ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Enseignes et afficheurs à LED

Affichages matriciels

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Pierre-Yves Rochat

Affichages matriciels

Pierre-Yves Rochat

Afficheurs

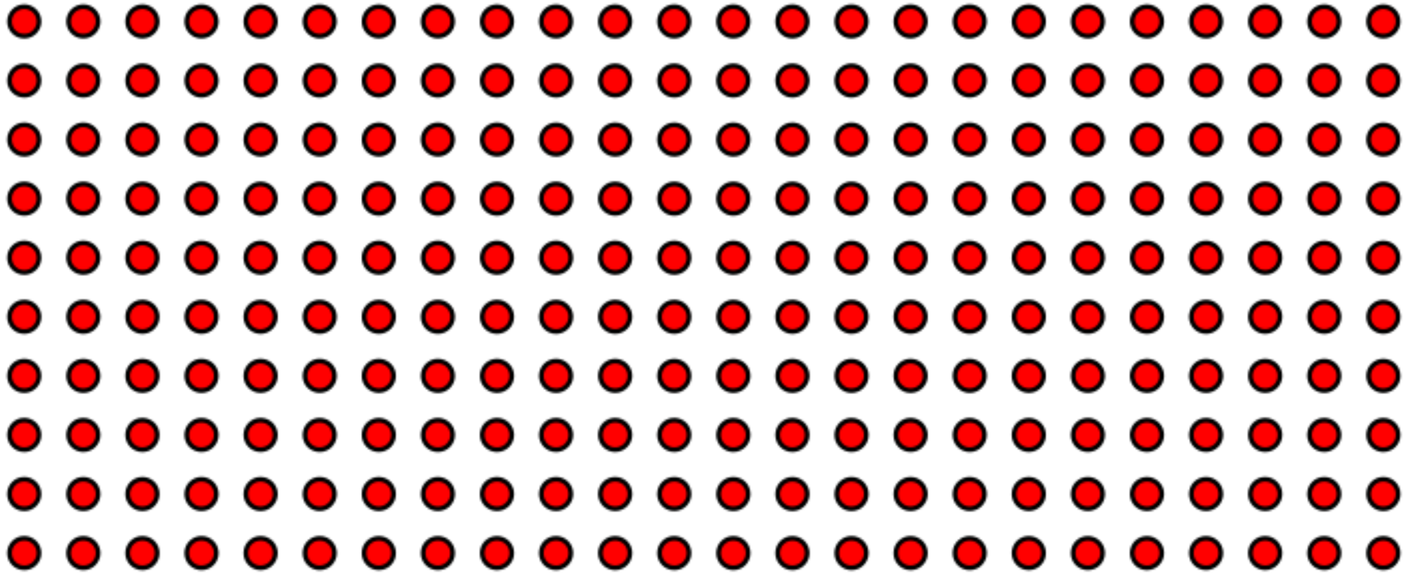
- Notion de pixel
- Caractéristique des afficheurs
- Matrices de LED
- Commandes par registres
- Programmation
- Génération et rafraîchissement

Notion de pixel

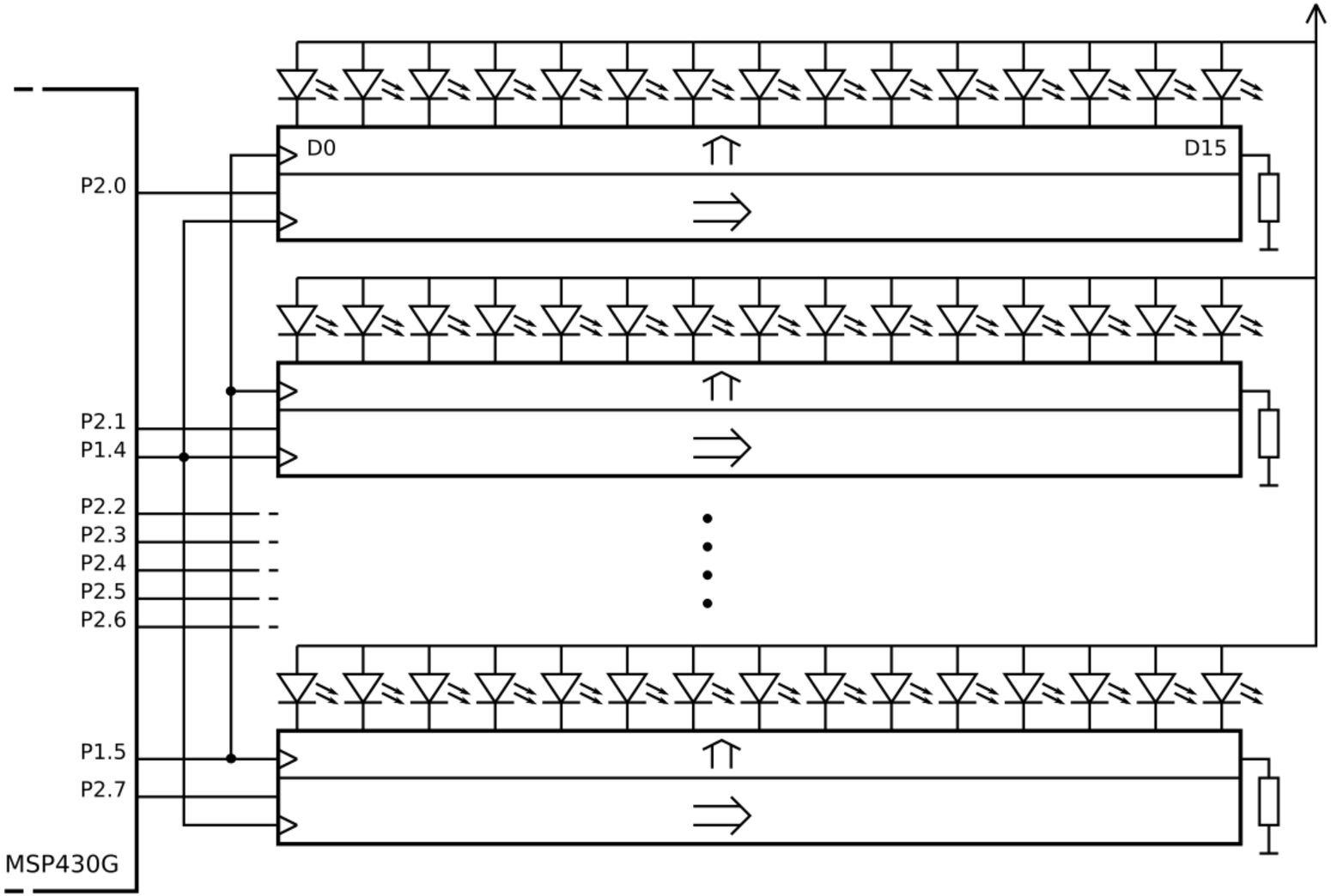
- **afficheur** : dispositif électronique permettant de présenter visuellement des données

- ensemble de **pixels**
- **résolution** : distance entre un pixel et son plus proche voisin
- **densité** : nombre de pixels par unité de surface

Afficheurs à LED



Commande des LED par des registres



Programme de commande

```
int main() {
  init(); // initialisations...
  uint8_t i;
  while (1) {
    for (i=0; i<16; i++) {
// envoie une colonne avec un seul pixel allumé
      P1OUT = (1<<(i&7)); // 1 col de 8 px, 1 seul allumé -
> dents de scie
      SerClockOn; SerClockClear; // envoie un coup d'horloge série
      ParCloclOn; ParClockClear; // envoie un coup d'horloge
    }
  }
}
```

Générateurs de caractères

```
const uint8_t GenCar [] { // tableau des pixels des caractères
  0b01111110, // caractère 'A'
  0b00001001, // Il faut pencher la tête à droite
  0b00001001, // pour voir sa forme !
  0b00001001,
  0b01111110,

  0b01111111, // caractère 'B'
  0b01001001, // Les caractères forment
  0b01001001, // une matrice de 5x7
  0b01001001,
  0b00110110,

  0b00111110, // caractère 'C'
  0b01000001, // Les caractères ont ici
  0b01000001, // une chasse fixe, c'est-à-dire
  0b01000001, // que tous les caractères ont
  0b01000001  // la même largeur en pixels
};
```

Affichage d'un texte

```
1 char *Texte = "ABC\0"; // texte, terminé par le caractère nul
2 const char *ptTexte; // pointeur vers le texte à afficher
```

Affichage d'un texte

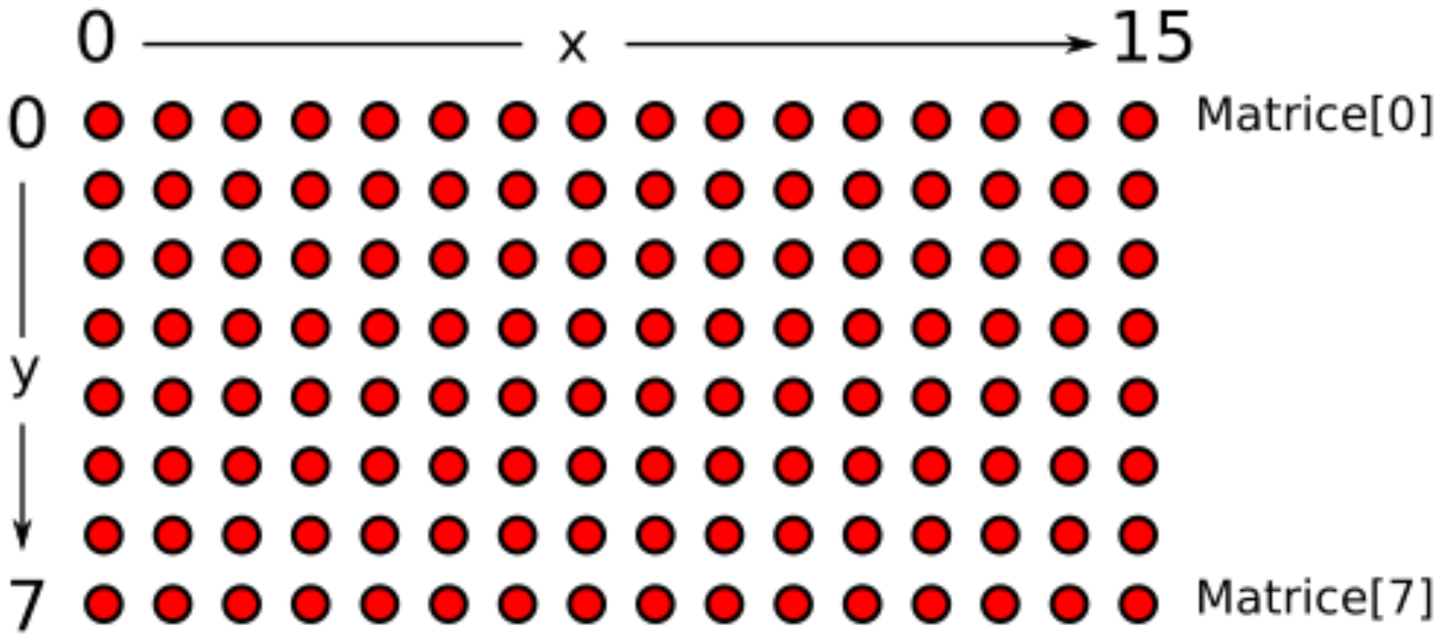
```
3  int main(void) {
4    init(); // initialisations...
5    while(1) { // le texte défile sans fin
6      ptTexte = Texte;
7      while (*ptTexte!='\0') { // boucle des caractères du texte
8        caractere = *ptTexte; // le caractère à afficher
9        idxGenCar = (caractere-'A') * 5; // conversion ASCII à index GenCar[]
10       for (i=0; i<5; i++) { // envoie les 5 colonnes du caractère
11         P2OUT = ~GenCar[idxGenCar++]; // 1 colonne du caractère (actif à 0)
12         SerClockSet; SerClockClear; // coup d'horloge série
13         ParClockSet; ParClockClear; // coup d'horloge parallèle
14         AttenteMs (delai);
15       }
16       ptTexte++; // passe au caractère suivant
17       P2OUT = ~0; // colonne vide, séparant les caractères
18       SerClockSet; SerClockClear; // coup d'horloge série
19       ParClockSet; ParClockClear; // coup d'horloge parallèle
20       AttenteMs (delai);
21     }
22   }
23 }
```

Séparer génération et rafraîchissement

- Géométrie pas toujours idéales
- Afficheurs multiplexés
- Génération et rafraîchissement séparés

Mémoire des pixels

```
1 #define NbLignes 8
2 uint16_t Matrice[NbLignes]; // mots de 16 bits, correspondant à une ligne
```



Dessin des points

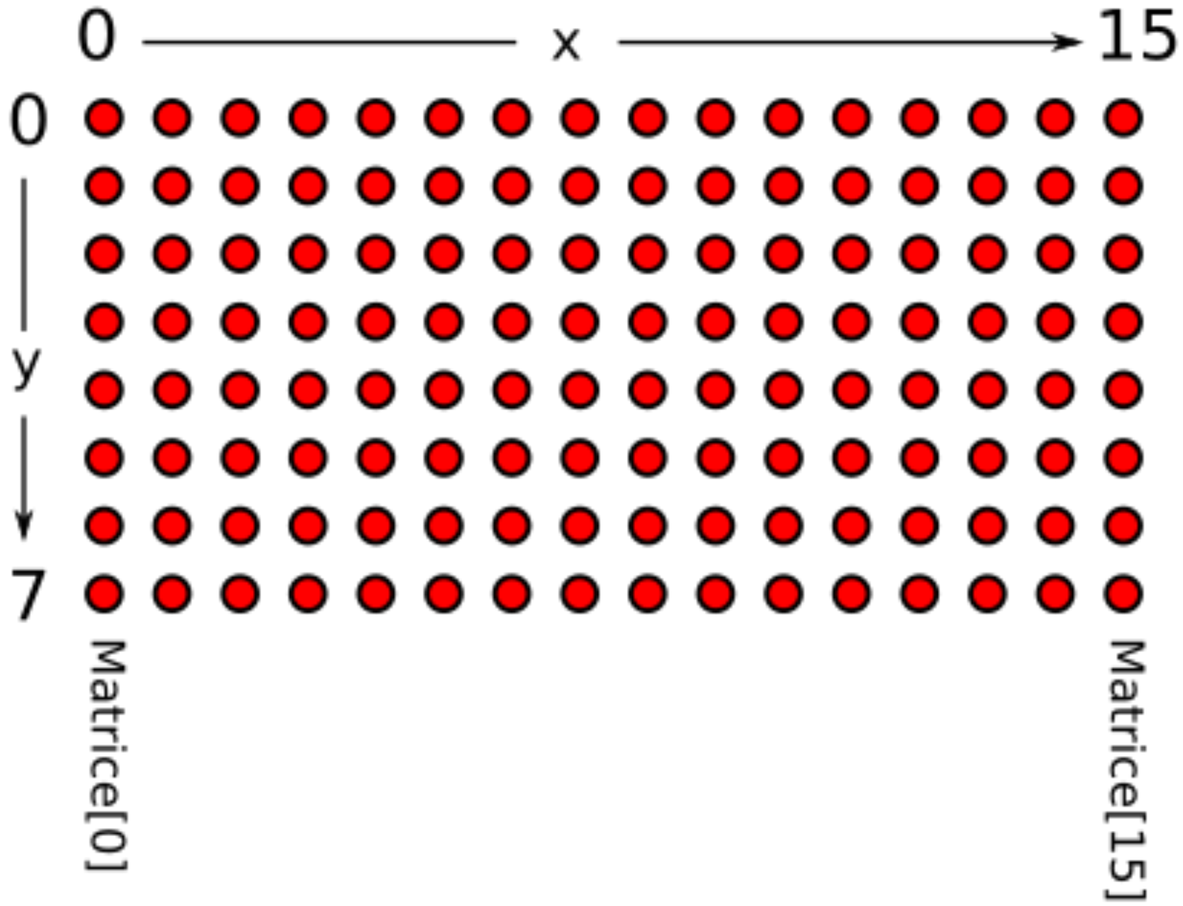
```
1  void AllumePoint(int16_t x, int16_t y) {
2    Matrice[y] |= (1<<x); // set bit
3  }
4
5  void EteintPoint(int16_t x, int16_t y) {
6    Matrice[y] &=~(1<<x); // clear bit
7  }
8
9  #define MaxX 16
10 #define MaxY NbLignes
11
12 void Diagonale() {
13   int16_t i;
14   for (i=0; i<MaxY; i++) {
15     AllumePoint(i*MaxX/MaxY, i);
16   }
17 }
```

Affichage de la matrice

```
1  void AfficheMatrice() {
2    for (uint16_t x=0; x<MaxX; x++) {
3      // Préparation des valeurs qui doivent être envoyées aux 8 registres:
4      for (uint16_t y=0; y<MaxY; y++)  {
5        if (Matrice[y]&(1<<x)) P2OUT &=~(1<<y); else P2OUT |= (1<<y);
6      }
7      SerClockSet; SerClockClear; // envoie un coup d'horloge série
8    }
9    ParClockSet; ParClockClear; // envoie les valeur sur les LED
10 }
```

Mémoire des pixels

```
1 #define NbColonnes 16
2 uint8_t Matrice[NbColonnes]; // mots de 8 bits, correspondant à une colonne
```



Mémoire des pixels

```
#define NbColonnes 16
uint8_t Matrice[NbColonnes]; // mots de 8 bits, correspondant à une colonne

void AfficheMatrice() {
  for (uint16_t x=0; x<MaxX; x++) {
    P2OUT = ~Matrice[x];
    SerClockSet; SerClockClear; // envoie un coup d'horloge série
  }
  ParClockSet; ParClockClear; // envoie les valeur sur les LED
}
```

Animations

- préparer une image en mémoire,
- envoyer son contenu sur l'afficheur,
- attendre le temps nécessaire,
- préparer une autre image
- ...

Ping !

```
1  void Ping() {
2    int16_t x=0;
3    int16_t y=0;
4    int8_t sensX=1;
5    int8_t sensY=1;
6    do {
7      AllumePoint(x,y);
8      AfficheMatrice();
9      AttenteMs(DELAI);
10     EteintPoint(x,y);
11     x+=sensX;
12     if(x==(MaxX-1)) sensX=(-1);
13     if(x==0) sensX=1;
14     y+=sensY;
15     if(y==(MaxY-1)) sensY=(-1);
16     if(y==0) sensY=1;
17   } while (!((x==0)&&(y==0)));
18 }
```

Affichages matriciels

- Notion de pixel
- Caractéristique des afficheurs
- Matrices de LED
- Programmation
- Génération et rafraîchissement