VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Xinwei Fang

Three-phases Brushless DC Motor Control

Technology and Communication

2012

FOREWARD

This thesis has been written for VAMK University of Applied Sciences. All the components were purchased by VAMK. The work space and testing place were located in the Technobotina Research Centre, VAASA, Finland.

There were plenty of people who supported and helped me during this long period of time. First, I would like to thank my thesis supervisor Dr Liu Yang, Lecturer in VAMK, who gave me strength and advice to finish my project, Laboratory Engineer, Jani Ahvonen, who gave me significant help when I was designing and making the driver PCB, and senior lecture of VAMK Jukka Matila and Santiago Chavez Vega. Special thanks to my families and friends, they also gave me considerable support and help.

Vaasa, Finland; 04,May,2012

Fang Xinwei

ABSTRACT

Author	Fang Xinwei
Title	Three-phase Brushless DC Motor Control
Year	2012
Language	English
Pages	62+4 Appendices
Name of Supervisor	Dr Liu Yang

With the increasing demand of using electric power and unpleasant of using brush motor; the usage of Three-phase Brushless Motor has been significantly developed in industry. However, with market-standard motor driver, there are many inconveniences for small institutions or companies in either application developments or education purposes. For instance, the size and the price of driver. Therefore, the object of this thesis was design modular sections (Software program, PCB driver and PCB connectors) to drive a brushless DC motor. The entire works are divided into four steps. 1) Pre-study, is for designing a circuit schematic and according to that to select the relevant components. 2) PCB design is for the hardware driver PCB design. This makes a flexible for size of hardware driver board. 3) Software programming, in this case, is based on M16C62P microcontroller in embedded C programming. The software control could also be done by FPGA or other types of microcontrollers as long as the minimum requirements are fulfilled. This makes a software control function flexible. 4) System testing tests system functionality and stability as well as PCB board and Software programming functionalities. With all work, the motor could finally be driven by microcontroller and hardware drivers with high level currents. The various possibilities can also be chosen to drive the Three-phase Brushless DC Motor with users demand and cheap price.

Keyword: Brushless DC Motor, Microcontroller, Motor Control.

CONTENTS

1 INTRODUCTION 9
1.1 Pre-study 9
1.2 PCB 10
1.3 Software Programming 10
1.4 System testing 10
1.5 Thesis overview 11
2 PRE-STUDY 12
2.1 Motor & Motor Parameters 12
2.2 P & N MOSFET 14
2.2.1 N channel MOSFET: 14
2.2.2 P channel MOSFET: 15
2.2.3 MOSFET circuit 15
2.3 MOSFET Driver 17
3 PRINT-CIRCUIT-BOARD 19
3.1 Motor Driver PCB 20
3.1.1 Prototype One 22
3.1.2 Prototype Two 23
3.1.3 Prototype Three 23
3.2 Connector PCB 24
3.2.1 Prototype One 24
3.2.2 Prototype Two 24
3.2.3 Prototype Three 24
4 MICROCONTROLLER PROGRAMMING 27
4.1 Program version I 27
4.1.1 PWM output (Software version I) 28
4.1.2 Adjusting frequency (version I) 34
4.2 Phase output analysis 37
4.3 Software Programming (Version II) 40
4.4 Reading Hall Sensor 41

4.5 PWM Wave Generation 43
5 SYSTEM TESTING 45
5.1 PCB Testing 45
5.2 Driver PCB Testing 45
5.2.1 Pull up MOSFET driver 53
5.2.2 P & N channel MOSFET 57
5.3 Motor Testing 62
5.3.1 Hall Sensor connection 62
5.3.2 Wave form 63
5.4 Analysis 63
6 CONCLUSION 65

LIST OF FIGURE AND TABLES

Figure 1. Thesis overview 11
Figure 2. Maxon™ EC 45 Flat datasheet [6] 12
Figure 3. NXP application note 13
Figure 4. Complementary configuration from the Skuba Robot team 16
Figure 5. MOSFET driver datasheet [3] 17
Figure 6. One phase circuit schematic 20
Figure 7. Pads logic three-phase schematic 21
Figure 8. Pads layout schematic 22
Figure 9. Pads logic and layout schematic for connector PCB 25
Figure 10. Frequency calculation 28
Figure 11. Relation between value of m and frequency 29
Figure 12. Sample code for PWM generation 31
Figure 13. Sample code for PWM update 32
Figure 14. Oscilloscope captured from MCU output 33
Figure 15. Oscilloscope captured from MCU output 34
Figure 16. Sample code for frequency update 35
Figure 17. Sample code for event counter 37
Figure 18.Maxon™ datasheet [8] 38
Figure 19. Relationship between hall sensor and output 39
Figure 20. Sample code for hall sensor feedback 41
Figure 21. Sample code for interrupt setting 42
Figure 22. Sample code for timer setting 43
Figure 23. Sample code for phase output 44
Figure 24. Oscilloscope captured from Maxon™ driver output 46
Figure 25. Oscilloscope captured from Maxon™ driver output 47
Figure 26. Oscilloscope captured from Maxon™ driver output 48
Figure 27. Oscilloscope captured from PCB driver output 49
Figure 28. Oscilloscope captured from PCB driver output 50

Figure 29. Oscilloscope captured from PCB driver output 50
Figure 30. Photos for power connection 51
Figure 31. Photos for ground connection 52
Figure 32. Oscilloscope captured from interference issue 53
Figure 33. MOSFET driver datasheet [3] 54
Figure 34. Pull- up MOSFET driver simulation 55
Figure 35. MOSFET pairs simulation 56
Figure 36. Short circuit scenario simulation 58
Figure 37. MOSFET driver datasheet [3] 59
Figure 38. Oscilloscope captured from PCB driver 60
Figure 39. Oscilloscope captured from PCB driver 61

LIST OF ABBREVIATIONS

DC: Direct Current

FPGA: Field-Programmable Gate Array

PCB : Printed Circuit Board

PWM: Pulse-width modulation

LCD: Liquid Crystal Display

BLDC: Brushless Direct Current

EC: Electronically Commutated

MCU: Microcontroller Unit

MOSFET: Metal–Oxide–Semiconductor Field-Effect Transistor

SMD: Surface-Mount Device

FPC: Flat Panel Cables

DEC: Digital EC Controller

RPM: Revolutions per minute

1 INTRODUCTION

As the short life time of brush motor working in high voltage conditions and the increasing demands of high voltage electrical motor, using microcontroller to control the brushless motor has been embedded into our every-day-use applications. In our robot team, the latest generation robot used brushless motor as the main power source to achieve the possibilities of fast moving, spinning around, kicking and dribbling. However, those industrialized motor drivers are not only hard to be adjusting for the single purpose of using but also limited the control current at the lower end. Moreover, it is also overly expensive for the small companies or education institutions to integrate it in the applications. Thus, the objective of this thesis is that using a self-designed circuit and microcontroller to control the Three-phase Brushless DC Motor. As it has been done, the multiple features could be added on this system as developer requiring. It will cost much less than ready-made system, and will be more flexible for being integrated in other complicated systems and will be working properly in high current condition.

The entire thesis for system controlling Three-phase Brushless DC Motor could be divided into four steps: 1) Pre-studying; 2) PCB designing and making; 3) Microcontroller Programming and 4) System Testing.

1.1 Pre-study

At the very first beginning, only the type of motor and microcontroller are fixed. They are a) Motor – Maxon™ EC 45 Flat DC motor (30W) and b) Microcontroller - Renesas™ M16C62P 16 bits microcontroller. The MOSFETs and the MOSFET drivers are selected according to the specifications of motor, which are specified by the current, voltage and other restrict parameters as well as the circuit schematic.

1.2 PCB

In the PCB(Printed Circuit Board) designing part, the two PCB boards are designed and made based on the previous robotic motor power schematic of the VAMK and the knowledge that was gained from the pre-study. One board is for supplying the current and voltage to the phase of motor inputs, which not only has power and ground connections, but also has 6-pin PWM connections form external Renesas™ MCU. Another one is a connector, in among of the driver board, brushless motor and MCU, which is connected among the driver board, the motor and the MCU.

1.3 Microcontroller Programming

The software programming is based on Embedded C programming, programmed for the Reneasa M16C62P microcontroller in the High-performance Embedded Workshop (HEW) environment

1.4 System testing

The system testing include the Programming bug fixing, the connection and various supplies (Current, Voltage and PWM inputs and outputs) checking and validation and stability evaluations. This session will be gone through the entire thesis process. For example, the driver circuit is tested after the components has been selected; with the simulated PWM signal from function generator, the stability of the circuit is checked after the PCB was built and the functions output is also checked from microcontroller after the programming was finished by using oscilloscope. Finally, when every individual sections are working fine, they are combined together to have final testing.

1.5 **Thesis overview**

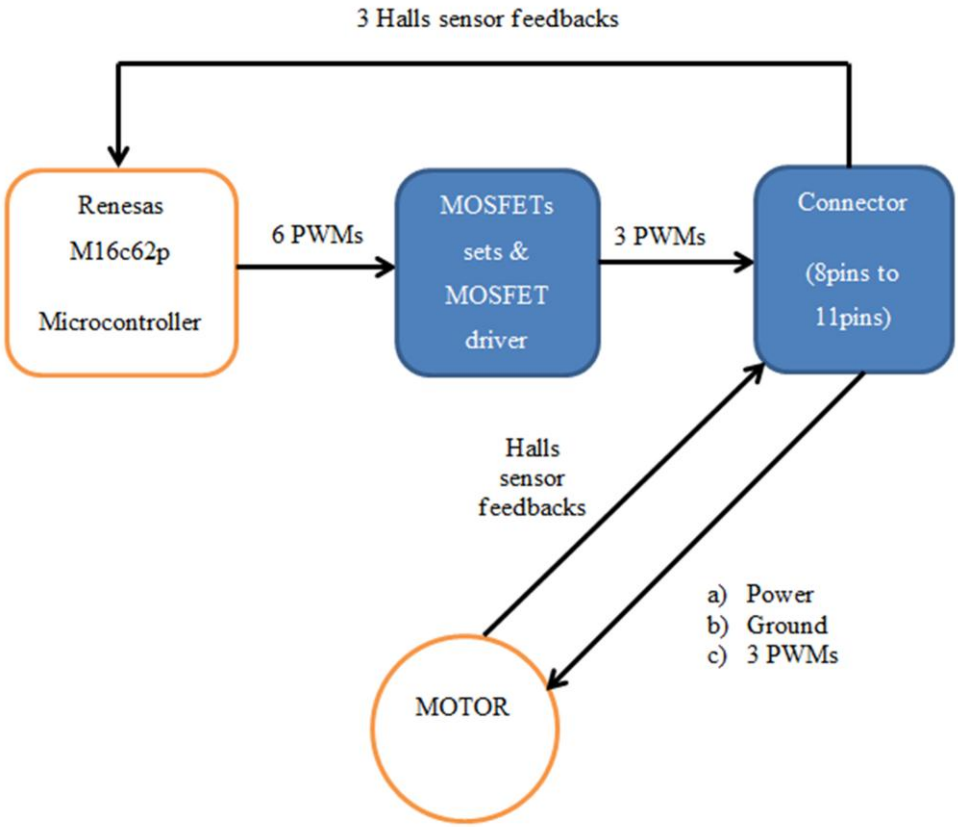


Figure 1. Thesis overview

Figure 1 shows the outline of the thesis project. The blocks with blue colour background will be the PCB that going to be designed and made in this project, and selection and testing of component is also a part of work. The blocks with orange colour outline include components that has already been selected and ready-to-use. The black colour arrays indicate what and how signals are passed by the each functional block. There is one thing should be noticed that the connection for PWM signal between connector and motor are using two wires for each pin because there might have very high current passing through the wires when driving the motor. Therefore, on the Figure 1, there are 6 pins occupied by 3 PWM signals.

2 PRE-STUDY

In pre-study, the first discussion would be how the circuit diagram will be and how the components could be selected based on datasheet and circuit diagrams. At the beginning only the motor type and microcontroller were selected. In this section, focus is mainly on how hardware part (PCB) could work properly.

2.1 Motor & Motor Parameters

The Maxon™ EC 45 Flat DC motor (30W) is used as the motor source. According to the datasheet, there are some critical parameters that should be noticed (seeing Figure 2.).

			with Hall sensors	200142
			sensorless	
Motor Data				
	Values at nominal voltage			
1	Nominal voltage	V		12.0
2	No load speed	rpm		4370
3	No load current	mA		151
4	Nominal speed	rpm		2860
5	Nominal torque (max. continuous torque)	mNm		59.0
6	Nominal current (max. continuous current)	A		2.14

Figure 2. Maxon™ EC 45 Flat datasheet [6]

A permanent magnet rotor and three-phase stator winding are embedded in the Brushless Direct current (BLDC) motor. As it is named, there are no brushes getting in use for commutation; instead, Hall sensors are used for detecting a rotor position and commutating are done by sensor inputs.

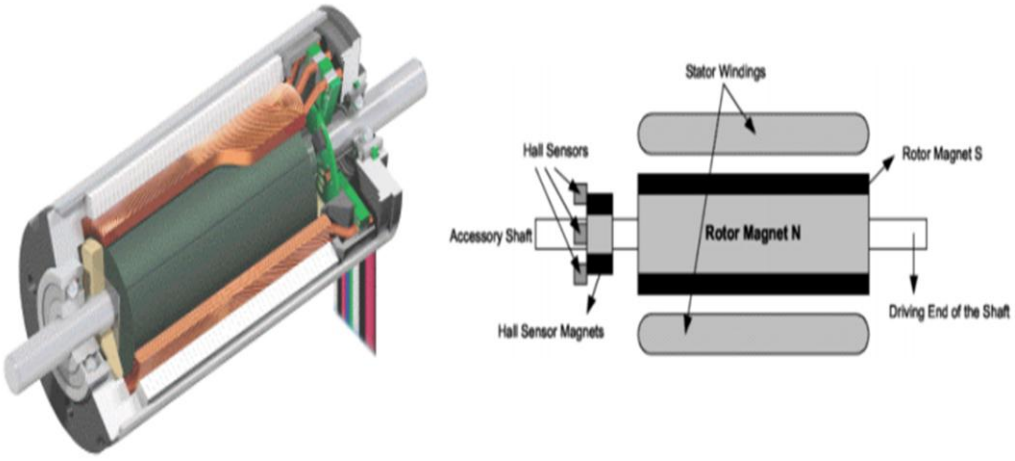


Figure 3. NXP application note

Figure 3. shows how it looks like inside of BLDC motor. In the Application Note from NXP, Maxon™ EC 40 DC motor (120W) was in use. The ARM 7 based MCU (Microcontroller Unit) was used to control the motor, which got more powerful functionalities than M16C62P microcontroller but also leaded a cost issue. The total cost of the project could be less than one ARM based Microcontroller.

Back to the design, at beginning, PWM signals are planned to input directly to drive the motor, however, the PWM output from the MCU did not content any current flow, which makes problem that each winding of motor could not get any current to magnetize following phase. Without a magnetized phase, the rotor will not rotate. Hence, it is necessary to have a driver board supplying the current to magnetize the motor phases. How the MOSFETs are selected and drive board is designed and worked will be discussed in following chapters.

2.2 P & N MOSFET

In the project, the MOSFET are used because high current and voltage of motor need to be switched by lower voltage. The microcontroller outputs 4.7V to 5V voltage as a high level for a PWM wave form and it does not have any current to supply the motor phase directly. Therefore, the MOSFETs are needed to solve this problem and supply the current.

The MOSFET could be thought of a good way of solving this. It changes resistance between drain (D) and source (S) pins with functional relations of voltage difference between gate (G) and source (S). The drain (D) and source (S) pin's resistance can be very high (almost like open circuit and no current flow) when there is no voltage difference between pin gate (G) and source (S). On the other hand, when there is a voltage difference between gate (G) and source (S) pins, the resistance will reduce functionally and current will flow through the source (S) and drain (D) pins. If the voltage is very high but under the limitation of MOSFET, the resistance will be a minimum. **[7]**

2.2.1 N channel MOSFET:

N channel MOSFET from International Rectifier ™ (IR) is selected in use. Model name IRF7413PbF which has $V_{DSS} = 30V$ and $R_{DS} = 0.011\Omega$ (when the current fully flow) [5]. For N channel MOSFET, the source (S) pin is connected to the ground. By changing voltage input to the gate (G) pin, the current flow could be adjusted. If the voltage in the gate (G) pin is increasing and higher than the source (S) pin (0V, because is connected to the ground), the current flow from source (S) to drain (D) will increase with voltage rising.

2.2.2 P channel MOSFET:

P channel MOSFET is also from International Rectifier ™ (IR). Model name IRF7424PbF which has $V_{DSS} = -30V$ and $R_{DS} = 0.0135\Omega / R_{DS} = 0.022\Omega$ (when $V_{GS} = -10V / V_{GS} = -4.5V$) [4]. The connection of P channel MOSFET is similar with N channel MOSFET, however, the difference is the source (S) pin is connected to the power instead of a ground. Thus, in order to make a current flow, the gate (G) voltage needs to be reduced when considering the source (S) pin always have a high voltage. In this case, the voltage different between gate (G) and source (S) pin will be made by decreasing the voltage of gate (G) pin which differ from the N channel MOSFET.

2.2.3 MOSFET circuit

Both P channel and N channel MOSFET make a similar functional work that receiving voltage signal (PWM waves) from gate (G) pin compare the gate (G) pin's voltage with and source (S) pin's, and decide how much resistance will be between the drain (D) pin and source (S) pin and how much current flow will go through those 2 pins. If the drain (D) pin is connected to the load, the current through the load will be depended on the voltage difference between gate (G) pin and source (S) pin. Moreover, if using PWM as input of gate (G) pin meaning that the PWM wave form (duty cycle, frequency etc.) can be controlled by the MCU and the motor adjustment can be done by only setting a parameter of MCU to modify the output PWM.

There are plenty applications guidance of Three-phase Brushless Motor mentioned how to use MOSFET circuit to supply current to the motor. For instance, in the Skuba team (A robot team in Kasetsart University, Thailand [2]) description, they were using 2 MOSFETs pairs (one P channel and one N channel) to driver one phase of motor. They built the 3 pairs of MOSFET pairs to drive a 3 phases brushless motor (Figure 4.).

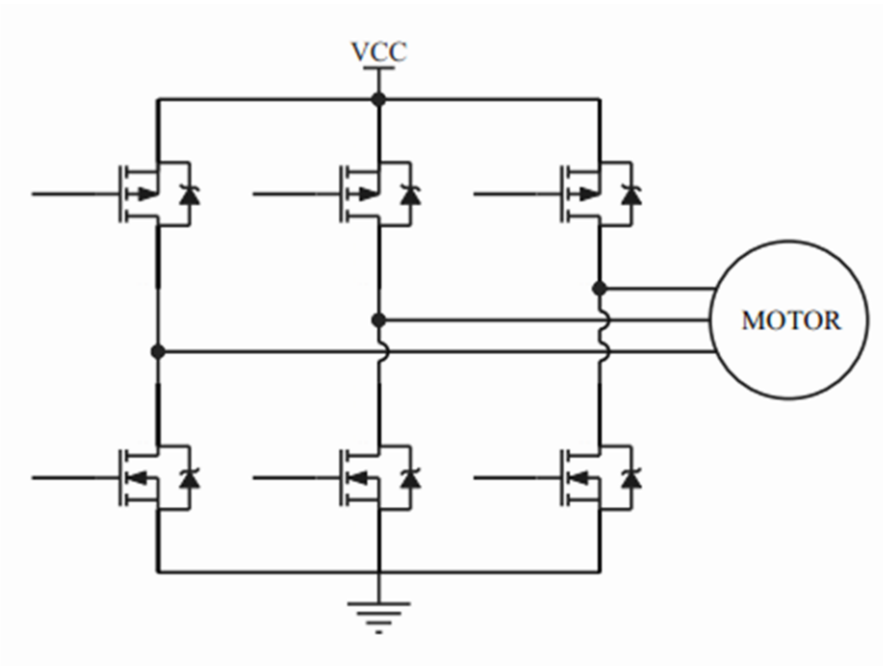


Figure 4. Complementary configuration from the Skuba Robot team

In Figure 4 indicates that the P channel MOSFET source pin (S) is connected to the power and drain (D) pins is connected to the motor while the gate pins (G) receiving a signal from external sources; similarly, the N channel MOSFET have a same connections with both drain (D) and gate (G) pins, and the difference is that the source (S) pin is connected to the ground. This connection is identical as what has been analysed above and it can make a current flow to drive each phase, however, with the proceeding, it results that the M16C62P microcontroller could not drive those MOSFET pairs directly. The reason is turned out that normal MOSFETs need high level voltage over 5V to make a voltage difference between pin source (S) and gate (G) to control the current flow and even through the 5V can drive the motor the high level voltage may various for MOSFET pairs and microcontroller output. This means that the P channel MOSFET will be never completely closed and made the current flow all the time whenever N channel MOSFET is closed or opened. Therefore, using microcontroller to control

MOSFET pair directly will NOT work. Some components should be added on the system for not only pulling-up voltage but also making a standard high level reference voltage to driver the MOSFETs pairs.

2.3 MOSFET Driver

As the problem that has been mentioned above, the extra components need to be used for pulling up the voltage of microcontroller PWM output and making the reference high level voltage. The MOSFET driver from MICROCHIP ™ is found out that can fulfil all the requirements. It is with model name TC4428AOCA that can receive 2 PWM signals from the microcontroller and pull the voltage up to the voltage that as higher as in the pin V_{DD}. Because the $V_{DD}$is connected the same power supply of P channel MOSFET, hence there will be the same high level voltage and only change when power supply changes.

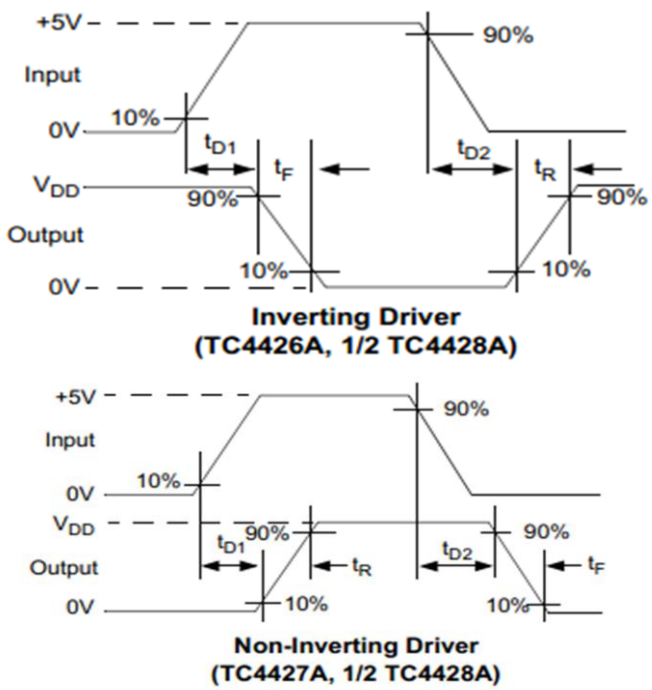


Figure 5. MOSFET driver datasheet [3]

As can be seen from the Figure 5, the MOSFET driver has two types, Inverting Driver and Non-Inverting Driver. The differences between those are Inverting Driver added a logic invertor gate at the output port and made an inversed output wave. Fortunately, the pull-up time delay under the tolerance, according to the examples of datasheet, which t_R (Time for rising) and t_F (Time for falling) are less than 10ns if input function signal is 100KHZ square wave. In addition, as the absolute maximum supply voltage of this MOSFET driver is 22V, the high level voltage can be pulled up to the range from 5V to 22V and it can drive the three pairs of MOSFET without any theoretical problem. After this, the making and design of PCB will be based on those components.

As having gone through the pre-study, the majority important components are selected. For each one phase of circuit, they have one MOSFET driver, one P channel and N channel MOSFET. With those components contribution, the PWM signal could carry with current flow to drive the motor. After this, the main work is shifted to the PCB board design and making.

3 PRINT-CIRCUIT-BOARD

In PCB part, all the designs are based on the out school's robotic schematic (some changes are applied during the process). PADs logic™ is used for circuit design and PADS layout™ is for physical component pin design according to the datasheet.

The breadboard is planned to use in building circuit, which would be cheaper, faster and easier. But after being checked all package descriptions of those three components, some of them do not have an In-line package. The P and N MOSFET only have a SMD (on the top of the surface) SO-8 packages. To avoid double layer of PCB in the driver board, the SOIC-8 in the MOSFET driver is chosen even through the In-line package is available. Therefore, in the driver board parts, except a $470\mu\Omega$ capacitor, all components are soldered on the surface.

3.1 Motor Driver PCB

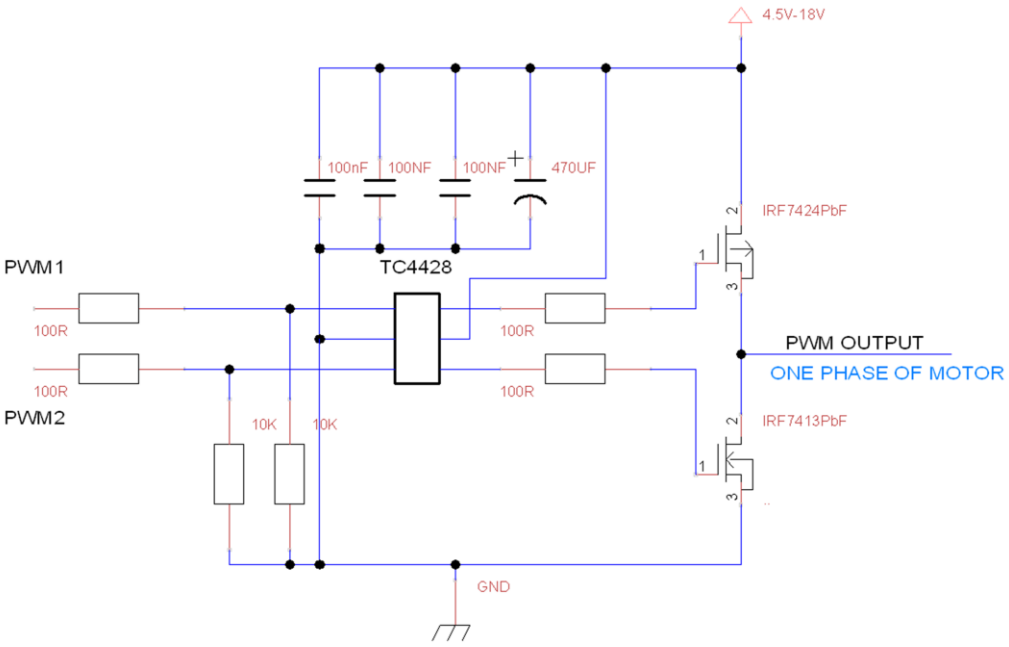


Figure 6. One phase circuit schematic

The Figure 6 shows the schematic of one phase driver circuit. In the driver board, three-phase of circuits are connected together to share the same power line and ground line. The capacitors are used for avoiding external interfering from outside of circuit. The 470uF capacitor can help the power line stable. The schematic of PCB and layout of PCB for three phase driver will be shown in the following.

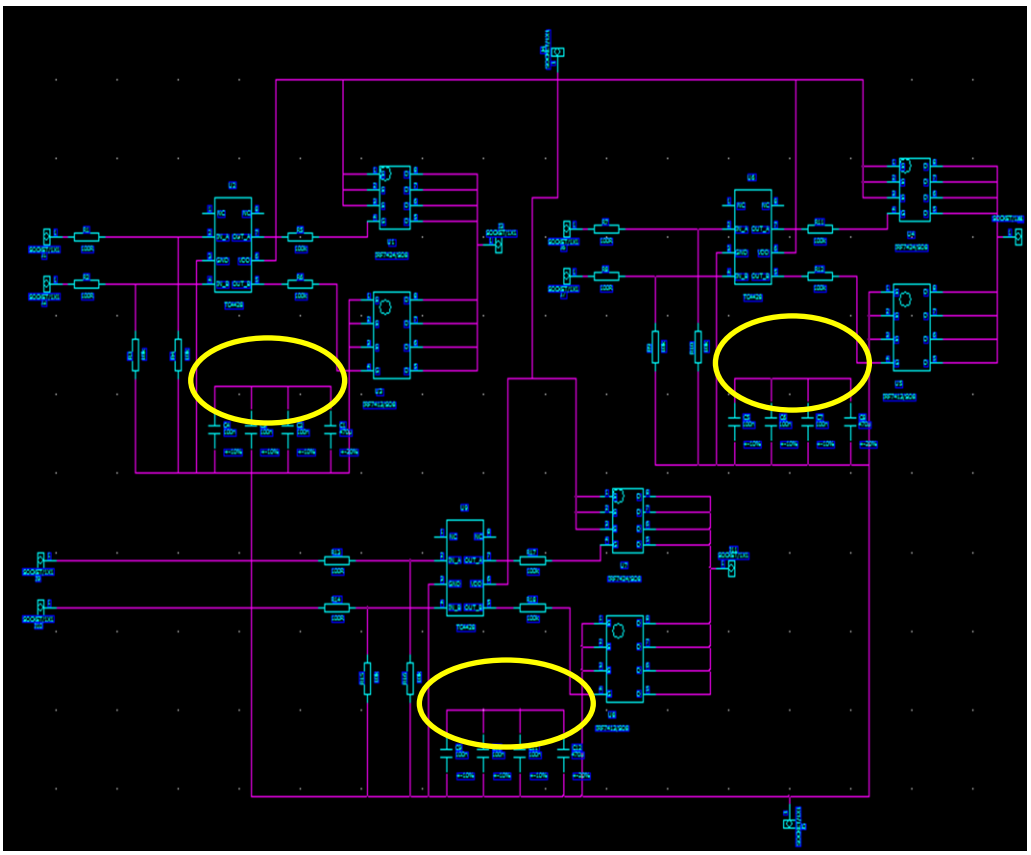


Figure 7. Pads logic three-phase schematic

Figure 7 indicates that combination of three pairs of motor driver circuit schematics. As it is shown, the power line and ground line are used in among those three input circuits. The following Figure 8 illustrates the layout of three phase driver circuit.

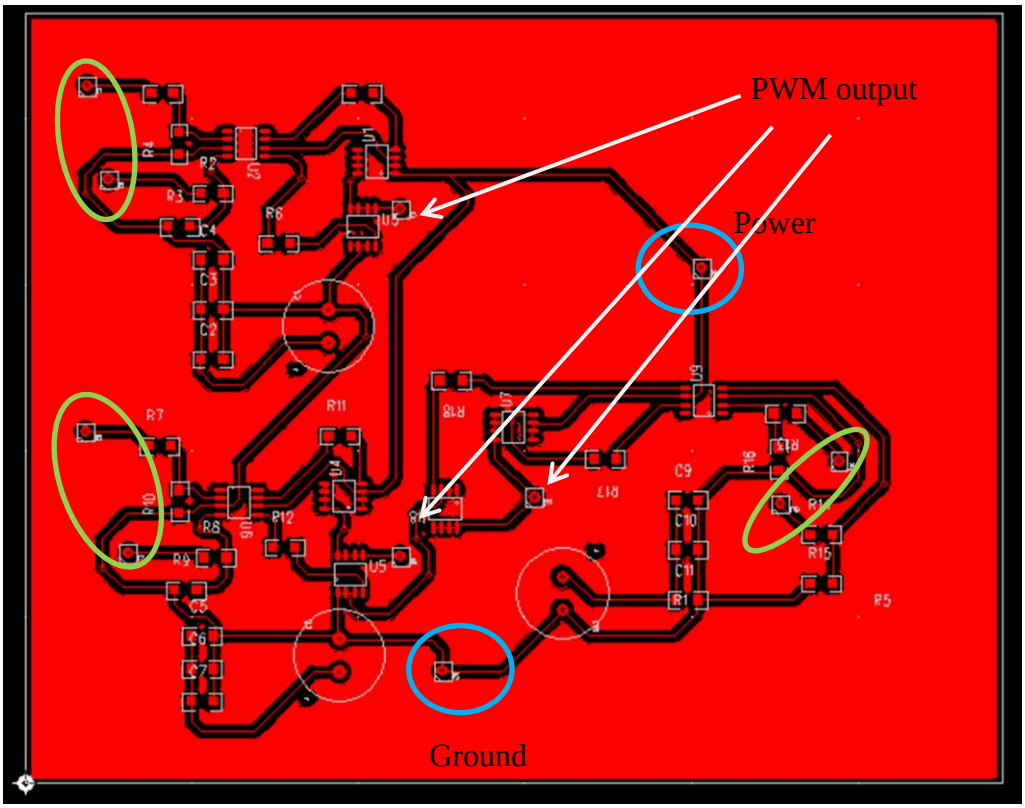


Figure 8. Pads layout schematic

Based on those schematic and Layout information, three prototypes are made in this driver PCB. But the first two has a serious drawback. The final version that used in testing is prototype 3.

3.1.1 Prototype One

It was the first PCB board. Therefore, there were some deadly mistakes made during the process. When using light to project the CAM on the copper board, less time was put in this procedure. Thus, the circuit image did not project on the copper board clearly even if the board was flushed for the long time. Second, flush time and temperature were not well controlled, which resulted in that some

traces had been flushed out and some parts still cover a red. After being polished, some areas on the board that cover red colour did not flow current and the some traces were even stripped from the board.

3.1.2 Prototype Two

With the experience that gained from the first miss. The projection time was set as 6 minutes, which work every well and the circuit clearly shown on the board. The board was placed in the flush liquid when the its temperature raised at 45 Celsius degree and made it to be flushed for 10 minutes until the all red cover was removed. This board was well flushed and polished. SMD resistors and capacitors were soldered properly. Then, the big problem occurred when MOSFETs were going to be soldered. The pin numbers were opposite because the face (Top side facing down) of transparent CAM picture was placed opposite. The instruction video was placed CAM picture facing down because it made for in-line packages PCB board that components were placed on the top and the copper and solder side was at bottom, that way of projecting CAM will lead the pins in the correct order in this project. Therefore, MSD PCB board should put the transparent CAM picture in other facing side (Top side facing up). Hence, prototype three was built.

3.1.3 Prototype Three

After missing two of board, this one was done properly. The multi-meter was used for testing all the connections. After that, the connection that missing from the protection capacitor set to the power line were checked out. In the Figure 7, the yellow circle pointed out where the missed power lines were. It was fixed by using a wire soldering between capacitor power point and the power line. The rest of connections were very well connected. In the Figure 8., the blue circle indicated the power point and the ground point, the yellow circle shown the 6 input PWM signal pins which 2 sets in a pair to drive one phase(two inputs signal

should be inversed) and the white arrays pointed to the 3 phase PWM signal output respectively.

3.2 Connector PCB

The circuit of connector is simpler than a driver part. However, still three of them have been made.

3.2.1 Prototype One

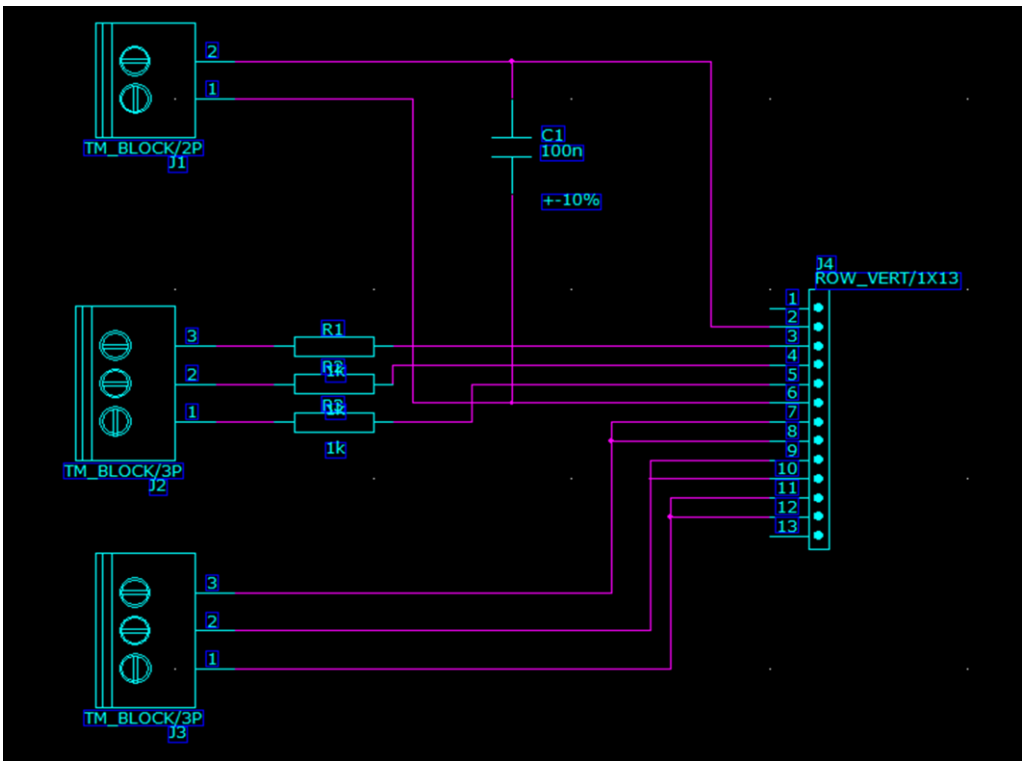
This one made at the same time as Prototype two of motor driver. Therefore, the same mistake was made that put wrong direction of transparency. There was a flat panel cables (FPC) connectors on the surface of board, which had a wrong pin connection. But the copper flush and polish was fine.

3.2.2 Prototype Two

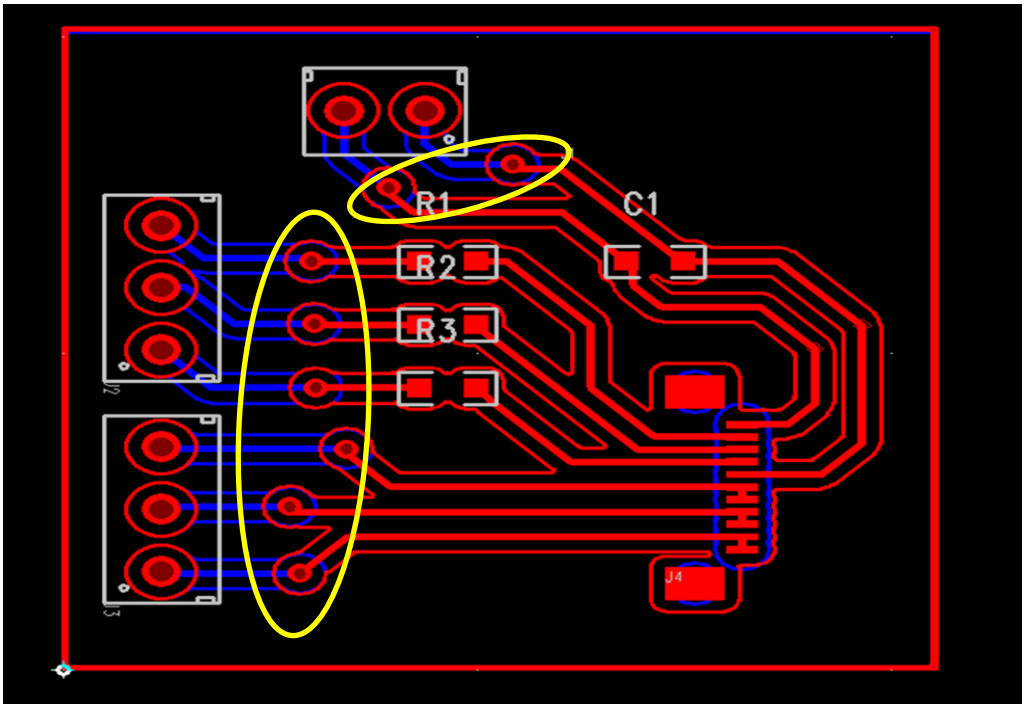
As being mentioned before, the double layer PCB was better to be voided because it will be harder to be made. Thus, even some connectors were very hard to be soldered on the front face, the single layer PCB still was chosen. This decision was proved very wrong when soldering those connectors. It was impossible to put a soldering tin and solder them.

3.2.3 Prototype Three

For this prototype, as the experience that gained from previous, the double layer PCB was made. A ‘via’ was made to connect two side of layer. When the circuit was projected on the copper, the both layers and pins should be at the same dimensional position. Hence, an extra “2 D outline” was added– a rectangle to distinguish a two side position. When the 2 rectangle on the separate transparency got superposition, it means that two layers will be at the right dimensional position. The Figure 9 will show how the connector’s schematic design **(a)** and PCB layout **(b)**.



(a)



(b)

Figure 9. Pads logic and layout schematic for connector PCB

From Figure 9 **(a)**, at the top, a two pins connector is for a Motor Power and ground; in the middle there is a three pins connector that is for three Hall sensors; at the bottom, there is another three pins connector that is connected to six pins of FPC connectors for motor winding. The reason that needs extra connection was the motor winding may carry high current when the motor is running, therefore, making two wires to carry high current to prevent the damage is necessary. From Figure 9 **(b)**, it has two colours that the blue one stands for a bottom layer and the red one is for top layer. The yellow circle indicates the 'via'. They need to be soldered at both sides of board so that the current will flow two faces. The components with red colour are SMD components which should be soldered on the top surface; and the one is connected with blue lines (Big connectors) are in-line packages, they should be soldered at bottom.

Three prototype of each PCB board is made because lack of experience of making PCB. Even the final version is not the perfect one, some connections are still missing and are connected with wires. However, the PCBs are passed testing and would be used in the final project.

4 MICROCONTROLLER PROGRAMMING

In software programming **version I**, the mainly focus is on how to generate the 120 degree phase difference PWM wave form and read the pulses from the external sources (Hall sensor). The programming work is analysed and separated as several modulations which are 1) PWM output, 2)Hall sensor receive, 3) calculate the PID value to re-adjust output and 4) display the important value (speed, RPM and etc.). However, after first time connected to the motor, it turns out that the program (**version I**) did NOT make a motor running even through it have similar function wave outputs. Section 4.1, are the old program, it can make a 'fake' 120 degree output but cannot drive the motor. In the following sections, the programs are analysed and re-designed. Comparing those two will let us know more about how brushless motor worked. In program version II, everything is re-designed the multiple timer inside of microcontroller are used for generate PWM waves according to the information that received from hall sensor. The programming can be done as three steps. First, make three INT interrupts to receive the hall sensor feedback. Second, declare the timer A1, 2 and 4 and timer B2, which set timer A as a one shot mode to shot pulses making PWM waves according to the timer B2 under flow. The PWM frequency will be depended on the counter speed of timer B2. At the last, use microcontroller pre-defined register (three phase motor control register) to output PWM waves to relevant ports with the right phase sequences.

4.1 Program version I

The program **version I** is designed based on the output wave from Maxon™ driver. Without deeply analysed how PWM wave derive the motor, the designed program is only output wave form with 120 degree PWM wave phase shift. It is just a copy of output from Maxon driver. When program has done, the output that

read from oscilloscope almost identical with Maxon™ driver. But in reality, it will not make motor running [8].

4.1.1 PWM output (Software version I)

In this section, first requirement is to output 50% (duty cycle) PWM wave. When talking about PWM that is generated from Renesas™ M16C62P is 8 bits PWM. Then, set the value of timer register to adjust its frequency. After that, using delay function to delay the each PWM making possible that three phase PWM has around 120 degree and under 10 degree tolerant. The last, using the ADC knob to set the value of frequency.

The difference between 16 bits PWM and 8 bits PWM is that 16 bits PWM is using entire 16 bit to generate the PWM meaning that the period time is fixed because there is no bits for frequency setting; however, the 8 bits, there is 8 bits set as PWM duty cycle and another 8 bits is to set the period time. Therefore, when using 8 bits PWM, there are several equations that should be noticed (seeing Figure 10.).

8 bits PWM (fi: the selected frequency f32, f8 and f1)

High level width:

$n \times (m + 1)/fi$ n: values set to timer Ai register's high-order address

Cycle time:

$(2^8 - 1) \times (m + 1)/fi$ m: values set to timer Ai register's low-order address

Figure 10. Frequency calculation

The clock frequency is 24MHZ and the f8 is selected which means that the

24MHZ is divided by 8 – 3MHZ. Both m and n were in range from 0 – 255.

$$\text{Duty Cycle} = \frac{High\ level\ width}{Cycle\ time} = \frac{\text{n} \times (\text{m} + 1)/\text{fi}}{(2^8 - 1) \times (\text{m} + 1)/\text{fi}} = \frac{n}{255} \quad (1)$$

In order to set half of duty, the value of n is calculated as 127.5. However, only an integer value could be stored in register. In this case, the value of n is fixed in **127**.

The frequency control is related with cycle time thus change the value of m can change the frequency.

$$\text{f} = \frac{1}{cycle\ time(period)} = \frac{1}{(2^8 - 1) \times (\text{m} + 1)/\text{f8}} = \frac{\text{f8}}{(2^8 - 1) \times (\text{m} + 1)} \quad (2)$$

Therefore the relation between value of m and frequency will be shown in Figure 11..

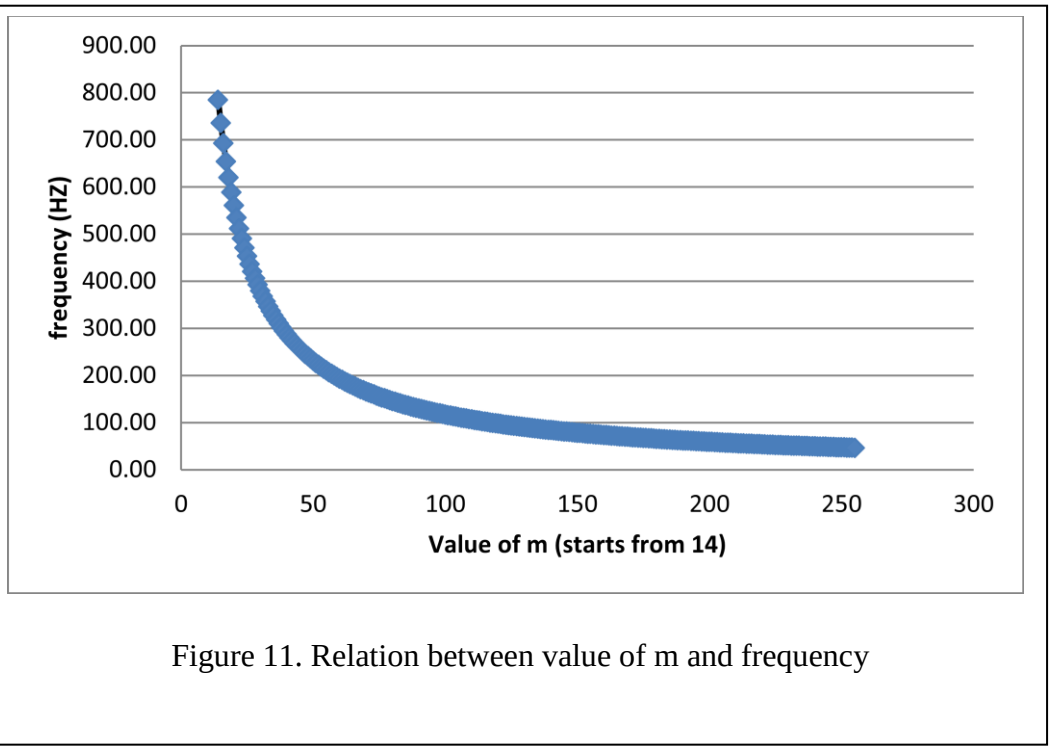


Figure 11. Relation between value of m and frequency

The Figure 11 indicates the relationship between value of m and frequency. The value of m starts from 14 because the values less than 14 lead the overly high frequency and correlations with value m are getting less. When value of m between 0 and 10, the frequency changes from 11764HZ to 1069HZ, which is 10 times difference. In comparison, when the value of m from 235 to 255, the frequency only changes in the middle from 40HZ to 50HZ. It means that the frequency are more easy to control when it is in the lower ends, and in the high end, the interval frequency changes will be getting large when value of m changes. Hence, in the programming, the values of m are selected from 14 to 255 meaning that the frequencies changes can be theoretically achieved from lower end 45.96HZ to higher end 785.31HZ.

Delay function is done by calculation. According to the result in pre-study, each phase has almost same frequency telling that the each PWM can have same value of n and m making same duty cycle and cycle time. Therefore, using another timer making an output and having a delay is possible. Thus, the 120 degree phase difference can be thought as one third of the PWM cycle time because each cycle time is 360 degree. The results of calculation that one third PWM are actually delay time of each PWM update. Delay is done by adding counter timer. Each round of interrupt timer has a certain time, if the counter matched the certain time will update the following PWM.

```
101
102         //timer a0  for A phase pwm
103         ta0mr=0x67;     //8bit
104         ta0=0x0000;     //initialise the timer
105         ta0s=1;         //timer start
106
107
108         //timer a1  for B phase pwm
109         ta1mr=0x67;     //8bit
110         ta1=0x0000;     //initialise the timer
111         ta1s=1;         //timer start
112
113
114         //timer a2  for C phase pwm
115         ta2mr=0x67;     //8bit
116         ta2=0x0000;     //initialise the timer
117         ta2s=1;         //timer start
118
119
120         //tiemr a3 for timer counter mode
121         ta3mr=0x40;     //normal count mode
122         ta3=29;         //0.01ms counter time
123         ta3ic=0x07;     //the highest periority
124         ta3s=0;         //not start at the beginning
```

```
158     void A_Phase_PWMUpdate(void){
159
160             ta0=NumN*256+NumberM;
161
162         }
163
164     void B_Phase_PWMUpdate(void){
165
166             ta1=NumN*256+NumberM;
167
168         }
169
170     void C_Phase_PWMUpdate(void){
171
172             ta2=NumN*256+NumberM;
173
174         }
175
```

Figure 12. Sample code for PWM generation

The Figure 12 shows how the PWM function timers are set. The timer a0, a1 and a2 are worked with PWM generation. It updates the PWM wave when they are called. The timer a3 is set as normal timer counter. It counts every 0.01ms because, when the PWM frequency achieved maximum (785.31HZ, m=14), the delay time is 0.42ms. If set 0.1ms as a counter, only 4 times are counted, and makes an inaccurate delay time calculation. The 0.01ms is the maximum counter speed that f8 counter source can achieve. f1 source is tested to counter faster and

result will lead system unstable. The timer a3 counter is only started between the previous PWM has updated and following one is going to update.

```
177
178   void timer_a3(void){
179
180       if(timerCounter<delay){
181           timerCounter++;
182       }
183       else{
184           indicator=1;
185           timerCounter=0;
186           ta3s=0;
187       }
188   }
189
```

```
206   void PWMupdate(void){
207
208   //update of Phase A
209       if (flag==0){
210           ta3s=1;
211           while(indicator!=1);
212           A_Phase_PWMUpdate();
213           flag=1;
214           indicator=0;
215       }
216
217   //update of Phase B
218       if (flag==1){
219           ta3s=1;
220           while(indicator!=1);
221           B_Phase_PWMUpdate();
222           flag=2;
223           indicator=0;
224       }
225
226   //update of Phase C
227       if(flag==2){
228           ta3s=1;
229           while(indicator!=1);
230           C_Phase_PWMUpdate();
231           flag=0;
232           indicator=0;
233       }
234
235   }
236
```

Figure 13. Sample code for PWM update

In Figure 13 function ‘PWMupdate’ and timer a3 are worked for generating a phase delay. The timer a3 is set as interrupt timer counter. Global variable flag and indicator are used for making sure each updates are in the correct order. When the program starts, the pre-saved value (delay time, M and N values) are used first. Flag with number of 0,1and 2 stands for following update PWM should be phase A, B and C respectively. When the previous PWM has up-to-date, the flag will

change to the next in order to update phase in sequence. The value of indicator is for making sure that the delay counter has been done properly. Flag will not be set before the indicators are set. A while loop was waiting for value of indicator is set and it will only be set when the interrupt timer has finished counting. When it finished counting, sets all the value as initialization and wait for next phase function call.

The results of output PWM function were shown below.

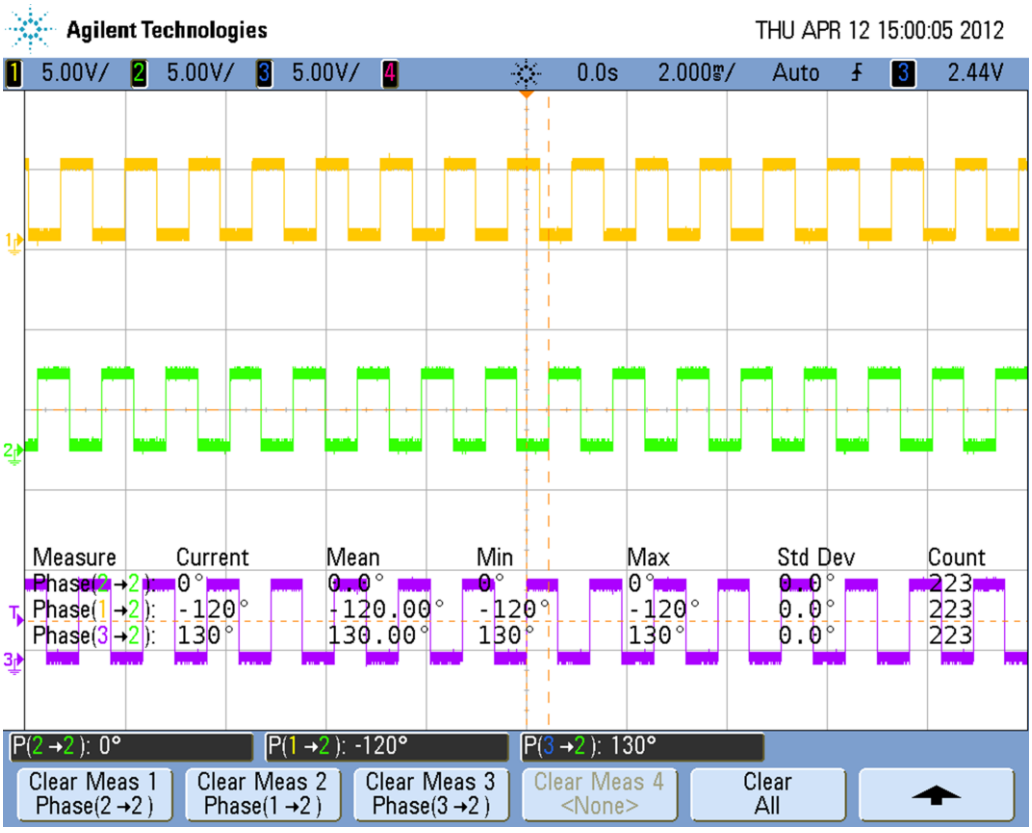


Figure 14. Oscilloscope captured from MCU output

The Figure 14 shows the phase difference from MCU when the m is set as 14 (the highest). The phase difference is under the tolerant.

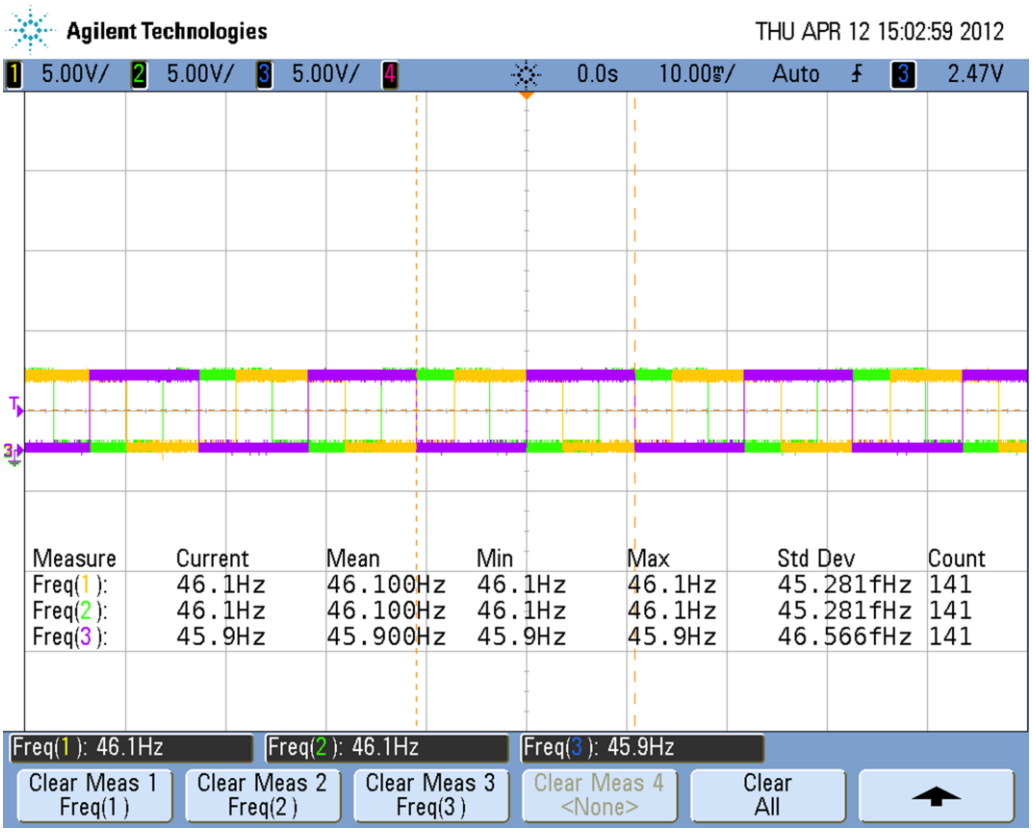


Figure 15. Oscilloscope captured from MCU output

The Figure 15 shows when the m is set as minimum value (255). The frequency are almost identical and it is also has 120 degree phase difference. The amplitudes are at the same level.

4.1.2 Adjusting frequency (version I)

As the output of Maxon™ DEC 24/3 (Digital EC Controller), the speed of motor will change according to the input frequency. The frequency could be changed by adjusting the value of m. Thus, to update the value of m in real time will make a possible to adjust the speed of motor. The ADC knob is used for changing the value of m.

```
142
143         // ADC settings
144         adcon0=0x08;          //repeat_mode software_trigger ad_disable
145         adcon1=0x28;           //10 bit ADC were selected
146         adcon2=0x00;           //p10 is selected
147         pd10 = 0;
148         adcon0 |=(1<<6);      //start AD conversion
149
150
151         //switch settings
152         ifsr=0x07;            //INT 0,1,2  both edges
153         int0ic=6;
154
```

a

```
260     void ADC_update(void){
261
262         temp=0;                       //vaule reset
263         temp2=0;                      //value reset
264
265         temp=ad0/10*255;              //temp value for number M calculation
266         temp2=temp/1023*10;           //temp value for number M calculation
267         NumberM=255-temp2;            //value increasing with clockwise
268
269
270         // the maximum value of m will be set as 14
271         if(NumberM<14){
272             NumberM=14;
273         }
274     }
```

b

```
194
195     void int0_int(void) {
196
197         tb0s=0;                         //close the other interrupt
198         ta0=ta1=ta2=0;                  //reset all the PWM output
199         ADC_update();                   //call the function get number of NEW m
200         period = (255*(NumberM+1));     //calculate delay
201         delay =period/(FI*3);           //calculate delay
202         timerCounter=0;                 //reset timer
203         flag=0;                         //reset flag
204         indicator=0;                    //reset indicator
205
206
207     }
```

c

Figure 16. Sample code for frequency update

As being shown in Figure 16**,** in (a), 10 bit ADC knob is used to correlate with value of M. and update with switch interrupt. The priority is set less than timer a3 counter meaning that the interrupt will only occur when the timer a3 finished

counting. This will prevent that value of m will change during the delay counters making inaccurate phase difference.

In Figure 16 (b), the ADC_update function, two temp values are made to calculate the possible m value. The correlation between number of m and ADC value will be:

$$\frac{\text{ADC value (known)}}{1023} = \frac{value\ of\ M(Unknown)}{255} \tag{3}$$

The equation can be derived as:

$$\text{value of M} = \frac{(\text{ADC value } \times 255)}{1023} \tag{4}$$

The problem is if ADC value is maximum (1023) and result of right side nominator will be 260865. Due to the reason that the MCU is 16bits, the maximum register is 16 bits meaning that the maximum value that could be stored was $2^{16} - 1$ (65535). Therefore, in a section b) temp is reduced 10 times and temp2 is increased 10 times. Using 255 minus the result value is making a function inverse that clockwise to adjust knob will increase the frequency. and the last line of code is fixed the maximum frequency to 785.31HZ, m=14.

Figure 16. (c) is using interrupt to update value of m. the priority of this interrupt is less than timer a3 and higher than timer b0, therefore, it will start after timer a3 finished, and start immediately whenever timer b0 is working or not. It is necessary to close the timer b0 when the interrupt start. All PWM generation timers and flags should be set to 0 to make a reset. Because the interrupt could be occurred at any time, for example between phase B and C update, the delay time will be various. Therefore, every time when interrupt event are occurred all timers and flags should be reset and all PWM timers should be updated at beginning.

Using ADC knob changes the value of m, but only updates the value when the switch interrupt key is pressed.

```
126
127         //event counter mode
128         ta4mr=0x41;     //event counter
129         pd8=0;          //set port as input
130         ta4s=1;         //open timer
131
132         //timer b0 event count mode A phase
133         tb0mr = 0x40;   //timer mode
134         tb0 = 29999;    //10ms counter
135         tb0s = 1;       //open timer
136         tb0ic=0x05;     // set priority
137
```

```
237
238     void timer_b0(void){
239
240         countFreq();
241
242     }
243
244     void countFreq(void){
245
246         if(++sec_count==100){
247
248             freq=currentVal-lastVal;
249             RPM=freq*60;
250             lastVal=currentVal;
251             sec_count=0;
252         }
253
254     }
255
```

Figure 17. Sample code for event counter

The ideal of programming **version I** is to output 120 degree PWM continues wave form that its frequency could be changed based on the value of ADC knob. The eventer counter is for reading a high level signal of hall sensor to calculate the speed of motor. However, it will not make motor running because it not fit the way of the brushless DC motor work.

4.2 Phase output analysis

As in comparison with Figure 24 to 26 and Figure 14 to 15, there has a similar output forms that 120 degree phase shift and same amplitude and frequency. The

version I did not work because it did not put the PWM wave as sequence. The motor is wired with hall sensor (seeing Figure 18.) [6].

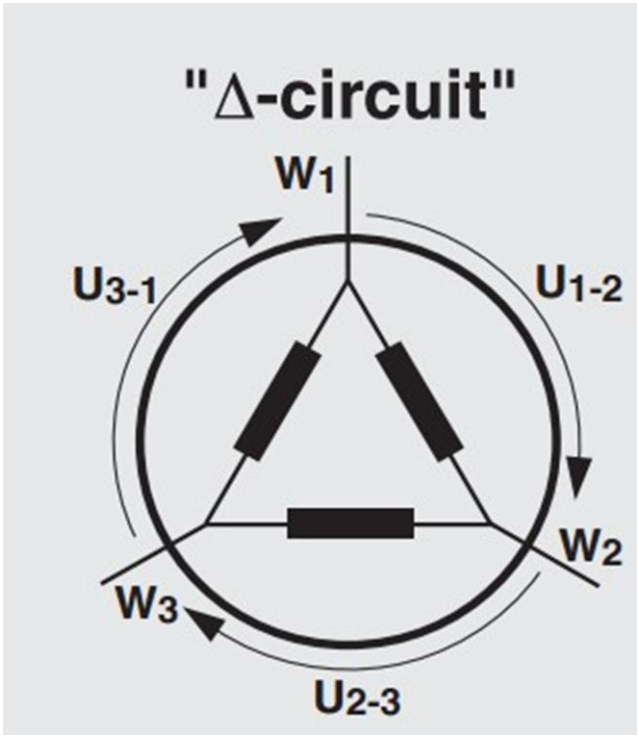


Figure 18.Maxon™ datasheet [8]

The points W1, W2 and W3 are connected to the driver board output. With this method of wiring, the input also need follow the certain sequence. Otherwise, three phases will all time power up and lock the motor. The following Figure 19. is also from Maxon™ datasheet. It tells how the sequence would be when driving the motors.

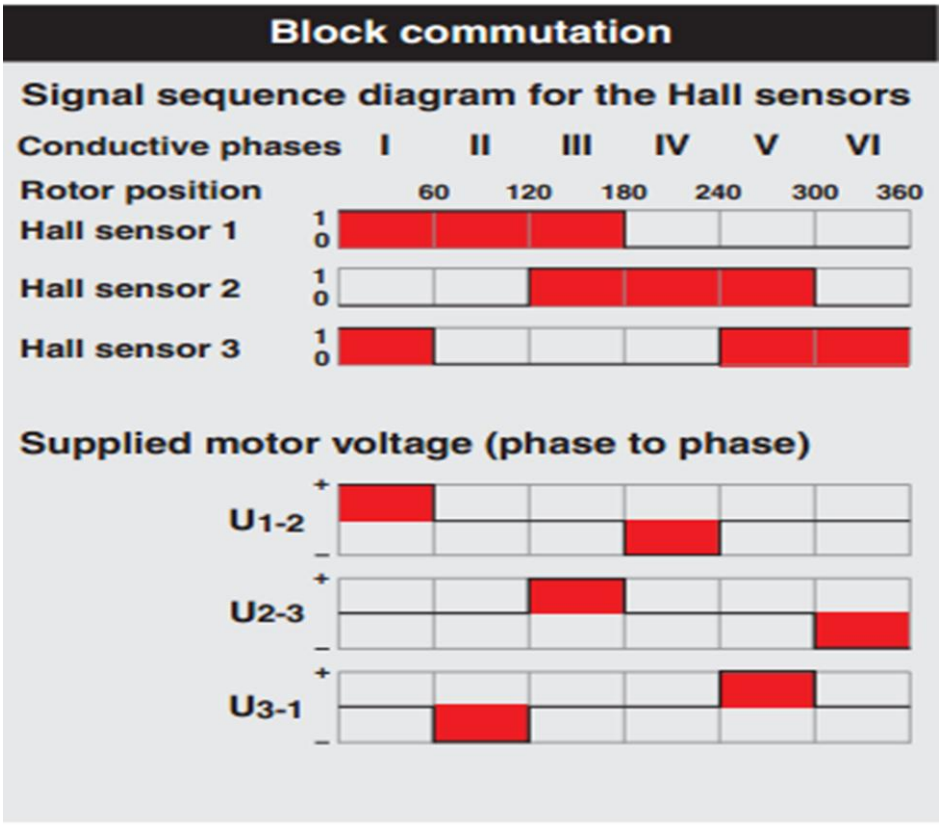


Figure 19. Relationship between hall sensor and output

As the above figures is indicated, the phase output should be triggered based on the feedback that is gained from the hall sensor. The output will open the P channel MOSFET to make a current flow to the motor, and the following phase N channel to flow the current out. This will active one phase of motor, and according to the position of the hall sensor rotor, the active phase will be changed. If those entire processes work at very fast speed, the output would look like 3 continue PWM wave with 120 degree time shift.

4.3 Software Programming (Version II)

As being analysed above, the output PWM wave should be triggered by the hall sensor signal input. Fortunately, the microcontroller itself has a functionality that can output three phase, six PWM signals, controlled by the INT interrupt. Therefore, the second version of the programming will be proceeding as following. 1) Reading the hall sensor feedback. According to the previous analysis, the PWM output signal will be based on the input of hall sensor feedback signal. Here, three INT interrupts are declared as receiving the hall signal. 2) As it has the feedback of rotor, the PWM output will be generated from timer A 1, 2 and 4 as one shot mode. The frequency of PWM output will be depended on the speed of timer B2 counter. It is necessary to set a correct timer register. 3) The output signal will be sent to the idb0 and idb1 register. Hence, the phase sequence control will be done by setting to those two registers [10].

4.4 Reading Hall Sensor

```
164   void int0_int(void)
165   {
166
167    if(edge0_flg==0)
168    {
169     edge0_flg=1;
170       rad_datw=60;     //Next times, angle setting  Next output phase = V  WB
171    }
172
173    else
174    {
175     edge0_flg=0;
176       rad_datw=240;    // Next times, angle setting Next output phase =  W    VB
177    }
178
179     rad_to_pwm();      // Next times, output phase setting
180     idb0=idb0_bufb;    // Output phase change
181   }
```

a

```
183   void int1_int(void)
184   {
185
186    if(edge1_flg==0)
187    {
188     edge1_flg=1;
189       rad_datw=180;   // Next times, angle setting Next output phase =  W  UB
190    }
191
192    else
193    {
194     edge1_flg=0;
195       rad_datw=0;     // Next times, angle setting  Next output phase = U   WB
196    }
197
198     rad_to_pwm();     //Next times, output phase setting
199     idb0=idb0_bufb;   // Output phase change
200   }
```

b

```
202   void int2_int(void)
203   {
204
205    if(edge2_flg==0)
206    {
207     edge2_flg=1;
208       rad_datw=300;   // Next times, angle setting Next output phase = U   VB
209     rad_to_pwm();
210    }
211
212    else
213    {
214     edge2_flg=0;
215       rad_datw=120;   // Next times, angle setting  Next output phase =  V   UB
216    }
217     rad_to_pwm();     // Next times, output phase setting
218     idb0=idb0_bufb;   // Output phase change
219   }
```

c

Figure 20. Sample code for hall sensor feedback

As can be seen from the Figure 20, the three INT interrupts are used for receiving the hall sensor information. The related ports are connected to the three of hall sensor pins on the connector board. Every time, when hall sensor sends high level signal to MCU will trigger the interrupt. The interrupt will proceed and re-set the output register.

```
30
31    #pragma INTERRUPT/B int0_int
32    #pragma INTERRUPT/B int1_int
33    #pragma INTERRUPT/B int2_int
34
```

a

```
118       pd8_2=0;
119       pd8_3=0;
120       pd8_4=0;
```

b

```
106       /*Use INT for position input*/
107      ifsr=0x07;      /*INT 0,1,2  both edges*/
108      int0ic=0x0f;    /*INT 0,1,2 interrupt level setting*/
109      int1ic=0x0e;    /*INT 0,1,2 interrupt level setting*/
110      int2ic=0x0d;    /*INT 0,1,2 interrupt level setting*/
```

c

Figure 21. Sample code for interrupt setting

In Figure 21 a) shows the declaration of interrupts. The /B means the speed of interrupt response timer will be much faster by using CPU register Bank1. b) Indicates that relevant ports are declared as input. c) Both edges are setting as interrupt and level setting is int0 highest and int2 lowest. Because when any of two interrupts occurs, the system will process the earlier one until it finished. It will avoid problem that two interrupts started at same time, two phases are opened at the same time, stops the rotor rotating.

4.5 PWM Wave Generation

As being mentioned above, the PWM waves are generated based on the input hall sensor. Once the interrupts are triggered, system will send a half width pulse out. The speed of sending pules (frequency) will depend on the speed of timer B2 counter.

```
    tb2s=0;

    prcr=0x02;   /* Protect*/
    ictb2=1;     /*  One TB2 interrupt at every other TB2 underflow*/
    invc0=0x07;  /*Select 3-phase motor control timer functionm, trianglar wave*/
    invc1=0x5a;
    prcr=0x00;   /*Protect end*/

    prcr=0x01;   /*Protect*/
    tb2sc=0x01;  /* has to be 1 */
    prcr=0x00;

    idb0=0x03f;    /*Set 3-phase output buffer register to " 1" */
    idb1=0x03f;    /*Set 3-phase output buffer register to "1" */

    ta1mr=0x02;   /*One-shot pulse mode */
    ta2mr=0x02;   /*One-shot pulse mode */
    ta4mr=0x02;   /*One-shot pulse mode */
    tb2mr=0x00;   /*Timer mode   */
    trgsr=0x55;   /*Trigger select register TB2 trigger */

    tb2=11999;  /*Carrier speed */
    ta4=6000;   /*width 50%*/
    ta1=6000;   /*width 50%*/
    ta2=6000;   /*width 50%*/
```

Figure 22. Sample code for timer setting

The timers and registers setting are based on the M16C/62P Group (M16C62P, M16C/62PT) Hardware Manual [9]. The invc0 and invc1 are three-phase control register and only re-write when the prcr register setting to 2. The setting is for modulated from triangle wave and counter source is selected as f1-24MHZ. the output will depend on speed of timer B2. Register idb0 and idb1 are initialized to 1. The b7 and b6 are revered as 0. Timer A 1, 2 and 4 are one shot mode in f1 counter source. The tb2 as normal counter mode with f1 counter source as well. Tb2 value is set to (12000-1) means that it would be 1/2000 of f1. The timer A are set half of timer tb2 means that the PWM high level width will be 50%. This value will be sent to idb0 and idb1 for phase output. The phases sequence control

would be done in side of interrupt shown in Figure 20. This program will check where the rotor is and according to the interrupt to select the next phase output.

```
216   void rad_to_pwm(void)
217   {
218    switch(rad_datw)
219    {
220     case 0:
221         idb0_bufb=0x0de;  /* Next output phase = U   WB  */
222      break;
223     case 60:
224         idb0_bufb=0x0db;  /* Next output phase = V  WB   */
225      break;
226     case 120:
227         idb0_bufb=0x0f9;  /* Next output phase =  V   UB  */
228      break;
229     case 180:
230         idb0_bufb=0x0ed;  /* Next output phase =  W  UB    */
231      break;
232     case 240:
233         idb0_bufb=0x0e7;  /* Next output phase =  W    VB  */
234      break;
235     case 300:
236         idb0_bufb=0x0f6;  /* Next output phase = U   VB    */
237      break;
238     default:
239         idb0_bufb=0x0ff;  /* Next output phase = off    */
240      break;
241    }
242   }
```

Figure 23. Sample code for phase output

Shown in Figure 23, after the selection of phase output is made, the next output phase can be chosen from this table. Every two circle of rotor rotate will be gone through all the six phase output.

With tested both version of codes. The version (II) code could drive the motor and make it with various speeds. The **version I** programming went to the wrong proceeding direction. Without deeply analysis of brushless DC motor, the copy of the wave form will NOT make the motor running.

5 SYSTEM TESTING

System testing is worked with every necessary section. First, after the driver PCB board is made, whether driver hardware working properly is checked. Then, after has done programming **version I**, PWM output from MCU is tested from oscilloscope. However, the tests of software were discussed in the section of Microcontroller Programming. In this section, the main concentration is on the PCB & Hardware testing and the system testing and analysis.

5.1 PCB Testing

For both PCBs (Motor drivers and Connectors) are tested connectivity and value of each resistance and capacitance. The connectivity is checked by multi-meter placing probes at beginning and ending of the closed circuit that it will beep if two points could have a current flow. For example, one probe is placed at the power or ground, and another probe hit the points that should be linked with a connection. The connection that missed from driver PCB Prototype Three (missing power line connected to the sets of capacitors – mentioned in Figure 7.) were checked out by this method. By using Multi-meter, resistors and capacitors are checked if they are placed at right place and all the value are correct.

5.2 Driver PCB Testing

After having checked connectivity, whether circuit works as demand that each phase of motor could have a PWM wave form output with current flow should be tested as well. This circuit is the most important part of the entire project working. Therefore, it takes longer time for testing it. The input PWM is simulated by function generator, which input the PWM to the driver circuit and output are read

by using oscilloscopes. One phase of circuit is tested at each time because they have the same design.

Measuring the output value from the Maxon™ DEC 24/3 (Digital EC Controller) is started first which is the motor driver that made by Maxon™ for its motor. In theory, if the output of the circuit can be identical with this. It would be able to drive the motor. Figure 24 to Figure 25 shows the PWM output from the controller as well as input for motor winding.

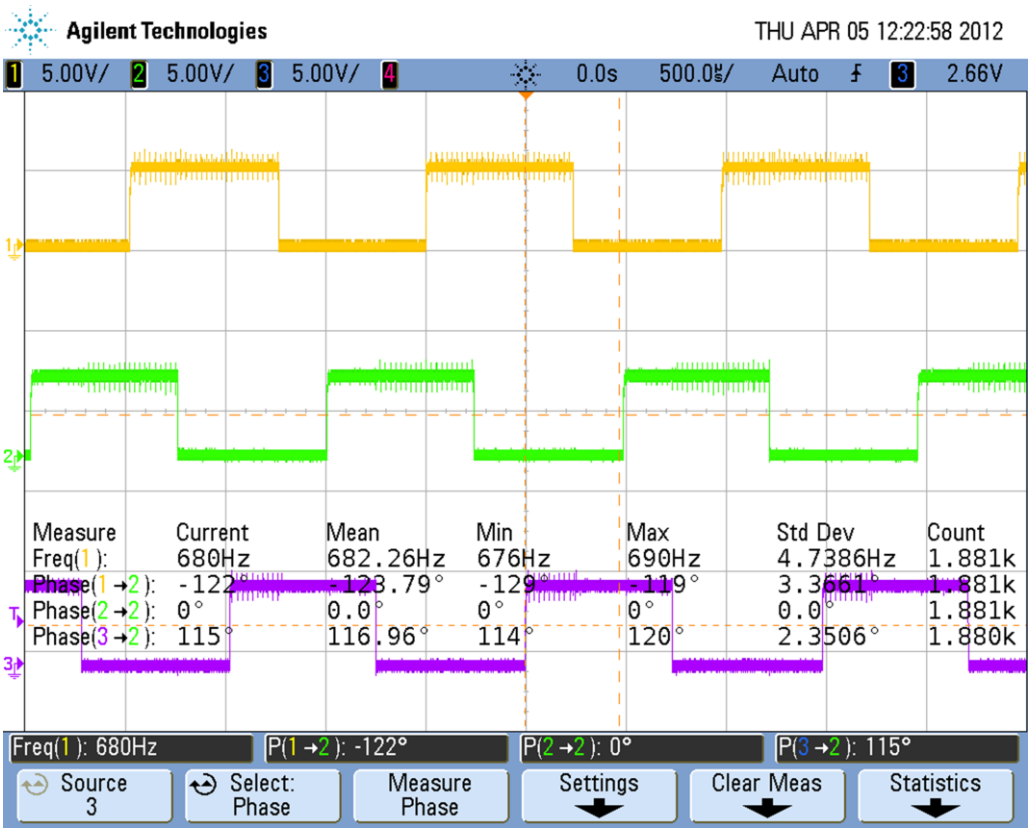


Figure 24. Oscilloscope captured from Maxon™ driver output

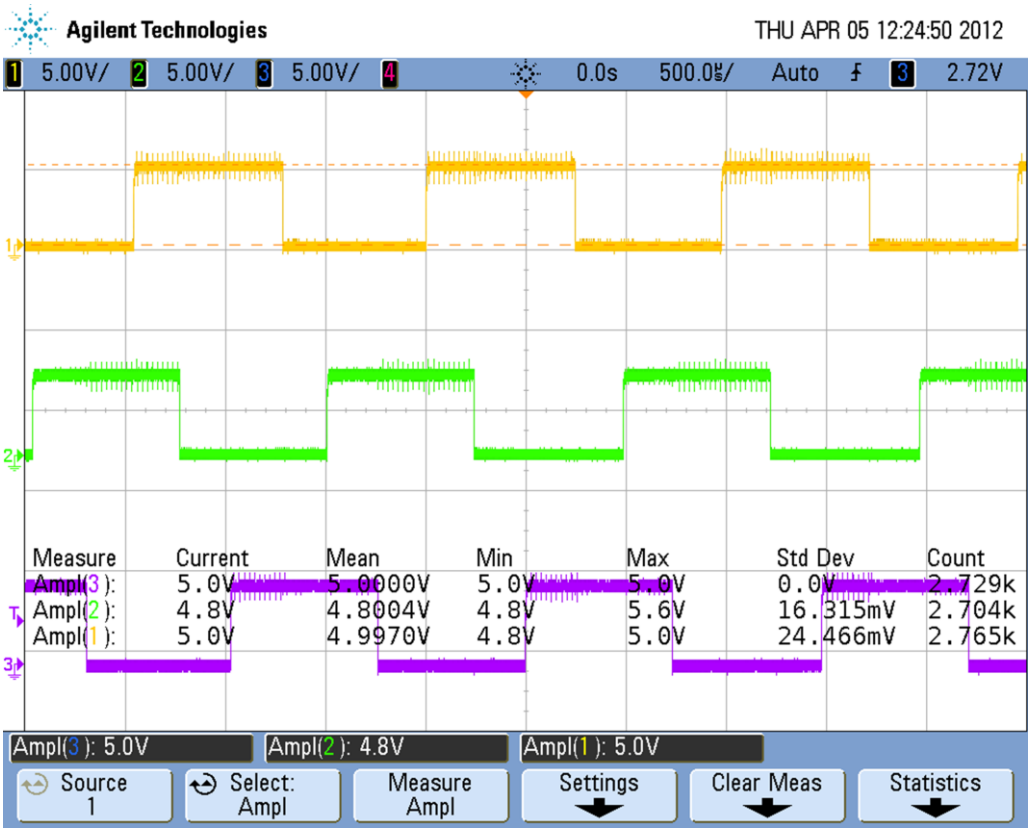


Figure 25. Oscilloscope captured from Maxon™ driver output

Figure 24 to Figure 26 are measured with three phase of output. The yellow, green and violet indicate probe 1, probe2 and probe3, which hit the output pin 1, 2 and 3 (three phase input pins) respectively. Three of those were all set 5V at voltage division and 500uS at time division meaning that each vertical block on the figure stands for 5V and each horizontal block shows 500us. From Figure 24, as data shown, each output PWM signal has 120 degree phase difference. In total 360 degree to drive on rotate of motor. With increasing of frequency, the period is reducing, therefore, the 120 degree phase difference are hard to be stable. According to the Figure 24, the 10 degree variance can be tolerated by the motor. Figure 25 records that high level voltage is around 4.7V to 5.0V and voltage form are a quit stable. As seen from figure, even each phase has around 120 degree phase difference, the frequency of each output were almost identical from the Figure 26. As can be seen from the Figures, it seems that there is no need to use

pull up MOSFET driver because the output PWM is around 5V. The high level voltage from MCU is also around 5V. The reason that using pull up MOSFET driver is not only to pull up the voltage as demand but also set the same voltage as source (s) pin of P channel MOSFET. If the supply voltage to the source (S) pin is 5.3V however the output voltage from MCU is only 4.7V. It would lead 0.6V voltage difference and make small current flow to make a short circuit. The Pull up MOSFET driver will avoid this situation due to its output voltage will be as same as power supply voltage which as well as connected to source (S) pin of P channel MOSFET.

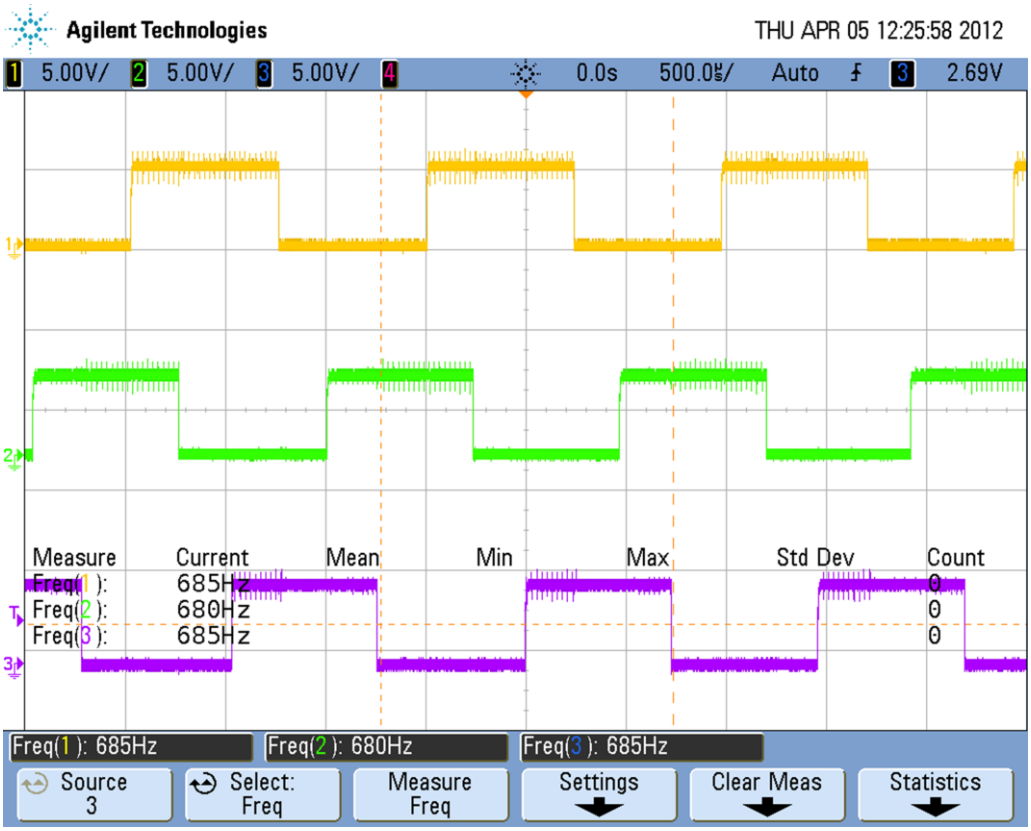


Figure 26. Oscilloscope captured from Maxon™ driver output

One set of capacitors (shown in Figure 7 with yellow circle) are connected with power line and power on the circuit under the current limit protection. Then 2

function generators are set as same frequency to simulate the PWM wave. After that, measurements are started.

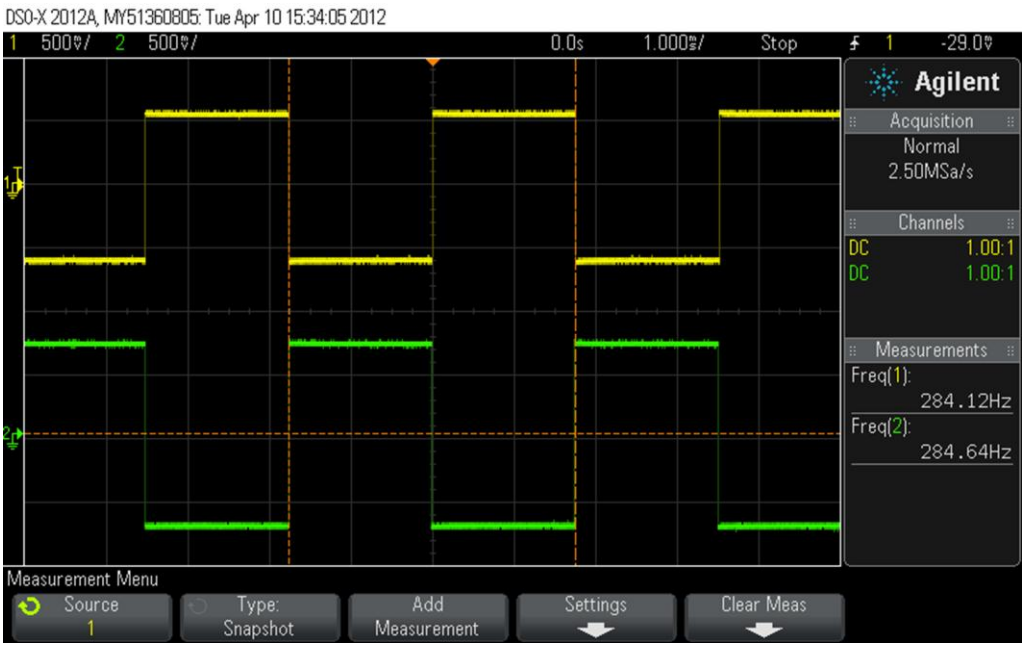


Figure 27. Oscilloscope captured from PCB driver output

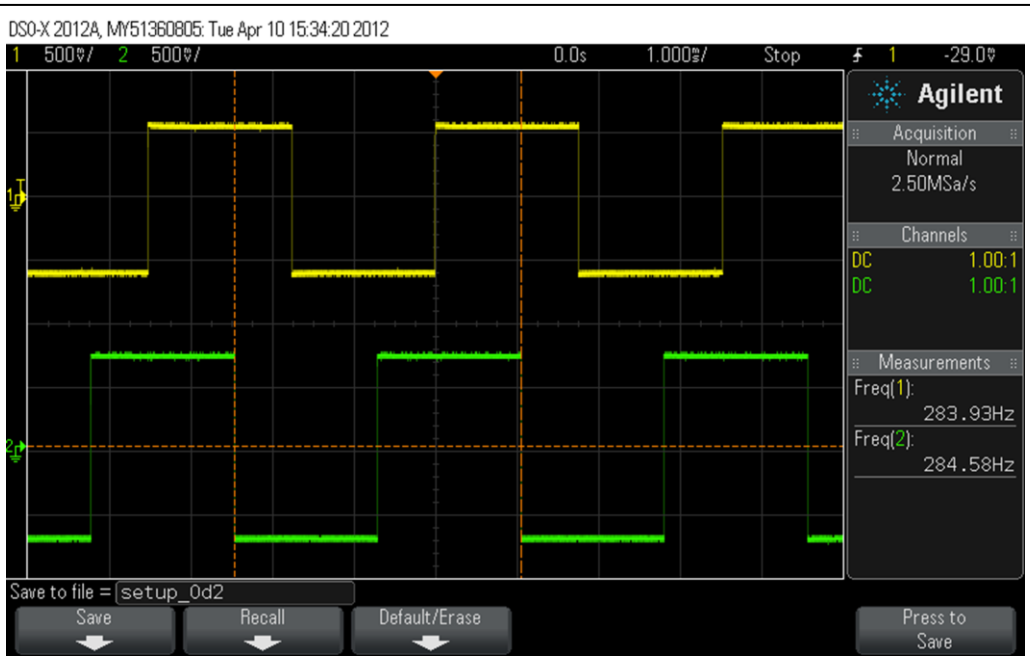


Figure 28. Oscilloscope captured from PCB driver output

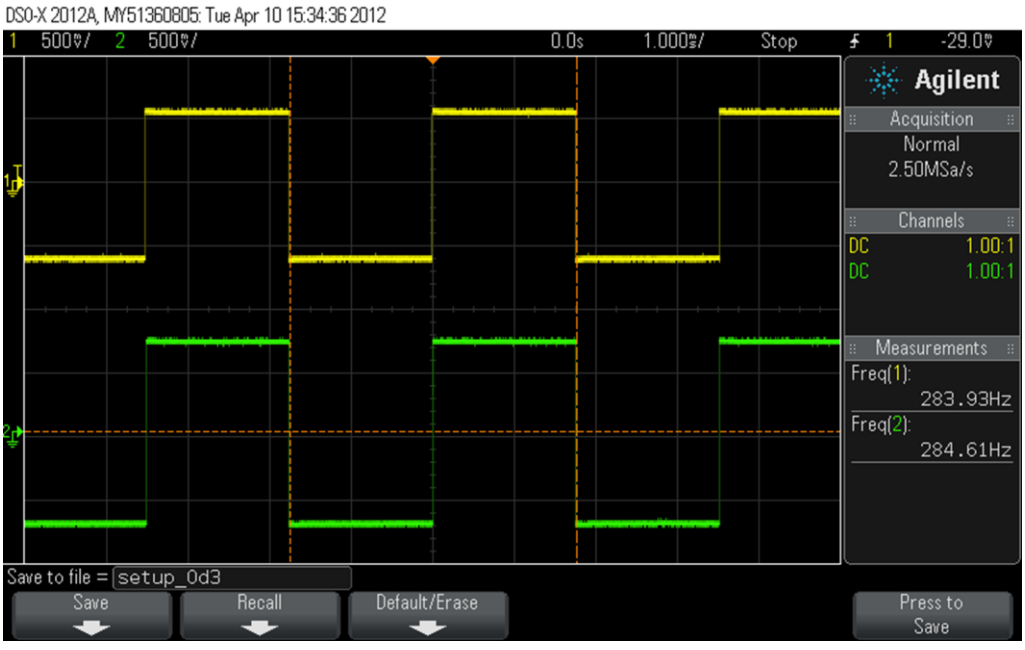


Figure 29. Oscilloscope captured from PCB driver output

However, there are plenty of problems during this simple process. First, the ground of two function generator are put together and are connected with power ground, which means that there is only one common ground in the circuit. Every time, when the PWM pins are connected, the current will go very high to above the limitation, which means a short circuit occurs. Then, power ground and digital ground are separated. It is better but unpredictable; sometime it still got a short circuit. Figure 27 and Figure 29 below shows the current limitation and how the grounds are connected.

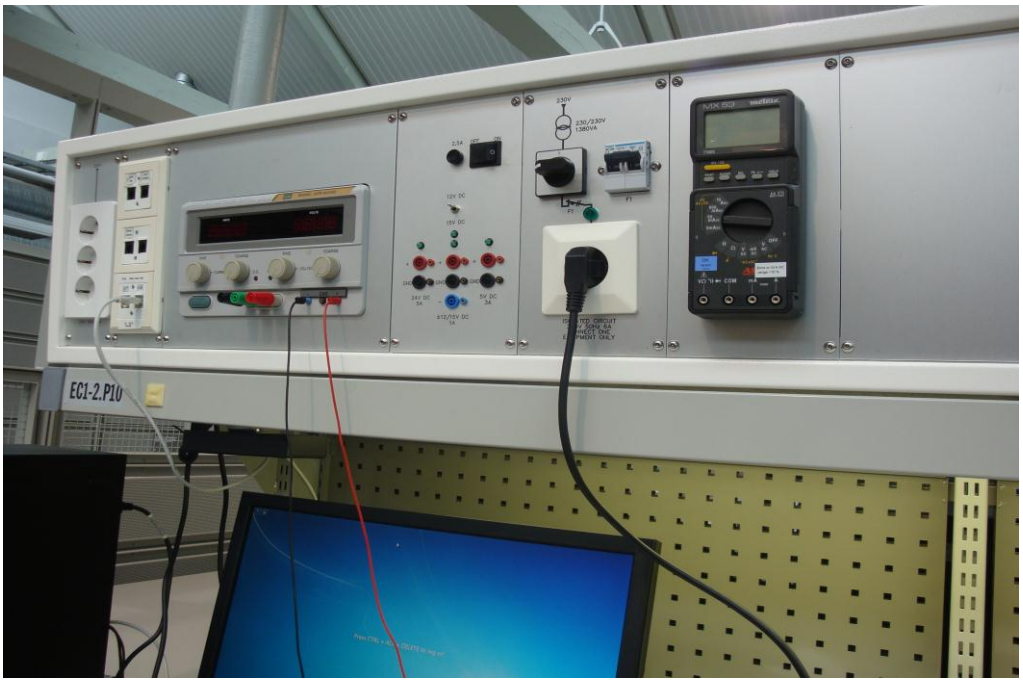


Figure 30. Photos for power connection

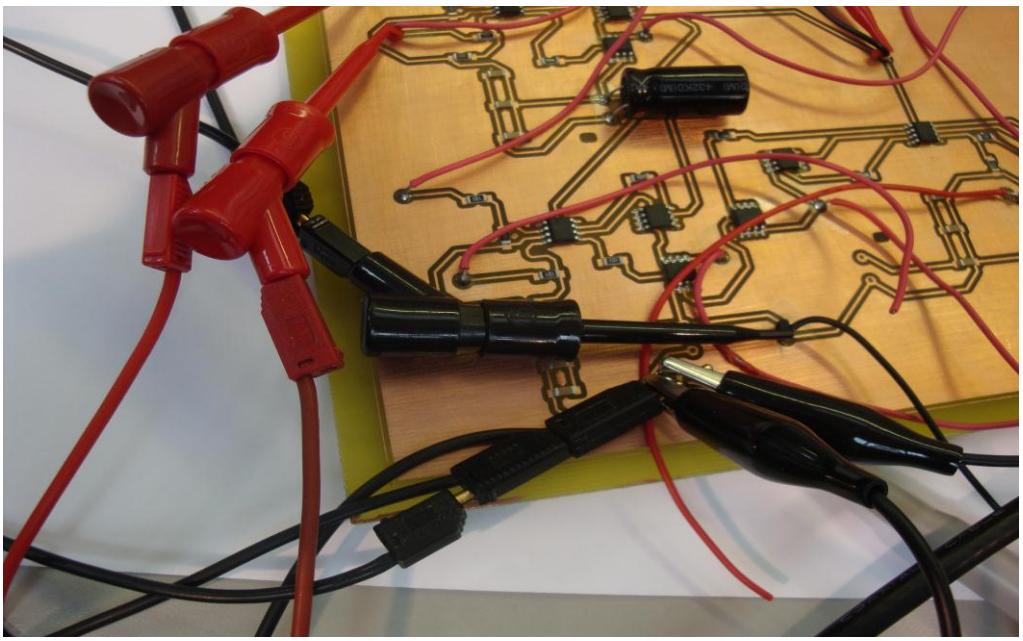


Figure 31. Photos for ground connection

Then, the same procedure is applied to the other phase of circuit. The other circuits are found that situations are even worse. If it is connected with one pins (connection to P MOSFET), the current is fine. Moreover if it is connected with other (connection to N MOSFET), the circuit is short. Therefore, each pins and its functionality are checked founding that when N MOSFET has a power input, the two phase circuit where far away from the capacitor set will have an unstable power input. It makes a P and N MOSFET in open state at same time, which results in a short circuit. The rest of capacitor sets are connected on the circuit to avoid this kind of interference issue. If no ground is connected, the interface will be like a sine wave showing Figure 32.

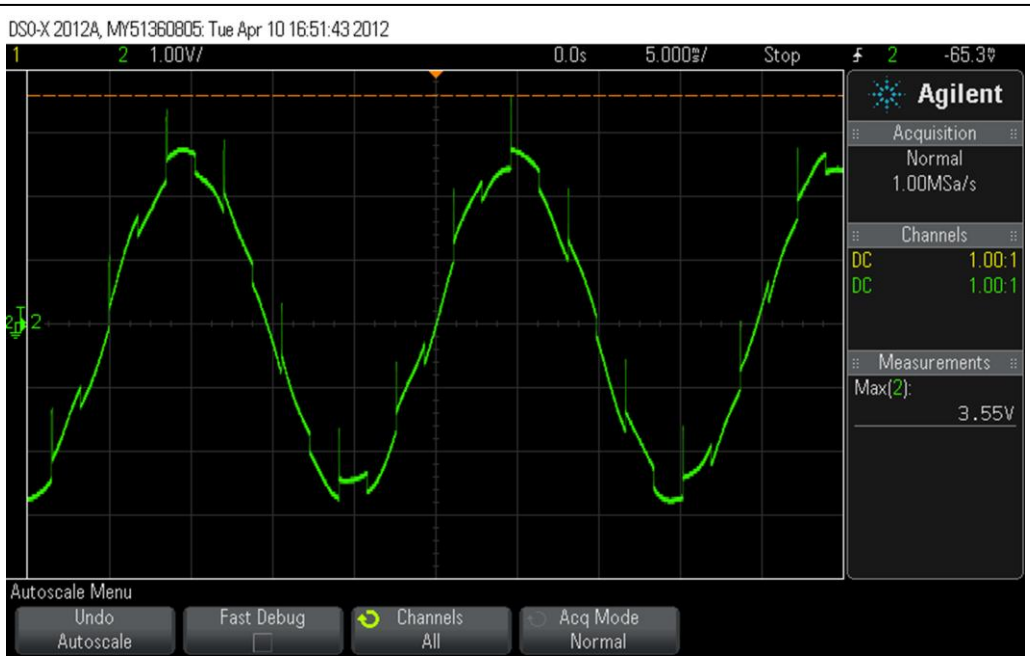


Figure 32. Oscilloscope captured from interference issue

Even the interference issue could be fixed by connecting to the ground; there is still a problem of instantaneous short circuit. Then, how the system worked will be discussed below.

5.2.1 Pull up MOSFET driver

This driver can pull up the voltage as demand. However, this MOSFET driver is found that has an inside logic inverter which is connected with N MOSFET.

In real measurements, the amplitude of output voltage is smaller than the input voltage. The strange situation shows that there is no pull up on the both pins while the inverse function is working fine. The maximum of voltage from function generator is 2000mV (2V), which are much less than high level voltage 4.7V – 5V.

DC CHARACTERISTICS

Electrical Specifications: Unless otherwise noted, over operating temperature range with 4.5V ≤ V_{DD} ≤ 18V.

Parameters	Sym	Min	Typ	Max	Units	Conditions
Input						
Logic '1', High Input Voltage	V_{IH}	2.4	—	—	V	
Logic '0', Low Input Voltage	V_{IL}	—	—	0.8	V	
Input Current	I_{IN}	−1.0 −10	— —	+1.0 +10	µA	0V ≤ V_{IN} ≤ V_{DD}

Figure 33. MOSFET driver datasheet [3]

As can be seen in Figure 33, the minimum Logic'1', High input voltage requires from datasheet was 2.4V. It seems that the pull up resistor did not work properly and it cannot be simulated by function generator. However, it turns out the results that the pull up MOSFET pair is burned. Then, another oscilloscope is placed for checking and founding that the older one is getting 10 times smaller as the real value. Meaning that when it 500mv it reality have 5V. At the last, the reason is the probe ratio setting should be 10.0 to 1 instead 1.0 to 1. When it achieved 2000mV, the voltage actually was 20V.The input side's voltage was much higher than the output (output voltage restricted by power supply which is 10V). Therefore, the MOSFET drivers were burned.

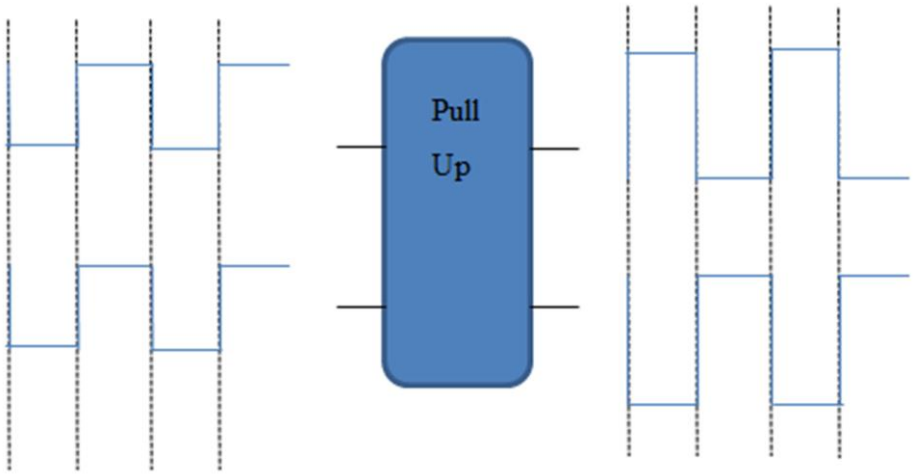


Figure 34. Pull- up MOSFET driver simulation

The Figure 34 shows how the PWM wave form changes in after going through the Pull up MOSFET driver. It not only pull the voltage up to the state that MOSFET requires but also inverted an N channel's wave form. After this, it went to MOSFET to control the current.

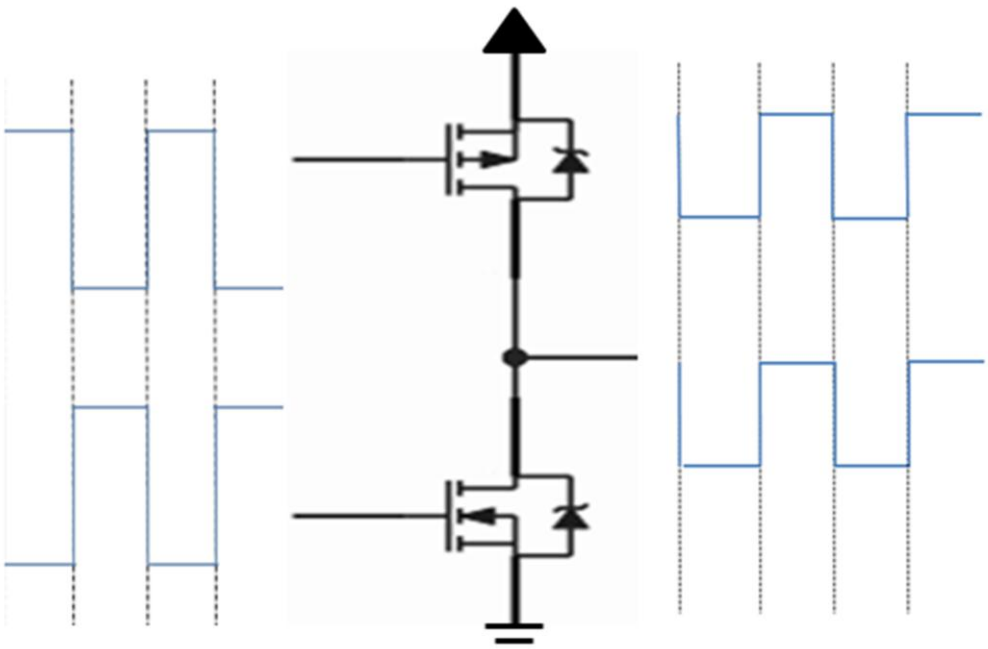


Figure 35. MOSFET pairs simulation

The Figure 30 analysed how the MOFET drivers the Motor. The power supply connected 5V to 30V voltage supply and other side are connected to the ground.

The drain (D) pins are connected together and gate (G) pin are controlled by PWM signal. Because the Pull up MOSFET driver and MOSFET are connected at the same power, therefore the voltage of PWM control signal would have the same voltage as the one from source pin of N channel (voltage potential would be zero).

5.2.2 P & N channel MOSFET

In P channel MOSFET (upper one), when the PWN signal from gate (G) pin as high, there is no voltage difference between the gate (G) and source (S - Power), thus, the output will be closed and no current will flow. On the contrary, when the PWM signal as low, there is full voltage difference between the gate (G) and source (S - power) pins, hence, the output will be open and maximum current will flow. Thus, when the signals from the P channel MOSFET, the low PWM and high PWM stands for the current flow close and open respectively. Similarly in N channel MOSFET (lower one), the N channel has same property but different with comparisons (P channel compare with power and N channel compare with the ground). When PWM from gate (G) as low, there is no voltage difference between the gate (G) and source (S - Ground), therefore, the output will be closed and no current flow as well. When the PWM signal as high, there was full voltage difference between the gate (G) and source (S - ground). The scenario is proved that the PWM signals of P and N channel MOSFET are inversed again in this functional area. In addition when both MOSFET open together it will be a short circuit.

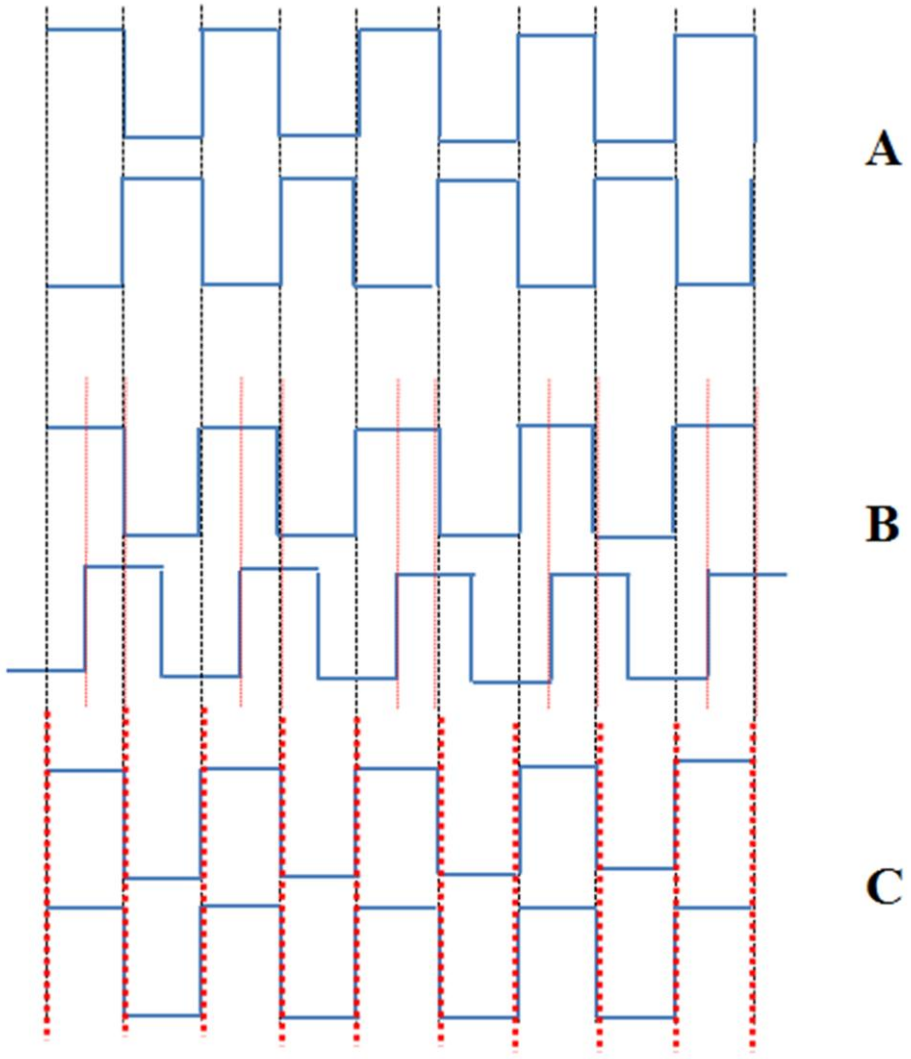


Figure 36. Short circuit scenario simulation

As being shown in the Figure 36, scenario C and B is a problem. Between the red lines, every two high state will cause a short circuit. In the C, it is what has happened when two function generators are connected together or use only one function generator. It will lead a direct short circuit. Scenario B is what happened when with two function generator are connected but separated the ground. Due to

the situation that the two frequencies are hard to be identical, the one wave is all time shifting to another wave. When the two high states meet, it would lead very fast short circuit. That is where the ‘beep’ sound came from. The very fast short circuit is under the component tolerance but will generator side effects. To avoid this, two PWMs need to be critical inversed. From this analysis, the PWM wave form through N channel is inversed twice. One is in pull up MOSFET driver another one is in N channel MOSFET. If one inversion can be avoided, the two PWM wave form will be critically opposite all time (scenario A). When P channel open, the current flow to the motor, and P channel will be closed simultaneously when N channel open, then the current will flow to the ground. Hence, both one period of P and N MOSFET wave can make one fully current flow. According to analysis, a Pull-Up MOSFET driver is changed from MICROCHIP ™TC4428AOCA to TC4427AOCA.

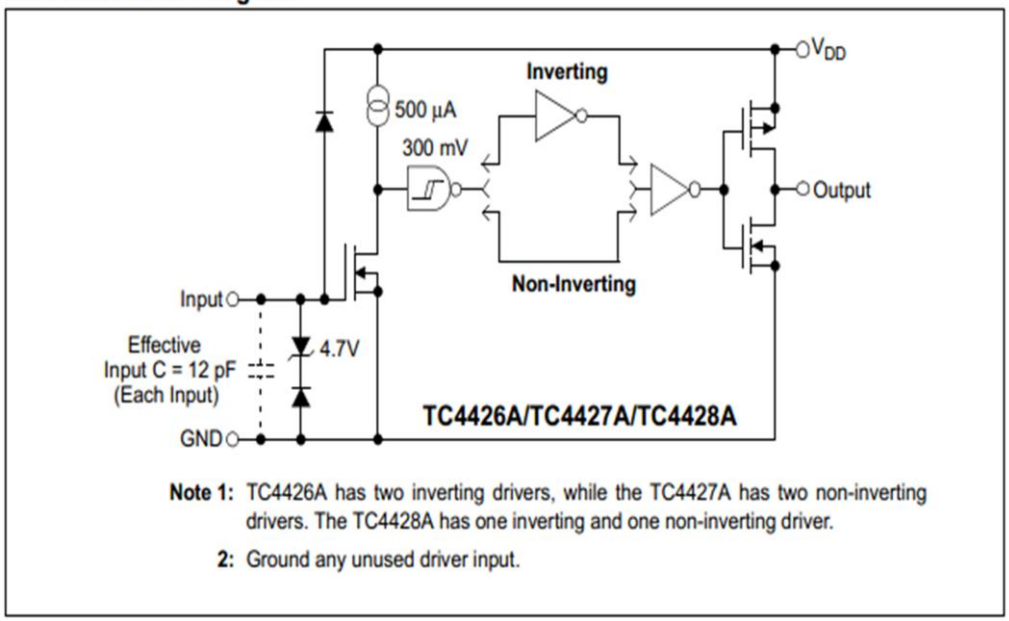


Figure 37. MOSFET driver datasheet [3]

After changing the both non-inverting pull up MOSFET drive, theoretically, the same two PWM input will result a PWM output with current flow. The following Figure 38 will the single PWM output wave form. The wave form is very stable if compared with previous results.

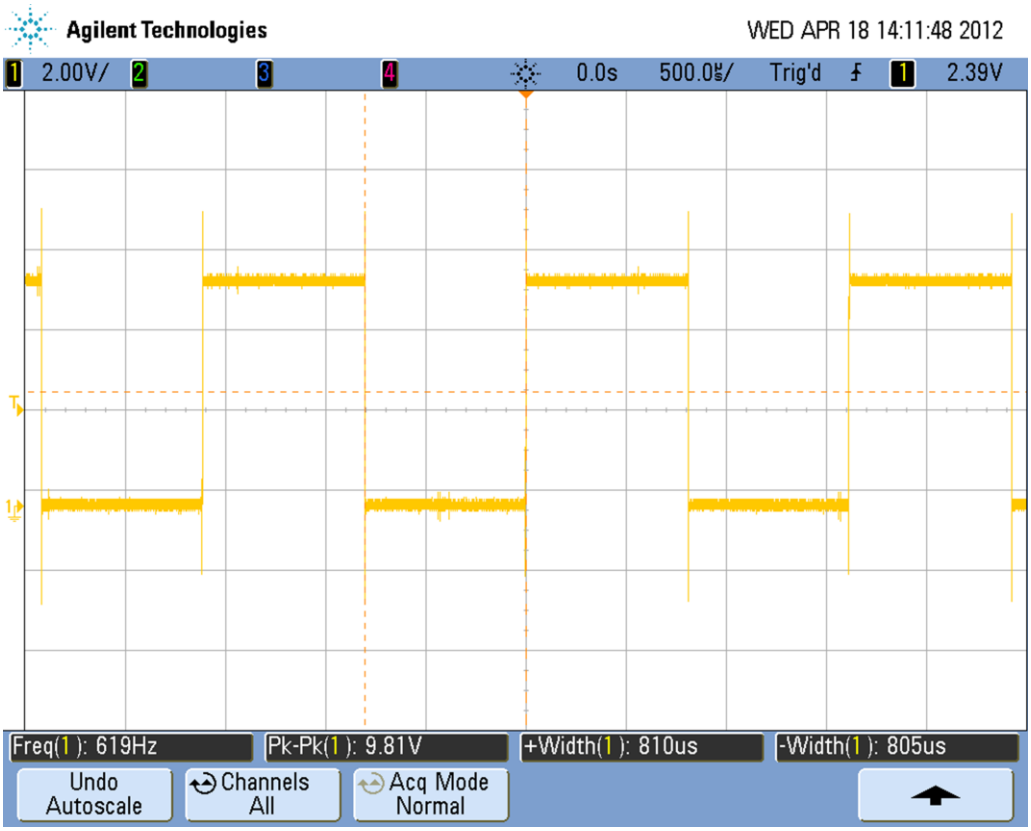


Figure 38. Oscilloscope captured from PCB driver

The input signals are generated from the MCU, because the M value has to be an integer, thus, the high level width and low level width were not ideally identical. However, it still very approaching 50% duty cycle.

Then, with output of 120 degree PWM, 3 phases of circuits were connected. The results will be shown below.

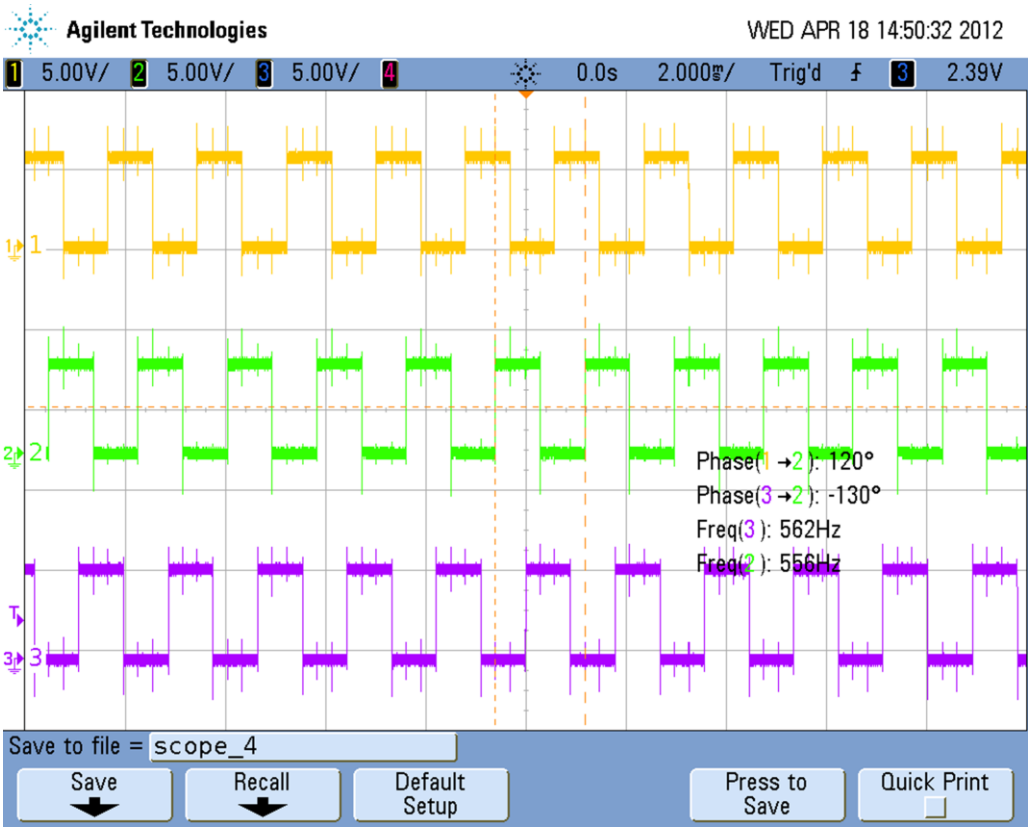


Figure 39. Oscilloscope captured from PCB driver

Each phase has 120 degree phase difference. This Figure 39 looks like the same one that output from MCU, but the difference is that this PWM wave form contained a current flow that could drive the motor.

However, there still is a strange phenomenon that when N channel MOSFET is connected individually, it will be a short circuit; it will be working fine, if the P channel MOSFET is connected. On the other hand, if P channels MOSFET is singular connected, there is no short circuit. After analysing all the possibilities, the reason could be derived from the previous discussion shown in Figure 34.. If there was no input to the P channel MOSFET, system thought that input is always low. According to the character of P channel MOSFET, when source pin (S) and gate (G) pin have a voltage potential difference, there will be current flow. Therefore, the P channel MOSFET will be always open if no PWM input is

connected. At this time, if one input PWM is connected to the N channel MOSFET, meaning that once it in open state, it will lead current flow from power to the ground making a short circuit. Hence, to void this badly short circuit only need connected P channel before the N channel MOSFET.

5.3 Motor Testing

The designed circuit is connected with ready-to-use program to the Motor, trying to make it working. However, the first time it do not working. The reason is from the programming (**version I**) are proved entirely wrong. This result has happened because too much focus is on hardware design and component selection in Pre-study and lacking of background of brushless motor control. Theoretically, the motor is driven by 120 degree shift PWM wave (output wave form from Maxon™ driver), however, in real time, it is not produced continuously. Because it interval too fast, it would be continues PWM output. It means that using timer function to generate the PWM and sends to the phases will NOT make the motor working. Therefore, the program is re-designed and deeply analysed how the Three-phase Brushless Motor would work. The new program (version II) could make a motor running but not proper working at the beginning. Thus, the version (II) program is used to work with the Motor system.

5.3.1 Hall Sensor connection

As being mentioned above, the three hall sensor pins are attached to the INT interrupt input pins. However, the orders of them are not very clear at the beginning. As it is known, there are U, U', V, V', W, W' six phases output in the microcontroller and winding 1, 2 and 3 in side of motor. The winding1 are linked with phase U and U', winding 2 and winding 3 are similarly connected to V, V'

and W, W’. But in connection with INT interrupt, the order should be shifted. At the first, the INT0 to INT2 are in order connected with hall sensor1 to 3. The motor will have no response when the powers are on. Thus, the shift order are attached, finally, the order that INT0 to winding 3, INT1 to winding 1, INT2 to winding 2 are proved as corrected order. If the hall sensor connections are not proper attached, the interrupt will trigger the wrong phase. This result that either the two adjacent phases are triggered making a lock of motor or two opposite phases are triggered making the motor doing nothing. The three-phase brushless DC motor will NOT run without correct hall sensor information inputs. Hence, to connected hall sensors in a right place would be the first thing to do when going to drive a motor.

5.3.2 Wave form

As the same way of checking output, the hall sensor points are placed on the oscilloscope. The output information is identical with the one that observed from the Maxon™ driver. There of them have the same frequency output high level voltage with 120 degree phase shift showing that the motor were driven in a correct method. Unfortunately, the input wave forms are hard to be measured because there are no ground can be touched for oscilloscope probe; the result would be very unstable due to the issues of environment interference.

5.4 Analysis

At the final testing, there still have some unsolved problems. Those problems would be listed here and analysed.

The motor will not start by itself because the output will only triggered by interrupt signal. In this case, there would be no interrupts occurred if not rotate motor by hand. Thus, every time when drive the motor from stillness needs using

hand to rotate the motor and making it triggers the interrupt. The direction of motor will be depended on the sequence of triggered interrupts.

Even through the motor could run properly, there is a big issue of current over flow. The motor driver system, so far, only works with current limitation protection condition. It might be caused by connection. The environment interference would make unstable pulses for microcontroller and it sending this signal to the driver board. If the interfering signal overlap the at the same time domain meaning that there are no critical inversed PWM waves input to the P and N channel MOSFET respectively. The results would be that the P and N channel MOSFET would open at the same time (very short time) and the current would flow from the power side to the ground directly making a short circuit. This might be avoided by using a better connection using a connector and shielded cable instead of unprotected copper wire.

6 CONCLUSION

As all works have been done, it is possible to drive the Three-phase Brushless Motor with modules. The various possibilities could be selected for driving the motor. The FPGA or microcontroller could be used for receiving the hall sensor information and sending driver PWM wave in software programming. The further software could be programmed for mare critical motor control, for instance, using hall information get the position of rotate to control the rotor rotates precisely or detecting the rotor position trigger the next phase of motor making possible that the motor not only can start running by itself but also can set the running direction. In further studies, the motor control could be become more advanced. The possibility of setting running direction, knowing the precise position of rotor and rotating the certain angle of distance could also be achieved. In the hardware, even though now the testing board was bigger than Maxon driver, but in the real application it can be embedded inside of other PCB by self-design. More importantly, the price of it would be much more favourable than that of a ready-made driver board, and it will be more flexible for small company and educational institutions for both application and educational purposes.

REFENCE

[1] Brushless DC motor control using the LPC2141; Application Note; AN10661; © NXP, 17 October 2007.

[2] Skuba 2011 Extended Team Description; Department of Computer Engineering; Faculty of Engineering Kasetsart University Bangkok, 10900, Thailand; 2011.

[3]1.5 A Dual High-Speed Power MOSFET drivers for TC4428A; Datasheet; MICROCHIP™ Technology Inc; 2006.

[4] HEXFET Power MOSFET for IRF7424; Datasheet; International IR Rectifier™.

[5] HEXFET Power MOSFET for IRF7413PbF; Datasheet; International IR Rectifier™.

[6] Maxon Flat motor EC 45 flat, 45mm, brushless, 30Watt; Datasheet; **Maxon™** Motor; 2006.

[7] A Beginner's Guide to the MOSFET; Web source: http://reibot.org/2011/09/06/a-beginners-guide-to-the-mosfet/; 2011.

[8] Maxon motor control 1-Q-EC Amplifier DEC 24/3; Operating Instructions; **Maxon™** Motor; 2007.

[9] M16C/62P Group (M16C62P, M16C/62PT) Hardware Manual; Hardware Manual; REJ09B0185-0241; **Renesas™** Technology; Jan 10, 2006.

[10] M32C/83 Group Concept of the Three-phase Motor Control Program; Application Note; REJ05B0147-0120Z; **Renesas™** Technology; Jun 25, 2003.