

Textul si imaginile din acest document sunt licentiate

Attribution-NonCommercial-NoDerivs
CC BY-NC-ND



Codul sursa din acest document este licentiat

Public-Domain

Esti liber sa distribui acest document prin orice mijloace consideri (email, publicare pe website / blog, printare, sau orice alt mijloc), atat timp cat nu aduci nici un fel de modificari acestuia. Codul sursa din acest document poate fi utilizat in orice fel de scop, de natura comerciala sau nu, fara nici un fel de limitari.

Asamblare statie meteo



Statia meteo este capabila de a masura 3 parametrii importanti: viteza, directia vantului si cantitatea de precipitatii. Senzorii din aceasta statie sunt formati doar din switch-uri reed si magneti, ceea ce inseamna ca sunt foarte usor de utilizat, dar pentru a functiona acestia au nevoie de o sursa de alimentare.

Componentele principale ale statiei meteo sunt:

- anemometrul este responsabil cu masurarea vitezei vantului. Analogic vorbind, acesta functioneaza asemeni unui buton conectat la o placa Arduino. La o viteza de 2.4km/h anemometrul inchide un switch la intervale de o secunda.
- Girueta determina directia vantului si utilizarea senzorului presupune citirea unei tensiuni

folosind un convertor analog-digital. In interiorul giruetei se afla o retea de 8 rezistori cu 8 switch-uri, care pot indica pana la 16 directii posibile. In plus, vei avea nevoie de un rezistor fix care impreuna cu celelalte 8 formeaza un divizor rezistiv. Pentru fiecare directie, senzorul genereaza o tensiune electrica. Tensiunea poate fi citita de catre o placa Arduino si corelata cu directia vantului.

- Pluviometrul determina cantitatea de precipitatii, se goleste singur, iar fiecare 0.2794 mm de ploaie produce declansarea unui switch. Aceasta declansare poate fi interpretata de catre o placa Arduino ca si un buton sau ca o „intrerupere“.

Statia meteo se asambleaza urmand pasii de mai jos:



1. Bratul care sprijina senzorii enumerati mai sus este alcatuit din 2 bare. Cele 2 bare se infig una intr-alta (vezi capatul de sectiune mai mica a barei de sus).

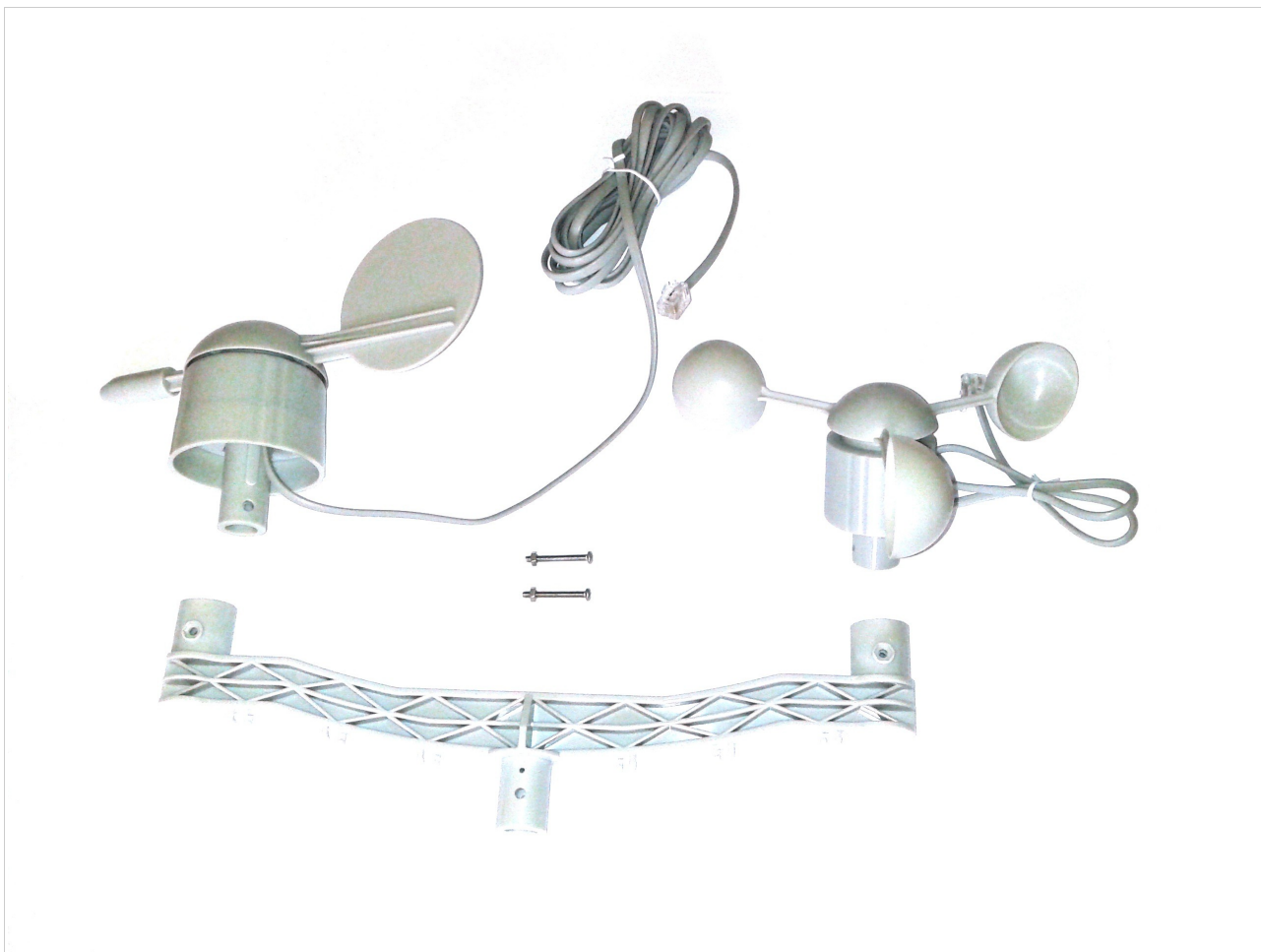


2. Vei obtine o bara mult mai mare avand la un capat o gaura pentru surub si un canal de fixare (vezi imaginea).
3. Urmeaza montarea anemometrului si a giruetei pe bara de sustinere folosind un suport de

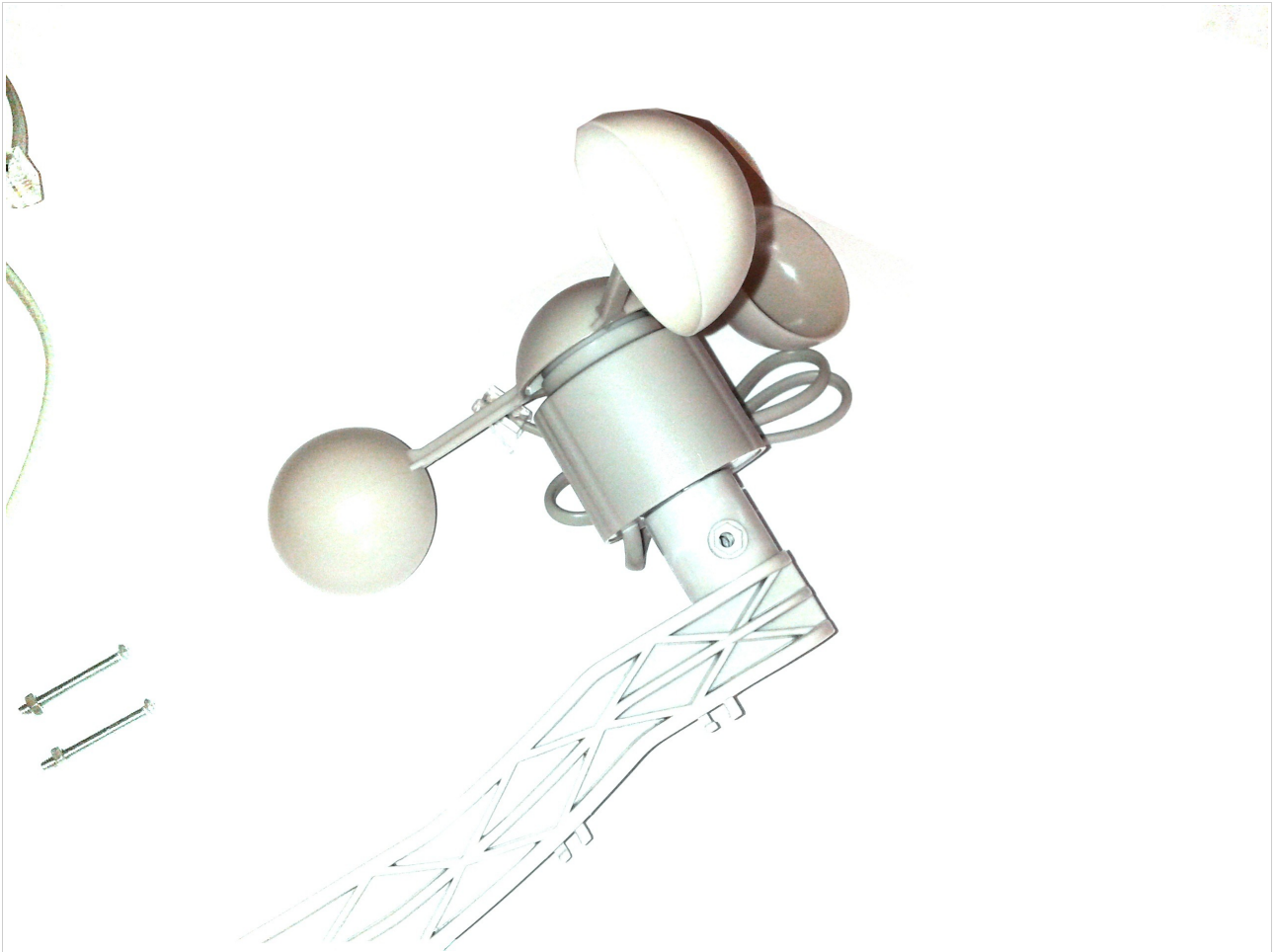
fixare. Suportul este cel din imagine, pe care se monteaza anemometrul si girueta, iar in partea de jos se fixeaza in bara de sustinere (vezi imaginea).



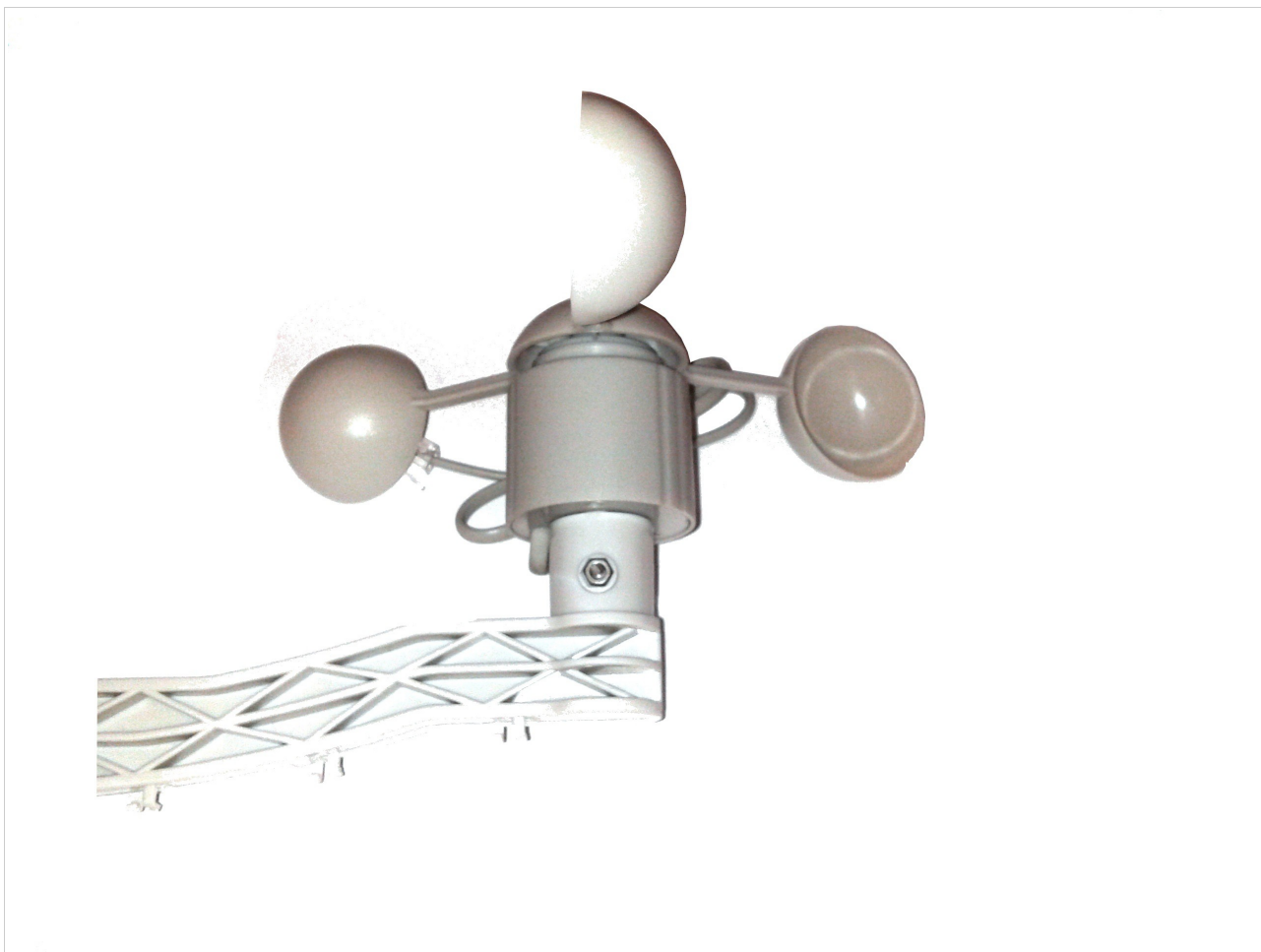
4. Inainte de a fixa suportul, mai intai trebuie sa montezi senzorii de vant. Montarea este simpla, tot ce trebuie sa faci este sa fixezi senzorii pe cele 2 brate si sa montezi surubul si piulita.



5. Anemometrul se fixeaza in suportul lui special, apoi se fixeaza ferm folosind surubul si piulita.

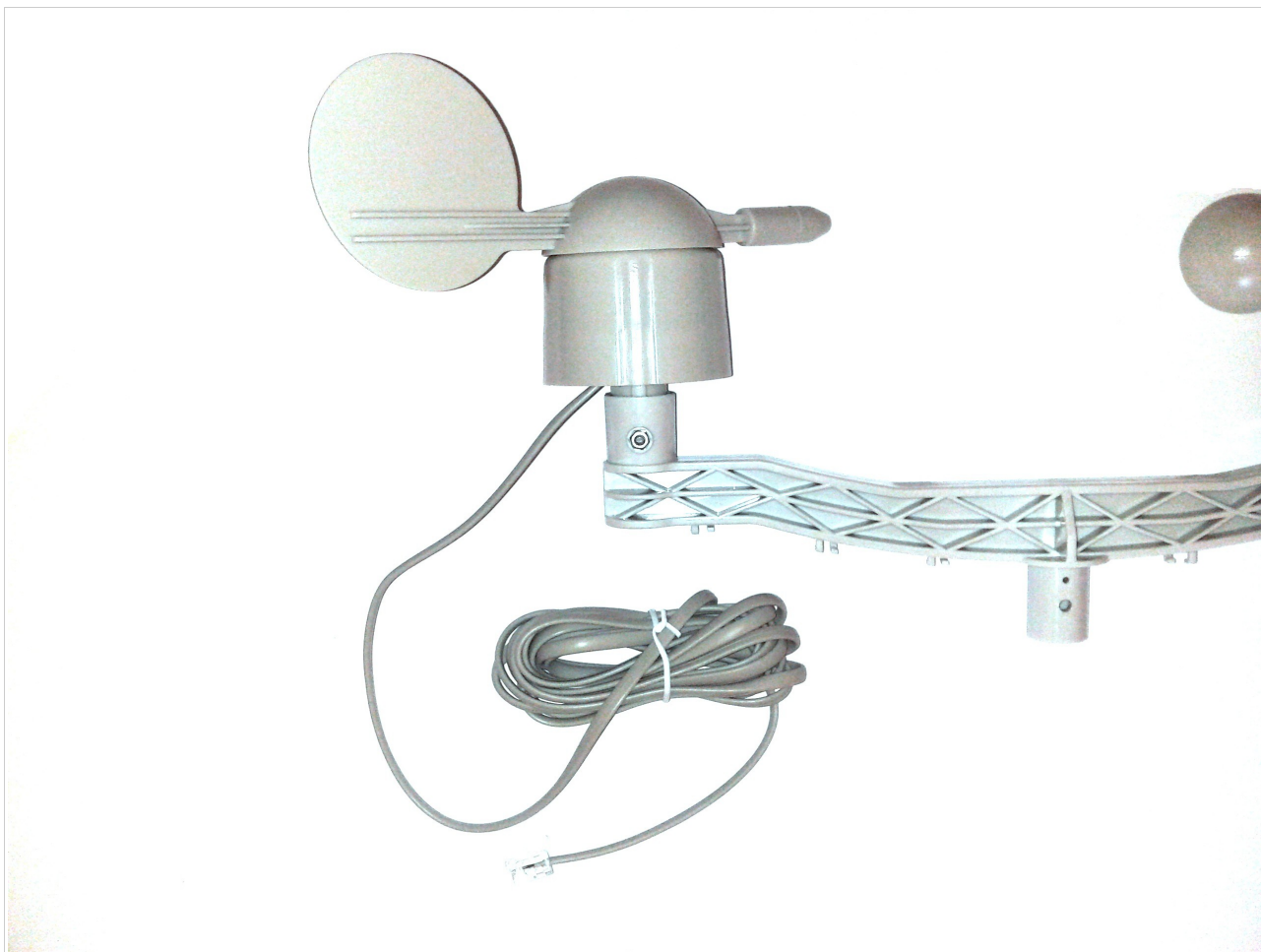






6. Girueta se monteaza in acelasi mod ca si anemometrul. Se monteaza in suport si se fixeaza ferm cu surubul si piulita.





7. Urmeaza sa montezi acest suport in bara de sustinere a statiei. Montarea este simpla, tot ce trebuie sa faci este sa fixezi suportul in capatul barei de sustinere si sa strangi cele 2 holsuruburi.

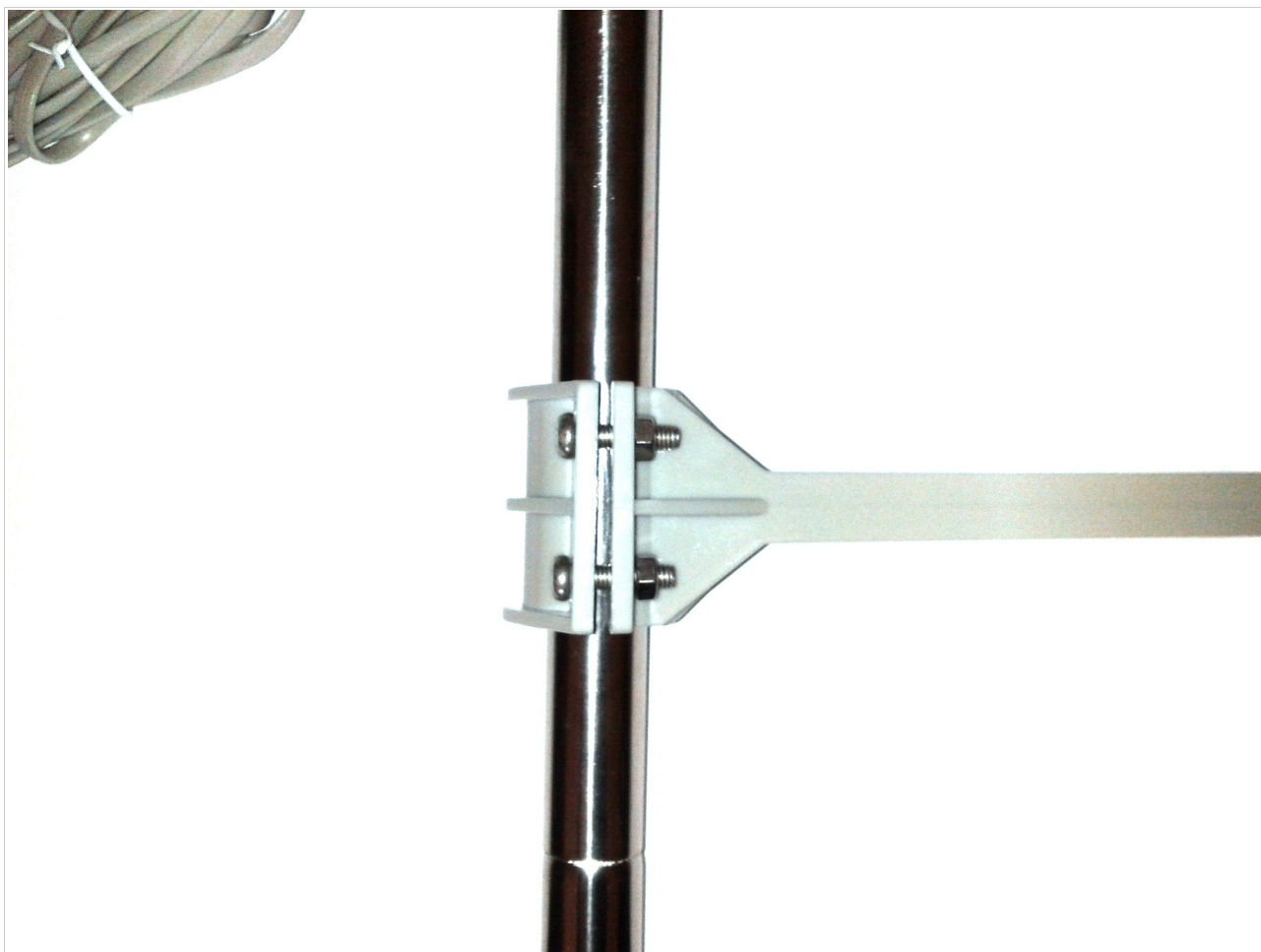




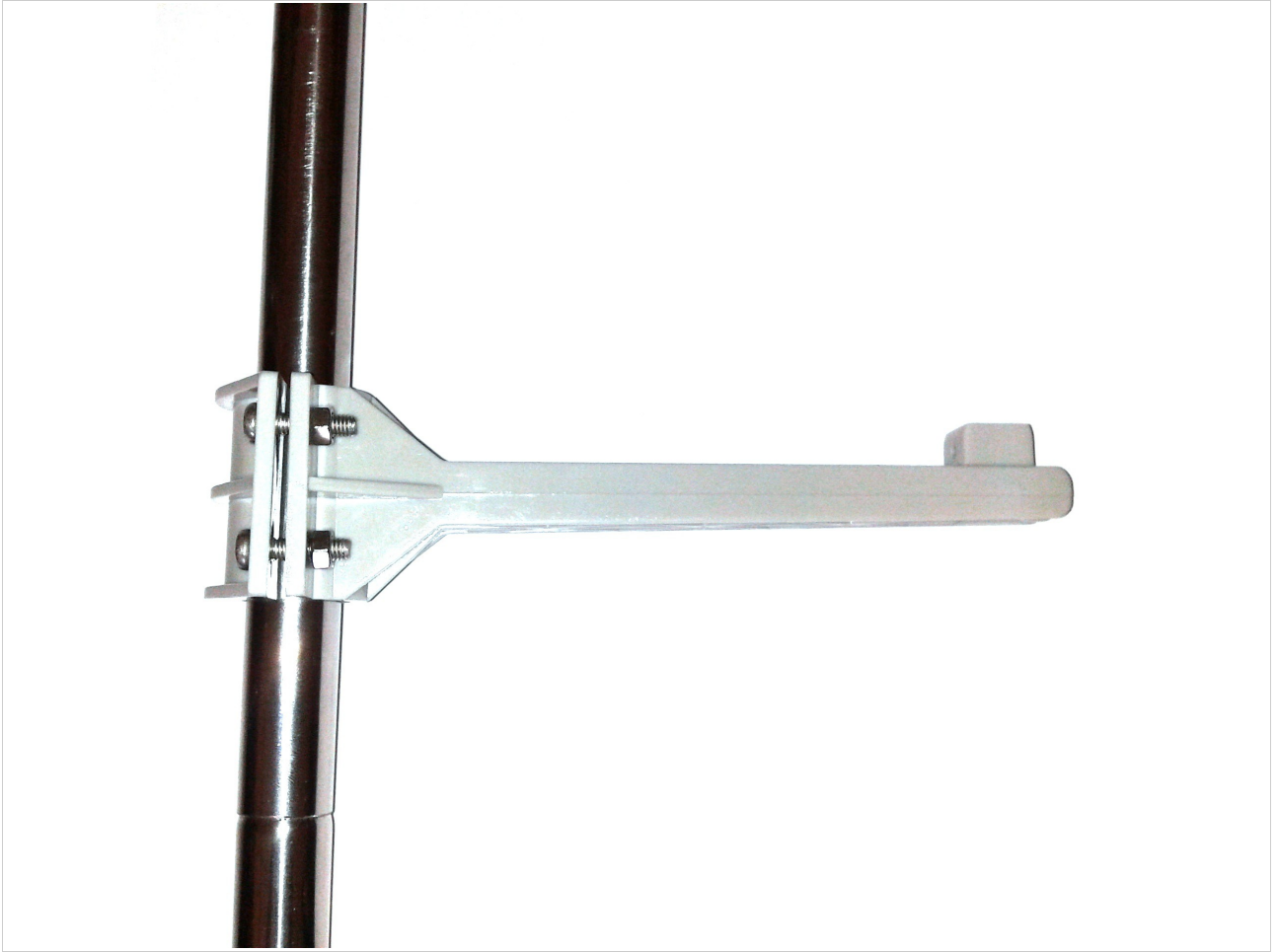
8. Pluviometrul se monteaza pe un brat secundar al statiei meteo.

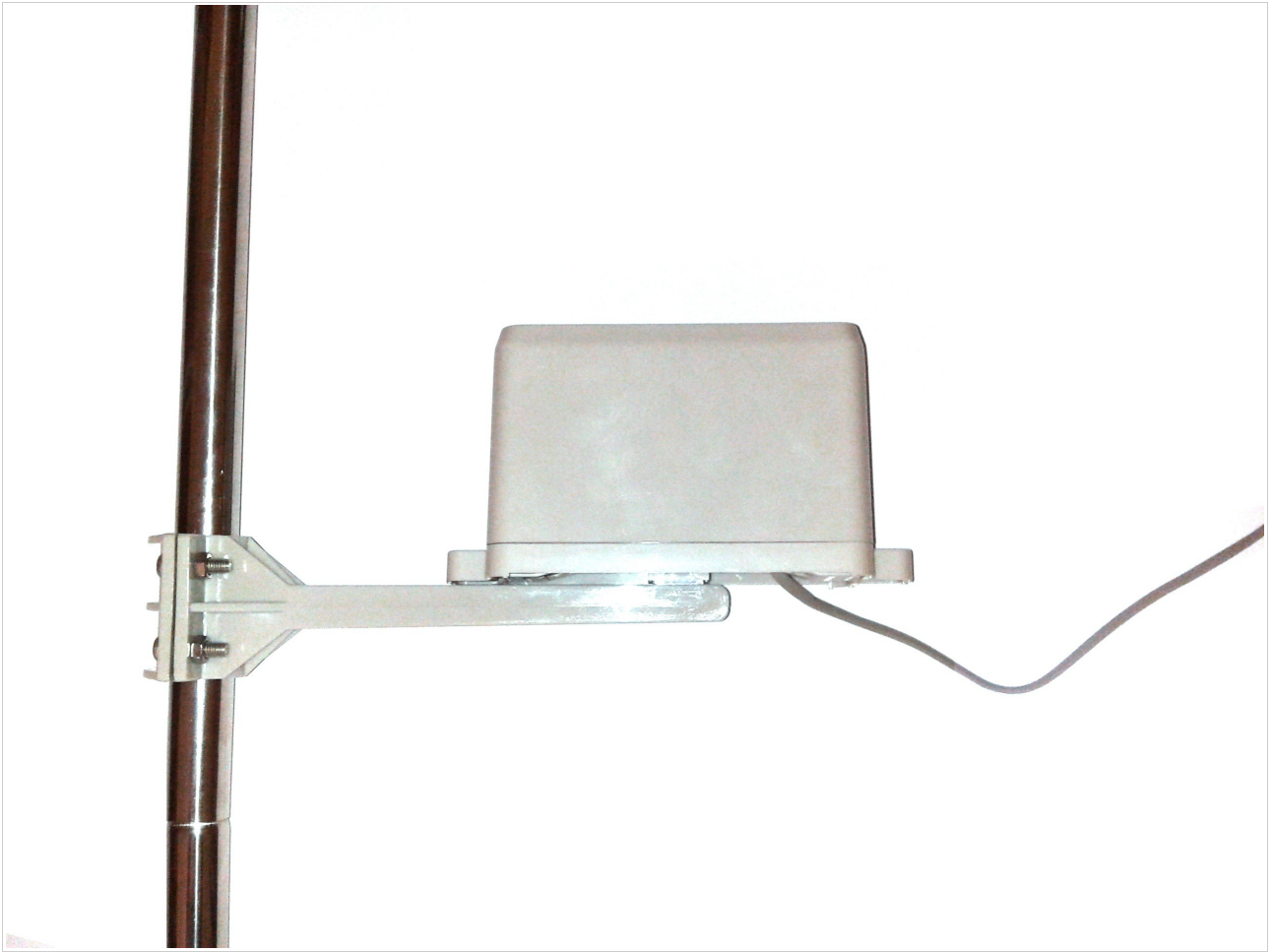


9. Mai intai trebuie sa fixezi bratul secundar de bara de sustinere si sa strangi ferm suruburile (vezi imaginea)



10. Pluviometrul se fixeaza in capatul bratului secundar prin intermediul unui holsurub.



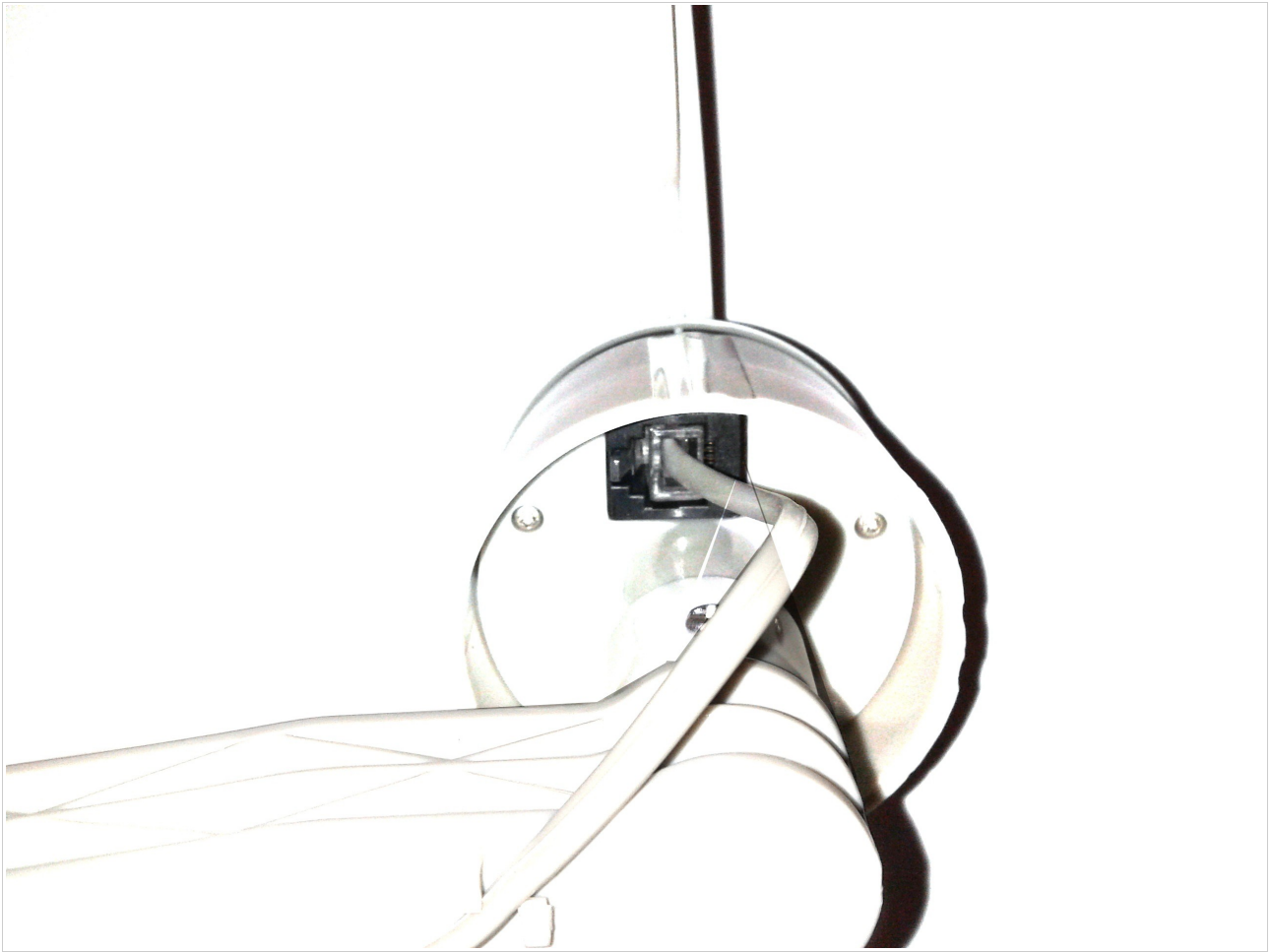




11. In urmatorul pas, trebuie sa pozezi cablurile senzorilor in clipsurile din plastic aflate dedesubt de suport. Cablul anemometrului este scurt, deoarece acesta se conecteaza in mufa giruetei (vezi imaginea)







12. Cablurile ramase (cele care practic se conecteaza in placa cu microcontroller) se fixeaza cu colierele din plastic de bara de sustinere a statiei.







13. Asamblarea a luat sfarsit !

Daca doresti sa realizezi o implementare rapida a statiei meteo folosind o placa Arduino, foloseste codul sursa de mai jos:

Sunt aspecte importante pe care trebuie sa le cunosti in avans:

- Sketch-ul de mai jos realizeaza citirea anemometrului si a giruetei dar nu si a pluviometrului.
- Diferenta de cod dintre pluviometru si anemometru este minima deoarece pluviometrul inchide un simplu contact la fiecare 0.2794 mm de precipitatii.
- Pentru conectarea anemometrului ai la dispozitie firele centrale ale cablului RJ11.
- Pentru conectarea giruetei ai la dispozitie firele exterioare ale cablului RJ11.
- Urmeaza explicatiile si diagrama de conectare din codul sursa de mai jos.

```
/* Arduino sketch for Weather device from Sparkfun.
Uses only the wind direction vane and the anemometer (not the rain
gauge) .
```

Although the inclination for a weather logger is to run it for a long time, due to the way Wiring.c implements the millis() function, this should be restarted, oh, monthly. The millis() functions overflows after about 49 days. We could allow for that here, and handle the wraparound, but you've got bigger problems anyway with the delay() function at an overflow, so it's best to "reboot".

```
=====
ANEMOMETER
=====
```

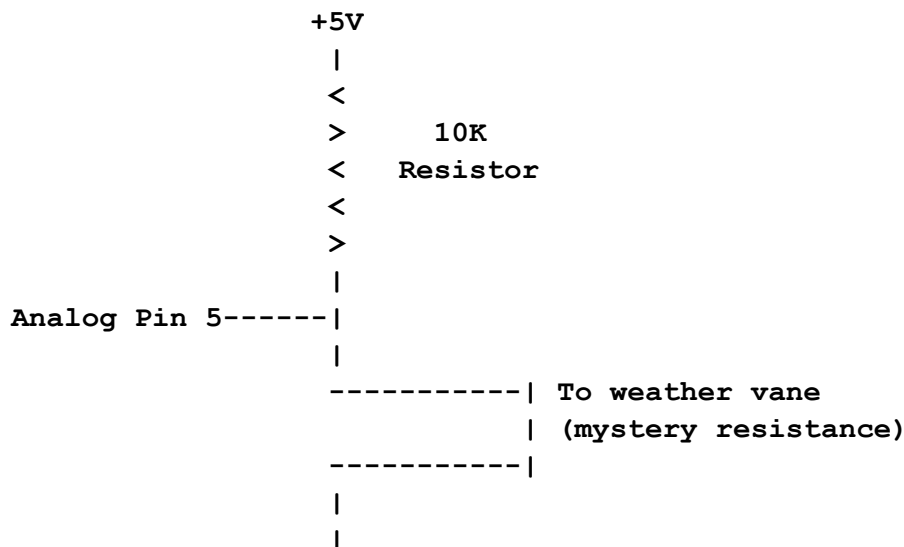
This is connected to Arduino ground on one side, and pin 2 (for the attachInterrupt(0, ...) on the other. Pin 2 is pulled up, and the reed switch on the anemometer will send that to ground once per revolution, which will trigger the interrupt. We count the number of revolutions in 5 seconds, and divide by 5. One Hz (rev/sec) = 1.492 mph.

```
=====
WIND DIRECTION VANE
=====
```

We use a classic voltage divider to measure the resistance in the weather vane, which varies by direction. Using a 10K resistor, our ADC reading will be:

$$1023 * (R / (10000 + R))$$

where R is the unknown resistance from the vane. We'll scale the 1023 down to a 255 range, to match the datasheet docs.




```

-----
---
-

```

The ADC values we get for each direction (based on a 255 max) follow, assuming that pointing away from the assembly center is sector zero. The sector number is just which 45-degree sector it is, clockwise from the "away" direction. The direction shown is assuming that "away" is West. Depending how you orient the system, you'll have to adjust the directions.

Sector	Reading	Direction
0	18	W
1	33	NW
2	57	N
7	97	SW
3	139	NE
6	183	S
5	208	SE
4	232	E

The values in the ADC table below list the midpoints between these, so our reading can vary a bit. We'll pick the first value that's >= our reading.

```

=====
RAIN GAUGE
=====

```

Not implemented here. Hey. I live in Seattle. It's ALWAYS raining. Who

cares how much?

Okay, it would probably be done the same way as the anemometer, and use

attachInterrupt(1, ...) on pin 3. Each interrupt represents .011 inches of rain, according to the docs.

```

*****
**/

```

```

#define uint unsigned int
#define ulong unsigned long

```

```

#define PIN_ANEMOMETER 2 // Digital 2
#define PIN_VANE 5 // Analog 5

```

```

// How often we want to calculate wind speed or direction
#define MSECS_CALC_WIND_SPEED 5000
#define MSECS_CALC_WIND_DIR 5000

```

```

volatile int numRevsAnemometer = 0; // Incremented in the interrupt
ulong nextCalcSpeed; // When we next calc the wind speed

```

```

ulong nextCalcDir;                // When we next calc the
direction
ulong time;                       // Millis() at each start of
loop().

// ADC readings:
#define NUMDIRS 8
ulong   adc[NUMDIRS] = {26, 45, 77, 118, 161, 196, 220, 256};

// These directions match 1-for-1 with the values in adc, but
// will have to be adjusted as noted above. Modify 'dirOffset'
// to which direction is 'away' (it's West here).
char *strVals[NUMDIRS] = {"W","NW","N","SW","NE","S","SE","E"};
byte dirOffset=0;

//=====
// Initialize
//=====
void setup() {
    Serial.begin(9600);
    pinMode(PIN_ANEMOMETER, INPUT);
    digitalWrite(PIN_ANEMOMETER, HIGH);
    attachInterrupt(0, countAnemometer, FALLING);
    nextCalcSpeed = millis() + MSECS_CALC_WIND_SPEED;
    nextCalcDir   = millis() + MSECS_CALC_WIND_DIR;
}

//=====
// Main loop.
//=====
void loop() {
    time = millis();

    if (time >= nextCalcSpeed) {
        calcWindSpeed();
        nextCalcSpeed = time + MSECS_CALC_WIND_SPEED;
    }
    if (time >= nextCalcDir) {
        calcWindDir();
        nextCalcDir = time + MSECS_CALC_WIND_DIR;
    }
}

//=====
// Interrupt handler for anemometer. Called each time the reed
// switch triggers (one revolution).
//=====
void countAnemometer() {

```

```

        numRevsAnemometer++;
    }

//=====
// Find vane direction.
//=====
void calcWindDir() {
    int val;
    byte x, reading;

    val = analogRead(PIN_VANE);
    val >>=2;                      // Shift to 255 range
    reading = val;

    // Look the reading up in directions table. Find the first value
    // that's >= to what we got.
    for (x=0; x<NUMDIRS; x++) {
        if (adc[x] >= reading)
            break;
    }
    //Serial.println(reading, DEC);
    x = (x + dirOffset) % 8;    // Adjust for orientation
    Serial.print("  Dir: ");
    Serial.println(strVals[x]);
}

//=====
// Calculate the wind speed, and display it (or log it, whatever).
// 1 rev/sec = 1.492 mph
//=====
void calcWindSpeed() {
    int x, iSpeed;
    // This will produce mph * 10
    // (didn't calc right when done as one statement)
    long speed = 14920;
    speed *= numRevsAnemometer;
    speed /= MSECS_CALC_WIND_SPEED;
    iSpeed = speed;           // Need this for formatting below

    Serial.print("Wind speed: ");
    x = iSpeed / 10;
    Serial.print(x);
    Serial.print('.');
    x = iSpeed % 10;
    Serial.print(x);

    numRevsAnemometer = 0;    // Reset counter
}

```

}