Delhi Technological University

Department of Electrical Engineering

Microprocessors & Microcontroller Applications
EE-306

SPEED CONTROL OF BLDC MOTORS FOR ELECTRIC VEHICLES

SUSHRUT MANU SINGH, 2K18/EE/215
YATHARTH AHUJA, 2K18/EE/249

Project Report
May 2021

Acknowledgement

We would like to express our deep sense of respect and gratitude to our project mentor Prof. Anupama of Department of Electrical Engineering, DTU, for providing the opportunity of carrying out this project and being the guiding force behind this work. We are deeply indebted for the support, advice and encouragement he provided without which the project could not have been a success.

Sushrut Manu Singh, 2K18/EE/215
Yatharth Ahuja, 2K18/EE/249

Contents

List of Figures

List of Tables

1 Introduction

Figure 1: *Vehicle driven by BLDC motor*

An electric motor drive is a power electronic circuitry which is needed to drive a motor. When these drives are incorporated with micro-controllers they can control the speed and torque of the motor. It controls the motor by varying the frequency of the electrical energy send to the motor. A standard inverter based drive can control the speed and torque of a motor, but a servo drive can be used to control the position of motor along with the speed and torque. The frequency is controlled by modulating the pulse width of the supply voltage, this technique is known as pulse width modulation (PWM).

Depending on a particular technical solution and selected control principle, different sensors for measuring of currents, voltages, velocity, acceleration, torque etc. in an electric drive are used. Other information, like pressure signal for controlling pumps and compressors, air humidity or temperature signal for controlling of fans etc. is also necessary. For that reason, the controller of the drive must process different information.

Microprocessor technique enables us to apply different control methods in today's control systems, including control of electric drives. In the drive controller, the signal processors with inbuilt analogue-digital converters, timers, pulse-width modulators and other devices, which simplify drive control, are commonly used. Today the numeric control of electric drives is widely used and the traditional analogue sensors will be replaced with the new digital sensors for measuring of speed and position.

2 Brushless DC Motors

Brushless DC motors do not use brushes. With brushed motors, the brushes deliver current through the commutator into the coils on the rotor. The rotor is a permanent magnet so that the coils do not rotate, but are instead fixed in place on the stator. Because the coils do not move, there is no need for brushes and a commutator.

With the brushed motor, rotation is achieved by controlling the magnetic fields generated by the coils on the rotor, while the magnetic field generated by the stationary magnets remains fixed. To change the rotation speed, you change the voltage for the coils. With a BLDC motor, it is the permanent magnet that rotates; rotation is achieved by changing the direction of the magnetic fields generated by the surrounding stationary coils. To control the rotation, you adjust the magnitude and direction of the current into these coils.

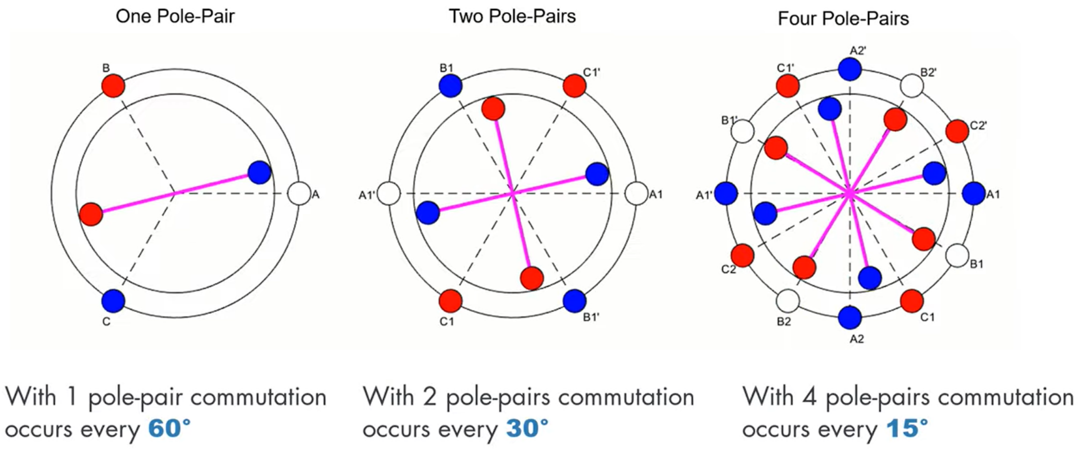
The brushless DC motor drive is used in record players, the tape drive for recorders, spindle drive in hard disks for computers, and low power drives in computers peripherals instruments and control systems. They also have applications in aerospace, in biomedical and in driving cooling fans, etc.

2.1 ADVANTAGES OF BLDC MOTORS

The major advantage of using a Brushless DC motor is its efficiency. In contrast with Brushed DC motors these motors provide maximum torque continuously when compared to the Brushed motors, which can reach the maximum torque only at specific points during the rotation. In order to get the same torque as a BLDC motor, a Brushed motor needs larger magnets which will increase it's size.

Another advantage is that BLDCs are fully controllable, they provide extremely precise control over the torque and the speed of rotation using feedback mechanisms implemented using controllers. This helps us reduce the energy consumption. In brushed motors the brushes and the commutator wear and tear due to continuous contact between them while the motor is operational. Due to the absence of brushes, BLDC motors have increased durability and low electric noise as there is no sparking during the commutation.

Because of their efficiency and durability, these motors are widely used in devices that run continuously for long duration. They are used in washing machines, air conditioners, fans and other consumer electronic devices because of their efficient nature. Brushless DC motors offers high torque to weight ratio, and high efficiency as compared to conventional induction motors.



With 1 pole-pair commutation occurs every 60°

With 2 pole-pairs commutation occurs every 30°

With 4 pole-pairs commutation occurs every 15°

Figure 2: *Commutation of a BLDC Motor*

Table 1: *Rotor position and corresponding phase voltage*

$\mathbf{H_A}$	$\mathbf{H_B}$	$\mathbf{H_C}$	$\mathbf{EMF_A}$	$\mathbf{EMF_B}$	$\mathbf{EMF_C}$
0	0	0	0	0	0
0	0	1	0	-1	+1
0	1	0	-1	+1	0
0	1	1	-1	0	+1
1	0	0	+1	0	-1
1	0	1	+1	-1	0
1	1	0	0	+1	-1
1	1	1	0	0	0

2.2 COMMUTATION OF BLDC MOTOR

The commutation of a Brushless DC motor is controlled electronically. To rotate the BLDC motor, the stator winding should be energized in a specific sequence. It is important to know the position of the rotor in order to understand which winding is energized following the energizing sequence. Rotor position is sensed using Hall Effect sensors embedded in the stator on the motor's non drying end. There are 6 different possible commutations because of these sensors. Based on the combination of these three Hall sensor signals, the exact sequence of commutation can be determined.

Hall effect sensors on the stators detect the position of the rotor by sensing the magnetic field. There are three Hall sensors on a typical PMBLDCs stator which work together to give six states of the rotor's position. In one complete rotation the excitation

Table 2: *Switching states of the inverter*

Rotor position	**Reference Currents**		
in degrees	$\mathbf{I_{ref1}}$	$\mathbf{I_{ref1}}$	$\mathbf{I_{ref1}}$
0 - 60	I_{ref}	$-I_{ref}$	0
60 - 120	I_{ref}	0	$-I_{ref}$
120 -180	0	I_{ref}	$-I_{ref}$
180 - 240	$-I_{ref}$	I_{ref}	0
240 - 300	$-I_{ref}$	0	I_{ref}
300 - 360	0	$-I_{ref}$	I_{ref}

of the stator winding is changed six times. Only two phases are active in a given period of time while the third phase remains inactive according to the commutation table given below.

2.3 COMMUTATION USING MICRO-CONTROLLER

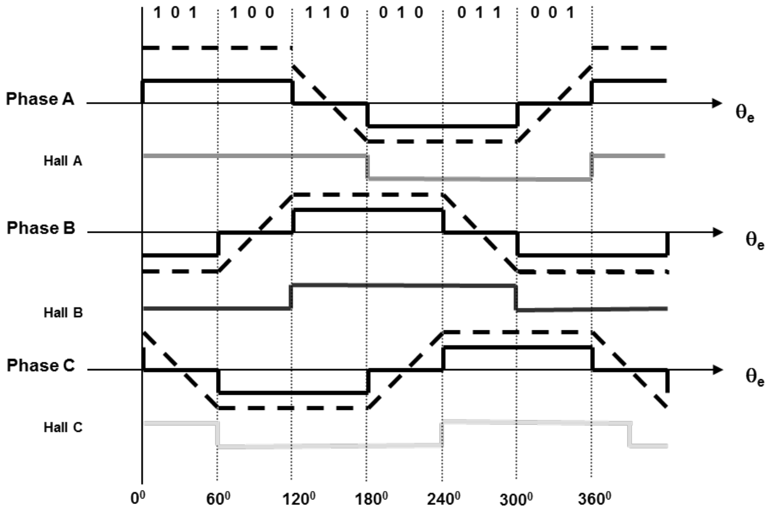


Figure 3: *Back EMF with respect to Hall signals*

When the BLDC motor rotates, each winding (3 windings) generates back EMF which opposes the main voltage. The 3 generated back EMF signals are 120° out of phase which is the same as the hall effect sensor signals. We use these signals to find out the position of the motor and energize the windings accordingly. The back EMF signals have phase shift of 30°. In every energizing sequence, two windings are energized and the third winding is left open. The open winding is used to detect the zero crossing, thus, the combination of all 3 zero cross over point are used to generate the energizing sequence. There are 3 zero crossings -

1. Phase A zero crossing
2. Phase B zero crossing
3. Phase C zero crossing

We detect the zero crossing events is by using comparators in the microcontroller. The comparator has 3 terminals: 2 inputs (positive and negative) and an output. Comparator output is logic high if the positive voltage is greater than the negative voltage, and logic low if the positive voltage is lower than the negative voltage. We need one comparators for each phase of the motor.

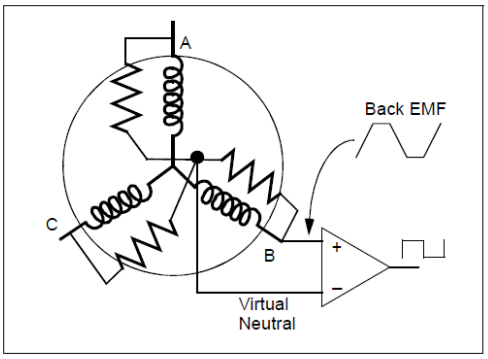
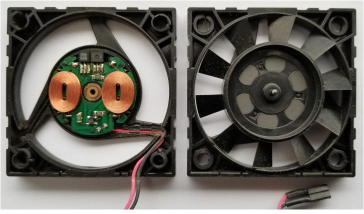


Figure 4: *Connections of comparator between phase A & B*

When the back EMF generated in the floating (open) winding crosses the zero point towards positive side, the comparator output makes a transition from low-to-high. When the back EMF generated in the floating winding crosses the zero point towards negative side, the comparator output makes a transition from high-to-low. By having three such comparator circuits, one on each of the phases gives three digital signals corresponding to the back EMF signal in the windings. The combination of these three signals is used to derive the commutation sequence.

3 Literature Review



Generally, in the established system brushed dc and conventional ac electric motors have been used very broadly. However, with respect to the recent advancements in magnet technology, power electronics and controllers has led to development of various motor of extended utility in drive systems like Switched Reluctance Motors (SRM), Permanent Magnet Synchronous Motors (PMSM) and Permanent Magnet Brushless DC Motors (PM BLDC)[1]. They are able to offer a higher efficiency, greater reliability, and more power density requiring less maintenance[2]. The control of such motors have been studied for long. A study presented by Ali Emadi et. al.[3] is done on development of low-cost IC based controller of BLDC motor which further instigates the development of such technology and implementation for a BLDC motor drive system.

Further we observe in study by Derammelaere et. al.[4] that BLDC machines offer a better performance in high speed applications and applications where continuous rotation is warranted. Whereas, PMSM machines offer better positioning characteristics but inflict expenses. In BLDC motors, the commutation relies on electronics, unlike in brushed DC motors[5]. Stator windings are energised in a sequence to cause rotation in BLDC and corresponding rotor position can be sensed by hall sensors as done in [6]. For this, most BLDC motors deploy 3 hall sensors inside the stator on the non-driving end of the motor.

Hence, the Hall sensors give a high or low signal when rotor magnetic poles pass from near the sensors. Hence, combination of 3 hall sensors can help us determine commutation sequence. Another type of commutation sequence sensing is through the method of Back-EMF Zero Crossing Detection method (Terminal Voltage Sensing) in which back EMF is sensed and triggers are passed as the values crosses zero. Trigger may be as simple as an RC time constant [7].

Apart from indirect sensing methods, the sensing methods exploit the fact that motors winding inductances are almost the same and do not vary with the rotor position. However in cases of IPM, the Free-wheeling Diodes Conduction Detection method (Terminal Current Sensing) helps us tackle the problem of variable winding inductance[8].

Many control methods deploy a PWM variation based strategy. The most prevalent control methods in this category are that of conventional 120°, elimination of virtual neutral point, techniques for low speed, high speed and small power applications, and direct current controlled.

4 Equipment Used

Table 3: *List of Equipment used in the proposed system*

S.No.	Equipment Used	Description	Estimated Cost
1)	ATMega328P	Microcontroller used in Arduino UNO	₹150
2)	PIC16F887	8-bit microcontroller	₹140
3)	IR2104S	Gate Driver for MOSFET	₹130
4)	IRF630	N-Channel Power MOSFET	₹100
5)	Crystal Oscillator	16 MHz crystal	₹50
6)	Switch	Reset switch for controller	₹10
7)	Passive Components	Resistors, Capacitors, Diodes	₹50

4.1 IR2104S GATE DRIVER

A gate IR2104 driver is acts as a power amplifier that accepts a low-power input from a micro-controller and produces a high-current input pulse for the gate of a high-power transistor such as a power MOSFET. A gate driver is used to interface between the micro-controller and the MOSFETs used in the inverter circuit. It provides a necessary barrier between the switches carrying high high levels of current from the delicate micro-controller.

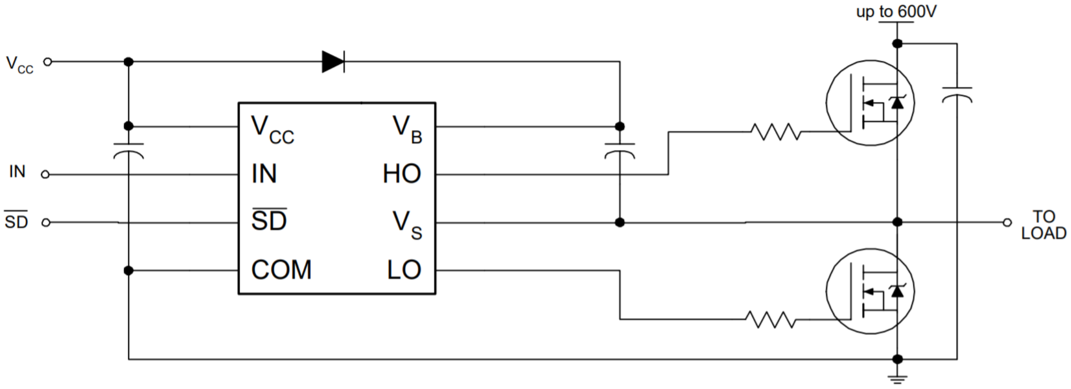


Figure 5: *Typical connections of the gate driver*

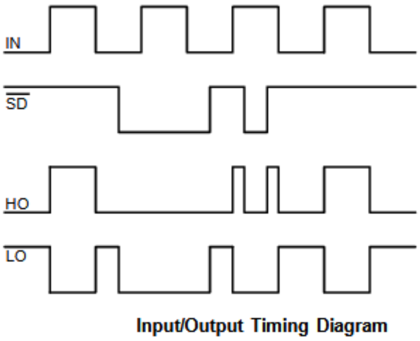


Figure 6: *Switching between HIGH and LOW side according to the IN and SD lines*

The operating voltage of IR2104S is 25 Volts and it can handle upto 600 Volts on the load side. The output drivers has a high pulse current buffer stage to reduce driver cross-conduction. The floating channel present can be used to drive an N-channel power MOSFET or IGBT in the high side configuration of 10 to 600 Volts.

4.2 IRF630 POWER MOSFET

IRF630 is a N-Channel, through the hole, power MOSFET. This MOSFET has extremely high dv/dt capability, very low intrinsic capacitances and gate charge minimized. The Drain to source voltage (V_{DS}) of the MOSFET is 200V and Gate to source voltage (V_{GS}) is ±20V. The Continuous drain current (I_D) is 9A and the Power dissipation (P_D) of the MOSFET is 75W

4.3 CRYSTAL OSCILLATOR

The 16 MHz Crystal Oscillator module is designed tohandle off-chip crystals that have a frequency of 4-16 MHz. The crystal oscillatorís output is fed to the System PLL as the input reference.

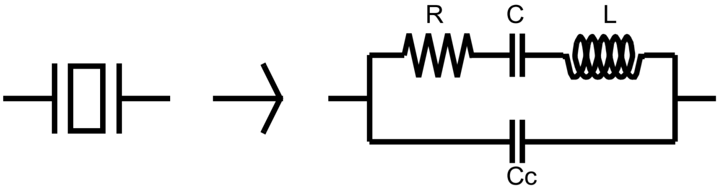


Figure 7: *Equivalent circuit diagram of the oscillator*

$$Frequency = \frac{1}{2\pi\sqrt{LC}} \tag{1}$$

4.4 ATMEGA328P MICRO-CONTROLLER

ATmega328P is a 8-bit micro-controller which can store 32 KB of data in programmable flash memory. It has 1024B EEPROM, 2 KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented Two-Wire serial interface, SPI serial port, a 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts.

This micro-controller is housed on an Arduino UNO board. The ATmega328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. ATmega328p allows 1 MIPS per MHz by executing instructions in a single clock cycle. This allows us to optimize the system for power consumption versus processing speed.

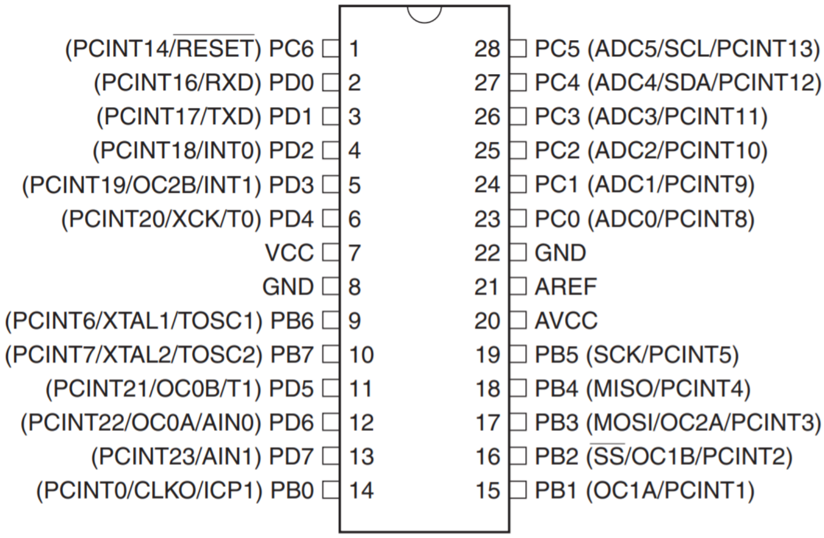


Figure 8: *Pinout of ATmega328P in 28 SPDIP package*

Instead of using the complete Arduino UNO board we have designed a custom bare-bones controller board by only the adding the necessary components for our application.

4.4.1 *Programming the registers of the Microcontroller*

A register is a memory space inside the CPU that can be operated upon rapidly. The individual bits of a register represent a specific function. The SRAM of ATmega328P is organised as follows: 32 registers, 64 I/O registers, 160 Ext I/O registers, 2048 Internal SRAM. These registers can be accessed through programming via the Arduino IDE. We can reduce the code size and power consumption and increase the speed of operation by programming the registers the microcontroller directly.

4.4.1.1 *DDRx, PORTx and PINx Registers*

1. **Data Direction Registers:** These registers specify whether the pins being used is either input or output. We can specify a pic as 0 (input) 1 (output). Pin PD7 corresponds to DDD0 in the DDRD register. We can initialise all pins with one single command instead of several pinMode commands-

   $$DDRD = 0b00001111; \tag{2}$$

2. **Internal Pull-Up Registers:** These registers can be used to enable internal pull-up of resistors of the input pins, these pins can be used as logic pins that can provide ON or OFF signals. We can specify PORTD4, PORTD5, PORTD6, and PROTD7 as 1s by using the following command-

   $$PORTD = 0b11110000; \tag{3}$$

3. **PINx:** This register reads state of Input Pins on port ‘x’. The PINx register simply reads the value from the pins. This value, as obvious, is digital. Although Port C pins have ADCs attached, they can still be used as digital GPIOs.

We can perform bitwise Operations on these registors to specify and keep track of these them easily. These opeartors can help us write code to alter state of only one pin while not disturbing the others. These 6 bitwise Operators are NOT(∼), AND (&), OR(|), EXCLUSIVE OR (∧), SHIFT LEFT («) and SHIFT RIGHT (»). Some examples of bitwise operator are given below, Eq. 4 turns port D2 high, Eq. 5 turns port D3 low. We can combine multiple equations to form Eq. 6 which turns port D1 and D3 high and port D2 to low.

$$PORTD| = (1 << PORTD1) \tag{4}$$

$$PORTD\& =\sim (1 << PORTD2) \tag{5}$$

$$PORTD| = (1 << PORTD0)\& \sim (1 << PORTD1)|(1 << PORTD2) \tag{6}$$

4.4.1.2 *Power Reduction Registers*

1. **Power Reduction ADC:** This is bit 0 of the register. Setting this PRADC bit shuts down the ADC. The ADC must be disabled before shut down. The analog comparator cannot use the ADC input MUX when the ADC is shut down.

2. **Power Reduction USART0:** This is bit 1 of the register. Setting this PRUSART0 bit shuts down the USART by stopping the clock to the module. When waking up the USART again, the USART should be re initialized to ensure proper operation.

3. **Power Reduction Serial Peripheral Interface:** This is bit 2 of the register. If using debugWIRE On-chip Debug System, this bit should not be written to one. Writing a logic one to this PRSPI0 bit shuts down the Serial Peripheral Interface by stopping

the clock to the module. When waking up the SPI again, the SPI should be re initialized to ensure proper operation.

4. **Power Reduction TWI0:** This is bit 7 of the register. Setting this bit shuts down the TWI 0 by stopping the clock to the module. When waking up the TWI again, the TWI should be re initialized to ensure proper operation.

4.4.2 *Connections of the Microcontroller*

The chip is connected to a stabilized supply voltage (V_{CC}) through Pin 7. Pin 8 and 9 are connected to the common ground. An external clock is needed for the controller to function, so we add a 16 MHz crystal oscillator between pins 9 and 10. We have added a small button switch between Pin 1 and ground to act as the RESET switch, which will start the program from the beginning, whenever it is pressed.

Pin 4,5 and 23-28 are used as analog I/O pins. We have used Pins 23 and 24 change the speed of the motor by connecting them to the ground through switches, when a button is pressed the micro-controller senses the change in voltage and alters the speed accordingly. Pins 13, 25 and 26 are used to sense the trapezoidal back EMF of the motor and carry out the commutation. While the digital I/O pins are used to trigger the IR2104S gate driver to switch the MOSFETs.

Pins 15, 16 and 17 can generate PWM signals where pin 15 and pin 16 are related with Timer1 module (OC1A and OC1B) and pin 11 is related with Timer2 module (OC2A). Both Timer modules are configured to generate a PWM signal with a frequency of about 31KHz and a resolution of 8 bits. The duty cycles of the PWM signals are updated when a pushbutton is pressed (speed up or speed down) by writing to their registers (OCR1A, OCR1B and OCR2A). The SD lines of the three IR2104S are connected to pins 17, 16 and 15 respectively for phase A, phase B and phase C.

The natural input is connected to Pin 12 and the the three phases of back EMF - A, B, and C are connected to pins 13,25 and 26 repectively. The analog comparator compares the positive input AIN0 (Pin 12) with the negative input which can be AIN1 (pin 13), ADC2 (pin 25) or ADC3 (pin 26). When the positive pin voltage is higher than the negative pin voltage, the output of the analog comparator ACO is set, and when the positive pin voltage is lower than the negative pin voltage, ACO is cleared.

4.5 PIC16F887 MICRO-CONTROLLER

The PIC16F887 microcontroller has two comparators. The positive input of this comparator is C1IN+ (pin 5) and the negative input can be multiple pins. These negetive input pins are C12IN0- (pin 2), C12IN1- (pin 3), C12IN2- (pin 36) or C12IN3- (pin 34). the virtual natural point is connected to the positive pin of the analog comparator C1IN+ (pin 5), phase A BEMF to C12IN0- (pin 2), C12IN1- (pin 3) and phase C BEMF to C12IN2- (pin 36). The comparator compares the virtual point with the BEMF of one phase.

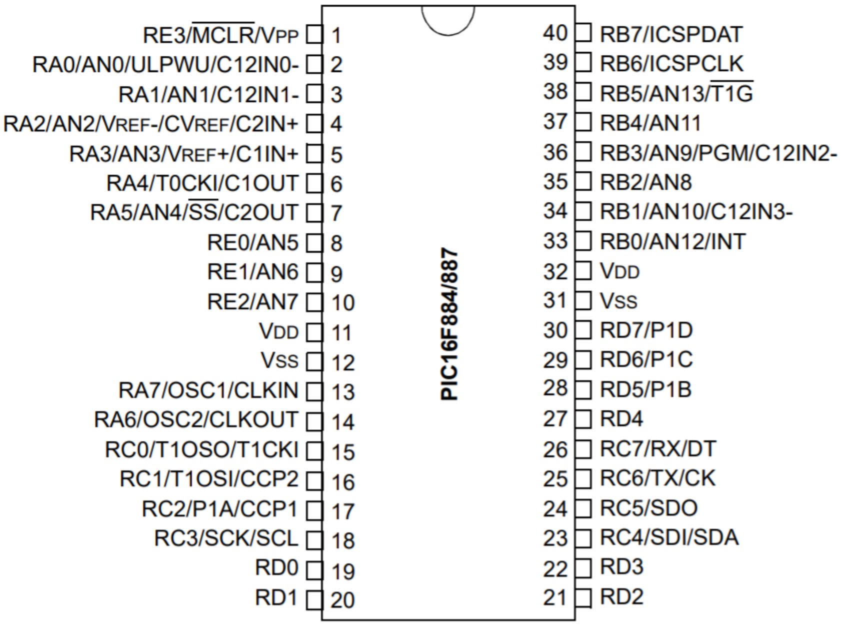


Figure 9: *Pinout of PIC16F887 in 40 PDIP package*

The HIN lines of the three IR2101 are connected to pins: P1D (RD7), P1C (RD6) and P1B (RD5) for phase A, phase B and phase C respectively. The LIN lines are connected to pins: RD4, RD3 and RD2. The PIC16F887 has one ECCP module (Enhanced CCP module) which can generate 4 PWM signals with the same frequency and controlled by one duty cycle. We use this module to supply pulses for the 3 MOSFETs. The speed of the BLDC motor is controlled by a potentiometer connected to analog channel AN4 (pin 7).

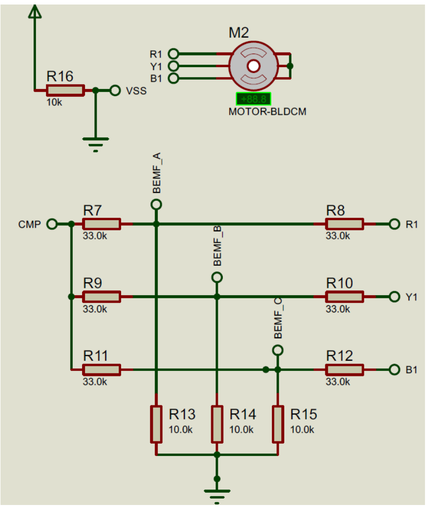


Figure 10: *Speed regulator and back EMF sensor connected to comparator*

5 Simulation and Results

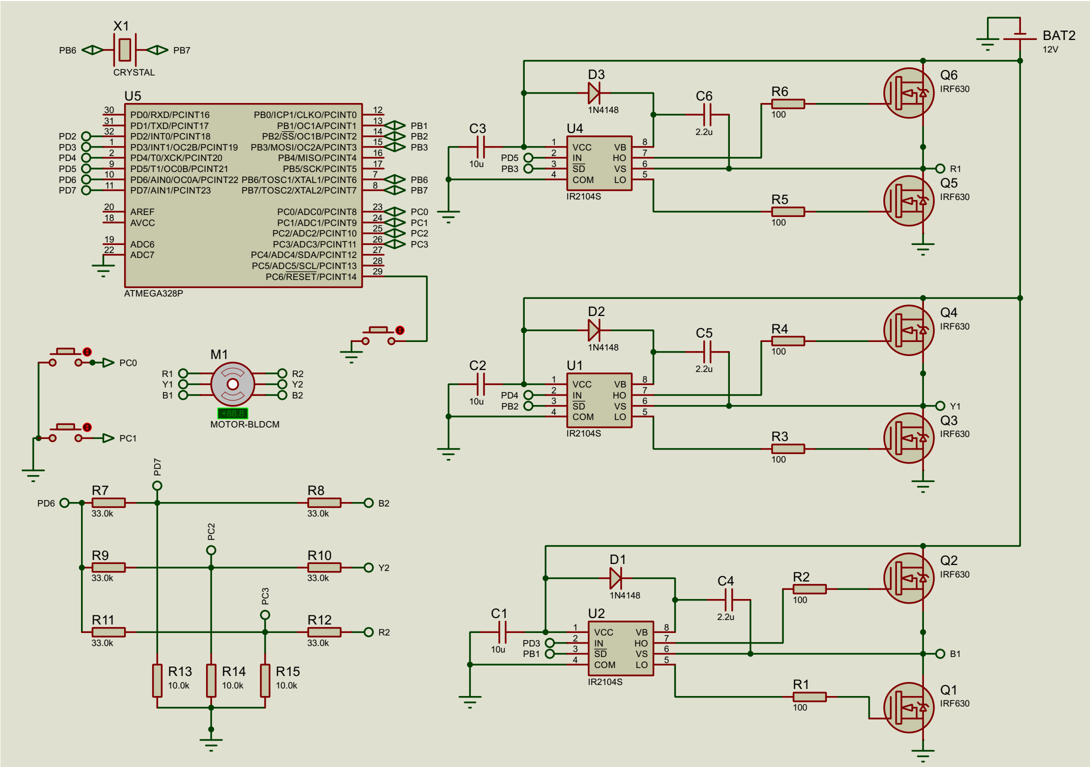


Figure 11: *Simulation of the system using ATmega328P in PROTEUS*

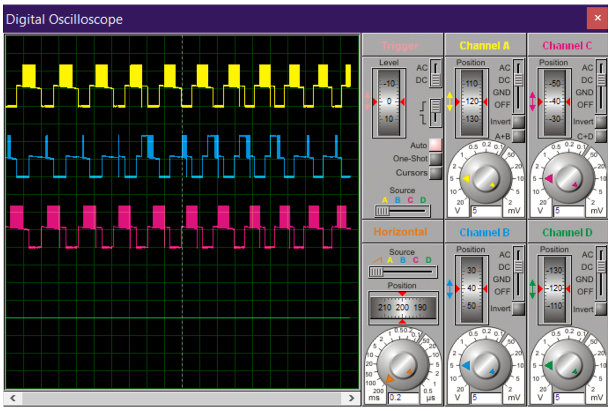


Figure 12: *Pulses generated viewed on an Oscilloscope*

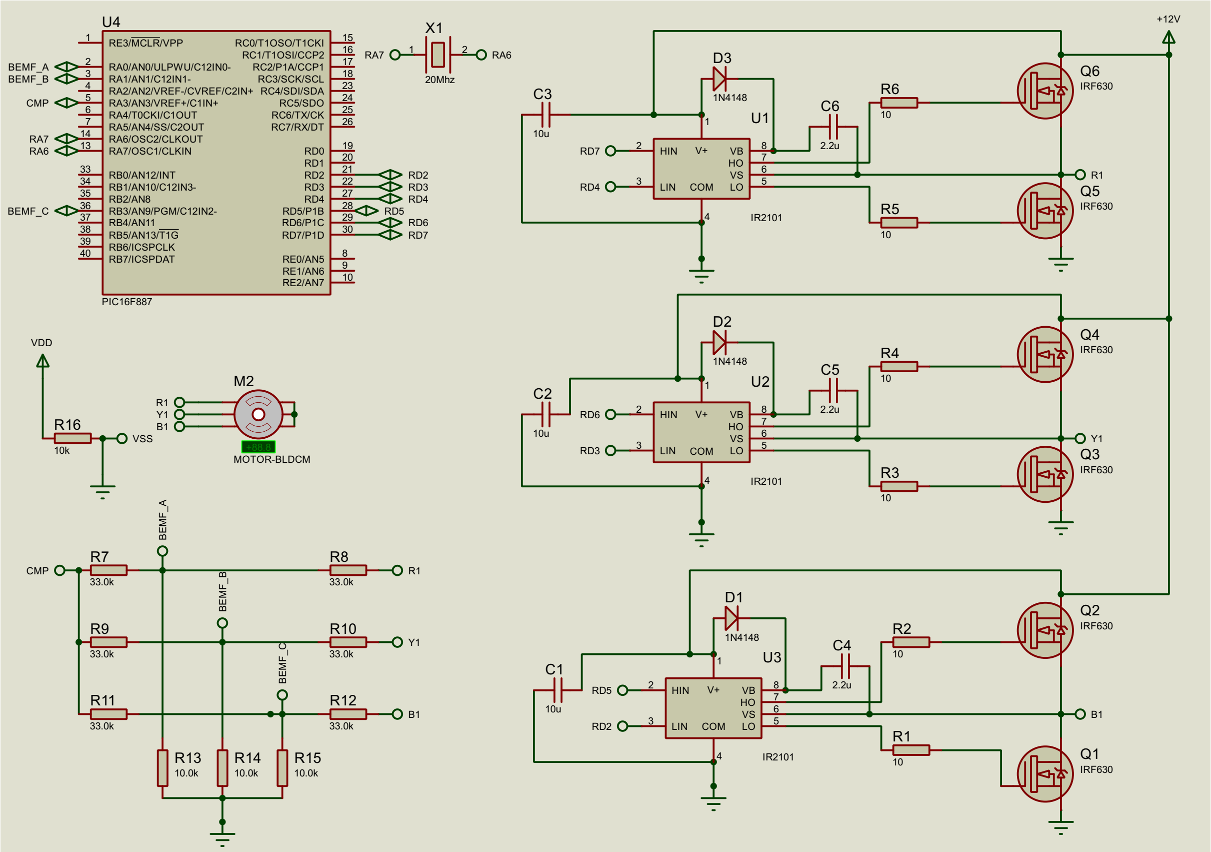


Figure 13: *Simulation of the system using PIC16F887 in PROTEUS*

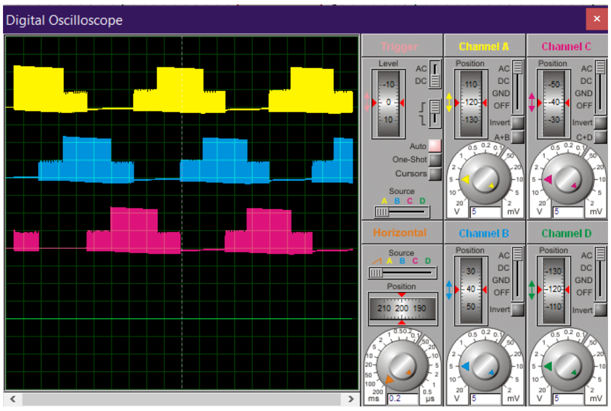


Figure 14: *Pulses generated viewed on an Oscilloscope*

6 Code

```
#define INC_SPEED       A0
#define DEC_SPEED       A1
#define PWM_MAX_DUTY    255
#define PWM_MIN_DUTY    50
#define PWM_START_DUTY  255

byte step_bldc = 0, motor_speed;
unsigned int i;
void setup() {
  DDRD  |= 0x38;          // Configure pins 3, 4 and 5 as outputs
  PORTD  = 0x00;
  DDRB  |= 0x0E;          // Configure pins 9, 10 and 11 as outputs
  PORTB  = 0x31;
  // Set clock source to clkI/O / 1
  TCCR1A = 0;
  TCCR1B = 0x01;
  // Set clock source to clkI/O / 1
  TCCR2A = 0;
  TCCR2B = 0x01;
  // Analog comparator setting
  ACSR   = 0x10;          // Disable and clear analog comparator interrupt
  pinMode(INC_SPEED, INPUT_PULLUP);
  pinMode(DEC_SPEED, INPUT_PULLUP);
}
// Analog comparator ISR
ISR (ANALOG_COMP_vect) {
  // BEMF debounce
  for(i = 0; i < 10; i++) {
    if(step_bldc & 1){
      if(!(ACSR & 0x20)) i -= 1;
    }
    else {
      if((ACSR & 0x20)) i -= 1;
    }
  }
  bldc_move();
  step_bldc++;
  step_bldc %= 6;
}
void bldc_move(){      // BLDC motor commutation function
```

```
  switch(step_bldc){
    case 0:
      AH_BL();
      BEMF_C_RISING();
      break;
    case 1:
      AH_CL();
      BEMF_B_FALLING();
      break;
    case 2:
      BH_CL();
      BEMF_A_RISING();
      break;
    case 3:
      BH_AL();
      BEMF_C_FALLING();
      break;
    case 4:
      CH_AL();
      BEMF_B_RISING();
      break;
    case 5:
      CH_BL();
      BEMF_A_FALLING();
      break;
  }
}

void loop() {
  delay(100);
  SET_PWM_DUTY(PWM_START_DUTY); // Starting PWM with duty cycle =
      PWM_START_DUTY
  i = 5000;
  // Motor start
  while(i > 100){
    delayMicroseconds(i);
    bldc_move();
    step_bldc++;
    step_bldc %= 6;
    i = i - 20;
  }
  motor_speed = PWM_START_DUTY;
  //ACSR |= 0x08;                 // Enable analog comparator interrupt
  while(1) {
    while(!(digitalRead(INC_SPEED)) && motor_speed < PWM_MAX_DUTY){
      motor_speed++;
      SET_PWM_DUTY(motor_speed);
      delay(100);
    }
    while(!(digitalRead(DEC_SPEED)) && motor_speed > PWM_MIN_DUTY){
      motor_speed--;
      SET_PWM_DUTY(motor_speed);
      delay(100);
    }
  }
}

void BEMF_A_RISING(){
  ADCSRB = (0 << ACME);  // Select AIN1 as comparator negative input
  ACSR |= 0x03;          // Set interrupt on rising edge
}
void BEMF_A_FALLING(){
  ADCSRB = (0 << ACME);  // Select AIN1 as comparator negative input
  ACSR &= ~0x01;         // Set interrupt on falling edge
}
void BEMF_B_RISING(){
  ADCSRA = (0 << ADEN); // Disable the ADC module
  ADCSRB = (1 << ACME);
  ADMUX = 2;            // Select analog channel 2 as comparator negative input
  ACSR |= 0x03;
}
void BEMF_B_FALLING(){
  ADCSRA = (0 << ADEN); // Disable the ADC module
  ADCSRB = (1 << ACME);
  ADMUX = 2;            // Select analog channel 2 as comparator negative input
  ACSR &= ~0x01;
}
void BEMF_C_RISING(){
  ADCSRA = (0 << ADEN); // Disable the ADC module
  ADCSRB = (1 << ACME);
  ADMUX = 3;            // Select analog channel 3 as comparator negative input
  ACSR |= 0x03;
}
void BEMF_C_FALLING(){
  ADCSRA = (0 << ADEN); // Disable the ADC module
  ADCSRB = (1 << ACME);
  ADMUX = 3;            // Select analog channel 3 as comparator negative input
  ACSR &= ~0x01;
}

void AH_BL(){
  PORTB  =  0x04;
  PORTD &= ~0x18;
  PORTD |= 0x20;
  TCCR1A = 0;          // Turn pin 11 (OC2A) PWM ON (pin 9 & pin 10 OFF)
  TCCR2A = 0x81;
}
void AH_CL(){
  PORTB =  0x02;
  PORTD &= ~0x18;
  PORTD |= 0x20;
  TCCR1A = 0;          // Turn pin 11 (OC2A) PWM ON (pin 9 & pin 10 OFF)
  TCCR2A = 0x81;
}
void BH_CL(){
  PORTB =  0x02;
  PORTD &= ~0x28;
  PORTD |= 0x10;
  TCCR2A = 0;          // Turn pin 10 (OC1B) PWM ON (pin 9 & pin 11 OFF)
  TCCR1A = 0x21;
}
void BH_AL(){
  PORTB =  0x08;
  PORTD &= ~0x28;
  PORTD |= 0x10;
  TCCR2A = 0;          // Turn pin 10 (OC1B) PWM ON (pin 9 & pin 11 OFF)
  TCCR1A = 0x21;
}
void CH_AL(){
  PORTB =  0x08;
  PORTD &= ~0x30;
  PORTD |= 0x08;
  TCCR2A = 0;          // Turn pin 9 (OC1A) PWM ON (pin 10 & pin 11 OFF)
  TCCR1A = 0x81;
}

void SET_PWM_DUTY(byte duty){
  if(duty < PWM_MIN_DUTY){
    duty = PWM_MIN_DUTY;
    }
  if(duty > PWM_MAX_DUTY){
    duty = PWM_MAX_DUTY;
    }
  OCR1A = duty;                // Set pin 9 PWM duty cycle
  OCR1B = duty;                // Set pin 10 PWM duty cycle
  OCR2A = duty;                // Set pin 11 PWM duty cycle
}
```

Bibliography

[1] Usman, A., Joshi, B. M., & Rajpurohit, B. S. (2016). A review of modeling, analysis and control methods of Brushless DCMotors. 2016 International Conference on Computation of Power, Energy Information and Communication (ICCPEIC). doi:10.1109/iccpeic.2016.7557254

[2] B. Tibor, V. Fedak and F. Durovsky, "Modeling and simulation of the BLDC motor in MATLAB GUI," Industrial Electronics (ISiE). 2011 IEEE International Symposium on, Gdansk, 2011, pp. 1403-1407.

[3] A. Sathyan, N. Milivojevic, Y. J. Lee, M. Krishnamurthy and A. Emadi, "An FPGA-Based Novel Digital PWM Control Scheme for BLDC Motor Drives," in iEEE Transactions on Industrial Electronics, vol. 56, no. 8, pp. 3040-3049, Aug. 2009

[4] Derammelaere, S., Haemers, M., Viaene, J.D., Verbelen, F., & Stockman, K. (2016). A quantitative comparison between BLDC, PMSM, brushed DC and stepping motor technologies. 2016 19th International Conference on Electrical Machines and Systems (ICEMS), 1-5.

[5] Gamazo-Real, J. C., Vázquez-Sánchez, E., & Gómez-Gil, J. (2010). Position and speed control of brushless DC motors using sensorless techniques and application trends. Sensors (Basel, Switzerland), 10(7), 6901–6947.

[6] Brushless DC Motor Control using the LPC2141 Application Note; AN10661, NXP Semiconductors: Eindhoven, the Netherlands, October 2007.

[7] Becerra, R.C.; Jahns, T.M.; Ehsani, M. Four-Quadrant Sensorless Brushless ECM Drive. In Proceedings of the Sixth Annual Applied Power Electronics Conference and Exposition (APEC 1991), Palm Springs, CA, USA, March 1991; pp. 202-209.

[8] Yeo, H.G.; Hong, C.S.; Yoo, J.Y.; Jang, H.G.; Bae, Y.D.; Park, Y.S. Sensorless Drive for Interior Permanent Magnet Brushless DC Motors. In Proceedings of the IEEE International Electric Machines and Drives Conference Record, Milwaukee, WI, USA, May 1997; pp. TD1/3.1-TD1/3.3.