oxbern

API pour cube à LEDs

Sujet de projet de spécialité 2A Ensimag

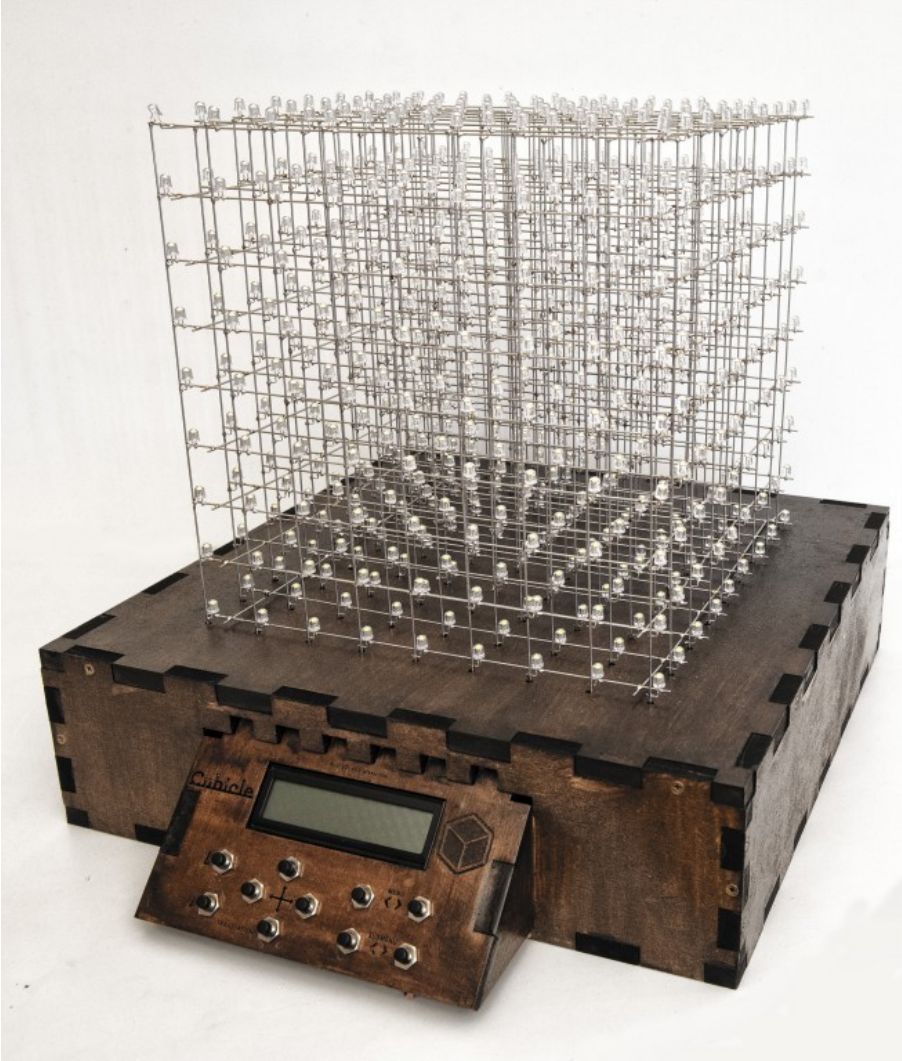
2015-2016

Encadrant : Charles Eynard

Contact : pierre.schefler@oxbern.com

Préambule

Le projet Oxbern vise à concevoir un afficheur 3D pour représenter des motifs cristallographiques (voir ci-dessous).



Il s'agit d'un réseau cubique tridimensionnel de 9 * 9 * 9 = 729 LEDs allumables individuellement. Un écran tactile 7 pouces (qui remplace le panneau avant visible sur la photo) permet de manipuler l'afficheur. A l'arrière, un slot SD permet d'insérer une carte SD qui contient les motifs à afficher. On trouve aussi un port USB afin de connecter l'afficheur à un ordinateur.

L'ensemble est contrôlé par un STM32 qui embarque FreeRTOS et les divers modules pour accéder à l'USB, à la carte SD, à l'écran, au LEDs, etc.

Objectif

Votre rôle est d'ajouter la possibilité au cube d'être contrôlé depuis un ordinateur. Il faut pour cela implémenter une API, à la fois du côté du PC via une librairie partagée, mais aussi du côté du firmware via une machine à états pour interpréter les paquets reçus par USB (le travail sera cependant largement prémâché sur le firmware).

Fonctionnalités

Communication avec le cube

La librairie doit tout d'abord s'occuper de la liaison série : l'initialiser et la rompre. Elle doit également gérer la transmission des paquets. Tout cela est une couche en dessous de l'API, mais est fondamental. L'utilisateur ne devrait pas avoir à vérifier par lui-même si de nouveaux paquets arrivent. La librairie devrait d'elle-même créer un thread qui s'occuperait de faire ça, et qui appellerait un callback réimplémenté par l'utilisateur (si et seulement si l'utilisateur a effectivement besoin de recevoir des données du cube).

En ce sens, les fonctions suivantes sont suggérées :

Fonction	**Description**
available() ;	Retourne true si un cube est disponible
connect();	Établir la liaison série
disconnect() ;	Rompre la liaison série

Contrôle des LEDs

Le plus important est bien entendu de pouvoir contrôler les LEDs. Au niveau du cube, des fonctions internes vous permettront de facilement implémenter les diverses commandes. Au niveau de l'API, voici les fonctions suggérées :

Fonction	**Description**
on(x, y, z) ;	Allumer la LED (x,y,z)
off() ;	Tout éteindre
off(x, y, z) ;	Éteindre la LED (x,y,z)
toggle(x, y, z) ;	Changer d'état la LED (x,y,z)

Pour éviter de surcharger l'USB (par exemple si l'utilisateur allume les LEDs une à une), ces fonctions modifieront un buffer interne (à vous de mettre en place une structure de données correspondante) qui sera alors transmis au cube via la

fonction **display()**. Cette fonction transmettra vraisemblablement une série de 0 et de 1 correspondant aux 729 LEDs du cube, informant sur l'état de chaque LED. La taille de données devrait être de 729 bits, soit 92 octets. Il vous est conseillé de réduire encore cela en compressant les données avant envoi. Le problème est plutôt au niveau de la décompression du côté du firmware, mais voici une page pour vous aider : http://demin.ws/blog/english/2012/09/18/miniz-minilzo/.

Mise à jour du firmware

La mémoire flash du STM32 est séparée en trois parties : le bootloader, le firmware 1 et le firmware 2. Après le démarrage, le micro-contrôleur passe soit sur le firmware 1, soit sur le 2. La mise à jour se fait donc en écrasant le firmware qui est inutilisé, puis en indiquant au STM32 qu'il faut utiliser le nouveau firmware.

L'API contiendra une fonction capable de passer en bloc tout un nouveau firmware. Bien entendu, il faudra s'assurer que l'intégralité du binaire est transférée, et que l'image n'a pas été réordonnée. Vous aurez donc à charge de trouver une méthode pour contrôler le bon déroulement du transfert.

Contraintes

Langage

Vous avez le choix entre du C et du C++(03 ou 11). A vous de voir ce qui semble le plus pratique.

(Notez qu'un format classique des librairies qui communiquent avec des objets par USB est de proposer une classe « Listener » dont on hérite et qui contient le callback de lecture et toutes les fonctions utiles (voir l'API Leap Motion par exemple) ; le langage serait donc plutôt du C++).

Compatibilité

La librairie doit pouvoir être utilisée sur Windows, Mac OS et Linux. Il vous est suggéré d'utiliser cmake, mais vous pouvez aussi créer des projets spécifiques pour chaque OS avec Visual Studio sur Windows par exemple, etc. Vous pourriez même créer une libraire avec Qt (d'autant plus que qextserialport permet de facilement mettre en place une liaison série).

Ressources

Le projet doit bien évidemment être accompagné d'une documentation (suggestion : Doxygen) et de nombreux exemples.

Le suivi du projet se fera via un dépôt github auquel vous serez invité au lancement du projet.

Bonus

Bindings

Si le temps le permet, il peut être intéressant de créer des wrappers ou des bindings de l'API vers d'autres langages (notamment vers du Python).

Outil de mise à jour du firmware

En marge des exemples de l'API, vous pourriez mettre au point un petit logiciel en ligne de commande permettant de mettre à jour le firmware du cube. Ce serait une bonne démonstration de la fonctionnalité, en plus d'être pratique pour la suite du projet.

Contrôle des périphériques

Il pourrait être intéressant de pouvoir contrôler l'écran tactile et la carte SD depuis l'ordinateur. Il faut en revanche réfléchir à l'utilisation de telles fonctionnalités à leur intégration dans la spécification de l'API.

Éléments fournis

Spécification de l'API

Une première spécification de l'API vous sera fournie au démarrage du projet. Elle présentera la structure des paquets (header, données, CRC) et les diverses fonctions suggérées. Vous serez libres de la modifier.

Firmware

Le firmware vous sera fourni avec une première implémentation de la machine à états correspondant aux spécifications de l'API susmentionnée. Vous aurez à charge de le compléter et/ou de le modifier.

Prototype

Un prototype du cube vous sera fourni, avec l'intégralité des cartes électroniques ainsi que le programmeur pour le STM32.

C’est parti

Gardez en tête également que ce sujet a été écrit dans l’optique de vous laisser un maximum de liberté. Un certain nombre d’éléments sont fournis pour vous aider à démarrer plus vite mais toute initiative sera la bienvenue. Vous êtes avant tout des collaborateurs.

Soyez créatifs et rigoureux.


GOOD LUCK WITH THAT
memecrunch.com