

Dead Time Compensation Implementation In MB9Bxxxx/MB9Axxxx Series

Associated Part Family: MB9BFXXXX/ MB9AFXXXX Series

This application note describes dead-time compensation implementation in- MB9Bxxxx/MB9Axxxx Series and the algorithm implementation of software and test performance.

Contents

1	Introduction.....	1	3	Dead-time compensation Implementation	8
1.1	Advantages and disadvantages of using a dead-time compensation function	1	3.1	How to detect current polar correctly	8
1.2	How to realize type	2	3.2	Dead-time compensation Software implementation	11
2	Dead-time Compensation Principle	2	4	Dead-time compensation Function Performance ...	20
2.1	Six phases inverter IPM Structure.....	2	4.1	Basic Verification	20
2.2	Motor Control Block Overview.....	3	5	Conclusion.....	28
2.3	Current aberration and compensation implement	5	6	Document History.....	29

1 Introduction

This application note describes the algorithm implementation of software and test performance. Error voltage vectors caused by dead-time effects of PWM inverter were given, the vector synthesis method was adapted to reduce amplitude and phase formulas of synthesized voltage vector produced by 3-phase stator windings under dead-time effects, the characteristic of synthesized voltage vector was analyzed with simulations. In order to make practical conduct time equal to ideal given time of switching devices, a dead-time compensation method based on time was proposed, simple arithmetic was obtained with the characteristic of space vector PWM (SVPWM). A dead-time compensation method based on voltage was proposed also to eliminate error voltage vector, compensation formulas were calculated in 3-phase and 2-phase static reference frame respectively corresponding to SPWM and SVPWM. Experimental results show that the proposed method can make motor phase current waveform sinusoidal, and improved the output performance of the inverter.

1.1 Advantages and disadvantages of using a dead-time compensation function

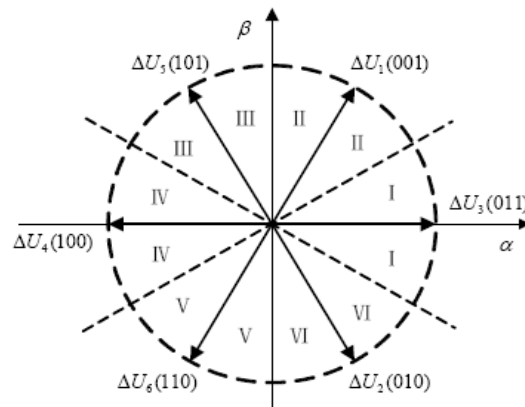
Advantages

One of most important reasons for dead-time compensation and perfect the sinusoidal wave is harmonious wave reduction. Which in turn , Because the six phase inverter IPM of the upper arm and the lower arm don't allow open together, so we need add the delay time at the upper arm and the lower arm switching that will abstain up and down IGBT short circuit cause the IPM destroyed. But this modifying cause the efficiency vector voltage changed. So we add the dead-time compensation reduce this modify effecting. Moreover, add the dead-time compensation can reduce the noise and vibration that motor torque change caused.

Disadvantages

During the dead-time compensation function, a modification on the sinusoidal-modulation pattern needs to be made in order to allow current to be real current. This pattern modification could generate some current ripple. If you detect the current positive or negative polar is error. Due to modification of patterns and correction of the same modifications, more CPU is used to implement this algorithm. Current sector define as shown bellows:

Figure 1. dead-time compensation sector region



1.2 How to realize type

One possible solution to this problem is to ignore current polar detect during these zero cross periods. This is not desirable since some algorithms, including the one used in this application note, require information from all three currents polar in order to add the dead-time compensation to cause the current is sinusoidal. Another solution is to base the estimate position to detect the three phase current polar. This could be one good solution, but this type will have some error at the current D-axis don't be equal to zero. The third solution is to using the assemble current to detect the three phase current polar. Moreover, to add the filters reduce the current wave. This would real detect the three phase current polar. Moreover, we can use the hall sensor to detect the current polar, but this need add the hardware cost. So the third solution is good type. We will focus on this type to describe the algorithm.

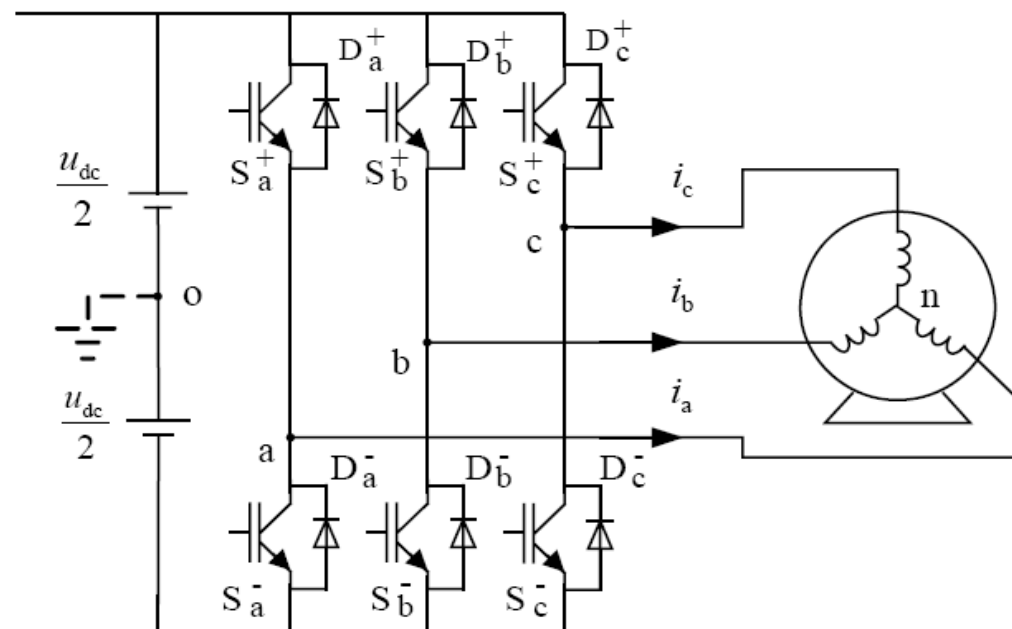
2 Dead-time Compensation Principle

2.1 Six phases inverter IPM Structure

2.1.1 Three phases current module

The PM synchronous motor is a rotating electric machine with a classic three-phase stator like that of an induction motor; the rotor has surface-mounted permanent magnets. IPM work in order to change DC voltage to sinusoidal wave. But such as upper arm IGBT S_a^+ and lower arm IGBT S_a^- at switching need add delay time to prevent the IGBT short circuit. So the real voltage will not the need vector if don't add the dead-time compensation. The IPM and the motor connect and structure as bellow:

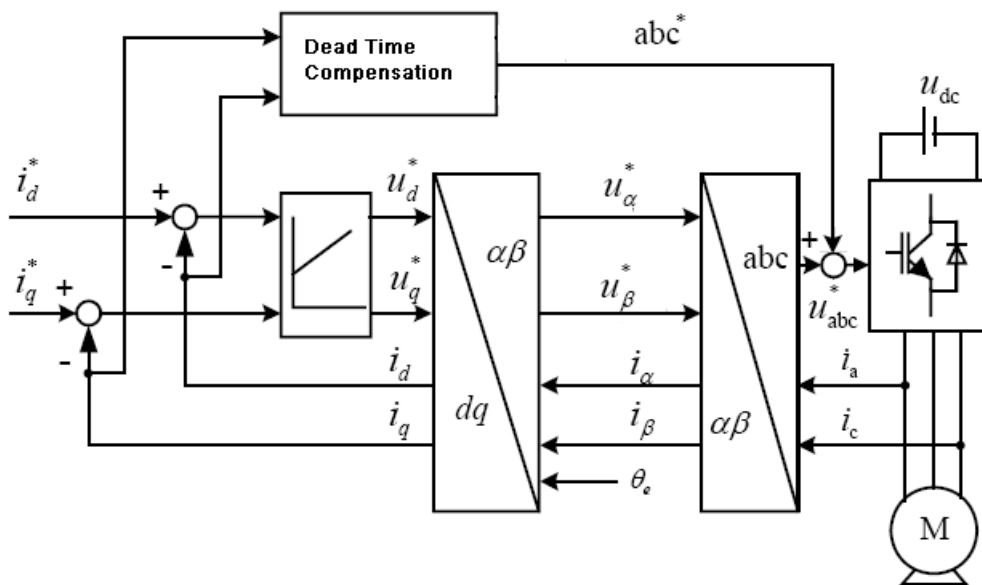
Figure 2. inverter and motor structure



2.2 Motor Control Block Overview

This section describes the PMSM FOC control theory with dead-time compensation. Figure 3 below shows the whole block.

Figure 3. PMSM FOC Block with Dead-time Compensation



Modules explanation:

1. When U_a , U_b , U_c change to the voltage vector, the right switching time will change because the delay time work after adding the dead-time .
2. Base the I_d and I_q filter value can obtain the current polar, so the dead-time infection can reduce overpass the dead-time compensation.

And i_a , i_b and i_c compare with the T_{er} base on the dead-time compensation table as shown below.

Table 1. three phase current positive or negative polar and dead-time compensation

Current Polars	Compensation value
$\begin{matrix} + & - & - \\ + & + & - \\ - & + & - \\ - & + & + \\ - & - & + \\ + & - & + \end{matrix}$	$\begin{matrix} t'_{aon} = t_{aon} - T_{er} \\ t'_{con} = t_{con} + T_{er} \\ t'_{bon} = t_{bon} - T_{er} \\ t'_{aon} = t_{aon} + T_{er} \\ t'_{con} = t_{con} - T_{er} \\ t'_{bon} = t_{bon} + T_{er} \end{matrix}$

The 3-phase currents are converted to a two axis system. This conversion provides the variables i_α and i_β from the measured i_a and i_b and the calculated i_c values. i_α and i_β are time-varying quadrature current values as viewed from the perspective of the stator.

3. The two axis coordinate system is rotated to align with the rotor flux using a transformation angle calculated at the last iteration of the control loop. This conversion provides the I_d and I_q variables from i_α and i_β . I_d and I_q are the quadrature currents transformed to the rotating coordinate system. For steady state conditions, I_d and I_q are constant. We need filter the I_d and I_q value to obtain the standard value, using the value can calculate the theta of current.
4. Error signals are formed using I_d , I_q and reference values for each. The I_d reference controls rotor magnetizing flux. The I_q reference controls the torque output of the motor. The error signals are input to PI controllers. The output of the controllers provides V_d and V_q , which is a voltage vector that will be sent to the motor.
5. A new transformation angle is estimated where v_α , v_β , i_α and i_β are the inputs. The new angle guides the FOC algorithm as to where to place the next voltage vector.

2.3 Current aberration and compensation implement

It is nature to insert a switching delay time in sinusoidal pulse width modulation (PWM) voltage fed inverters to prevent a short circuit in the DC link. This causes well known dead-time effects which distorts the output voltage and current. Many compensation schemes are proposed to overcome the drawbacks. Based on the traditional average dead-time compensation techniques, an improved method was proposed to advance the performance of dead-time compensation. The method is based on SVPWM strategy and it can be implemented with software without any extra hardware. Simulation results demonstrate the validity of the proposed method.

The current only across maintain current diode at the dead-time moment. At this moment the current will decrease until zero value whatever the current polar. Such as the S_a phase as the below structure, if we set the current is positive that flow from the inverter to motor. Otherwise, the current will define the negative polar.

When $S_a > 0$, the current will has two statement, one is normal work statement that upper arm IGBT VT1 will close and the lower arm IGBT VT4 will open, the current flow from the inverter to the motor cross the VT1. The other statement is dead-time moment. The IGBT VT1 and VT4 will shut together, this moment the current will cross maintain current diode VD4 as the same as upper flow. At this condition, the upper IGBT work time will decrease about the dead-time long. So need to add as equal to the decrease time.

When $S_a < 0$, the current will has two statement also, one is normal work statement that lower arm IGBT VT4 will open and the upper arm IGBT VT1 will close, the current flow from the motor to the inverter cross the VT4. The other statement is dead-time moment. The IGBT VT1 and VT4 will shut together, this moment the current will cross maintain current diode VD1 as the same as upper flow. At this condition, the lower IGBT work time will increase about the dead-time long. So need to reduce the time for the dead-time long.

As below topic, due to the dead-time exist, the current don't control by the IGBT switching at the dead-time moment. But the current increase or decrease is confirmed by the current polar. Moreover, IGBT need some time at switching and IGBT and diode will engross the voltage. So all the condition will cause the voltage are not real vector.

Figure 4. one phase different current polar

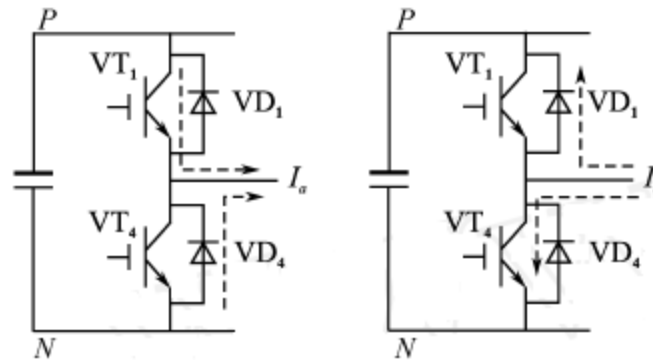
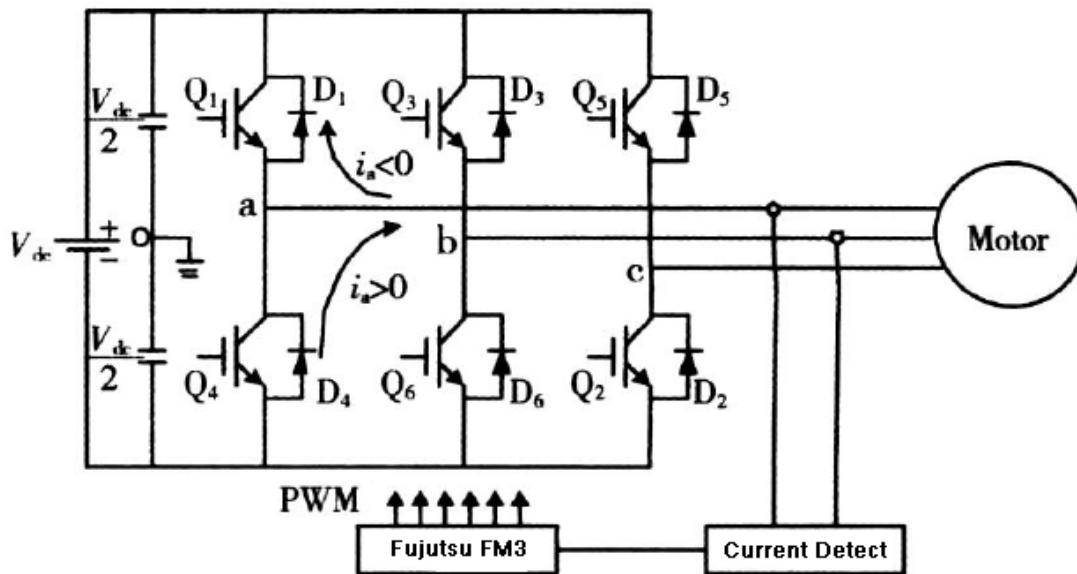
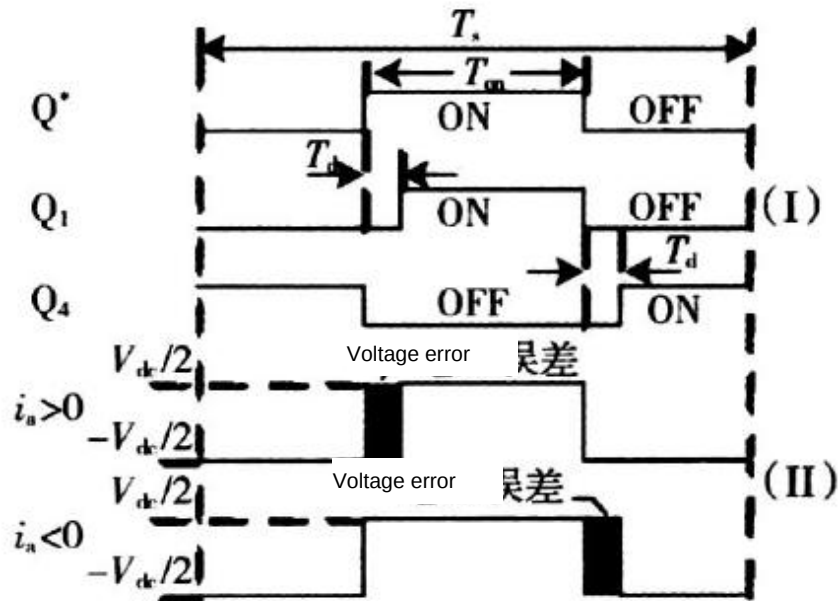


Figure 5. Current polar director



So we will to modify the pulse by below figure base on the current polar.

Figure 6. Compensation the time of PWM



Moreover, some motor don't have the sinusoidal wave and that have the harmonious wave, and the phase current will became as below figure. So add upper dead-time compensation don't change the wave to sinusoidal. We must to reduce the harmonious wave, that wave may be change to sinusoidal. Such as the I_a , we reduce the harmonious wave, the I_a current will reach the sinusoidal as below figure.

Figure 7. Simulator the three phase's current

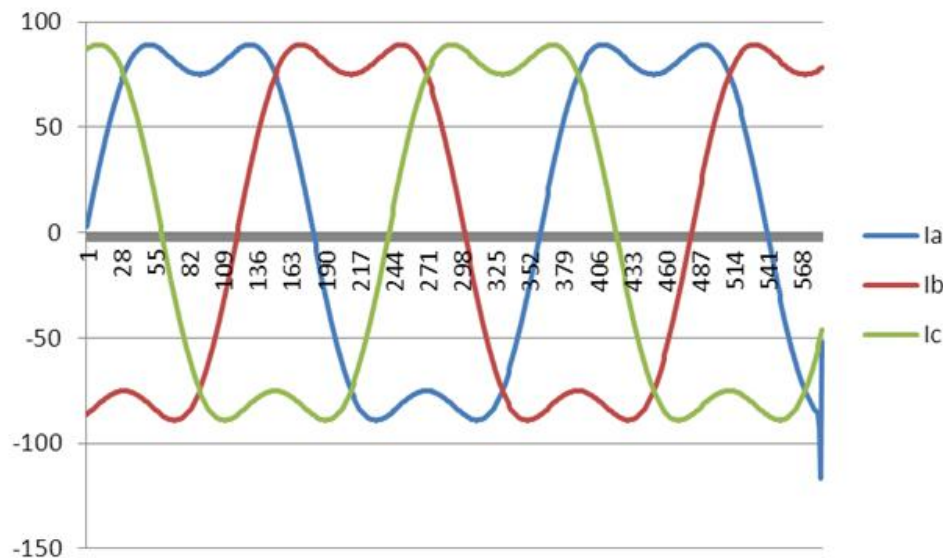
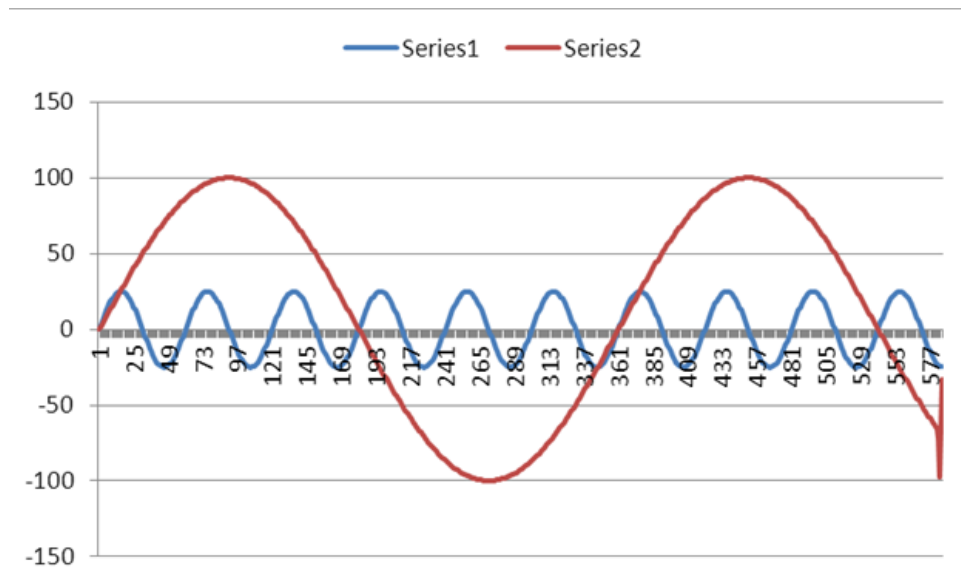


Figure 8. decrease the harmonious wave current



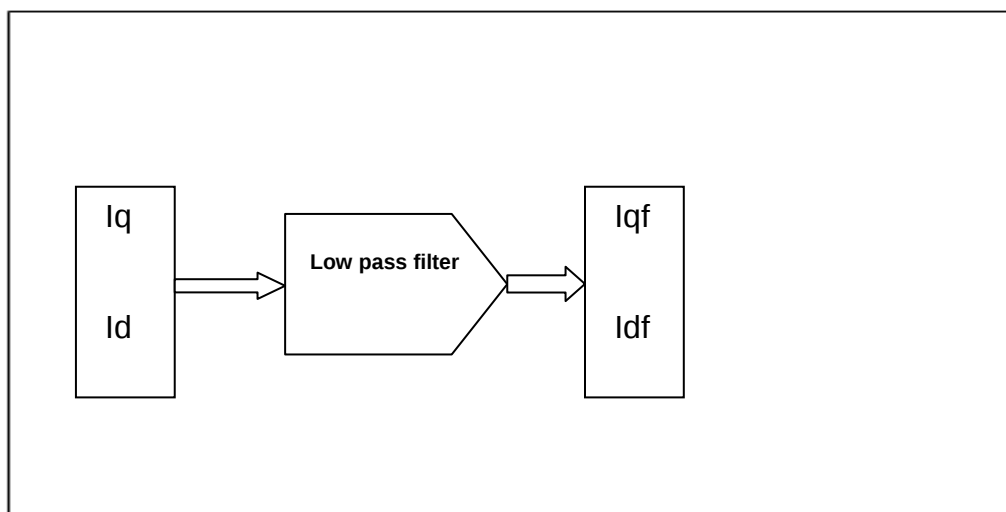
3 Dead-time compensation Implementation

3.1 How to detect current polar correctly

3.1.1 Obtain the current vector

In control system has some yawp, in order to obtain the data cleanly, we will add the low pass filter. The I_q and I_d input and I_{qf} and I_{df} will output. So the high frequency yawp will be filtered. And the flow chart as below:

Figure 9. Dead-time low pass filter



The rotor axis I_{qf} and I_{df} change to state axis I_α and I_β by inverse Clarke transforms. At the same time we can obtain the current angle. The equation as below:

Figure 10. Clarke transforms equation

$$\begin{bmatrix} i_{\alpha} \\ i_{\beta} \end{bmatrix} = \begin{bmatrix} \cos \omega t & -\sin \omega t \\ \sin \omega t & \cos \omega t \end{bmatrix} \begin{bmatrix} i_{df} \\ i_{qf} \end{bmatrix} = I_s \begin{bmatrix} \cos \theta_i \\ \sin \theta_i \end{bmatrix}$$

Figure 11. sector and current polar

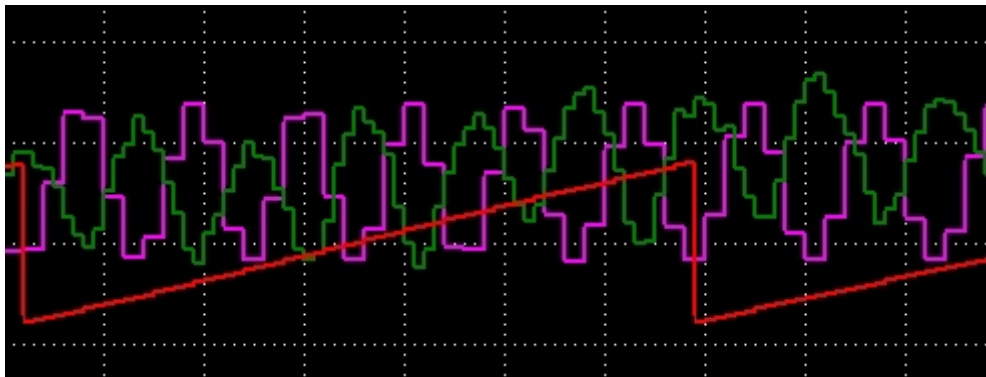
By the current angle θ_i from the equation was calculated. We can judge the current vector what it is sector at state axis. Also the compensation vector will be ensured.

Sector	i_a	i_b	i_c
I	+	-	-
II	+	+	-
III	-	+	-
IV	-	+	+
V	-	-	+
VI	+	-	+

3.1.2 Special motor dead-time compensation

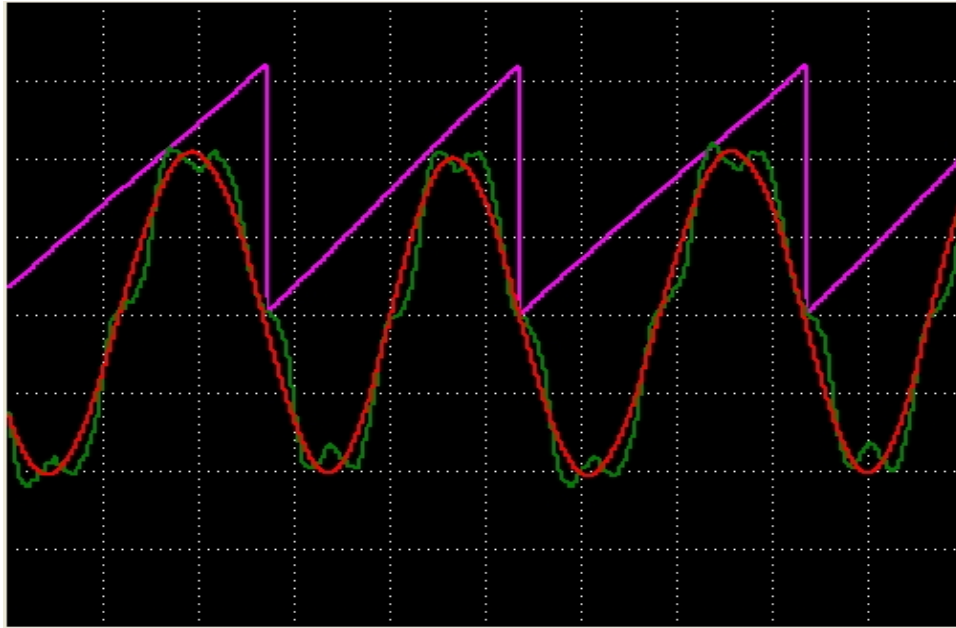
Now has many motor has sinusoidal wave. But a lot of motor using isn't sinusoidal wave now. So we need reduce this characteristic effect. The operation as shown bellows:

Figure 12. torque effect at no-sinusoidal BEMF



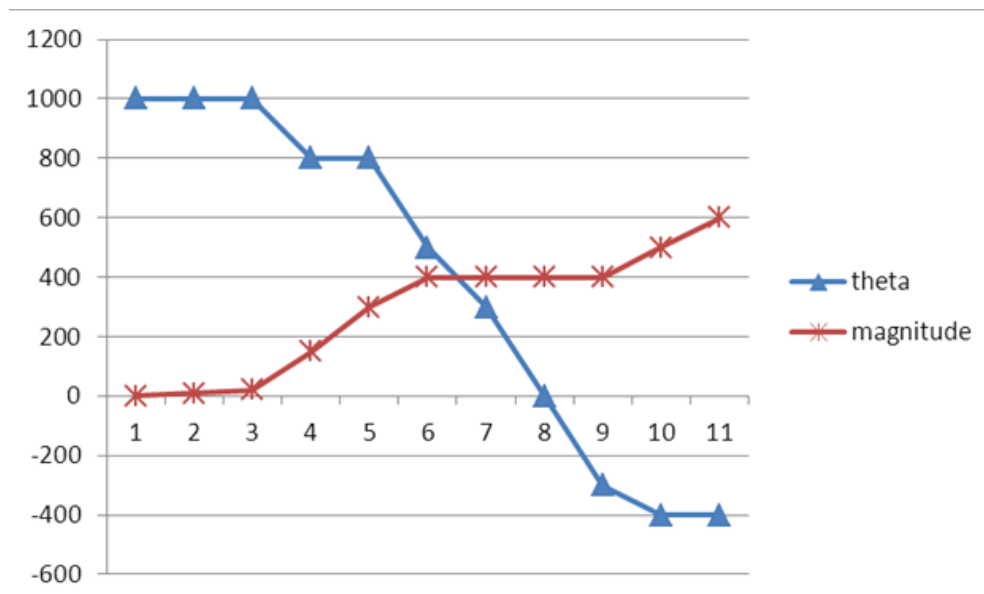
We add harmonious wave compensation. The torque will become stand.

Figure 13. Add compensation phase current



But the motor run at different current. It has different harmonious waveform. So we need add the different phase angle and magnitude. As follow figure.

Figure 14. Relationship Between current and angle

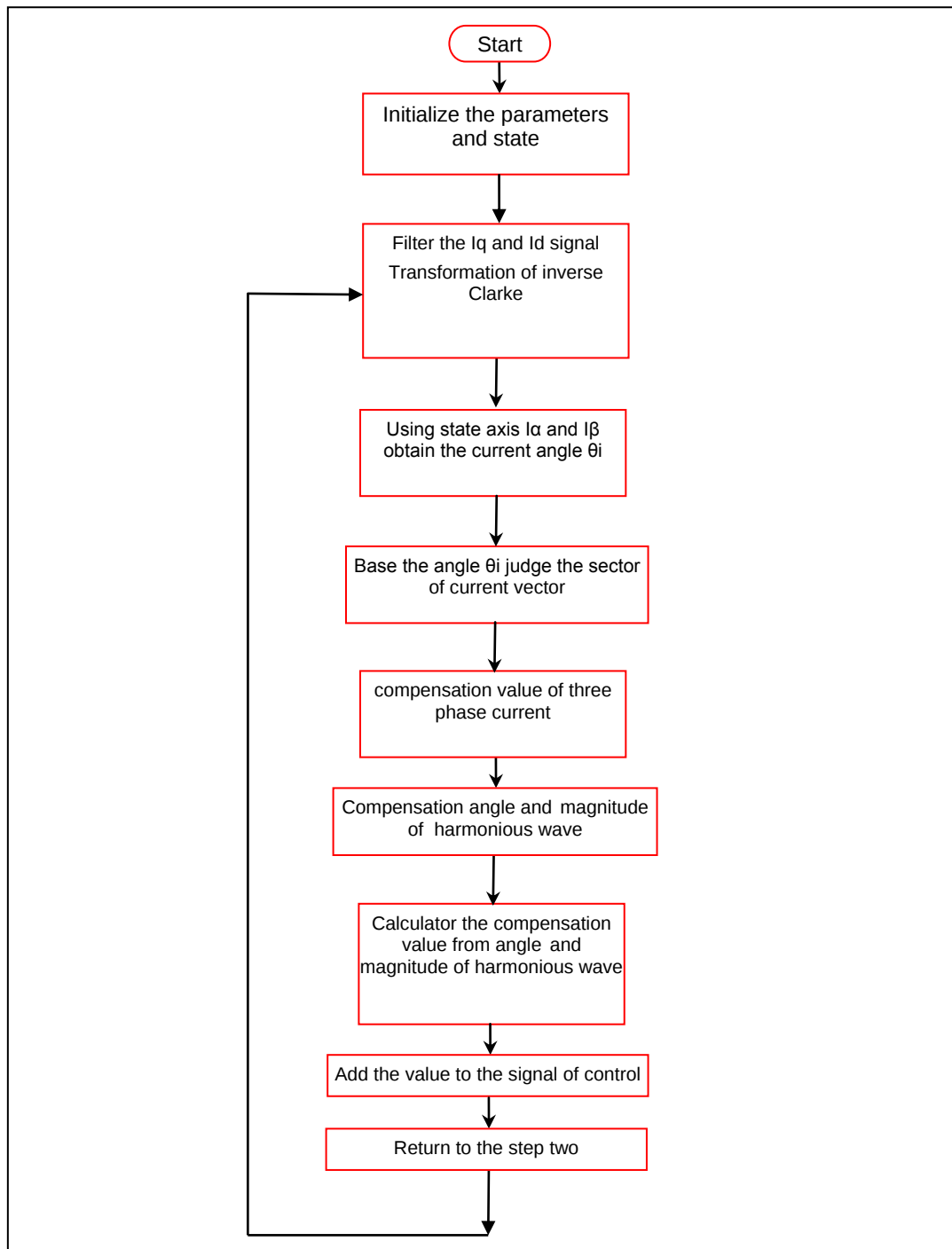


3.2 Dead-time compensation Software implementation

3.2.1 Software Flowchart

By up explain to write the flow chart as below.

Figure 15. Dead-time Compensation Software Flowchart



Algorithm flow explanation:

1. Initialize the parameters and state.
2. Filter the I_q and I_d signal, using filter signal to transformation of inverse Clarke.

Using state axis I_α and I_β obtain the current angle θ_i ;

Base the angle θ_i judge the sector of current vector.

Base the table to compensation value of three phase current;

If the motor is no-sinusoidal wave BEMF, to obtain the compensation angle and magnitude of harmonious wave.

Calculator the compensation value from angle and magnitude of harmonious wave.

Add the value to the signal of control.

Return to the step two.

3.2.2 Software code implement

Function name: DT_Compensation

Description: realize the dead-time compensation

Input: none

Output: none

void DT_Compensation(void)

{

//to realize the dead-time function

DT_Compensation:

```

    PUSH    {R4-R7,LR}
    SUB     SP,SP,#+4
    LDR.N   R0,??DT_Compensation_1
    LDR     R0,[R0, #+0]
    CMP     R0,#+0
    BNE.N   ??DT_Compensation_2
    LDR.N   R5,??DT_Compensation_1+0x4
    LDR.N   R3,??DT_Compensation_1+0x8
    LDR.N   R1,??DT_Compensation_1+0xC
    LDRSH   R0,[R1, R0]
    MOVW    R2,#+2501
    CMP     R0,R2
    ITT     GE
    LDRHGE  R4,[R3, #+20]
    LDRSHGE R6,[R5, #+20]
    BGE.N   ??DT_Compensation_3
    LDRSH   R0,[R1, #+0]
```

```

ASRS    R4,R0,#+8
LDRH    R2,[R3, R4, LSL #+1]
LDRSH   R0,[R5, R4, LSL #+1]
ADD     R3,R3,R4, LSL #+1
LDRH    R3,[R3, #+2]
SUBS    R3,R3,R2
ADD     R4,R5,R4, LSL #+1
LDRSH   R4,[R4, #+2]
SUBS    R5,R0,R4
LDRSH   R1,[R1, #+0]
ADD     R4,R1,R1, LSL #+2
LSLS    R1,R4,#+1
ASRS    R1,R1,#+8
MOVS    R6,#+10
SDIV    R4,R1,R6
ADD     R7,R4,R4, LSL #+2
SUB     R1,R1,R7, LSL #+1
MULS    R3,R3,R1
SDIV    R3,R3,R6
ADDS    R4,R3,R2
MULS    R1,R1,R5
SDIV    R1,R1,R6
SUBS    R6,R0,R1

??DT_Compensation_3:
LDR.N   R5,??DT_Compensation_1+0x10
LDR.N   R7,??DT_Compensation_1+0x14
LDRH    R0,[R7, #+0]
ADD     R1,R0,R0, LSL #+1
ADD     R0,R6,R1, LSL #+1
ADDW    R0,R0,#+1000
UXTH    R0,R0
BL      Sin
MULS    R0,R0,R4
ASRS    R0,R0,#+12
STR     R0,[R5, #+16]
LDRH    R0,[R7, #+0]
ADD     R1,R0,R0, LSL #+1
ADD     R0,R6,R1, LSL #+1
  
```

```

UXTH    R0,R0
BL      Sin
MULS    R0,R0,R4
ASRS    R0,R0,#+12
B.N     ??DT_Compensation_4
??DT_Compensation_2:
LDR.N   R5,??DT_Compensation_1+0x10
MOVS    R0,#+0
STR     R0,[R5, #+16]
??DT_Compensation_4:
STR     R0,[R5, #+12]
LDR.N   R0,??DT_Compensation_1+0x18
LDR.N   R1,??DT_Compensation_1+0x1C
LDR.N   R2,??DT_Compensation_1+0x20
LDR     R3,[R2, #+0]
LDRSH   R4,[R1, #+0]
MULS    R3,R3,R4
STR     R3,[R0, #+0]
LDR.N   R3,??DT_Compensation_1+0x24
LDR.N   R6,??DT_Compensation_1+0x28
LDR.N   R4,??DT_Compensation_1+0x2C
LDR     R5,[R4, #+0]
LDRSH   R7,[R6, #+0]
MULS    R5,R5,R7
STR     R5,[R3, #+0]
LDR     R5,[R0, #+0]
LDR     R7,[R3, #+0]
SUBS    R5,R5,R7
ASRS    R5,R5,#+8
LDR     R2,[R2, #+0]
LDRSH   R6,[R6, #+0]
MULS    R2,R2,R6
STR     R2,[R0, #+0]
LDR     R2,[R4, #+0]
LDRSH   R1,[R1, #+0]
MULS    R1,R1,R2
STR     R1,[R3, #+0]
LDR     R0,[R0, #+0]
  
```

```

LDR    R1,[R3, #+0]
ADDS   R0,R1,R0
ASRS   R0,R0,#+1
SDIV   R1,R0,R5
CMP    R5,#+1
BLT.N  ??DT_Compensation_5
CMP    R0,#+1
BLT.N  ??DT_Compensation_6
CMP    R1,#+222
BLT.N  ??DT_Compensation_7
??DT_Compensation_8:
MOVS   R0,#+1
B.N    ??DT_Compensation_9
??DT_Compensation_7:
MOVS   R0,#+6
B.N    ??DT_Compensation_9
??DT_Compensation_6:
CMN    R1,#+221
BGE.N  ??DT_Compensation_10
??DT_Compensation_11:
MOVS   R0,#+4
B.N    ??DT_Compensation_9
??DT_Compensation_10:
MOVS   R0,#+5
B.N    ??DT_Compensation_9
??DT_Compensation_5:
CMP    R0,#+1
BLT.N  ??DT_Compensation_12
CMN    R1,#+221
BLT.N  ??DT_Compensation_8
MOVS   R0,#+2
B.N    ??DT_Compensation_9
??DT_Compensation_12:
CMP    R1,#+222
BGE.N  ??DT_Compensation_11
MOVS   R0,#+3
??DT_Compensation_9:
LDR.N  R5,??DT_Compensation_1+0x10
  
```

```

STRB    R0,[R5, #+0]
LDR.N   R0,??DT_Compensation_1+0x30
LDRH    R0,[R0, #+0]
LSRS    R1,R0,#+1
STR     R1,[R5, #+8]
LDRH    R0,[R5, #+6]
LDRB    R2,[R5, #+0]
SUBS    R2,R2,#+1
CMP     R2,#+5
BHI.N   ??DT_Compensation_13
TBB     [PC, R2]
DATA
??DT_Compensation_0:
DC8     0x3,0x17,0x20,0x2F
DC8     0x3E,0x4D
THUMB
??DT_Compensation_14:
LDRH    R2,[R5, #+2]
ADDS    R2,R1,R2
STRH    R2, [R5, #+2]
LDRH    R3, [R5, #+4]
SUBS    R3, R3, R1
STRH    R3, [R5, #+4]
SUBS    R0, R0, R1
LDR.N   R1,?? DT_Compensation_1+0x34
LDRH    R2, [R1, #+0]
UXTH    R3, R3
CMP     R3, R2
ITT     CS
MOVCS   R2, #+0
STRHCS  R2, [R5, #+4]
??DT_Compensation_15:
LDRH    R1,[R1, #+0]
UXTH    R0, R0
CMP     R0, R1
BCC.N   ??DT_Compensation_13
MOVS    R0, #+0
B.N     ??DT_Compensation_13
  
```

??DT_Compensation_16:

```

LDRH  R2, [R5, #+2]
ADDS  R2, R1, R2
STRH  R2, [R5, #+2]
LDRH  R2, [R5, #+4]
ADDS  R2, R1, R2
STRH  R2, [R5, #+4]
SUBS  R0, R0, R1
LDR.N R1, ??DT_Compensation_1+0x34
B.N   ??DT_Compensation_15

```

??DT_Compensation_17:

```

LDRH  R2, [R5, #+2]
SUBS  R2, R2, R1
STRH  R2, [R5, #+2]
LDRH  R3, [R5, #+4]
ADDS  R3, R1, R3
STRH  R3, [R5, #+4]
SUBS  R0, R0, R1
LDR.N R1, ??DT_Compensation_1+0x34
LDRH  R3, [R1, #+0]
UXTH  R2, R2
CMP   R2, R3
ITT   CS
MOVCS R2, #+0
STRHCS R2, [R5, #+2]
B.N   ??DT_Compensation_15

```

??DT_Compensation_18:

```

LDRH  R2, [R5, #+2]
SUBS  R2, R2, R1
STRH  R2, [R5, #+2]
LDRH  R3, [R5, #+4]
ADDS  R3, R1, R3
STRH  R3, [R5, #+4]
ADDS  R0, R1, R0
LDR.N R1, ??DT_Compensation_1+0x34
LDRH  R1, [R1, #+0]
UXTH  R2, R2
CMP   R2, R1

```

```

    BCC.N  ??DT_Compensation_13
    MOVS   R1,#+0
    STRH   R1,[R5, #+2]
    B.N    ??DT_Compensation_13
??DT_Compensation_19:
    LDRH   R2,[R5, #+2]
    SUBS   R2,R2,R1
    STRH   R2,[R5, #+2]
    LDRH   R3,[R5, #+4]
    SUBS   R3,R3,R1
    STRH   R3,[R5, #+4]
    ADDS   R0,R1,R0
    LDR.N  R1,??DT_Compensation_1+0x34
    LDRH   R4,[R1, #+0]
    UXTH   R2,R2
    CMP    R2,R4
    BCC.N  ??DT_Compensation_20
    MOVS   R2,#+0
    STRH   R2,[R5, #+2]
    B.N    ??DT_Compensation_20
??DT_Compensation_21:
    LDRH   R2,[R5, #+2]
    ADDS   R2,R1,R2
    STRH   R2,[R5, #+2]
    LDRH   R3,[R5, #+4]
    SUBS   R3,R3,R1
    STRH   R3,[R5, #+4]
    ADDS   R0,R1,R0
    LDR.N  R1,??DT_Compensation_1+0x34
??DT_Compensation_20:
    LDRH   R1,[R1, #+0]
    UXTH   R3,R3
    CMP    R3,R1
    ITT    CS
    MOVCS  R1,#+0
    STRHCS R1,[R5, #+4]
??DT_Compensation_13:
    STRH   R0,[R5, #+6]
  
```

```
ADD    SP,SP,#+4
POP    {R4-R7,PC}    ;; return
Nop
DATA
SECTION `.iar_vfe_header`:DATA:REORDER:NOALLOC:NOROOT(2)
SECTION_TYPE SHT_PROGBITS, 0
DATA
DC32 0

SECTION __DLIB_PERTHREAD:DATA:REORDER:NOROOT(0)
SECTION_TYPE SHT_PROGBITS, 0

SECTION __DLIB_PERTHREAD_init:DATA:REORDER:NOROOT(0)
SECTION_TYPE SHT_PROGBITS, 0

END
}
```

4 Dead-time compensation Function Performance

4.1 Basic Verification

4.1.1 Simulator of algorithm

The signal of control adding the compensation of dead-time and no compensation of dead-time compared. As below figure:

Figure 16. three phases current no compensation

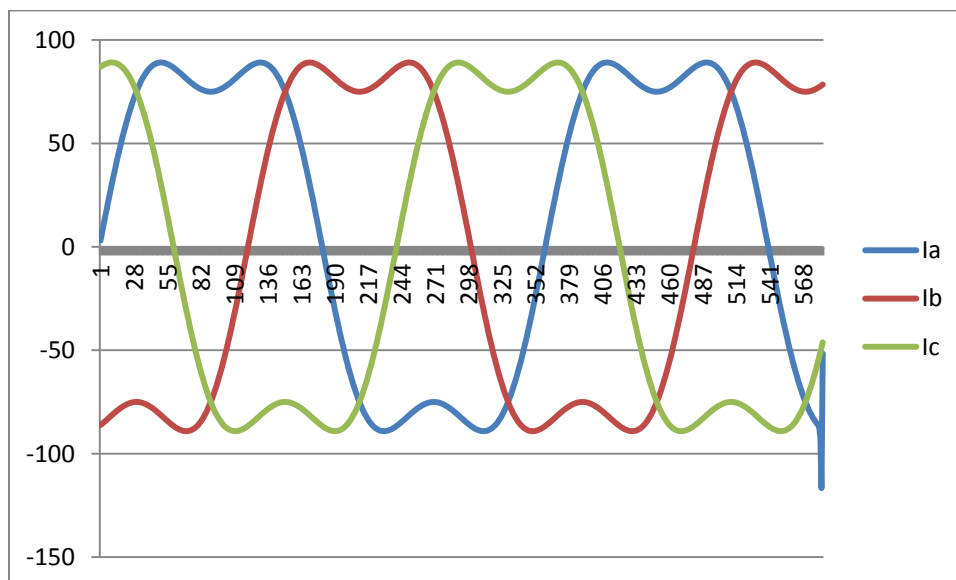


Figure 17. state axis current no compensation

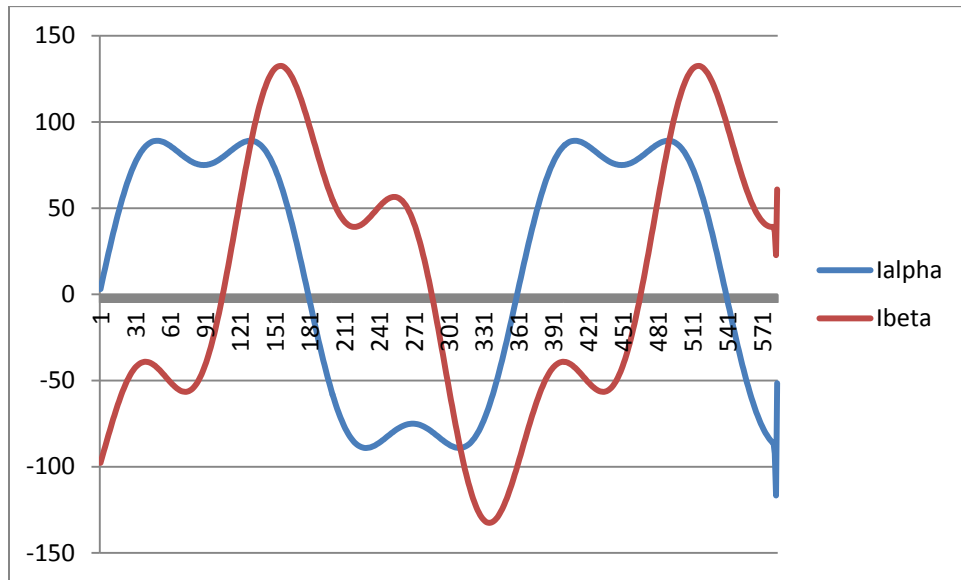


Figure 18. rotor axis current no compensation

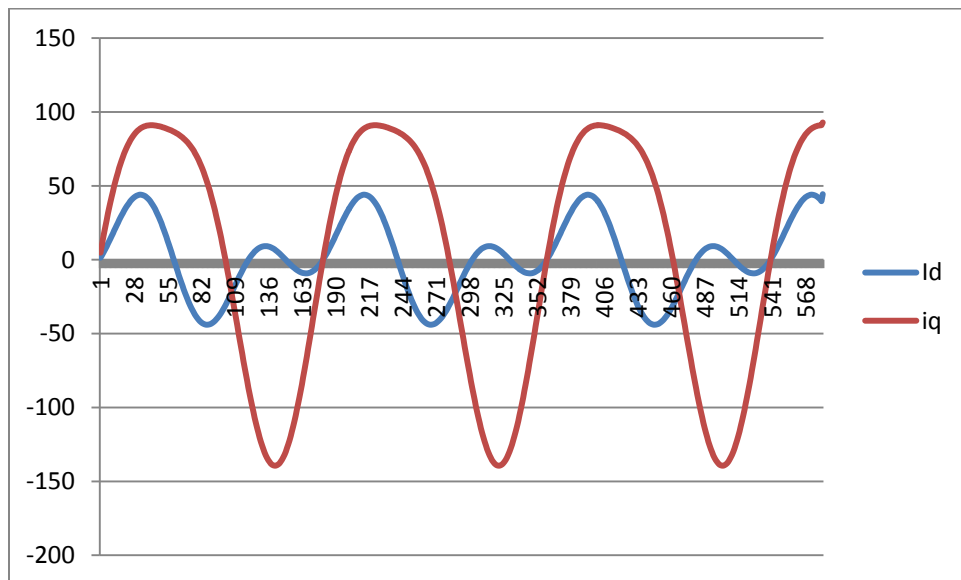


Figure 19. three phases current add compensation

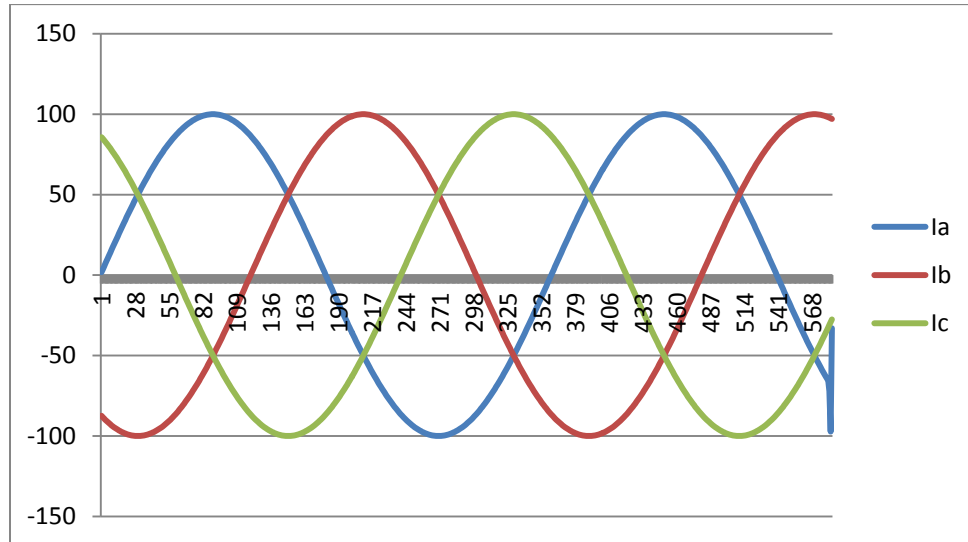


Figure 20. state axis current add compensation

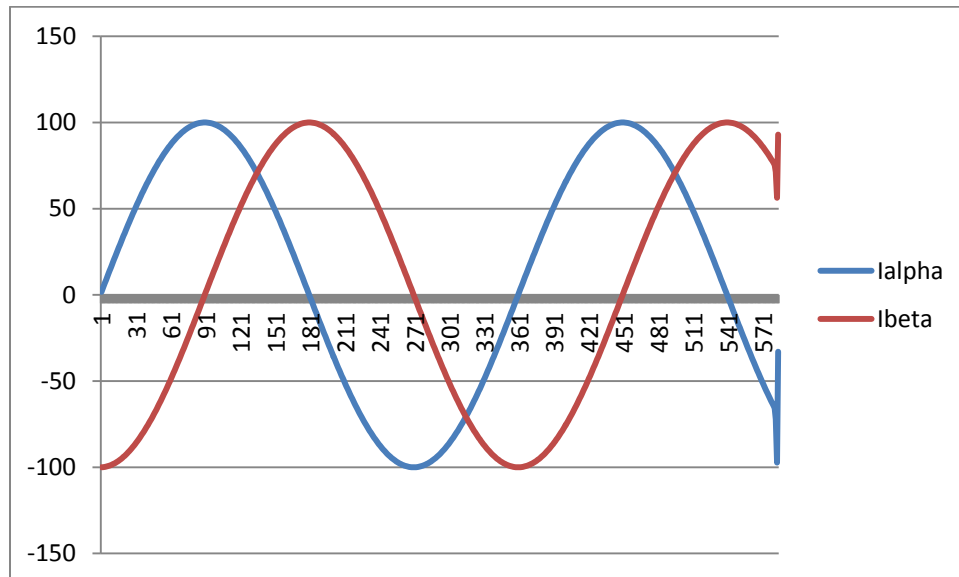
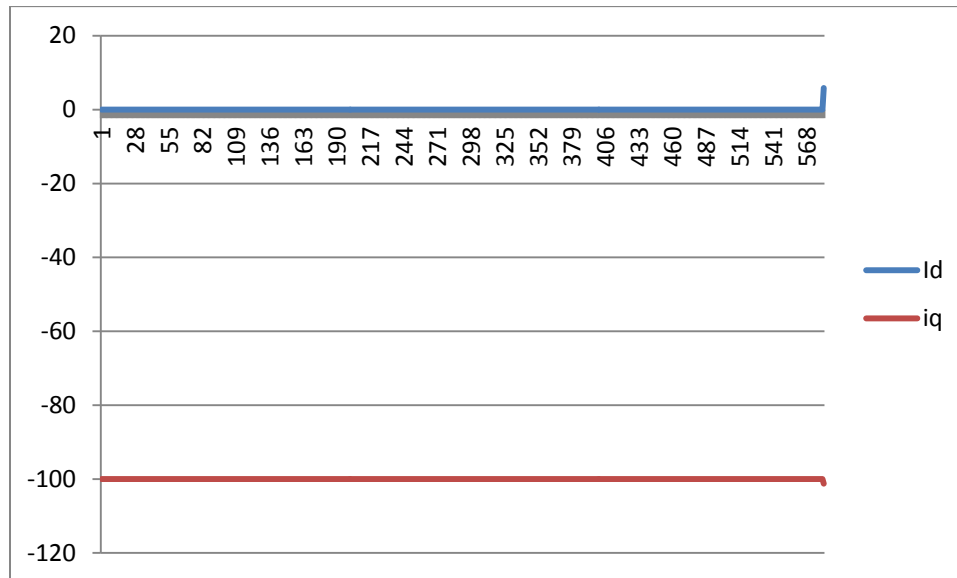


Figure 21. rotor axis current adds compensation



4.1.2 Test waveform of run motor

By the theory of the up expound, realize the algorithm and obtain the perfect performance, as show as below figure no compensation and added compensation waveform:

Figure 22. no dead-time compensation waveform of first motor

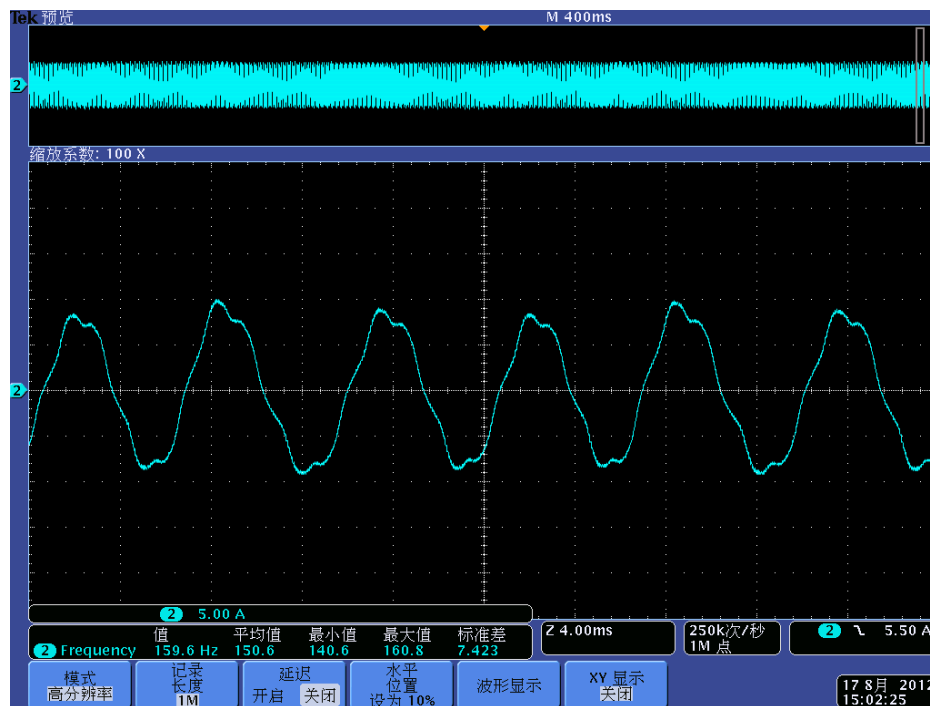


Figure 23. added dead-time compensation of first motor

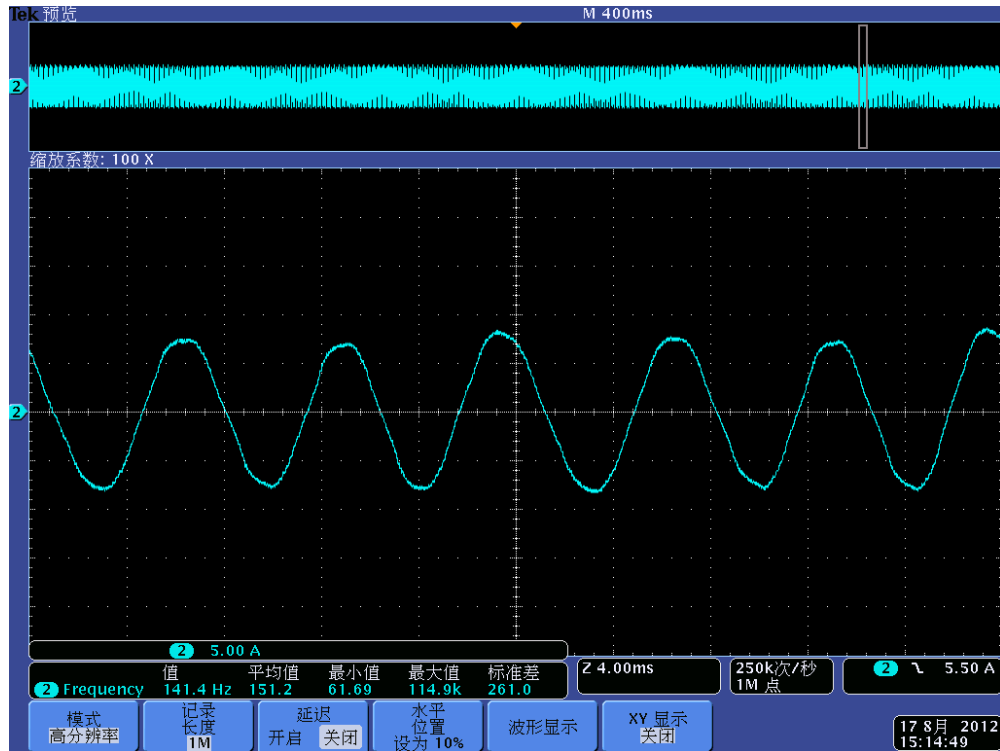


Figure 24. zoom out no dead-time compensation of first motor

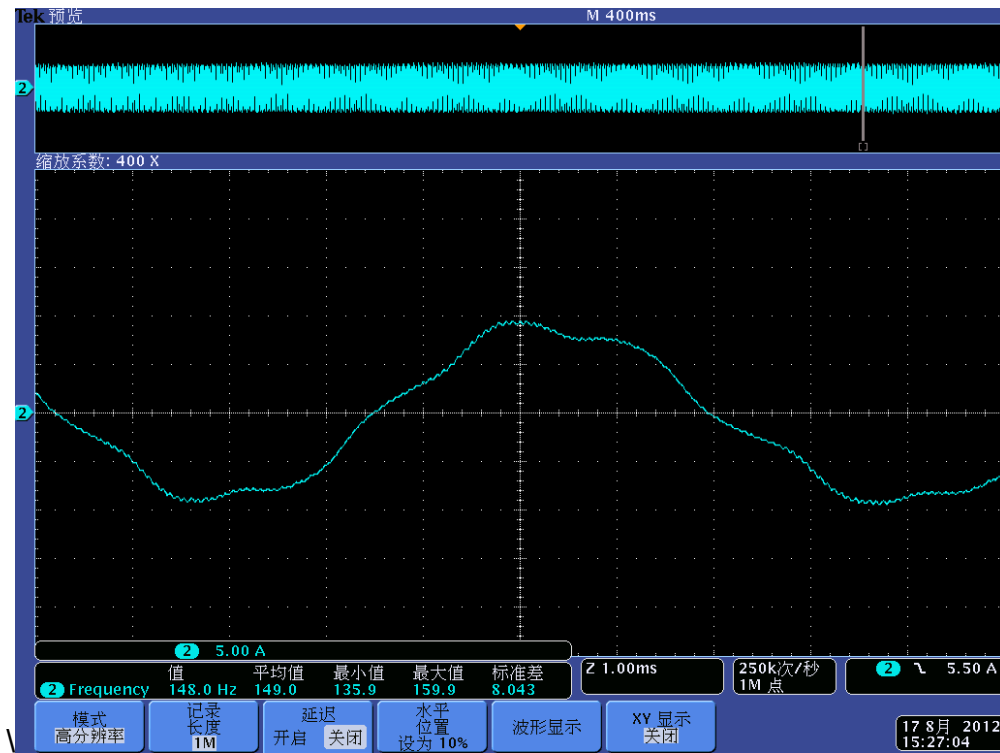


Figure 25. zoom out added dead-time compensation of first motor

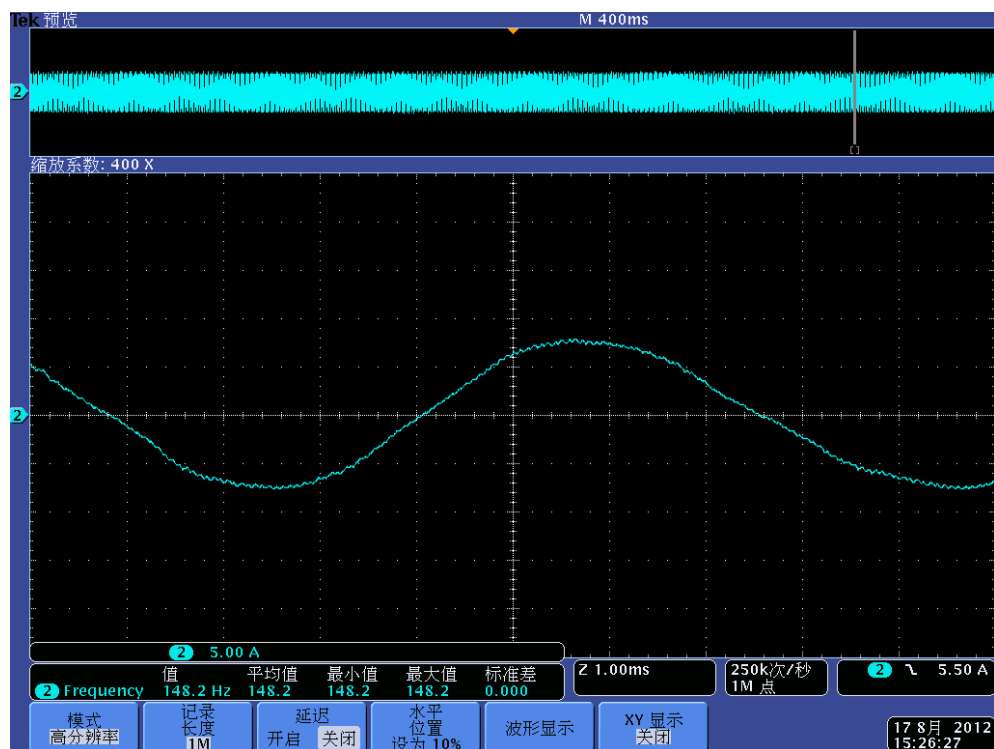


Figure 26. no dead-time compensation waveform of other motor

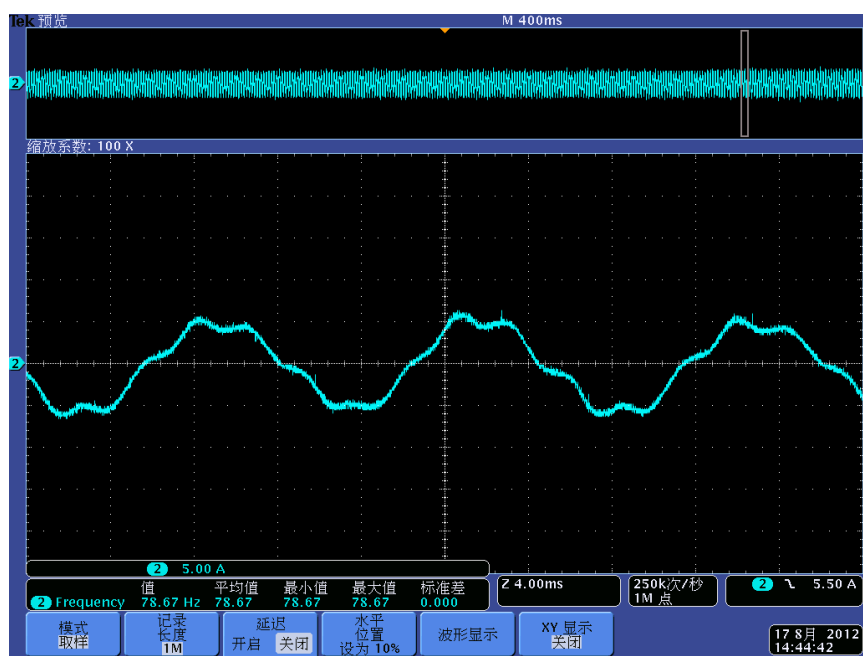


Figure 27. added dead-time compensation waveform of other motor

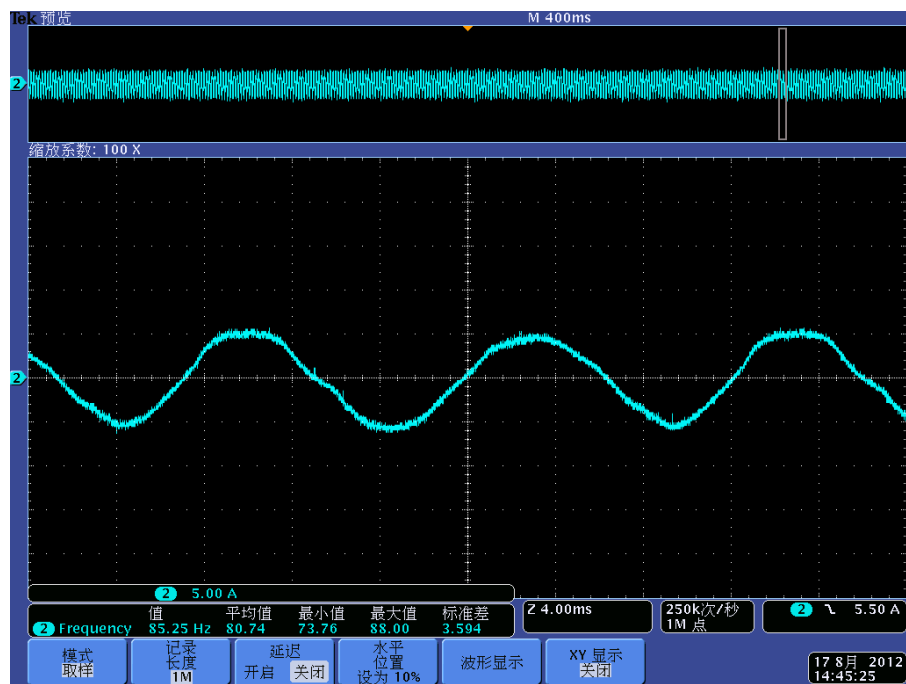


Figure 28. no dead-time compensation waveform of open loop control

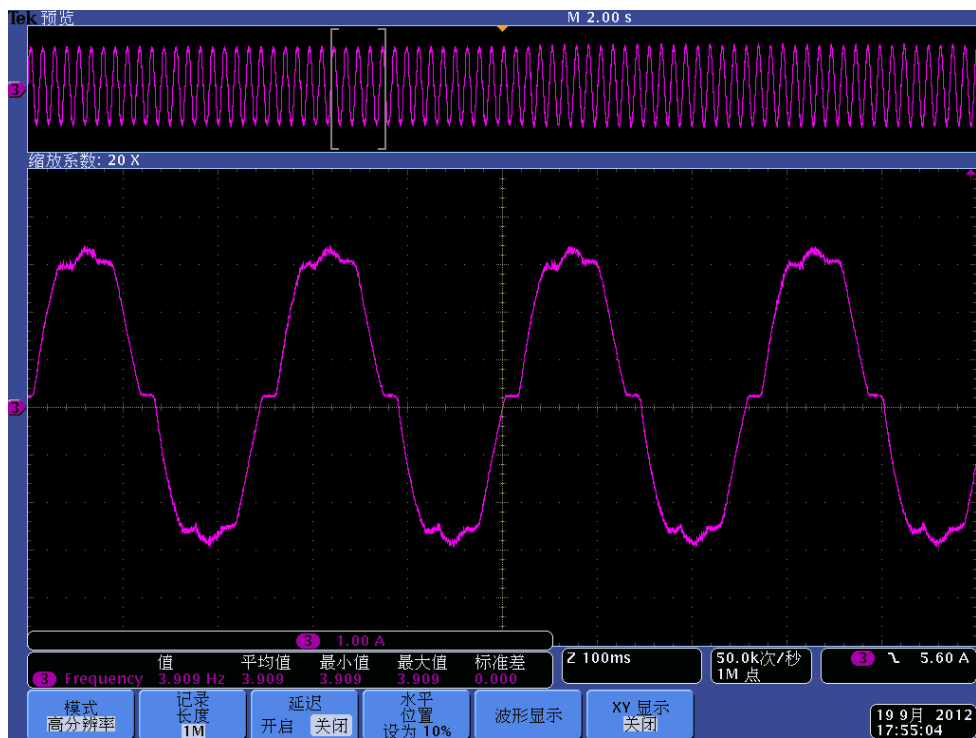


Figure 29. dead-time compensation changing waveform of open loop control

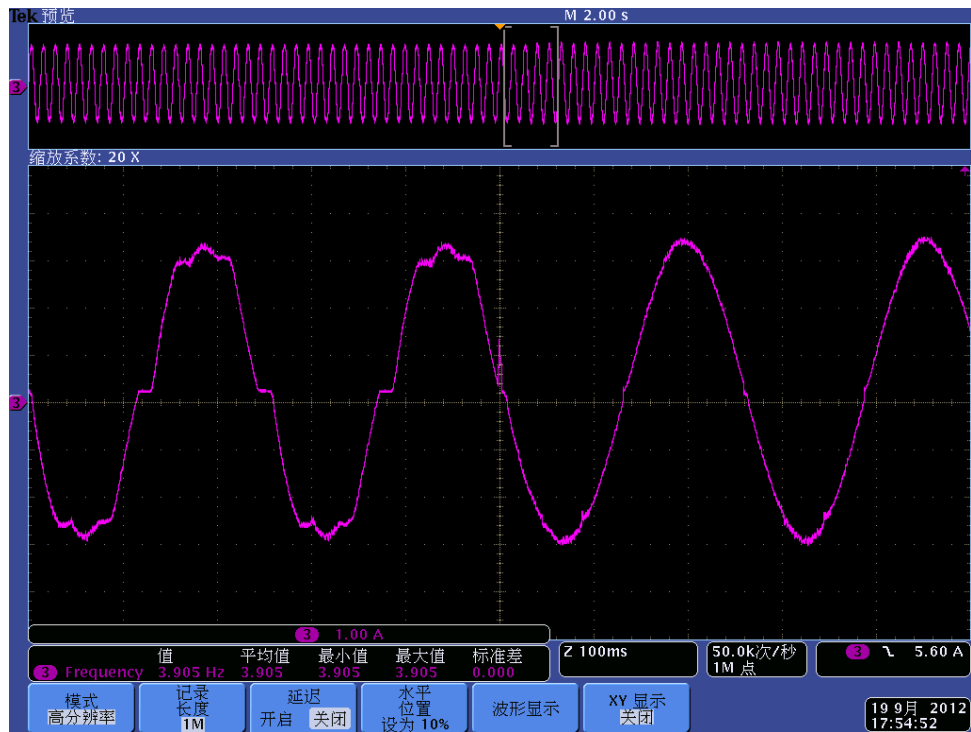
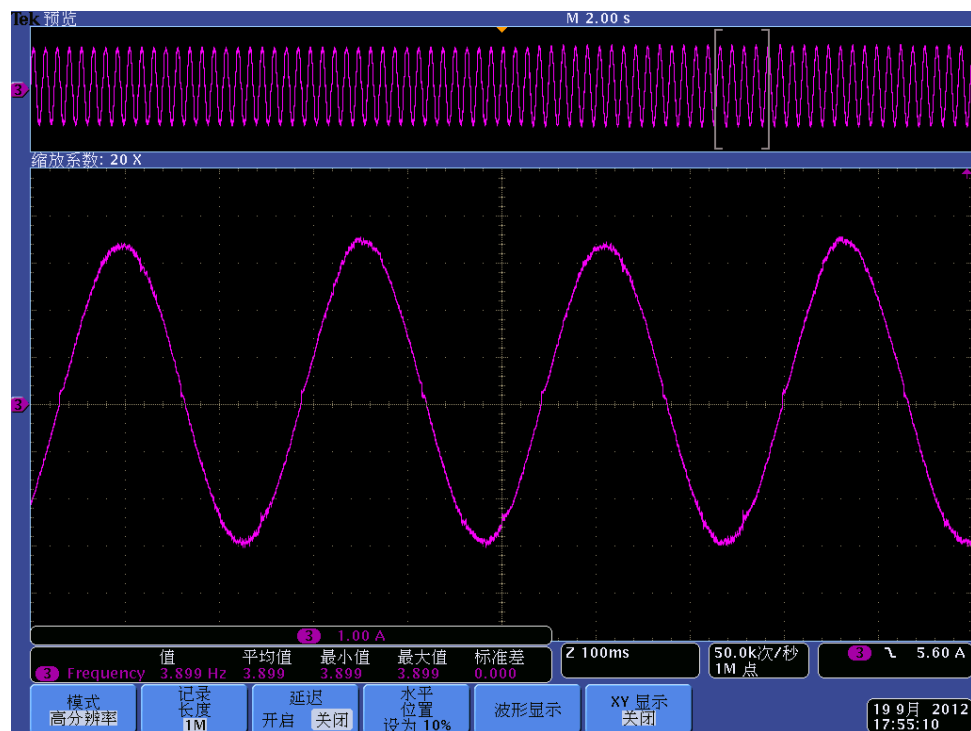


Figure 30. add dead-time compensation waveform of open loop control



From above, dead-time compensation works very well. From the test motors purpose, also decrease the yawp and harmonious wave. So the dead-time compensation theory can carry into execution.

5 Conclusion

As for a vector control system of a permanent magnet synchronous motor, this paper analyzes the influence of dead-time on output voltage in detail, and discusses the relationship between the position of output voltage vector and the direction of three-phase current in a space vector figure. In addition, a method of dividing three-phase current into six regions and a control strategy that compensates only one phase voltage in every region are proposed in the paper. The proposed scheme can acquire the direction of three-phase current judging by the position of output voltage, and it avoids the phenomenon of several zero-crossing in current sampling. At last, experiments which are uncompensated and compensated are carried out with the Fujitsu MB9Bxxxx/MB9Axxxx Series, and the results verify this method has a good compensation effect.

6 Document History

Document Title: AN205406 - Dead Time Compensation implementation in MB9Bxxxx/MB9Axxxx Series

Document Number: 002-05406

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	—	FCZH	08/27/2012	Initial release
			09/15/2012	Added more test waveform
*A	5042015	FCZH	12/17/2015	Converted Spansion Application Note "MCU-AN-510122-E-11" to Cypress format
*B	5807441	AESATMP8	07/13/2017	Updated logo and Copyright.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at [Cypress Locations](#).

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#) | [PSoC 6](#)

Cypress Developer Community

[Forums](#) | [WICED IOT Forums](#) | [Projects](#) | [Videos](#) | [Blogs](#) | [Training](#) | [Components](#)

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.



Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2012-2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.