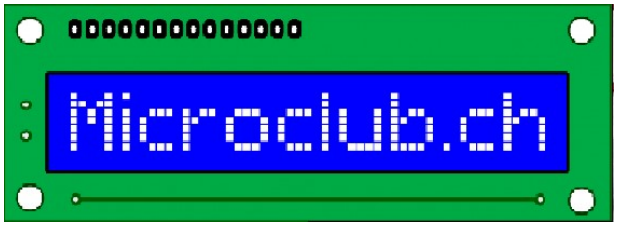
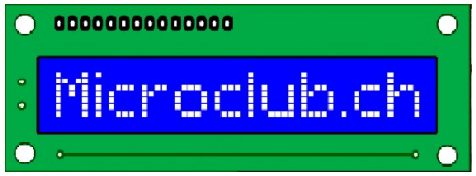


Finesses du C
22.2.2019

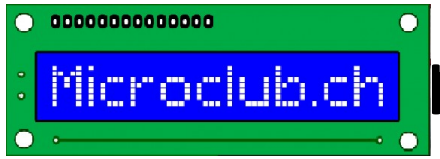
Rolf Ziegler 02/2019

Finesses du C



1. Serial.print de « float » formatés
2. Structures et unions exemple
3. Fonctions bloquantes, solution avec timeout

Rolf Ziegler 02/2019



mprimer des « float »s Sous C/Arduino

- float x=1/333;
- Serial.print(« 1/333=«)
- Serial.println(x);

Résultat = ??

Rolf Ziegler 02/2019

Utilisation de Serial.print

```
#include <Arduino.h>

    float variable=100.0;
    int cnt=0;

    void setup() {
        Serial.begin(115200);
    }

void loop() {
        Serial.print(variable / cnt);
        Serial.println();
        cnt++;
        delay(1000);
    }
```

Affiche 50.00, 33.33, 25.00, pas de possibilité de formatage

Rolf Ziegler 02/2019

Imprimer des « float »
Utilisation de Serial.print

```
#include <Arduino.h>

float testVar = 100.0;
int cnt = 0;
char buffer[20];

void setup() {
Serial.begin(115200);
Serial.println(" ESP32 Ready");
}
void loop() {
sprintf(buffer,"100/cnt=%3.1f",testVar/cnt);
cnt++;
Serial.println(buffer);
delay(1000);
}
```

Affiche 50.0, 33.3, 25.0,…….

Avec %05.1 on aura: 050.0, 033.3, 025.0

Rolf Ziegler 02/2019

Imprimer des « float »
Utilisation de Serial.print

```
#include <Arduino.h>

float variable=1000;
int cnt=0;
void setup() {
Serial.begin(115200);
}

void loop() {

int temp=(int) (variable*10)/cnt;
sprintf(buffer, "%03u.%01u\n",
temp/10,temp%10);

cnt++;
Serial.print(buffer);

delay(1000);
}
```

Affiche 50.0, 33.3, 25.0,…….
Avec %05.1 on aura: 050.0, 033.3, 025.0

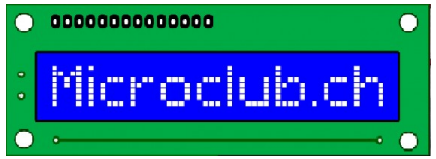
Rolf Ziegler 02/2019

Imprimer des « float »
Utilisation de dtostref

```
void loop()
{
char buffer[10];

cnt++;

dtostrf(testVar/cnt,2,3,buffer);
// sprintf(buffer, "%3.1f\n", testVar / cnt);

Serial.print(buffer);
Serial.println();
delay(1000);
}
```

Imprime: inf 100.000 50.000 33.333 25.000 …..
….20.000 16.667 14.286 12.500 11.111 10.000 9.091

Rolf Ziegler 02/2019



Afficher sur OLED
Utilisation de dtostref

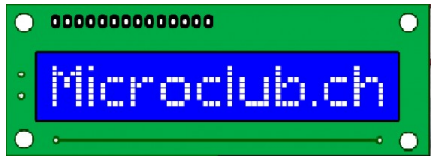
dtostrf(double number, signed char width, unsigned char prec, char *s)

.....

```
dtostrf((counter/i),3,2,buffer);
display.drawString(10, 40, buffer);
display.display();
```

.....

Rolf Ziegler 02/2019

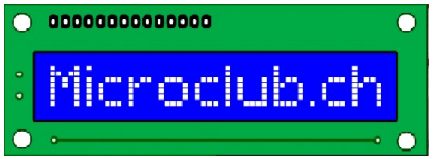


Les finesses du C
Structures et Unions

Je peux définir des structures pour regrouper des valeurs

```
struct gps{
    int lat=0;
    int long=0;
};

gps  avion;
gps  aeroport
```

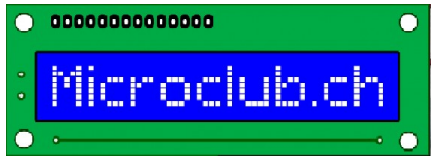
Rolf Ziegler 02/2019



Les finesses du C
Structures et Unions

```
void setup()
{
        Serial.begin(115200);
        Serial.println();
        Serial.println();

        aeroport.lat = 55;
        aeroport.long = 10;

        Serial.print(">lat aero: ");
        Serial.print(avion.lat);
        Serial.print(" long avion: ");
        Serial.println(avion.long);

        avion=aeroport;  // copie lat et long de aeroport vers avion
        ou
        avion.lat= aeroport.lat; // ne copie que lat
```

Rolf Ziegler 02/2019



Les finesses du C
Structures et **Unions**

Problème: mémorisation en EEPROM de variables float, int ou autre types.

Les routines EEPROM mémorisent des bytes ou champs de bytes/char. Il faut donc une astuce !!

Rappel: char, int8_t =➔ 1 byte en mémoire

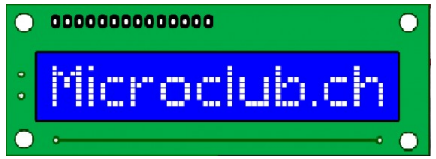
int16_t, unsigned int ➔ 2 bytes

float ➔ 4 bytes en mémoire

double ➔ 8bytes

On peut demander la grandeur en C avec « sizeof(float) » qui retourne « 4 »

Rolf Ziegler 02/2019



Les finesses du C
Structures et **Unions**

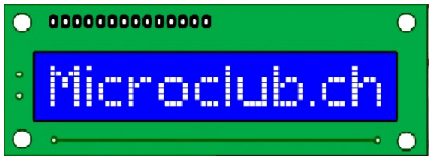
```
typedef union memory{
    unsigned char array[sizeof(float)];
    float gain;
} farr;
```

farr devient le type de données (mieux farr_t)

et on peut donc se faire une variable complex avec

Comme: int variable;

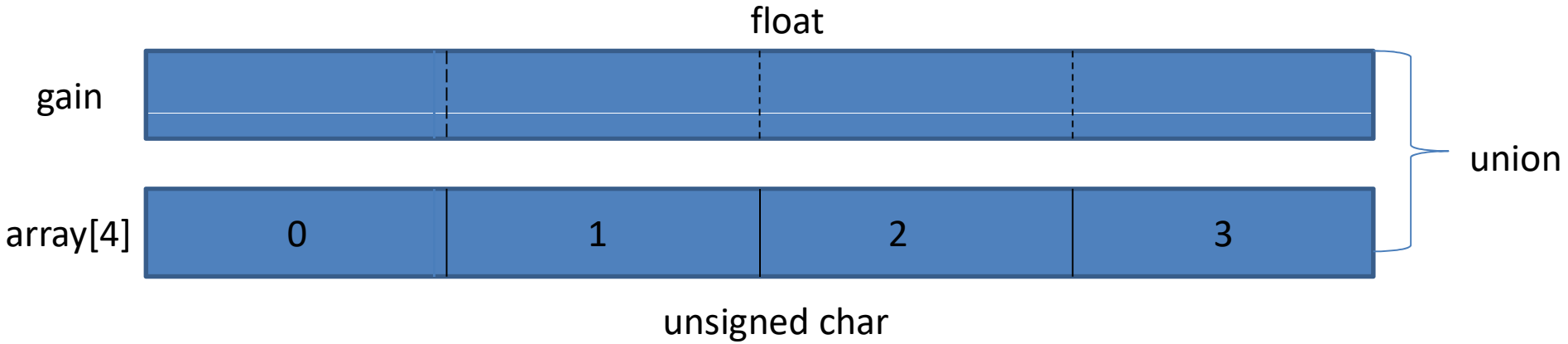
On écrit: **farr mem;** // mem.gain et mem.array
deviennent des variables à la même place en mémoire



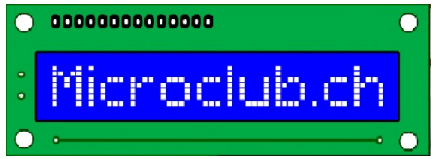
Les finesses du C

Unions résumé

En mémoire la variable float gain est à la même adresse que les bytes de array

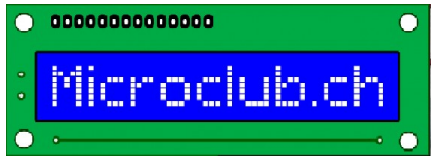


En mémoire(RAM) gain et array[4] occupent les même places.
En EEPROM, nous écrirons les 4 bytes et les resituerons dans le même ordre



Les finesses du C
Structures et **Unions**

```
typedef union memory
{
unsigned char array[sizeof(float)]; // float == 4 bytes
float gain;
} farr_t;

farr_t mmin; // variable pour l'entrée et écriture dans EEPROM
farr_t mmout; // variable pour la lecture en EEPROM

......
mmin.gain = 10.4;
EEPROM.write(0, mmin.array[0]); // il faudra enregister les 4 bytes
et relire avec:
mmout.array[0] = EEPROM.read(0);
```



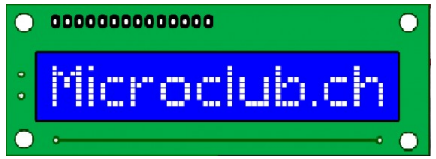
Finesses du C
Code bloquant (Wifi ESP)

Code bloquant:

```
while (WiFi.status() != WL_CONNECTED)
   {
   delay(500);
   Serial.print(".");
   }
```

Tant que notre module ne connecte pas, nous restons dans la boucle !!! Situtation si notre mot de passe ou SSID ne sont pas correcte ou si le roueur WIFI ne fonctionne pas.

Rolf Ziegler 02/2019



Finesses du C
Code bloquant (Wifi ESP)

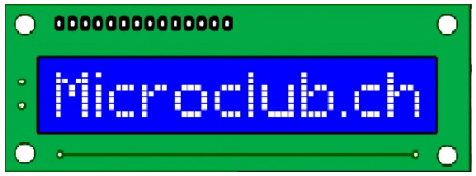
Solution: rentre le code non-bloquant

```
#define TIME_OUT 20
while (WiFi.status() != WL_CONNECTED && timeout < TIME_OUT)
    {
    delay(500); // attente ½ seconde
    Serial.print(".");
    timeout++; // si timeout nous desactivons OSC
    }
```

Après 10 secondes ou moins, nous sortons de la boucle et pouvons tester

```
if (timeout < TIME_OUT){
    // exécuter code avec wifi
    } else {
    // exécuter code sans WIFI ou afficher erreur ou alarm/buzzer !!
}
```

Rolf Ziegler 02/2019



Questions ?