Getting Physical with Arduino and Galilieo

# Introduction

The goal in this project is to build something using Arduino and Galileo that allows the users to interact with the real world. This project utilizes some of the unique capabilities of Arduino and Galileo and requires programs that run on the PC/server side.

The **Intel Galileo Gen 2** is a powerful platform designed for prototyping high performance physical computing, embedded computing, internet of things (IoT), and wearable computing systems. Intel Galileo are compatible with Arduino in the high-level programming language, the IDE, and the onboard pins. More importantly, Galileo is an Intel x86 compatible, Pentium class System-on-a-Chip (SoC), with a 400 MHZ CPU and 256MB Ram (the original Arduino board only has 32KB Ram). Because of this, Galileo is capable of running a full-strength OS (Windows, Linux), high-level programming languages such as Python, and computation intensive tasks such as real-time image processing.



**Arduino** is an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The hardware consists of a simple open hardware design for the Arduino board with an 8bit Atmel micro-controller and on-board I/O support. The software consists of a standard programming language and the boot loader that runs on the board. The Arduino hardware is programmed using a Wiring-based language (syntax + libraries), similar to C++ with some simplifications and modifications, and a Processing-based IDE. Both the hardware and the software tools support all major OSes such as Windows, Mac OS X, and Linux for development purposes. Arduino UNO has an ATmega328 processor, 32 KB flash (for saving programs), 2 KB RAM, 14 Digital I/O pins and 7 analog I/O pins. The hardware of the Arduino is a board containing an ATmega microcontroller surrounded by several digital and analog I/O pins. The board also has a USB port which can power-up the board and exchange when necessary. There are two major usage scenarios of Arduino. First, Arduino can be used as an interface board to send information to the host computer and generate physical activities via lights/sounds/actuators by following the host computer’s instructions. Second, Arduino can also be used as a standard alone, highly portable computer without the help of a host computer. In the second mode, The program should be able to fit into the 32 KB on board flash memory and 9V power via either a battery or an AC Adapter is necessary.

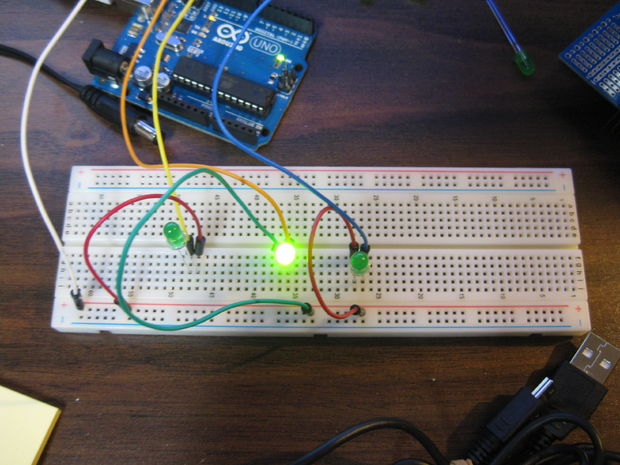


# Layout

The following components were designed using Intel Galileo processors:

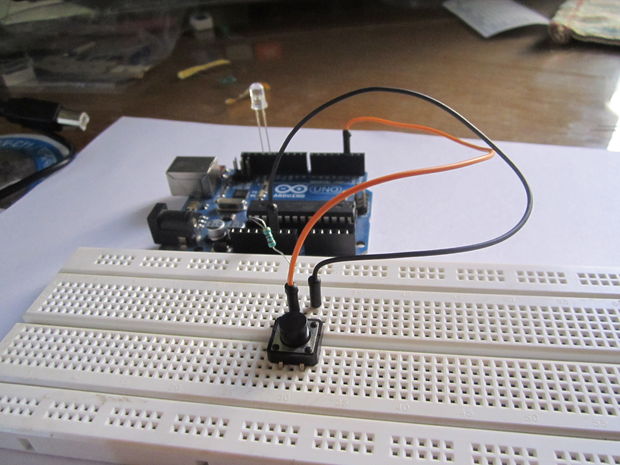
***1. Blinking Multiple LEDs automatically:***

In this assignment, we drove multiple LEDs, i.e., three LEDs, by using three LEDs. The LED was turned for 500 milli-seconds and then turned off and then the next LED was turned on. The layout of the connection is as follows:



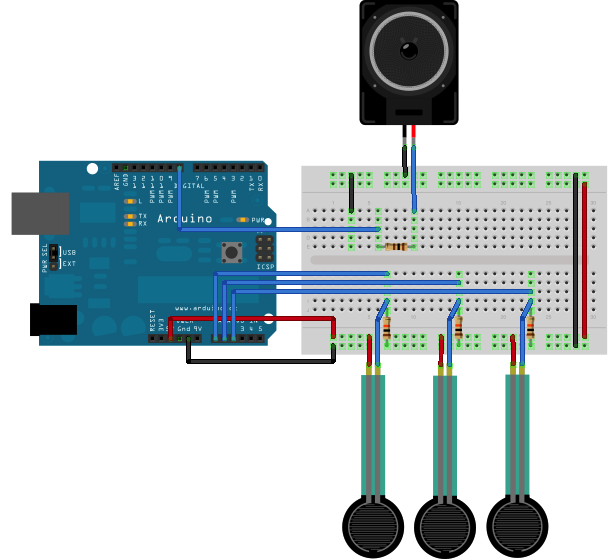
***2.*** ***Driving Multiple LEDs with a push button:***

In this assignment, we used one push button to control three LEDs. The three LEDs were used to represent a 3-bit digit counter and pressing the button each time added the internal state of the counter by one and hence changed the state of the three LEDs. The layout of the connection is as follows:



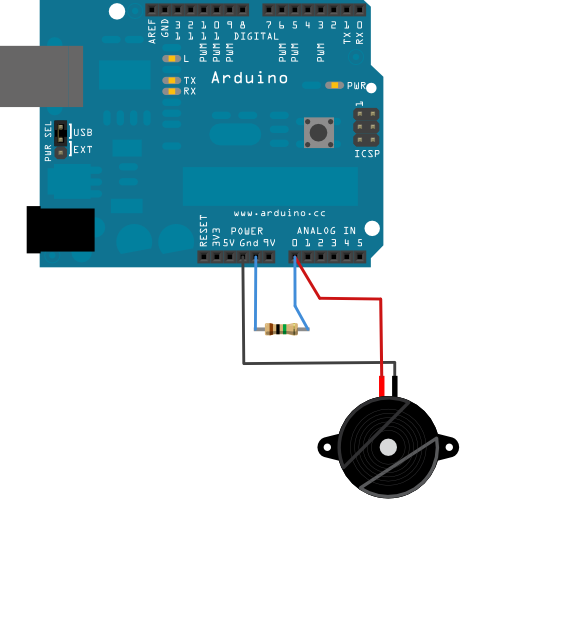
***3.*** ***Two-key musical keyboard :***

For this assignment, we used a Galileo board, 8 ohm speaker, 3 force sensing resistors, 3 10K ohm resistors, 100 ohm resistors, jumper wires and breadboard. The two key musical keyboard was implemented using the following layout:



*4.* ***Knock Sensor:***

For this assignment, we used a Galileo board, 1 mega ohm speaker, piezo electric disk, solid surface, jumper wires and breadboard. The knock sensor was implemented using the below layout:



*5.* ***Serial Communication:***

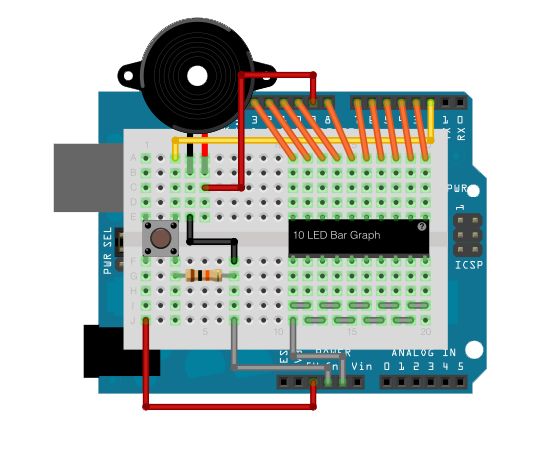
For this assignment, we connected two force sensors and one photocell to Intel Galileo, read values from these sensors and sent them to the PC. From the PC side, we wrote a program to visualize these sensor values (the easiest visualization is displaying both the numeric value and a corresponding bar chart for each channel).

***6.*** ***Kitchen Timer:***

For a kitchen timer, we need press and hold a button and it will count up it multiples of five minutes, until you release the button. Upon doing so the timer will flash, and begin counting down. This timer includes an alarm and a display, with a piercing piezo buzzer to get your attention. The purpose behind this kitchen timer is to create a device to alert the user when the food is prepared and ready to eat. A python program has been designed to alert the user by sending him an email indicating that the food is ready.

To design a kitchen timer, we used an Intel Galileo board, Breadboard, Resistor, Mini push button. 10 bar LED graph, piezo electric buzzer and a 10K resistor.

The layout for the connection is as follows:



# Citations

<http://www.instructables.com/id/Arduino-Kitchen-Timer/step2/Build-It/>

<https://learn.adafruit.com/downloads/pdf/arduino-lesson-17-email-sending-movement-detector.pdf>

http://www.arduino.cc

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

http://arduino.cc/en/ArduinoCertified/IntelGalileo