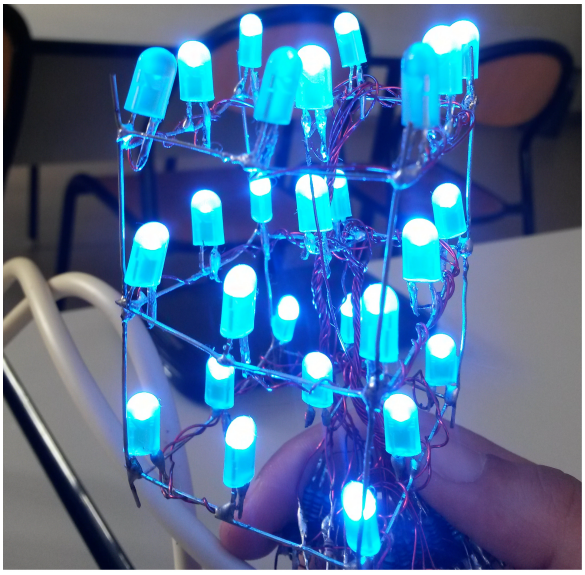
Compte rendu projet électronique

Julia OUZZINE - Pierre-Emmanuel NOVAC

29 mai 2016

1 Introduction

Le but final est de réaliser un cube de $3 \times 3 \times 3$ LEDs avec l'Arduino, contrôlable par Bluetooth et répondant à un changement de son environnement.



2 Fonctionnalités

- Contrôle indépendant de 27 DELs
- 4 animations préprogrammées
- Choix de l'animation à l'aide d'une application Android (transmission Bluetooth)
- Détection de l'orientation et réaction en conséquence : lorsque le cube change d'orientation, la face du bas s'allume
- Interaction avec l'environnement : capteur de distance, lorsqu'un obstacle est détecté, toutes les LEDs s'allument

Initialement, un capteur de contact était prévu. Face à la difficulté pour trouver des ressources sur la réalisation d'un tel capteur sur une surface transparente, nous avons préféré nous rabattre sur un capteur de distance que nous maitrisions déjà.
Une possibilité pour ce capteur de contact serait d'utiliser l'effet capacitif du corps humain sur une plaque de verre recouverte d'ITO. Un premier essai n'a cependant pas été concluant et c'est un matériau coûteux.

3 Montage

3.1 Matériel

27 LEDs bleues Nous n'en connaissons pas les caractéristiques exactes, mais des mesures révèlent une tension de seuil d'environ 2,9V, et nous ne dépasserons pas un courant de 20mA.

27 résistances de 120Ω Les résistances serviront à limiter le courant circulant dans les LEDs. Avec 120Ω, on obtient un courant d'environ 18mA.

Arduino L'Arduino Uno, utilisant un microcontrôleur ATmega328, sera utilisé. Il contrôlera l'allumage des LEDs tout en traitant les informations reçues des capteurs et par Bluetooth.

4 registres à décalage 74HC595N Pour des questions de courant trop élevé et de nombre de sorties limitées, il n'est pas possible de relier directement les 27 LEDs à l'Arduino. Elles seront connectées à des registres à décalage 8-bit fournissant donc 8 sorties par composant.

Module Bluetooth HC-06 La communication par Bluetooth avec une tablette (sous Android) se fera grâce au module HC-06 (le mode maître n'est pas utile).

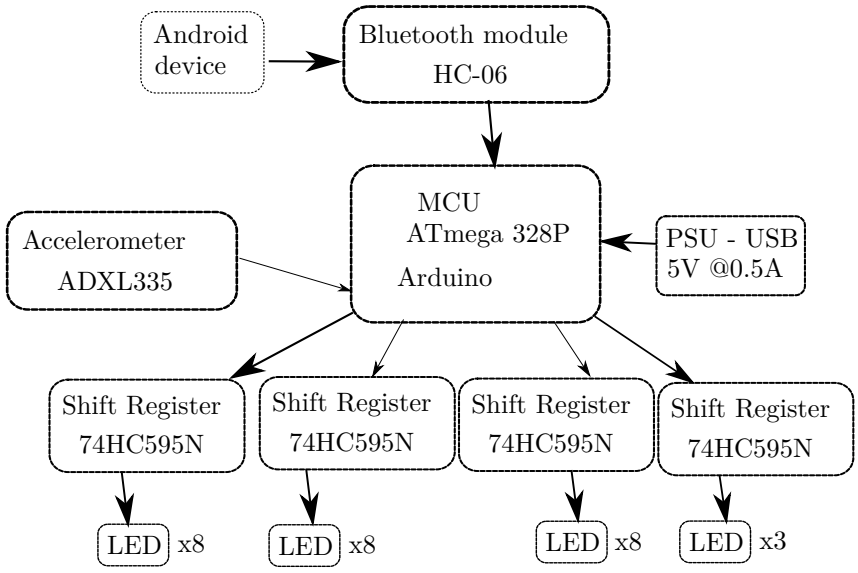
Accéléromètre ADXL335 L'accéléromètre permettra de détecter l'orientation du cube. À noter que les sorties de l'ADXL335 sont analogiques. Nous ferons en sorte d'en extraire des informations de pitch et roll.

Capteur de distance HC-SR04 C'est un capteur à ultrason qu'on utilisera pour détecter la présence d'un obstacle proche du cube.

Divers 1 résistance $1k\Omega$, 1 résistance $2k\Omega$, fil de cuivre émaillé 0,4mm (pour relier les LED aux registres à décalage), plaque de prototypage (pour connecter les modules), câbles divers

Remarque Un cable téléphonique de 8 fils a été utilisé pour connecter le cube à l'arduino. 2 fils sont utilisés pour l'alimentation, 3 pour la communication avec les registres à décalage. Les 3 autres peuvent être utilisés pour l'accéléromètre, mais il en manque 2 pour le capteur de distance.

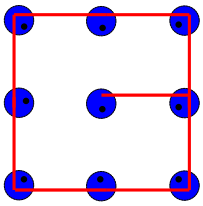
3.2 Schéma fonctionnel



3.3 Branchements

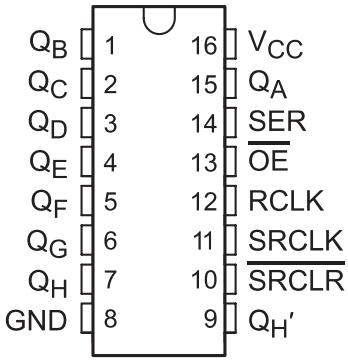
3.3.1 Cube de LED

Pour chacun des 3 étages du cube, on réalise une structure identique de 9 LED.
Les anodes des LEDs sont connectées en elles et sont à souder de manière à former un carré de 9 LEDs.

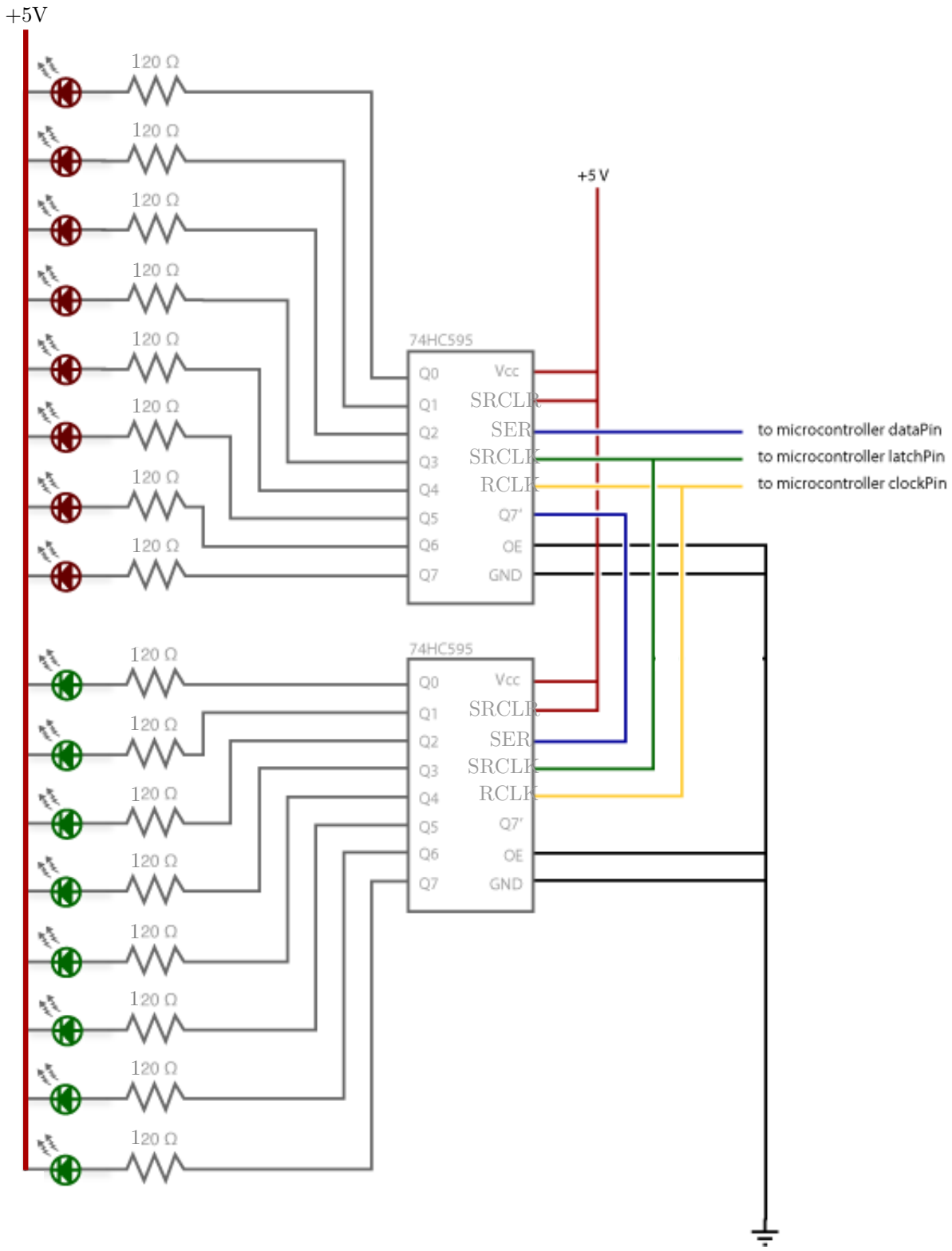


On soude ensuite un fil par cathode de chaque LED, et une résistance de 120Ω au bout de ce fil.
On superpose les 3 étages et on relie les 4 coins de chaque étage en soudant une tige métallique qui permet alors de relier les anodes des étages entre elles et de maintenir la structure.

3.3.2 Registres à décalage

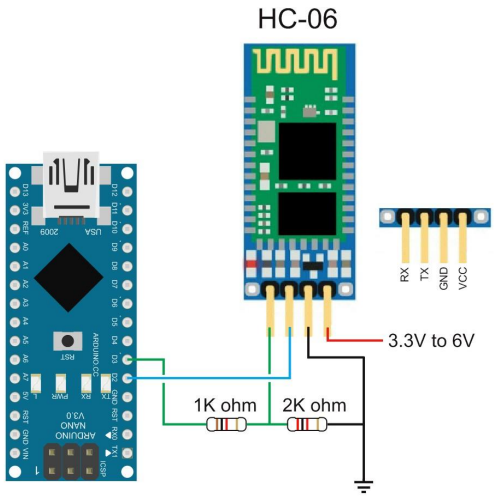


Les résistances sont à souder aux sorties Q_A à Q_H, c'est à dire pattes 15 et 1 à 7 des registres à décalage.
Les pattes V_{CC} et GND de chaque registre à décalage sont à relier respectivement à l'alimentation 5V et à la masse.
$\overline{OE}$ est à relier à GND, puisqu'on souhaite activer les sorties en permanence. $\overline{SRCLR}$ est à relier à V_{CC}, puisqu'on ne souhaite pas réinitialiser le registre.
SER, $RCLK$ et $SRCLK$ du premier registre à décalage sont à relier respectivement aux pattes 11, 12 et 8 de l'Arduino.
On connectera $RCLK$ et $SRCLK$ en parallèle sur les autres registres, et on connectera la sortie Q'_H d'un registre à l'entrée SER du suivant.
On n'oubliera pas de relier la structure (anode) des LED à l'alimentation 5V.


+5V
120 Ω
120 Ω
120 Ω
120 Ω
120 Ω
120 Ω
120 Ω
120 Ω
+5 V
74HC595
Q0
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Vcc
SRCLR
SER
SRCLK
RCLK
Q7'
OE
GND
to microcontroller dataPin
to microcontroller latchPin
to microcontroller clockPin
74HC595
Q0
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Vcc
SRCLR
SER
SRCLK
RCLK
Q7'
OE
GND
120 Ω
120 Ω
120 Ω
120 Ω
120 Ω
120 Ω
120 Ω
120 Ω

3.3.3 Module Bluetooth

Ce module est à relier à l'alimentation 5V et à la masse, la patte TX à la patte 6 de l'Arduino et la patte RX à la patte 7 de l'Arduino par l'intermédiaire d'un pont diviseur comme illustré ci-dessous.



3.3.4 Accéléromètre

Les sorties de ce capteur étant analogiques, il faut les relier aux entrées analogiques de l'Arduino : X sur A0, Y sur A1 et Z sur A2. Il devra être relié à l'alimentation 3,3V et à la masse.

3.3.5 Capteur de distance

Il suffit de le connecter à l'alimentation 5V de l'Arduino et à la masse, la patte trigger à la patte 3 de l'Arduino et la patte echo à la patte 4 de l'arduino.

4 Programmation

4.1 Programmation par interruption

Étant donnée que l'Arduino doit gérer en même temps d'une part l'allumage des LEDs et d'autre part la réception et le traitement des données des différents modules, la programmation séquentielle n'est pas adaptée. Il faut donc faire appel à de la programmation par interruption. Ici, la boucle principale du programme met à jour l'état des LED, tandis qu'une interruption appellée périodiquement (grâce au timer n°2 de l'ATmega328) se charge de traiter les données des capteurs et du Bluetooth. La routine d'interruption est définie dans le fichier principal, au même titre que la boucle principale loop() et la fonction d'initialisation setup().

4.2 Animations et contrôle des LEDs

Les animations sont programmées dans le fichier Animation.cpp. Celui-ci repose sur l'utilisation de la bibliothèque ShiftRegister74HC595 qui simplifie le contrôle des registres à décalage pour l'allumage des LEDs. Il est nécessaire de configurer le numéro des LEDs dans Animation.h.

4.3 Bluetooth

Le fichier Bluetooth.cpp gère la réception de données par Bluetooth.
On peut configurer le nom du périphérique Bluetooth et son code PIN dans Bluetooth.h.

Remarque Le fichier fournit LED_Cube.kwl peut être importé dans l'application Bluetooth Electronics pour pouvoir contrôler le cube.

4.4 Accéléromètre

Les données de l'accéléromètre sont traitées dans le fichier Accelerometer.h. Il se charge de convertir les valeurs lues sur les entrées analogiques en angles de pitch et roll. Une fonction définit le numéro de face à allumer selon l'intervalle dans lequel se trouvent les valeurs de pitch et de roll. La correspondance entre numéro de face et numéro de LED à allumer se fait à l'aide d'un tableau dans Animation.cpp
La mise à jour de ces données se fait dans la routine d'interruption et à chaque changement d'orientation, on allume la face correspondante.

4.5 Capteur de distance

Le traitement des informations du capteur de distance se fait directement dans la routine d'interruption. Il s'agit simplement d'allumer toutes les LEDs lorsque le temps de réception de la réponse est inférieur à $1600\mu s$.

5 Difficultés rencontrées

Sans avoir les composants, nous avions commencé à réfléchir à la construction du cube et à la programmation. À l'origine, nous avions prévu d'utiliser des contrôleurs de LED TLC5940. Malheureusement, les composants n'étant plus en stock, nous avons dû nous rabattre sur une autre solution qui a été d'utiliser des registres à décalage. Les recherches initiales se sont donc révélées inutiles et il a fallu recommencer.
D'autre part la construction du cube et la soudure des composants a pris plus de temps que prévu (5H à deux). Nous avions donc pris un retard conséquent pour la programmation. Nous n'avons donc pas pu finaliser correctement la partie logicielle, mais cela reste fonctionnel dans l'ensemble.

L'accéléromètre n'est malheureusement pas utilisable pour deux raisons : il est attaché à la plaque de prototypage et non pas au cube (ce qui perd de son intérêt) et à cause d'un problème d'alimentation, les valeurs lues sont incorrectes.
En effet, les LEDs consommant un courant important, lorsqu'elle sont toutes allumées la tension de l'alimentation chute à 4,5V, ce qui fausse les valeurs lues sur les entrées analogiques. Il en résulte une variation de quasiment 90° sur les valeurs de pitch et de roll entre LEDs éteintes et LEDs allumées.

6 Améliorations possibles

- Cube 8x8x8 LEDs
- Capteur de contact
- Apparence : boîte, masquer les câbles, etc…
- Plus d'animations

7 Ressources

- `https ://www.arduino.cc/en/Tutorial/ShiftOut`
- ShiftRegister74HC595 : `https ://github.com/Simsso/Shift-Register-74HC595-Arduino-Library`
- Code source : `https ://github.com/piernov/LED_Cube_Arduino`