ATtiny 85

Krympa projekt från Arduino Uno

Material:

Arduino Uno

Attiny 85

Breadboard

Jumper wire

Kondensator 10µF

Börja med att öppna Arduino-programmet.

I Arduino väljer du exempel och laddar ner ArduinoISP (in-system programmer) till UNO. Detta gör att Arduino är förberedd för att användas för att programmera ATtiny med.

Att få programmet att fungera för att ladda ner till ATtiny finns beskrivet på MIT´s sida http://highlowtech.org/?p=1695

(kolla även deras coola projekt)

En kort beskrivning på hur du gör:

I Programmet gå till Fil/Inställningar

I fältet Additional Boards Manager URLs: skriv:

https://raw.githubusercontent.com/damellis/attiny/ide-1.6.x-boards-manager/package_damellis_attiny_index.json

Om det redan står något i fältet separera länkarna med komma

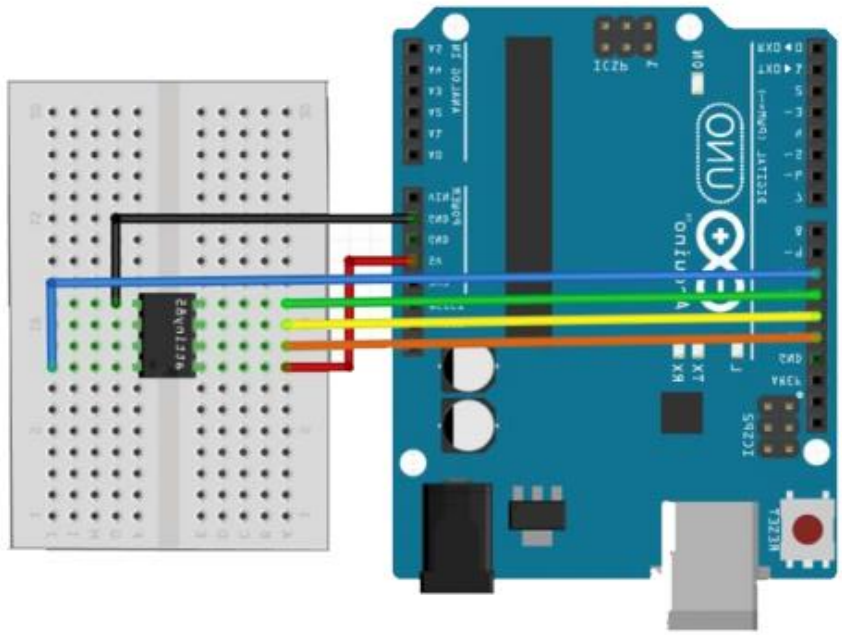
Klicka OK.

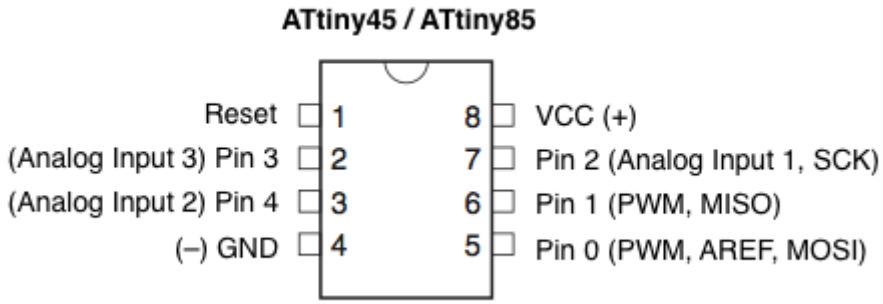
Öppna Boards Manager under Verktyg/Kort

Längst ner ska du nu se ATtiny

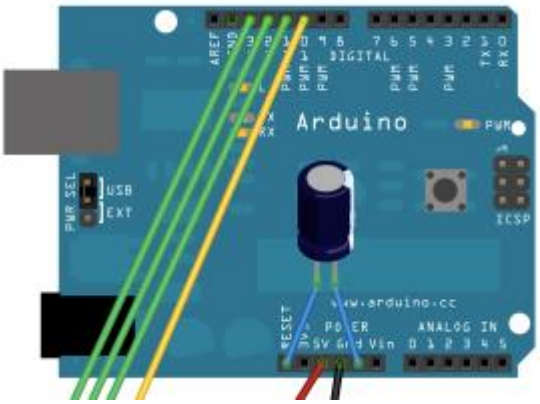
Klicka Install och du får då en bekräftelse på att installering är klar.

Koppla ihop Arduino och ATtiny85





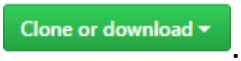
Sätt kondensatorn mellan GND och reset.



Använda ATtiny85 med servo

För att kunna programmer ATtiny85 för bruk med servo behöver du ladda ner Adafruit_SoftServo library.

https://github.com/adafruit/Adafruit_SoftServo

Klicka på 

.

Du behöver inte extrahera mappen utan gå till Skiss/Inkludera bibliotek/Lägg till .ZIP bibliotek. Sök rätt på den nedladdade mappen och välj den.

Nu är det dags att skapa kod.

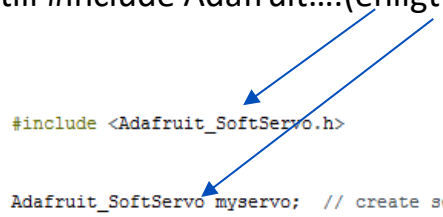
Nedan har jag tagit exemplet Sweep som finns under Fil/Exempel/Servo. Exemplet är för en servo

Byt ut

```
#include <Servo.h>

Servo myservo;  // c
```

till #include Adafruit….(enligt nedan)



```
#include <Adafruit_SoftServo.h>

Adafruit_SoftServo myservo;  // create servo object to control a servo

int pos = 0;    // variable to store the servo position

void setup() {
  myservo.attach(0);  // Turn a pin 0 into a servo driver
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);           // tell servo to go to position in variable 'pos'
    delay(15);                    // waits 15ms for the servo to reach the position
     myservo.refresh();          // You must call this at least once every 50ms to keep the servos updated
  }

  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
   myservo.refresh();                // You must call this at least once every 50ms to keep the servos updated
  }
}
```

Där det står

```
void setup() {
  myservo.attach(9);
}
```

måste du byta pin nr då ATtiny inte har så många pins. Välj t ex pin 0.

Nu är det dags att ställa in programmet.

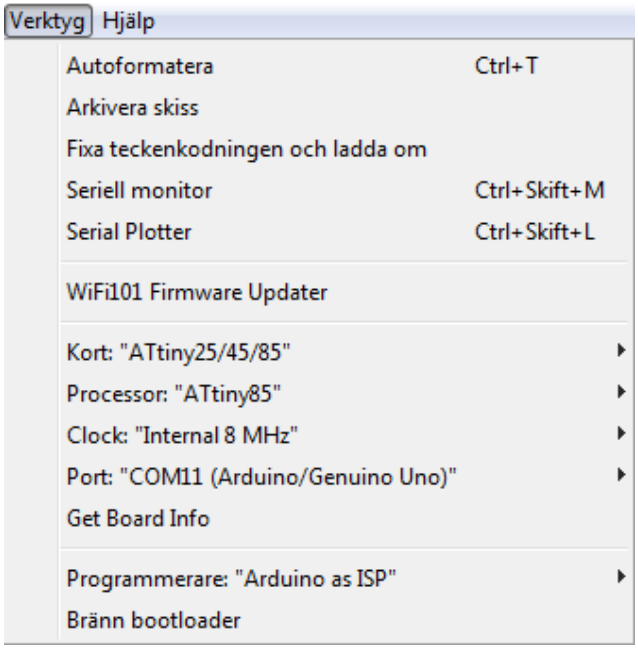
Under Verktyg ändra:

Kort: till ATtiny

Processor: till ATtiny

Clock: till 8MHz Ska du bara programmera LED ska du inte byta till 8 MHz (då ska det fortsätta vara 1MHz)

Programmerare: till Arduino as ISP



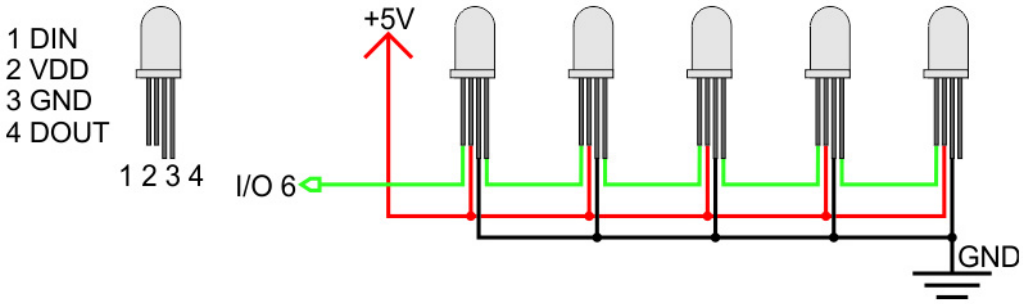
Nu kopplar du in Arduino med ATtiny och klickar Bränn bootloader. (förbereder ATtiny för koden).

När det är klart kan du klicka Ladda upp och föra över programmet till ATtiny. 

Dags att koppla ur Arduino och sätta i batterier och servo.

Använda ATtiny85 med NeoPixels

Jag använder SJ-F5_DIFF från Lavicel



Bibliotek för NeoPixels finns här:
https://github.com/adafruit/Adafruit_NeoPixel

I övrigt går processen till på samma sätt och även NeoPixlar behöve 8MHZ. Bränn Bootloader och ladda ner kod.

Kod nedan som jag använt.

```
#include <Adafruit_NeoPixel.h>

#define PIN 0 //define what pin to use



Adafruit_NeoPixel strip = Adafruit_NeoPixel(5, PIN, NEO_RGB + NEO_KHZ800);
// First number is number of NeoPixels used (here 5)

// IMPORTANT: To reduce NeoPixel burnout risk, add 1000 uF capacitor across
// pixel power leads, add 300 - 500 Ohm resistor on first pixel's data input
// and minimize distance between Arduino and first pixel.  Avoid connecting
// on a live circuit...if you must, connect GND first.

void setup() {
  strip.begin();
  strip.show(); // Initialize all pixels to 'off'
}

void loop() {
 // Some example procedures showing how to display to the pixels:
colorWipe(strip.Color(255, 0, 0), 50); // Red
colorWipe(strip.Color(0, 255, 0), 50); // Green
colorWipe(strip.Color(0, 0, 255), 50); // Blue
  // Send a theater pixel chase in...
  theaterChase(strip.Color(127, 127, 127), 50); // White
  theaterChase(strip.Color(127,   0,   0), 50); // Red
  theaterChase(strip.Color(  0,   0, 127), 50); // Blue
```

```
  rainbow(20);
  rainbowCycle(20);
  theaterChaseRainbow(50);
}

//Fill the dots one after the other with a color
void colorWipe(uint32_t c, uint8_t wait) {
  for(uint16_t i=0; i<strip.numPixels(); i++) {
      strip.setPixelColor(i, c);
      strip.show();
      delay(wait);
  }
}

void rainbow(uint8_t wait) {
  uint16_t i, j;

  for(j=0; j<256; j++) {
    for(i=0; i<strip.numPixels(); i++) {
      strip.setPixelColor(i, Wheel((i+j) & 255));
    }
    strip.show();
    delay(wait);
  }
}

// Slightly different, this makes the rainbow equally distributed throughout
void rainbowCycle(uint8_t wait) {
  uint16_t i, j;

  for(j=0; j<256*5; j++) { // 5 cycles of all colors on wheel
```

```
   for(i=0; i< strip.numPixels(); i++) {
    strip.setPixelColor(i, Wheel(((i * 256 / strip.numPixels()) + j) & 255));
   }
   strip.show();
   delay(wait);
  }
}

//Theatre-style crawling lights.
void theaterChase(uint32_t c, uint8_t wait) {
 for (int j=0; j<10; j++) {  //do 10 cycles of chasing
   for (int q=0; q < 3; q++) {
    for (int i=0; i < strip.numPixels(); i=i+3) {
     strip.setPixelColor(i+q, c);    //turn every third pixel on
    }
    strip.show();

    delay(wait);

    for (int i=0; i < strip.numPixels(); i=i+3) {
     strip.setPixelColor(i+q, 0);       //turn every third pixel off
    }
   }
  }
}

//Theatre-style crawling lights with rainbow effect
void theaterChaseRainbow(uint8_t wait) {
 for (int j=0; j < 256; j++) {     // cycle all 256 colors in the wheel
   for (int q=0; q < 3; q++) {
     for (int i=0; i < strip.numPixels(); i=i+3) {
      strip.setPixelColor(i+q, Wheel( (i+j) % 255));    //turn every third pixel on
```

```
      }
      strip.show();

      delay(wait);

      for (int i=0; i < strip.numPixels(); i=i+3) {
        strip.setPixelColor(i+q, 0);       //turn every third pixel off
      }
    }
  }
}

// Input a value 0 to 255 to get a color value.
// The colours are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos) {
  WheelPos = 255 - WheelPos;
  if(WheelPos < 85) {
   return strip.Color(255 - WheelPos * 3, 0, WheelPos * 3);
  } else if(WheelPos < 170) {
    WheelPos -= 85;
   return strip.Color(0, WheelPos * 3, 255 - WheelPos * 3);
  } else {
   WheelPos -= 170;
   return strip.Color(WheelPos * 3, 255 - WheelPos * 3, 0);
  }
}
```