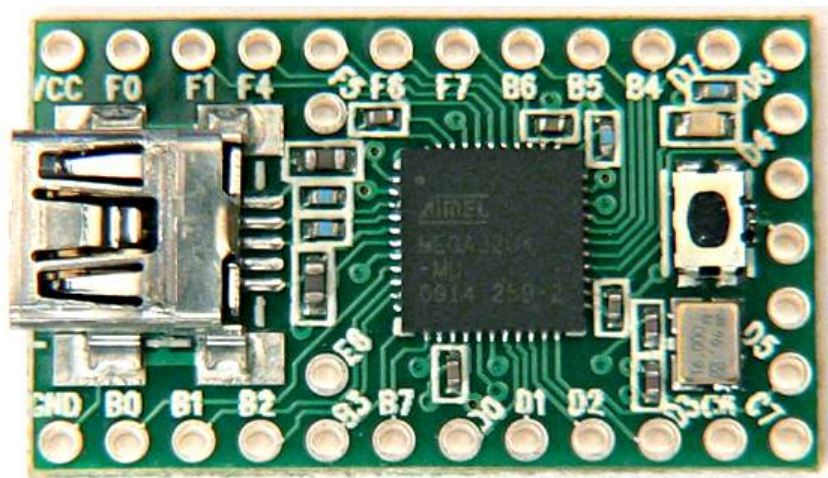


Teensy USB Development Board

Application Note

Shengzhe Gao
3/25/2014

Product:
Teensy 2.0



The Teensy USB Development Board is a complete USB-based microcontroller development system. Only a standard Mini-B USB cable (sold separately) is needed to connect to a PC or Macintosh. This application note describes the basic function and switch control codes for the Teensy 2.0 related with our project.

Introduction

The Teensy is a small, inexpensive microcontroller so that when used with the Teensyduino add-on for Arduino , can be considered part of the larger Arduino family.



A microcontroller can be thought of as a small, programmable computer with it's own program space (ROM), working memory (RAM), non-volatile memory (EEPROM), input ports and output ports.

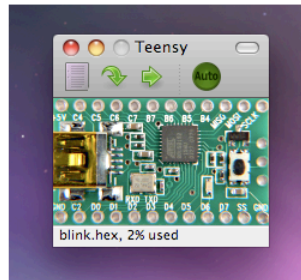
Unlike a laptop or a smartphone, a microcontroller does not come with speakers, visual display, keyboard or mouse built in. We need to build any input and output peripherals and connect them as a circuit to the microcontroller. In our project, we connect it to Odraid.

The Arduino is an open-source microcontroller platform and really revolutionised prototyping and development by becoming ubiquitous. This was partly due to a low cost entry point, easy to program IDE (integrated development environment), excellent documentation and a strong community with a willingness to share projects and code.

The Teensy belongs in this family and is related to the Arduino in terms of functionality and programming. We will be using the Teensy because it is cheap and easy to integrate into our project as a switch control.

Software Setup

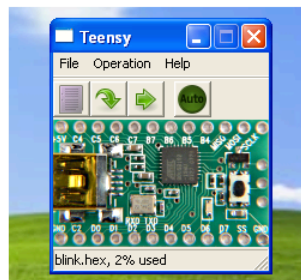
1. Install the Teensy Loader in your system.



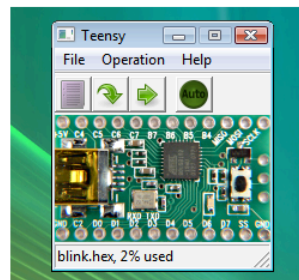
[Macintosh OS X 10.5](#)



[Linux \(Ubuntu\)](#)

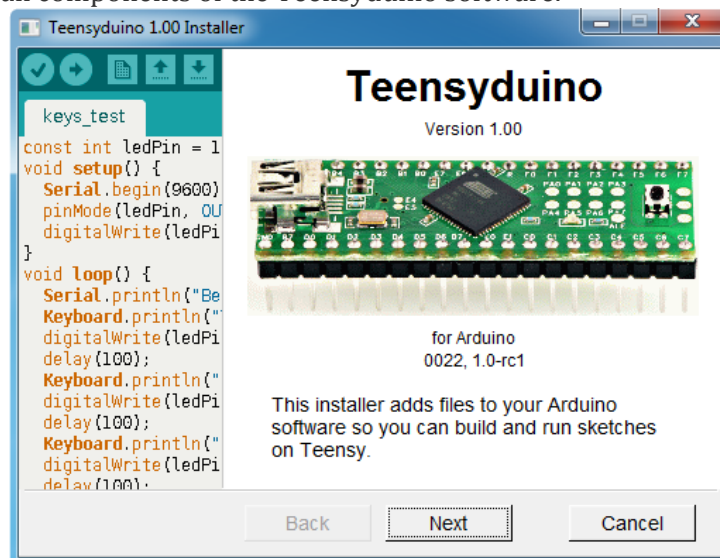


[Windows XP](#)

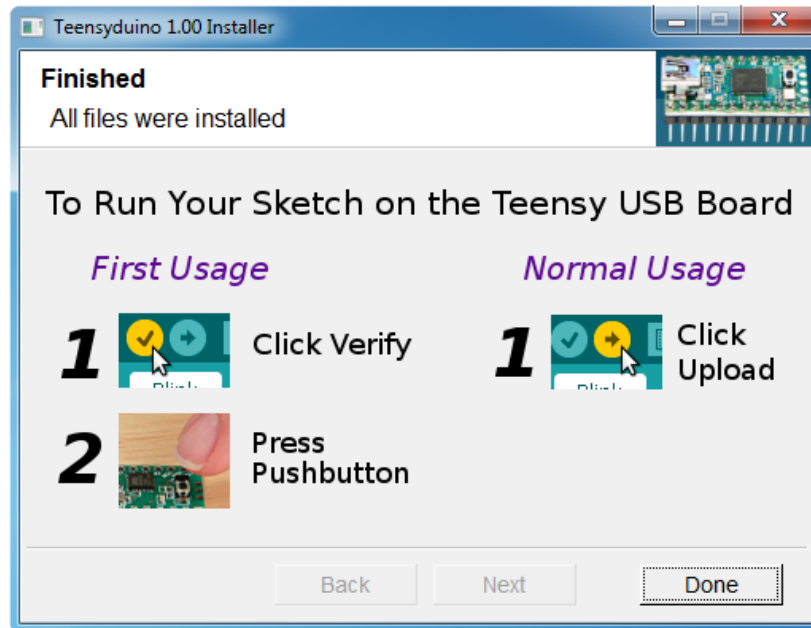


[Windows 7 & Vista](#)

2. Install the Arduino.
3. Install the Teensyduino installer.
 - Run the Teensyduino installer
 - The Teensyduino installer adds the necessary support files to Arduino
 - During the install for the Teensyduino software, you will be asked to locate your Arduino software
 - Install all components of the Teensyduino software.



4. When installation is finished, you will see this final screen. Just click Done to quit the installer.

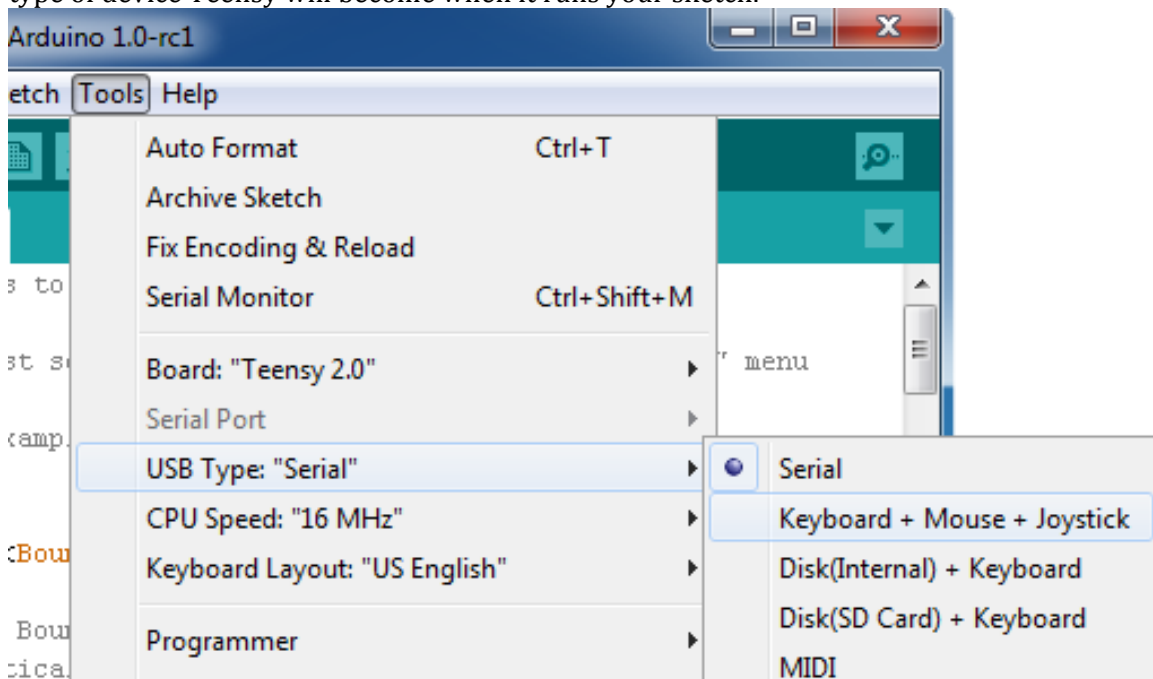


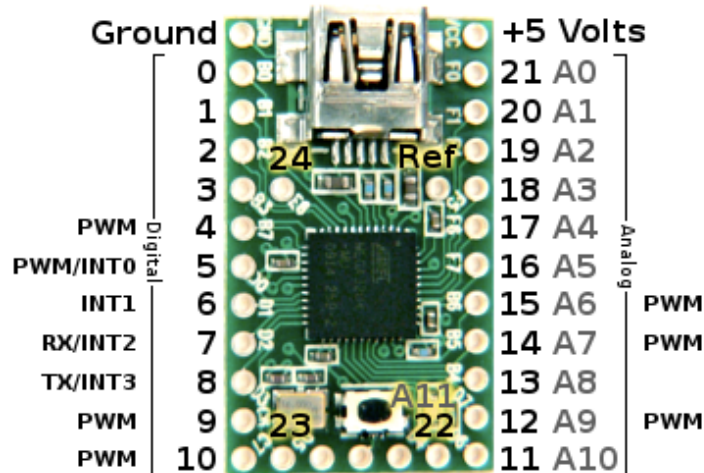
Basic Function

In general, the Arduino software and the Teensy software interact like this:

- 1) You write your code in Arduino
- 2) You upload your code using the appropriate command in Arduino
- 3) This automatically launches the Teensy loader
- 4) You may need to press reset on the Teensy board in order to upload the program
- 5) The Teensy loader software will inform you of a clean reboot of the board

Teensy is not limited to only serial device type. The Tools > USB Type menu can select the type of device Teensy will become when it runs your sketch.





Here we can see the Teensy board and its pin layout. The top left pin is ground and the top right pin is 5V. The remaining pins along the left hand side are numbered 0 - 10, whilst the remaining pins along the right hand side are labeled 11 - 21 AND A0 - A10.

The pin numbers from 0 - 21 refer to these pins operating as "digital" pins - these pins can be either inputs or outputs, and can respond to or control signals that are either HIGH or LOW, GROUND or 5V, 0 or 1. This includes things like buttons and switches as inputs, and LED lights as outputs.

The pin numbers from A0 - A10 are analog inputs. These pins can read a voltage that is anywhere from 0V to 5V, and return this voltage as a data value. These pins can be used for things like potentiometer knobs, sliders, light dependent resistors and so on - things that aren't just on or off, but have a continuous value from 0 - 5V.

Note that there are four more pins inside of the Teensy board, marked 22 A11, 23, 24 and REF. 22 A11 is an analog input pin AND a digital pin whereas 23 and 24 are digital pins.

In summary, the Teensy has 25 IO pins, 12 of which are analog input pins and 13 of which are digital only pins. The analog input pins can act as digital pins as well.

The Teensy has roughly 30KB of ROM program space and roughly 2KB of RAM working memory. The Teensy has an onboard LED that is connected to digital pin 11.

Switch Control Code

1. Using USB Keyboard

When you select "USB Keyboard" from the Tools -> USB Type menu, the Teensy becomes a USB keyboard and mouse while running your program.

Your PC or Mac will detect a new keyboard. Then your program can send keystrokes which your computer will recognize as coming from a standard USB keyboard. There are two ways you can make your Teensy send USB keystrokes.

2. Basic Code

Keyboard.print() works the same way as Serial.print(), except the message is typed as keystrokes. You can print strings, numbers, single characters with all the same control as Serial.print().

Here is a very simple example, using Keyboard.print().

```
int count = 0;
void setup() { } // no setup needed
void loop() {
  Keyboard.print("Hello World");
  Keyboard.println(count);
  count = count + 1;
  delay(5000); }
```

When this program is run, it will type "Hello World" and the increasing count.

3. Using pushbuttons

Many applications need to send messages to a computer when pushbuttons are pressed.

The Bounce library provides the easiest way to reliably detect pushbuttons changes. This code shows the basic approach. Just `update()` each object, and then use the `fallingEdge()` to check if it has changed.

```
void loop() {

  // Update all the button objects.
  button1.update();
  button2.update();
  button3.update();
  // Check each button for "falling" edge.
  // falling = high (not pressed - voltage from pullup resistor)
  //          to low (pressed - button connects pin to ground)
  if (button1.fallingEdge()) {      Keyboard.println("B1 press"); }
  if (button2.fallingEdge()) {      Keyboard.println("B2 press"); }
  if (button3.fallingEdge()) {      Keyboard.println("B3 press"); }
```

For a complete example: **File > Examples > Teensy > USB_Keyboard > Buttons.**

4. The code in our project

```
#include <Bounce.h>

Bounce button0 = Bounce(0, 10); //run
Bounce button1 = Bounce(1, 10); //stop
Bounce button2 = Bounce(2, 10); //record
Bounce button3 = Bounce(3, 10); //view0,1,2,3,4

int count = 0;

void setup() {

  pinMode(0, INPUT_PULLUP);
  pinMode(1, INPUT_PULLUP);
  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);
  pinMode(4, INPUT_PULLUP);
  pinMode(5, INPUT_PULLUP);
  pinMode(6, INPUT_PULLUP);

}

void loop() {

  button0.update();
  button1.update();
  button2.update();
  button3.update();
  button4.update();
  button5.update();
  button6.update();

  if (button0.fallingEdge()) {
    Keyboard.println("./bscript");
  }
  if (button1.fallingEdge()) {
    Keyboard.println("stop");
  }
  if (button2.fallingEdge()) {
    Keyboard.println("record");
  }
  ...
}
```

Based on the code, there are four switches and two spare switched. The four switches give different order to the system, like run, stop, record and camera view change. The two spare switches are designed for future use.

Conclusion

In our project, we need to design four switches to deliver different orders to our system. We choose Teensy 2.0 as our switch control because it is small and cheap. The code for Teensy is based on C language, which we have already learned in class. While in our project, the teensy is used as USB keyboard, in some other project, the teensy can be used as USB serial, mouse, joystick, MIDI, flight sim, UART serial.