Inworks electronics kits overview

Overview of all Inworks kits and components, including suggested usage and reference to learning materials and sample code.

Last update:

Jenny Filipetti

14 January 2020

Arduino digital sandbox

This kit is designed to familiarize users with the Arduino platform and language without requiring any electronics or wiring knowledge. It is used in the Inworks Basic Arduino workshop.

Sample code

- Inworks Arduino Digital Sandbox companion code
- https://learn.sparkfun.com/tutorials/digital-sandbox-arduino-companion

Required libraries

None for built-in components.

Motor control

This kit includes a range of different kinds of motors that might be used in student projects. Motor control is a more advanced Arduino topic because usage frequently requires high current draw, load-bearing calculations, the need for external bridged power, and/or the use of specific motor control drivers. Users can permanently damage Arduino devices if they do not attend to these concerns!

Servo motors are a good gateway motor for learning because low-voltage servo motors under minimal load can often be powered directly from the Arduino. This is not the case for most stepper or DC motors.

How to use a servo motor

Servo motors are used for controlling movement. Most servos including the ones in this kit have a range of motion of only 180°; however “continuous rotation” servo motors are available, usually at a higher cost. Users set a specific desired angle to which the motor will move. Servos run significantly faster than stepper motors. They can be connected to a gearbox for increased torque. A servo will hold its current position only if constantly actively instructed to do so.

Hookup guide with Arduino: https://learn.adafruit.com/adafruit-arduino-lesson-14-servo-motors

Required libraries: Servo.h (Arduino built-in)

How to use a stepper motor

Stepper motors are used for controlling movement in 360°. Users set a desired number of steps for the motor to complete. Each motor is manufactured to permit a certain number of steps. They are highly accurate but typically move much more slowly than servos. They are stable in resting position.

Hookup guide with Arduino: See ELEGOO guide. The stepper motor requires the included stepper motor control driver to function. Alternative stepper wiring circuits and drivers do exist.

Required libraries: Stepper.h (Arduino built-in). Other specialized libraries also exist to simplify specific applications.

How to wire a DC or vibration motor

DC motors are continuous rotation motors, like those used in laptop or handheld spray fans. They can be purchased in a variety of speeds and maximum torque values to fit application needs. A vibration motor is just a small DC motor built with an unbalanced load.

Hookup guide with Arduino: https://learn.adafruit.com/adafruit-arduino-lesson-13-dc-motors

Required libraries: None

Adafruit Feather kit

This kit is designed for novice users who are trying to create a more specialized applications particularly in the field of IoT or connected devices. It includes an ESP32 board (wifi and Bluetooth) in addition to multiple shields which stack directly on top of the board to add functionality without the user needing to have a knowledge of electronics or wiring. Installation is non-trivial so it is recommended that novice users simply use the Inworks laptops.

About the ESP32 Huzzah! Feather board

https://learn.adafruit.com/adafruit-huzzah32-esp32-feather

Required libraries: None to turn on but depends on usage. See ESP32 section and FeatherWing tutorials in chart below for additional specifics.

Additional notes: Requires Arduino ESP32 compatability and the Adafruit ESP32 Feather board, both of which are already installed on Inworks laptops. Also requires SiLabs CP2104 Driver, linked in the tutorial.

About the shields ("wings" in Adafruit Feather-speak)

Name	Function	Hookup guide with Arduino
GPS	GPS coordinate tracking	https://learn.adafruit.com/adafruit-ultimate-gps-featherwing
NeoPixel 4x8	Displays a grid of individually adjustable RGB LED lights	https://learn.adafruit.com/adafruit-neopixel-featherwing
Music Maker	Plays audio files stored on a microSD card. Requires microSD card and speakers for full use!	https://learn.adafruit.com/adafruit-music-maker-featherwing/overview
Mini color TFT	User interface: touchscreen display and joystick	https://learn.adafruit.com/adafruit-mini-tft-featherwing/overview
Servo/PWM	Control of up to eight motors	https://learn.adafruit.com/adafruit-8-channel-pwm-or-servo-featherwing

Proximity and orientation

For hookup guide for all below components, see ELEGOO guide unless otherwise specified. No libraries are required unless otherwise specified.

Orientation

Digital tilt sensor + Tilt-switch detector

Designed to send a different signal across the signal line when the module is tilted versus at rest (relative to plane of gravity). The “digital tilt sensor” is the bare component; the “tilt-switch detector” includes a 10k resistor and headers on the breakout board.

Additional hookup guide: https://learn.adafruit.com/tilt-sensor

3-axis accelerometer

Required libraries: Wire.h (Arduino built-in, also included in ELEGOO sample code). Advise students to not be alarmed by the comments or how the library is functioning! It uses bit-shifting in order to obtain the different values which is a lower-level computing tactic not many students are familiar with. However the sample code stores the values in several integer variables, which students can then easily work with elsewhere in their code.

Presence and proximity

PIR sensor

A PIR sensor is used to detect motion of any infrared-emitting bodies (e.g. humans, animals). This sensor is adjustable for distance (3-7 meters), time delay, and trigger modes (single/repeating). Refer to ELEGOO tutorial or datasheet for details. However, students should be advised that PIR sensors can be finicky! In addition, the sensor requires a startup time of at least about 60 seconds during which it is prone to give false output. The PIR sensor as provided has a viewing angle of about 110°; however if a project requires a more localized viewing range, the Fresnel lens can be carefully temporarily removed and “blinders” placed around the active part of the sensor.

Counter-indications: UV light (e.g. sunlight) is known to interfere with infrared sensors!

Additional hookup guide: https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/overview

Ultrasonic detector

An ultrasonic sensor detects the distance of whatever environmental objects are in its range of focus. It does so by emitting an ultrasonic pulse and calculating the distance based on how long it takes for that pulse to bounce back. They possess a fixed range of detection and cannot be selectively focused.

Infrared avoidance module

An infrared sender and receiver pair built into a single breakout board so to detect obstacles at close range. It does so by emitting an infrared pulse and assuming there is an obstacle present if the receiver senses the reflected rays. The potentiometer included on this breakout board permits for adjusting the range up to a maximum of approximately 7cm.

Redbot bumper

Designed for use in simple robotics applications to detect obstacles at close range. This component is nothing more than a clever arrangement of wire, electricity, and metal hardware, mimicking the behaviour of a switch

Hookup guide with Arduino: https://learn.parallax.com/tutorials/robot/shield-bot/robotics-board-education-shield-arduino/chapter-5-tactile-navigation-9

Touch and disturbance

Shock detector

Detects vibrations and shaking, including those transmitted through the air. Generates a momentary signal while actively detecting vibration.

Tap detector

Detects knocks or taps transmitted through solid materials. Generates a momentary signal while actively detecting taps.

Other

RTC (Real Time Clock)

Keeps accurate track of the time.

Line-tracking module

Designed for use in simple robotics applications for obstacle detection or to support line-following behaviour for small detection distances (approximately 1cm). The potentiometer included on this breakout board permits for sensitivity adjustments.

User interface

This kit includes a variety of user interface design elements that might be used in student projects.

Screens

2-line LCD screen

For character text display.

Hookup guide with Arduino: See ELEGOO guide.

Required libraries: LiquidCrystal (Arduino built-in)

TFT touchscreen

For flexible output display and touch input. Note: this is a more advanced component. While the example sketches can be run by a novice, most desired interactions require non-trivial understanding of computing concepts such as spatial layout, timing, and variables.

Hookup guide with Arduino: https://learn.adafruit.com/adafruit-2-dot-8-color-tft-touchscreen-breakout-v2

Required libraries: Depends on usage. See hookup guide.

Additional notes: The ELEGOO guide also contains sample code and two libraries (Elegoo_GFX and Elegoo_TFTLCD). However this hardware uses the same ILI9341 IC as the TFT board that Adafruit sells and I would recommend their libraries as easier to work with.

Tactile controls

For hookup guide for all below components, see ELEGOO guide. No libraries are required unless otherwise specified.

Number pad

A number pad is based on capacitive sensing and can be custom-made in this way. This module is designed for plug-and-play use.

Required libraries: Requires Keypad.h for plug-and-play use. Library included in ELEGOO guide.

Joystick

A joystick is a relatively simple component designed to permit input in both the X and Y directions.

Rotary encoder

A rotary encoder returns the angular position of its shaft.

Potentiometer

Buttons and switches

These simple components differ moderately in functionality, wiring, and/or user experience, but all can be used in Arduino using the simple digitalRead() command. They can also be useful as electrical hardware without Arduino, when wired so as to interrupt or connect the flow of electricity from one or more power sources.

Tutorial on buttons and switches: https://learn.sparkfun.com/tutorials/switch-basics/all

Push button

Advised for momentary push input. A push button appears to have four legs, but it is actually composed of effectively just two, each one of which juts out in an opposite direction from the button as though it was one individual wire passing from one side through the button out through the other. When the push button is not pressed, there is no connection across the two independent legs. When closed, all of the legs are connected.

Required libraries: None

Hookup guide with Arduino: https://www.arduino.cc/en/tutorial/button

Slide switch + Toggle switch

Advised for persistent switch input. The switches in this kit are all SPDT switch. Switches can be categorized as one of the following:

- SPST (single-pole, single-throw) with two legs. Serves to complete or not complete a circuit.
- SPDT switch (single-pole, double-throw) with three legs
- DPDT (double-pole, double-throw) with six legs
- Rarely, switches with more legs and poles/throws.

Required libraries: None

Hookup guide with Arduino: Depends on component. For SPST, intercept within the flow of electricity. For SPDT, middle leg to Arduino input in and outer legs to ground and power respectively.

Buzzer pack

A basic kit of buzzers to generate simple audio output with Arduino, for example alert tones.

An active buzzer requires only DC voltage to generate a tone, while a passive buzzer requires a signal, which can be generated using the tone() command in Arduino.

Required libraries: None.

Passive buzzer

Passive buzzers require the built-in tone() command.

Code reference for tone(): https://www.arduino.cc/reference/en/language/functions/advanced-io/tone

Active buzzer (small and large)

An active buzzer can simply be turned on or off with an Arduino digital or PWM pin.

Adjusting the volume of an active buzzer without an Arduino:
http://www.learningaboutelectronics.com/Articles/How-to-vary-the-volume-of-a-buzzer

Environmental sensing

For hookup guide for all below components, see ELEGOO guide unless otherwise specified. No libraries are required unless otherwise specified.

Ambient conditions

Photoresistor

Photoresistors or light-dependent resistors (LDR) or photocells permit the measurement of ambient light levels.

Hookup guide with Arduino: https://learn.adafruit.com/photocells/overview

Required libraries: None

TMP36 temperature sensor +

This kit includes both the common TMP36 bare component (standalone) and a DS18B20 chip sensor breakout board. Both are used to measure ambient temperature.

Hookup guide with Arduino: https://learn.adafruit.com/tmp36-temperature-sensor/overview

Required libraries: None

Digital temperature sensor (DS18B20 breakout)

This kit includes both the common TMP36 bare component (standalone) and a DS18B20 chip sensor breakout board. Both are used to measure ambient temperature.

Hookup guide with Arduino: https://learn.adafruit.com/tmp36-temperature-sensor/overview

Required libraries: OneWire.h and DallasTemperature.h libraries, included in ELEGOO guide

DHT22 Humidity sensor + Temperature and humidity sensor (DHT11 breakout)

This kit includes both the DHT22 model sensor (standalone) and a DHT11 model sensor breakout board. Both are used to measure ambient temperature and humidity. The DHT22 has a wider operating range (-40 to 125 C and 0 to 100% humidity, compared to 0-50C and 20-80% humidity) and narrower margin of error (0.5 C and 2-5%, compared to 2C and 5%). The DHT11 model samples twice as quickly and has a plug-and-play breakout board form factor which may be helpful for quick prototyping.

Hookup guide with Arduino: https://learn.adafruit.com/dht

Required libraries: DHT.h by Adafruit recommended for use with either sensor (linked in above tutorial). An alternative DHT library included in ELEGOO guide.

Sound

Big sound detector + Small sound detector

Both breakout boards output the presence and general magnitude of detected sound volume via digital and analog pins. Students should be advised these sensors detect approximate volume only (which is impacted by other attributes of sound). As a result, it is useful out of the box for a clap or knock detector but less so for music-related applications (e.g. LEDs that illuminate in time with music) unless there is a very clear distinction in volume during desired "on" versus "off" behaviour. This behaviour can of course be modulated via code for more nuanced results using the analog pin output.

Required libraries: None

Other

Flame detector

Detects infrared light of a particular wavelength range, typically corresponding to flame. This breakout generates both an analog and a digital result, accessible via their respective pins.

Soil moisture sensor

This sensor is intended to be placed with the legs inside the soil whose moisture level is to be measured.

Linear hall effect magnetic sensor

Used for detecting the presence and general magnitude of a magnetic field. Technically a hall effect sensor requires that the magnetic field be perpendicular to the sensor although this should not make a difference in basic applications such as detecting proximity to a magnet. This breakout generates both an analog and a digital result, accessible via their respective pins.

Magnetic spring

Also used for detecting the presence of a magnet or magnetic field. Technically a reed switch requires that the magnetic field be parallel to the sensor's wires (in the small glass capsule) although this should not make a difference in basic applications such as detecting proximity to a magnet.

Flex and pressure sensing

Force sensitive resistors (100g to 10kg)

These sensors measure the presence (and rough magnitude) of applied pressure. The models in the Inworks kits are breadboard-friendly. Do **not** solder! When no pressure is applied, resistance is larger than 1MΩ.

Hookup guide with Arduino: https://learn.sparkfun.com/tutorials/force-sensitive-resistor-hookup-guide

Flex sensors

The resistance of this sensor increases as its form is bent. It is often used in wearable applications and soft robotics. For best results, securely mount the base and bottom portion and only allow the actual flex sensor to flex, not the base.

Hookup guide with Arduino: https://learn.sparkfun.com/tutorials/flex-sensor-hookup-guide

ESP32 IoT kit

This kit was used in the IWKS 4120 IoT class from 2017-2019 so that every student possessed a connected device with which they can experiment and prototype.

Warning: Unlike the Arduino Uno, the ESP32 is a 3.3V logic board! Users cannot connect inputs or outputs designed for 5V power and/or logic without a step-up power block (or separate power supply) to meet power needs ***and/or*** a logic-level converter to interface with the device pins to match logic level.

Preparing your personal computer to use Arduino with the ESP32

The ESP32 is not an Arduino device; however we can add ESP32 support to the Arduino IDE in order to use the Arduino programming language to program our ESP32 the Thing. The Inworks laptops are already set up to enable writing and uploading Arduino code to the ESP32 but students using a personal device must follow the steps described in the hookup guide.

ESP32 Thing Hookup Guide: https://learn.sparkfun.com/tutorials/esp32-thing-hookup-guide

Testing device functionality

See “Getting started with the ESP32 Thing” and “ESP32 Getting Started” companion code.

Recommended libraries

The following libraries are recommended as needed for the specified functionality to take advantage of ESP32 core functionality and connected device applications.

- WiFi.h (Arduino built-in)
- HTTPClient.h (ESP32 built-in)
- ArduinoJson.h by Benoit Blanchon to parse JSON attained from web APIs
- ESP32 BLE Arduino by Neil Kolban for a variety of Bluetooth functionality
- PubSubClient by Nick O’Leary for MQTT messaging