

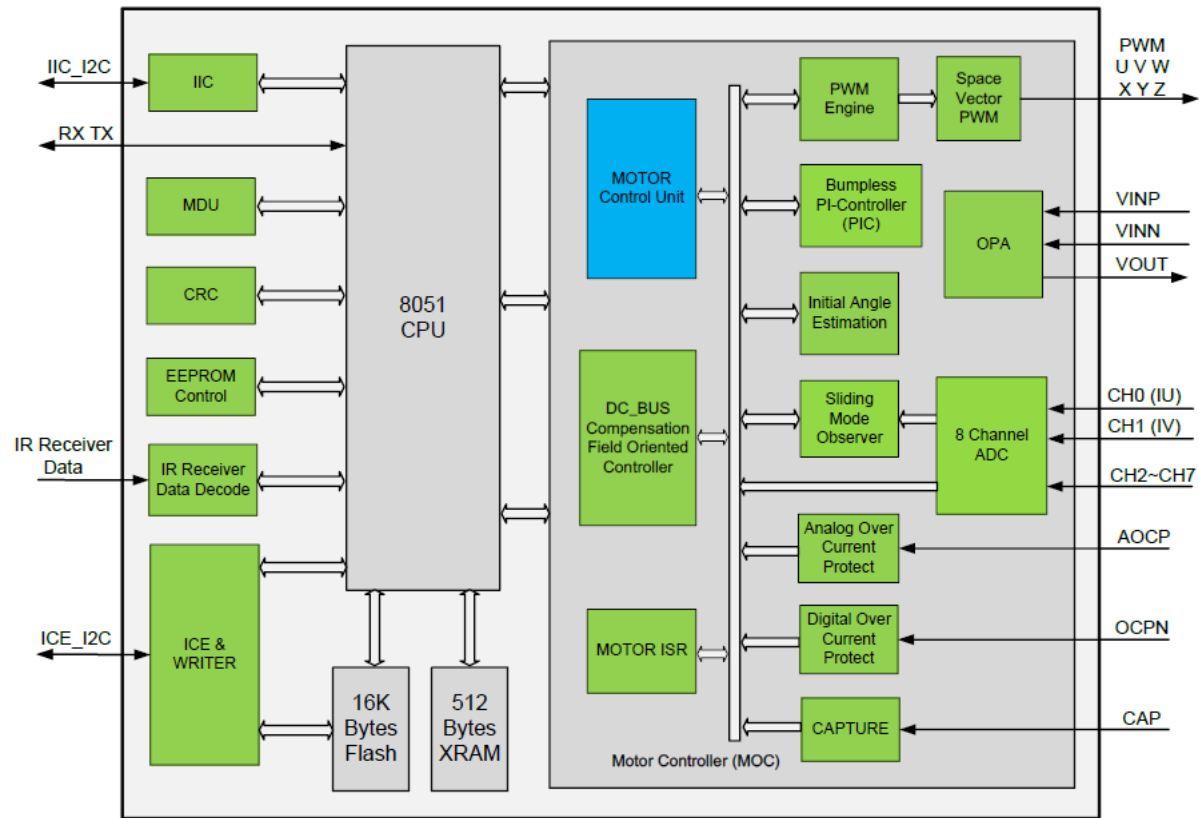
MDSF40

Sample Code Sensorless Introduce

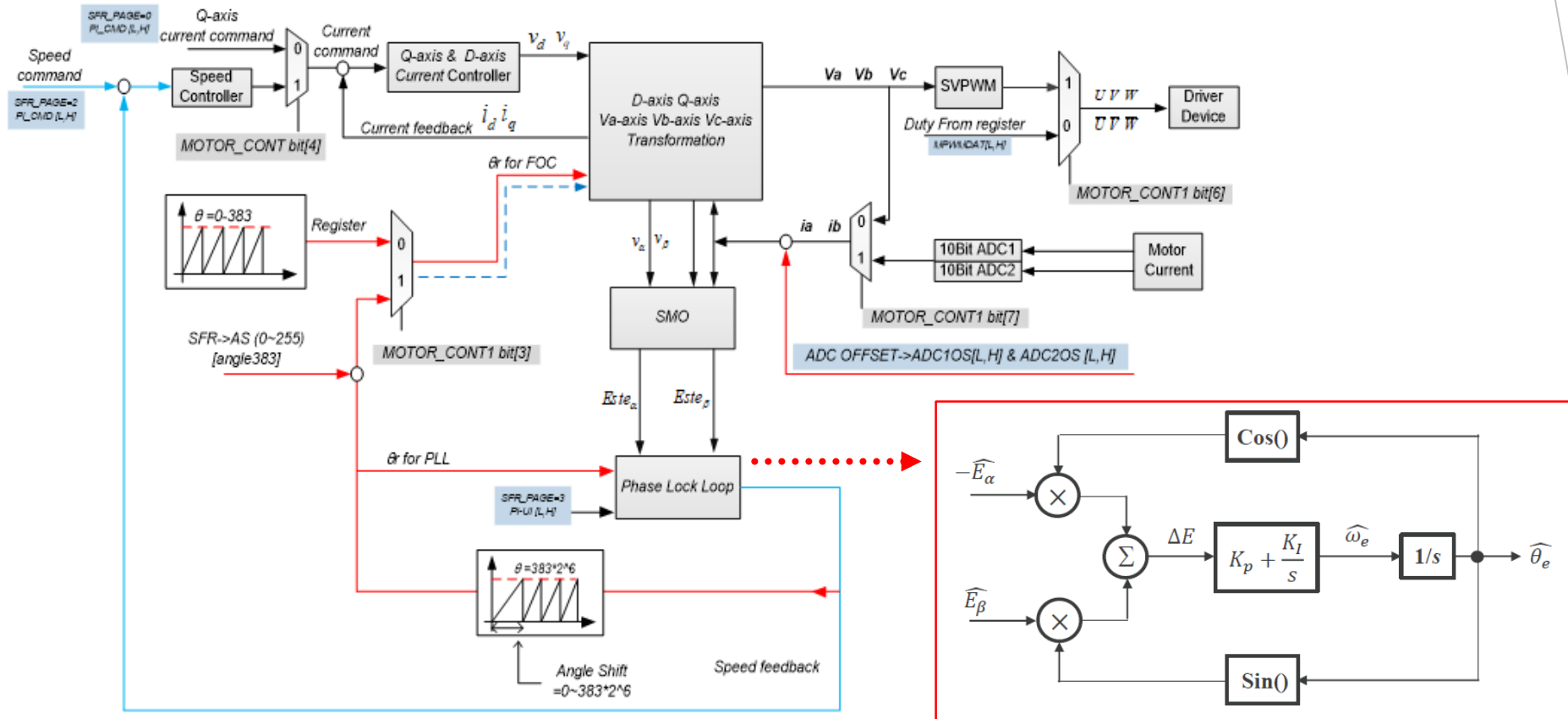


MDSF架構

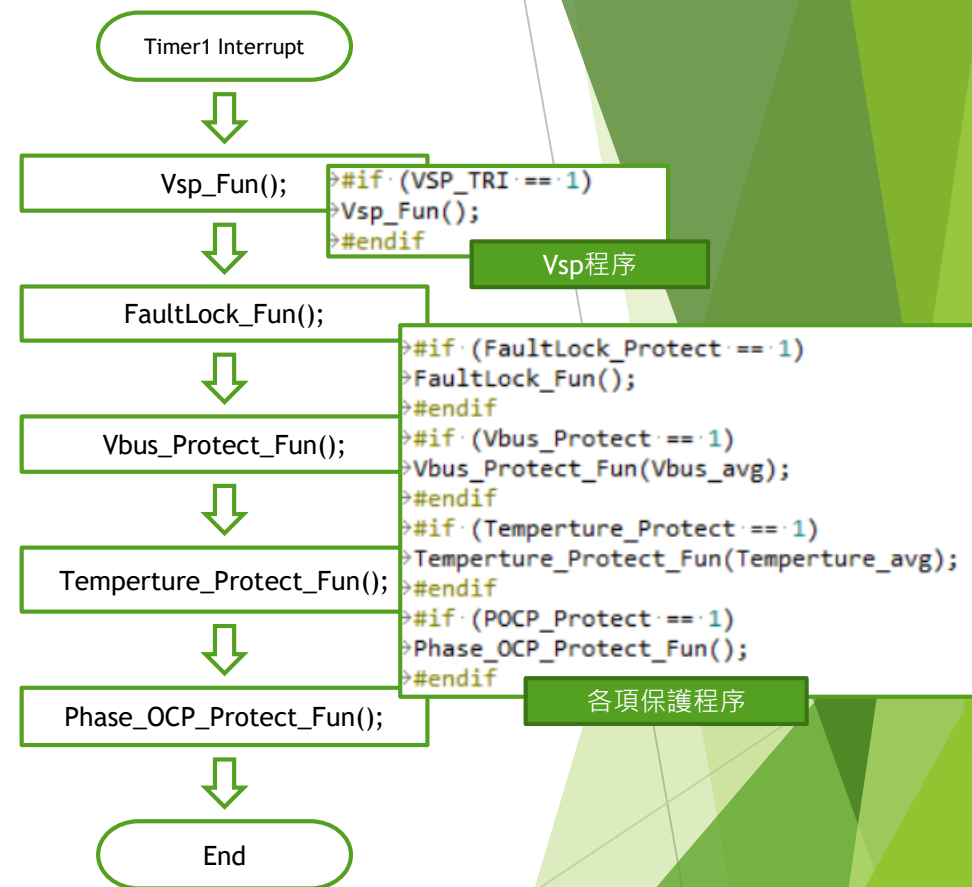
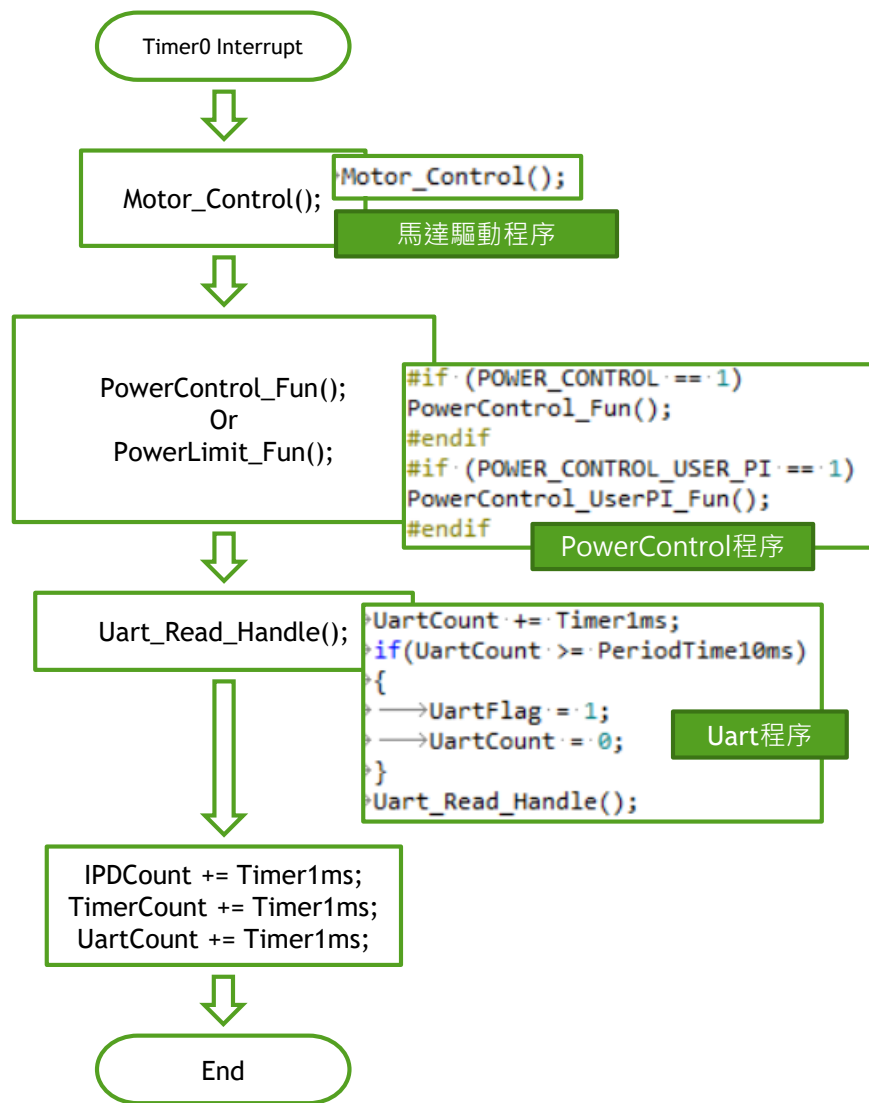
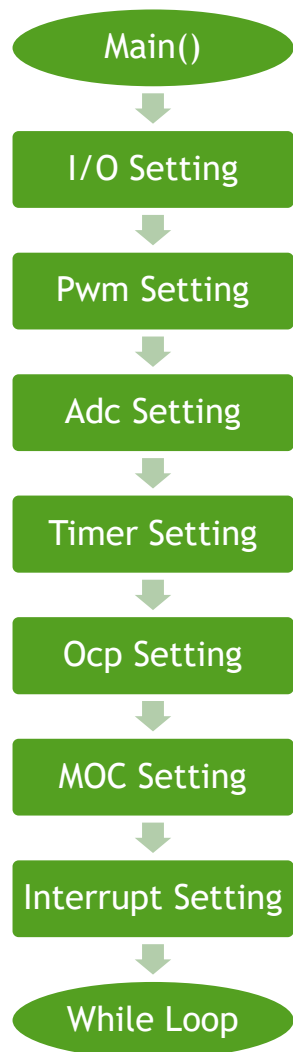
MDSF架構



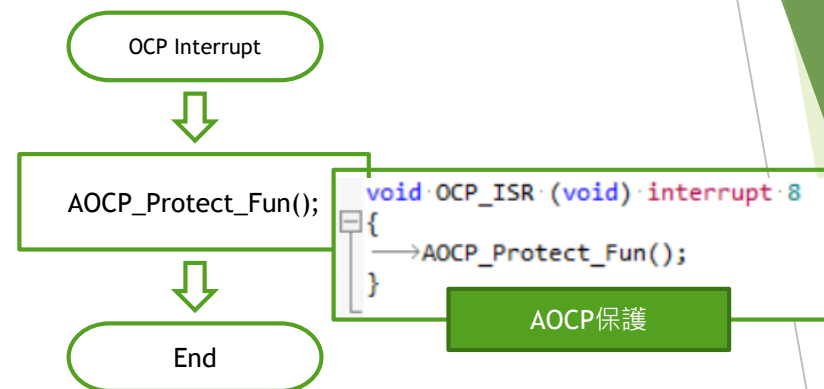
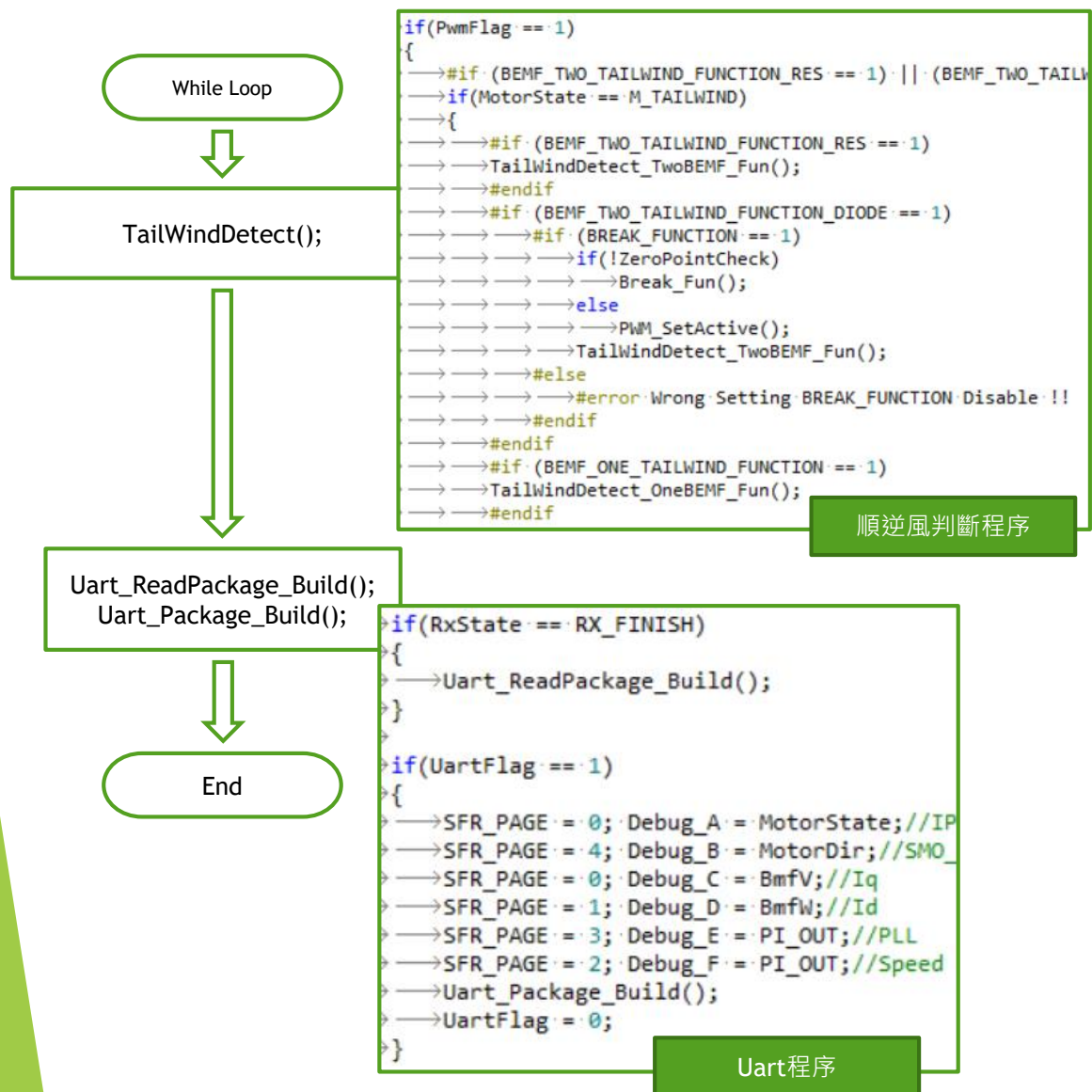
MOC架構



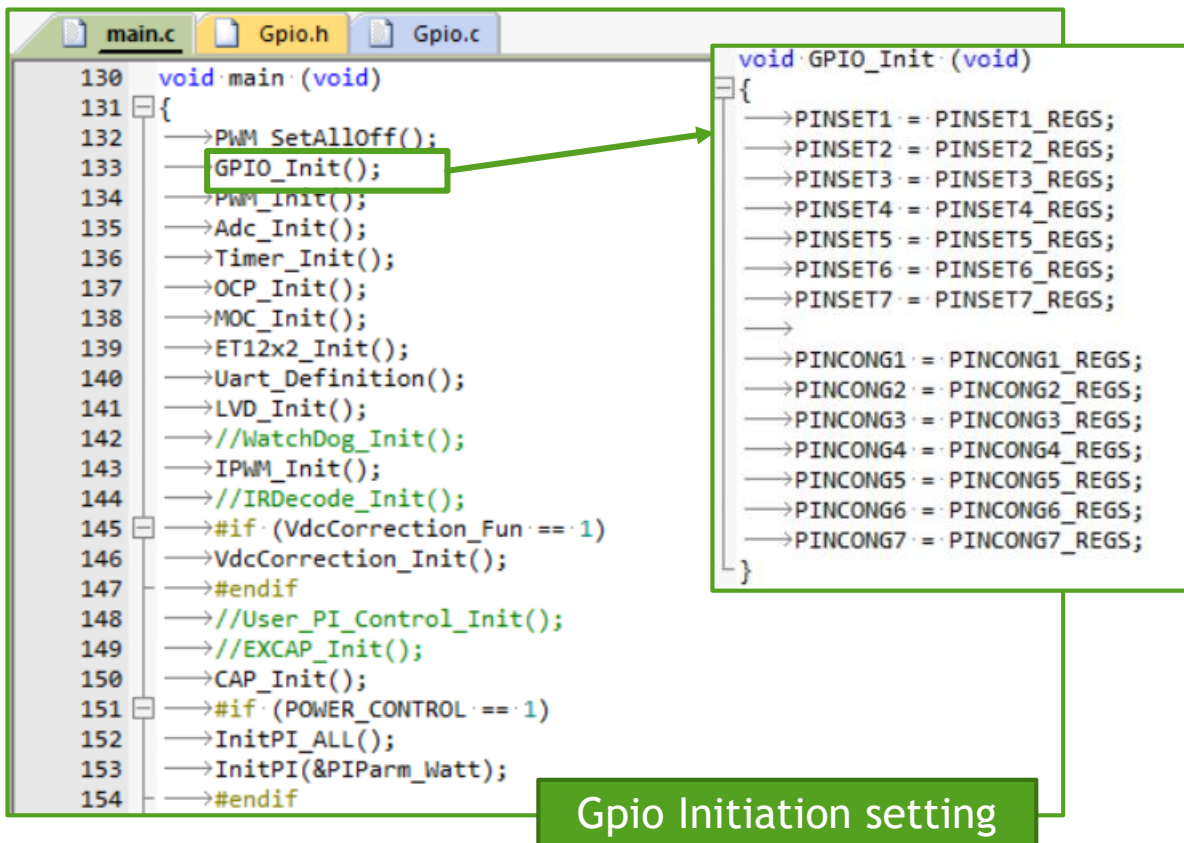
程式流程圖



程式流程圖



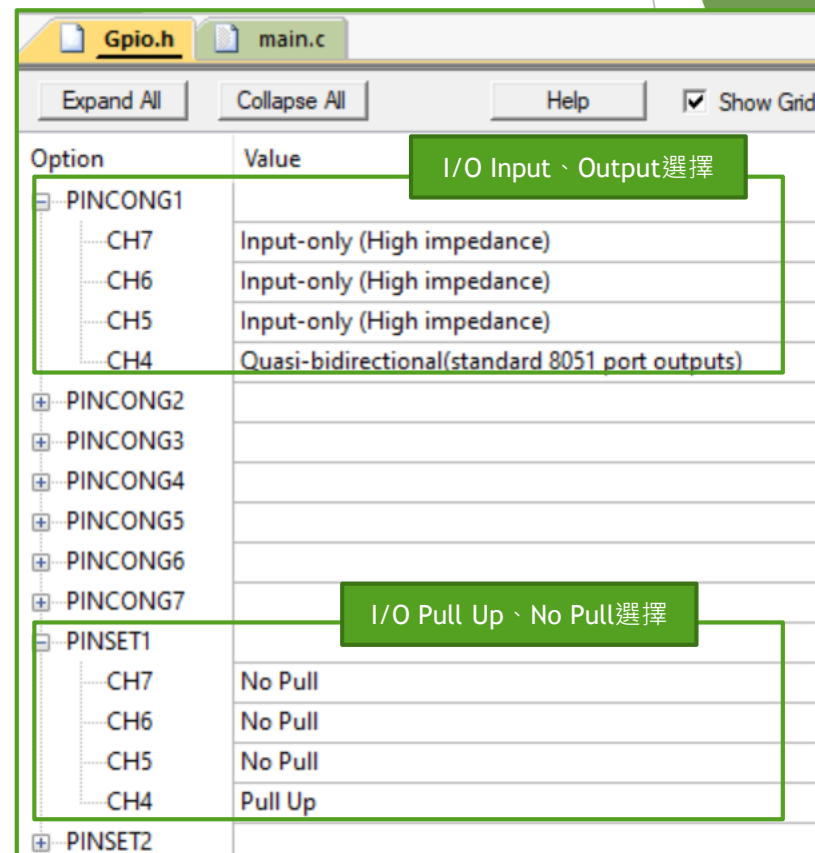
I/O Initiation setting



```
130 void main(void)
131 {
132     →PWM_SetAllOff();
133     →GPIO_Init();
134     →PWM_Init();
135     →Adc_Init();
136     →Timer_Init();
137     →OCP_Init();
138     →MOC_Init();
139     →ET12x2_Init();
140     →Uart_Definition();
141     →LVD_Init();
142     →//WatchDog_Init();
143     →IPWM_Init();
144     →//IRDecode_Init();
145     →#if (VdcCorrection_Fun == 1)
146     →VdcCorrection_Init();
147     →#endif
148     →//User_PI_Control_Init();
149     →//EXCAP_Init();
150     →CAP_Init();
151     →#if (POWER_CONTROL == 1)
152     →InitPI_ALL();
153     →InitPI(&PIParm_Watt);
154     →#endif
}
```

```
void GPIO_Init(void)
{
    →PINSET1 = PINSET1_REGS;
    →PINSET2 = PINSET2_REGS;
    →PINSET3 = PINSET3_REGS;
    →PINSET4 = PINSET4_REGS;
    →PINSET5 = PINSET5_REGS;
    →PINSET6 = PINSET6_REGS;
    →PINSET7 = PINSET7_REGS;
    →
    →PINCONG1 = PINCONG1_REGS;
    →PINCONG2 = PINCONG2_REGS;
    →PINCONG3 = PINCONG3_REGS;
    →PINCONG4 = PINCONG4_REGS;
    →PINCONG5 = PINCONG5_REGS;
    →PINCONG6 = PINCONG6_REGS;
    →PINCONG7 = PINCONG7_REGS;
}
```

Gpio Initiation setting



Option	Value
PINCONG1	
CH7	Input-only (High impedance)
CH6	Input-only (High impedance)
CH5	Input-only (High impedance)
CH4	Quasi-bidirectional(standard 8051 port outputs)
PINCONG2	
PINCONG3	
PINCONG4	
PINCONG5	
PINCONG6	
PINCONG7	
PINSET1	
CH7	No Pull
CH6	No Pull
CH5	No Pull
CH4	Pull Up
PINSET2	

I/O Input、Output選擇

I/O Pull Up、No Pull選擇



Pwm Initiation setting

```
main.c  Gpio.h  Gpio.c
130 void main(void)
131 {
132     →PWM_SetAllOff();
133     →GPIO_Init();
134     →PWM_Init();
135     →Adc_Init();
136     →Timer_Init();
137     →OCP_Init();
138     →MOC_Init();
139     →ET12x2_Init();
140     →Uart_Definition();
141     →LVD_Init();
142     →//WatchDog_Init();
143     →IPWM_Init();
144     →//IRDecode_Init();
145     →#if (VdcCorrection_Fun == 1)
146     →VdcCorrection_Init();
147     →#endif
148     →//User_PI_Control_Init();
149     →//EXCAP_Init();
150     →CAP_Init();
151     →#if (POWER_CONTROL == 1)
152     →InitPI_ALL();
153     →InitPI(&PIParm_Watt);
154     →#endif
```

```
void PWM_Init(void)
{
    →SFR_PAGE = 0;
    →MPWMDATA = MPWMDATA_REGS;
    →SYNC = 0x55;
    →MPWMINV = MPWMINV_REGS;
    →SYNC = 0x55;
    →MPWMDB = MPWMDB_REGS;
    →SYNC = 0x55;
    →PWM_SetAllOff();
}
```

Pwm Initiation setting

注意PWM SWAP配置，需要設定正確!!

Pwm.h	
Expand All	Collapse All
Help	Show Grid
Option	Value
Set MPWM SWAP	MDSF40
Set MPWMDATA	
Set PWM Frequency (unit : Hz)	27000
Set MPWMINV	
U INV	Non-Inverse
X INV	Non-Inverse
V INV	Non-Inverse
Y INV	Non-Inverse
W INV	Non-Inverse
Z INV	Non-Inverse
Set MPWMDB	
Deadband Time	Deadband Time 1.5us
BASE_RPM (unit : rpm)	32767

BASE_RPM : 代表PLL_OUT 32767對應的實際轉速的標么值



Adc Initiation setting

```

main.c  Gpio.h  Gpio.c
130 void main(void)
131 {
132     →PWM_SetAlloff();
133     →GPIO_Init();
134     →PWM_Init();
135     →Adc_Init();
136     →Timer_Init();
137     →OCP_Init();
138     →MOC_Init();
139     →ET12x2_Init();
140     →Uart_Definition();
141     →LVD_Init();
142     →//WatchDog_Init();
143     →IPWM_Init();
144     →//IRDecode_Init();
145     →#if (VdcCorrection_Fun == 1)
146     →VdcCorrection_Init();
147     →#endif
148     →//User_PI_Control_Init();
149     →//EXCAP_Init();
150     →CAP_Init();
151     →#if (POWER_CONTROL == 1)
152     →InitPI_ALL();
153     →InitPI(&PIParm_Watt);
154     →#endif
    
```

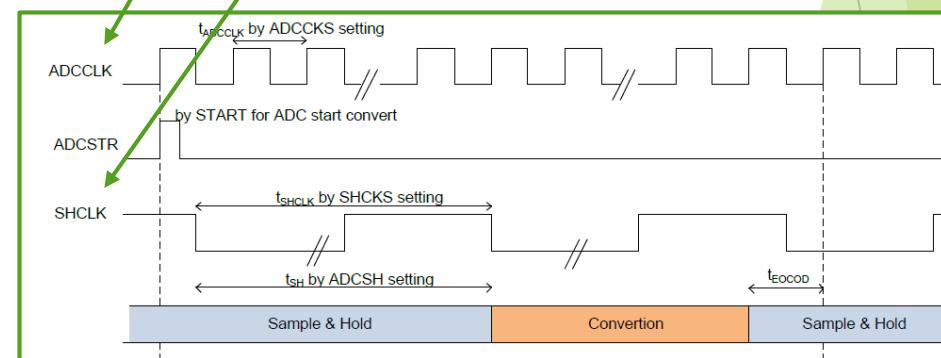
Adc Initiation setting

```

void Adc_Init(void)
{
    →ADCCONT = ADCCONT_REGS;
    →ADCSTR = ADCSTR_REGS | OPA_GAIN_REGS;
    →
    →SFR_PAGE = 0;
    →ADCOFST = 512; //ADCOFST_Init: 512
    →SFR_PAGE = 1;
    →ADCOFST = 512; //ADCOFST_Init: 512;
}
    
```

AdcCKS、SHCKS 設定

Option	Value
ADCCONT	
ADCCH	CH0
ADCCKS	24MHz
ADCDS	ADCD2 LSB
ADCSH	1 clock
ADCPD	Normal
ADCSTR	
SHCKS	6MHz



Timer Initiation setting

The image displays a code editor on the left and a peripheral configuration tool on the right. In the code editor, the `main.c` file is open, showing a `main` function. A green box highlights the `Timer_Init();` call on line 136, with an arrow pointing to the `Timer_Init` function definition in `Timer.h`. The `Timer_Init` function initializes various timer registers, including `PFCON`, `TMOD`, `TH0`, `TL0`, `TR0`, `TH1`, `TL1`, `TR1`, `T2CON`, `TH2`, `TL2`, and `TR2`.

The peripheral configuration tool on the right shows the configuration for the timer. The `Timer.h` file is selected. The configuration table is as follows:

Item	Value
PFCON	
TOPS	F_PER/12 (2MHz)
T1PS	F_PER/12 (2MHz)
SRELPS	F_PER/64
TMOD	
T0 Mode	16-bit Counter/Timer (Not auto-reload)
C/T0	Timer
GATE0	--
T1 Mode	16-bit Counter/Timer (Not auto-reload)
C/T1	Timer
GATE1	--
T2CON	
T2PS	F_PER/12 (2MHz)
T2 Mode	16-bit Counter/Timer (Not auto-reload)
TIMER0	<input checked="" type="checkbox"/>
Interrupt TIMER0_FREQ (unit : Hz)	1000
TIMER1	<input checked="" type="checkbox"/>
Interrupt TIMER1_FREQ (unit : Hz)	100
TIMER2	<input type="checkbox"/>
IT0	1 : External interrupt is activated at falling edge on input pin
IT1	1 : External interrupt is activated at falling edge on input pin

A green box highlights the `Interrupt TIMER0_FREQ (unit : Hz)` setting, which is set to 1000. A green label "Timer 中斷頻率設定" (Timer Interrupt Frequency Setting) points to this value.

Timer Initiation setting

OCP Initiation setting

```

130 void main(void)
131 {
132     →PWM_SetAllOff();
133     →GPIO_Init();
134     →PWM_Init();
135     →Adc_Init();
136     →Timer_Init();
137     →OCP_Init();
138     →MOC_Init();
139     →ET12x2_Init();
140     →Uart_Definition();
141     →LVD_Init();
142     →//WatchDog_Init();
143     →IPWM_Init();
144     →//IRDecode_Init();
145     →#if (VdcCorrection_Fun == 1)
146     →VdcCorrection_Init();
147     →#endif
148     →//User_PI_Control_Init();
149     →//EXCAP_Init();
150     →CAP_Init();
151     →#if (POWER_CONTROL == 1)
152     →InitPI_ALL();
153     →InitPI(&PIParam Watt);

```

```

void OCP_Init(void)
{
    →AOCPCONT = AOCPCONT_REGS;
    →OCPCONT = OCPCONT_REGS | OCP;
}

```

Option	Value
Set AOCPCONT	
I_SHORT	0.3V
AOCPEN	Enable
DOCPEN	Disable
Set OCPCONT	
OCPMS	Auto Mode
AOCPCONT	0x1F

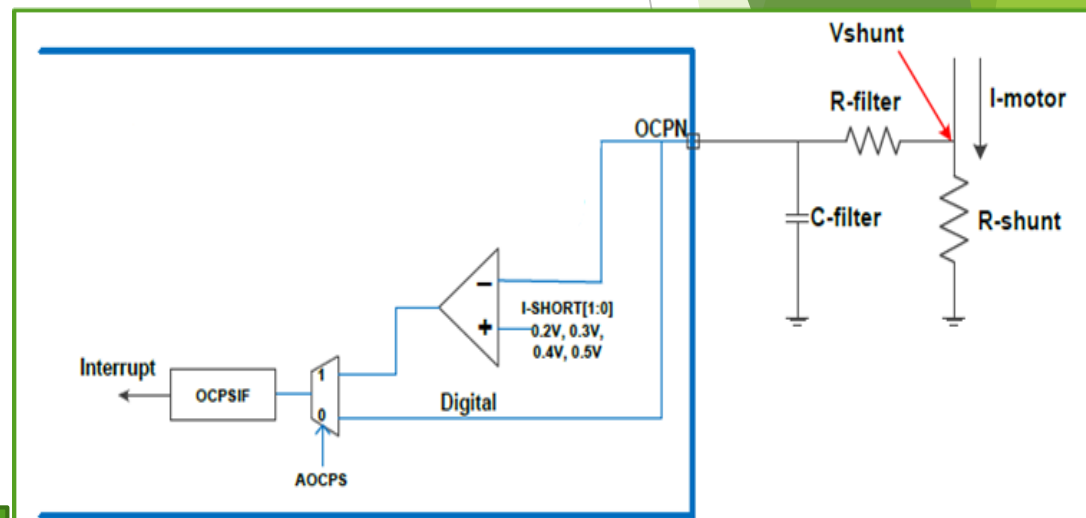
OCP Initiation setting

OCP 保護設定

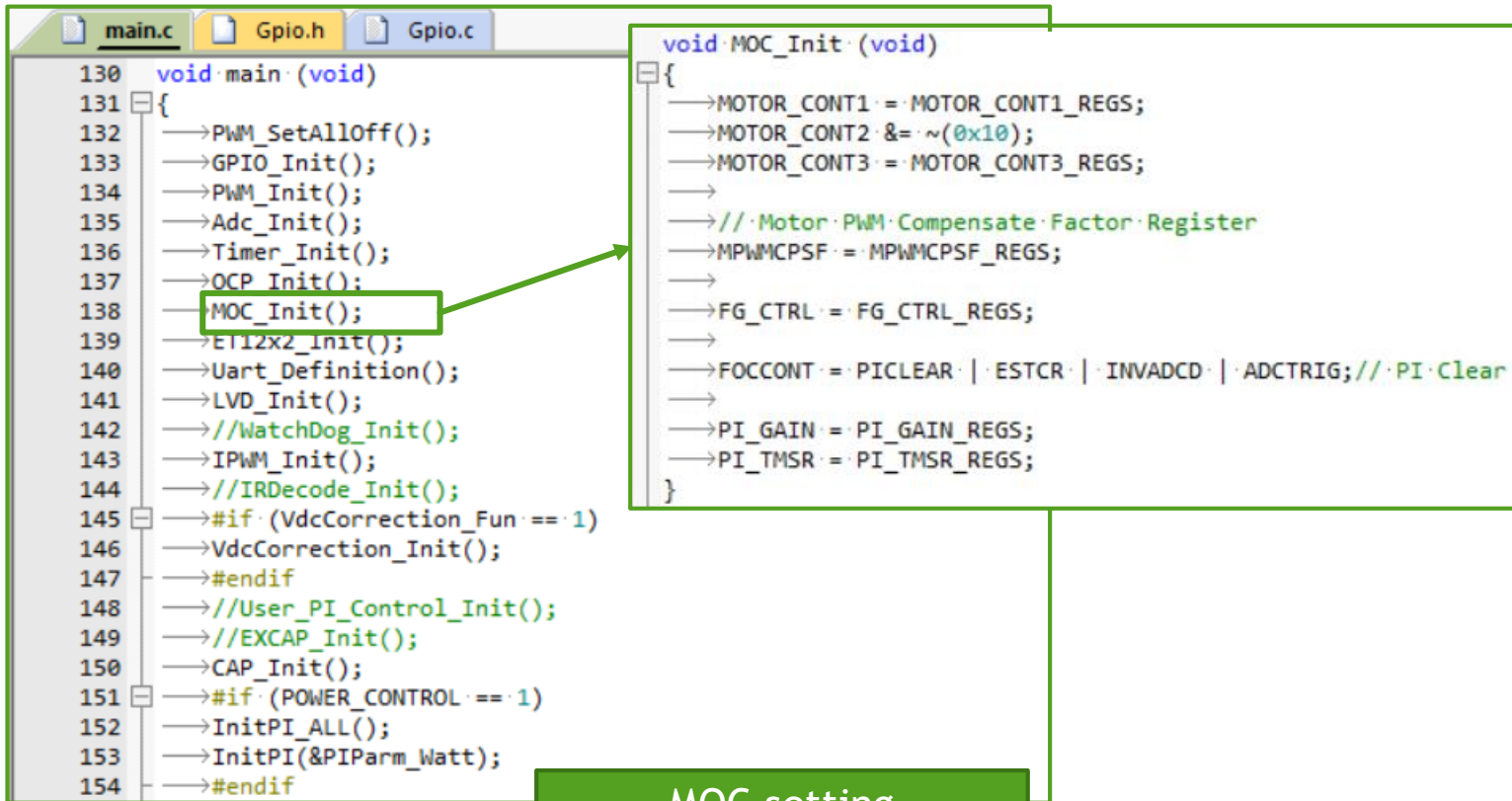
過電流設定 I_SHORT (單位: V)

 $I_SHORT(A) = \text{過流電壓設定} / \text{限流電阻}$ 例如: 過流電壓設定 0.4V, 限流電阻 0.1歐姆, 則 $I_SHORT = 0.4 / 0.1 = 4A$

可經由OCPCONT[7](OCPST)檢知, 或開啟中斷IEN1[0](OCPSIE), 中斷相量 interrupt 8
發生時硬體會依據OCPCONT[0](OCPMS)設定先行保護MOS, 後由程式做相對應處理



MOC setting



```
main.c  Gpio.h  Gpio.c

130 void main(void)
131 {
132     →PWM_SetAllOff();
133     →GPIO_Init();
134     →PWM_Init();
135     →Adc_Init();
136     →Timer_Init();
137     →OCP_Init();
138     →MOC_Init();
139     →EI12x2_Init();
140     →Uart_Definition();
141     →LVD_Init();
142     →//WatchDog_Init();
143     →IPWM_Init();
144     →//IRDecode_Init();
145     →#if (VdcCorrection_Fun == 1)
146     →VdcCorrection_Init();
147     →#endif
148     →//User_PI_Control_Init();
149     →//EXCAP_Init();
150     →CAP_Init();
151     →#if (POWER_CONTROL == 1)
152     →InitPI_ALL();
153     →InitPI(&PIParm_Watt);
154     →#endif

void MOC_Init(void)
{
    →MOTOR_CONT1 = MOTOR_CONT1_REGS;
    →MOTOR_CONT2 &= ~(0x10);
    →MOTOR_CONT3 = MOTOR_CONT3_REGS;
    →
    →//Motor PWM Compensate Factor Register
    →MPWMCPSF = MPWMCPSF_REGS;
    →
    →FG_CTRL = FG_CTRL_REGS;
    →
    →FOCCONT = PICLEAR | ESTCR | INVADCD | ADCTRIG; //PI Clear
    →
    →PI_GAIN = PI_GAIN_REGS;
    →PI_TMSR = PI_TMSR_REGS;
}
```

MOC setting



Interrupt Setting

Interrupt setting

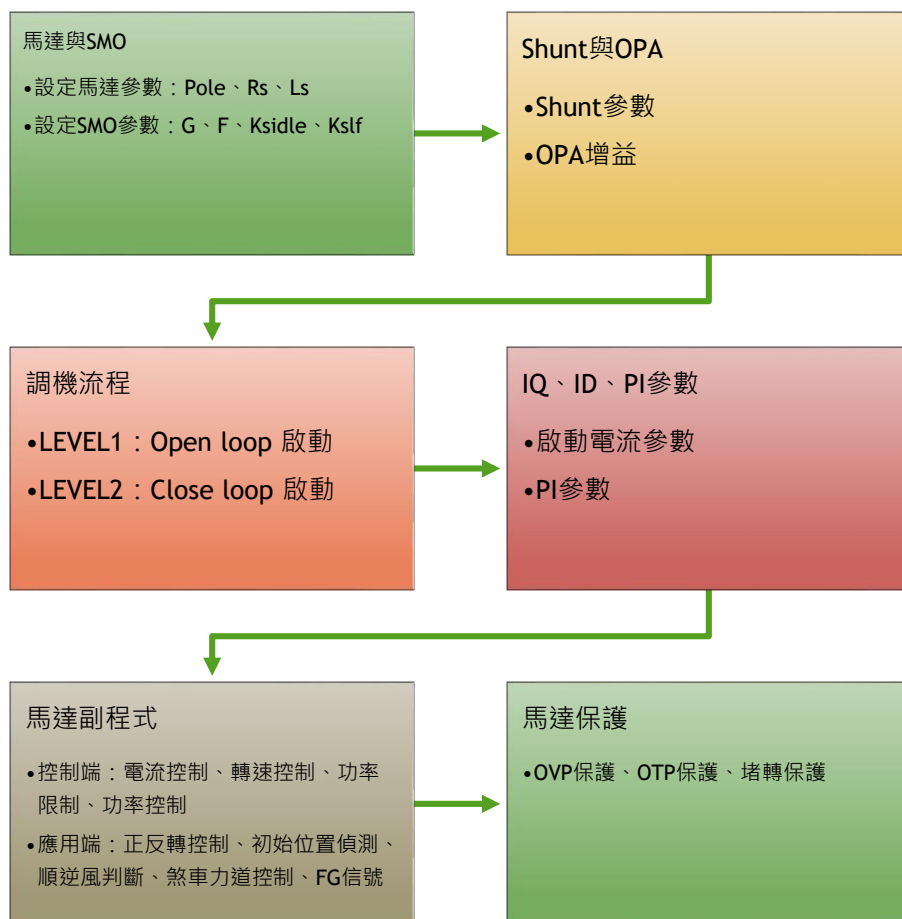
Interrupt setting

```
main.c
75 void main(void){
76     →PWM_SetAllOff();
77     →GPIO_Init();
78     →PWM_Init();
79     →Adc_Init();
80     →Timer_Init();
81     →OCP_Init();
82     →MOC_Init();
83     →Interrupt_Init();
84     →
85     →#if (Uart_Debug == 1)
86     →Uart_Definition();
87     →#endif
88     →
89     →#if (Uart_Debug == 0)
90     →led_a_OUTPUT;
91     →led_b_OUTPUT;
92     →led_c_OUTPUT;
93     →led_a = 0;
94     →led_b = 0;
95     →led_c = 0;
96     →#endif
97     →
98     →//LVD_Init();
99     →WatchDog_Init();
100    →CAP_Init();
101    →
102    →#if (CONTROL_MODE == Power_Control) || (CONTROL_MODE == 2)
103    →InitPI_ALL();
104    →InitPI(&PIParm_Watt);
105    →#endif
```

```
{
→EX0 = 0; //External0_ISR interrupt 0
→ET0 = 1; //Timer0_ISR interrupt 1
→EX1 = 0; //External1_ISR interrupt 2
→ET1 = 1; //Timer1_ISR interrupt 3
→ESP = 1; //Uart_ISR interrupt 4
→ET2 = 0; //Timer2_ISR interrupt 5
→OCPSIE = 1; //OCP_ISR interrupt 8
→ADCIE = 1; //ADC_ISR interrupt 9
→MPWMMINIE = 0; //PwmMin_ISR interrupt 10
→MPWMMAXIE = 1; //PwmMax_ISR interrupt 11
→IICIE = 0; //IIC_ISR interrupt 12
→LVDIIE = 0; //LowVoltage_ISR interrupt 13
→WDIIE = 0; //WatchDog_ISR interrupt 14
→CAPIE = 0; //Cap_ISR interrupt 15
→//IP0 = 0x0C; //Interrupt Priority
→//IP1 = 0x06; //Group 2 > 3 > 1 > 0
→EA = 1; //Allow interrupt
}
```

main.h	
Expand All	Collapse All
Help	
<input checked="" type="checkbox"/> Show Grid	
Option	Value
Group0 - LVDIF IE0	Level_0
Group1 - WDTIF TF0	Level_0
Group2 - OCPSIF ADCIF IE1	Level_3
Group3 - MPWMMINIF MPWMMAXIF TF1	Level_0
Group4 - SPIF(TI, RI)	Level_0
Group5 - CAPIF TF2	Level_0

Motor Control 調機流程 (1)



Motor.h	
Expand All	Collapse All
Help	Show Grid
Option	Value
+ Set motor parameters	
+ Set Rshunt and OPA_Gian	
+ Set the motor tuning process	
+ Set FOC LOOP Parameter	
+ Set motor control program	
+ Set Fairwind and Headwind judgment function	
+ Set motor protection function	
TailWind Determine Time(unit : ms)	200
Stop_Fun Time (unit : ms)	200
+ Set Protection to retry	
+ Error code(MotorErrorState)	



Motor Control 調機流程 (2)

Motor.h	
Expand All Collapse All Help	
Option	Value
[-] Set motor parameters	
Motor_Pole	2
Motor SMO_G	16000
Motor SMO_F	32346
Motor SMO_Kslf	8000
Motor SMO_Z-Corr	32767
Motor SMO Kslide (Sat)	16000
Motor SMO MaxSmcError (Limit)	32767

設定馬達極數(必填)。

馬達參數不熟悉情況下，SMO參數建議初始設定如下：

SMO參數G：16000

SMO參數F：32000

SMO參數Kslf：8000

SMO參數ZCorr：32767

SMO參數Kslide：32767

SMO參數MaxSmcError：32767

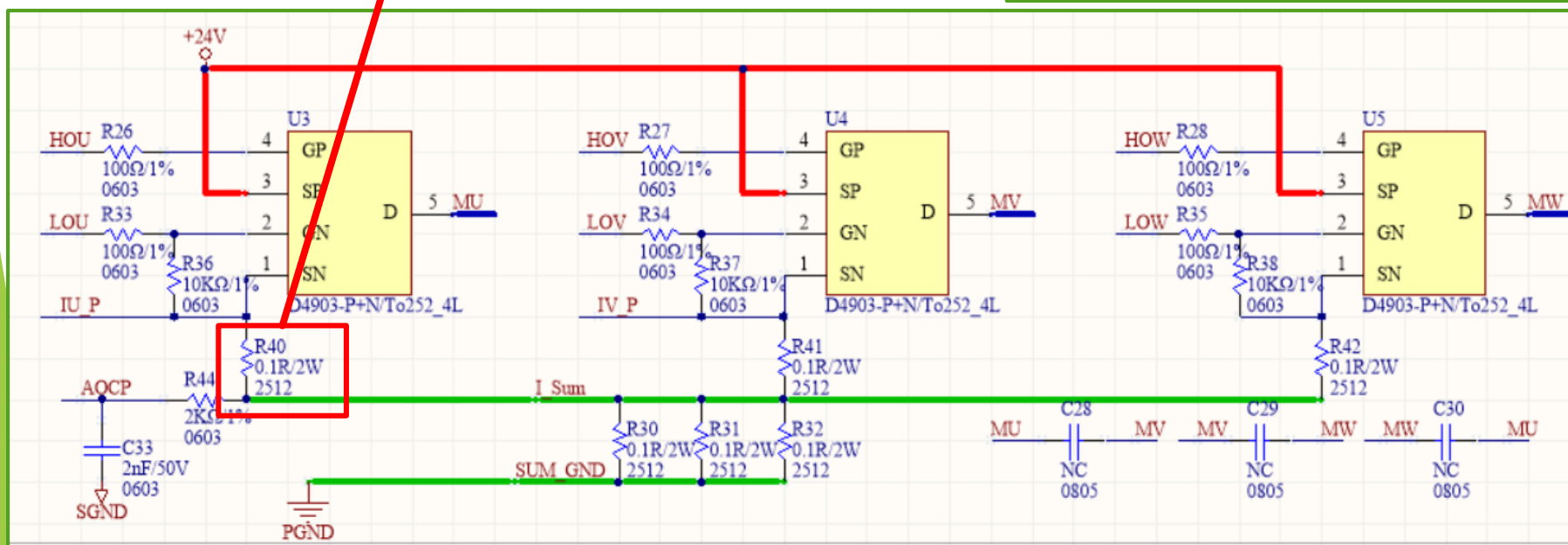
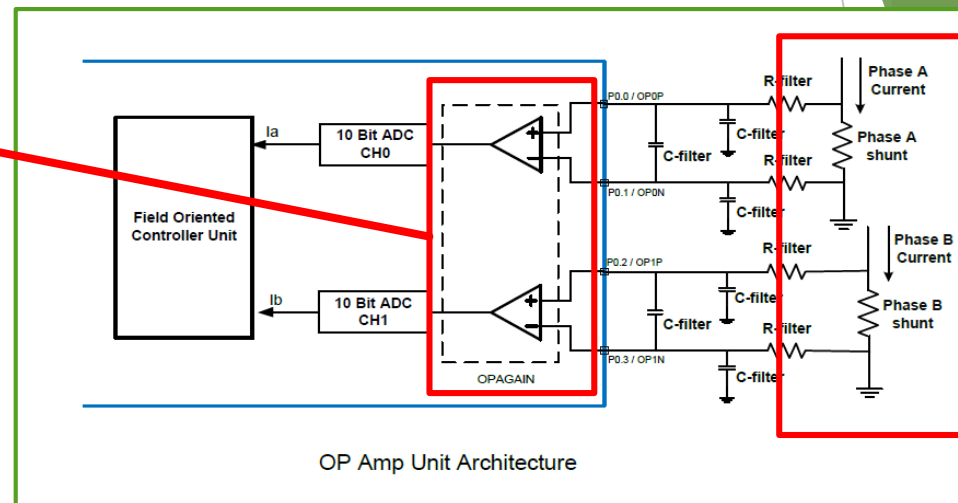
SMO參數SMOGain：327677



Motor Control 調機流程 (3)

Set Rshunt and OPA_Gian	
Rshunt (unit : 0.1m Ohm)	1000
OPA_Gian	5 Gain

填寫Shunt電阻，與放大器倍率
設計上注意Shunt電阻電壓差，不得超過： $\pm 0.5V$



Motor Control 調機流程 (4)

OpenLoop

- SmoPLL強制角度啟動，需手動銜接閉迴路

CloseLoop

- 銜接閉迴路時，系統穩定運轉，代表整體馬達參數正確，之後即可設定LEVEL2

Set the motor tuning process	
FOC_Control_Stage	CloseLoop
Set IQ parking duration(unit : ms)	10
Set SMO_PLL initial speed (unit : 10rpm)	1
Set SMO_PLL end speed (unit : 10rpm)	300
Set PLL accumulation	2
Set SMO_RAMP acceleration slope (unit : m...	1
Set SMO_DELAY Delay time (unit : ms)	10

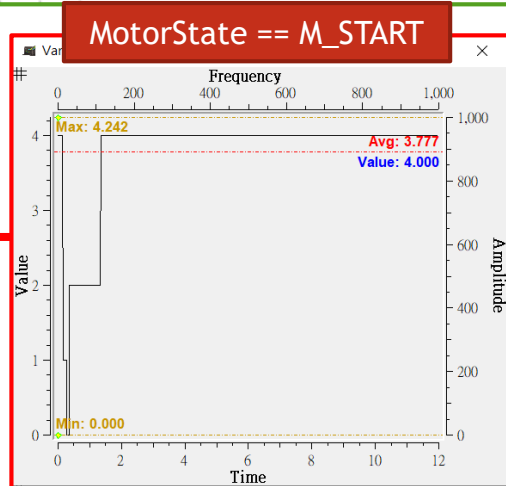
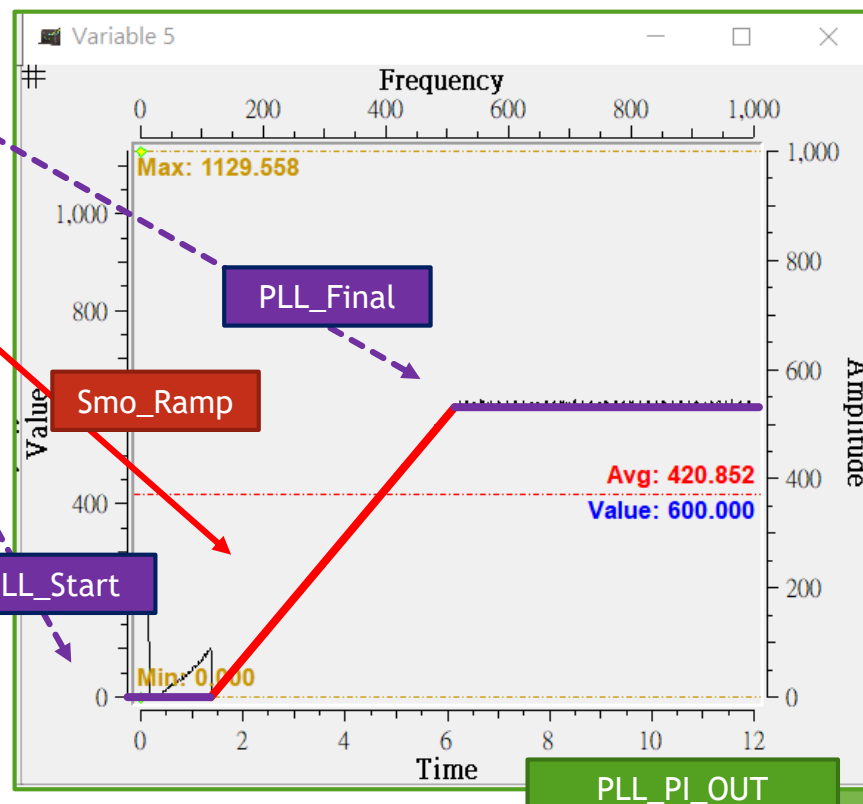
Set FOC LOOP Parameter	
IQ	
Set IQ Current parameter	
Set IQ Initial current (unit : mA)	0
Set IQ Starting current (unit : mA)	440
Set IQ End current (unit : mA)	450
IQ_TailWind Value (unit : mA)	500



Motor Control 調機流程 (5)

Set the motor tuning process		
FOC_Control_Stage	1.	OpenLoop
Set IQ parking duration(unit : ms)	10	
Set SMO_PLL initial speed (unit : 10rpm)	1	2.
Set SMO_PLL end speed (unit : 10rpm)	300	
Set PLL accumulation	2	
Set SMO_RAMP acceleration slope (unit : ms)	1	
Set SMO_DELAY Delay time (unit : ms)	10	
Set FOC LOOP Parameter		
IQ		
Set IQ Current parameter		
Set IQ Initial current (unit : mA)	0	
Set IQ Starting current (unit : mA)	440	
Set IQ End current (unit : mA)	450	
IQ_TailWind Value (unit : mA)	500	

1. 設定調機流程 OpenLoop
2. 設定開迴路"啟動 - 結束"轉速
3. 設定開迴路"啟動 - 結束"電流

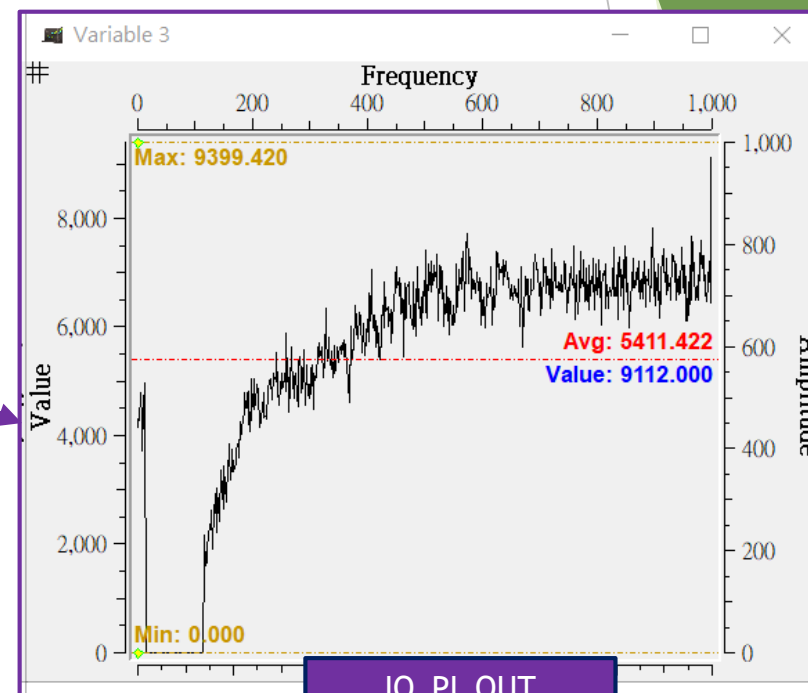


```
enum MotorStatus
{
    →M_OFF = 0,
    →M_INIT = 1,
    →M_TAILWIND = 2,
    →M_IPD = 3,
    →M_START = 4,
    →M_RUN = 5,
    →M_STOP = 6,
    →M_BREAK = 7,
    →M_ERROR = 8,
    →M_BMF_BREAK = 9
};
```

Motor Control 調機流程 (6)

Set the motor tuning process	
FOC_Control_Stage	OpenLoop
Set IQ parking duration(unit : ms)	10
Set SMO_PLL initial speed (unit : 10rpm)	1
Set SMO_PLL end speed (unit : 10rpm)	300
Set PLL accumulation	2
Set SMO_RAMP acceleration slope (unit : ms)	1
Set SMO_DELAY Delay time (unit : ms)	10
Set FOC LOOP Parameter	
IQ	
Set IQ Current parameter	
Set IQ Initial current (unit : mA)	0
Set IQ Starting current (unit : mA)	440
Set IQ End current (unit : mA)	450
IQ_TailWind Value (unit : mA)	

設定Openloop啟動電流



定位時間

啟動電流

結束電流

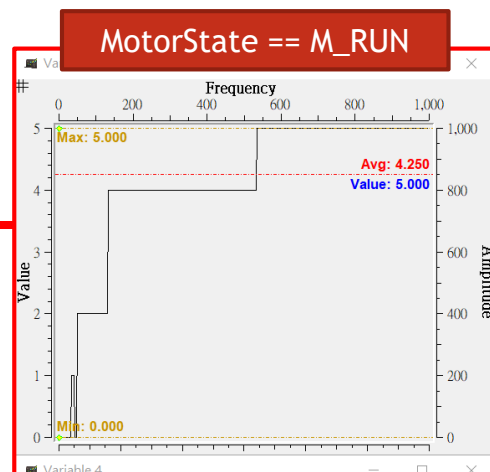
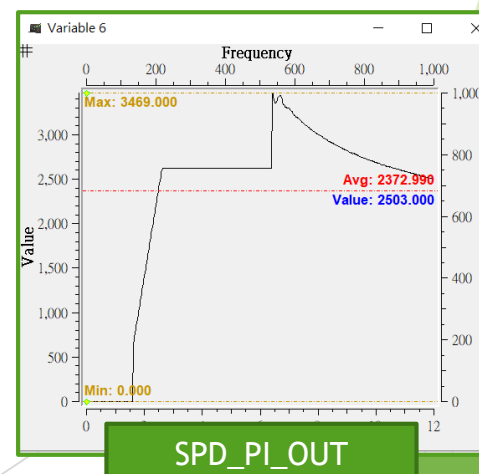
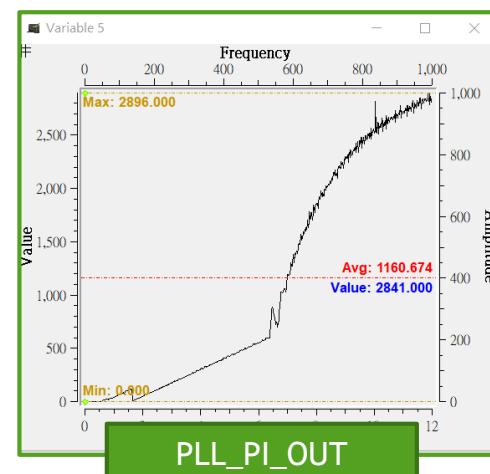
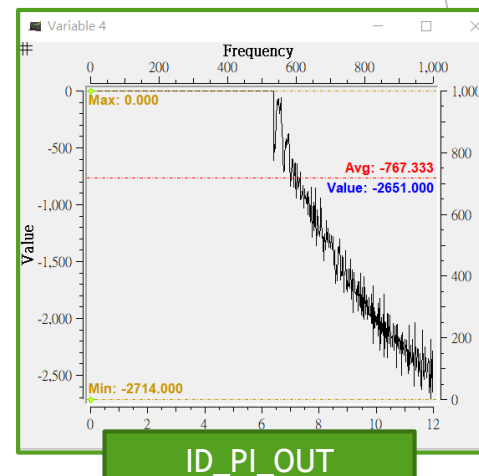
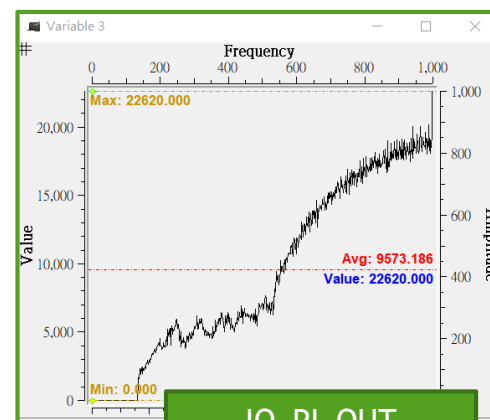
順逆風啟動電流



Motor Control 調機流程 (7)

1. 開迴路運轉調完成
2. 設定調機流程 CloseLoop
3. 進入閉迴路階段，調機完成

Set the motor tuning process	
FOC_Control_Stage	CloseLoop
Set IQ parking duration(unit : ms)	10
Set SMO_PLL initial speed (unit : 10rpm)	1
Set SMO_PLL end speed (unit : 10rpm)	300
Set PLL accumulation	2
Set SMO_RAMP acceleration slope (unit : ms)	1
Set SMO_DELAY Delay time (unit : ms)	10



```
enum MotorStatus
{
    →M_OFF = 0,
    →M_INIT = 1,
    →M_TAILWIND = 2,
    →M_IPD = 3,
    →M_START = 4,
    →M_RUN = 5,
    →M_STOP = 6,
    →M_BREAK = 7,
    →M_ERROR = 8,
    →M_BMF_BREAK = 9
};
```



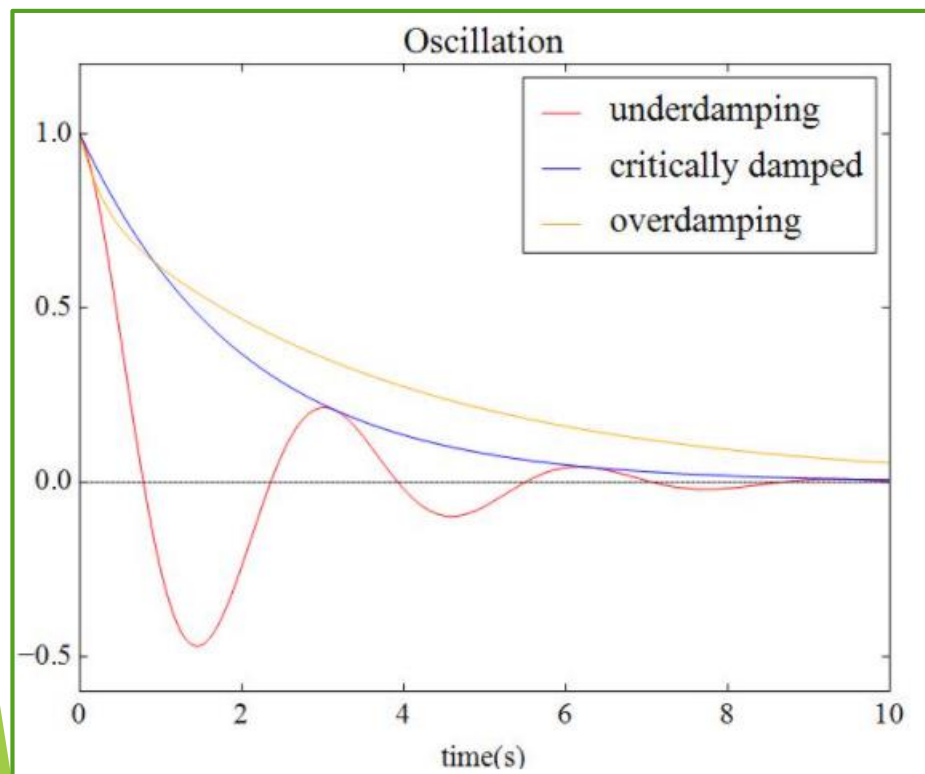
Motor Control 調機流程 (8)

各項PI控制器參數，依造不同馬達參數微調。

在自動控制理論中，系統響應分類如下：

1. 過阻尼：PI 控制器調整方向， $K_p \uparrow$ 、 $K_i \downarrow$ 。
2. 欠阻尼：PI 控制器調整方向， $K_p \downarrow$ 、 $K_i \uparrow$ 。
3. 臨界阻尼： K_p 、 K_i 為理想值。

K_t 為反積分終結參數，預設32767 即可。



Set IQ PI parameters	
Kp parameters	28000
Ki parameters	160
Kt parameters	32767
MaxLimit parameters	32767
MinLimit parameters	32767

Set ID PI parameters	
Kp parameters	28000
Ki parameters	160
Kt parameters	32767
MaxLimit parameters	32767
MinLimit parameters	32767

Set SPEED PI parameters	
Start Kp	0
Final Kp	6000
Ki parameters	130
Kt parameters	32767
MaxLimit (unit : mA)	550
MinLimit (unit : mA)	0
Speed Cycle parameters	20

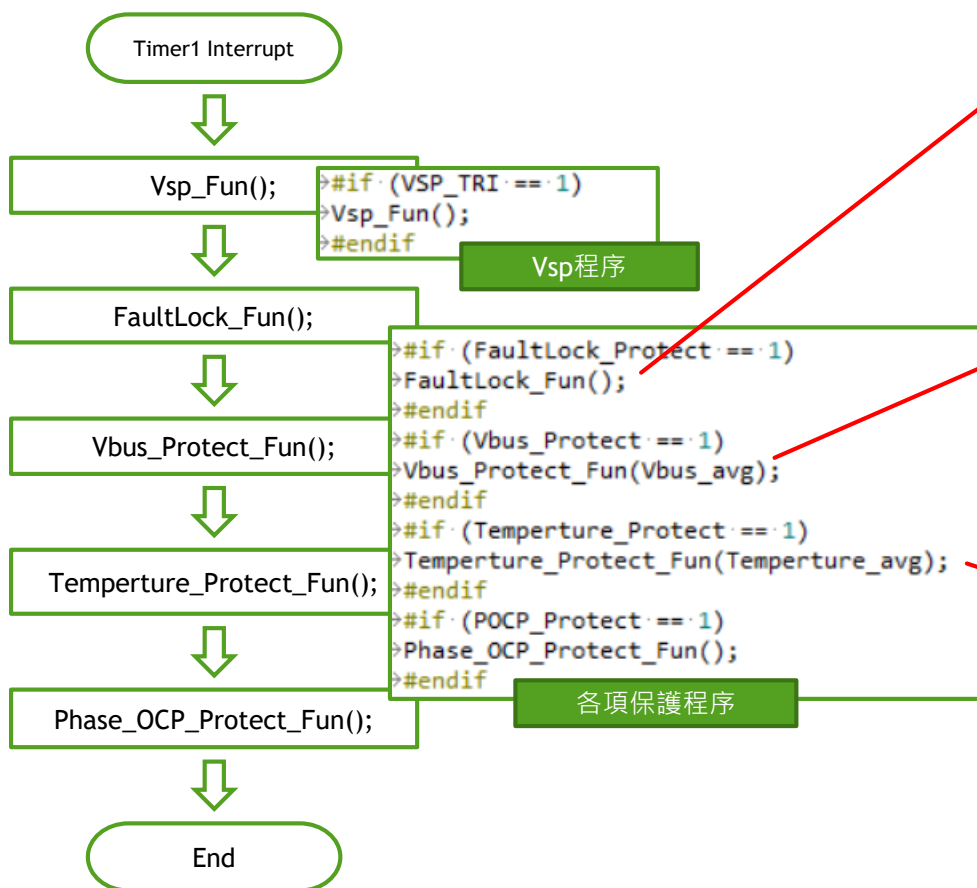
PLL	
Start Kp	1000
Final Kp	1000
Ki parameters	50
HeadWind/TailWind	
Kt parameters	32767
MaxLimit parameters	32767
MinLimit parameters	32767

PI_GAIN	
PLL KI Gain x16	Enable
PLL KP Gain x16	Enable
SPEED KI Gain x16	Disable
SPEED KP Gain x16	Enable
ID KI Gain x16	Disable
ID KP Gain x16	Disable
IQ KI Gain x16	Disable
IQ KP Gain x16	Disable



Motor Control 調機流程 (9)

馬達各項保護功能



Locked-rotor protection (LRP)	<input checked="" type="checkbox"/>
Motor speed abnormally high value (unit : 10rpm)	13000
Motor speed abnormally low value (unit : 10rpm)	600
LRP DURATION (unit : ms)	500

Overvoltage/Undervoltage protection (OVP/UVP)	<input checked="" type="checkbox"/>
Set Vbus A/D Channel	CH2
Set Vbus rate parameter	2160
OVP Values (unit : 0.1V)	3800
OVP recovery Values (unit : 0.1V)	3750
UVP recovery Values (unit : 0.1V)	1450
UVP Values (unit : 0.1V)	1400
BUS_VOLT_DURATION (unit : ms)	50

1.選擇OVP_CH

2.需調整正確倍率

3.調整OVP保護

Over temperature protection(OTP)	<input checked="" type="checkbox"/>
Set OTP A/D Channel	CH5
OTP A/D Values (unit : Val)	670
OTP recovery A/D Values (unit : Val)	620
OVER_TEMPERATURE_LOAD_REDUCE_VALUE (unit : Val)	670
TEMPERATURE_DURATION (unit : ms)	500



Motor Control 調機流程 (10)

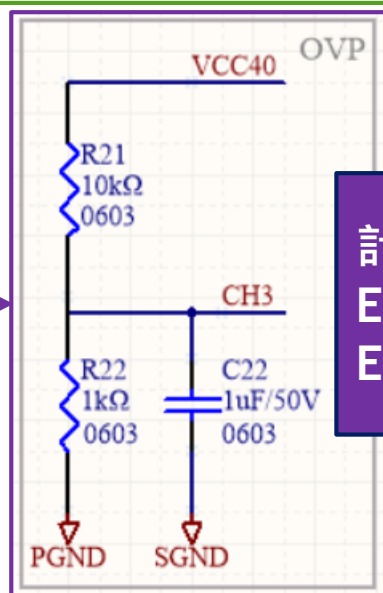
馬達各項保護功能

Overvoltage/Undervoltage protection (OVP/UVLP)	<input checked="" type="checkbox"/>
Set Vbus A/D Channel	CH2
Set Vbus rate parameter	2160
OVP Values (unit : 0.1V)	3800
OVP recovery Values (unit : 0.1V)	3750
UVP recovery Values (unit : 0.1V)	1450
UVP Values (unit : 0.1V)	1400
BUS_VOLT_DURATION (unit : ms)	50

1. 選擇OVP_CH

2. 需填寫OVP分壓正確倍率

3. 調整OVP保護



計算Vbus倍率參數, $V_{cc}=36V$, $AD_Val=669$

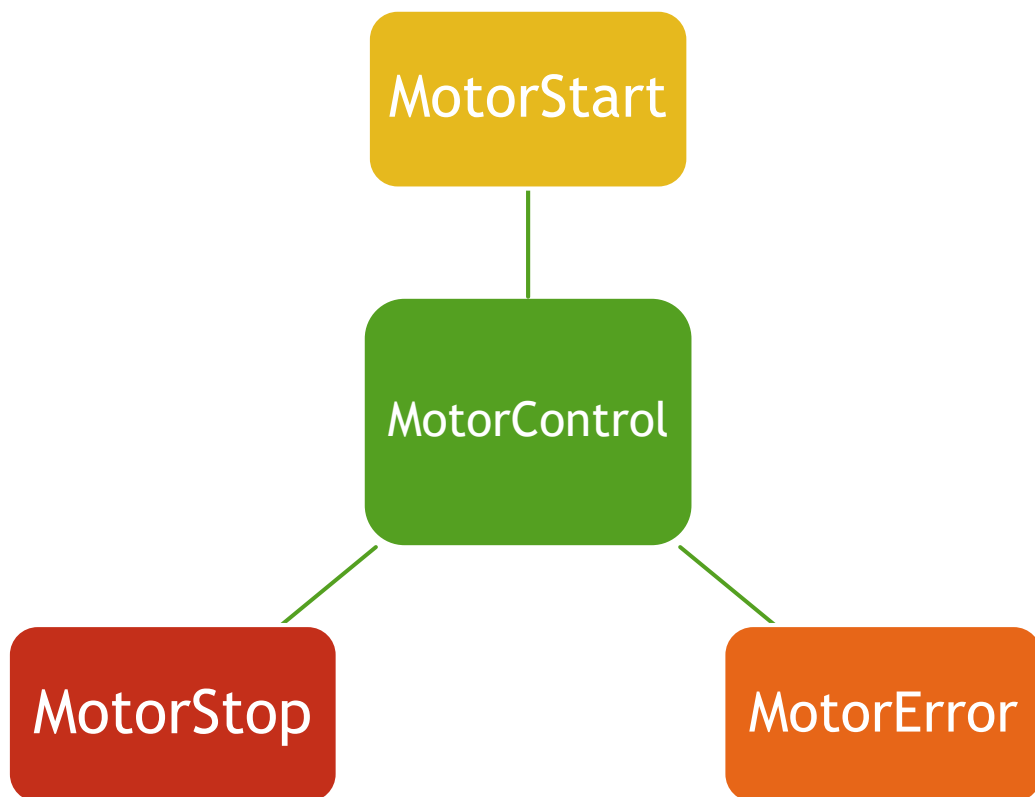
Ex1. Vbus倍率參數= $669/36=18.5833$

Ex2. Vbus倍率參數= $((R22/(R21+R22))/5)*1023=18.6$



馬達控制程序流程 (1)

馬達驅動程序流程



馬達驅動程序

```
void Motor_Control(void)
{
    → #if (CW_CCW_FUNCTION == 1)
    → if (CCWFlag != CCWFlagOld) // 正反转控制
    → MotorState = M_INIT;
    → CCWFlagOld = CCWFlag;
    → #endif

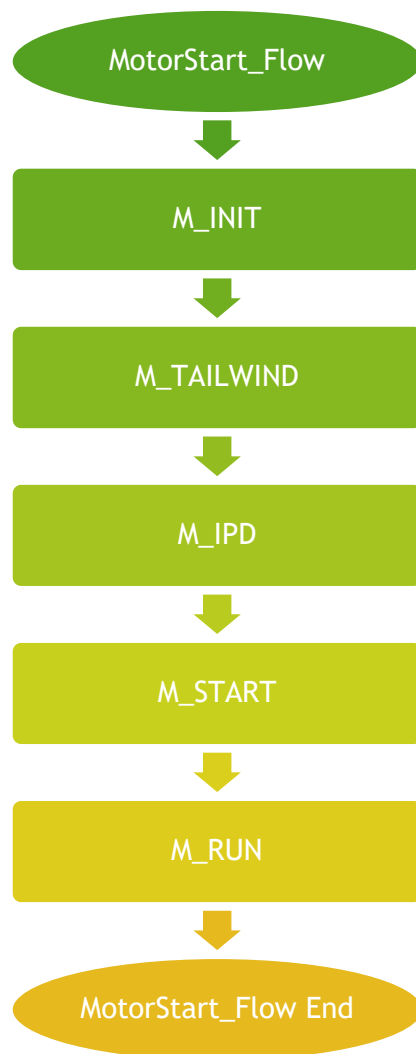
    → if (MotorErrorState)
    → {
    →     → if (MotorState != M_OFF)
    →     → ResetMOC();
    →     → PWM_SetAllOff();
    →     → MotorStartRetry_Flow();
    →     → #if ((FG_CTRL_REGS & 0x80) == 0x80)
    →     → FG_DISABLE;
    →     → #endif
    → }

    → else if ((SystemState & 0x04) == 0x04) // 啟動
    → {
    →     → MotorStart_Flow();
    → }

    → else
    → {
    →     → if (MotorState != M_OFF)
    →     → ResetMOC();
    →     → PWM_SetAllOff();
    →     → MotorStartRetryCount = 0;
    →     → MotorErrorState = Clear;
    →     → MotorState = M_OFF;
    →     → #if ((FG_CTRL_REGS & 0x80) == 0x80)
    →     → FG_DISABLE;
    →     → #endif
    → }
}
```



馬達啟動程序流程 (1)



馬達啟動程序

```
void MotorStart_Flow(void)
{
    switch (MotorState)
    {
        case M_INIT:
            MotorInit_Fun();
            if (CW CCW FUNCTION == 1) // 正反轉控制
                break;
        case M_TAILWIND:
            if (TAILWIND FUNCTION == 1)
                break;
        case M_IPD:
            if (IPD FUNCTION == 1)
                break;
        case M_START:
            if (StartUpState == S_IPD) // IPD Start Up
                IPDStart_Fun();
            if (StartUpState == S_TAILWIND) // Tailwind Start Up
                TailWindStart_Fun();
            break;
        case M_RUN:
            if (CURRENT CONTROL == 1)
            if (SPEED CONTROL == 1)
                break;
```



順逆風啟動機制 (1)

目前有實現出順逆風啟動的方法：

1. 順逆風判斷(Two BEMF分壓)
2. 順逆風判斷(Diode BEMF分壓)

目前有實現出順風啟動的方法：

1. 順風判斷(One BEMF分壓)

至少需要兩相BEMF才能做順逆風判斷，
做順風判斷則一相BEMF即可。

每個區塊都有規畫順逆風調整流程：

LEVEL_1：將程序停止至M_TAILWIND。

LEVEL_2：將程序執行至M_START、M_RUN。

```
enum MotorStatus
{
    →M_OFF = 0,
    →M_INIT = 1,
    →M_TAILWIND = 2,
    →M_IPD = 3,
    →M_START = 4,
    →M_RUN = 5,
    →M_STOP = 6,
    →M_BREAK = 7,
    →M_ERROR = 8,
    →M_BMF_BREAK = 9
};
```

Set Fairwind and Headwind judgment function	
BEMF Fairwind/Headwind judgment (resistance) Enable/Di...	<input checked="" type="checkbox"/>
BEMF Fairwind/Headwind judgment (Diode) Enable/Disable	<input checked="" type="checkbox"/>
BEMF TailWind Fun (One BEMF) Enable/Disable	<input checked="" type="checkbox"/>

BEMF Fairwind/Headwind judgment (resistance) Enable/Disable	<input checked="" type="checkbox"/>
BEMF_TAILWIND_IQ_OUT_VALUE (unit : Val)	8000
BEMF_V_CH	CH4
BEMF_W_CH	CH5
BEMF_TAILWIND_SOP	LEVEL 1
BEMF_TAILWIND_SPEED_MAX (unit : 10rpm)	1200
BEMF_TAILWIND_SPEED_MIN (unit : 10rpm)	300
BEMF_HEADWIND_SPEED_MAX (unit : 10rpm)	1200
BEMF_HEADWIND_SPEED_MIN (unit : 10rpm)	300

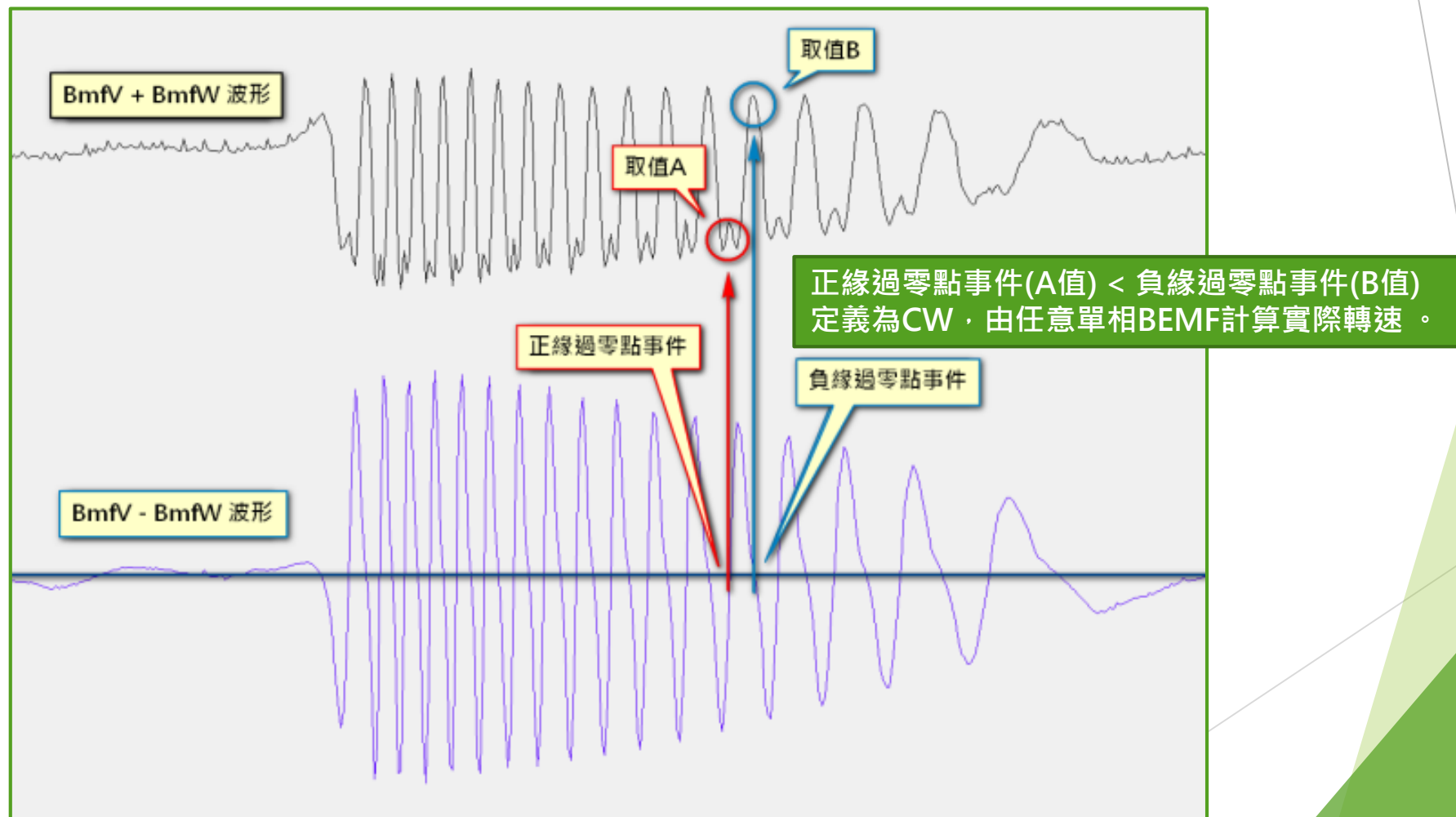
BEMF Fairwind/Headwind judgment (Diode) Enable/Disable	<input checked="" type="checkbox"/>
BEMF_TAILWIND_IQ_OUT_VALUE (unit : Val)	8000
BEMF_V_CH	CH4
BEMF_W_CH	CH5
BEMF_TAILWIND_SOP	LEVEL 2
BEMF_TAILWIND_SPEED_MAX (unit : 10rpm)	1200
BEMF_TAILWIND_SPEED_MIN (unit : 10rpm)	300
BEMF_HEADWIND_SPEED_MAX (unit : 10rpm)	600
BEMF_HEADWIND_SPEED_MIN (unit : 10rpm)	300

BEMF TailWind Fun (One BEMF) Enable/Disable	<input checked="" type="checkbox"/>
BEMF_TAILWIND_IQ_OUT_VALUE (unit : Val)	8000
BEMF_CH	CH4
BEMF_CH_LATEST_THETA	120Deg
BEMF_TAILWIND_SOP	LEVEL 2
BEMF_TAILWIND_SPEED_MAX (unit : 10rpm)	600
BEMF_TAILWIND_SPEED_MIN (unit : 10rpm)	180



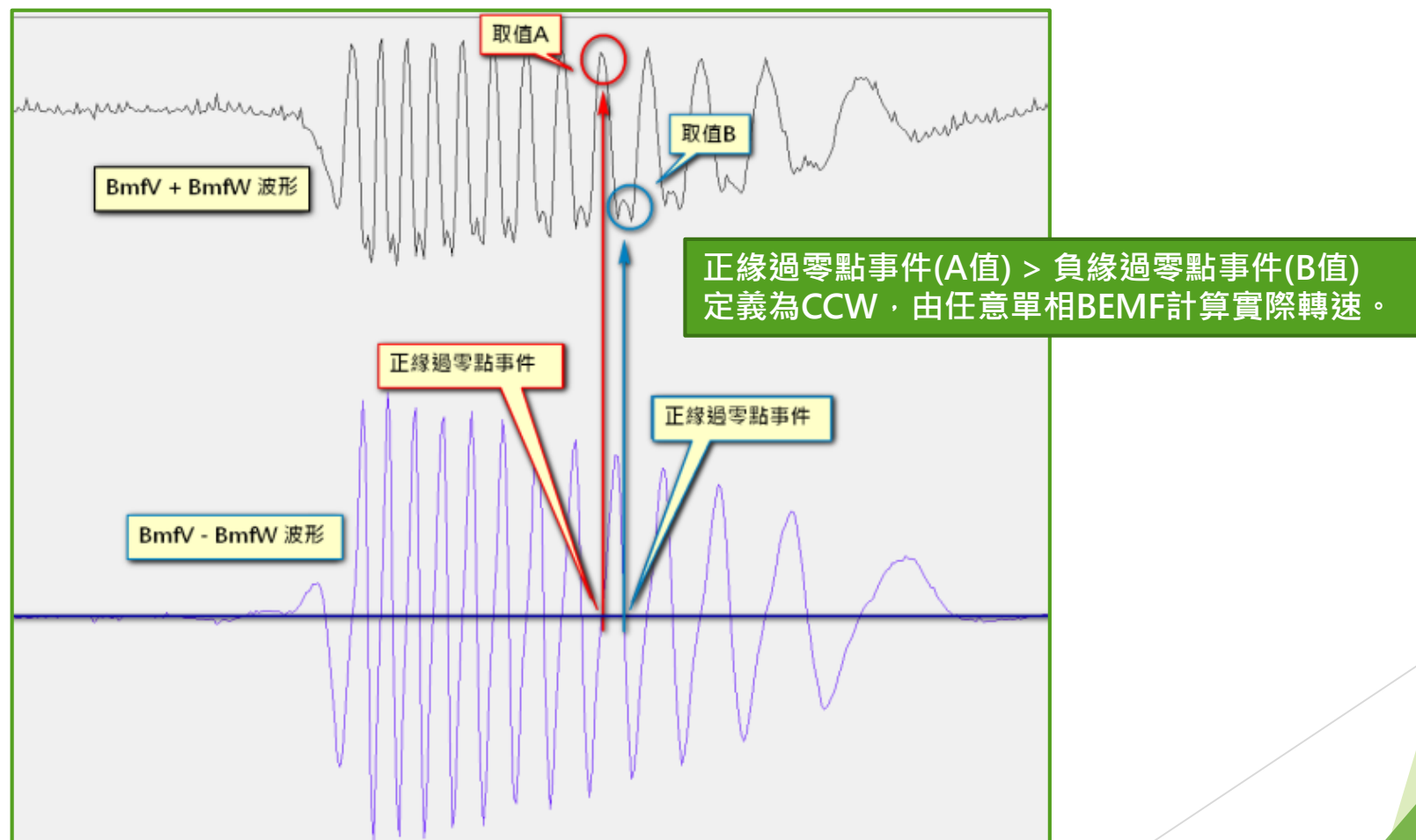
順逆風啟動機制 (2)

順逆風機制 - 任意兩相BEMF回授方式



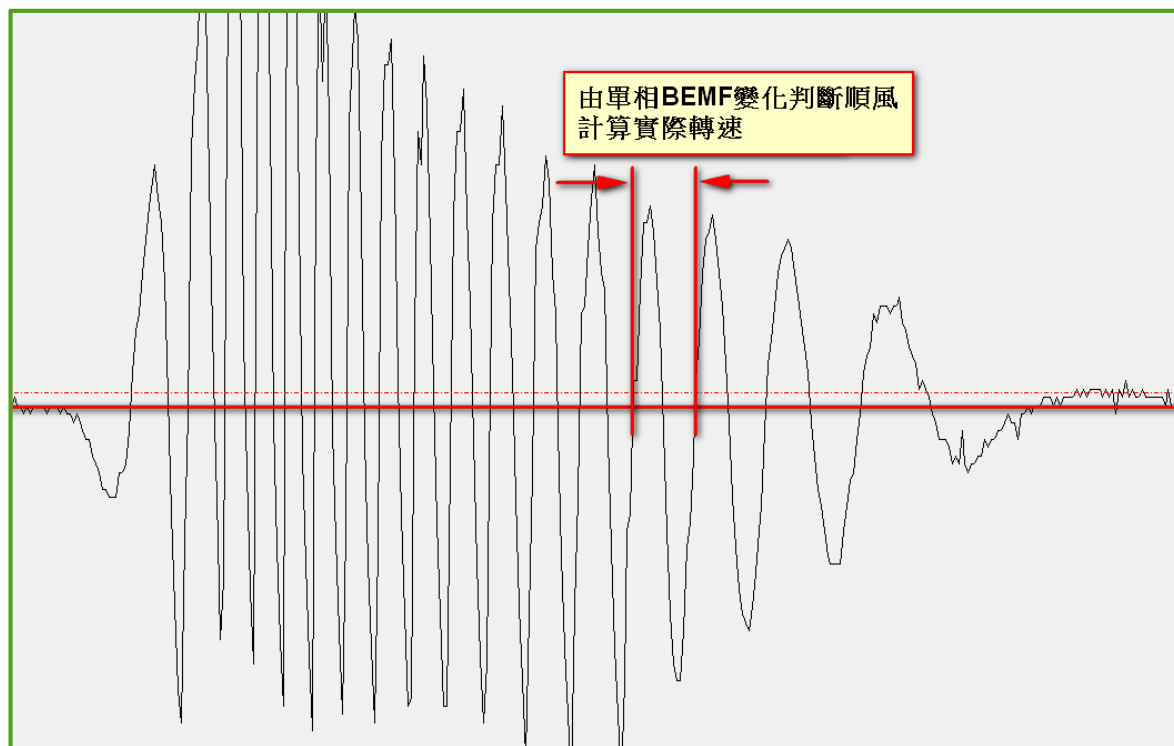
順逆風啟動機制 (3)

順逆風機制 - 任意兩相BEMF回授方式



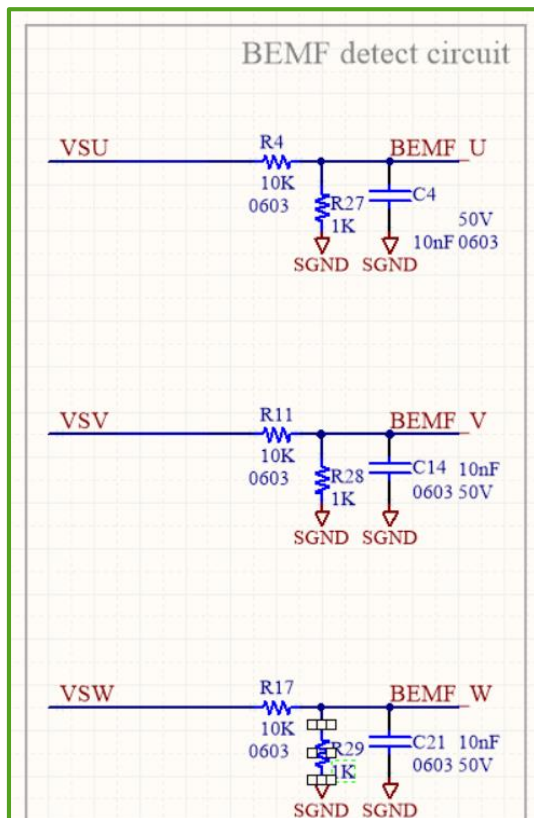
順逆風啟動機制 (4)

順逆風機制 - 單相BEMF回授方式

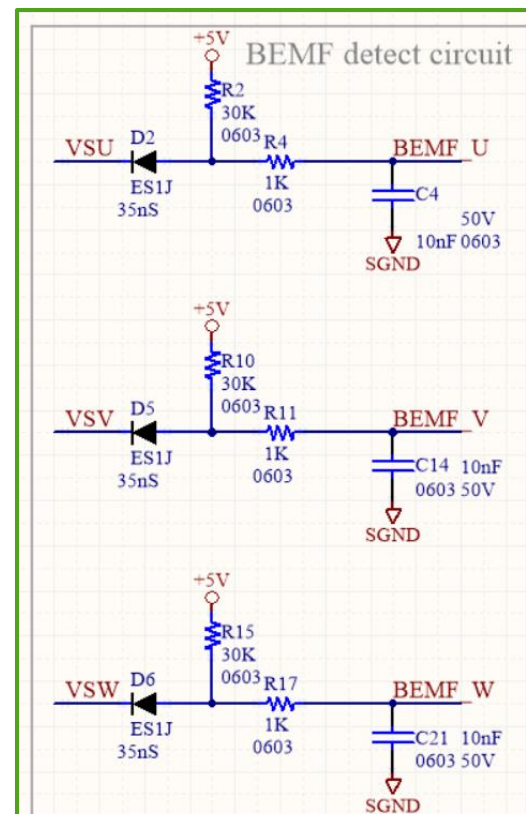


順逆風啟動機制 (5)

BEMF回授電路



分壓電阻：此線路可讀取到 各相BEMF的變化，但須注意分壓電阻的匹配，避免BEMF過大，導致ADC腳位損壞。



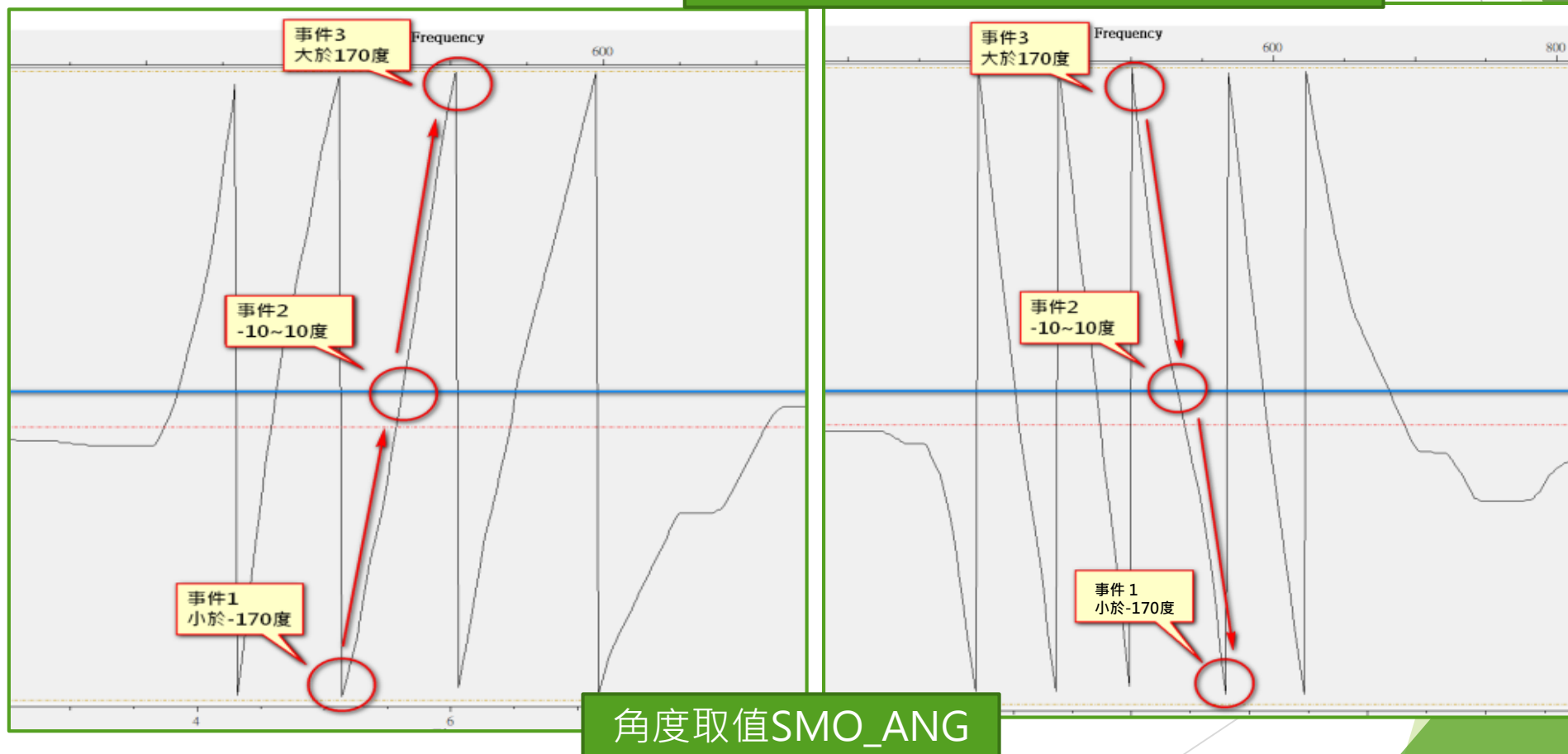
Diode分壓電阻：線路需要下臂使用PWM煞車，才能讀取到各相BEMF的變化。



順逆風啟動機制 (6)

順逆風機制 - 煞車回授電流方式

1. 事件1 > 事件2 > 事件3 > 定義為 CW
2. 事件3 > 事件2 > 事件1 > 定義為 CCW



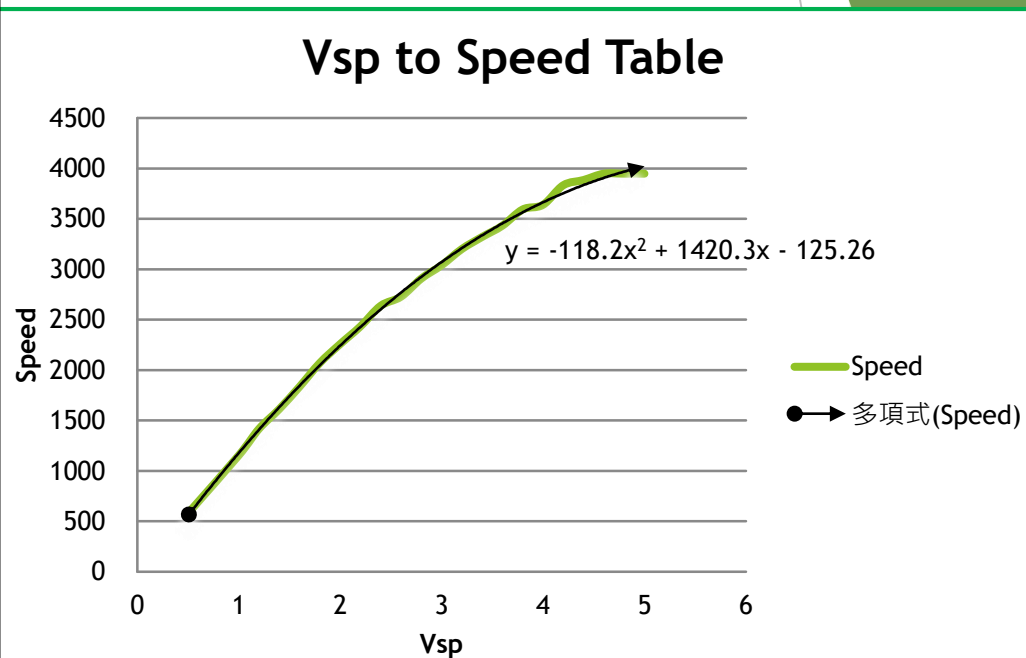
LookUpTable

LookupTable.h LookupTable_Fun main.c

Expand All Collapse All Help Show

Option	Value
Output : LookupTable_Data[1] (unit : Iq_Cmd)	327
Output : LookupTable_Data[2] (unit : Iq_Cmd)	1300
Output : LookupTable_Data[3] (unit : Iq_Cmd)	3750
Output : LookupTable_Data[4] (unit : Iq_Cmd)	6100
Output : LookupTable_Data[5] (unit : Iq_Cmd)	8300
Output : LookupTable_Data[6] (unit : Iq_Cmd)	9300
Input : Vsp_Data[1] (unit : Vsp_Val)	102
Input : Vsp_Data[2] (unit : Vsp_Val)	192
Input : Vsp_Data[3] (unit : Vsp_Val)	400
Input : Vsp_Data[4] (unit : Vsp_Val)	602
Input : Vsp_Data[5] (unit : Vsp_Val)	804
Input : Vsp_Data[6] (unit : Vsp_Val)	1002

Vsp(V)	I(mA)	Speed
0.51	40	590
1	70	1153
1.2	100	1417
1.4	130	1613
1.6	170	1835
1.8	220	2071
2	280	2258
2.2	340	2433
2.4	410	2638
2.6	460	2728
2.8	550	2908
3	620	3040
3.2	720	3201
3.4	800	3321
3.6	890	3433
3.8	1010	3594
4	1070	3640
4.2	1230	3834
4.4	1290	3886
4.6	1360	3949
4.8	1360	3949
5	1360	3949



```

→ if(Vsp_avg >= 102) // 204 = 1V
→ {
→   SystemState |= 0x04;
→   #if 1
→   CurrentCmd = look1_binlx(Vsp_avg, rtConstP.uDLookupTable_bp01Data, rtConstP.uDLookupTable_tableData, 5);
→   #else
→   CurrentCmd = (int)((float)Vsp_avg * IQ_GAIN);
→   if(CurrentCmd > IQ_MAX_LIMIT_VALUE)
→   CurrentCmd = IQ_MAX_LIMIT_VALUE;
→   else if(CurrentCmd < IQ_MIN_LIMIT_VALUE)
→   CurrentCmd = IQ_MIN_LIMIT_VALUE;
→   #endif
→ }

```

“6筆資料”，宣告LookupTable[5]

“宣告LookupTable[5]” >> 填“5”

利用差分建表方式，將Vsp - Speed Table
完整建立對應曲現出來。

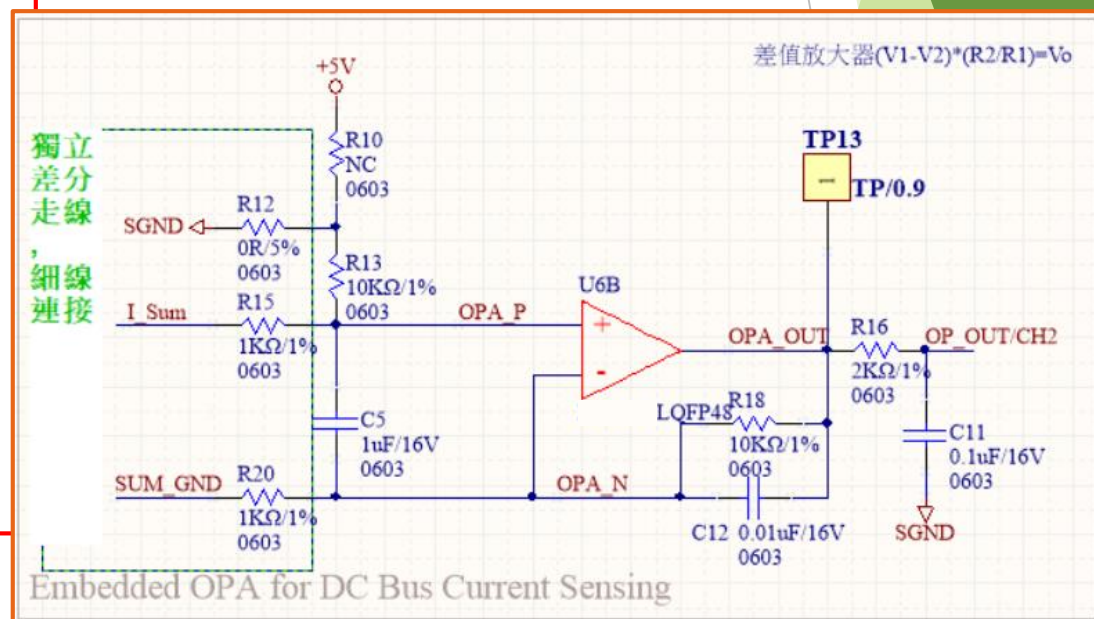
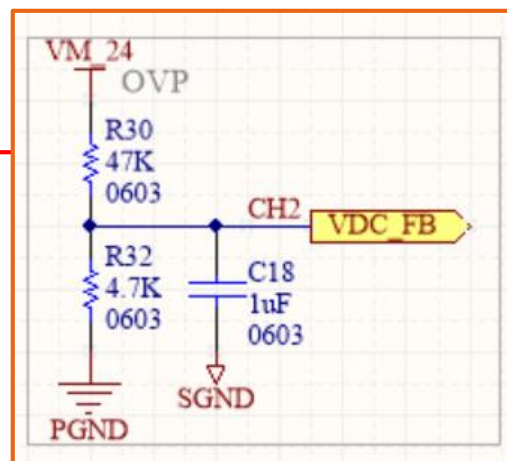


PowerControl (1)

PowerControl_Fun

取樣Vbus、Ibus

```
void ADC_ISR (void) interrupt 9
{
    → AdcFlag = 1;
    → #if (Vbus_Protect == 1)
    → Vbus_avg = Adc_Channel(V_BUS_CH);
    → #endif
    → #if ((POWER_CONTROL == 1) || (POWER_LIMIT == 1) || (POWER_CONTROL_USER_PI == 1))
    → Ibus_avg = Adc_Channel(I_BUS_CH);
    → #endif
    → #if (BEMF_TAILWIND_FUNCTION_4 == 1)
    → if (MotorState == M_TAILWIND) {
    → → BmfU = Adc_Channel(BEMF_CH);
    → }
    → #endif
    → #if (BEMF_TAILWIND_FUNCTION_1 == 1) || (BEMF_TAILWIND_FUNCTION_3 == 1)
    → if (MotorState == M_TAILWIND) {
    → → BmfV = Adc_Channel(BEMF_V_CH);
    → → BmfW = Adc_Channel(BEMF_W_CH);
    → }
    → #endif
    → #if (VSP_TRI == 1)
    → Vsp_avg = Adc_Channel(4);
    → Vsp_avg = 1023 - Vsp_avg;
    → #endif
    → #if (Temperature_Protect == 1)
    → Temperature_avg = Adc_Channel(TEMPERATURE_CH);
    → #endif
}
```



PowerControl (2)

PowerControl_Fun

```

if(MotorState == M_RUN)
{
    #if (POWER_CONTROL_USER_PI_SOP == 2)
    if(UserPI_PowerControlDelayCount > POWER_CONTROL_DELAY_DURATION)
    {
        UserPI_PowerControlDelayCount = 0;
        Watt = (int)((float)Vbus_avg * Ibus_avg * dPOWER_GAIN);
        USER_PI_ACTIVE;
        USER_CMD_MACRO(WATT_LIMIT_VALUE);
        USER_FB_MACRO(Watt);
        GET_USER_OUT_MACRO(CurrentCmd);
        CurrentCmdTemp = CurrentCmd;
        IQ_CMD_MACRO(CurrentCmdTemp);
    }
    else
    {
        UserPI_PowerControlDelayCount++;
    }
}
else
{
    Watt = (int)((float)Vbus_avg * Ibus_avg * dPOWER_GAIN);
    CurrentCmdTemp = CurrentCmd;
    IQ_CMD_MACRO(CurrentCmdTemp);
}
#endif
}

```

Power_Control & Power_Limit	
Set the rated output power (max) (unit : 0.01W)	1800
Set power magnification parameters (unit : 10 ⁻⁵)	100
Set I_BUS A/D Channel	CH2
POWER_SOP	LEVEL 2
Power_LookUpTable Enable/Disable	<input type="checkbox"/>

LEVEL_1 : 不執行 PowerControl_Fun
LEVEL_2 : 執行 PowerControl_Fun

```

#define POWER_CONTROL 1
#if (POWER_CONTROL == 1)
#define WATT_LIMIT_VALUE (unsigned long) 800
#define POWER_CONTROL_DELAY_DURATION 2
#define I_BUS_CH 2
#define POWER_PI_OUT (float) 300/1000
#define POWER_PI_OUT_VALUE (signed short)((float) POWER_PI_OUT * I_AMPLIFIER)
#define dPOWER_GAIN ((float) 594/100000)
#define POWER_CONTROL_SOP 2
#if (CURRENT_CONTROL == 0)
#error Wrong setting POWER_CONTROL and CURRENT_CONTROL !!!
#endif

```

只有在CURRENT_CONTROL，才可以使用POWER_CNOTROL。
#error Wrong setting POWER_CONTROL and CURRENT_CONTROL !!!



IPD程式流程 (1)

AOCPCONT		Address = EEH				Reset Value = 0xE7H		
Analog OCP Control Register								
Bit	DOCPNEN	AOCPEN	OPAPD	IPD	----	I_SHORT[3:0]		
	7	6	5	4	3	2	1	0
Type	R	R	----	----	----	R/	R/W	R/W
DOCPNEN [7]	Digital OCPN enable: 0 : Disable 1 : Enable							
AOCPEN [6]	Analog OCP enable: 0 : Disable 1 : Enable							
OPAPD [5]	OPA Power Down 0 : Normal 1 : OPA Power Down							
IPD [4]	IPD (Initial Position Detect) Path Select 0 : IPD Current Compare from AOCPP Path 1 : IPD Current Compare from OPA Path							
I_ SHORT [2:0]	Analog OCP SHORT level select : (OCP interrupt : OCPIF) 000 : 0.15V 001 : 0.2V 010 : 0.25V 011 : 0.3V 100 : 0.35V 101 : 0.4V 110 : 0.45V 111 : 0.5V(default)							

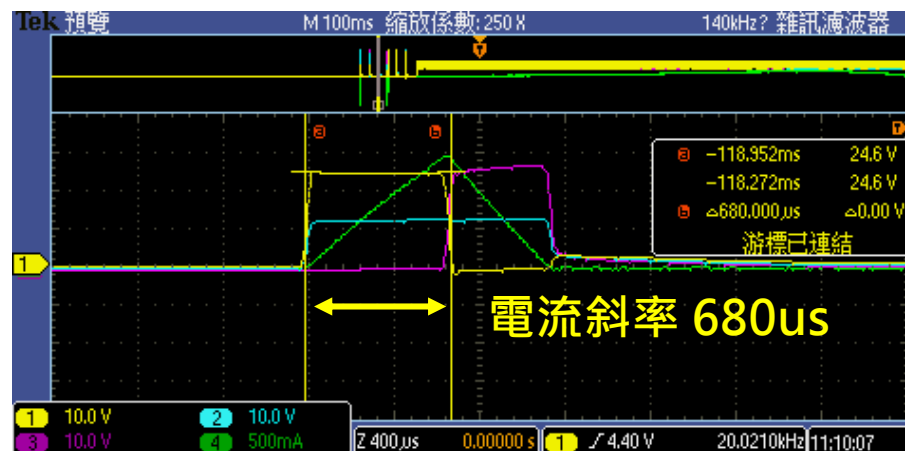
設定 AOCPCONT : IPD_LEVEL

Set IPD LEVEL	
I_SHORT	0.15V
AOCPEN	Enable
DOCPEN	Disable
IPD Path Select	IPD Current Compare from AOCPP Path
Set IPD IAECYC	
IAECYC	24MHz

選擇IPD路徑

當馬達電流斜率太緩慢時，需調低IPD_CYC：

1. 電流斜率時間 < 1.3ms : IPD_CYC 設定48MHz
2. 電流斜率時間 < 2.6ms : IPD_CYC 設定24MHz
3. 電流斜率時間 < 5.2ms : IPD_CYC 設定12MHz
4. 電流斜率時間 < 10.4ms : IPD_CYC 設定 6MHz



IPD程式流程 (2)

IPD程式流程建議分步驟執行

```
void IPDDetect_Fun(void)
{
    #define IPDAdvanceAng 30
    switch (IPD_Detect_State)
    {
        case 0:
            AOCPCONT = IPD_LEVEL;
            Break_Fun(); // (註: 當 IPD 時 N+N Gate Driver 需要下臂預充電)
            IPD_Cnt++;
            if (IPD_Cnt > 20)
            {
                IPD_Detect_State = 1;
                IPD_Cnt = 0;
            }
            break;
        case 1:
            WatchDog_Disable(); // (註: 當 IPD 時 會觸發 WatchDog Reset, Disable WatchDog)
            IPD_Init();
            WatchDog_Init();
            IPD_Detect_State = 2;
            break;
        case 2:
            if (IPDPattern == 4) LatestTheta = (64+IPDAdvanceAng)<<6;
            else if (IPDPattern == 5) LatestTheta = (128+IPDAdvanceAng)<<6;
            else if (IPDPattern == 2) LatestTheta = (192+IPDAdvanceAng)<<6;
            else if (IPDPattern == 3) LatestTheta = (256+IPDAdvanceAng)<<6;
            else if (IPDPattern == 6) LatestTheta = (320+IPDAdvanceAng)<<6;
            else if (IPDPattern == 1) LatestTheta = (383+IPDAdvanceAng)<<6;
            else LatestTheta = (383+IPDAdvanceAng)<<6;
            MotorErrorState &= ~(AOCPCONT);
            MotorState = M_START;
            IPD_Detect_State = 0;
            break;
        default:
            break;
    }
}
```

AOCPCONT : IPD_LEVEL
需要預先設定

Set IPD LEVEL	
I_SHORT	0.15V
AOCPEN	Enable
DOCPEN	Disable
IPD Path Select	IPD Current Compare from AOCP Path
Set IPD IAECYC	
IAECYC	24MHz

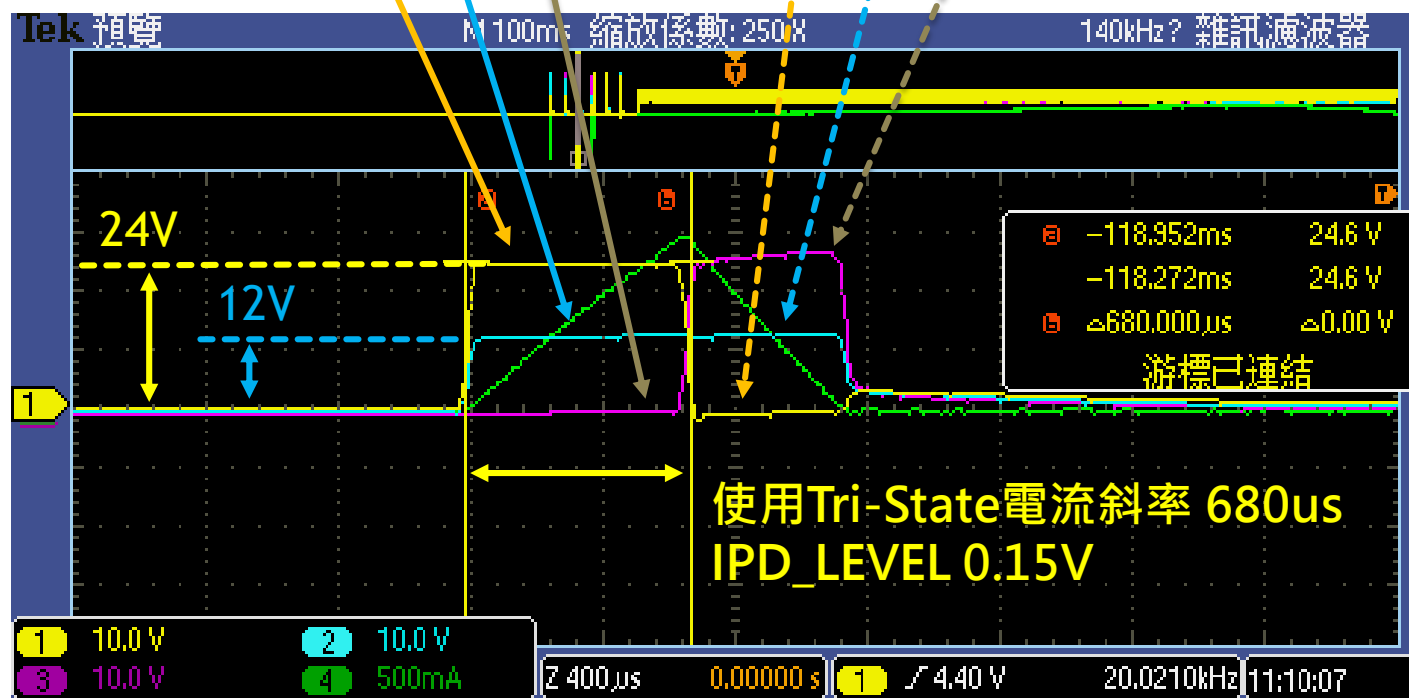
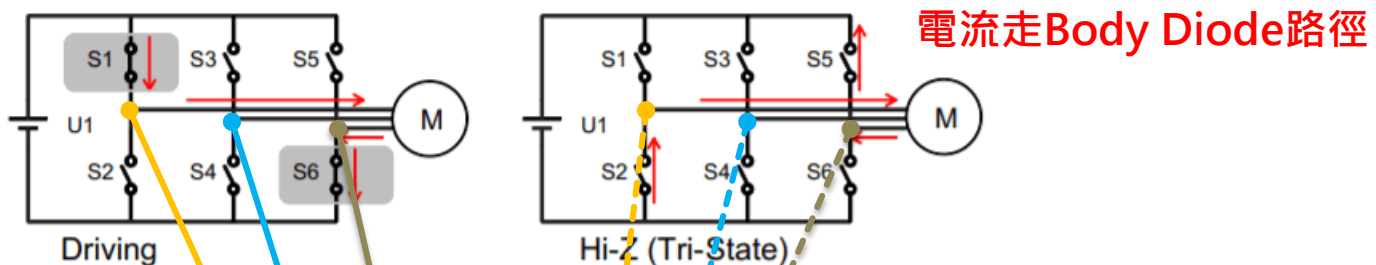
P+N、N+N Gate Driver 需要下臂預充電 20 ~ 30ms

當啟用IPD時，需先WatchDog Disable，避免IPD過程中觸發WatchDog

LatestTheta初始角度，因馬達不同可作微調，
目前參數為預設建議值。
IPDAdvanceAng作為進相角調整。

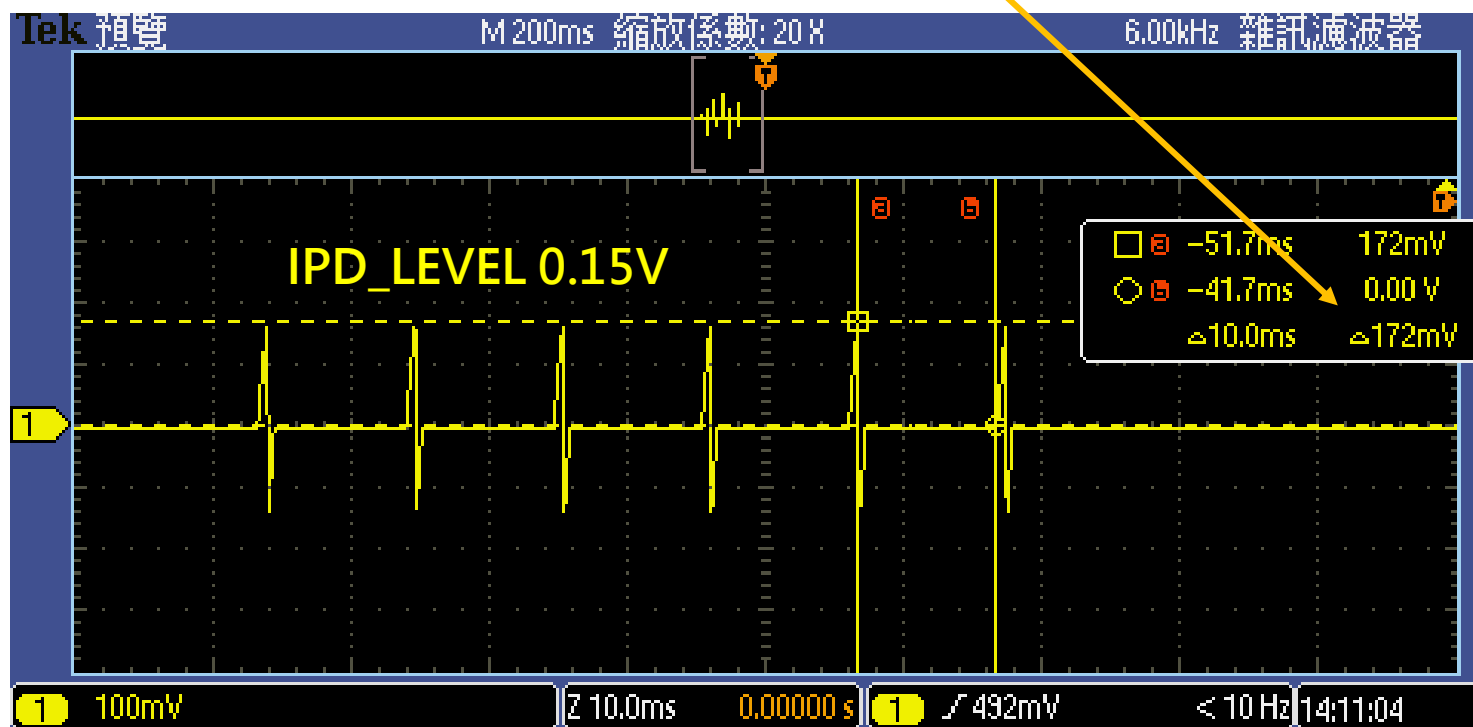


IPD程式流程 (3)



IPD程式流程 (4)

Set IPD LEVEL	
I_SHORT	0.15V
AOCPEN	Enable
DOCPEN	Disable
IPD Path Select	IPD Current Compare from AOC Path
Set IPD IAECYC	
IAECYC	24MHz



MDRFD0應用程式支援項目

項目	MDRFxx	MDSFxx	備註
Vsp控制	○	○	
電流控制	○	○	
轉速控制	○	○	
功率控制	○	○	MDRFD0使用上，注意Code Size
功率限制	○	○	MDRFD0使用上，注意Code Size
正反轉控制	○	○	
初始位置偵測	○	○	
順逆風偵測	○	○	MDSF40利用Cap計算轉速 MDRFD0利用Pwm中斷計算轉速
IR Decoder	×	○	MDRFD0需要用Cap計算解碼
斷電記憶	×	○	MDRFD0需要外部EEPROM

