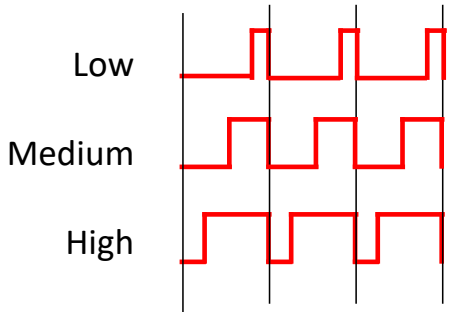
Digital and Embedded Systems
Professor Thomas Bräunl

Tutorial 6 – Actuator Programming in C

1. DC MOTORS

The CPU needs to generate a rectangle signal at a high frequency (e.g. 1kHz – 20 kHz). The uptime divided by the cycle time is called the plus-width-ratio [0% .. 100%] and determines the power setting for the motor.



Write a **C subroutine** that has the PWM ratio as a parameter (0 ..100) and generates the desired PWM curve at 1kHz.

TTGO Solution 1 – Switch IO-Pin

```
pinMode(1, OUTPUT); // init pin 1 as output

void PWM(int ratio)
{ while (…) // e.g. run until button press
   { digitalWrite(1, 1);  // set pin 1 to high
     … // wait
     digitalWrite(1, 0);  // set pin 1 to low
     … // wait
   }
}
```

TTGO Solution 2 – Use internal PWM function

```
void setup()
{ // setup PWM: channel, freq., bits resolution
  ledcSetup(PWM_CH, 500, 8);
  ledcAttachPin(PWM_pin, PWM_CH);
}
void loop()
{ // changing PWM range, [0..255] for 8 bits
  ledcWrite(PWM_CH, rate);
  …
}
```

2. SERVOS

Servos also take a PWM input but are quite different to DC motors. The PWM frequency has to be a low 50Hz (cycle time 20ms) and the uptime needs to be between 0.5ms and 2ms.
A short uptime moves the servo head to the left position; a long uptime moves it to the right position. The servo always runs with its standard speed.

Write a **C subroutine** that generates a PWM ratio to drive a servo.

3. DC MOTOR CALIBRATION

The PWM-percentage is usually not linear to the idle speed of a DC motor (and under load everything changes again). Some motors need ~30% PWM to even get started and reach at 69% PWM already 90% of their speed.
If you take some measurements for each desired motor idle speed (e.g. 10%, 20%, … 100%) and note the required PWM ratio, you can set up a calibration table.
Form this into an array of integer values and you can then select the correct PWM value in software.