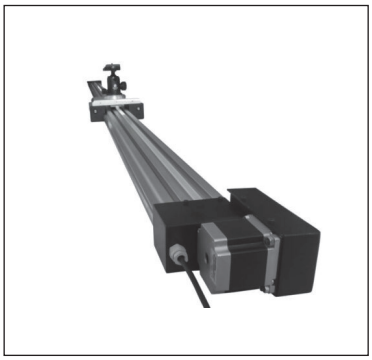
Dossier technique : rail de travelling



1. Découverte du produit et de la problématique technique

Le système d'étude proposé est un rail de travelling pour réaliser des photos en time lapse ou des vidéos avec un déplacement lent pour éviter les plans fixes et donner du dynamisme.

Le déplacement est linéaire dans le cas étudié et le programme est très simplifié : il suffit d'entrer la valeur de position souhaitée mais cela nécessite de positionner le chariot en butée côté moteur pour le démarrage du mouvement.

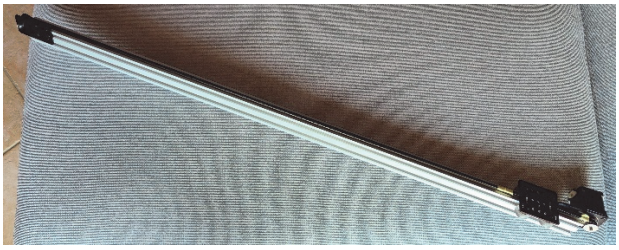


Figure 1 : chariot en position initiale

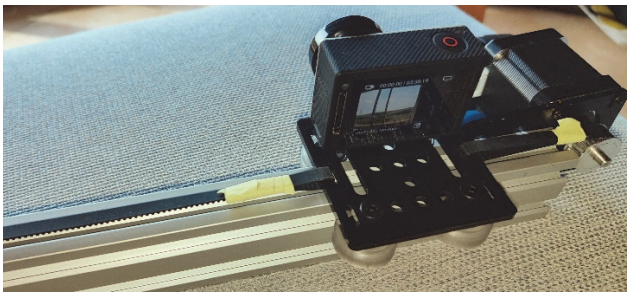


Figure 2 : prêt pour la prise de vue en position initiale

Dans le système étudié :

- Un microcontrôleur pilote un moteur pas à pas pour effectuer un mouvement de translation à un appareil de prise de vue. Ceci via un ensemble poulies-courroie.

- Ce même microcontrôleur reçoit par sa liaison série l'instruction de déplacement vers une position sur le rail. La position est exprimée en pas.

Remarque :

Le système doit obligatoirement réaliser sa prise de position initiale (POM) à la mise sous tension afin d'éviter de se retrouver en butée en bout de rail.

Si cette opération préalable n'est pas effectuée, la partie opérative (poulies-courroie) peut être endommagée, ainsi que le matériel embarqué.

a. Caractéristiques

- Dimensions du rail : 107 cm × 2 cm × 4 cm.
- Longueur utile de déplacement du chariot : 84 cm.
- Moteur 200 pas par tours piloté en 1/8 de pas.
- Poulie 30 dents de 2 mm d'espacement.
- Position envoyée par liaison série.
- Alimentation sur une batterie lithium ion 2s (7,2v) ou par alimentation secteur.

b. Cahier des charges

Modifications à apporter :

- Paramétrage de la Prise Origine Machine (position initiale du chariot) :
  - Déplacement lent vers le capteur situé près du moteur.
  - Lorsque le capteur est déclenché, faire un déplacement inverse très lent pour tout juste relâcher le capteur.
  - Enregistrement de la position 0 à l'instant où le capteur est relâché.
- Limiter la saisie d'une position entre 0 et 22400, 22400 étant le nombre maximal de micro pas faisables sur ce rail.

2. Conception

Le capteur employé pour la prise de position initiale est de type tout ou rien. Il s'agit d'un capteur de type micro switch avec un connecteur Grove pour faciliter son brachement sur la carte Arduino.

Le fichier numérique de simulation à utiliser est « `9 Rail Travelling DRV8825 Homing Serie.pdsprj` ».

Exemple de déclaration d'une variable pour un capteur dans Proteus ou Arduino :

int monCapteur = 2 ;

Cette ligne indique que l'on déclare un capteur dont la broche s'appelle monCapteur et se trouve sur l'entrée numérique 2 de l'Arduino.

Schéma structurel pour la simulation du comportement.

Sur le schéma suivant (figure 3) :

- Le capteur est présent sur le schéma et connecté par défaut sur la broche D7 de l'Arduino. Le branchement en D7 peut être modifié.
- Un terminal de liaison série permet de simuler le comportement et d'entrer la valeur de position souhaitée, des messages décrivent le comportement du système. Un afficheur LCD peut être utilisé.

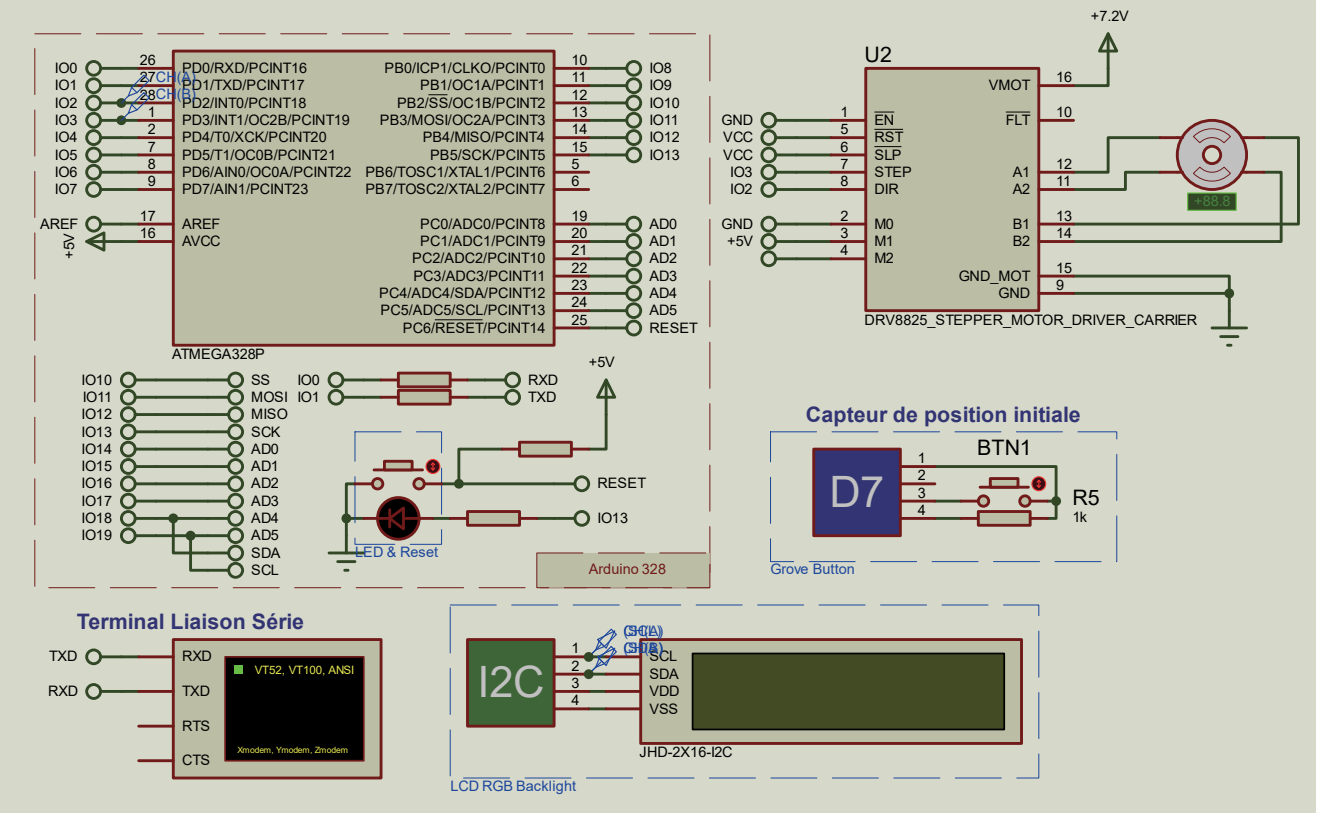


Figure 3 : schéma structurel pour la simulation

3. Simulation

La simulation du comportement du système est menée avec le logiciel Proteus.

Le driver et la rotation du moteur ne sont pas simulés ici, c'est la validation du comportement du système qui est recherchée.

Le chariot part lentement d'une position 0 en direction du capteur. Lorsque le capteur est déclenché (appuyé) un message doit l'indiquer. Le chariot doit repartir très lentement en sens inverse pour relâcher le capteur et s'arrêter aussitôt. Un message indiquera que la position initiale a été trouvée. C'est la nouvelle position 0.

Ensuite entrer différentes valeurs de positions via la liaision série et vérifier que seules les positions dans les limites de la valeur maximum calculée sont acceptées.

Extraits du programme existant incomplets

- Partie déclaration du capteur :

```
#include <AccelStepper.h>

const int dirPin = 2;
const int stepPin = 3;
AccelStepper stepper(AccelStepper::DRIVER, stepPin, dirPin);

// COMPLETER Declaration du capteur :

                              // switch en position initiale sur la broche 7


long Position;
int deplacement_fait = 1;
long position_init = -1;

void setup() {
  Serial.begin(9600);
  pinMode(home_switch,INPUT);
  delay(5);

  stepper.setMaxSpeed(800);    // Deplacement lent pour atteindre le home_switch
  stepper.setAcceleration(800);
  Serial.println("Vers position initiale...     ");
    delay(500);
  while(!digitalRead(home_switch)) {   // deplacement vers le home_switch
    stepper.moveTo(position_init);
    position_init--;
    stepper.run();

  }
```

- Partie relâchement du capteur pour prise de position initiale (le zéro) :

```
  Serial.println("Capteur atteint ! Relachement...");
  stepper.setCurrentPosition(0); // Enregistre la position declenchant le switch
  delay(500);


  stepper.setMaxSpeed(50);
  stepper.setAcceleration(10);
  position_init = 1;  // Pour relacher le switch : repartir dans le sens inverse

  while(                        ) { // COMPLETER  : Recherche de la position limite relachant le switch




  }

  stepper.setCurrentPosition(0); // Enregistre la position initiale
  Serial.println("Position initiale atteinte.");
  Serial.println("  ");
  stepper.setMaxSpeed(5000);  // pour des deplacements plus rapides en fonctionnement normal
  stepper.setAcceleration(2000);

} // Setup fini et en position initiale
```

- Partie saisie du déplacement à effectuer :

```
void loop() {
  while(Serial.available()>0) {
    deplacement_fait=0;

    Position = Serial.parseInt(); // Recupere le nombre de pas de deplacement demande

    if(                              ) { // COMPLETER la condition : Verifier si la position est hors du rail
      Serial.println(Position);
      Serial.println(" en dehors du rail... Nouvelle position ?");
    }
    else {
      Serial.print("Deplacement vers la position ");
      Serial.println(Position);
      stepper.moveTo(Position); // Entre la position dans Accelstepper
      delay(500);              // Et y va apres une demie seconde
    }
  }

  if( Position>=0 && Position<=(             ) ) { //COMPLETER avec la position maximale du rail utilisable
    if((stepper.distanceToGo() !=0 )) {  // si position non atteinte
      stepper.run();            // Deplace le chariot vers la position
    }

    if((deplacement_fait==0) && (stepper.distanceToGo() == 0)) {
      Serial.println("Position atteinte.");
      Serial.println("-------------------");
      Serial.println("Entrer la nouvelle position.");
      deplacement_fait = 1;
    }
  }
}
```

Exemple de résultat attendu pour une simulation :

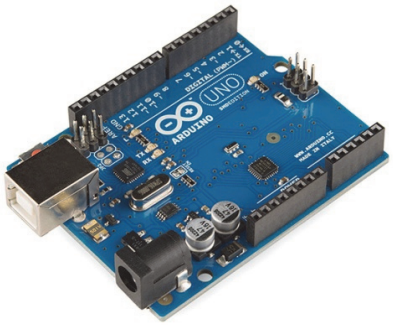
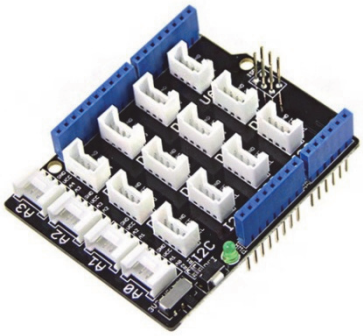
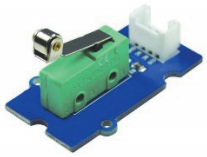
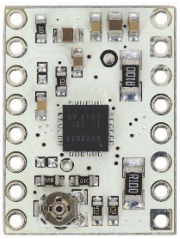
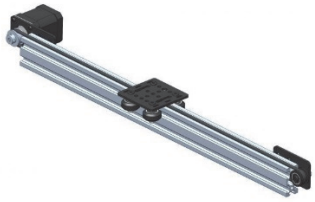
```
Legacy Virtual Terminal
Vers position initiale...
Capteur atteint ! Relachement...
Position initiale atteinte.

Deplacement vers la position 53
Position atteinte.
-------------------
Entrer la nouvelle position.
25300
 en dehors du rail... Nouvelle position ?
Deplacement vers la position 2100
Position atteinte.
-------------------
Entrer la nouvelle position.
Deplacement vers la position 0
Position atteinte.
-------------------
Entrer la nouvelle position.
```

4. Expérimentation

Pour effectuer l'expérimentation, le matériel suivant est à disposition :

- une carte de développement Arduino ;
- un driver de moteur pas à pas (par exemple DRV8834) ;
- un ensemble rail moteur poulies courroie chariot ;
- un capteur tout ou rien à galet ;
- une alimentation / batterie ;

 Carte Arduino Uno R3	 Shield base Grove
 Capteur tout ou rien Grove	 Driver pour moteur pas à pas
 Ensemble rail moteur poulies courroie chariot	 Batterie 7.2V ou alimentation