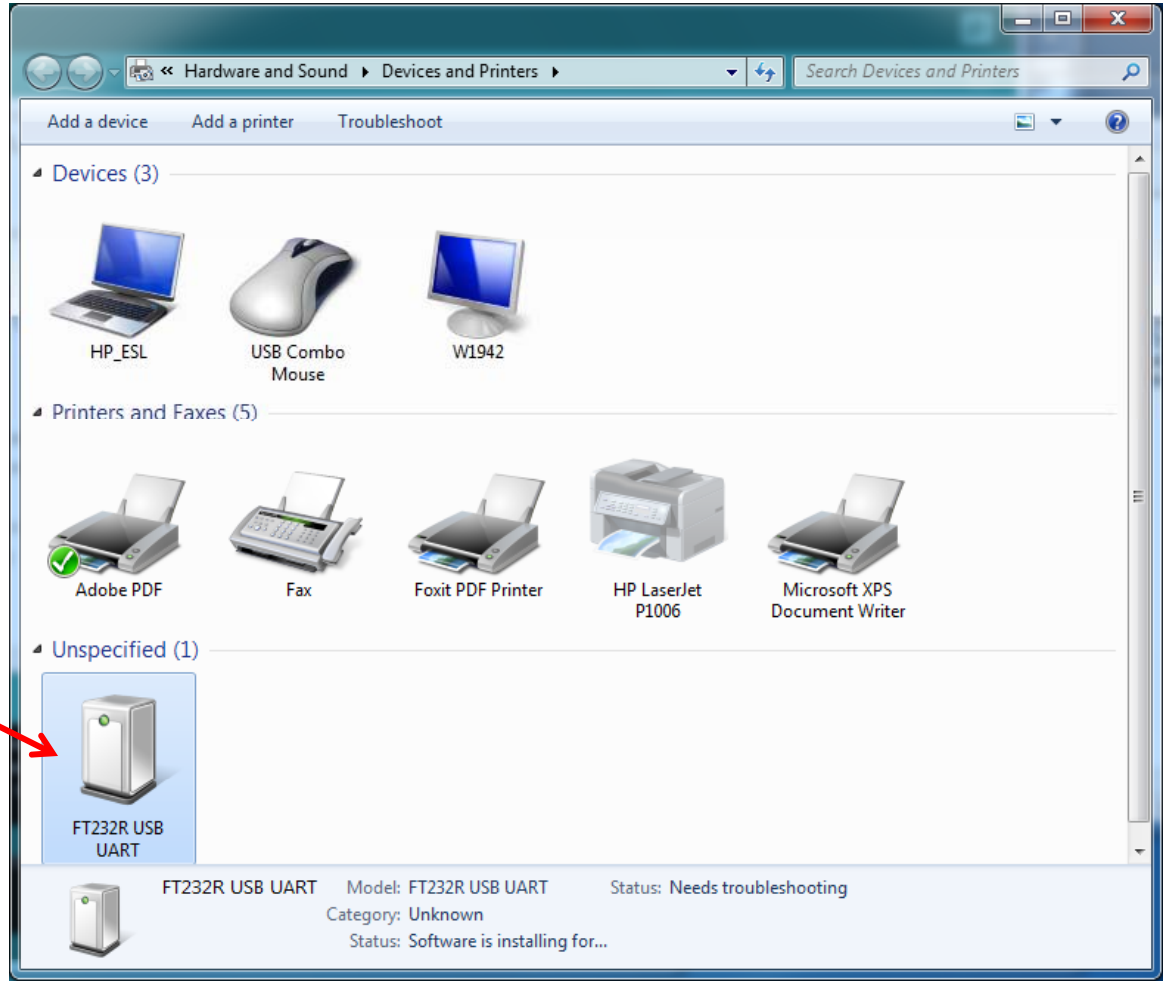
- **pow( )** คำนวณเลขยกกำลัง (Power)
  - http://arduino.cc/en/Reference/pow
- **sqrt( )** คำนวณค่ารากที่สอง (Square Root)
  - http://arduino.cc/en/Reference/sqrt
- **sin( )** คำนวณค่า Sine สำหรับมุมในหน่วยเรเดียน (radian)
  - http://arduino.cc/en/Reference/sin
- **cos( )** คำนวณค่า Cos สำหรับมุมในหน่วยเรเดียน (radian)
  - http://arduino.cc/en/Reference/cos
- **tan( )** คำนวณค่า Tangent สำหรับมุมในหน่วยเรเดียน (radian)
  - http://arduino.cc/en/Reference/tan

การเลือกบอร์ด Arduino เพื่อใช้งาน



การเชื่อมต่อกับบอร์ด Arduino ผ่าน USB-to-Serial

USB-to-Serial
โดยใช้ชิป FT232RL



สำหรับ Linux/Ubuntu
สามารถใช้งานได้เลย
ไม่ต้องติดตั้ง Driver
เพิ่มเติม

ขั้นตอนการใช้งานบอร์ด Arduino ร่วมกับวงจรบนเบรดบอร์ด

1. คอมไพล์โค้ด (Build)
2. เชื่อมต่อบอร์ด Arduino กับคอมพิวเตอร์ด้วยสาย USB
3. อัพโหลด (Upload) โปรแกรมที่ได้ไปยังบอร์ด Arduino
4. ต่อวงจรบนเบรดบอร์ด (ยังไม่ต้องต่อแรงดันไฟเลี้ยง +5V และ GND)
5. ตรวจสอบความถูกต้องของวงจรบนเบรดบอร์ด (ดูว่าต่อวงจรถูกต้องหรือไม่)
6. ต่อสายสัญญาณอินพุต-เอาต์พุตจากบอร์ด Arduino ไปยังจุดเชื่อมต่อในวงจรบนเบรดบอร์ด
7. ป้อนแรงดันไฟเลี้ยง +5V และ Gnd จากบอร์ด Arduino ไปยังแรงดันไฟเลี้ยงและ Gnd ของวงจรบนเบรดบอร์ด
8. ตรวจสอบการทำงานของวงจร (ดูว่าวงจรทำงานได้ตามที่ต้องการหรือไม่)

ตัวอย่างการสร้างคลาส C++ สำหรับ RGB LED

```
////////////////////////////////////////////////////////////////////////
#ifndef __RGB_LED_H
#define __RGB_LED_H

#include <inttypes.h>
#include "Arduino.h"
// This class implements a color controller for a single RGB LED
// that can change its color by adjust the duty cycles of
// three PWM signals used to control that LED.
namespace esl {
  class RGB_LED {
    public:
      // constructor
      RGB_LED( byte redPin, byte greenPin, byte bluePin,
               boolean inverse=false );
      // instance member methods
      void setColor( uint32_t color );  // set the RGB color
      void setColor( byte r, byte g, byte b ); // set the RGB color

    private:
      // private member method
      void update();
      // private member fields
      byte pins[3], rgb[3];
      boolean inverse; // inverse the PWM?
  }; // end class
} // end namespace

#endif // __RGB_LED_H
////////////////////////////////////////////////////////////////////////
```

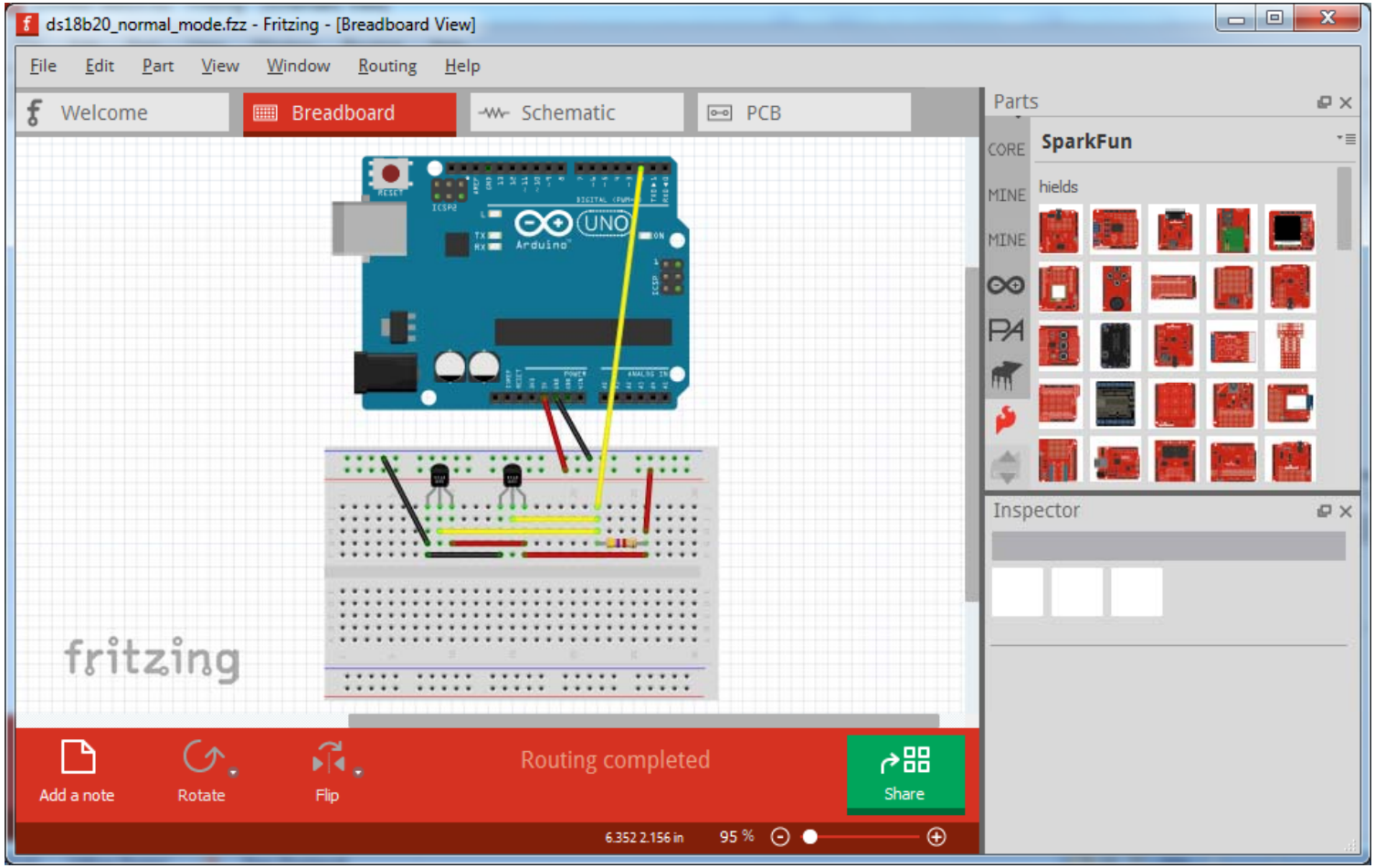
ตัวอย่างการสร้างคลาส C++ สำหรับ RGB LED

```
#include "RGB_LED.h"
using namespace esl;

RGB_LED::RGB_LED( byte redPin, byte greenPin, byte bluePin, boolean inverse ) {
  // save the Arduino pins for PWM output
  pins[0] = redPin;
  pins[1] = greenPin;
  pins[2] = bluePin;
  this->inverse = inverse;
  for ( int i=0; i < 3; i++ ) {
    pinMode( pins[i], OUTPUT ); // use this pin as output
  }
  setColor( 0x00, 0x00, 0x00 ); // set initial color to 0x00 (off)
}
void RGB_LED::setColor( byte r, byte g, byte b ) {
  rgb[0] = r; rgb[1] = g; rgb[2] = b;
  update();
}
void RGB_LED::setColor( uint32_t color ) {
  setColor( (color >> 16) & 0xff, (color >>  8) & 0xff, color & 0xff );
}
void RGB_LED::update() {
  byte duty_cycle;
  for ( uint8_t i=0; i < 3; i++ ) {
    duty_cycle = (inverse) ? (255-rgb[i]) : rgb[i];
    analogWrite( pins[i], duty_cycle );
  }
}
```

ตัวอย่างการออปเจคจากคลาส

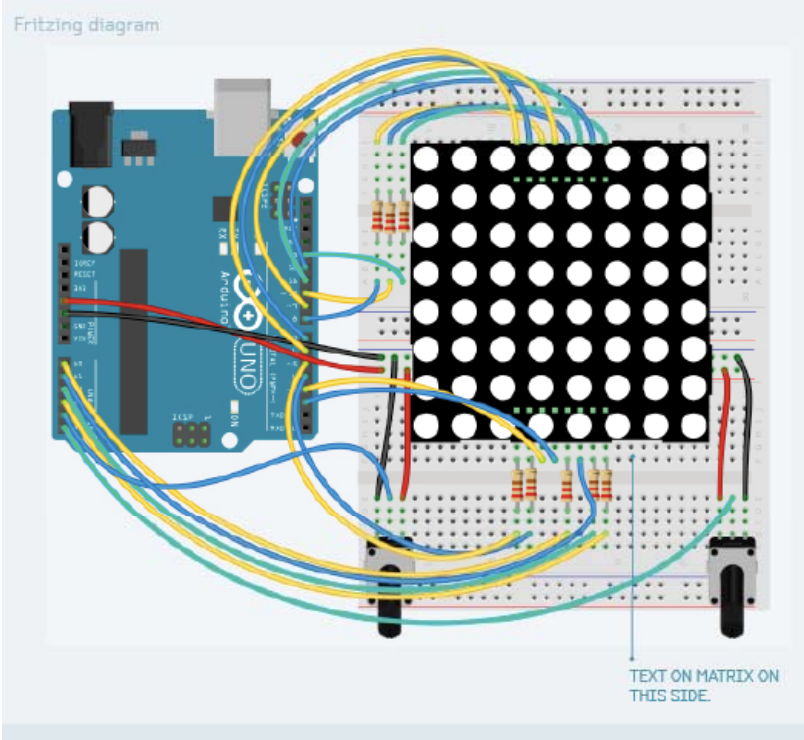
```
#include "RGB_LED.h"

using namespace esl;

// For Uno: PWM pins are 3, 5, 6, 9, 10, and 11.
RGB_LED led(5,9,10,false); // use RGB LED with a common cathode (-> no PWM inverting).
// RGB_LED led(5,9,10,true); // use RGB LED with a common anode (-> PWM inverting).

uint32_t colors[] = {
   0xff0000 /*red*/, 0x00ff00 /*green*/, 0x0000ff /*blue*/,
   0xffff00, 0x00ffff, 0xff00ff, 0 /*off*/ };

void setup() {
   // empty
}

void loop() {
  int num_colors = sizeof(colors) / sizeof(colors[0]);
  for ( int i=0; i < num_colors; i++ ) {
    led.setColor( colors[i] ); // change color
    delay(500);
  }
}
```

ตัวอย่างการใช้ซอฟต์แวร์ Fritzing สำหรับฝึกต่อวงจร

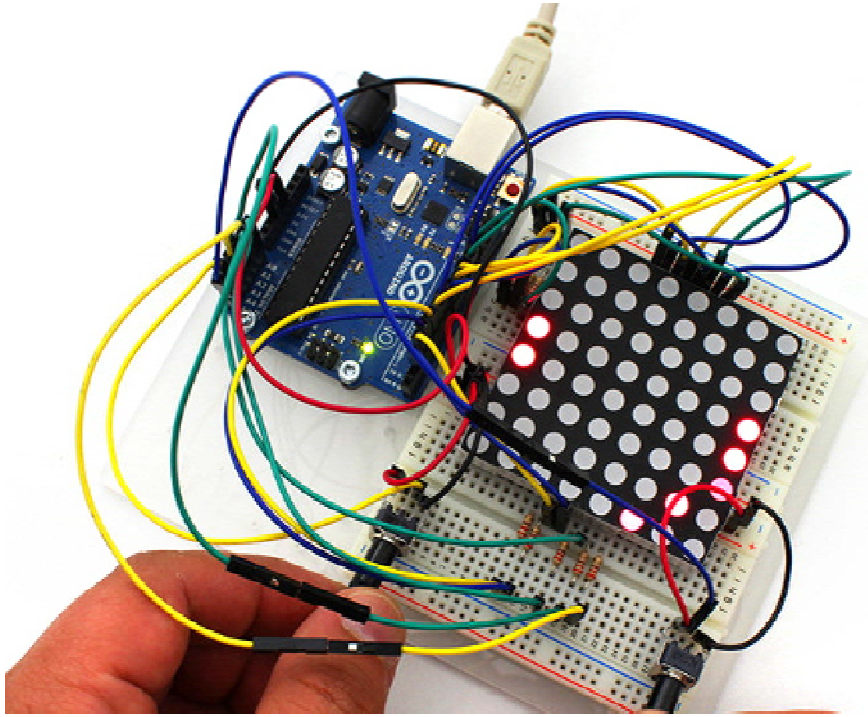


http://fritzing.org/

ตัวอย่างการใช้งาน Fritzing

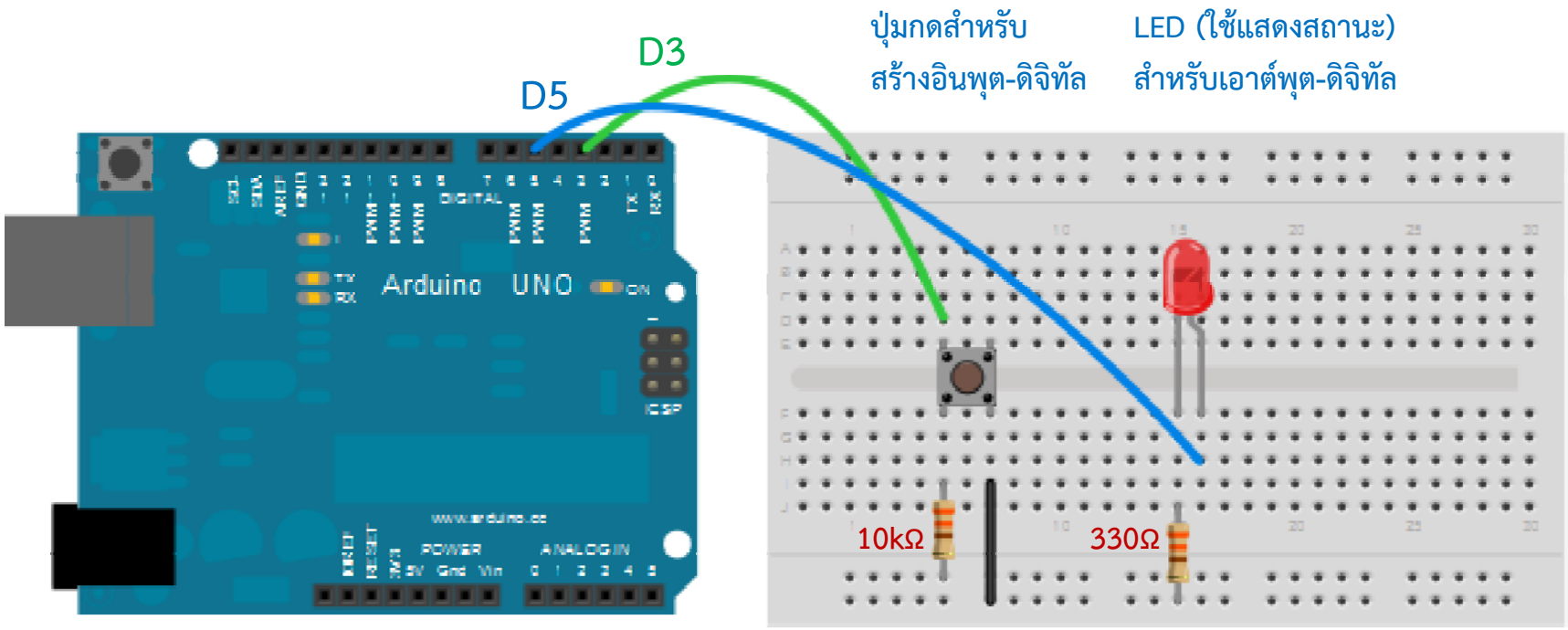


วาดแผนการต่อวงจรด้วย Fritzing



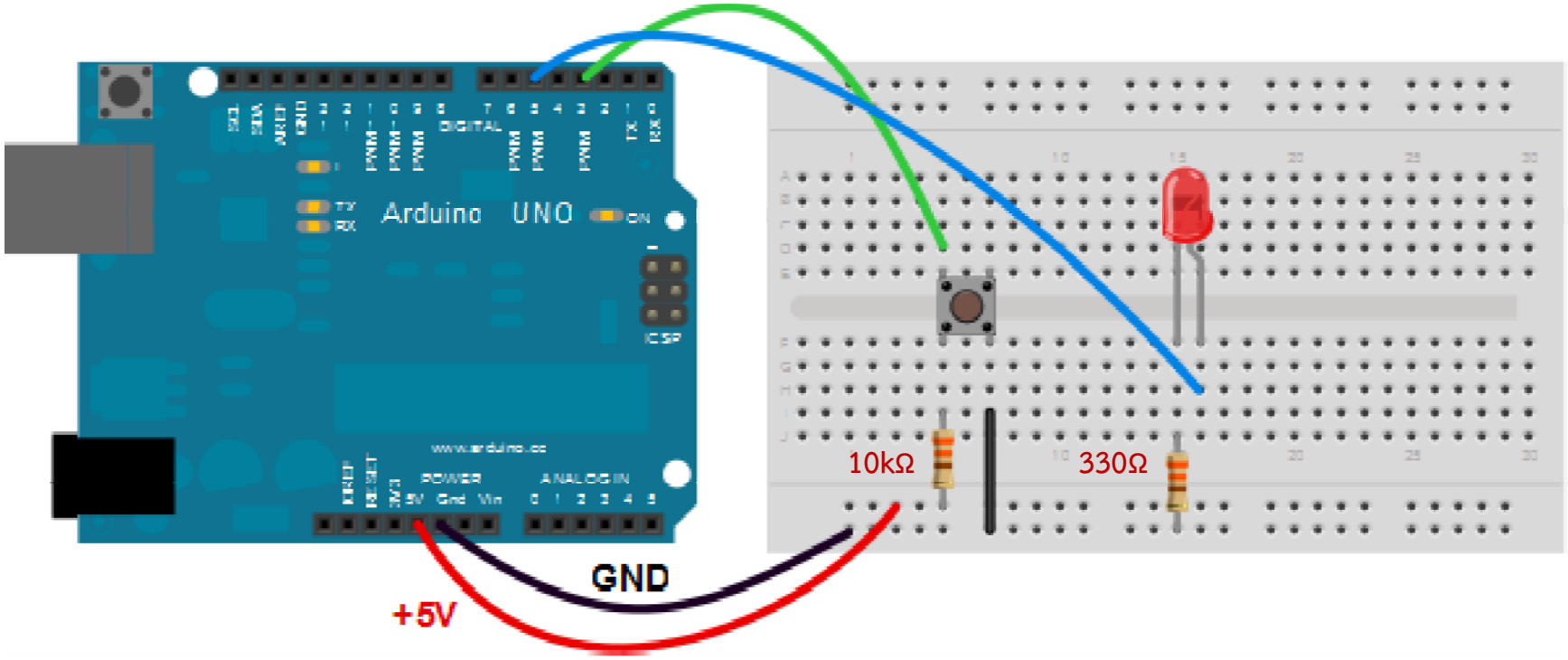
ต่อวงจรด้วยฮาร์ดแวร์จริง

การใช้งานบอร์ด Arduino ร่วมกับเบรดบอร์ด



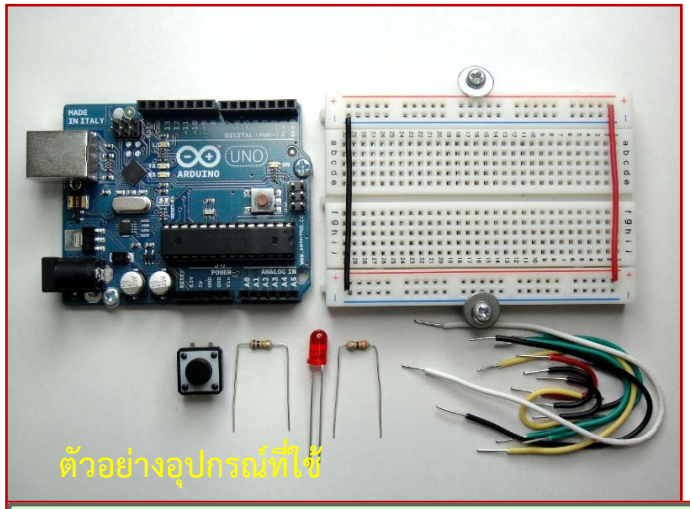
ต่อวงจรบนเบรดบอร์ด และเชื่อมต่อสายสัญญาณระหว่างบอร์ด Arduino และวงจรบนเบรดบอร์ด

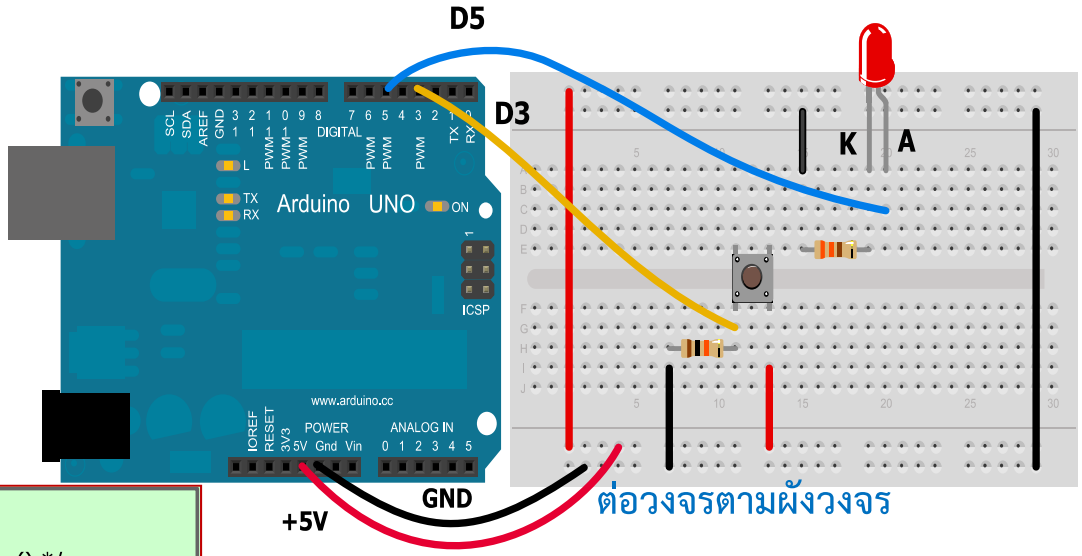
ตัวอย่างการต่อ Arduino กับวงจรบนเบรดบอร์ด



เมื่อตรวจสอบและมั่นใจแล้วว่า วงจรถูกต้อง จึงป้อนแรงดันไฟเลี้ยง +5V และ Gnd

ตัวอย่างโจทย์



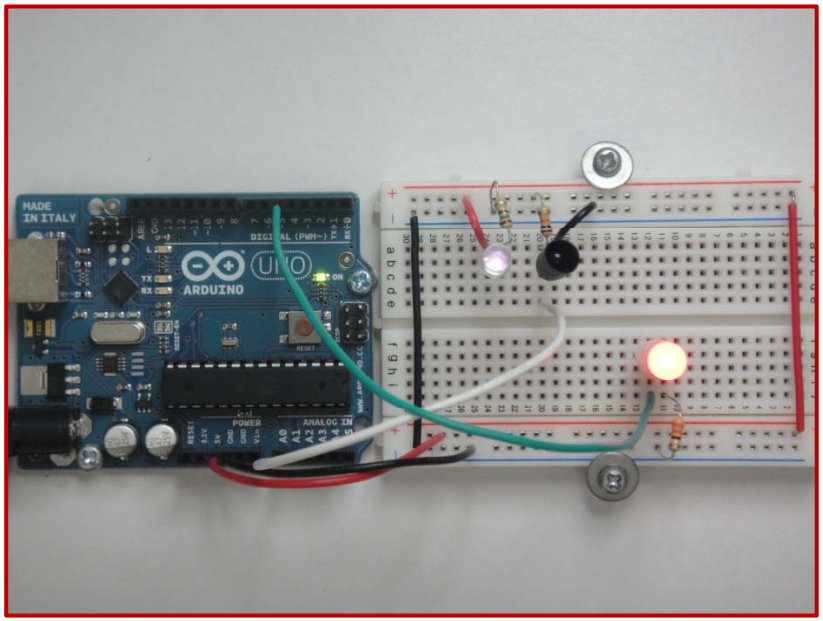



```
/* โค้ด Arduino: อ่านค่าจากขาดิจิทัล D3 โดยใช้คำสั่ง digitalRead()
แล้วนำค่าที่ได้ไปเขียนเป็นค่าเอาต์พุตที่ขา D5 โดยใช้คำสั่ง digitalWrite() */

const byte BUTTON_PIN = 3;      // ขาอินพุตที่ต่อกับวงจรปุ่มกด
const byte LED_PIN = 5;    // ขาเอาต์พุตที่ต่อกับวงจร LED
boolean value;  // ใช้สำหรับเก็บค่าที่อ่านได้จากขาดิจิทัล-อินพุต

void setup() {
  pinMode( LED_PIN, OUTPUT );   // ใช้ขา LED_PIN เป็นเอาต์พุต
}

void loop() {
  value = digitalRead( BUTTON_PIN );   // อ่านค่าจากขาดิจิทัล-อินพุต
  digitalWrite( LED_PIN, value );   // เขียนค่าไปยังขาดิจิทัล-เอาต์พุต
}
```

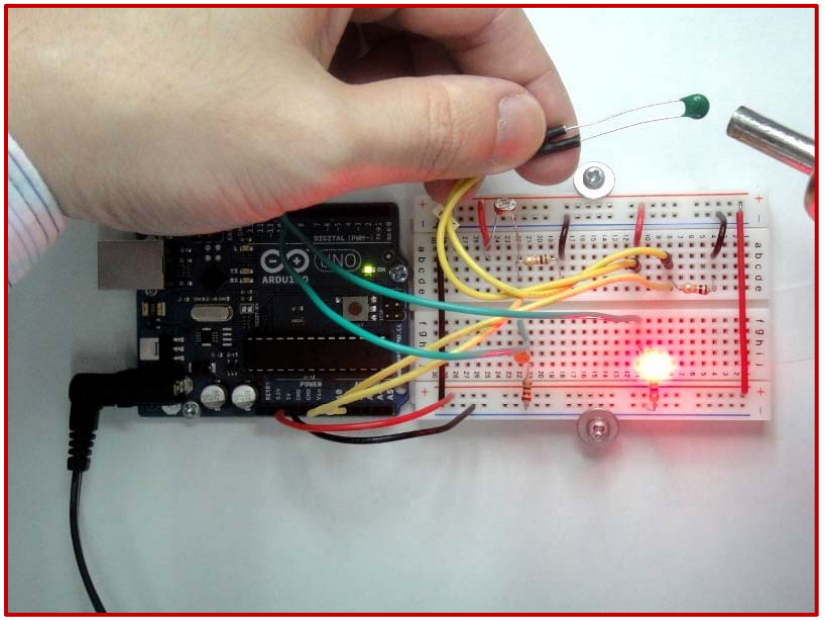
ใช้โค้ด Arduino ที่กำหนดให้ในการทดลอง

สังเกตพฤติกรรมการทำงานของวงจร
เช่น กดปุ่มแล้วสังเกตที่ LED

ตัวอย่างโจทย์การทดลอง

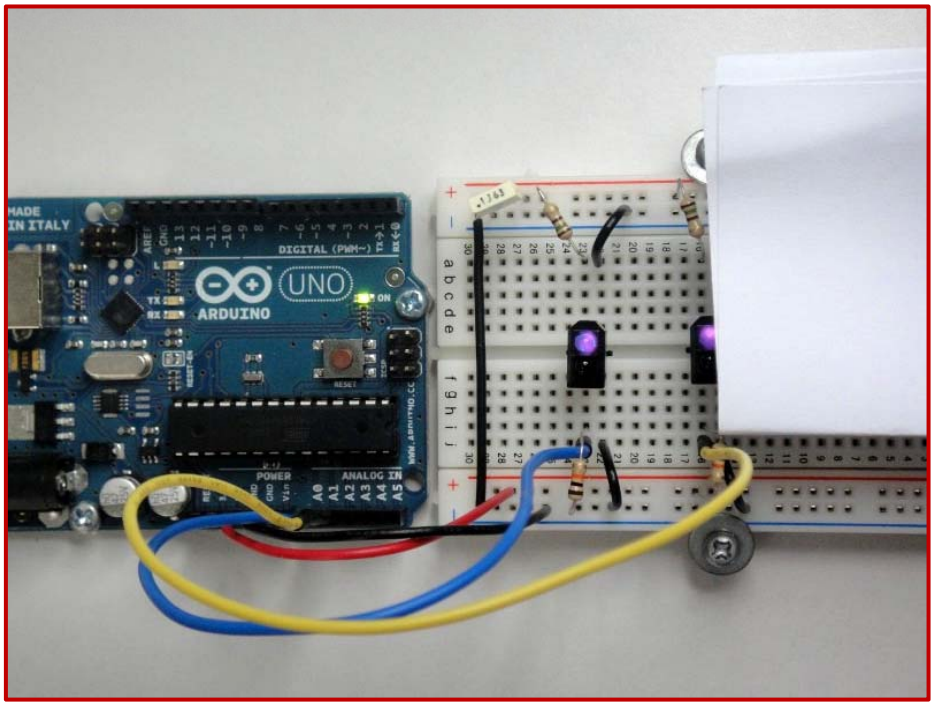


การตรวจจับวัตถุในระยะใกล้ด้วยแสงอินฟราเรด
(ต่อวงจร IR LED และ Phototransistor)
และแจ้งเตือนด้วย LED



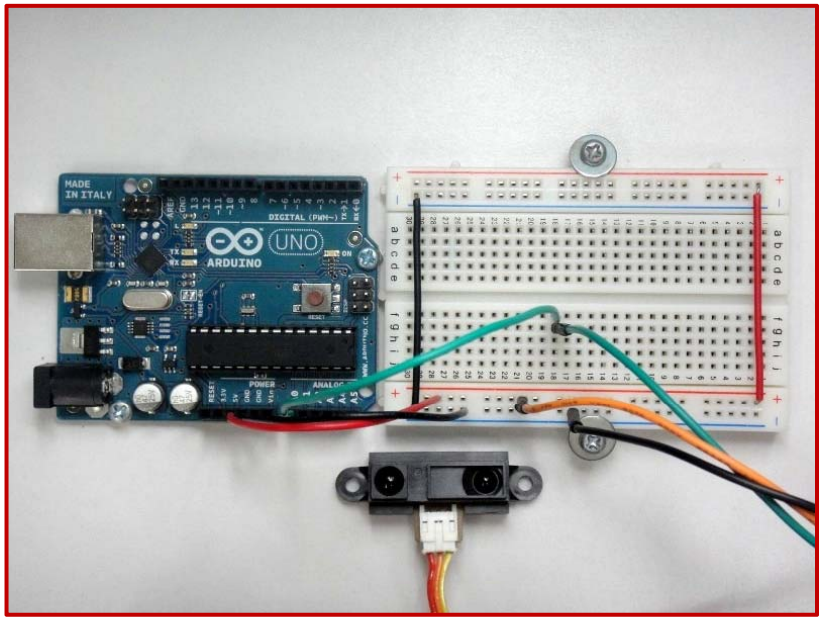
การวัดระดับและการเปลี่ยนของอุณหภูมิ
ด้วยเทอร์มิสเตอร์ (Thermistor) และ
แจ้งเตือนด้วย LED เมื่ออุณหภูมิสูงเกิน

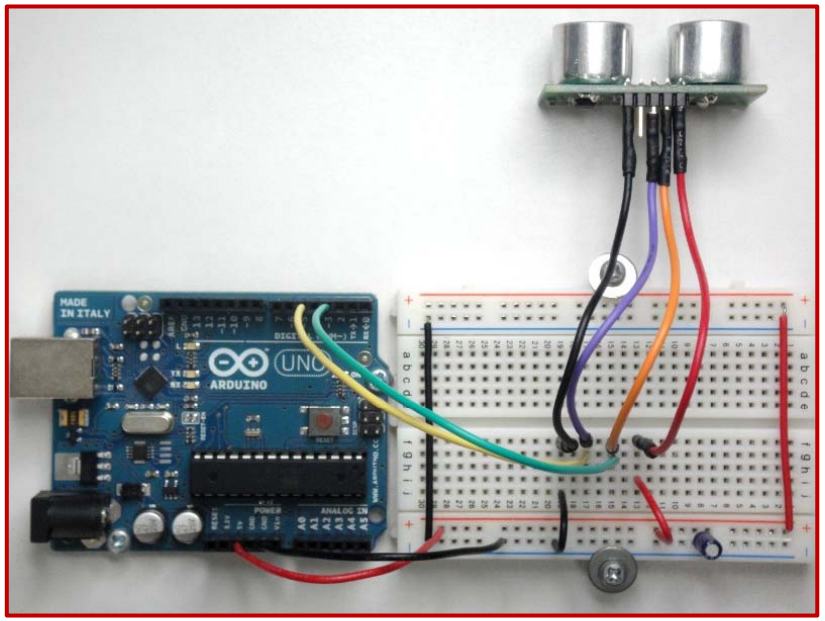
ตัวอย่างโจทย์การทดลอง



การแยกสีพื้นระหว่างขาวและดำ ด้วยเซนเซอร์แสงอินฟราเรด

ตัวอย่างโจทย์การทดลอง





การวัดระยะห่างด้วยเซนเซอร์ Ultrasonic / Infrared Range Finder
(อุปกรณ์ที่นิยมนำมาใช้กับหุ่นยนต์เคลื่อนที่เพื่อตรวจจับวัตถุกีดขวาง)

ตัวอย่างโจทย์การเขียนโปรแกรมและต่อวงจร

- จงเขียนโปรแกรม Arduino โดยใช้วงจรที่มีปุ่มกดและ LED เป็นอินพุตและเอาต์พุตตามลำดับ และให้มีพฤติกรรมการทำงานดังนี้
  - เมื่อเริ่มต้นทำงาน LED จะต้องไม่ติด (ไม่สว่าง)
  - เมื่อกดปุ่มแล้วปล่อยในแต่ละครั้ง จะทำให้ LED เปลี่ยนสถานะ (สลับสถานะ ติด-ดับ)
- จงเขียนโปรแกรม Arduino โดยใช้วงจรที่มีปุ่มกดและ LED เป็นอินพุตและเอาต์พุตตามลำดับ (ใช้วงจรเดิม) และมีพฤติกรรมการทำงานดังนี้
  - เมื่อเริ่มต้นทำงาน ให้ LED กระพริบ “LED Blinking” (ด้วยอัตราคงที่)
  - เมื่อกดปุ่มแล้วปล่อยในแต่ละครั้ง จะทำให้ LED เปลี่ยนสถานะ (สลับสถานะระหว่าง “กระพริบ” และ “ไม่กระพริบ/ดับ”)