Les interruptions avec arduino

Les interruptions

Interruptions

Si vous créez un projet qui repose fortement sur des données de capteurs précises et que vous devez donc vous assurer de lire et d'enregistrer toute variation de valeur, il peut être difficile d'écrire un programme capable de faire autre chose correctement. En effet, le microcontrôleur est constamment occupé à lire les valeurs. Pour contourner ce problème, vous pouvez utiliser des interruptions qui peuvent vous aider à lire les entrées d'un encodeur rotatif ou d'un bouton-poussoir, par exemple, sans avoir à insérer de code dans votre fonction de boucle.

Cette fonctionnalité peut s'avérer particulièrement utile pour les développeurs d'un GIGA R1, dont les circuits deviennent de plus en plus complexes.

Toutes les broches GPIO du GIGA R1 peuvent être utilisées pour les interruptions.

La syntaxe de création d'une fonction d'interruption doit être incluse dans

void setup()

et est comme suit :

attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)

- pin
- représente le numéro de broche de la broche à laquelle votre capteur d'entrée est connecté,
- ISR
- est la fonction qui est appelée chaque fois que l'interruption est déclenchée, et doit être définie par vous quelque part dans votre croquis.
- mode
- définit quand l'interruption doit être déclenchée et peut être l'un des quatre modes prédéfinis.

Les différents modes utilisables sont :

- LOW
- déclenche l'interruption lorsque la broche est basse.
- CHANGE
- se déclenche chaque fois que la broche change de valeur.
- RISING

- se déclenche lorsque la broche passe de bas à haut.
- FALLING
- se déclenche lorsque la broche passe de haut en bas.

Cet exemple de croquis allumera ou éteindra une LED connectée à la broche 13 chaque fois qu'un bouton-poussoir connecté à la broche 2 est enfoncé ou relâché :

interuption.ino

```
const byte ledPin = 13;

const byte interruptPin = 2;

volatile byte state = LOW;


void setup() {

  pinMode(ledPin, OUTPUT);

  pinMode(interruptPin, INPUT_PULLUP);

  attachInterrupt(digitalPinToInterrupt(interruptPin), blink, CHANGE);

}


void loop() {

  digitalWrite(ledPin, state);

}


void blink() {

  state = !state;

}
```