Lab 3

Wemos D1 (ESP8266) capteurs et visualisation des résultats

3.1 Carte Wemos D1 spécifications

Dans ce TP nous allons explorer les fonctionnalités de la carte Wemos D1. Cette carte est très compacte mais elle peut être complétée par de nombreux extensions (capteurs, actuateurs) faciles à mettre en ouvre.
Le processeur de ESP8266 est beaucoup plus rapide que le AVR de la carte Nano/Uno, et sa mémoire RAM et EEPROM sont 10/20 fois plus grandes.
La carte intègre un modem WiFi fonctionnant, selon le choix, en mode **Station** ou point d'accès - **Access Point**.
Elle peut fonctionner en mode faible consommation (**DeepSleep**) permettant son alimentation avec moins de 1mA.

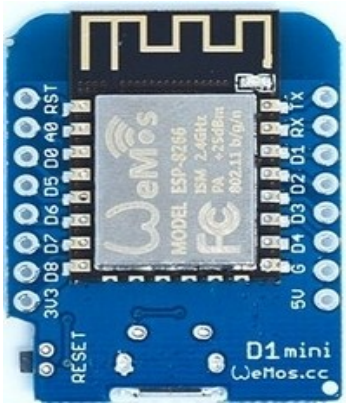

Figure 3.1 Face externe de la carte Wemos D1

La carte Wemos D1 a 11 entrées/sorties numériques , tous les pins supportent **interrupt/pwm/I2C/one-wire** (à l'exception de D0)

- une entrée analogique (3.2V max)
- un connecteur Micro USB
- compatible avec Arduino
- compatible avec NodeMCU

Les capacités de traitement et de la mémoire:

Microcontroller	**ESP-**8266EX
Operating Voltage	3.3V
Digital I/O Pins	11
Analog Input Pins	1(Max input: 3.2V)
Clock Speed	80MHz/160MHz
Flash	4M bytes
Length	34.2mm
Width	25.6mm
Weight	10g

Les entrées/sorties (pins) :

Pin	**Function**	**ESP-8266 Pin**
TX	TXD	TXD
RX	RXD	RXD
A0	Analog input, max 3.3V input	A0
D0	IO	GPIO16
D1	IO, SCL	GPIO5

D2	IO, SDA	GPIO4
D3	IO, 10k Pull-up	GPIO0
D4	IO, 10k Pull-up, BUILTIN_LED	GPIO2
D5	IO, SCK	GPIO14
D6	IO, MISO	GPIO12
D7	IO, MOSI	GPIO13
D8	IO, 10k Pull-down, SS	GPIO15
G	Ground	GND
5V	5V	5V
3V3	3.3V	3.3V
RST	Reset	RST

3.2 Cartes d'extension - capteurs de Temperature-Humidité

Dans ce paragraphe nous présentons deux cartes d'extension pour la Wemos D1 3.1.1 : la carte avec capteur SHT30 et la carte avec capteur DHT21 , les deux fonctionnant sur le bus I2C.

3.2.1 SHT30

;

Figure 3.2 SHT30 ***shield*** pour Wemos D1 Wemos D1

```
#include <WEMOS_SHT3X.h>
SHT3X sht30(0x45);

void setup() {
  Serial.begin(9600);
}

void loop() {
  sht30.get();
  Serial.print("Temperature in Celsius : ");
  Serial.println(sht30.cTemp);
  Serial.print("Temperature in Fahrenheit : ");
  Serial.println(sht30.fTemp);
  Serial.print("Relative Humidity : ");
  Serial.println(sht30.humidity);
  Serial.println();
  delay(1000);
}
```

3.2.2 DHT21

Le deuxième exemple est basé sur DHT21 fonctionnant sur le bus I2C

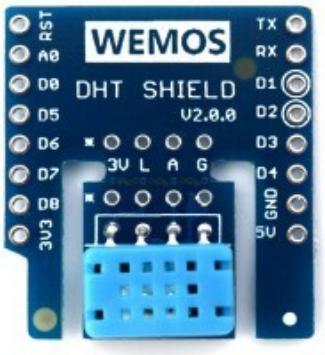

Figure 3.3 DHT21 *shield* pour Wemos D1 Wemos D1

```
#include <WEMOS_DHT12.h>
DHT12 dht12;

void setup() {
  Serial.begin(9600);
}

void loop() {

  if(dht12.get()==0){
    Serial.print("Temperature in Celsius : ");
    Serial.println(dht12.cTemp);
    Serial.print("Temperature in Fahrenheit : ");
    Serial.println(dht12.fTemp);
    Serial.print("Relative Humidity : ");
    Serial.println(dht12.humidity);
    Serial.println();
  }
  delay(1000);
```

3.3 Horloge Temps Réel – RTC (DS3231)

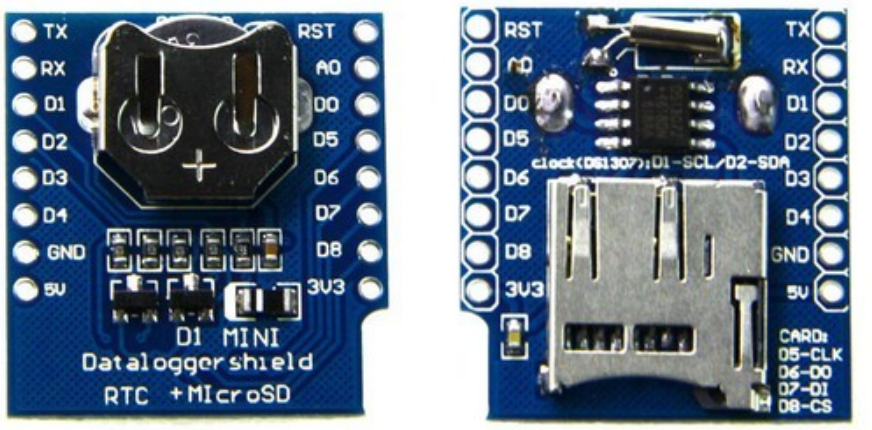

Figure 3.4 RTC (et micro SD card) *shield* pour Wemos D1

Pour connaître et préserver la valeur du temps réel il est nécessaire d'exploiter un circuit RTC (Real Time Clock), par exemple DS3231 ou DS1307. Le circuit RTC est connecté via le bus I2C.
Le cde suivant montre comment initialiser et lire l'horloge RTC:

```
#include <Wire.h>
#include "RTClib.h"
RTC_DS3231 rtc;
char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday",
"Thursday", "Friday", "Saturday"};

void setup () {
 Wire.begin(D2,D1);  // Wire - I2C SDA,SDC pins for Wemos D1
  Serial.begin(9600);
  delay(3000); // wait for console opening
  if (! rtc.begin()) {
    Serial.println("Couldn't find RTC");
    while (1);
  }

  if (rtc.lostPower()) {
    Serial.println("RTC lost power, lets set the time!");
    // following line sets the RTC to the date & time this sketch was compiled
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
    // This line sets the RTC with an explicit date & time, for example to set
    // January 21, 2014 at 3am you would call:
    //rtc.adjust(DateTime(2017, 9, 6, 9, 33, 0));
  }
}

void loop () {
    DateTime now = rtc.now();
    Serial.print(now.year(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.day(), DEC);
```

```
    Serial.print(" (");
    Serial.print(daysOfTheWeek[now.dayOfTheWeek()]);
    Serial.print(") ");
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.println();
    Serial.print(" since midnight 1/1/1970 = ");
    Serial.print(now.unixtime());
    Serial.print("s = ");
    Serial.print(now.unixtime() / 86400L);
    Serial.println("d");
    delay(3000);
}
```

Exercice :

Ajouster le temps sur l'heure de Paris.

3.4 GPS avec Wemos D1 – NEO-6M

L'exemple suivant montre comment utiliser un simple modem GPS basé sur le circuit EO-6M. Ici le module communique avec lz carte Wemos D1 via un lien de série type UART.

Figure 3.5. NEO-6M GPS module connecté à la carte Wemos D1 par la biais d'un bus UART

La connexion du module NEO-6M est réalisé par le bus UART avec les pins :

NEO-6M	Wemos D1
GND	GND
VCC	3.3V
RxD	D7
TxD	D6

3.4.1 Code Arduino avec la bibliothèque `TinyGPS++.h`

```
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
static const int RXPin = D6, TXPin = D7;  // UART connection
static const uint32_t GPSBaud = 9600;
// The TinyGPS++ object
TinyGPSPlus gps;
// The serial connection to the GPS device
SoftwareSerial ss(RXPin, TXPin);

void setup() {
  Serial.begin(9600);
  ss.begin(GPSBaud);
  Serial.println("Testing attached GPS module to Wemos D1 board");
}

void loop() {
  // displays information every time a new sentence is correctly encoded
  while (ss.available() > 0)
    if (gps.encode(ss.read()))
      displayInfo();
  if (millis() > 5000 && gps.charsProcessed() < 10) {
    Serial.println("No GPS detected: check wiring.");
   // while(true);
  }
}
```

```
void displayInfo() {
  Serial.print("Location: ");
  if (gps.location.isValid()) {
    Serial.print(gps.location.lat(), 6);
    Serial.print(",");
    Serial.print(gps.location.lng(), 6);
  } else {
    Serial.print("INVALID");
  }

  Serial.print("  Date/Time: ");
  if (gps.date.isValid()) {
    Serial.print(gps.date.month());
    Serial.print("/");
    Serial.print(gps.date.day());
    Serial.print("/");
    Serial.print(gps.date.year());
  } else {
    Serial.print("INVALID");
  }
  Serial.print(" ");
  if (gps.time.isValid()) {
    if (gps.time.hour() < 10) Serial.print("0");
    Serial.print(gps.time.hour());
    Serial.print(":");
    if (gps.time.minute() < 10) Serial.print("0");
    Serial.print(gps.time.minute());
    Serial.print(":");
    if (gps.time.second() < 10) Serial.print("0");
    Serial.print(gps.time.second());
    Serial.print(".");
    if (gps.time.centisecond() < 10) Serial.print("0");
    Serial.print(gps.time.centisecond());
    delay(400);
  } else {
    Serial.print("INVALID");
  }
  Serial.println();
}
```

3.5 Utiliser afficheurs OLED et TFT

3.5.1 Afficheur OLED avec Wemos D1

Un exemple du sketch pour tester les fonctions de l'afficheur **OLED SSD1306** connecté à la carte Wemos D1.

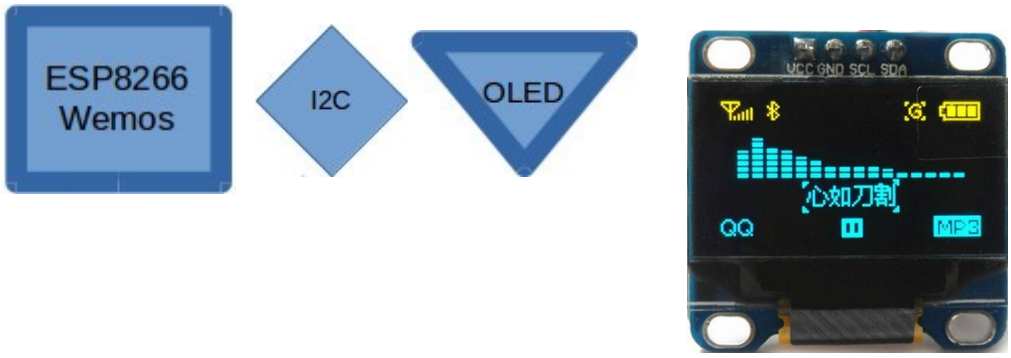

Figure 3.6. Un écran OLED (SSD1306) connecté à la carte Wemos D1 par la biais d'un bus I2C

```
#include <Wire.h>
#include "OLED.h"
OLED display(D2, D1);  // SDA - D2, CL - D1 : wemos D1

void setup() {
  Serial.begin(9600);
  Serial.println("OLED test!");
  // Initialize display
  display.begin();
  display.print("Hello World");
  delay(3*1000);
  display.print("Le module IoT de SMTR est le plus avance de tous les
enseignements techniques dans le departement ETN.");
  delay(3*1000);
  display.clear();
  delay(3*1000);

  display.print("TOP-LEFT"); // Test message postioning
  display.print("4th row", 4);
  display.print("RIGHT-BOTTOM", 7, 4);
  delay(3*1000);
  // Test display OFF
  display.off();
  display.print("3rd row", 3, 8);
  delay(3*1000);
  // Test display ON
  display.on();
  delay(3*1000);
}
int r = 0, c = 0;
void loop() {  // moving message
  r = r % 8;
  c = micros() % 6;
```

```
  if (r == 0)
    display.clear();
  display.print("Hello World", r++, c++);
  delay(500);
}
```

3.5.2 Utiliser afficheur TFT couleur ILI9341 (QVGA) avec Wemos D1

Cet exemple démontre comment afficher du texte sur le TFT avec un Wemos D1. La carte Wemos D1 lit la valeur d'un capteur analogique attaché à la broche A0, et écrit la valeur sur l'écran TFT, avec la mise à jour chaque
seconde.

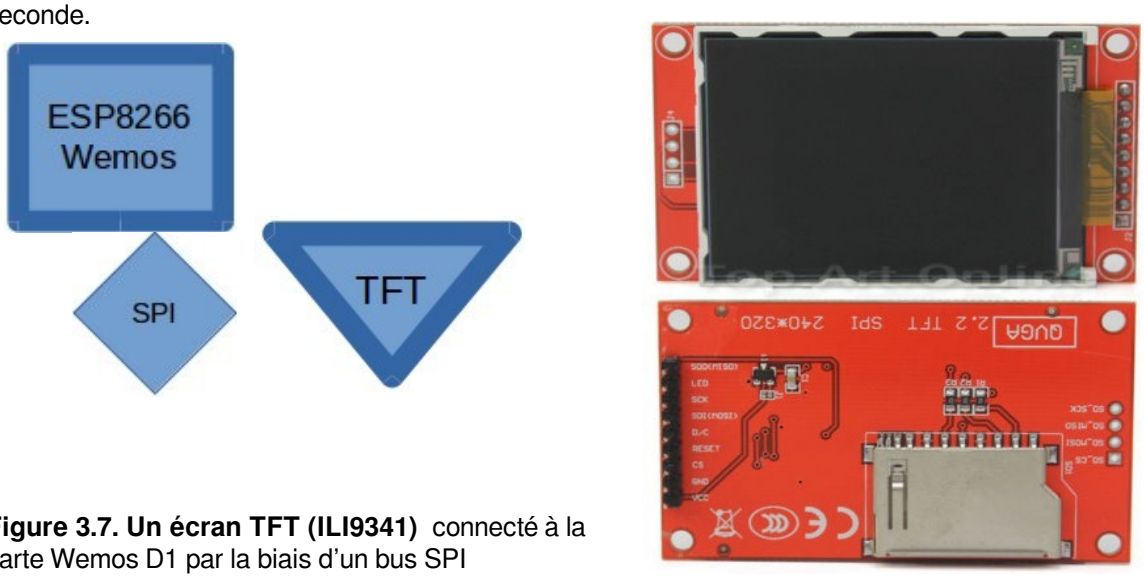

Figure 3.7. Un écran TFT (ILI9341) connecté à la carte Wemos D1 par la biais d'un bus SPI

```
#include "SPI.h"
#include "Adafruit_GFX.h"
#include "Adafruit_ILI9341.h"
// pin definition for the Wemos D1 (shield)
#define TFT_DC D4
#define TFT_CS D8 // Wemos
// create an instance of the library
Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC);
uint16_t channel_color[] = {
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE,      /* 255, 165,   0 */
  ILI9341_YELLOW,      /* 255, 255,   0 */
  ILI9341_GREEN,       /*   0, 255,   0 */
  ILI9341_CYAN,        /*   0, 255, 255 */
  ILI9341_MAGENTA,     /* 255,   0, 255 */
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE,      /* 255, 165,   0 */
  ILI9341_YELLOW,      /* 255, 255,   0 */
  ILI9341_GREEN,       /*   0, 255,   0 */
  ILI9341_CYAN,        /*   0, 255, 255 */
  ILI9341_MAGENTA,     /* 255,   0, 255 */
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE      /* 255, 165,   0 */
};
```

```
// char array to print to the screen
char sbuff[32];

void setup() {
  tft.begin();
  tft.setRotation(3);
  // clear the screen with a black background
   tft.fillScreen(ILI9341_BLACK);
  tft.setTextSize(2);
  tft.setTextColor(ILI9341_WHITE, ILI9341_RED);
  tft.setCursor(0, 0);
  tft.print(" ETN-SMTR ");
  tft.setTextColor(ILI9341_WHITE, ILI9341_GREEN);
  tft.print(" WiFi ");
  tft.setTextColor(ILI9341_WHITE, ILI9341_BLUE);
  tft.print(" Afficheur");
  //tft.fillScreen(ILI9341_BLACK);
  tft.setCursor(0, 32);
}

void loop() {

  // Read the value of the sensor on A0
  int lumv = analogRead(A0);
  // set the font color
   //tft.fillScreen(ILI9341_BLACK);
     tft.setTextSize(4);
   tft.setTextColor(ILI9341_WHITE, ILI9341_RED);
  // print the sensor value
    tft.setCursor(64, 128);
    sprintf(sbuff,"LUM:%d",lumv);
    tft.print(sbuff);
  // wait for a moment
  delay(1000);
}
```

Exercices :

Afficher les données captées par différents capteurs sur l'écran OLED (SSD1306).

- Température et Humidité - avec la carte d'extension et circuit SHT30
- Horloge – composant DS3231
- GPS data – module NEO-6M

Le même sujet avec l'affichage des donnés GPS sur un écran TFT