ARD2	ESP32 : Commande en PWM	Durée 2h.

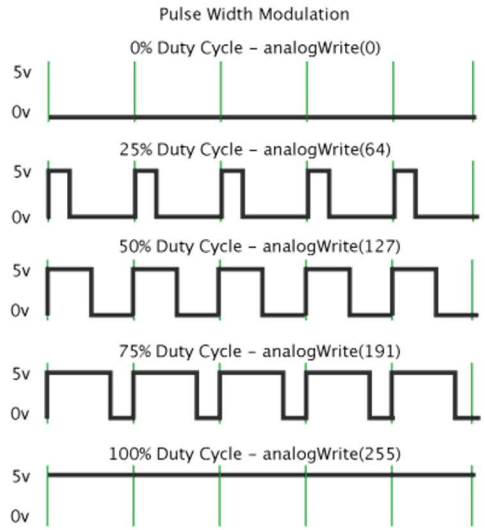
Le développement se fait sous l'IDE Arduino. Choisir la carte NodeMCU-32S

Type de carte: "NodeMCU-32S"
Upload Speed: "921600"
Flash Frequency: "80MHz"
Port: "COM13"

Les projets sont à sauvegarder dans l'espace de travail sous KWARTZ, sous répertoire **Arduino_ESP32**.

1 – Présentation d'un signal en PWM

Un signal en PWM (*Pulse Width Modulation ou Modulation de Largeur d'Impulsion MLI)* permet de générer un signal carré à rapport cyclique variable.
Sous Arduino "classique" on peut faire appel à la fonction ***analogWrite(pin, value)***



Avec la carte ESP 32, on peut faire appel :

- Aux fonctions ledc, prévues pour le contrôle de luminosité de leds
- Aux fonctions MCPWM, prévues pour la commande de moteurs.

Par soucis de simplicité, on utilise les fonctions ledc.

Pour le contrôle de leds, le microcontrôleur dispose de 8 timers et 16 canaux.

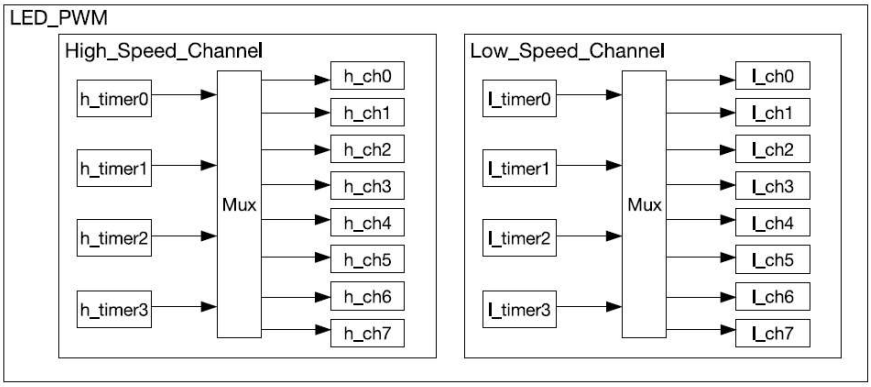


Figure 84: LED_PWM Architecture

La bibliothèque ledc associe les timers et les canaux de la manière suivante :

```
]/*
  * LEDC Chan to Group/Channel/Timer Mapping
** ledc: 0  => Group: 0, Channel: 0, Timer: 0
** ledc: 1  => Group: 0, Channel: 1, Timer: 0
** ledc: 2  => Group: 0, Channel: 2, Timer: 1
** ledc: 3  => Group: 0, Channel: 3, Timer: 1
** ledc: 4  => Group: 0, Channel: 4, Timer: 2
** ledc: 5  => Group: 0, Channel: 5, Timer: 2
** ledc: 6  => Group: 0, Channel: 6, Timer: 3
** ledc: 7  => Group: 0, Channel: 7, Timer: 3
** ledc: 8  => Group: 1, Channel: 0, Timer: 0
** ledc: 9  => Group: 1, Channel: 1, Timer: 0
** ledc: 10 => Group: 1, Channel: 2, Timer: 1
** ledc: 11 => Group: 1, Channel: 3, Timer: 1
** ledc: 12 => Group: 1, Channel: 4, Timer: 2
** ledc: 13 => Group: 1, Channel: 5, Timer: 2
** ledc: 14 => Group: 1, Channel: 6, Timer: 3
** ledc: 15 => Group: 1, Channel: 7, Timer: 3
*/
```

Par exemple, le canal 0 et le canal 1 utilise le même timer.

2 – Génération d'un signal en PWM sur carte ESP32

⇨ Tester le programme suivant (et ouvrir le moniteur) :

```
#define OUT1 2

#define FREQ 5000
#define CANAL0 0
#define RESOLUTION 10 //Resolution 8, 10, 12, 15

int RAPPORT=512;

void setup() {
 Serial.begin(115200);
 pinMode(OUT1,OUTPUT);
 ledcSetup(CANAL0, FREQ, RESOLUTION);
 ledcAttachPin(OUT1, CANAL0);
}

void loop() {
 //PWM entre 0 et 1023 avec résolution 10 bits
 Serial.print("Rapport cyclique de : ");Serial.print((RAPPORT*100)/1024);Serial.println(" %");
 ledcWrite(CANAL0, RAPPORT);
 delay(2000);
}
```

⇨ Relever le signal généré (et modifier le programme) pour un rapport cyclique de 25%, 50% et 75%. Mesurer pour chaque cas le rapport cyclique et la fréquence. Mettre les relevés dans le compte rendu.

⇨ Recommencer (sans enregistrer les relevés) la manipulation pour une fréquence de 20 kHz, puis une fréquence de 100 kHz. Conclure.

⇨ Recommencer la manipulation pour une fréquence de 100kHz en modifiant la valeur de la résolution à la valeur de 8 bits. Conclure.

3 – Génération de plusieurs signaux en PWM

⇨ Compléter le programme précédent pour avoir 2 signaux (sorties 2 et 4) avec la même fréquence (5kHz) et des rapports cycliques différents (25% et 50%), en utilisant les canaux 0 et 1.

⇨ Relever les signaux à l'oscilloscope pour vérifier le résultat.

⇨ Modifier le programme pour avoir 2 fréquences différentes pour les 2 signaux (rapports cycliques de 25% et 50%), toujours en utilisant les canaux 0 et 1. Relever les signaux et conclure.

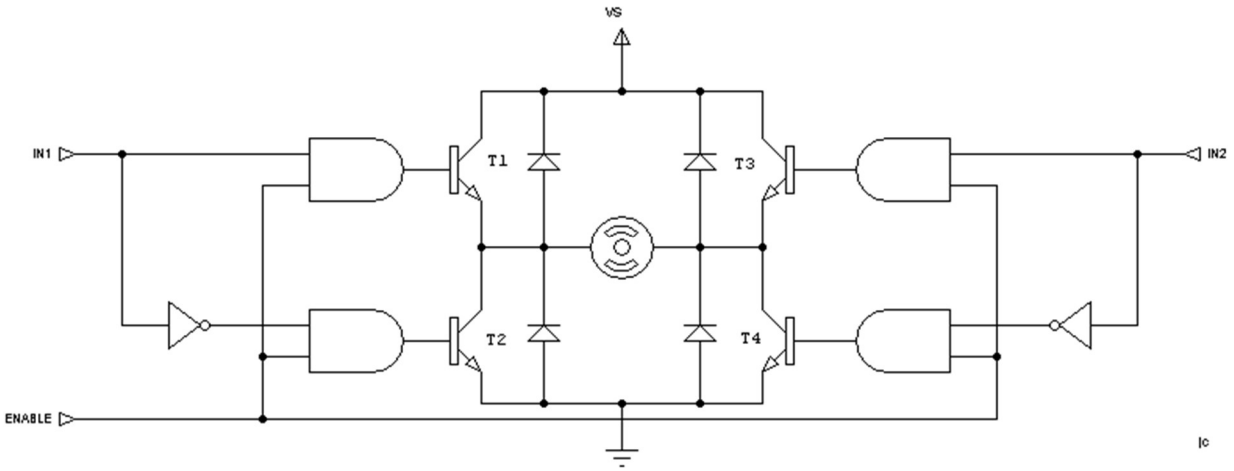
⇨ Refaire la même tentative avec les canaux 0 et 2. Conclure et essayer de donner une explication.

4 – Commande d'un moteur en PWM

Pour cette partie, il faut utiliser la maquette avec l'interface de puissance L298 et le moteur à courant continu.

Le circuit L298 (documentation en annexe) est une interface de puissance qui peut commander 2 moteurs à courant continu.

Une fois câblé avec les diodes de roues libres et le moteur, le schéma de l'interface pour un moteur est le suivant :



Un programme de test permettant de faire varier la vitesse du moteur est donné ci-dessous :

```
#define In1 13
#define In2 14
#define EnA 12
#define FREQ  20000
#define CHANEL0 0
#define RESOLUTION 10

int RAPPORT=0;
String ST;

void setup() {
  Serial.begin(115200);
  Serial.setTimeout(1);
  pinMode(In1,OUTPUT);
  pinMode(In2,OUTPUT);
  pinMode(EnA,OUTPUT);
  ledcSetup(CHANEL0, FREQ, RESOLUTION);
}

void MoteurA_Sens1(unsigned int vitesse){
  digitalWrite(In2,LOW);
  digitalWrite(EnA,HIGH);
  ledcAttachPin(In1, CHANEL0);
  ledcWrite(CHANEL0, vitesse);
}

void loop(){
  ST="";
  if (Serial.available() > 0){
    ST=Serial.readString();
    RAPPORT=ST.toInt();
  }
  MoteurA_Sens1(RAPPORT);
}
```

⇨ En consultant le site Arduino, indiquer le rôle des fonctions suivantes :

- Serial.available()
- Serial.readString()
- Serial.setTimeout()
- ST.toInt() ⇨ fonction sur chaîne de caractères (String)

⇨ Tester le programme et préciser les valeurs entrées dans le moniteur pour avoir un rapport cyclique de 50% et 75%.

⇨ Proposer et tester un sous-programme pour faire tourner le moteur dans l'autre sens. Faire constater au prof.