h e p i a

Haute école du paysage, d'ingénierie
et d'architecture de Genève

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences
Western Switzerland

CoRES - Communicating, Reconfigurable, Embedded Systems
P. Buffo
G. Da Silva Marques

Dynamic matrix display ou Écran matriciel dynamique

Atelier Systèmes Embarqués
2024 - 2025

Situation

Un écran est un appareil permettant à un utilisateur de visualiser de manière graphique des informations provenant d'un ordinateur. Un écran possède :

- **Une dimension physique** : souvent exprimée dans le sens de sa diagonale et en pouces (par exemple, 24").
- **Une définition** : un nombre de pixels le long de l'axe X, ainsi qu'un nombre de pixels le long de l'axe Y (par exemple 1920x1080). Ce rapport définit également le ratio d'un écran (Par exemple 16 :9).
- **Une résolution** :soit une notion de densité de pixels (exprimée en points par pouce (PPP)).

Le terme **résolution** est souvent employé à tort en français pour parler de **définition**. Cette faute commune nous provient très probablement de l'anglais ou le terme "`resolution`" est correct.

De manière conventionnelle, ces caractéristiques sont figées. En effet, la taille d'un écran ne change pas et de nouveaux pixels ne viennent pas se rajouter ou s'enlever en cours de route. Exception faite d'un pixel mort... certes.

Plus récemment des écrans pliables sont apparus, on se retrouve parfois alors à n'employer qu'un nombre réduit des pixels à disposition. Mais qu'en est-il de ces écrans géants qui recouvrent des immeubles ? Ils sont composés d'un nombre variable de plus petites matrices de LED, qui, une fois assemblées, forment un écran pouvant atteindre des tailles démesurés. Une source s'interface avec une première matrice de LED, aussi appelée dalle, qui elle-même transférera les informations aux autres dalles au travers d'un protocole de communication.

Objectif

HEPIA développe depuis plusieurs années déjà une carte nommée HepiaLigth. Nous en sommes à la version 3. Cette carte dispose d'un microcontrôleur **RP2040**, d'une matrice de LED de 16x16, et de 4 connecteurs d'interconnexion le long de ses 4 points cardinaux. Il est donc possible de créer une matrice d'une plus grande taille en interconnectant plusieurs d'entre elles.

Votre objectif est de créer un système d'affichage de taille dynamique à l'aide d'HepiaLight 3 et qui s'interface avec Linux sous la forme d'un **`framebuffer`**. Ainsi, il pourra être utilisable par

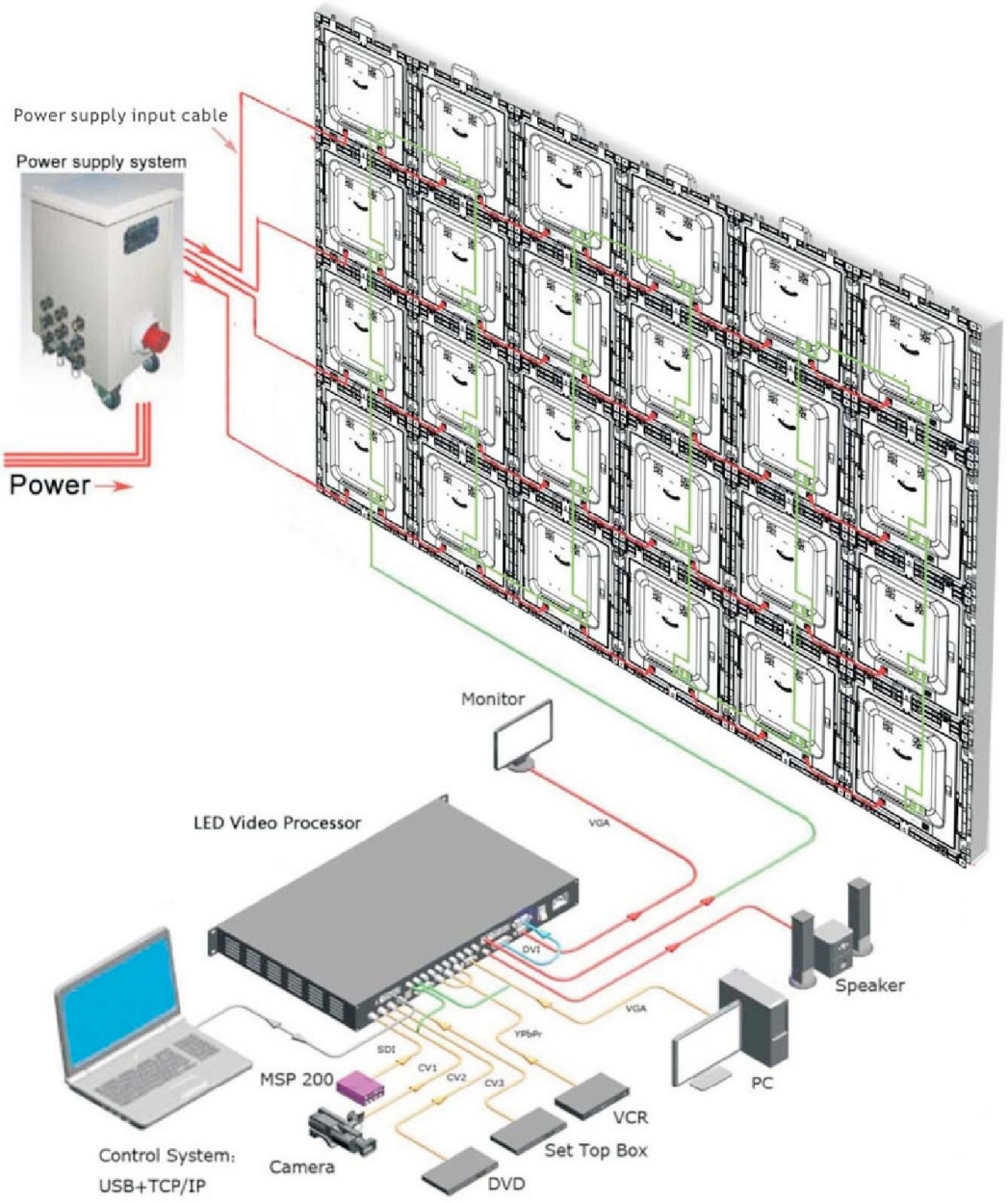

Figure 1. *Schéma type d'un écran constitué d'un ensemble de matrices*

n'importe quelle application fonctionnant sur Linux. Pour démontrer l'intérêt de l'aspect dynamique de votre écran, vous réaliserez un jeu de la vie qui s'adaptera à votre matrice lorsque vous rajouterez des HepiaLight 3 en cours de partie.

Un framebuffer est une couche d'abstraction graphique indépendante du matériel pour afficher des éléments graphiques dans une console sans avoir besoin de bibliothèques spécifiques.

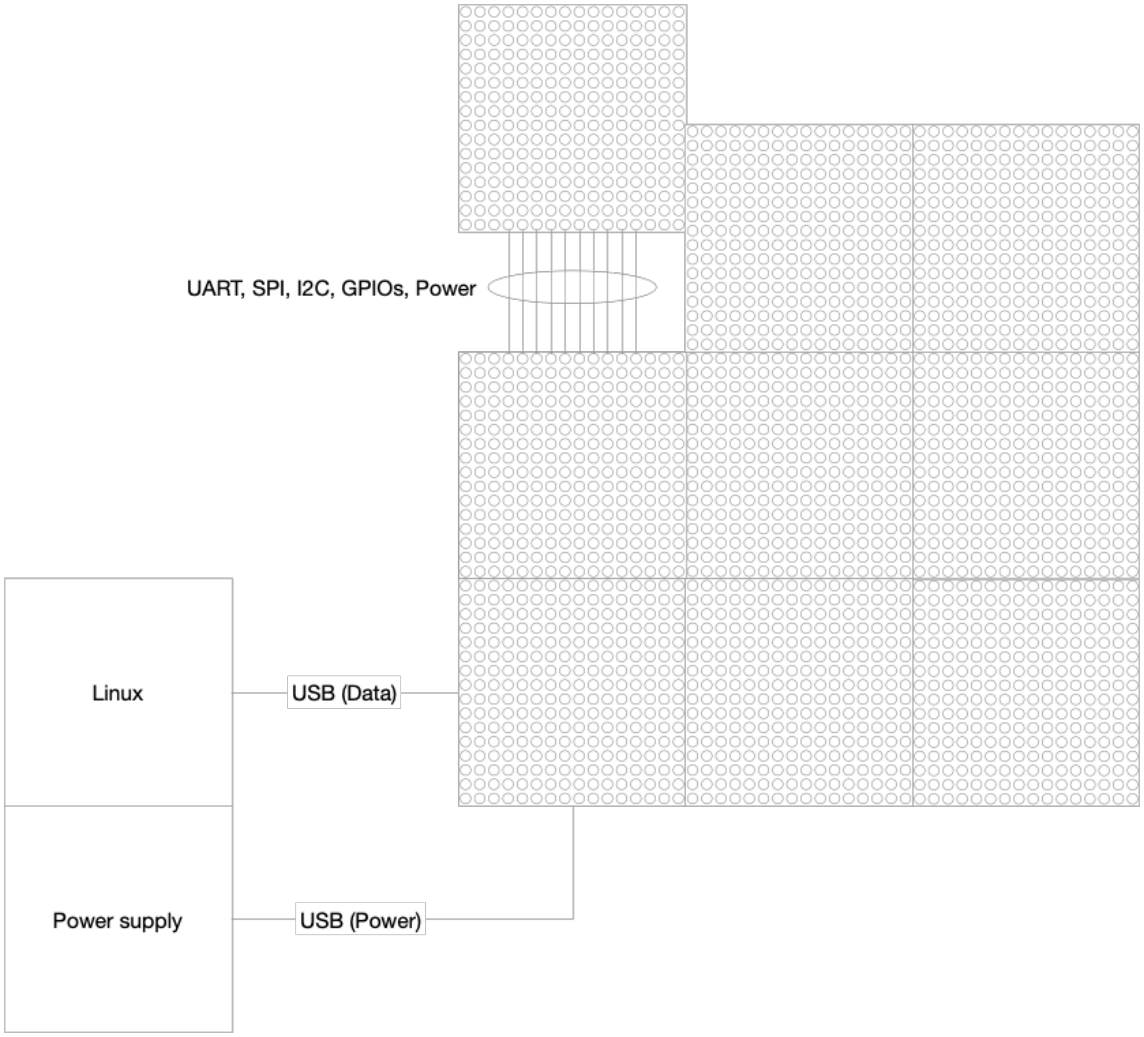

Figure 2. *Schéma du projet*

Vous allez écrire différents composants logiciels pour que votre système fonctionne :

- Un firmware fonctionnant sur les HepiaLight 3, ce firmware doit être commun entre toutes les cartes. Une carte, dite **master**, sera interfacée avec votre PC à travers le connecteur USB. La méthode d'élection d'une carte **master** est libre. Le bouton au dos de la carte peut par exemple être utilisée pour cela. La détection de présence d'un port USB connecté peut aussi être une solution.
- Un driver Linux qui s'interface avec le **RP2040 master** et qui expose un **`framebuffer`**. A chaque fois qu'une carte est rajoutée sur la matrice, votre driver devra également changer les propriétés du **`framebuffer`** en indiquant sa nouvelle taille.
- Un jeu de la vie écrit en C et fonctionnant sur Ubuntu, utilisant votre **`framebuffer`** et qui suive activement les changements de définition de votre matrice, de sorte que votre jeu se mette à se propager sur de nouvelles matrices connectées.

Indications supplémentaires

- Pour des questions de simplicité on rajoute toujours des nouvelles lignes ou des nouvelles colonnes au système. A vous de choisir si vous considérez qu'une nouvelle ligne/-colonne est présente dès qu'on insère le première, ou la dernière carte la constituant.
- Votre **`framebuffer`** exposera une taille de grille correspondant à l'intégralité des pixels des matrices. A vous de choisir si vous souhaitez déterminer à quelle matrice appartient un pixel donné dans votre driver sur Linux ou dans votre firmware sur le **RP2040**.
- Il vous faudra trouver une solution permettant de détecter et d'adresser de manière dynamique les nouvelles matrices connectées. L'utilisation du port UART est conseillée pour la communication, même si vous pouvez utiliser l'intégralité du connecteur à disposition.
- Vous allez utiliser le port USB de votre **RP2040**. La création d'une classe USB personnalisée est conseillée, vous pouvez utiliser le projet **tinyusb** comme point de référence.

Idées Bonus

- Vous pouvez créer aussi des nouvelles propriétés dans votre **`framebuffer`** vous permettant d'exposer la topologie de matrices qui ne sont pas des rectangles, mais de forme diverses.
- Votre système pourrait supporter des interconnexions de cartes en 3 dimensions.
- Faire tourner une application graphique classique sur votre écran.

Bon travail.