Projets bargraphe

1 Thermomètre interactif

Le capteur LM75 mesure la température ambiante, affichée sur l'écran OLED en °C. Le bargraphe LED s'illumine proportionnellement à la température, et la LED RGB change de couleur (bleu pour froid, vert pour modéré, rouge pour chaud).

Utilisation série : Transmettre en temps réel les données de température au PC pour surveiller les variations. (affichage dans le traceur série de l'IDE Arduino)

```
float lireTemperature();
void afficherBargraphe(float temperature);
void changerCouleurRGB(float temperature);
```

2 Jeu de réflexe avancé

Les LEDs du bargraphe s'allument dans une séquence aléatoire qui s'accélère à chaque niveau. L'utilisateur doit appuyer sur le bouton poussoir quand la Led D5 est allumée. À chaque succès, l'écran OLED affiche le score et passe au niveau suivant. Une erreur remet le score à zéro.

```
void allumerSequenceAleatoire();
bool verifierReponse();
void afficherScore(int score);
```

https://arduinofactory.fr/langage-arduino-nombre-aleatoire/

3 Système d'alarme de sécurité

Description : Transformez votre Arduino en un système d'alarme simple. Utilisez le capteur à bille pour détecter un mouvement ou une intrusion (comme l'ouverture d'une porte ou un changement d'inclinaison). Une fois activé, déclenchez une alerte visuelle : le bargraphe clignote en rouge, la LED RGB change de couleur rapidement, et l'écran OLED affiche un message d'avertissement comme "Intrusion détectée !". Un appui sur le bouton poussoir désactive l'alarme.

Étapes principales :

En mode veille, l'Arduino surveille en permanence le capteur à bille.
Lorsqu'un mouvement est détecté, l'alarme s'active.
Un appui long sur le bouton réinitialise et désactive l'alarme.
Les données peuvent être envoyées via la liaison série pour suivi.

```
bool alarmeActive = false;

void surveillerCapteur();
void declencherAlarme();
void afficherMessageAlarme(bool alarmeActive);
void desactiverAlarme();
```

4 Station météo minimaliste

Mesurez la température avec le LM75 et le niveau de lumière ambiante avec la LDR. Les données sont affichées en temps réel sur l'OLED (par exemple, température et luminosité en graphique ou histogramme). Ajoutez la connexion série pour transmettre ces informations à un PC. (Affichage dans le plotter de l'IDE Arduino)

```
typedef struct {
float temperature;
int luminosite;
}Meteo;
Meteo lireDonneesMeteo();
void afficherDonneesOLED(Meteo donnees);
```

5. Contrôleur de luminosité avec la LDR

Le niveau de lumière ambiante est mesuré par la LDR. Les LEDs du bargraphe ajustent automatiquement leur intensité pour s'adapter. Par exemple, plus il fait sombre, plus les LEDs deviennent lumineuses. L'OLED affiche le pourcentage d'intensité.

```
int lireLDR();
void ajusterLuminositeBargraphe(int luminosite);
void afficherPourcentageOLED(int pourcentage);
```

6. Animation lumineuse réactive

En cas de détection de mouvement par le capteur à bille, lancez des animations lumineuses sur le bargraphe LED et faites varier la couleur de la LED RGB. Affichez un message correspondant sur l'OLED (par exemple : "Mouvement détecté").

```
bool detecterMouvement();
void lancerAnimation();
void afficherMessageMouvement(bool mouvement);
```

7. Chronomètre de précision (2 étudiants)

Description : Ce projet consiste à créer un chronomètre interactif où l'utilisateur doit arrêter le temps précisément à un certain seuil (par exemple, 5 secondes). Le bouton poussoir sert à démarrer et arrêter le chronomètre. Le temps écoulé est affiché sur l'écran OLED, et le bargraphe LED indique visuellement l'écoulement du temps en se remplissant progressivement. La LED RGB clignote en vert si l'utilisateur s'arrête près du seuil défini, ou en rouge en cas d'écart important.

Lorsque le bouton est pressé pour démarrer, le chronomètre se lance et les LEDs du bargraphe s'allument progressivement.
Appuyez à nouveau sur le bouton pour arrêter le temps
Affichez le temps mesuré sur l'écran OLED.
Comparez le résultat au seuil défini et donnez un feedback visuel avec la LED RGB :
Vert : proche du seuil (±0,5 seconde).
Rouge : au-delà de cette tolérance.
Utilisation série : Transmettez les résultats au PC pour garder un historique des temps enregistrés.

```
unsigned long tempsDebut = 0;
unsigned long tempsFin = 0;
unsigned long tempsEcoule = 0;

void demarrerChronometre();
void arreterChronometre();
void afficherTempsOLED(unsigned long tempsEcoule);
void donnerFeedback(unsigned long tempsEcoule, unsigned long seuil);
```

8. Réveil lumineux intelligent

Configurez une heure de réveil en série. Les LEDs RGB et le bargraphe s'allument progressivement pour simuler un lever de soleil. L'écran OLED affiche l'heure actuelle et un message doux ("Réveillez-vous !"). Utilisez le LM75 pour ajuster l'intensité en fonction de la température ambiante.

```
void simulerLeverSoleil();
void afficherHeureEtMessage(int heure);
```

9. Jeu d'adresse avec OLED

Le jeu affiche un curseur mobile sur l'écran OLED représentant une cible. L'utilisateur utilise le potentiomètre pour déplacer un autre curseur et aligner son positionnement avec celui de la cible. Le curseur sur l'écran OLED se déplace automatiquement à chaque niveau, augmentant la difficulté (par exemple, vitesse accrue ou mouvements erratiques). En cas de succès, un message de victoire est affiché, et en cas d'échec, un message d'encouragement apparaît.

Étapes principales :

Cible mobile : Le curseur de la cible (par ex., une petite barre ou un point) se déplace horizontalement ou verticalement de manière automatique et contrôlée.
Interaction utilisateur : Le potentiomètre ajuste la position d'un deuxième curseur sur l'écran OLED.
Validation : Lorsque les deux curseurs s'alignent, le score augmente, et le niveau suivant commence avec une cible plus difficile.
Feedback visuel : La LED RGB clignote en vert pour un alignement réussi ou en rouge en cas de tentative ratée.
Utilisation série : Le score et les résultats peuvent être envoyés au PC via liaison série pour suivi ou statistiques.

```
int positionCible = 0;
int positionUtilisateur = 0;

void afficherCibleOLED(int position);
void afficherCurseurUtilisateurOLED(int position);
bool verifierAlignement(int positionCible, int positionUtilisateur);
void miseAJourScoreEtNiveau(bool reussi);
```

https://javl.github.io/image2cpp/
https://www.instructables.com/How-to-Display-Images-on-OLED-Using-Arduino/

https://github.com/PaulStoffregen/Time