Controller de ventilator pentru PC in functie de temperatura

Introducere

Proiectul consta intr-un sistem care masoara temperatura dintr-o zona a carcasei unui PC si modifica viteza unui ventilator in functie de temperatura citita. Practic, vreau sa fac un controller simplu de ventilator, separat de placa de baza, care sa poata porni ventilatorul mai tare atunci cand temperatura creste.

Ideea a pornit de la faptul ca ventilatoarele de PC pot fi destul de zgomotoase daca merg mereu la turatie mare, dar daca merg prea incet pot sa nu raceasca suficient. Din cauza asta, vreau sa controlez ventilatorul prin PWM, in functie de temperatura masurata de un senzor.

Fata de un termometru simplu care doar afiseaza temperatura sau da o alarma, proiectul meu va incerca sa si reactioneze la temperatura, prin modificarea turatiei ventilatorului. Ma gandesc sa iau in calcul nu doar temperatura curenta, ci si cat de repede creste temperatura, ca ventilatorul sa porneasca mai agresiv daca sistemul incepe sa se incalzeasca repede.

Proiectul mi se pare util deoarece:

- poate reduce zgomotul ventilatorului cand temperatura este mica;
- poate creste racirea cand temperatura devine mai mare;
- este un exemplu practic de folosire a unui senzor analogic si a unui semnal PWM;
- poate fi extins ulterior cu mai multi senzori sau cu afisarea turatiei ventilatorului.

Descriere generala

Schema bloc aproximativa:

Pentru masurarea temperaturii voi folosi un termistor NTC. Acesta va fi pus intr-un divizor de tensiune, iar Arduino va citi tensiunea rezultata pe un pin analogic. Din acea valoare se va aproxima temperatura.

Ventilatorul nu va fi alimentat direct din Arduino, deoarece are nevoie de 12V si de un curent mai mare decat poate oferi un pin de microcontroller. De aceea, voi folosi un MOSFET sau un modul MOSFET, care va functiona ca un intrerupator controlat de Arduino. Arduino va trimite un semnal PWM, iar MOSFET-ul va controla alimentarea ventilatorului.

Pentru partea de feedback, voi folosi cateva LED-uri:

- verde pentru temperatura normala;
- galben pentru temperatura medie;
- rosu pentru temperatura ridicata.

Daca mai am timp, as vrea sa adaug si un display pe care sa afisez temperatura curenta sau procentul de PWM trimis catre ventilator.

Functionarea generala va fi:

- Arduino citeste temperatura de la senzor;
- calculeaza o valoare pentru viteza ventilatorului;
- trimite semnal PWM catre MOSFET;
- LED-urile indica nivelul aproximativ al temperaturii.

O idee initiala pentru controlul ventilatorului este sa folosesc o functie de forma:

```
PWM = valoare_minima + k * (temperatura - prag)
```

Ulterior pot modifica functia astfel incat sa tina cont si de rata de crestere a temperaturii.

Hardware Design

Lista initiala de componente:

Componenta	Cantitate	Rol
Arduino UNO / placa compatibila	1	Controleaza proiectul
Breadboard	1	Pentru prototip
Termistor NTC 10kΩ	1	Senzor de temperatura
Rezistor 10kΩ	1	Pentru divizorul de tensiune al termistorului
Ventilator 12V DC	1	Elementul de racire
MOSFET N-channel logic-level sau modul MOSFET	1	Controleaza ventilatorul prin PWM
Dioda 1N4007 / 1N5819	1	Protectie pentru ventilator
Rezistor 220Ω / 330Ω	1	Pentru poarta MOSFET-ului
Rezistor 10kΩ	1	Pull-down pentru poarta MOSFET-ului
LED verde	1	Temperatura normala
LED galben	1	Temperatura medie
LED rosu	1	Temperatura ridicata
Rezistoare 220Ω	3	Pentru LED-uri
Fire jumper	20+	Conexiuni intre Arduino, breadboard si module
Sursa 12V	1	Alimentare ventilator
Cablu USB	1	Programare / alimentare Arduino

Conexiuni propuse:

Pin Arduino	Componenta

A0	Divizorul de tensiune cu termistor
D9	Semnal PWM pentru MOSFET
D10	LED verde
D11	LED galben
D12	LED rosu

Partea de termistor va fi realizata cu un divizor de tensiune. Un capat al termistorului va fi legat la 5V, celalalt intr-un punct comun cu rezistorul de 10kΩ, iar acel punct va fi citit pe A0. Celalalt capat al rezistorului va fi legat la GND.

Pentru ventilator, Arduino va controla doar poarta MOSFET-ului. Ventilatorul va fi alimentat separat la 12V, iar MOSFET-ul va comuta conexiunea catre masa. Masa sursei de 12V trebuie legata cu masa Arduino, ca semnalul PWM sa aiba aceeasi referinta.

Nu am stabilit inca exact modelul final al MOSFET-ului, dar voi folosi unul logic-level sau un modul MOSFET compatibil cu Arduino.

Software Design

Descrierea codului aplicatiei (firmware):

- mediu de dezvoltare (if any) (e.g. AVR Studio, CodeVisionAVR)
- librarii si surse 3rd-party (e.g. Procyon AVRlib)
- algoritmi si structuri pe care planuiti sa le implementati
- (etapa 3) surse si functii implementate

Rezultate Obtinute

Care au fost rezultatele obtinute in urma realizarii proiectului vostru.

Concluzii

Download

O arhiva (sau mai multe daca este cazul) cu fisierele obtinute in urma realizarii proiectului: surse,

scheme, etc. Un fisier README, un ChangeLog, un script de compilare si copiere automata pe uC creaza intotdeauna o impresie buna .

Fisierele se incarca pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul in care se incarca fisierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (daca este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteti avea si o sectiune de jurnal in care sa poata urmari asistentul de proiect progresul proiectului.

Bibliografie/Resurse

Lista cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** si **Resurse Hardware**.

Export to PDF