Arduino cours 1

Utilisation des mots #define , void setup (),pinMode,), void loop(),delay, OUTPUT, INPUT_PULLUP,DigitalWrite { } ;

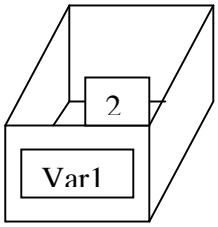
Explication du programme qui fait clignoter une LED

On définit les ‘ pattes ‘(ou pin dans le langage électronique) qui seront utilisées .
Chaque patte est numérotée et gravée sur la platine arduino
Si on souhaite utiliser les pattes 13 et 5 on écrira

```
#define   LED   13    // LED est un synonyme de  patte 13
#define   BP4   4     // BP4  est un synonyme de  la patte 4

// une autre solution serait d'écrire  int LED=13 ;
                       // mais ça prend plus de mémoire

//ensuite on écrit un 'setup' qui va préciser ce  que font ces pattes

void setup ()                          //chaque ligne du programme se termine par ; pour indiquer que la
ligne est

                                        // finie
{                                      // l'accolade { signifie début du setup
pinMode (LED , OUTPUT) ;               //la pin 13 est configurée en sortie
pinMode (BP4 ,INPUT_PULLUP) ; // la pin 4 est configurée  en entrée 5 V
                                       //un bouton poussoir sera cablé entre la patte 4 et la masse
                                       // si on appuie sur le poussoir  BP=LOW (0v)sinon BP=HIGH(5v)
}                                      // l'accolade }   signifie fin du setup


void loop()
 {                                      // l'accolade { signifie début du loop (du programme en boucle)
  digitalWrite (LED, HIGH) ;           // cette instruction met 5V sur  la patte 13 (la led s'allume)
                                       //sur laquelle une LED et une résistance  sont connectées vers la masse
delay (1000) ;                         // pendant 1000 ms  (donc 1 seconde) on ne fait rien

digitalWrite (LED, LOW) ;              // cette instruction met 0V  la patte 13  (la led s'eteint)
delay (2000) ;                         // pendant 2000 ms  (donc 2 secondes) on ne fait rien

}                                      //l'accolade }  signifie fin du loop , repartir au debut du loop {
```

Nota : nous avons dû utiliser une dizaine de mots pour écrire ce premier programme comme on doit apprendre un minimum de mot quand on veut parler une langue , cependant il faudra beaucoup moins de mots pour parler ARDUINO que pour parler une langue .
BP4 n'a pas été utilisé (suite au prochain numéro !)

Arduino cours 2 utilisation de variables



Un des éléments les plus utilisés c'est la variable .
Nous allons imager une variable de la façon suivante :

Soit 3 boites qui contiennent des nombres positifs et négatifs.
Une boite ne peut pas contenir de hors e ces **limites** (-2147453648 à +2147453648)
On va nommer ces boites var1 ,var2 …. (on pourrait les nommer compteur ..)
en début de programme on écrira

int var1=2 ; (cette boite ne peut pas contenir qu'unentier compris entre les **limites**)
(**int** cette boite ou cette variable ne peut pas contenir un nombre à virgule)
int var2=10 ; ((cette boite ou cette variable contient 10)
int var3=20 ; ((cette boite ou cette variable contient 20)
int var4=0 ; (cette boite ou cette variable contient 0)

Si la variable ne contient que des nombres positifs ente 0et255 on choisir le type **byte**

byte var5=12 ;(cette boite ne peux contenir que des nombres entre 0 et 255)

Nous venons de définir 5 variables
Dans le programme loop() on pourra écrire
var3= var1+ var2 ; la boite var3 va contenir 2+10=12 (var3 ne contient plus 20) donc var3=12

si on écrit var3=var3+5 cela veut dire qu'on ajoute 5 dans la boite var3 ,donc var3 contient 12+5=17

(si on fait une division on n' aura pas les chiffres après la virgule :exemple si var3=16 var2=5var4= var3/var2=)3

Si nous souhaitons travailler avec des nombre décimaux il vaut remplacer **int** par **float**

float var3=3.14153 ; (cette boite peux aussi contenir des nombres entiers)
(les limites **sont -3.4028235 E+38 à -3.4028235E+38)**
(E+38 signifie que ce nombre peut contenir 38 chiffres) ça dépasse le montant de ma retraite !!)

Il existe des boîtes pour contenir de nombreux type de variable (nous les utiliserons en fonction de ce qu'on veut stocker, on peut stocker des nombres, des lettres,des mots ..)
voici quelques types de boites

int ,float ,long,car,string,byte, unsigned int ……
Nous utiliserons ces types en fonction des besoins .le fait de bien choisir le type permet de consommer moins de mémoire (par exemple si on sait que le nombre à stocker ne dépassera pas 255 on choisira le type **byte**.)
Nota : si on définit une variable en début de programme on pourra la modifier n'importe où même dans les sous programmes .
Si on définit une variable dans un sous programme, elle ne sera pas modifiable c'est **une variable locale.**
Si elle est définie au début du programme, c'est une **variable globale** la variable sera modifiable partout dans le programme et dans les sous programmes.

Arduino cours 3 : sous programme

Quand un morceau de programme doit être utilisé plusieurs fois ,on peut utiliser un sous programme
Exemple :
Dans le cours 1 nous avions programmé un clignotant . Si cette partie de programme est utilisée plusieurs fois ,au lieu de recopier ces lignes ,on peux écrire un sous programme .
Exemple
Le programme clignotant était le suivant (voir cours 1)

```
#define   LED   13    // initialisation quelles sont les pattes utilisées
 #define   BP  4
void setup ()            //set up      //les pattes sont des entrées ou des sorties ?

{ pinMode (LED , OUTPUT) ;
   pinMode (BP ,INPUT_PULLUP) ;
}
void loop()
 {
  digitalWrite (LED, HIGH) ;
 delay (1000) ;
digitalWrite (LED, LOW);
delay (2000) ;
}
```

Il suffit de remplacer loop par clignotant
et de placer ce programme avant ou après le programme loop

```
 // sous programme
void clignotant ()
 {
  digitalWrite (LED, HIGH) ;
 delay (1000) ;
digitalWrite (LED, LOW) );
delay (2000) ;
}
//le programme loop devient
 void loop(){
clignotant() ; //exécution du sous programme
}
```

Ce programme fera clignoter en permanence la LED.
On pourrai ainsi exécuter le « clignotant » en fonction d'une condition.
Cela fera l'objet du cours 4 (réaliser une action sous condition (boucle if then else))
Passage de paramètre : si l'on souhait modifier les paramètre on peut réécrire le sous programme en intégrant des paramètres . Le sous programme s'écrira de la façon suivante :

```
void clignotant (int t1,int t2)
 {
  digitalWrite (LED, HIGH) ;
 delay (t1) ;
digitalWrite (LED, LOW) );
delay t2) ;
}
```

il sera intégré dans le programme de la façon suivante :

```
void loop(){
  clignotant (1000,2000 ) // on pourra ainsi modifier les temps de temporisation du clignotement .
```

Arduino cours 4 : Exécution d'une tache en fonction d'une condition

Exemple :on souhaite déclencher un clignotement si on appuie sur un bouton poussoir
On utilisera la fonction conditionnelle :**si la condition est vraie exécuter le sous programme**
dans le langage arduino cela s'écrira
if (condition est vrai)
{
exécuter les instructions entre les accolades si la condition est vrai
}
exemple : (allumer le clignotant si le bouton poussoir câblé entre la patte D4 et la masse(GND) ou si le bouton cablé ente la patte D5 et la masse(GND) est appuyé .le clignotement sera différent suivant le BP actionné)

// copier le texte ci dessous dans le logiciel arduino et ça doit marcher !

```
//initialisation
 #define   LED   13  //une led est relié à la patte 13 via une resistance de 1k et porte le nom LED)
 #define   BP4    4    //un bouton poussoir est connecté entre la patte d4 et GND)
 #define   BP5    5    //un bouton poussoir est connecté entre la patte d5 et GND)
// (on pourrait  remplacer bp4 par « cligno_lent »  et bp5 par « cligno_rapide »  pour que le programme soit plus parlant .

void setup ()
  {
    pinMode (LED , OUTPUT) ;  //la Led est une sortie
    pinMode (BP4 ,INPUT_PULLUP) ;//BP4 est une entrée avec une résistance pullup interne
                                  //BP4 sera au niveau bas si il est appuyé
    pinMode (BP5 ,INPUT_PULLUP) ;
  }

void clignotant (int t1,int t2)
 {
   digitalWrite (LED, HIGH) ;
   delay (t1) ;
   digitalWrite (LED, HIGH) ;
   delay (t2) ;
}
void loop()
{

  if (digitalRead (BP4)== LOW )     // attention on utilise deux signes = pour  vérifier une égalité ==
                                    //  et non pas =)
                                   //sinon c'est considéré comme une affectation
      {
       clignotant (2000,2000 );
      }

     if (digitalRead (BP5)==LOW )
       {
        clignotant (500,500 );
       }
 }
```

digitalRead (BP4) permet de lire l'état du BP4 (LOW ou HIGH)

Arduino cours 5 : Répétition d'une action (boucle for)
Si l'on souhaite répéter une action plusieurs fois on utilisera la syntaxe
Par exemple réaliser 5 fois le clignotement

```
For  (int nb=1; nb<=5;nb++)
{
clignotant (1000,1000);
}
```

on souhaite faire clignoter 5 fois la led toutes les 2 secondes 5 fois si on appuie sur BP4 et
faire clignoter 5 fois la led toutes les 1/2 secondes 5 fois si on appuie sur BP5 et
le programme loop () du **cours 4** deviendra
//programme modifié
/ /initialisation (identique au cours 4)
// setup() (identique au cours 4)
// **modification du programme loop()**

```
void loop()
{

    if (digitalRead (BP4)== LOW )     // attention on utilise deux signes = pour  vérifier une égalité (== et non pas =)
                                      //sinon c'est considéré comme une affectation
  {                                   // début du if (si en français )
        For  (int nb=1; nb<=5; nb ++)      //la variable nb prend la valeur 1 puis 2 puis 3 jusque 5
                                           //   (nb++ signifie que nb est incrémenté de 1 en 1
        {                                   // debut de la boucle
           clignotant (2000,2000);          //clignotement  lent toutes les 2 secondes
        }                                   // fin de la boucle

  }                                 // fin  du if (si en français )

 if (digitalRead (BP5)==LOW )
  {

        For  (int nb=1; nb<=5;nb ++)
       {
        clignotant (500,500 );
        }
}

}
```

Nota : pour aller plus loin ! (mais c'est juste pour aller plus loin !!!)

Le pas dans la boucle peut être différent de 1
par exemple :

```
        For  (int nb=0;  nb<=50  ;  nb 5+=)    //pour  nb  variant de 0 à 50 en comptant de 5 en 5
       {
        clignotant (500,500 );
        }

        For  (int nb=50;  nb>=0;  nb 5 -=)   //pour nb variant de 50 à 0  en décomptant de 5 en 5
       {
        clignotant (500,500 );
        }
```

Arduino cours 6 : faire une action **tant que** la condition est vrai.

On souhaite faire clignoter la led tant qu 'on appuie sur le bp4.

Après avoir initialisé le programme et le setup

```
void loop()
{
While (digitalRead (bp4))
    {
      clignotant (500,500 );
     }
}
```

// Faire 5 fois une action
// Toujours après avoir initialisé le programme et le setup

```
void loop()
var2=0
{
   While (var<5)
  {
   clignotant (500,500 );
   var2=var2+1; //il faut que la variable var2 évolue dans le while sinon on ne sort pas de la boucle     while
  }
}
```

Cours 7 : **entrée analogique** (potentiomètre , sortie entre 0 et 5V d'un capteur,tout ce qui sort entre 0 et 5V)

Nous avons traité les entrées digitales ,un bouton poussoir ne permet de choisir que
2 états LOW ou HIGHT.
Un potentiomètre permet de générer une tension variable , mais un ' arduino' ne connaît pas l'analogique .
Cependant il existe des pattes sur lesquelles on peut connecter une tension entre 0 et 5 Volts .

Comment ça marche !
Le potentiomètre comporte 3 broches les 2 extrémités sont reliées au GND (0V) et au 5V .
La patte du milieu est reliée à une entrée analogique par exemple A0.
Suivant la position du bouton du potentiomètre la valeur varie entre 0 et5V.
La procédure consistera à définir une variable (on la nommera par exemple valpot0).
puis à lire la valeur de tension (présente sur point milieu du potentiomètre) .
La valeur analogique (entre 0 et 5volt) est convertie en un nombre compris entre 0 et 1023.

Pour en savoir plus mais pas utile pour la programmation

Nous comptons dans la **base 10** car les nombreS sont composé de 10 symboles 0 1 2 …….9
Le micro processeur est composé d'interrupteurs ouvert ou fermé 0 ou 1 (2 symboles) c'est une base 2

(Le nombre est stocké dans un registre à 10 bits donc si les 10 bits sont à 1 on à la valeur maximale
ça donne ça !
1111111111=1*1+1*2+1*4+1*8+ …… +1*256= **1023**

autre exemple **base 2**	0 ou 1	0	0	0	0	0	1	1	0	1	0
	que multiplie	512	256	128	64	32	16	8	4	2	1
	base10 =	0	+	0 +	0 +	0 +	0	+ 16	+ 8	+ 0	+ 2 + 0 = 26

il existe aussi une base **hexadecimale** (avec 16 symboles 0,1,,,,,8,9,A,B,C,D,E,F ou 26 s'écrit 1A
(bon j'arrête là car tout ça c'est une autre histoire et vous allez me quitter !!!)
//voici le programme :

```
#define  Pot  A0;  (la patte du milieu du potentiomètre est commectée en A0
#define  LED  13  // LED de la platine
int valpot=0;        //variable qui va stocker la tension lue. On l'initialise à 0.
int temps1=0;
void setup() {
                  pinMode (LED,OUTPUT); //Mode OUTPUT pour le pin de LED
                 digitalWrite(LED,HIGH);//On allume la LED
                }

void loop() {
  valpot0=analogRead(A0); //lit la valeur de tension et la stocke dans la variable
                    //si le potentiomètre est au milieu valpot=512

 temps1=valpot0*10
  digitalWrite(LED,HIGH); //on allume la LED

  delay(temps1); //on attend en fonction de la variable temps1  soit=  10*512=5120ms=5.12secondes
  digitalWrite(LED,LOW); //on éteint la LED
  delay(temps1 ); //on attend 1.512 seconde
}
```

Certains capteurs fourniront une tension entre 0 et 5V en fonction d'une température,d'une humidité …
ils ont une sortie OUT qui est l'équivalent de la patte du milieu du potentiomètre .