Arduino: Playground

Manuals and Curriculum

Hardware and Related Initiatives

Board Setup and Configuration

Development Tools

Interfacing With Hardware
- Output
- Input
- Interaction
- Storage
- Communication

Interfacing with Software

Code Library and Tutorials

Electronics Technique

Sources for Electronic Parts

Arduino People/Groups & Sites

Exhibition

Languages

PARTICIPATE
- create an account
- suggestions
- formatting suggestions
- all recent changes
- PmWiki
- WikiSandBox training
- Basic Editing
- Cookbook (addons)
- Documentation Index)

login logout edit SideBar admin

The **Servo Timer 1** Library drives servos on **pins 9 and 10** by using the timer1 hardware. The API is patterned after the wiring.org servo library, but the code is different.

Standard Methods

attach(int)
  Turn a pin into a servo driver. Calls pinMode. Returns 0 on failure.
detach()
  Release a pin from servo driving.
write(int)
  Set the angle of the servo in degrees, 0 to 180.
read()
  return that value set with the last write().
attached()
  return 1 if the servo is currently attached.

Extra Methods

refresh()
  Ignored, for compatibility with the Software Servo Library.
setMinimumPulseWidth(uint16_t)
  set the duration of the 0 degree pulse in microseconds.
setMaximumPulseWidth(uint16_t)
  set the duration of the 180 degree pulse in microseconds.

Safety Quirk

Even though you attach a servo, it won't receive any control signals until you send its first position with the write() method to keep it from jumping to some odd arbitrary value. *This statement may or may not be true.*

Size

The library takes about 450 bytes of flash and 2+(4*servos) bytes of SRAM.

The Code

You can find the code in http://www.arduino.cc/playground/uploads/ComponentLib/servotimer1.zip. Unpack that into your lib/targets/libraries folder to add the library.

Advantages

This library controls the servos completely in hardware. You should expect less than one clock cycle of jitter and to never see an extended, delayed, or missing pulse. It also will not disturb any of your program's other timing.

Limitations

This only works on pins 9 and 10. If you use this library you may not use either pin as an analog output. There are no software tasks needed to keep the servos running, it is all done in hardware. If you detach() both servos, you could use timer1 for something else for a bit and then attach() a servo and it would start functioning as a servo channel again.

An Example

The following code lets you send strings like "90s" and "80w" to position servos on pin 14 and 15 (analog in 0 and 1) to 90 degrees and 80 degrees. You can also use "d" to detach the servo on pin 15 and "a" to reattach it. Ok, it is a silly program but it works for testing.

```
//Example code for using ServoTimer1 library
// hardware control of up to two servos, on Arduino pins 9 & 10

->#include <ServoTimer1.h>

ServoTimer1 servo1;
ServoTimer1 servo2;

void setup()
{
  pinMode(1,OUTPUT);
  servo1.attach(9);
  servo2.attach(10);
  Serial.begin(19200);
  Serial.print("Ready");
}

void loop()
{
  static int v = 0;

  if ( Serial.available()) {
    char ch = Serial.read();

    switch(ch) {
      case '0'...'9':
        v = v * 10 + ch - '0';
        break;
      case 's':
        servo1.write(v);
        v = 0;
        break;
      case 'w':
        servo2.write(v);
        v = 0;
        break;
      case 'd':
        servo2.detach();
        break;
      case 'a':
        servo2.attach(15);
        break;
    }
  }

  Servo::refresh(); // not needed, for compatibility with the software servo library
}
```