Motor Programming

Smetankin Daniel

Intro

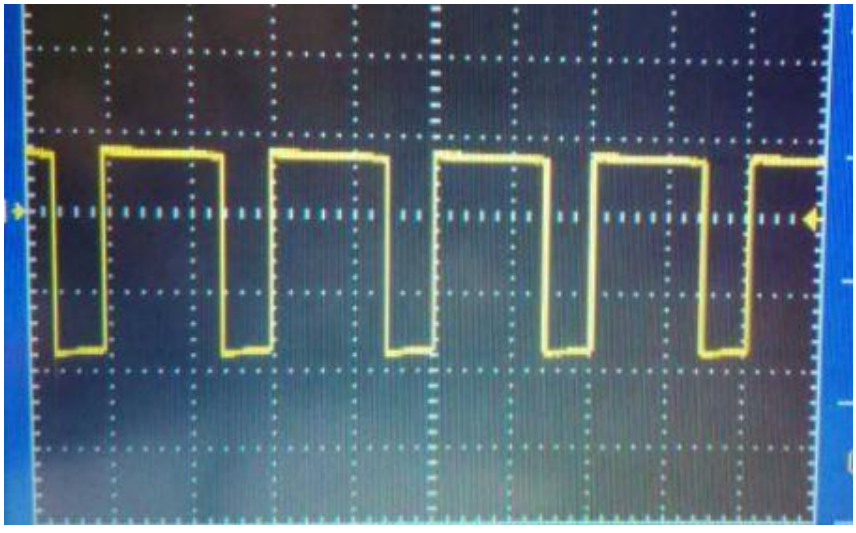
In this report the programming of the motor of the car and the hardware that you need to control the motor.

Materials and Methods

HAMEG HM 8040-3 Power supply
HAMEG HM 8012 MultiMeter
HAMEG HMO300 oscilloscope
NodeMcu
Breadboard
Jumper cables
DC servomotor
Ardumoto by sparkfun (Used as a H- Bridge)

Results

There weren´t any results. Because the voltage, the ardumoto board was getting was insufficient. This can be explained because the ardumoto is made to put on top of an Arduino so it can supply it with the correct voltages. I tried to copy these voltages with an HAMEG power supply but failed. But on the input pin of the ardumoto I was getting a correct PWM signal as u can see bellow. So theoretically my test would work if I would use a L298N H-Bridge instead of an ardumoto because L298N does not depend on an mcu suppling it the right voltages.

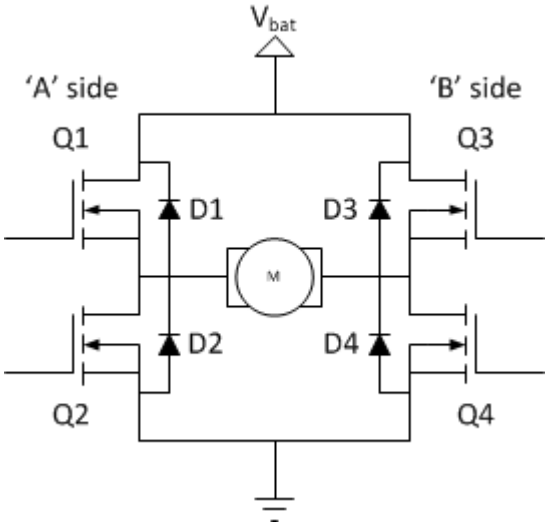


Information & Conclusion

H-Bridge

Because we cannot power the motor from a pin of the nodemcu we will need an H-Bridge.

In general an H-bridge is a rather simple circuit, containing four switching element, with the load at the center, in an H-like configuration:

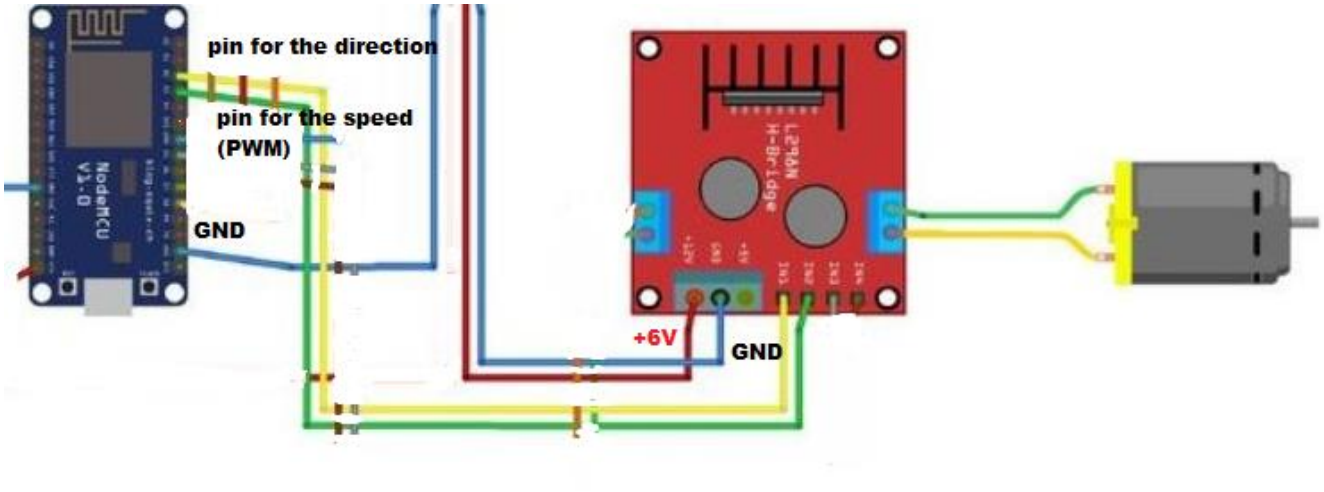


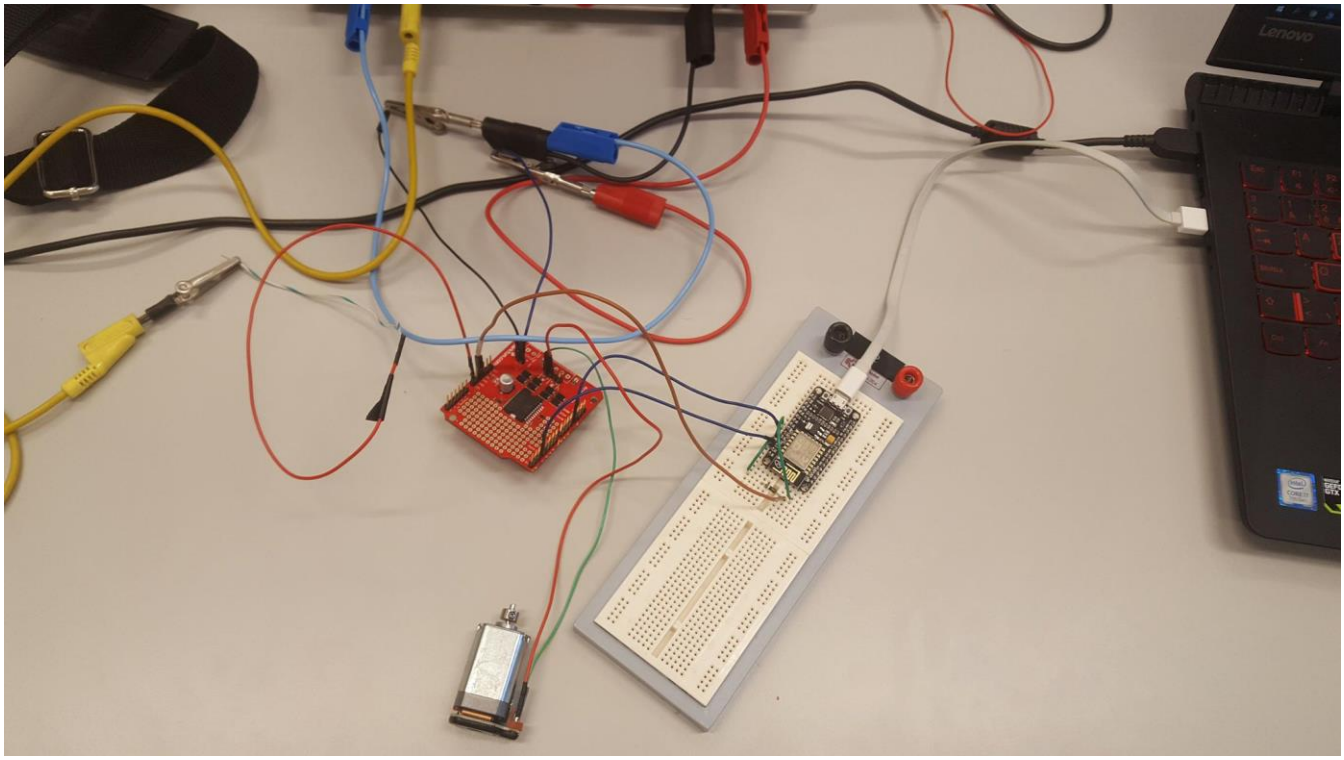
The switching elements (Q1..Q4) are usually bi-polar or FET transistors, in some high-voltage applications IGBTs. Integrated solutions also exist but whether the switching elements are integrated with their control circuits or not is not relevant for the most part for this discussion. The diodes (D1..D4) are called catch diodes and are usually of a Schottky type.**[3]**

The top-end of the bridge is connected to a power supply (battery for example) and the bottom-end is grounded.
In general all four switching elements can be turned on and off independently, though there are some obvious restrictions. Though the load can in theory be anything you want, by far the most pervasive application if H-bridges is with a brushed DC or bipolar stepper motor (steppers need two H-bridges per motor) load. In the following I will concentrate on applications as a brushed DC motor driver. **[3]**

Setup

The connection is simple you connect the ground of the nodemcu to the ground of the H-Bridge. And power the H-bridge with the preferred voltage , witch in our case is 6V for the DC motor. Then you connect 2 pins from the nodemcu to the H-bridge. 1 will send a digital signal the other one a analogue signal. The digital signal will represent the direction of the DC motor and the analogue value will represent the speed. Analogue value comes from a PWM supported pin. And the final step is to connect the DC motor to the H-bridge.





Programming

The circuit is based on ESP8266, and the software is written in the Arduino IDE. We didn´t open the car yet so we don´t know what kind of motor powers the car. But we will assume it is a standard servo motor attached to a PWM to regulate the vehicle's speed. A value of 0 to 1023 can be send to the PWM witch will control the cars speed**.[1]**

1. Defining

Here are de most important variables defined.

```
#define PIN_MOTOR_A_IA D2
#define PIN_MOTOR_A_IB D3

#define PIN_MOTOR_B_IA D5
#define PIN_MOTOR_B_IB D6


int motor_speed;
bool motor_dir;
int force_turn;
```

2. MotorSpeed

Here servomotor A is configured witch is responsible for moving the car back of forward. MotorSpeed will read an integer m_speed which can have a value between 0 and 1023. With 1023 a value that represents the full power of the servomotor. Motor_dir is a parameter which is used to check in what direction the car wants to move, straight ahead or reverse. The Boolean dir == true, when the car wants to drive straight ahead and dir == false when the car wants to reverse. The if statement checks if dir == true , if it is it will write an analogue value of 1023-PWM to input D2, and it will send a digital 1 or HIGH to input D3. If it is not true it will send the analogue value of the variable pwm , and a digital 0 or LOW.

```
void MotorSpeed(int m_speed){
  MotorSpeed(m_speed, motor_dir);
}

void MotorSpeed(int m_speed, bool dir){
  int pwm = m_speed % 1024;
  motor_speed = pwm;
  motor_dir = dir;
  if( dir ){
    analogWrite(PIN_MOTOR_A_IA, (1023 - pwm));
    digitalWrite(PIN_MOTOR_A_IB, HIGH);
  }else{
    analogWrite(PIN_MOTOR_A_IA, pwm);
    digitalWrite(PIN_MOTOR_A_IB, LOW);
  }
}
```

3. MotorTurn

Here a different servomotor is configured the B servomotor which is responsible for the turning of the car. MotorTurn will read a variable turn of which the absolute value will be calculated. If the absolute value of variable turn is smaller then 0 the car will go left. If this value is bigger then 0 it will turn right , and if the value is equal to 0 the car will keep going straight or is going to keep reversing, this depends on the direction of motor A. To control the motor it is the same method as in MotorSpeed. An analogue value will ben send to 1 pin of the motor more specific to the PWM (Pulse With Modulation) which determines de position of the servomotor. Then a digital value will be send to another pin of the same motor , if this value is low the motor will reverse the rotation direction, if it is HIGH the motor will keep turning with its original rotation direction.

```
void MotroTurn( short turn ){
  force_turn = abs(turn);
  if( turn < 0 ){
    MotroTurnLeft();
  }else if( turn > 0 ){
    MotroTurnRight();
  }else if(turn == 0){
    MotroTurnStraight();
  }
}
```

```
void MotroTurnLeft(){
  analogWrite(PIN_MOTOR_B_IA, force_turn);
  //digitalWrite(PIN_MOTOR_B_IA, HIGH);
  digitalWrite(PIN_MOTOR_B_IB, LOW);
}

void MotroTurnRight(){
  int pwm = 1024 - force_turn;
  analogWrite(PIN_MOTOR_B_IA, pwm);
  //digitalWrite(PIN_MOTOR_B_IA, LOW);
  digitalWrite(PIN_MOTOR_B_IB, HIGH);
}

void MotroTurnStraight(){
  analogWrite(PIN_MOTOR_B_IA, 0);
  //digitalWrite(PIN_MOTOR_B_IA, LOW);
  digitalWrite(PIN_MOTOR_B_IB, LOW);
}
```

Reference List

1. http://fritzing.org/projects/diy-wifi-rc-car-with-esp8266-and-arduino-ide
2. http://geek.adachsoft.com/home/article/id/27/n/DIY-WIFI-RC-car-with-ESP8266-and-Arduino-IDE
3. https://www.engineersgarage.com/contribution/dc-motor-control-using-h-bridge