Axel Segogne

PRÉSENTATION DU SCHÉMA ÉLECTRIQUE ET DU PROGRAMME

sommaire

-les composants

-le schéma électrique

-le programme

Les composants

Une carte arduino

Pourquoi ?

- Une grande liberté de programmation grâce aux librairie qui sont fournies par leur équipe et la communauté.
- La maîtrise du logiciel dans arduino
- Documentation facile d'accès
- Le nombre de sorties et d'entrées est beaucoup plus important sur l'arduino que sur un zelio

P= 304mW = 0.3W

Moteur pas à pas (28BYJ-48)

Pourquoi ?

Avantages :

-Petite taille

-Facilité de programmation, doublé d'une grande précision

-Peu coûteux

Inconvénients :

-Peut soulever jusqu'à 500g(déterminé grâce au calcul)

-Ne fait pas directement un mouvement de translation, on doit passer par une crémaillère

P = 0.6W

Capteur OMRON E3Z-D82

Pourquoi ?

Avantages :

- Il permet de détecter des objets et de renvoyer un front haut quand il y en a un devant et un front bas quand il y a rien devant

Inconvénients :

- S'alimente en 12V et renvoi l'information en 12V

P = 0.036W

Vérin électrique

Pourquoi ?

Avantages :

- Il permet de faire un mouvement de translation directe et de revenir grâce au ressort

Inconvénients :

- S'alimente en 12V
- Prend de la place

Les caractéristiques minimales du vérin après test et calculs :

Course= 100mm

I = 0.8mA

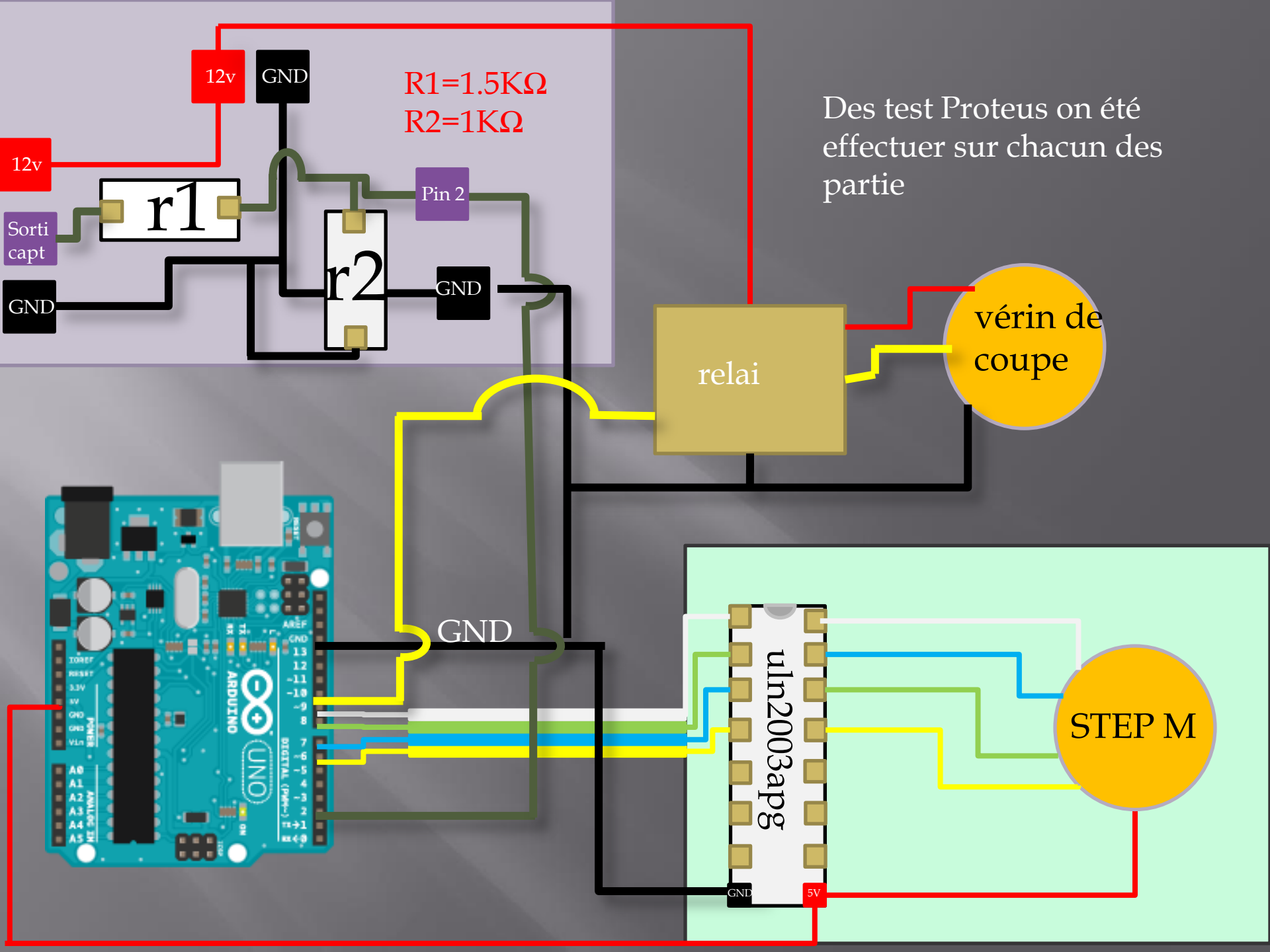
on acceptera donc les moteur > 0.2Nm

u=12V

ω=50mm/s

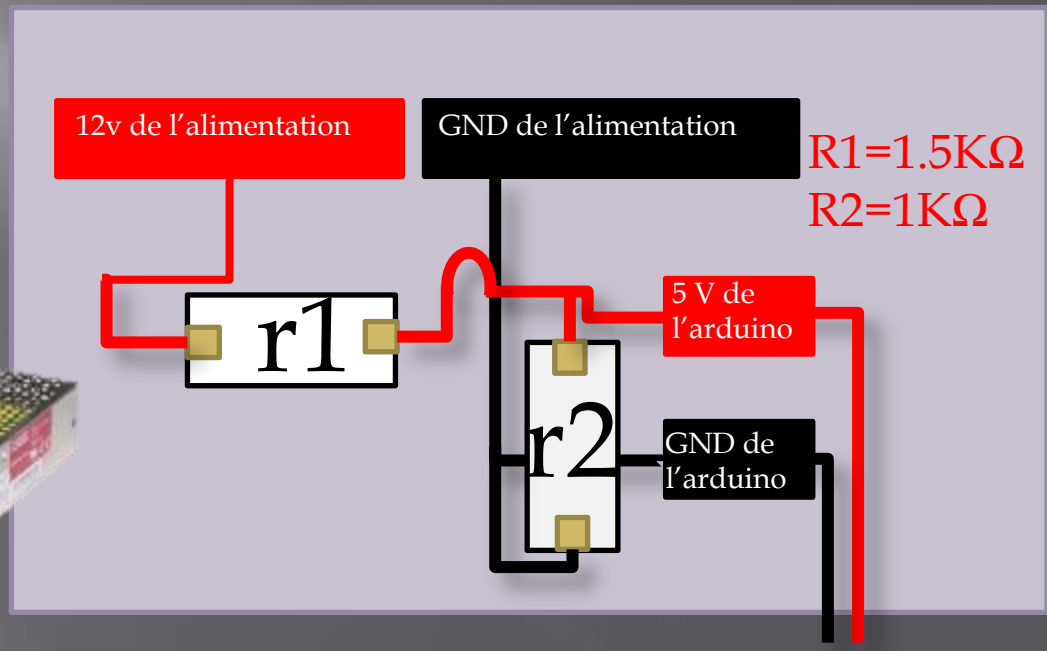
P = 0.9mW =0.0009W

LE SCHÉMA


12v
GND
R1=1.5KΩ
R2=1KΩ
12v
r1
Pin 2
Sorti capt
r2
GND
GND
Des test Proteus on été effectuer sur chacun des partie
relai
vérin de coupe
GND
uln2003apg
STEP M
GND
5V

Alimentation

la puissance totale consommée par le système est inférieure à 40W
∑P=2.16W
L'alimentation sera un TMX 015-112 car elle était disponible cette alimentation est inferieur a 15W

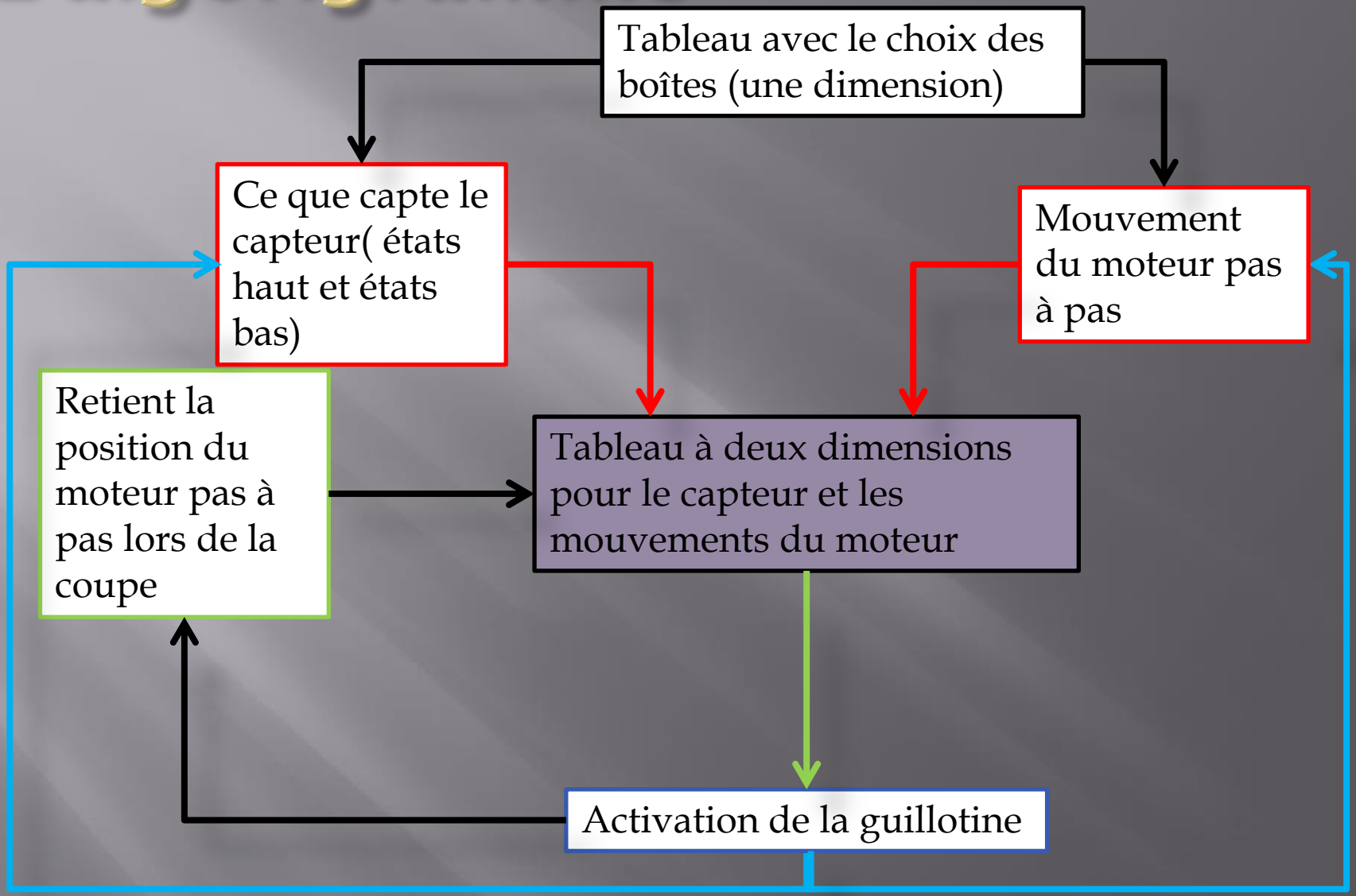


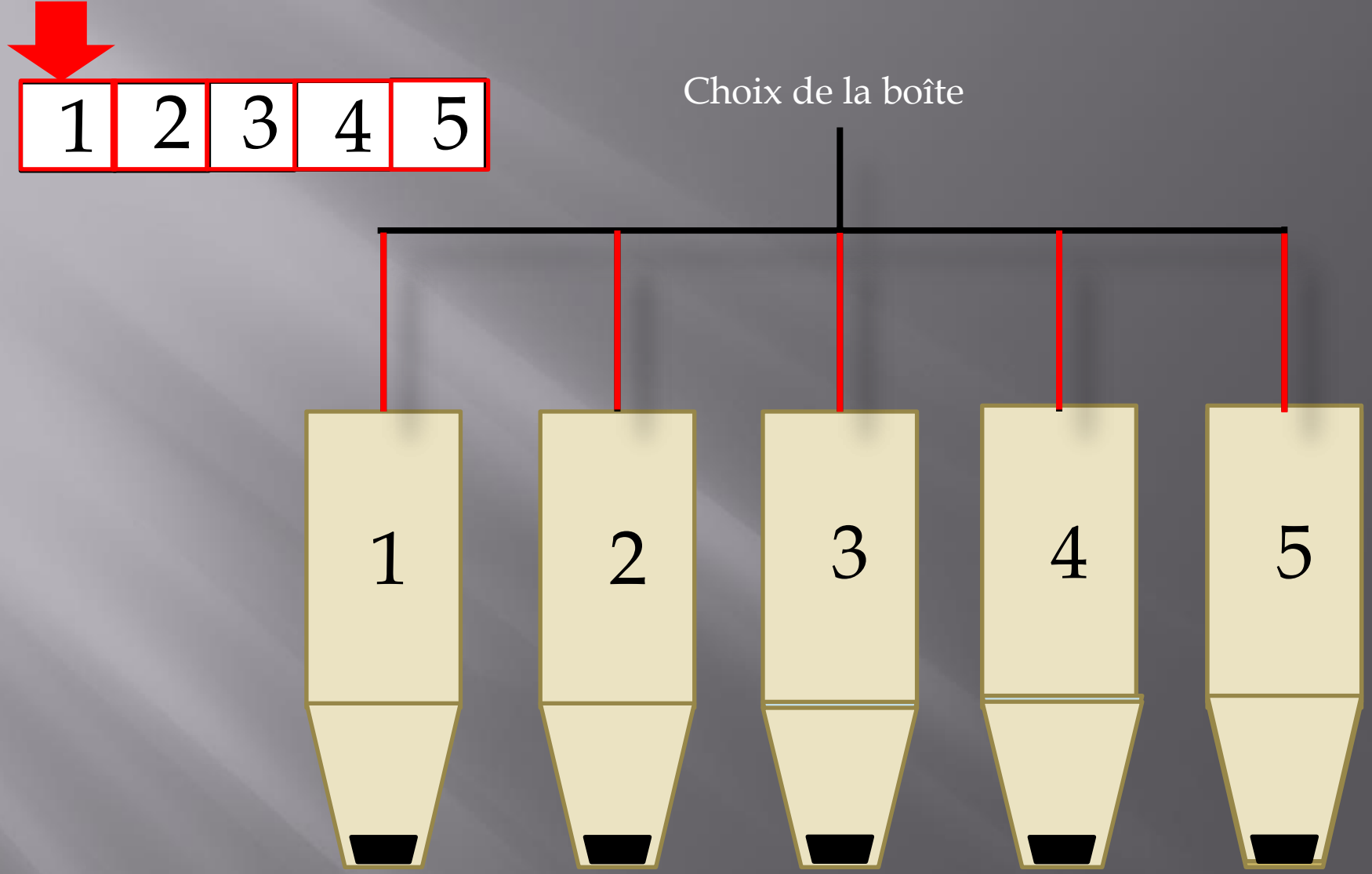
LE PROGRAMME

Le principe

- On commence par un tableau à une dimension.
- Un autre tableau à deux dimensions.
- Le principe de ce programme est donc de compter les états haut et les états bas correspondant au passage ou non d'un médicament

L'algorigramme




1 2 3 4 5
Choix de la boîte
1
2
3
4
5

Position du moteur en pas	
Captation du capteur	

Position du moteur en pas	X pas	Y pas	Z pas	A pas	B pas
Captation du capteur	Etat bas	Etat haut	Etat bas	Etat haut	Etat bas

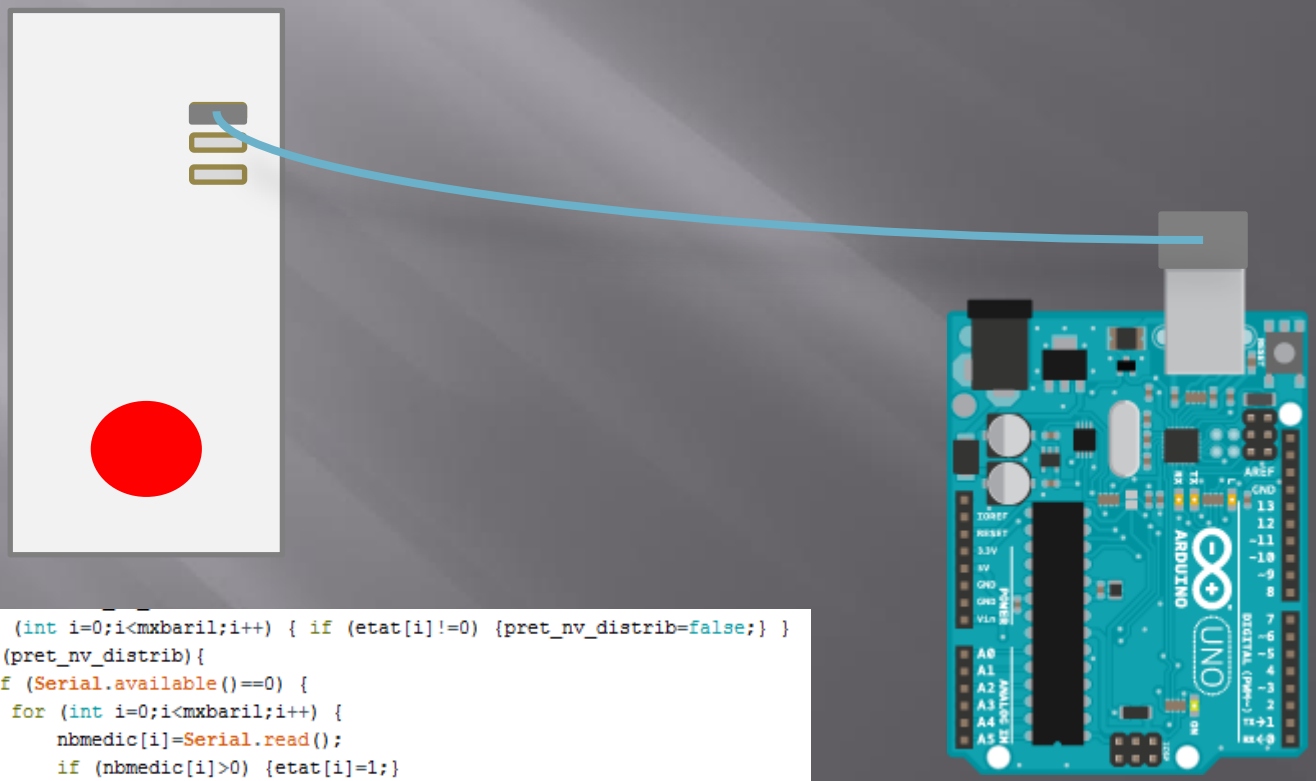
Pour activer la guillotine
Le moteur pas à pas va devoir faire :

$\frac{A+Y}{2}$=nb de pas pour amener le médicament à la découpe en continuant d'avancer

Si le moteur pas à pas arrive au maximum de pas entre le pas 0 et le nombre capteur , alors il fait le nombre de pas inverse et fait tomber une nouvelle plaquette

Liaison entre arduino et l'interface:

Pour celà on a pensé à faire une connexion sans fil par module bluetooth ou wifi or la machine est fixe et doit être près d'un ordinateur. On a donc choisi une connexion filaire.



```
for (int i=0;i<mxbaril;i++) { if (etat[i]!=0) {pret_nv_distrib=false;} }
if (pret_nv_distrib){
  if (Serial.available()==0) {
    for (int i=0;i<mxbaril;i++) {
        nbmedic[i]=Serial.read();
        if (nbmedic[i]>0) {etat[i]=1;}
    }
    numedic=1;
  }else {numedic=1000;}
}
```

Le programme

initialisation

medicpratic | Arduino 1.8.1

Fichier Édition Croquis Outils Aide

medicpratic

```
#include <Stepper.h>
#define STEPS 64
const int mxbaril=2;
const int mxespvide=20;
const int pos_capt=1561; //position juste avant que le piston arrive sur le capteur (sans le couper) a definir
const int dist_coupe=937; //cran piston entre pos_capt et la guillotine. a definie
const int maxmedic=20; //nombre maximale d'un type de medicament à la même personne a definir

int Mot_guill[mxbaril]; //broche arduino de la guillotine de chaque baril
int Capt[mxbaril]; //broche arduino du capteur de chaque baril

int nbmedic[mxbaril]; //nombre de medicament à délivrer sur chaque baril (pour un patient)
int pospist[mxbaril]; //position du piston
int frontbas[mxbaril]; //position du dernier front bas
int num_esp_vide[mxbaril]; //numero espace vide sur une plaquette
int pos_esp_vide[mxbaril][mxespvide]; //position (piston) de chaque espace vide sur une plaquette
int etat[mxbaril]; //variable d'etat pour le suivit du programme.
boolean debut_plaquette[mxbaril]; //on est on debut de la plaquette
boolean obstacle[mxbaril]; //présence actuel d'un medic face au rayon lumineux

int numedic; //distribution des medicaments par multiplicité, numedic est le suivi de la multiplicité

Stepper Moteur(STEPS,6,7,8,9);
Stepper Moteur2(STEPS,2,3,4,5);
```

Void setup

```
void setup(){
  Mot_guill[0]=10;  Capt[0]=11;
  Mot_guill[1]=13; Capt[1]=12;
  Moteur.setSpeed(5);
  Moteur2.setSpeed(5);
  for (int i=0;i<mxbaril;i++) {
    pinMode(Mot_guill[i],OUTPUT);
    digitalWrite(Mot_guill[i],LOW);
  }
  for (int i=0;i<mxbaril;i++) {
    pinMode(Capt[i],INPUT);
    etat[i]=0;
    initbaril(i);
  }
  Serial.begin(9600);

}
```

Void loop

```
void loop(){

  boolean pret_nv_distrib=true;
  for (int i=0;i<mxbaril;i++) { if (etat[i]!=0) {pret_nv_distrib=false;} }
  if (pret_nv_distrib){
    if (Serial.available()==0) {
      for (int i=0;i<mxbaril;i++) {
          nbmedic[i]=Serial.read();
          if (nbmedic[i]>0) {etat[i]=1;}
      }
      numedic=1;
    }else {numedic=1000;}
  }
  for (int i=0;i<mxbaril;i++) {
    if (nbmedic[i]>=numedic) {Medicsuiv(i);} //si le baril i doit distribuer un numedic_ieme medicament alors il execute
  }
  boolean pret_prochain_numedic=true;
  for (int i=0;i<mxbaril;i++) { if (!(etat[i]==-1||etat[i]==0)) {pret_prochain_numedic=false;} }
  if (pret_prochain_numedic) {
    if (numedic<1000) {numedic++;}
    for (int i=0;i<mxbaril;i++) {
       if (nbmedic[i]>=numedic) {etat[i]=1;}
       else {etat[i]=0;}
    }
  }
}
```

Void medicsuiv

```
void Medicsuiv(int b){
// distribue un medicament sur le baril
// on avance d'un pas jusque ce qu'on  trouve une position de coupe sur la plaquette
//pendant ce temps on memorise les positions de coupe ulterieures
  if (etat[b]==1){
    if (!(
          (debut_plaquette[b]|| (pospist[b]<pos_esp_vide[b][num_esp_vide[b]+1]+dist_coupe))
            &&(pospist[b]<pos_capt+dist_coupe)
       )){etat[b]=2;}
    else{
      avance_un_cran(b);
      pospist[b]=pospist[b];
      if (pospist[b]<pos_capt){
        //test les fronts haut et bas jusqu'à arrivée du piston au capteur
        //après initule de tester (plaquette finie)
        if ((debut_plaquette[b])&&testobst(b)) {
          obstacle[b]=true;
          debut_plaquette[b]=false;
        }
        if ((obstacle[b])&&!testobst(b)) {
          frontbas[b]=pospist[b];
          obstacle[b]=false;
        }
        if(!(debut_plaquette[b])&&!(obstacle[b])&&(testobst(b))) {
          num_esp_vide[b]=num_esp_vide[b]+1;
          pos_esp_vide[b][num_esp_vide[b]]=(pospist[b]+frontbas[b])/2;
        }
      }
    }
  }
  if (etat[b]==2) {
      if (pospist[b]>=pos_capt+dist_coupe) {
          retour_piston(b);
          initbaril(b);
      }else{
        guillottine(b);
      }
      etat[b]=-1;
  }
}

void initbaril(int b){
  for (int n=0;n<mxespvide;n++){
    for (int i=0;i<mxbaril;i++) { pos_esp_vide[b][i]=100000;}
  }
  debut_plaquette[b]=true;
  pospist[b]=0;
  frontbas[b]=0;
  obstacle[b]=false;
}
```