©3 arzack

Fichier Édition Affichage Rechercher Terminal Aide

Les Universités du Tetal@b

/media/user/UBACK/arzack>


Des
Proto
(pseudo)
Threads
pour
l'Arduino
Proto-
Threads
ATMEGA328P U
Func-
tions

Proto Thread : mise en pratique

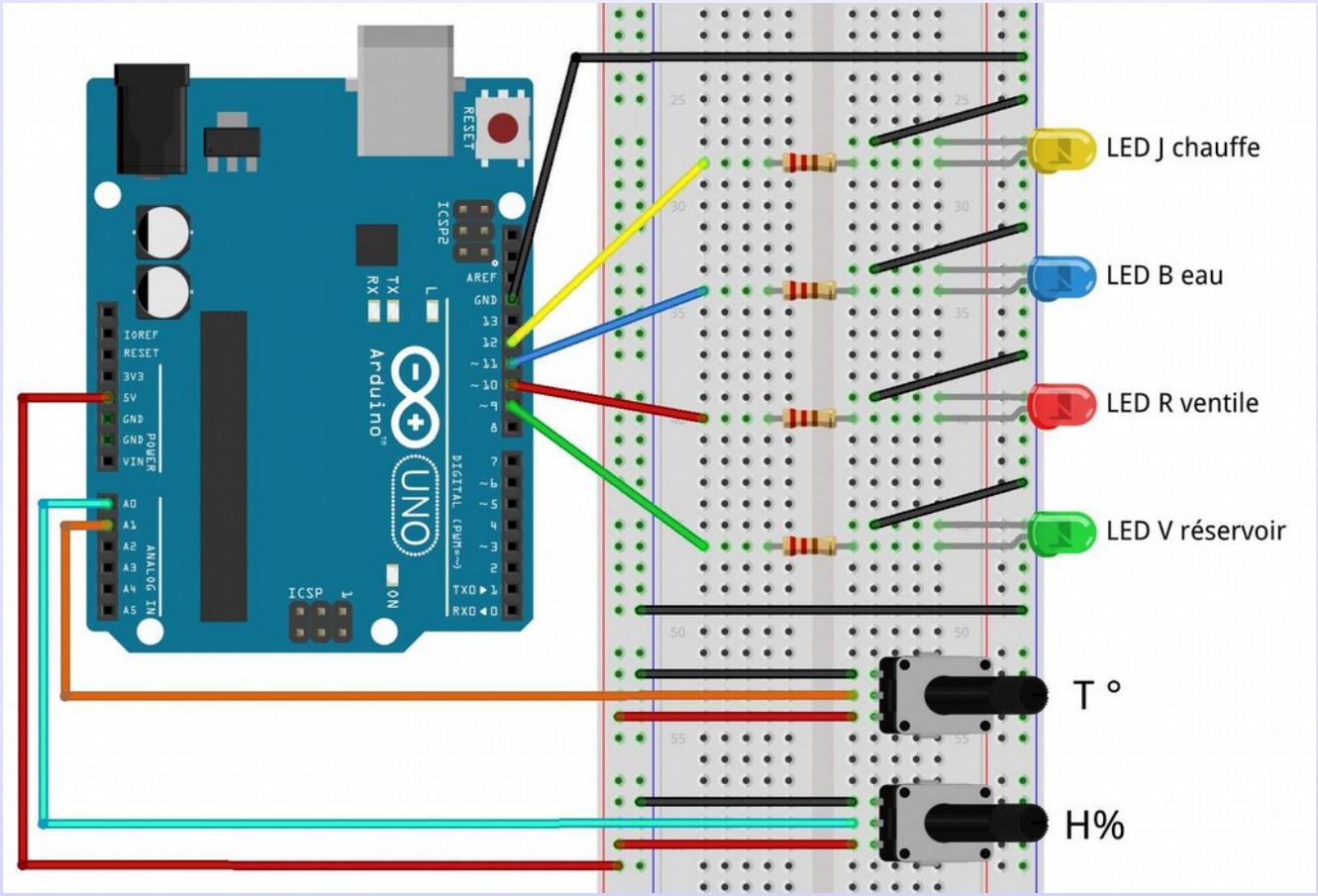
- Vous avez compris la 1ére partie ?
- Alors rien ve vaut une mise en œuvre !
- Nous allons simuler la gestion d'une µ serre

Mise en pratique 1

- Nous utilisons 4 LEDs
  - La LED jaune signale le chauffage.
  - La LED bleu signale l'arrosage.
  - La LED Rouge en mode PWM signale le % du ventilateur.
  - La LED Verte signale le niveau du réservoir.
    Débute à 100 % puis baisse de luminosité à chaque arrosage.
    Quand le réservoir est vide elle clignotement rapide et la LED bleu reste éteinte. //? et la LED Rouge passe à 100%.
- Et 2 potentiomètres
  - Un pour simuler la variation d'humidité H% sur A0.
  - Un pour simuler la variation de température T° sur A1.

Mise en pratique 2

- Montage :



Mise en pratique 3

- Spécification de la simulation
  - Si la température est < 15°C, un cycle de chauffe.
  - Si la température > 25 °C, le ventilateur tourne en proportion.
  - Si l'Humidité < 50%, un cycle arrosage.
  - Si l'Humidité >70 %, le ventilateur tourne en proportion.
  - On calibre le potentiomètre T° de 0V à 5V pour 0° à 50°.
  - On calibre le potentiomètre H% de 0V à 5V pour 30% à 100%.
  - Le réservoir est vide après 100 cycles d'arrosage.
  - Un cycle (chauffe ou arrosage) fait 5sec ON puis 20sec OFF .
  - Le RESET remet en conditions initiales.

Mise en pratique 4

- Un peu de réflexion !
  - Combien de Protothreads ?
  - Comment synchroniser les opérations dans un cycle ?
  - Quels paramètres vont contrôler chaque led ?
  - Quelles sont les constantes / paramètres globaux ?
  - Quelles dépendances entre eux ?
  - Qu'est-ce qui pose problème ?
- Vous avez des idées ?
- Allons-y !

Mise en pratique 5

- Combien de Proto Threads (PT) ?
  - Il y a 4 actions et 2 potentiomètres => 6 PT ?
  - On aurait pu regrouper les mesures dans 1 seul PT
  - Il y a 2 modes pour la LED Verte => 7 PT ?
- Quels sont leurs paramètres ?
  - Cadence d'acquisition pour T° et H%.
  - Les temps actifs / inactifs pour Jaune & Bleu.
  - La 1/2 période de clignotement de Verte.
  - On peut alors écrire une boucle principale.

Mise en pratique 6

- Boucle principale avec 7 PT

```
void loop() {
  ptTemp(&ptT,200);        // acquisition t° chaque 200ms
  ptHumi(&ptH,500);       // acquisition % chaque 500ms
  ptVenti(&ptV);              // pilotage ventilation (LED Rouge)
  ptChauffe(&ptC,3,3);    // pilotage chauffage de 3s et
inactivité de 3s (LED Jaune)
  ptEau(&ptE,3,5);           // pilotage arrosage de 3s et
inactivité de 5s (LED Bleu)
  ptNiveau(&ptN) ;          // affichage niveau d'eau, PWM Verte
de 254 à 0 (LED Verte)
  ptAlarme(&ptA,500);    // alarme réserve vide , clignote à 1Hz
(LED Verte)
}
```

Mise en pratique 7

- Variables globales et synchronisation
  - Les conditions température et humidité sont mise à jour par les PT ptTemp & ptHumi et sont connues des autres PT par l'intermédiaire des variables temp & humi. L'arrosage incrémente le compteur global nb_cycles et poste le booléen global alarme.
- Ce qui donne dans le code :

```
#include <pt.h>
static struct pt ptT, ptH, ptV, ptC, ptE, ptN, ptA ; // proto threads
int temp=0; int humi=0;
Bool alarme = false ; int nb_cycles = 0 ;
```

Mise en pratique 8

- Le décor est en place, voyons les acteurs
- Les acquisitions :

```
static int ptTemp(struct pt *pt, int repit) {
static unsigned long ttl = 0;
  PT_BEGIN(pt);
  while(1) {
    ttl = millis() + repit ;
    PT_WAIT_UNTIL(pt, millis() > ttl);
    temp = acquis(A1, 0, 50);  // pour ptHumi on a : humi = acquis(A0,  30,
100);
  }
  PT_END(pt);
}
```

Mise en pratique 9

- Le PT ptAlarme clignotant :

```
static int ptAlarme (struct pt *pt, int duree)
{
  Static unsigned long = ttl ;
  PT_BEGIN(pt);
  while ( alarme ) {
    ttl = millis + duree ;
    PT_WAIT_UNTIL(pt, millis() > ttl);
    digitalWrite(VERTE, !digitalRead(VERTE));
  }
  PT_END(pt);
}
```

Mise en pratique 10

- Le PT ptNiveau du niveau d'eau,
  - avec 100 cycles pour vider le réservoir :

```
static int ptNiveau(struct pt *pt)
{
  static int nb_old = 0; int ratio;
  BEGIN(pt);
   while ( ! alarme) {
    nb_old = nb_cycles;
    ratio = map(nb_cycles, 0, 100, 254, 0);
    ledPWM(VERTE, ratio);
    PT_WAIT_UNTIL(pt, nb_cycles > nb_old ); // attente nouvel arrosage
  }
  PT_END(pt);
}
```

Mise en pratique 11

- Le PT ptChauffe :

```
static int ptChauffe(struct pt *pt,  uint16_t actif, uint16_t inactif )
{
static unsigned long ttl = 0;
BEGIN(pt);
   while ( 1) {
    PT_WAIT_UNTIL(pt,  temp < 15);
    digitalWrite(JAUNE, HIGH);
    ttl = millis() + actif ;
    PT_WAIT_UNTIL (pt, millis() > ttl);
    digitalWrite(JAUNE, LOW);
    ttl = millis() + inactif ;
    PT_WAIT_UNTIL (pt, millis() > ttl);
  }
  PT_END(pt);
} // Chauffe
```

Mise en pratique 12

- Le PT ptEau :

```
static int ptEau(struct pt *pt,  uint16_t actif, uint16_t inactif )
{
static unsigned long ttl = 0;
BEGIN(pt);
   while ( ! alarme) {
    PT_WAIT_UNTIL(pt,  humi < 50);
    digitalWrite(ROUGE, HIGH);
    ttl = millis() + actif ;
    nb_cycles++;
    alarme = ( nb_cycles > 100);
    PT_WAIT_UNTIL (pt, millis() > ttl);
    digitalWrite(ROUGE, LOW);
    ttl = millis() + inactif ;
    PT_WAIT_UNTIL (pt, millis() > ttl);
  }
  PT_END(pt);
} // Eau
```

Mise en pratique 13

- Le PT ptVenti :

```
static int ptVenti(struct pt *pt)
{
  static int temp_old = 0, humi_old = 0; int t_ratio=0, h_ratio = 0, i ;
  static unsigned long ttl = 0;
  PT_BEGIN(pt);
  while ( 1 ) {
    if (temp < 25 && humi < 70) digitalWrite(ROUGE, LOW);
    PT_WAIT_UNTIL(pt,  temp > 25 || humi  > 70);
    temp_old = temp;
    humi_old = humi ;
    if (temp > 25) t_ratio = map(temp, 0, 50, 0,254);
    if (humi > 70) h_ratio = map(humi, 30, 100, 0,254);
    i = max(t_ratio , h_ratio );
    ledPWM(ROUGE, i) ;
    PT_WAIT_UNTIL(pt,  temp != temp_old  || humi != humi_old );
  }
  PT_END(pt);
} // Ventilateur
```

Merci pour votre attention :-)