單相直流-交流轉換器實驗模組

PEK-110

使用手冊

固緯料號 **NO. 82EK-11000M01**

ISO-9001 認證企業

GW INSTEK

本手冊所含資料在印製之前已經過校正，但因固緯電子實業股份有限公司不斷改善產品，所以保留未來修改產品規格、特性以及保養維修程式的權利，不必事前通知。

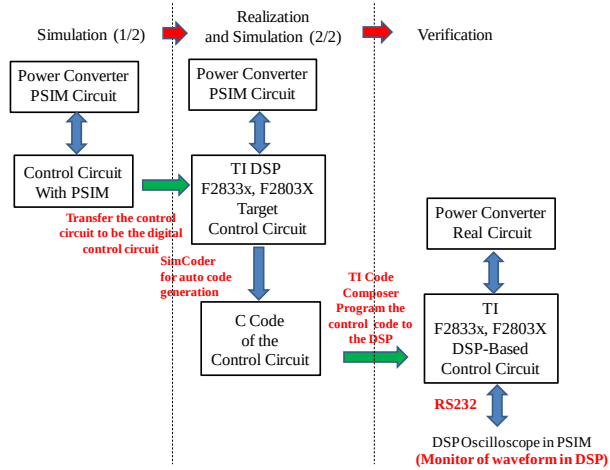
固緯電子實業股份有限公司
新北市土城區中興路 **7-1** 號

目錄

簡介

電力轉換器採用數位控制是目前工業界產品的發展趨勢，數位控制可以提升電力轉換器的功能及性能，提高產品的附加價值，而且越來越多的電力轉換產品已開始採用數位控制技術。本教具的實施方式如圖 1.1 所示，目的在提供電力轉換器採用數位控制的學習平台，讓使用者透過 PSIM 軟體，除以模擬方式學習電力轉換器的原理、分析及設計外，亦可透過 PSIM 之 SimCoder 工具將控制電路轉換為數位控制程式，並可實際將以 DSP 取代之電路再作一次模擬，最後並可將透過模擬驗證過之控制程式燒錄於 DSP 晶片中，再透過 DSP 作控制及通訊，以驗證所設計電路及控制器之正確性。

圖 1.1 本教具使用程序



本教具的特點主要包含以下幾項

1. 可同時提供電力電子之分析、設計、模擬與實作驗證。

2. 在 PSIM 下以建立硬體電路的方式完成程式撰寫並燒錄程式，使完全不會 DSP 韌體撰寫的學習者亦能輕鬆完成程式撰寫，快速進入數位控制領域。

3. 本教具提供完備的實驗教材，包括 SimCoder 使用，以建立硬體方式撰寫程式的方法、詳細說明教具各部份電路，詳盡的實驗電路原理與設計，PSIM 電路模擬檔，DSP 硬體規劃及設定，程式燒錄方法等。

4. 本教具提供完整之教學投影片供教師及使用者參考。

5. 本教具後續若有增加實驗項目，免費提供之前購買者使用。

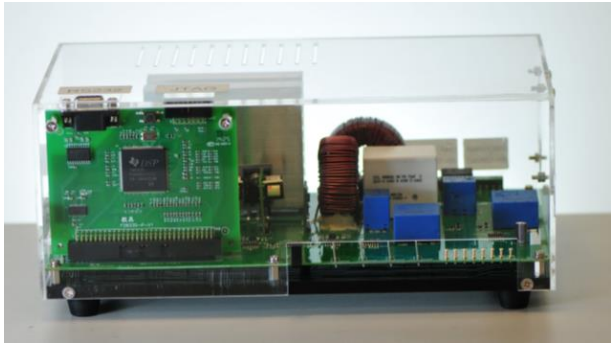
單相 Inverter 模組教具簡介

單相 Inverter 實驗模組如圖 1.2 所示，目前可提供以下 5 個實驗

1. 單電壓極性切換正弦式 PWM (Unipolar voltage switching SPWM)
2. 雙迴路電感電流控制之獨立式變流器(Stand alone inverter with dual loop inductor current control)
3. 單相市電並聯變流器(Grid connected single-phase inverter)
4. 無橋式 PFC AC-DC 轉換器(Totem Pole Bridgeless PFC AC-DC converter)
5. 全橋式 AC-DC 切換式整流器(Full-bridge AC-DC switching rectifier)

圖 1.2

單相 Inverter 實驗模組



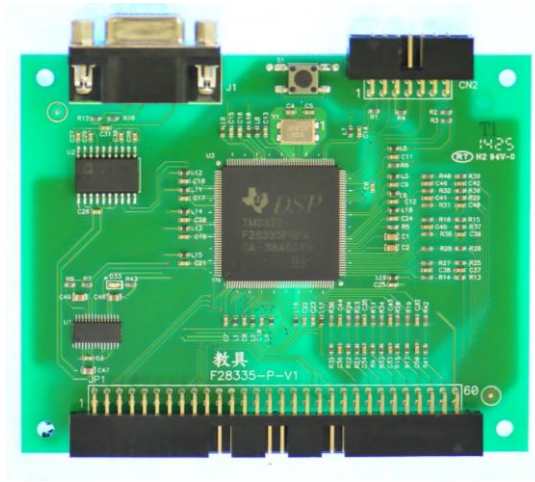
這些實驗模組除 Inverter 主電力電路外尚包括以下組件

DSP 控制模組

- 提供 TI F28335 模組。
- 各模組均具備隔離之 RS-232 通訊介面，可在實驗過程中將 DSP 內部信號傳回 PSIM 上觀測。

圖 1.3

DSP 控制模組



輔助電源模組

輸入電壓範圍為 100~250Vac 並提供+15V, -15V, 12V, 5V 等多組隔離電源輸出，總計最高 23W 之輸出，規格如表 1.1。

Description	Symbol	Min	Typ	Max	Units
Input					
Voltage	V_{IN}	100		250	VAC
Frequency	f_{LINE}	47	50/60	63	Hz
Output					
Output Voltage 1	V_{OUT1}	11.4	12	12.6	V
Output Current 1	I_{OUT1}	0.1	0.5	0.6	A
Output Voltage 2	V_{OUT2}	11.4	12	12.6	V
Output Current 2	I_{OUT2}	0.1	0.5	0.6	A
Output Voltage 3	V_{OUT3}	14.25	15	15.75	V
Output Current 3	I_{OUT3}	0.1	0.2	0.24	A
Output Voltage 4	V_{OUT4}	-14.25	-15	-15.75	V
Output Current 4	I_{OUT4}	-0.1	-0.2	-0.24	A
Output Voltage 5	V_{OUT5}	4.75	5	5.25	V
Output Current 5	I_{OUT5}	0.5	1	1.2	A
Total Output Power	P_{OUT}	7.505	23	28.98	W

表 1.1

圖 1.4

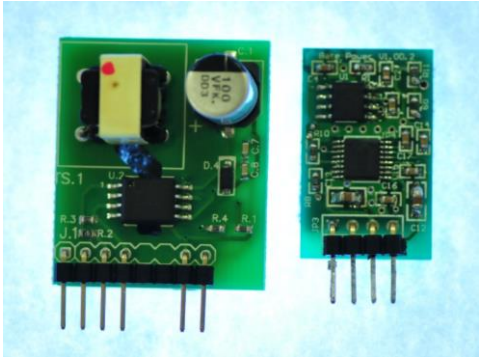
Flyback 輔助電源模組

驅動電源與開關驅動器模組

- 利用本模組只要一 12V 隔離電源便可提供任意多組輸出之隔離電源，解決複雜多組隔離驅動電源問題。
- 驅動器模組可提供高頻及高電流(2A)之驅動能力，並具備米勒效應之保護電路以避免誤動作發生。

圖 1.5

驅動電源與開關驅動器模組



JTAG 燒錄模組

提供隔離保護燒錄，避免因未隔離而在實驗過程導致電腦燒毀。

圖 1.6

JTAG 燒錄模組

實驗目標

本教具以電路分析、設計、模擬、實驗等過程進行問題導向學習，根據轉換器規格進行量化設計其電力電路與控制器，並藉由 PSIM 模擬驗證，SimCoder 撰寫程式過程，使讀者徹底深入了解 inverter 之相關技術，培養以下能力

1. 電力轉換器之分析與設計能力。
2. PSIM 電路模擬能力。
3. 電力轉換器之控制器設計能力。
4. DSP 之數位控制技術(透過 SimCoder 輔助完成程式撰寫)。
5. 硬體與韌體之規劃及整合能力。
6. Step by step 完成電路製作與驗證能力。

本書章節說明

本書之章節安排如下

簡介	簡略介紹本教具之電路組成、實驗方式、實驗目標、本書各章之內容等。
PSIM 簡介	簡略介紹 PSIM 之組成與功能，讓使用者可以更加了解 PSIM 能夠協助轉換器電路分析與設計之工作內容。
教具硬體及設備說明	詳細介紹本教具中各個組成電路之工作原理與設備使用之方法。
實驗 1 單電壓極性切換正弦式 PWM	學習 Sinusoidal PWM 單電壓極性切換之原理、Inverter 模組之開迴路電壓及電流量測方法，TI F28335 DSP IC 腳位設定、DSP 之 PWM 及 A/D 模組設定、RS232 監控 DSP 內部信號之方法等。
實驗 2 雙迴路電感電流控制之獨立式變流器	學習單相全橋式 inverter 之模式化方法、電流迴路及電壓迴路控制器設計、RMS 電壓迴路設計、inverter 之硬體規劃及 SimCoder 程式撰寫等。
實驗 3 單相市電並聯變流器	學習單相市電並聯 inverter 之鎖相迴路方法、電流迴路及電壓迴路控制器設計、硬體規劃及並網之 SimCoder 程式撰寫等。
實驗 4 無橋式 PFC AC-DC 轉換器	學習 Totem pole 無橋式 PFC 轉換器之原理、電流迴路及電壓迴路控制器設計、硬體規劃及 PFC 之 SimCoder 程式撰寫等。
實驗 5 全橋式 AC-DC 切換式整流器	學習全橋式 AC-DC 切換式整流器之原理、電流迴路及電壓迴路控制器設計、硬體規劃及 SimCoder 程式撰寫等。

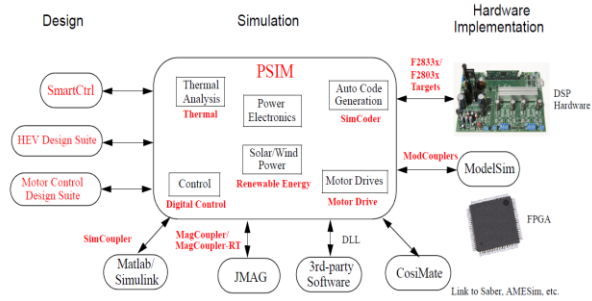
附錄 A SimCoder 概述	詳細介紹 PSIM 之 SimCoder 以及 TI F28335 Target 之使用方法。

PSIM 簡介

PSIM 為一專門為各式電力電子、馬達驅動及電力轉換等系統所設計之模擬軟體，其特點為：功能全面、元件完整、模擬速度快、模擬結果精確、非常容易使用等，為目前國際上學術界及業界常用之教學與研究軟體，因此本書藉此軟體當成平台，使讀者能與國際之研究及教學同步。

PSIM 具有模擬、設計及硬體電路實現等全方位能力，提供之功能如圖 2.1 所示，除了其主體可以提供電力電子電路模擬外，亦包含了以下模組

圖 2.1 PSIM 提供之模擬功能



Motor Driver 模組	包含了直流馬達、直流無刷馬達、鼠籠式感應馬達、繞線式感應馬達、永磁同步馬達、同步馬達、開關磁阻馬達及各式速度、位置、轉矩迴授裝置、各式機械負載及驅動源裝置等，可以作各式馬達及發電機等應用系統之模擬。
Digital Control 模組	包含了各式離散元件如 zero-order hold, z-domain 轉移函數、數位濾波器、quantization blocks 等可執行數位控制及分析的元件。

SimCoupler 模組	用以作為 PSIM 與 Simulink 之介面，使 PSIM 與 Simulink 可作 Co-simulation(同時模擬)，讓原先使用 Simulink 的使用者亦可使用原先發展的技術，因使用 PSIM 改善其模擬速度及收斂性，亦讓原 PSIM 的使用者可以透過 Simulink 使用 Matlab 豐富的 Toolbox 功能。
Thermal 模組	提供實際功率半導體元件特性的模型，可以計算功率半體元件的損失，作為計算溫升及設計散熱機構參考，使用者亦可以根據實際元件資料手冊自己建立考慮 Thermal 的功率半導體元件。
Renewable Energy 模組	包含太陽光電(PV)模組、風機(wind turbine)及電池模型。
SimCoder 模組	可以將控制電路自動轉化為 C 程式，並可透過 TI Composer 對 DSP 晶片作燒錄，透過 PSIM 提供一個硬體與韌體人員可以彼此討論的平台，建立更緊密協同開發的環境。
F2833X Target	包含一些 TI DSP F2833X 之元件資料庫，可以自動產生燒錄 F2833X 的程式。
F2803X Target	包含一些 TI DSP F2803X 之元件資料庫，可以自動產生燒錄 F2803X 的程式。
MagCoupler 模組	提供 PSIM 與磁路分析軟體 JMAG 之介面，使二者可作 Co-simulation。
MagCoupler-RT 模組	提供 PSIM 與磁路分析軟體 JMAG 資料檔之連結。
ModCoupler 模組	提供 PSIM 與 ModelSim 之介面，使二者可作 Co-simulation。其包括兩個版本，ModCoupler-VHDL 支援 VHDL 程式以及 ModCoupler-Verilog 支援 Verilog 程式

HEV Design Suite	預設一些設計樣版(template)提供混合電動車(HEV)之傳動系統(powertrain system)輔助設計
Motor Control Design Suite	預設一些設計樣版(template)提供感應馬達、線性及非線性永磁同步馬達驅動器的輔助設計

除此 PSIM 亦提供與 CosiMate 之連結，藉由 CosiMate 平台可與許多軟體作 co-simulation，包括 Matlab/Simulink, ModelSim, Saber (from Synopsys), Easy5 and Adams (from MSCSoftware), Inventor (from Autodesk), AMESim (from LMS), GT-Power (from Gamma Technologies)等等，細節請連結 www.chiastek.com。

教具硬體及設備說明

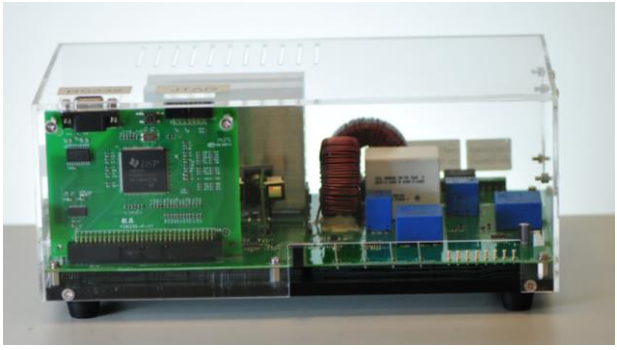
本教學實驗之教具為單相變流器，輸入設備為 GW PSW 250-4.5 360W，輸入電壓操作範圍為 60V~200V，輸出設備為 GW GPL-100，可操作在整流性負載或電阻性負載，量測輸出功率使用 GW GPM-8212，示波器為 GW GDS-2304A，在市電並聯實驗中以 GW APS-7050 模擬市電，這些設備在本章會概略介紹。

功率級電路

本實驗教具之實體如圖 3.1，電路圖為圖 3.2，操作之輸入電壓因安全性考量故設定在 70V，避免使用者在尚未熟悉電路操作的情況下發生危險，變流器的輸入端會先經過一個 5A 的保險絲，接著一個 330uF/450V 輸入電解電容，隨後一個由四顆 MOS 組成的全橋式變流器，MOS 的驅動電路將在後面說明，緊接一個 L-C 組成的二階低通濾波器(電感為 661.5uH，電容為 10uF/400V)，以衰減變流器之高頻切換輸出，使輸出電壓為低頻之正弦波，其 RMS 值為 40V。

圖 3.1

單相 Inverter 實驗模組



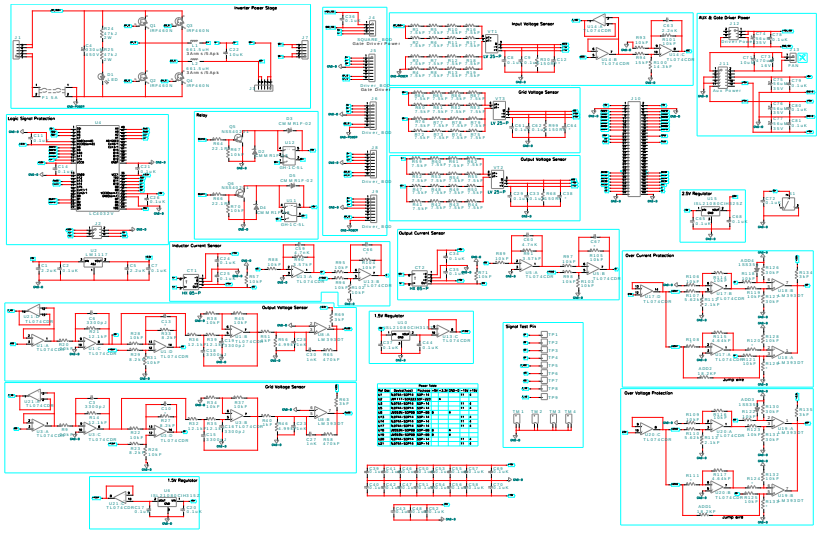


圖 3.2 單相 Inverter 電路圖

在控制及回授電路方面，輸入電壓(V_{DC})，輸出電壓(V_0)，輸出電流(I_0)，電感電流(I_L)與市電電壓(V_S)等五個參數經過取樣後送入 DSP，以下分別說明這五個參數送入 DSP 前的取樣過程。

1. 由圖 3.2 摘錄輸入電壓的取樣過程，如圖 3.3，經 16 顆電阻衰減 1/30k 倍送入型號為 LEM LV 25-P 的取樣 IC，經 IC 的轉換比例放大 2.5 倍後，再乘上 150Ω 後得到 VDCS 信號，再經倍率 0.976 的 OP 放大器後送入 DSP，取樣衰減的比例如(3.1)所示。

$$Gain = \frac{1}{30k} \times 2.5 \times 150 \times 0.976 = 1.22 \times 10^{-2} \qquad (3.1)$$

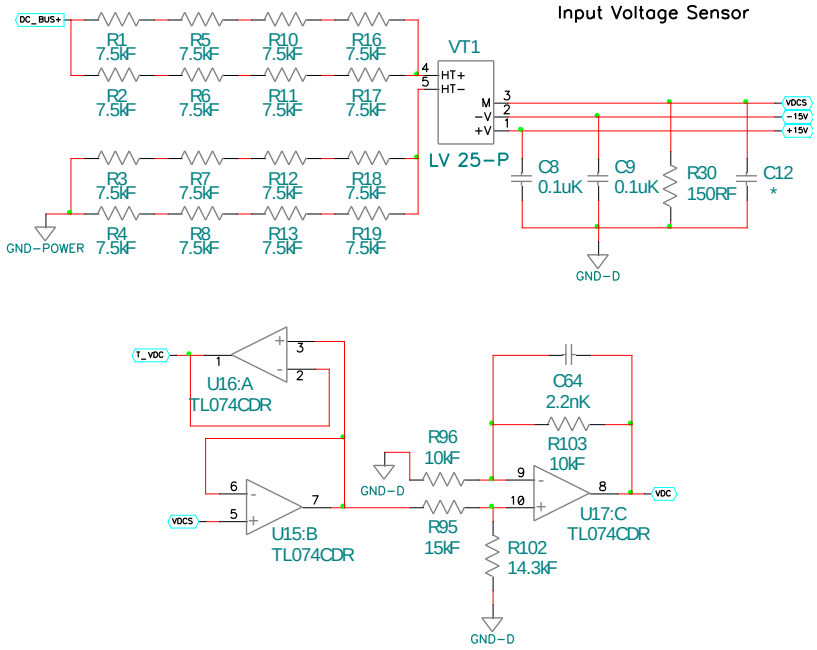


圖 3.3 輸入電壓取樣電路

2. 由圖 3.2 摘錄輸出電壓的取樣過程，如圖 3.4，經 16 顆電阻衰減 1/30k 倍送入型號為 LEM LV 25-P 的取樣 IC，經 IC 的轉換比例放大 2.5 倍後，再乘上 150Ω 後，經倍率 0.496 的 OP 放大器後送入 DSP，取樣衰減的比例如(3.2)所示。因輸出電壓為交流信號，但送入 DSP 的信號只能從 0~3V，必須疊加一個 1.5V 的準位使其能符合要求，即為 U1 之第 12 腳位。

$$Gain = \frac{1}{30k} \times 2.5 \times 150 \times 0.496 = 6.2 \times 10^{-3} \qquad (3.2)$$

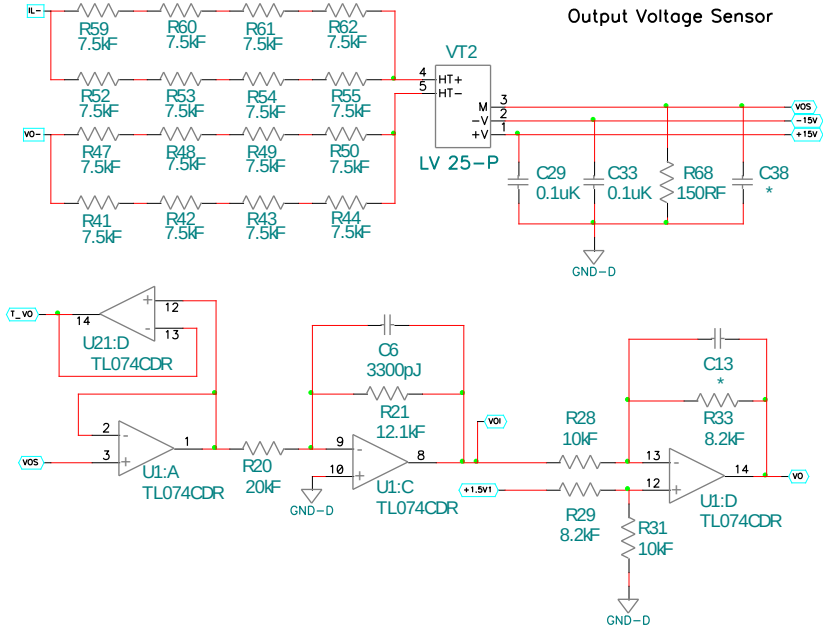


圖 3.4 輸出電壓取樣電路

3. 由圖 3.2 摘錄輸出電流的取樣過程，如圖 3.5，經電流感測 IC LEM HX 05-P，其轉換比例為 8×10^{-5} 倍，再乘上 10kΩ，再經倍率 0.357 的 OP 放大器後送入 DSP，取樣衰減的比例如(3.3)所示。因輸出電流為交流信號，但送入 DSP 的信號只能從 0~3V，必須疊加一個 1.5V 的準位使其能符合要求，即為 U5 之第 5 腳位。

$$Gain = (8\times10^{-5})\times10k\times0.357 = 0.286 \qquad (3.3)$$

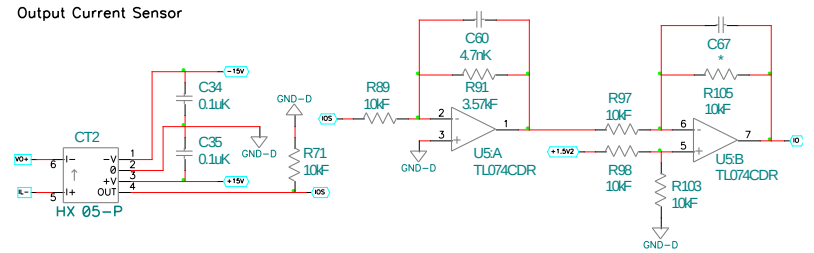


圖 3.5 輸出電流取樣電路

4. 由圖 3.2 摘錄電感電流的取樣過程，如圖 3.6，經電流感測 IC LEM HX 05-P，其轉換比例為 8×10^{-5} 倍，再乘上 10kΩ，再經倍率 0.36 的 OP 放大器後送入 DSP，取樣衰減的比例如(3.4)所示。因電感電流為交流信號，但送入 DSP 的信號只能從 0~3V，必須疊加一個 1.5V 的準位使其能符合要求，即為 U13 之第 5 腳位。

$$Gain = (8\times10^{-5})\times10k\times0.36 = 0.288 \qquad (3.4)$$

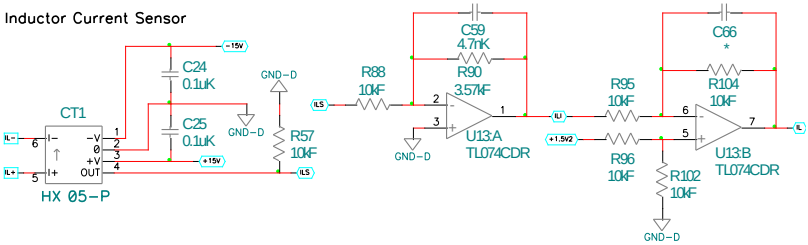


圖 3.6 電感電流取樣電路

5. 由圖 3.2 摘錄市電電壓的取樣過程，如圖 3.7，經 16 顆電阻衰減 1/30k 倍送入型號為 LEM LV 25-P 的取樣 IC，經 IC 的轉換比例放大 2.5 倍後，再乘上 150Ω 後，經倍率 0.496 的 OP 放大器後送入 DSP，取樣衰減的比例如(3.5)所示。因市電電壓為交流信號，但送入 DSP 的信號只能從 0~3V，必須疊加一個 1.5V 的準位使其能符合要求，即為 U3 之第 12 腳位。

$$Gain = \frac{1}{30k}\times2.5\times150\times0.496 = 6.2\times10^{-3} \qquad (3.5)$$

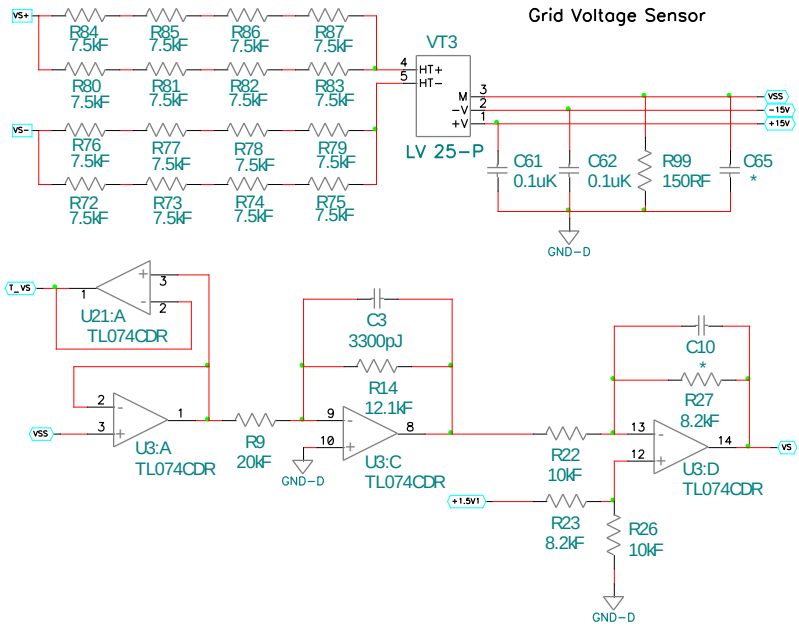


圖 3.7 市電電壓取樣電路

硬體電路有設計過電壓與過電流保護，常態為高準位，一旦電壓或電流過高則輸出變為低準位，觸發保護動作使變流器停止運作以免損壞。電路如圖 3.8。

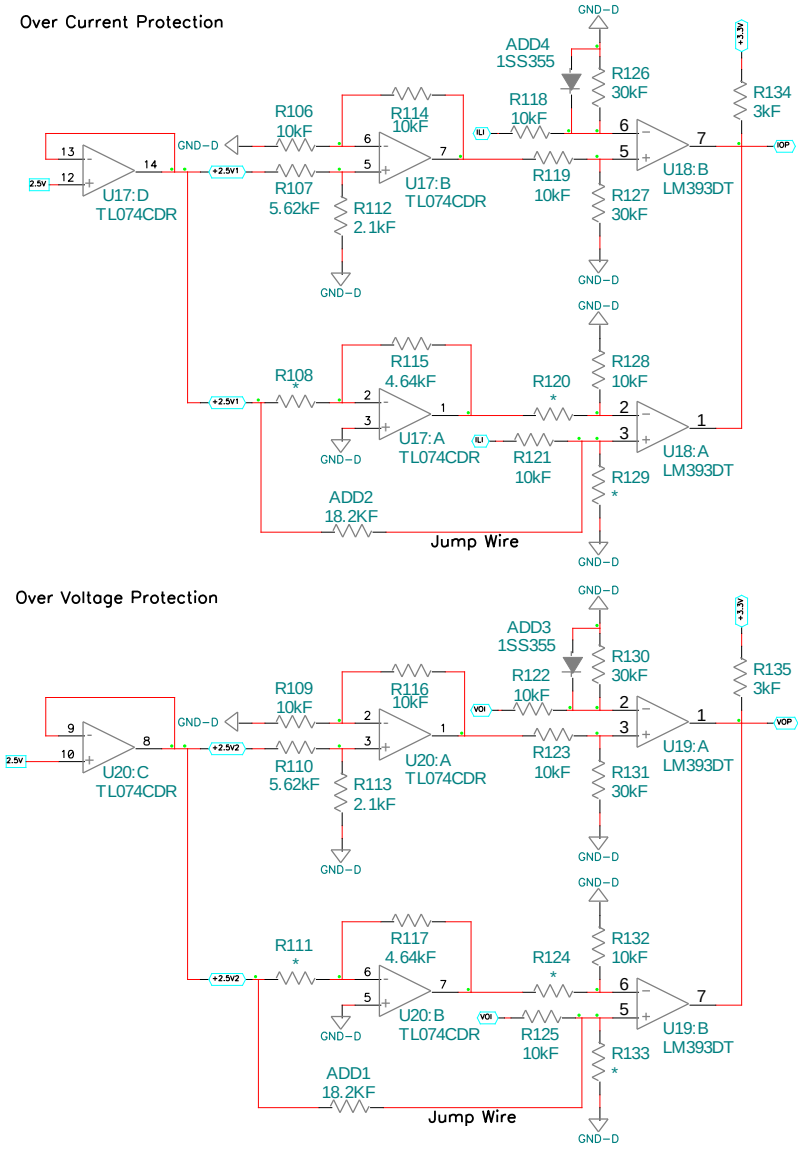


圖 3.8 過電壓與過電流保護電路

單相變流器教具提供下列測試點供使用者量測：

1. DSP 送出 PWM1~PWM4 至 CPLD LC4032V 後得到信號 Q1~Q4，再經過驅動電路進而驅動 MOS，信號 Q1~Q4 可由量測點觀測。

2. 輸入電壓(VDC)經 16 顆電阻衰減 1/30k 倍送入取樣 IC，經 IC 的轉換比例放大 2.5 倍，再乘上 150Ω 後，即為量測點的觀測值，衰減倍率如(3.6)：

$$Gain = \frac{1}{30k} \times 2.5 \times 150 = 0.0125 \quad (3.6)$$

3. 電感電流(I_L)經轉換比例為 8×10^{-5} 倍的電流感測器，再乘上 10kΩ，即為量測點的觀察值，衰減倍率如(3.7)：

$$Gain = (8\times10^{-5}) \times 10k = 0.8 \quad (3.7)$$

4. 輸出電流(I_0)轉換比例為 8×10^{-5} 倍的電流感測器，再乘上 10kΩ，即為量測點的觀察值，衰減倍率如(3.8)：

$$Gain = (8\times10^{-5}) \times 10k = 0.8 \quad (3.8)$$

5. 輸出電壓(V_0)經 16 顆電阻衰減 1/30k 倍送入取樣 IC，經 IC 的轉換比例放大 2.5 倍，再乘上 150Ω 後，即為量測點的觀測值，衰減倍率如(3.9)：

$$Gain = \frac{1}{30k} \times 2.5 \times 150 = 0.0125 \quad (3.9)$$

6. 市電電壓(Vs)經 16 顆電阻衰減 1/30k 倍送入取樣 IC，經 IC 的轉換比例放大 2.5 倍，再乘上 150Ω 後，即為量測點的觀測值，衰減倍率如(3.10)：

$$Gain = \frac{1}{30k} \times 2.5 \times 150 = 0.0125 \quad (3.10)$$

DSP 控制電路

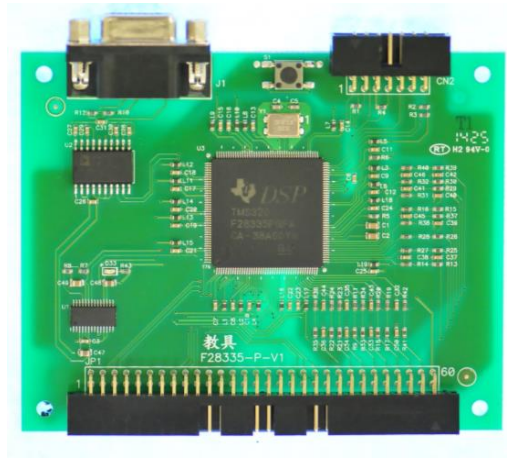
DSP 控制電路以 TI TMS320F28335 實現之硬體如圖 3.9，電路圖為圖 3.10，由雙輸出穩壓 IC 提供 3.3V 與 1.8V，3.3V 即為 28335 IC 的工作電源。信號送入 DSP 前會先經過二極體箝位電路以確保進入 IC 之電壓都在 0~3V 之間避免損壞 DSP。透過隔離的 RS232 通信介面,在實驗中可即時將 DSP 內部信號傳回 PSIM 中的示波器觀測。F28335 控制板各腳位輸出定義如下表

	Pin		
+5V in	1	2	+5V in
GND	3	4	GND
GPIO-00 / EPWM-1A	5	6	GPIO-01 / EPWM-1B / MFSR-B
GPIO-02 / EPWM-2A	7	8	GPIO-03 / EPWM-2B / MCLKR-B
GPIO-04 / EPWM-3A	9	10	GPIO-05 / EPWM-3B / MFSR-A / ECAP-1
GPIO-06 / EPWM-4A / SYNCI / SYNCO	11	12	GPIO-07 / EPWM-4B / MCLKR-A / ECAP-2
GPIO-08 / EPWM-5A / CANTX-B / ADCSOC-A	13	14	GPIO-09 / EPWM-5B / SCITX-B / ECAP-3
GPIO-10 / EPWM-6A / CANRX-B / ADCSOC-B	15	16	GPIO-11 / EPWM-6B / SCIRX-B / ECAP-4
GPIO-48 / ECAP5 / XD31 (EMIF)	17	18	GPIO-49 / ECAP6 / XD30 (EMIF)
GPIO-50	19	20	GPIO-51
GPIO-12 / TZ1n / CANTX-B / MDX-B	21	22	GPIO-13 / TZ2n / CANRX-B / MDR-B
GPIO-15 / TZ4n / SCIRX-B / MFSX-B	23	24	GPIO-14 / TZ3n / SCITX-B / MCLKX-B
GPIO-24 / ECAP1 / EQEPA-2 / MDX-B	25	26	GPIO-25 / ECAP2 / EQEPB-2 / MDR-B
GPIO-26 / ECAP3 / EQEPI-2 / MCLKX-B	27	28	GPIO-27 / ECAP4 / EQEPS-2 / MFSX-B
GPIO-16 / SPISIMO-A / CANTX-B / TZ-5	29	30	GPIO-17 / SPISOMI-A / CANRX-B / TZ-6
GPIO-18 / SPICLK-A /	31	32	GPIO-19 / SPISTE-A / SCIRX-B

SCITX-B			
GPIO-20 / EQEP1A / MDX-A / CANTX-B	33	34	GPIO-21/ EQEP1B/ MDR-A/ CANRX-B
GPIO-22 / EQEP1S / MCLKX-A / SCITX-B	35	36	GPIO-23/ EQEP1I/ MFSX-A / SCIRX-B
GPIO-28 / SCIRX-A / -- / TZ5	37	38	GPIO-29 / SCITX-A / -- / TZ6
GPIO-30 / CANRX-A	39	40	GPIO-31 / CANTX-A
GPIO-32 / I2CSDA / SYNCI / ADCSOCA	41	42	GPIO-33 / I2CSCL / SYNCO / ADCSOCB
ADCIN-B7	43	44	ADCIN-A7
ADCIN-B6	45	46	ADCIN-A6
ADCIN-B5	47	48	ADCIN-A5
ADCIN-B4	49	50	ADCIN-A4
ADCIN-B3	51	52	ADCIN-A3
ADCIN-B2	53	54	ADCIN-A2
ADCIN-B1	55	56	ADCIN-A1
ADCIN-B0	57	58	ADCIN-A0
GND	59	60	GND

圖 3.9

DSP 控制電路



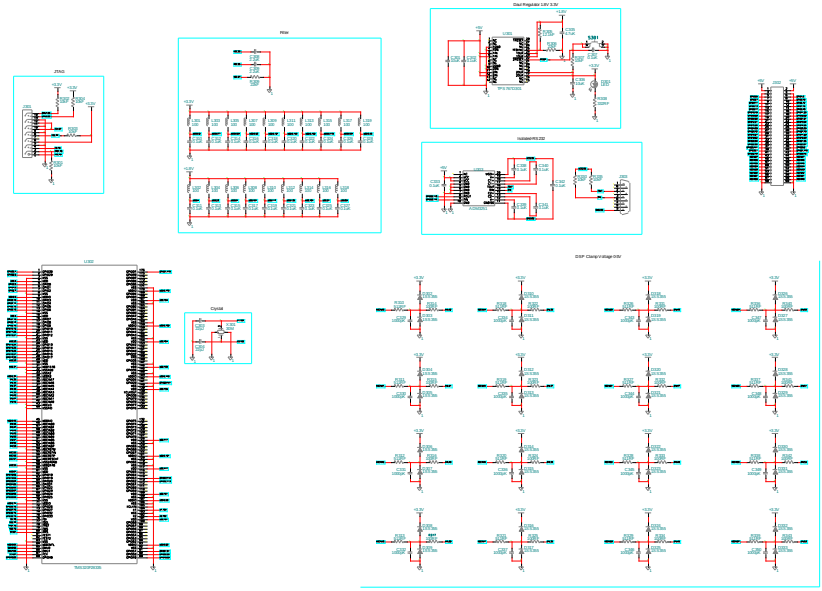
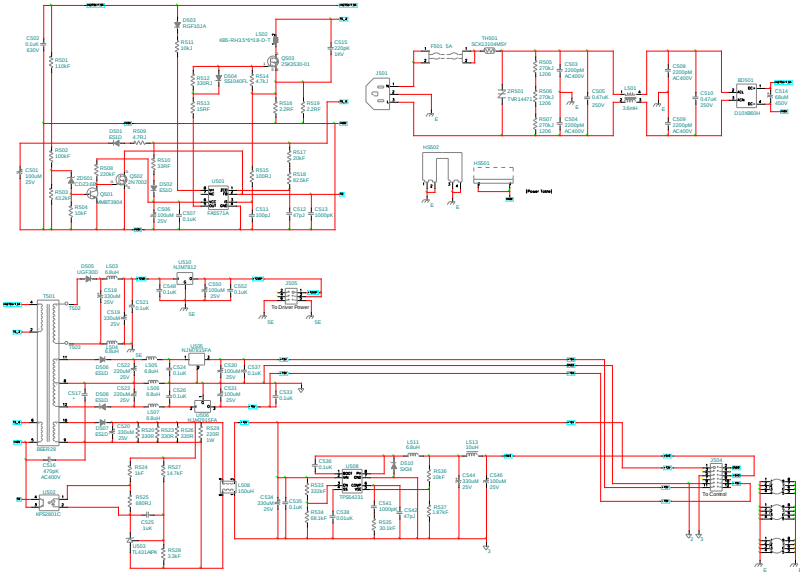


圖 3.10 F28335 電路圖

輔助電源

此模組以 Flyback 為設計架構，輸入電壓範圍為 100~250V，輸出為三組不共地的隔離電源，分別是 (1)12V (2)12V, 5V (3)15V, -15V 實體為圖 3.11，電路如圖 3.12

圖 3.11
輔助電源



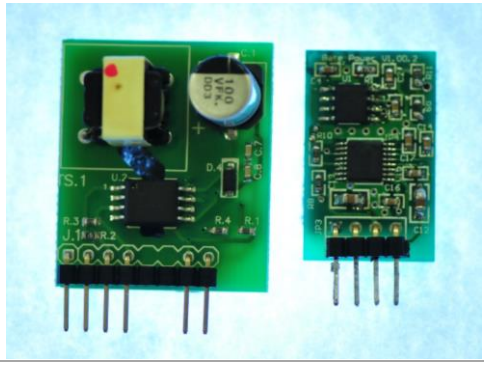
3.12 輔助電源電路

驅動電路

驅動電源模組由 Gate Driver 板與 Gate Driver Power 板組成，可提供多組隔離電源，圖 3.13 左為 Gate Driver，右為 Gate Driver Power，圖 3.14, 3.15 為電路圖。輸入一個 12V 電壓至 Gate Driver Power，其輸出為±12V 之方塊波。Gate Driver 的輸入為此±12V 之方塊波與由 DSP 產生的 PWM 信號，輸出為驅動 MOS 的信號。Gate Driver 藉由變壓器與光耦合驅動 IC 達到隔離的目的。

圖 3.13

電路驅動電路模組



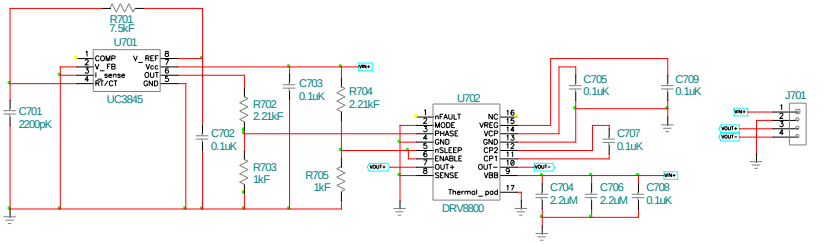


圖 3.14 Gate Driver Power 電路圖

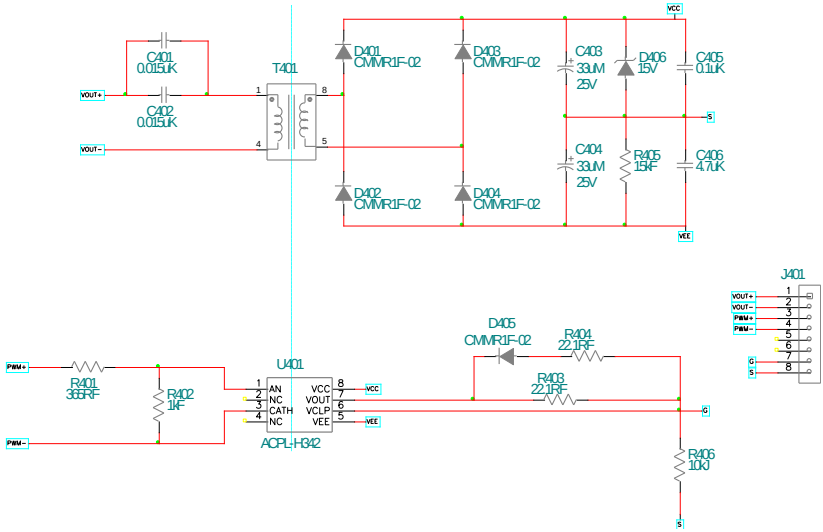


圖 3.15 Gate Driver 電路圖

JTAG 燒錄電路

此模組可將程式碼由電腦端燒錄至 DSP 晶片，硬體電路如圖 3.16，電路圖為圖 3.17，以 USB 與電腦端連接，JTAG 與 DSP 端連接。

圖 3.16

USB_JTAG 燒錄電路

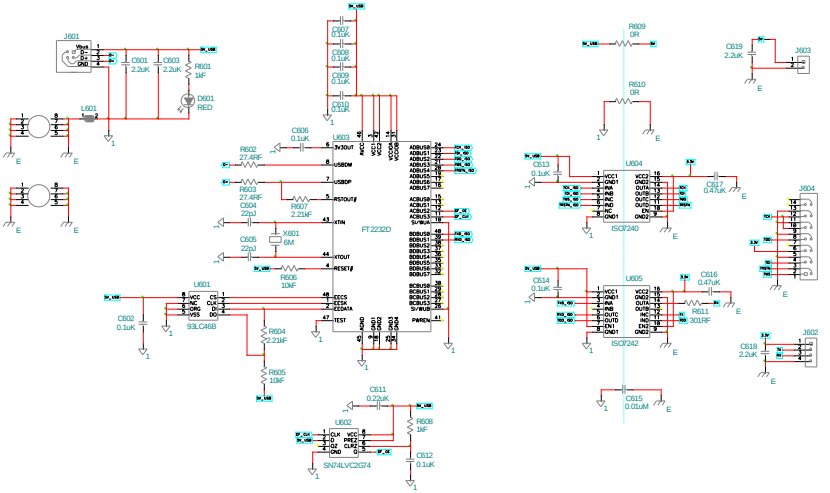


圖 3.17 USB_JTAG 電路

實驗 1 單電壓極性切換正弦式 PWM

實驗目的

- 學習 Sinusoidal PWM(SPWM)切換之原理
- Inverter 模組之開迴路電壓及電流量測方法
- TI F28335 DSP IC 腳位設定
- DSP 之 PWM 及 A/D 模組設定
- RS232 監控 DSP 內部信號之方法
- 熟悉本實驗之硬體電路操作

實驗原理

正弦式 PWM(SPWM)變流器之原理

單相全橋式變流器如圖 4.1 所示，其開關之切換控制乃由一低頻正弦波信號($V_{control}$)與一高頻之三角波(V_{tri})比較以產生開關之觸發信號，稱為正弦式(sinusoidal) PWM (SPWM)，L-C 則形成一二階的低通濾波器，用以衰減變流器輸出之高頻切換項，使輸出電壓為低頻之正弦波。

交流負載可分為線性及非線性負載，線性負載包括電阻性(電壓及電流同相)、電感性(電流相位落後電壓)及電容性(電流相位領前電壓)，非線性負載則為電流失真之整流性負載。為同時提供上述負載，變流器之輸出必須同時具備四象限的工作能力，典型負載為電感性之變流器的輸出電壓及電流波形如圖 4.2(a)所示，一交流週期內變流器同時經歷四象限之操作。橋式轉換器(Bridge converter)具有四象限工作能力，因此是變流器常被採用的電路架構。單相全橋式變流器功率潮流(power flow)方向之定義如圖 4.1 之箭頭所標示，第 I 及第 III 象限功率 P_o>0，稱為變流模式(inverter mode)，第 II 及第 IV 象限功率 P_o<0，稱為整流模式(rectifier mode)。雖然以瞬時功率而言一週期同時經歷四象限，功率有進有出，但以平均功率而言，圖 4.2(a)為操作在變流模式，因其電流落後電壓的角度<90 度，反之若電流落後電壓的角度>90 度則操作在整流模式。全橋式變流器其開關具有如圖 4.2(b)所示之 8 種導通模式，例如在(T_{A+}, T_{B-})對角線開關導通時輸出電壓及電流均為正，P_o>0 其操作在 v_o-i_o 平面的第 I 象限；在(D_{A+}, D_{B-})對角線二極體導通時輸出電壓為正但電流為負，P_o<0 其操作在第 IV 象限；在(T_{A-}, T_{B+})對角線開關導通時輸出電壓及電流均為負，P_o >0 其操作在第 III 象限；在(D_{A-}, D_{B+})對角線二極體導通時輸出電壓為負但電流為正，P_o<0 其操作在第 II 象限。另外其具有四種輸出短路(v_o =0)之飛輪導通模式，如 i_o 軸上所標示的四種導通模式，包 i_o 電流為正時之(T_{A+}, D_{B+})、(D_{A-}, T_{B-})以及 i_o 電流為負時之(T_{A-}, D_{B-})、(D_{A+}, T_{B+})。

對於單相變流器而言，SPWM 可分為雙電壓極性切換(Bipolar-voltage switching)與單電壓極性切換(Unipolar-voltage switching)二方式，分述如下

A. 雙電壓極性切換

雙電壓極性切換的比較方式如圖 4.3(a)所示，全橋式電路對角線的開關為成對觸發，且同一臂之開關的觸發信號為互補，但需要加入一空白時間(Dead-time)，以防止開關切換瞬間同時導通。控制電壓 $v_{control}$ 與周期性鋸齒波 v_{tri} 比較決定開關之責任週期，當 $v_{control}>v_{tri}$ 時觸發 (T_{A+}, T_{B-})，無 i_o 方向為何將使 A 臂接到高準位(i_o >0 透過 T_{A+}, v_o <0 透過 D_{A+})而 B 臂接到低準位(i_o>0 透過 T_{B-}, i_o <0 透過 D_{B-})，因此 v_o = V_d；反之 $v_{control}<v_{tri}$ 時則觸發(T_{A-}, T_{B+})，無論 i_o 方向為何將使 A 臂接到低準位而 B 臂接到高準位，因此 i_o =-V_d。變流器輸出波形如圖 4.3(b)所示，v_o 在+V_d 與-V_d 二準位間切換，因此稱為雙電壓極性切換。

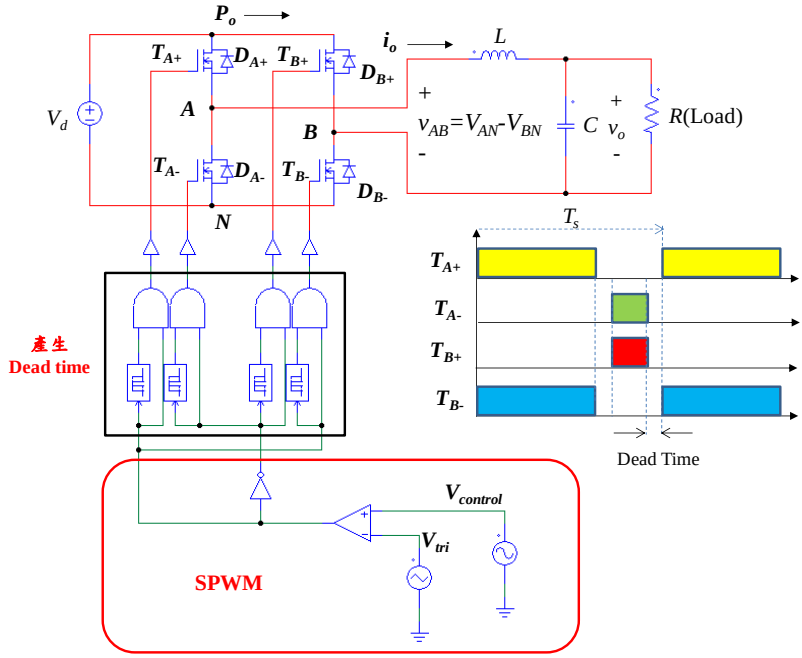
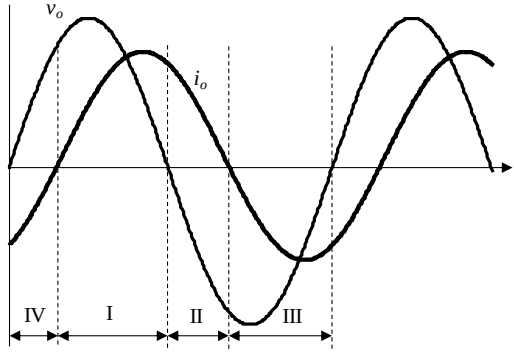


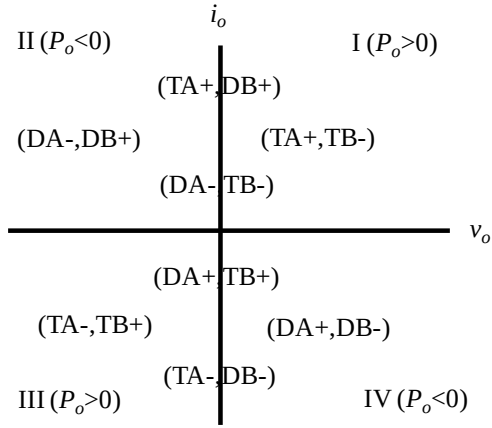
圖 4.1 單相全橋式變流器(使用雙電壓極性切換)

圖 4.2

全橋式轉換器之輸出波形及元件導通狀態



(a)電感性負載下之輸出電壓及電流波形



(b)全橋式變流器之 8 種導通狀態

SPWM 可以定義二參數：

振幅調制指數(Amplitude modulation index)

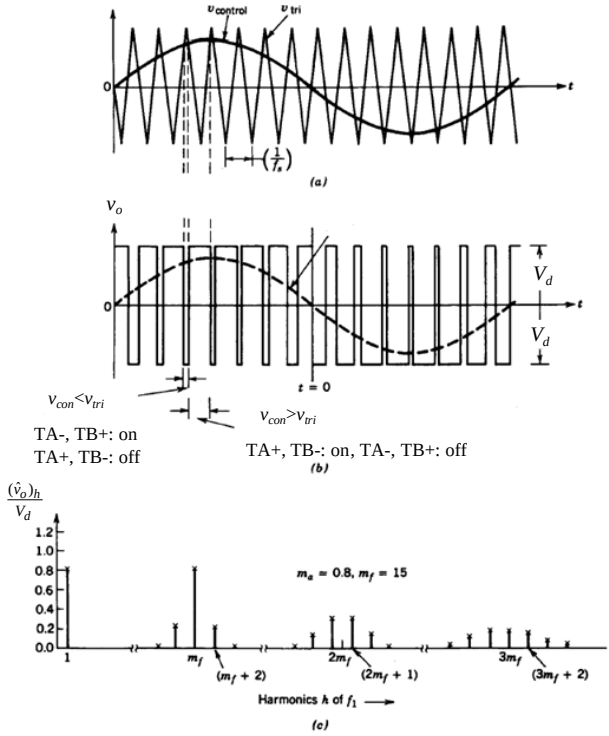
$$m_a = \frac{\hat{v}_{control}}{\hat{v}_{tri}} \tag{4.1}$$

頻率調制指數(Frequency modulation index)

$$m_f = \frac{f_s}{f_1} \tag{4.2}$$

其中 $\hat{v}_{control}$ 及 $\hat{v}_{tri}$ 分別為控制電壓與三角波的振幅，f_1 及 f_s 分別為控制電壓與三角波的頻率。輸出電壓 v_o 之頻譜如圖 4.3(c)所示，其基本波大小與 m_a 成正比，其他諧波則出現在 m_f 的整倍數及其邊帶附近。

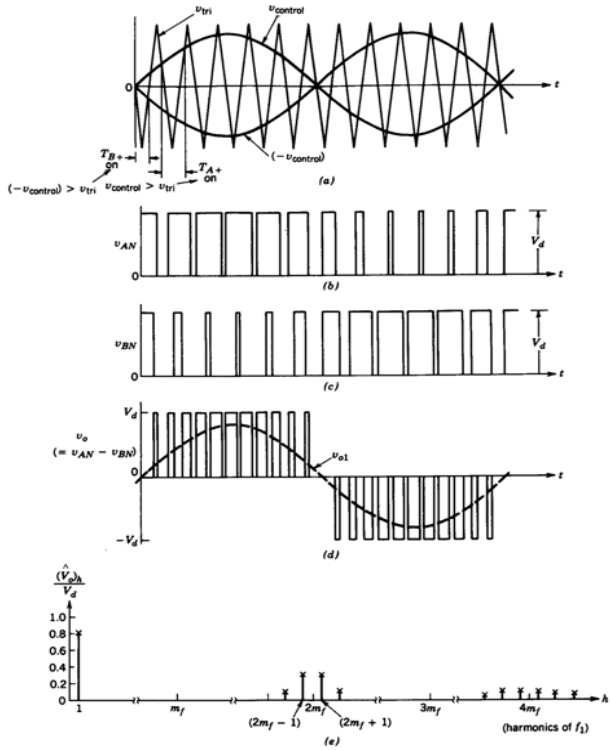
圖 4.3
雙電壓極性切換



B. 單電壓極性切換

單電壓極性切換的比較方式如圖 4.4(a)所示，A、B 二臂有各自的控制電壓分別與三角波比較，同一臂之開關的觸發信號為互補。二控制電壓為反相，$v_{controlA}$=+$v_{control}$，$v_{controlB}$=-$v_{control}$。藉由此比較結果所得變流器之輸出波形如圖 4.4(b)所示，v_o 在+V_d 與 0 或-V_d 與 0 準位間切換，因此稱為單電壓極性切換。單電壓極性切換使用到圖 4.2 中所有 8 個導通模式包括飛輪模式。輸出電壓 v_o 之頻譜如圖 4.4(c)所示，其基本波大小與 ma 成正比，其他諧波則出現在 2m_f 及 2m_f 整倍數的邊帶附近。與雙電壓極性切換之輸出相較，基本波大小為相同，但具有兩倍等效切換頻率以及較低的電壓變化量，因此可以使用較小的 LC 濾波元件，故較常被採用。

圖 4.4
單電壓極性切換



單相 SPWM 變流器輸出電壓基本波與 m_a 之關係如圖 4.5 所示，m_a <1 稱為線性區，輸出電壓基本波與 m_a 成正比。1< m_a <3.24 稱為過調制區，輸出電壓基本波振幅($\frac{(\hat{v}_o)_1}{V_d}$)超過 1。$m_a$ >3.24 稱為方塊波區，因為輸出電壓已成為方塊波，其基本波振幅($\frac{(\hat{v}_o)_1}{V_d}$)= $4/\pi$ 。如圖 4.6 所示在過調制區(1< m_a <3.24)下輸出電壓將出現低頻諧波，此對於一般正弦式輸出波形之變流器是需要禁止的，因為其會造成輸出電壓具有較高之失真率。然而過調制區及方塊波區之操作對於馬達驅動器則常被使用，因較高之基本波可以獲致較高之轉矩，對於馬達瞬間加重載及啟動時是相當有幫助的。而低頻諧波所造成之脈動轉矩可以藉由機械的低頻濾波特性予以克服，以降低振動及噪音。

圖 4.5

單相 SPWM 變流器輸出基本波電壓與 m_a 之關係

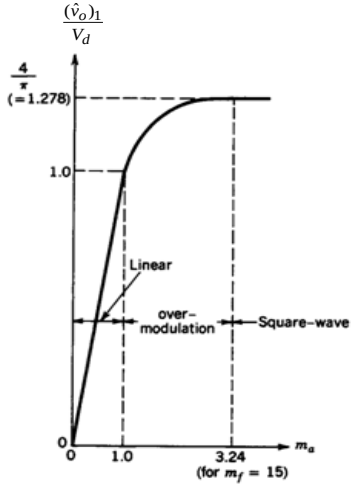
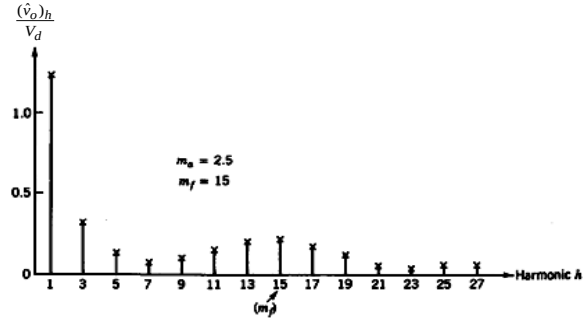


圖 4.6

單相 SPWM 變流器 m_a>1 之輸出電壓諧波頻譜



電路模擬

Inverter 電路參數如下：

V_d=70V, V_{tri}=5V_{pp}/18kHz, L=661.5mH*2, C=10μF, R=14Ω,

V_{conA}, V_{conB}=2.4V_p/60Hz

採用單電壓極性切換之模擬電路如圖 4.7 所示，模擬結果如圖 4.8 所示。

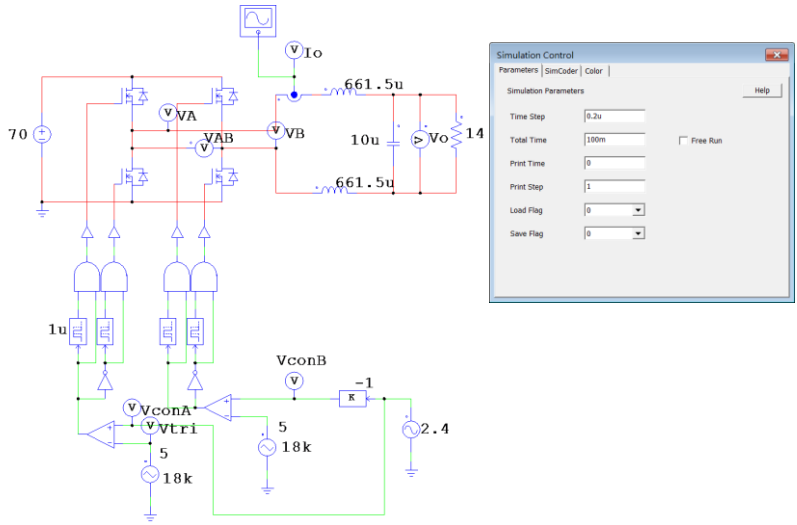


圖 4.7 採用單電壓極性切換之模擬電路

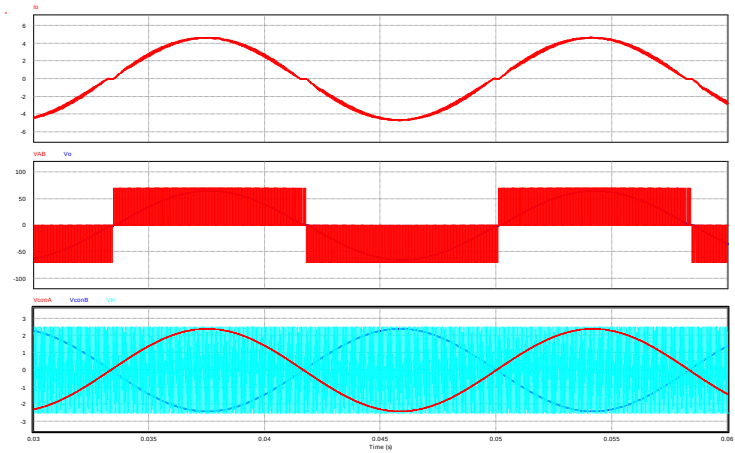


圖 4.8 圖 4.7 採用單電壓極性切換之模擬結果

SimCoder 程式規劃

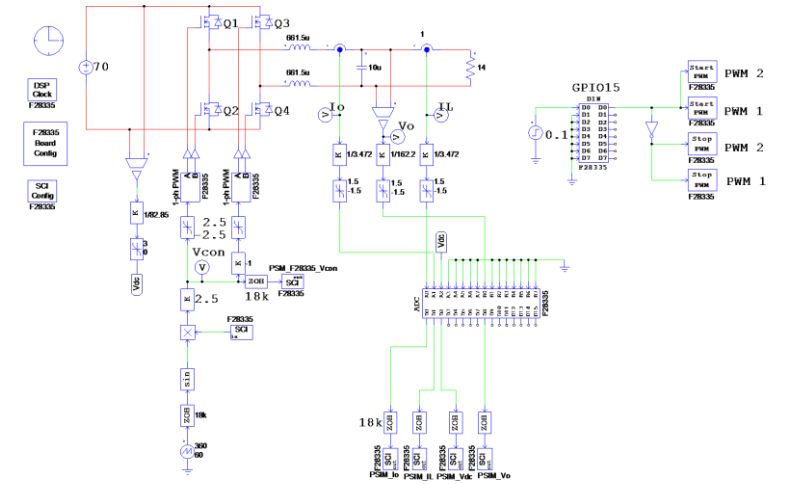
根據圖 4.7 之類比式單電壓極性切換模擬電路，改為以 TI F28335 實現控制電路之 SimCoder 電路規劃如圖 4.9 所示，其中感測電路之增益硬體上設定為：

電流感測增益 K_s=1/3.472

電壓感測增益 K_v=1/162.2

電壓及電流感測信號在進入 DSP A/D 接腳前均經過一限制器使輸入電壓被鉗位在±1.5V。以上硬體電路僅供 SimCoder 模擬用，由於實際實作電路上 DSP 只接受 0~3V 信號，此 AC 信號會被提升 1.5V 後再經過 3V 之鉗位後才進入 A/D 接腳，亦即 DSP 對於 AC 信號乃以 1.5V 當成 0。

SimCoder 各部份元件之設定如下



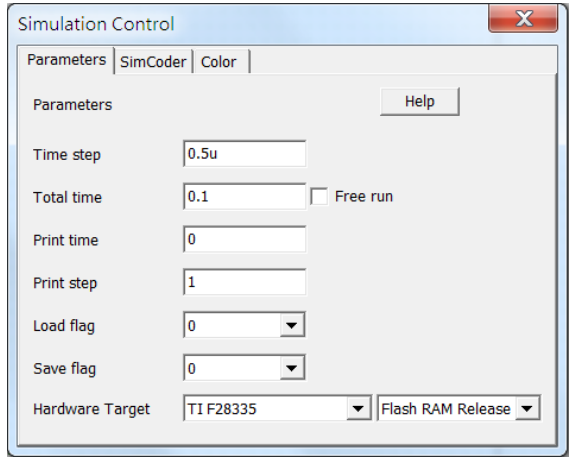
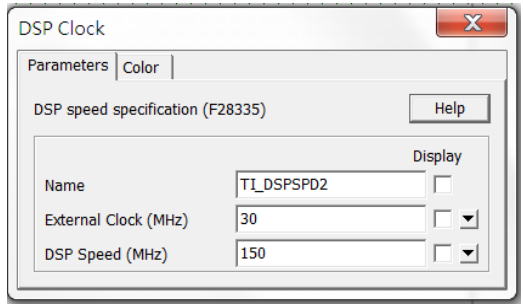


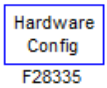
圖 4.9 採用單電壓極性切換之 SimCoder 模擬電路

DSP Clock 設定

DSP Clock 可採用外部設定亦或內建，採用外部設定需自行更改 External Clock 參數(default=30MHz)，採用內建則使用 DSP speed 150MHz。本實驗採用內建值。



28335 硬體設定(Hardware Config)



本實驗 DSP F28335 之硬體設定必須在如下表中以點選打勾方式設定如下

- 使用兩組 PWM，A 臂採用 PWM1(GPIO 0, GPIO 1)，B 臂採用 PWM2(GPIO 2, GPIO 3)。
- 使用一組 DI (Digital Input，GPIO 15)，可以利用外部電路來啟動 PWM。
- 使用一組串列通訊 SCI (SCI C, GPIO62, 63)用以在電路動作時，將信號透過 RS232-USB 方式傳至電腦，監控系統之工作。

GPIO 0	Digital Input	Digital Output	V PWM		
GPIO 1	Digital Input	Digital Output	V PWM	Capture	
GPIO 2	Digital Input	Digital Output	V PWM		
GPIO 3	Digital Input	Digital Output	V PWM	Capture	
GPIO 4	Digital Input	Digital Output	PWM		
GPIO 5	Digital Input	Digital Output	PWM	Capture	
GPIO 6	Digital Input	Digital Output	PWM		
GPIO 7	Digital Input	Digital Output	PWM	Capture	
GPIO 8	Digital Input	Digital Output	PWM		
GPIO 9	Digital Input	Digital Output	PWM	Capture	Serial Port
GPIO 10	Digital Input	Digital Output	PWM		

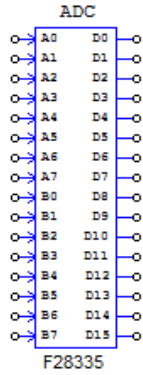
GPIO 11		Digital Input	Digital Output	PWM	Capture	Serial Port
GPIO 12		Digital Input	Digital Output	Trip-Zone		
GPIO 13		Digital Input	Digital Output	Trip-Zone		
GPIO 14		Digital Input	Digital Output	Trip-Zone	Serial Port	
GPIO 15	V	Digital Input	Digital Output	Trip-Zone	Serial Port	
GPIO 16		Digital Input	Digital Output	Trip-Zone	SPI	
GPIO 17		Digital Input	Digital Output	Trip-Zone	SPI	
GPIO 18		Digital Input	Digital Output	Serial Port	SPI	
GPIO 19		Digital Input	Digital Output	Serial Port	SPI	
GPIO 20		Digital Input	Digital Output	Encoder		
GPIO 21		Digital Input	Digital Output	Encoder		
GPIO 22		Digital Input	Digital Output	Encoder	Serial Port	
GPIO 23		Digital Input	Digital Output	Encoder	Serial Port	
GPIO 24		Digital Input	Digital Output	PWM	Capture	Encoder
GPIO 25		Digital Input	Digital Output	PWM	Capture	Encoder
GPIO 26		Digital Input	Digital Output	PWM	Capture	Encoder
GPIO 27		Digital Input	Digital Output	PWM	Capture	Encoder
GPIO 28		Digital Input	Digital Output	Serial Port		
GPIO 29		Digital Input	Digital Output	Serial Port		
GPIO 30		Digital Input	Digital Output			

GPIO 31	Digital Input	Digital Output		
GPIO 32	Digital Input	Digital Output		
GPIO 33	Digital Input	Digital Output		
GPIO 34	Digital Input	Digital Output	PWM	Capture
GPIO 35	Digital Input	Digital Output	Serial Port	
GPIO 36	Digital Input	Digital Output	Serial Port	
GPIO 37	Digital Input	Digital Output	PWM	Capture
GPIO 38	Digital Input	Digital Output		
GPIO 39	Digital Input	Digital Output		
GPIO 40	Digital Input	Digital Output		
GPIO 41	Digital Input	Digital Output		
GPIO 42	Digital Input	Digital Output		
GPIO 43	Digital Input	Digital Output		
GPIO 44	Digital Input	Digital Output		
GPIO 45	Digital Input	Digital Output		
GPIO 46	Digital Input	Digital Output		
GPIO 47	Digital Input	Digital Output		
GPIO 48	Digital Input	Digital Output	PWM	Capture
GPIO 49	Digital Input	Digital Output	PWM	Capture
GPIO 50	Digital Input	Digital Output	Encoder	

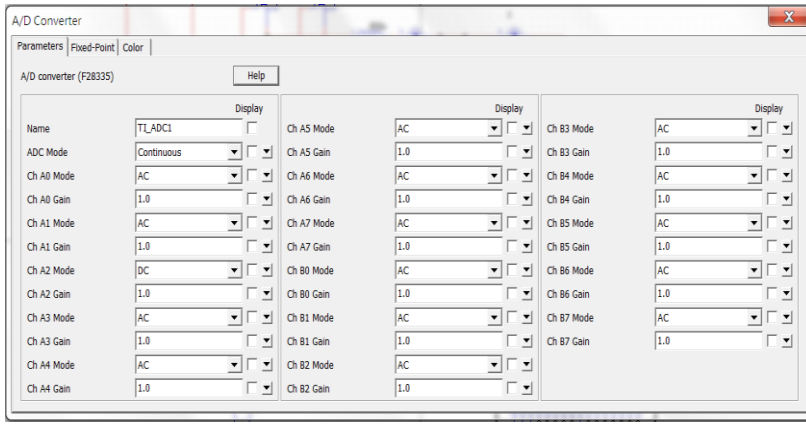
GPIO 51	Digital Input	Digital Output		Encoder	
GPIO 52	Digital Input	Digital Output		Encoder	
GPIO 53	Digital Input	Digital Output		Encoder	
GPIO 54	Digital Input	Digital Output		SPI	
GPIO 55	Digital Input	Digital Output		SPI	
GPIO 56	Digital Input	Digital Output		SPI	
GPIO 57	Digital Input	Digital Output		SPI	
GPIO 58	Digital Input	Digital Output			
GPIO 59	Digital Input	Digital Output			
GPIO 60	Digital Input	Digital Output			
GPIO 61	Digital Input	Digital Output			
GPIO 62	Digital Input	Digital Output	V	Serial Port	
GPIO 63	Digital Input	Digital Output	V	Serial Port	
GPIO 64	Digital Input	Digital Output			
GPIO 65	Digital Input	Digital Output			
GPIO 66	Digital Input	Digital Output			
GPIO 67	Digital Input	Digital Output			
GPIO 68	Digital Input	Digital Output			
GPIO 69	Digital Input	Digital Output			
GPIO 70	Digital Input	Digital Output			

GPIO 71	Digital Input	Digital Output		
GPIO 72	Digital Input	Digital Output		
GPIO 73	Digital Input	Digital Output		
GPIO 74	Digital Input	Digital Output		
GPIO 75	Digital Input	Digital Output		
GPIO 76	Digital Input	Digital Output		
GPIO 77	Digital Input	Digital Output		
GPIO78	Digital Input	Digital Output		
GPIO 79	Digital Input	Digital Output		
GPIO 80	Digital Input	Digital Output		
GPIO 81	Digital Input	Digital Output		
GPIO 82	Digital Input	Digital Output		
GPIO 83	Digital Input	Digital Output		
GPIO 84	Digital Input	Digital Output		
GPIO 85	Digital Input	Digital Output		
GPIO 86	Digital Input	Digital Output		
GPIO 87	Digital Input	Digital Output		

AD Converter(ADC)設定

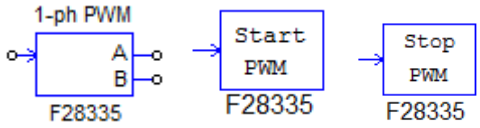


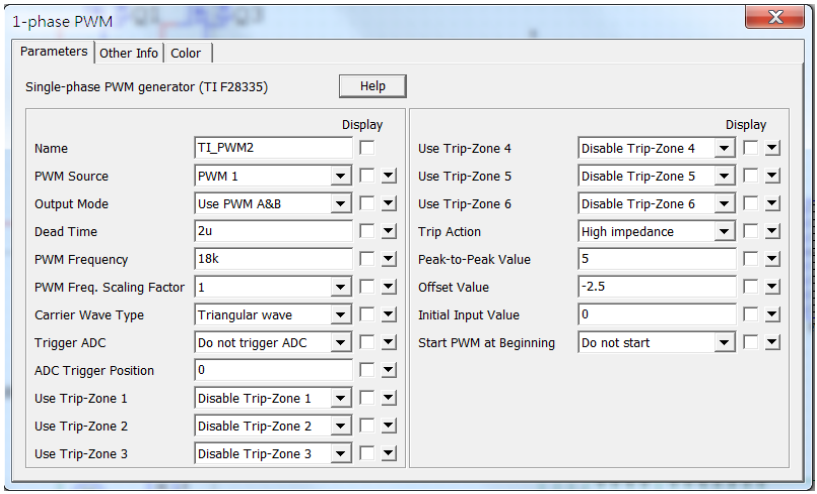
F28335 共有 16 組 ADC，分成 A0~A7 及 B0~B7 等通道，ADC 模組需設定 ADC 取樣模式(ADC Mode)，此處設定連續(Continuous)取樣模式，其次需設定各通道輸入之模式(Mode)以及增益(Gain)，本實驗總計回授 4 信號，分別為負載電流(I_L, A0))、電感電流(I_0, A1))、輸入電壓(V_d, A2)及輸出電壓(V_0, B0)，只有輸入電壓設定為 DC 模式，其餘為 AC 模式，各 ADC 之增益均設為 1。



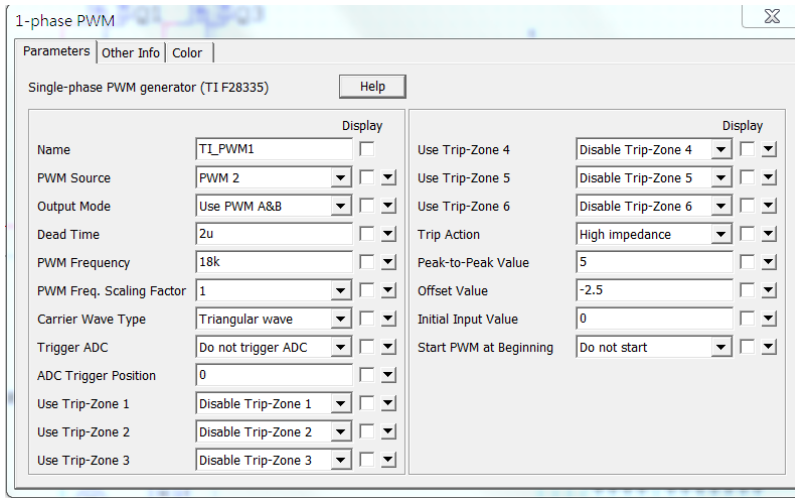
PWM 設定

採用單相 PWM，A 臂採用 PWM1，B 臂採用 PWM2，PWM 採用 18kHz 之三角波，dead time 2μs，PWM 三角波之振幅設定為 Vp=10V，offset=-5V，亦即介於-5V~+5V 之三角波。PWM 設為起始狀態不起動，其乃藉由 Start PWM 及 Stop PWM 等二模組來控制其啟動及截止，PWM1 及 PWM2 各有各自之 Start PWM 及 Stop PWM 模組。





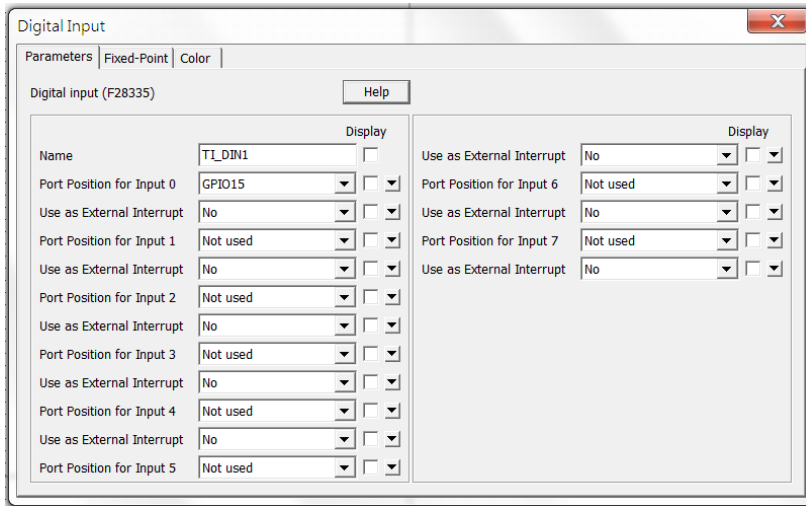
PWM 1



PWM 2

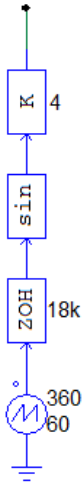
Digital Input(DI)設定

本實驗透過 DI(使用 GPIO 15)用以啟動及關閉 PWM1 及 PWM2。DI 輸入信號乃由 DSP 外部信號所產生。



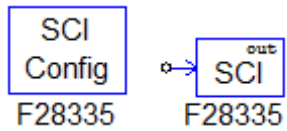
正弦波信號產生

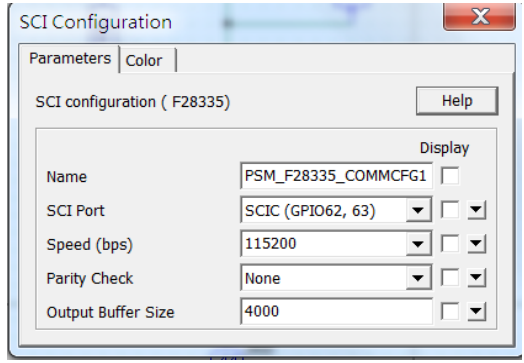
利用 PSIM 之鋸齒波(Sawtooth)產生器產生一 60Hz 振幅(即代表角度)360V 之信號，經由一 18kHz 之 ZOH(zero-order-hold)取樣及 Sin 函數後再乘上一增益 K 用以產生 PWM 之控制電壓。PWM 之振幅調制指數可以利用 K 值來設定。

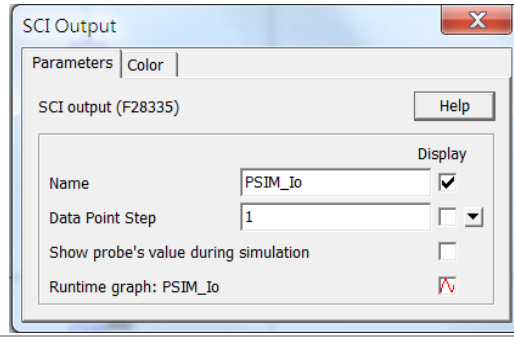


SCI 設定

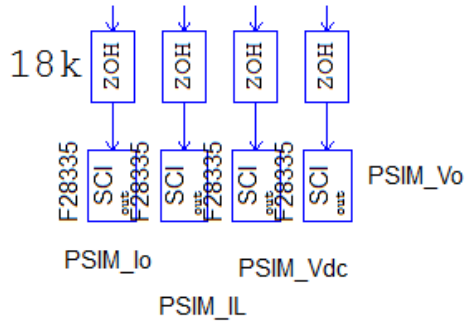
本實驗透過 SCI 將信號傳回電腦，SCI 設定(SCI Config)及 SCI out 二模組之設定如下。SCI 之通訊速度設為 115200(bps)，不使用 parity check，緩衝區大小(buffer size)設定為 4000 點。SCI output 則設定為每一點都傳輸。







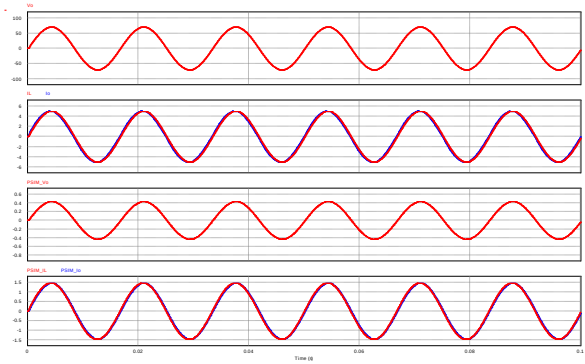
⚠注意	SCI out 模組必須配合 ZOH 模組使用，ZOH 模組用以定義 SCI 之中斷頻率。



SimCoder 電路模擬結果

圖 4.10

採用 SimCoder 電路之模擬結果



自動產生 Code

可以利用 PSIM 下之 Simulate => Generate Code 將上述變流器電路之 SimCoder 控制電路部分轉換成 C Code。PSIM 會在與所模擬電路同一目錄下產生一與模擬檔案名稱相同的子目錄，且將自動產生的 Code 及在 TI Code Composer 下一 project 所需相關的檔案放於此子目錄下。

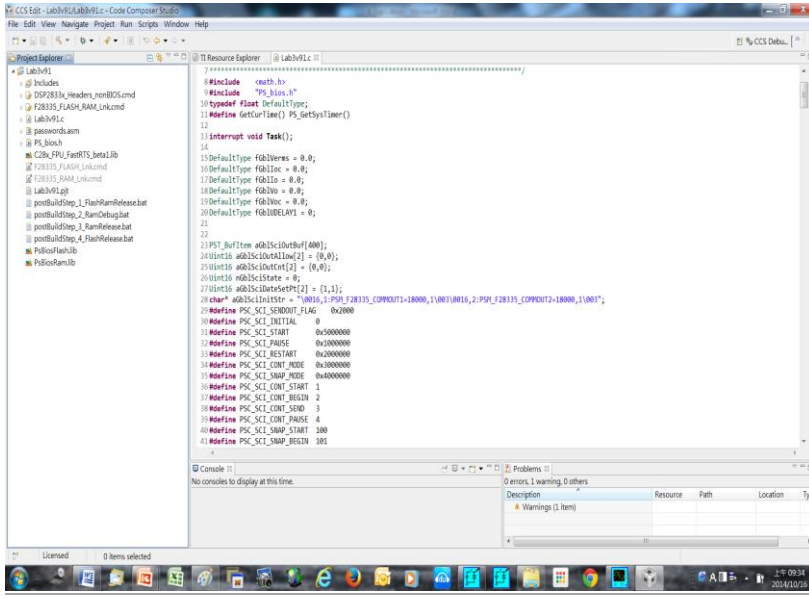
例如：Lab1.psimsch 會產生一子目錄 Lab1

利用 TI Code Composer 進行編譯及燒錄

進入 TI Code Composer Studio

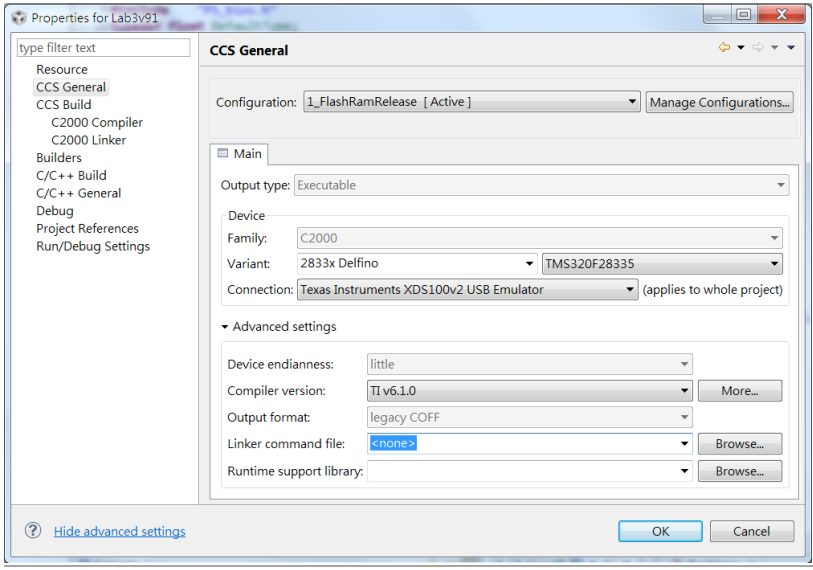
1. 叫出上述次目錄(Lab1)下之 project: Lab1.pjt
   Project => Import Legacy CCSv3.3 Project => 選擇子目錄下之 Lab1.pjt => Next => Finish

便會得到如下畫面，點選 Lab1.c 亦可檢視 SimCoder 所產生之 C 檔：



2. 點選 Lab1[Active, FlashRamRelease] => Build all 檢查有無錯誤， Warnings 可以忽略。

3. 點選 Lab1[Active, FlashRamRelease]=>滑鼠右鍵 =>作以下設定。



4. RUN => Debug =>將程式燒錄至 DSP IC=>可以拔除 JTAG=>進行實驗

實驗量測

實驗設備與教具配置如圖 4.11，直流電源供應器 PSW160-7.2 連接到 PEK-110 的輸入端子 J1，其輸出端子 J3 先經過交流功率表 GPM-8213 後接到被動式負載 GPL-100。圖 4.12, 4.13 為開關觸發信號波形，圖 4.14, 4.15 為輸出電壓、輸出電流與電感電流的量測波形。

圖 4.11

實驗設備配置圖

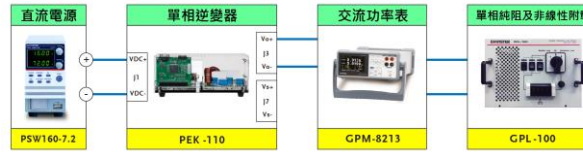


圖 4.12

單電壓極性切換開關觸發信號之量測波形(1/2)

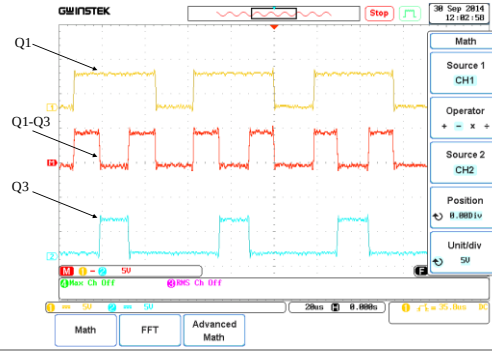


圖 4.13

單電壓極性切換開關觸發信號之量測波形(2/2)

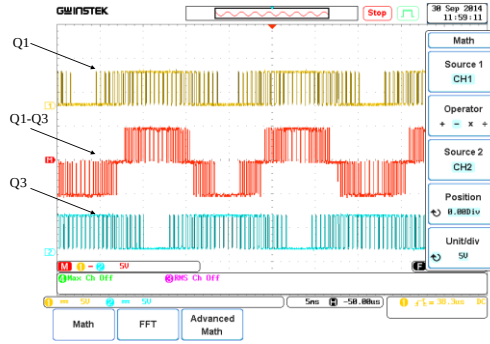


圖 4.14

變流器輸出電壓及電感電流之量測波形

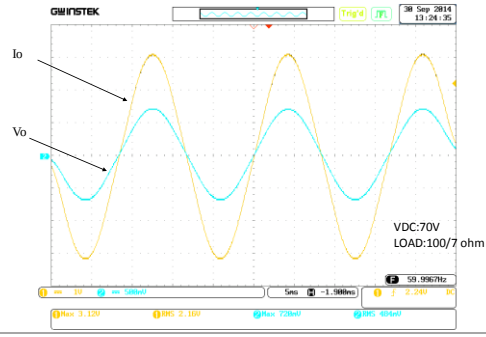
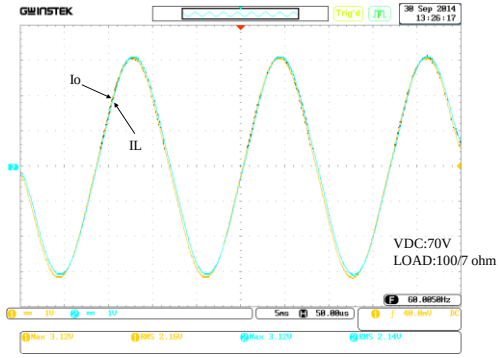


圖 4.15

變流器電感電流及負載電流之量測波形



當硬體電路動作後，可以在 PSIM 下打開 DSP Oscilloscope，將 RS232-USB 傳回之信號在電腦上顯示。

打開 DSP Oscilloscope 方法：

Utilities => DSP Oscilloscope =>如圖 4.16，設定 Serial port(可由電腦控制台下之裝置管理員獲得或設定)，Baud rate(需與 SCI Configuration 統一)，Parity check(需與 SCI Configuration 統一) => 設定完成後，再點選 Connect，若連結成功，則會出現如圖 4.17、圖 4.18 及圖 4.19 之綠燈及傳回之信號畫面。若要存取 DSP Oscilloscope 結果，操作程序如圖 4.20 所示，點選 Save，選擇存檔目錄位置，接下來便可由 SIMVIEW 來觀察及處理這些信號。

圖 4.16

DSP Oscilloscope 之設定

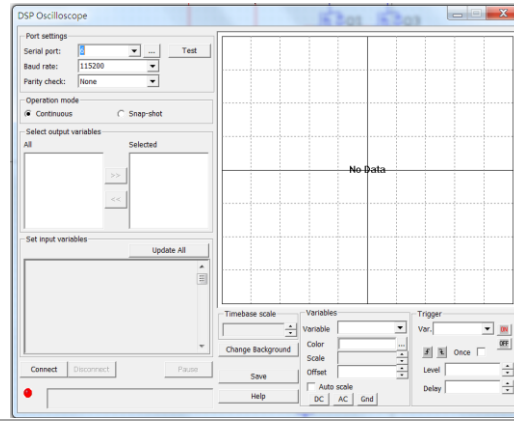


圖 4.17

DSP Oscilloscope 顯示之變流器輸出電壓及電感電流之波形

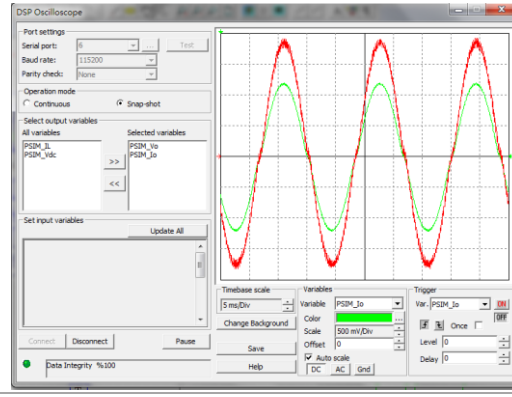


圖 4.18

DSP Oscilloscope 顯示之變流器電感電流及負載電流之波形

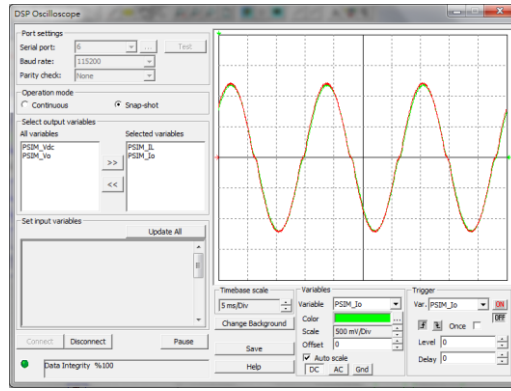


圖 4.19

DSP Oscilloscope 顯示之輸入電壓波形

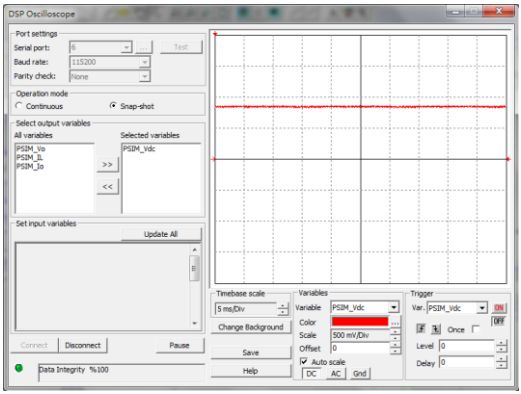
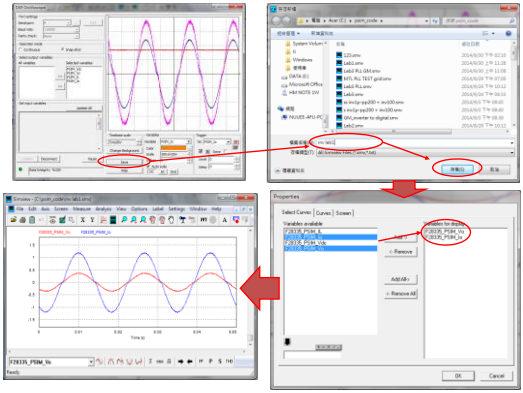


圖 4.20

DSP Oscilloscope 顯波形存檔後轉換至 SIMVIEW 之方法



實驗 2 雙迴路電感電流控制之獨立式變流器

實驗目的

- 學習單相全橋式 inverter 之模式化方法
- 電流迴路及電壓迴路控制器設計
- RMS 電壓迴路設計
- Inverter 之硬體規劃及 SimCoder 程式撰寫等

實驗原理

變流器電路工作原理與模型推導

全橋式轉換器之電路架構如圖 5.1 所示，其採用雙迴路控制，外迴路為電壓迴路，其誤差用以產生內迴路之電感電流命令，內迴路為電感電流迴路，其誤差用以產生 PWM 之控制電壓。PWM 乃採用正弦式 (sinusoidal) PWM 切換方式用以產生開關之觸發信號，L-C 則形成一二階的低通濾波器，用以衰減變流器之高頻切換輸出項使輸出電壓為低頻之正弦波。

由圖 5.1 可推得：

$$C\frac{dV_c}{dt} = I_{cap} = I_o - I_L \tag{5.1}$$

$$L\frac{dI_o}{dt} = V_{AN} - V_{BN} - V_c = S_A V_d - S_B V_d - V_c \tag{5.2}$$

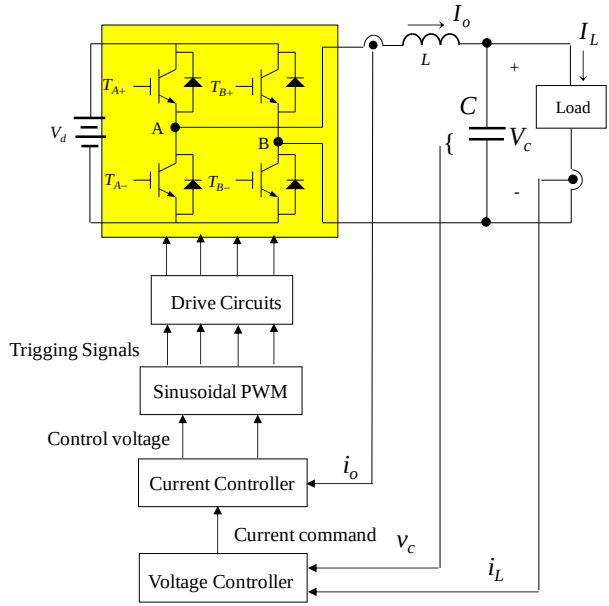
S_A 及 S_B 為分別為 A 臂及 B 臂開關之切換函數：

$$S_i = \frac{1}{2} + \frac{v_{coni}}{2v_{tm}} \quad (i = A, B) \tag{5.3}$$

v_{conA} 及 v_{conB} 分別為 A 臂及 B 臂之 PWM 控制電壓。若開關採用單極性電壓切換(uni-polar voltage switching)，則：

$$v_{conB} = -v_{conA} \tag{5.4}$$

圖 5.1
單相全橋式變流器之電路架構



由(5.3)可得：

$$S_A = \frac{1}{2} + \frac{v_{coni}}{2v_{tm}} \quad S_B = \frac{1}{2} + \frac{-v_{coni}}{2v_{tm}} \tag{5.5}$$

其中 v_{tm} 為 PWM 三角波之振幅。將(5.5)代入(5.2)可得：

$$L\frac{dI_o}{dt} = \frac{v_{con}}{v_{tm}}V_d - V_c \tag{5.6}$$

令

$$k_{pwm} = \frac{V_d}{v_{tm}} \tag{5.7}$$

可得：

$$L\frac{dI_o}{dt} = k_{pwm}v_{con} - V_c \tag{5.8}$$

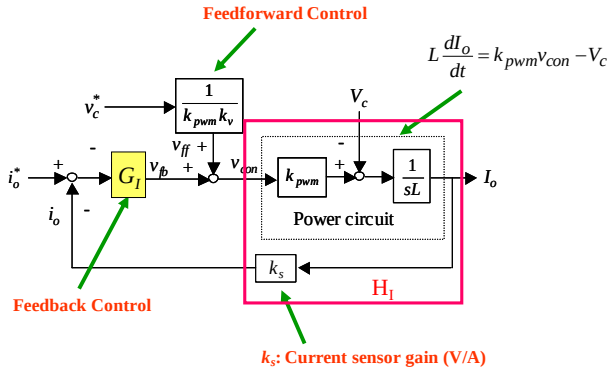
上述之(5.6)及(5.8)可分別用以設計圖 5.1 中之電壓及電流迴路。

控制器設計

A. 電流迴路控制器的設計

電流迴路設計如圖 5.2 所示，其中之電力電路方塊乃利用(5.8)所繪，k_v 及 k_s 分別為電壓及電流之感測增益。其採用迴授配合前向控制(feedforward control)方式，利用電壓命令 v_c^* 乘上增益 $1/k_{pwm}k_v$ 來直接抵消輸出電壓 V_c 對於電流迴路之擾動。

圖 5.2
電流控制迴路設計



電流控制器 G_I 可為 P、PI 及 Type 2 等控制器來設計，當使用 P 控制器時，$G_I=k_1$，由圖 5.2 可得：

$$\frac{i_o}{i_o^*} = \frac{\frac{k_1 k_s k_{pwm}}{L}}{s + \frac{k_1 k_s k_{pwm}}{L}} = \frac{u_i}{s + u_i} \tag{5.9}$$

此處極點 u_I 相當於電流迴路之頻寬，由之可得：

$$k_1 = \frac{u_i L}{k_s k_{pwm}} \tag{5.10}$$

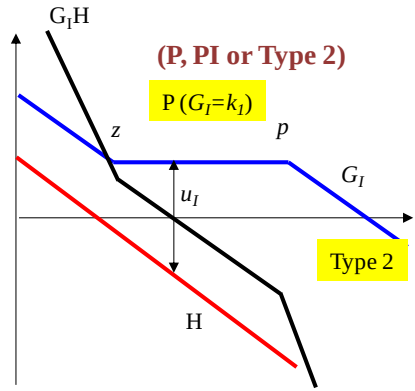
若使用 Type 2 則為：

$$G_I(s) = \frac{K_1(s+z)}{s(s+p)} \tag{5.11}$$

電流迴路波德圖設計如圖 5.3 所示，設計法則如下

1. 設定 u_I 為切換頻率的 1/10~1/8，若採用 P 控制可以直接求得 k_1。若採用 Type 2 則跳至步驟 2。
2. 設定 $z = \frac{u_I}{3}$ 。
3. 設定 $p = 3u_I$ 。
4. 利用 $G_I(u_I)H_I(u_I)$=1 求出 k_1

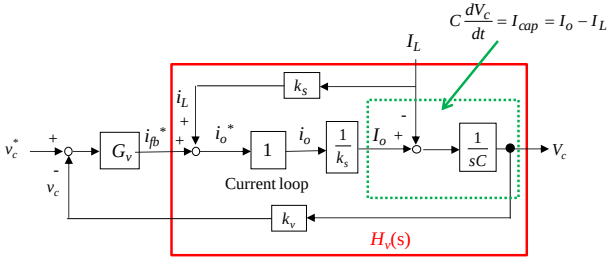
圖 5.3
電流控制迴路響應波德圖



B. 電壓迴路控制器的設計

電壓迴路控制方塊圖如圖 5.4 所示，其中電力電路方塊乃利用(5.1)所繪。此外假設電流迴路響應之頻寬(u_I)較電壓迴路頻寬高四倍以上，則電流迴路之響應在分析電壓迴路響應時可以被視為 1。電壓迴路控制器亦採用前向控制與回授控制並用，由於有感測負載電流，因此電壓控制器利用感測之負載電流(i_L)加入電流命令中作為前向控制用以直接消除負載電流對於電壓迴路之擾動。

圖 5.4
電壓控制迴路方塊圖



電壓控制器(G_v)可以利用比例(P)或比例積分(PI)控制器來設計，當使用 P 控制時，$G_v=k_2$，由圖 5.4 可得：

$$\frac{v_o}{v_o^*} = \frac{\frac{k_2 k_v}{C}}{s + \frac{k_2 k_v}{k_s C}} = \frac{u_v}{s + u_v} \tag{5.12}$$

若使用 PI 控制器：

$$G_v = \frac{K_2(s+z)}{s} \tag{5.13}$$

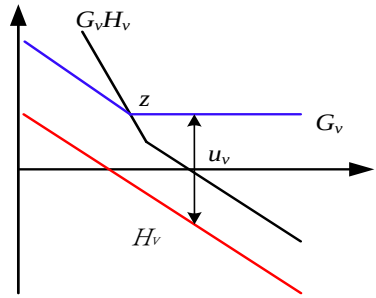
電壓迴路波德圖設計如圖 5.5 所示，P 或 PI 控制器設計其電壓迴路頻寬 u_v 可設計為電流迴路頻寬(u_I)的 1/4，即 $u_v=u_I/4$。當使用 P 控制器時：

$$k_2 = \frac{u_v C}{k_v} \tag{5.14}$$

當使用 P 控制器時，設計法則如下

1. 設定 u_v 為 $\frac{u_I}{4}$
2. 設定 $z = \frac{u_V}{3}$
3. 利用 $G_v(u_v)H_v(u_v)$=1 求出 k_2

圖 5.5
電壓控制迴路響應波德圖



C. 加入電壓均方根值迴路以增進電壓調整率

為使變流器的輸出具備良好之電壓調整率，可以在電壓迴路之外再加入一均方根值迴路如圖 5.6(a)所示，其利用 H_{rms} 計算輸出電壓之均方根值 v_{cm}，再與均方根值的命令 v_{cm}^* 比較及經過 G_m 調整後產生一振幅修正信號 A_{mr} 用以修正原來的振幅命令 A_{m0}，得到最終之振幅命令 A_m，A_m 再與單位正弦波 sinωt 相乘後得到瞬時之電壓命令 v_c^*。H_{rms} 計算均方根值必需利用均方根值(RMS)定義來計算如下，以得到精確之均方根值：

$$v_{cm} = \sqrt{\int v_c^2 d\omega t / 2\pi} \qquad (5.11)$$

(5.11)之計算結果一般需經一低通濾波器處理，低通濾波器可表示為：

$$H_{LPF} = \frac{a}{(s+a)} \tag{5.12}$$

其截止頻率 a 一般設定在 20Hz 以下，G_m 一般採用比例積分控制，以獲致精確之電壓調整率。然而由於 RMS 之計算相當耗費記憶體資源，因此一般 RMS 值計算可以利用圖 5.6(b)之簡化方塊圖來設計，其乃利用整流(ABS 方塊)及低通濾波(H_{LPF})計算正弦波之平均值，再乘以 $\frac{\pi}{2\sqrt{2}}$ (=1.11)得到 RMS 值，考慮濾波信號含有二次漣波，可以再加入一帶通濾波器(H_{BPF})計算此二次漣波，再由原信號扣除以得到平緩之 RMS 值。

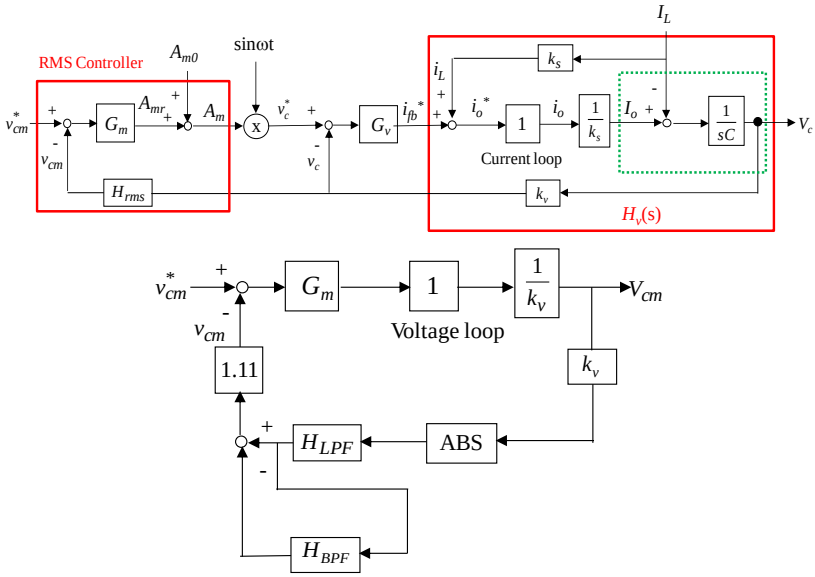


圖 5.6 (a)電壓均方根值控制迴路方塊圖；(b)簡化方塊圖

電路模擬

Inverter 規格

- P_o=115W, V_o=40Vac/60Hz,
- V_d = 70V, v_{tri}=18kHz/5Vpp, k_s = 1/3.472V/A, k_v = 1/162.2,
- 採用單電壓極性切換, 空白時間 2μs,
- L = 1.323mH, C= 10μF

控制器設計

- u_I = 18kHz/10=1.8kHz = 11310 rad/s => k_1 = 1.68
- u_v = 1.8kHz/4=450Hz = 2827 rad/s => k_2 = 4.6

根據上述參數所建制之模擬電路如圖 5.7 所示，在線性負載下之模擬結果如圖 5.8 所示。將上述控制電路之均方根值計算改為圖 5.6(b)之方式，模擬電路如圖 5.9

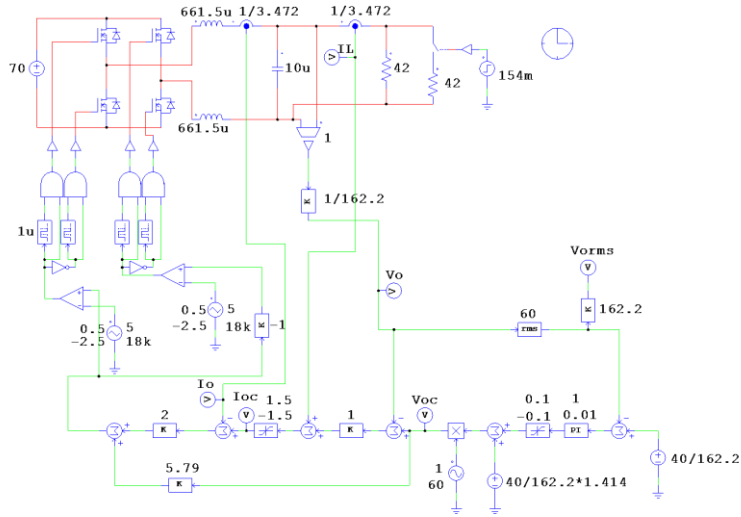


圖 5.7 雙迴路控制變流器之模擬電路

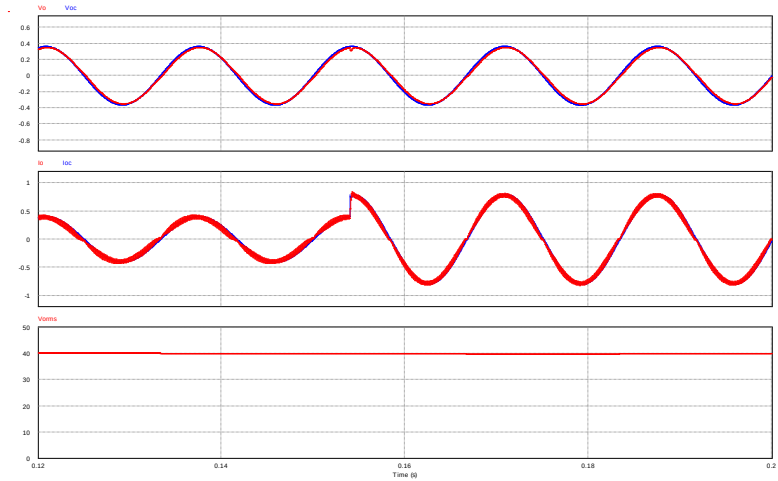


圖 5.8 雙迴路控制變流器之模擬結果

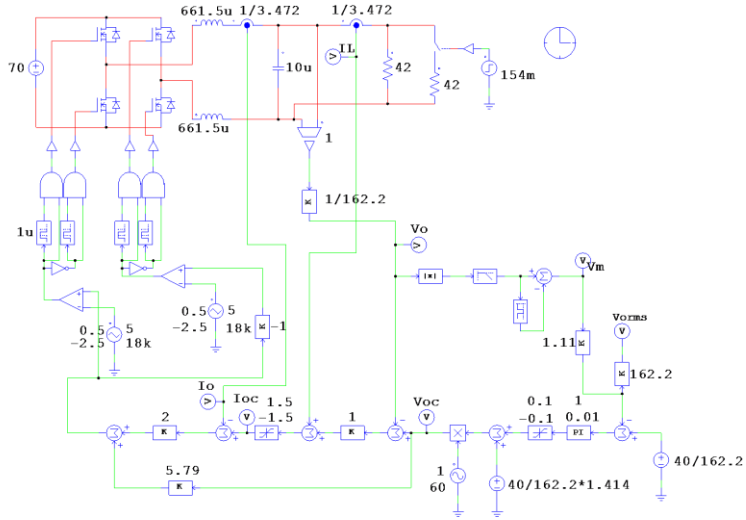


圖 5.9 將 RMS 計算改為圖 5.6(b)方式線性負載之模擬電路

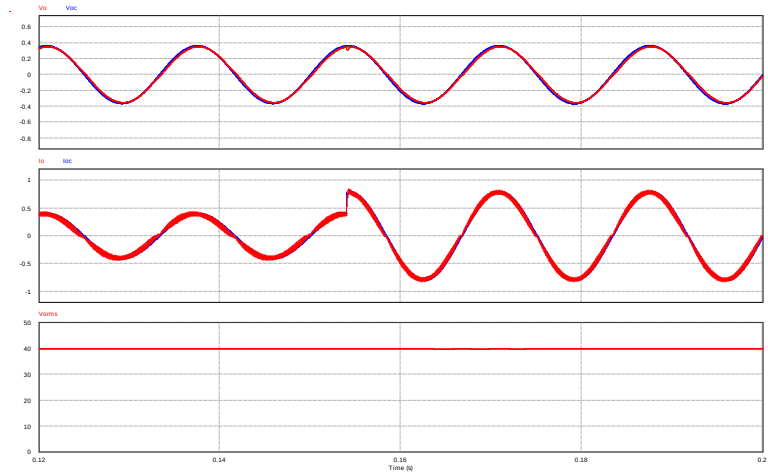


圖 5.10 將 RMS 計算改為圖 5.6(b)方式線性負載之模擬結果

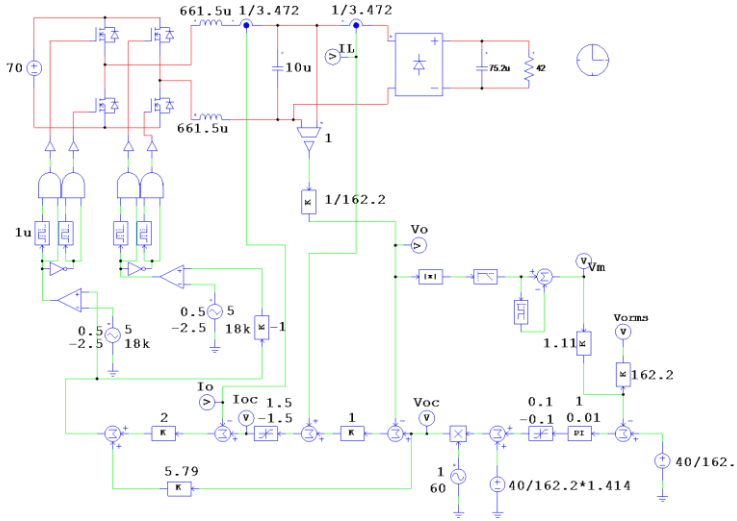


圖 5.11 將 RMS 計算改為圖 5.6(b)方式非線性負載之模擬電路

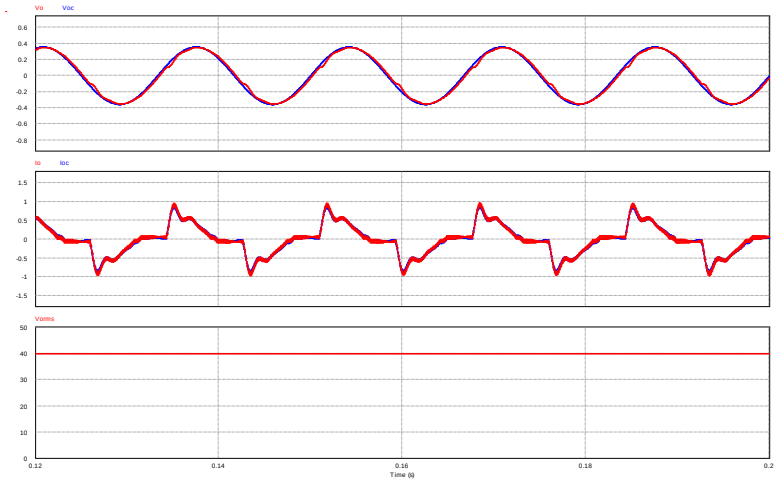


圖 5.12 將 RMS 計算改為圖 5.6(b)方式非線性負載之模擬結果

SimCoder 程式規劃及電路模擬

根據圖 5.9 之變流器模擬電路，改為以 TI F28335 實現控制電路之 SimCoder 電路規劃如圖 5.13 所示，其中 AD 取樣為連續方式，但各感測信號均經由一 18kHz 之 ZOH。計算 RMS 之低通及高通數位濾波器均藉由 PSIM Utilities 下之 s2z Converter 將圖 5.9 之類比濾波器經由數位化獲得，其取樣頻率 18kHz，數位化方式採用 Back Euler 方法。在線性 R 負載(0.95A=>1.9A)及非線性(RCD, C=75.2μF, R=42Ω)之模擬結果分別如圖 5.14 及圖 5.15 所示，其均與類比之模擬結果相當接近。

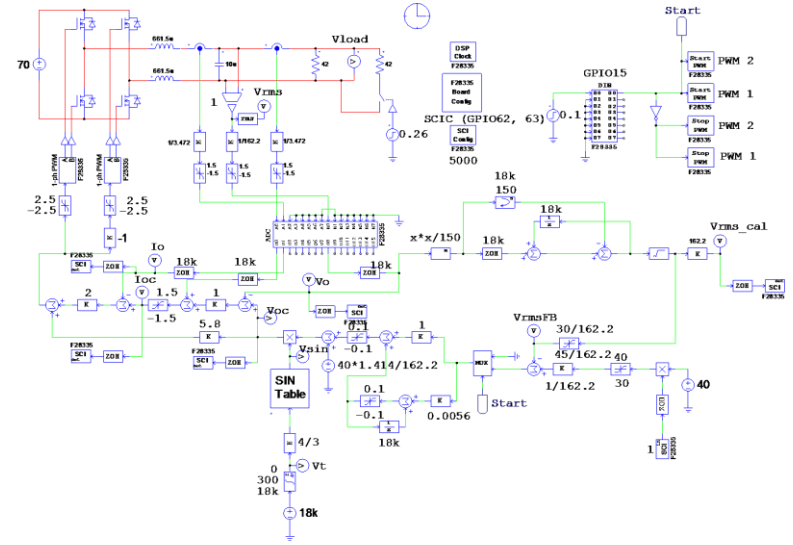


圖 5.13 以 SimCoder 方式建置之變流器模擬電路

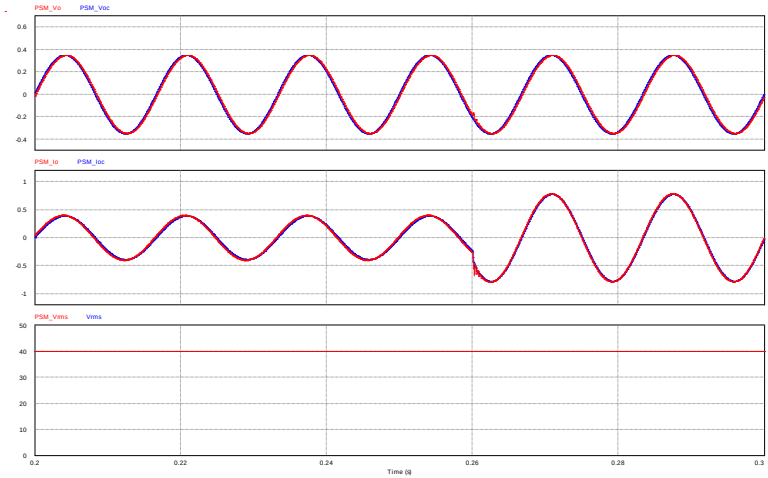


圖 5.14 線性負載下之模擬結果

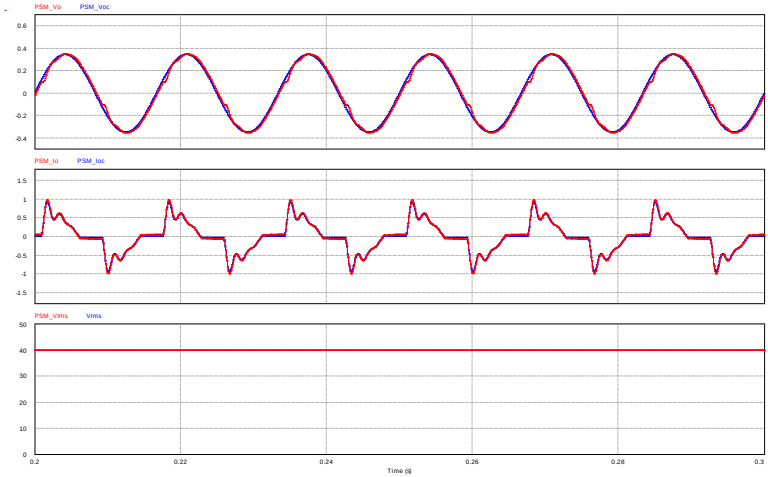


圖 5.15 非線性負載下之模擬結果

實驗量測

實驗設備與教具配置如圖 5.16，直流電源供應器 PSW160-7.2 連接到 PEK-110 的輸入端子 J1，其輸出端子 J3 先經過交流功率表 GPM-8213 後接到被動式負載 GPL-100。圖 5.17 為線性負載的量測波形，圖 5.18 為 RS232 傳回的量測結果，圖 5.19 為非線性附載的量測波形，圖 5.20 為 RS232 傳回的結果。

圖 5.16

實驗設備配置圖

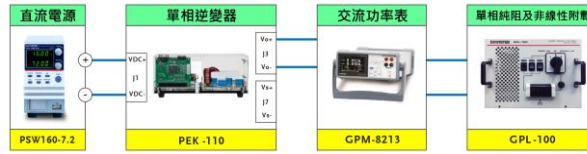


圖 5.17

線性負載下之量測結果

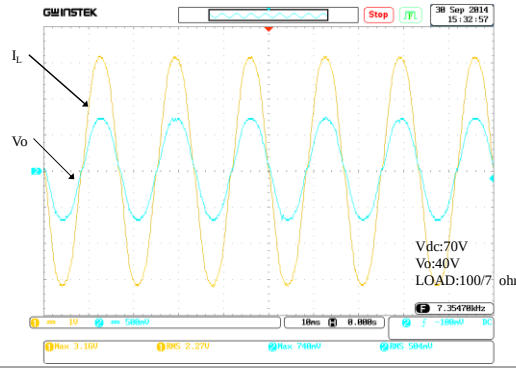


圖 5.18

線性負載下 RS232 傳回之量測結果(a)

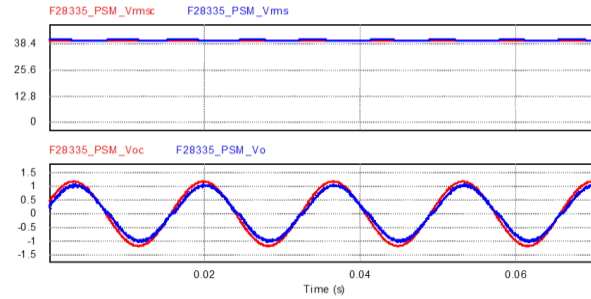


圖 5.18

線性負載下 RS232 傳回之量測結果(b)

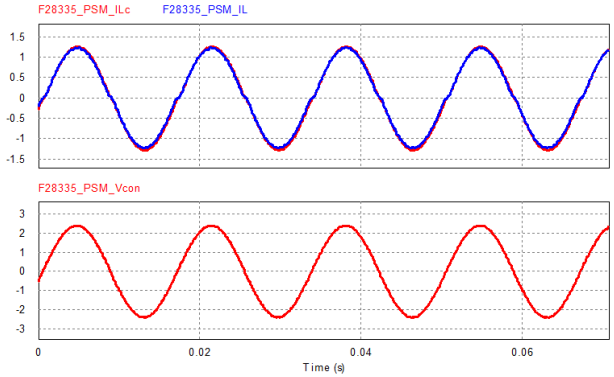


圖 5.19

非線性負載下之量測結果

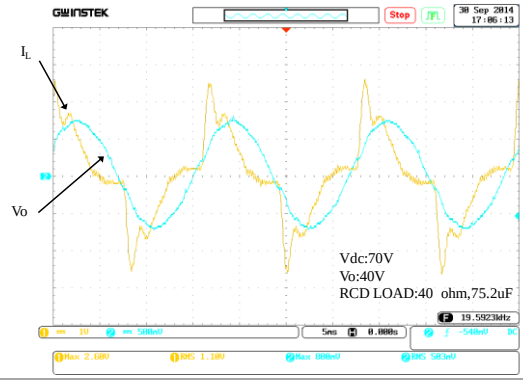


圖 5.20

非線性負載下 RS232 傳回之量測結果(a)

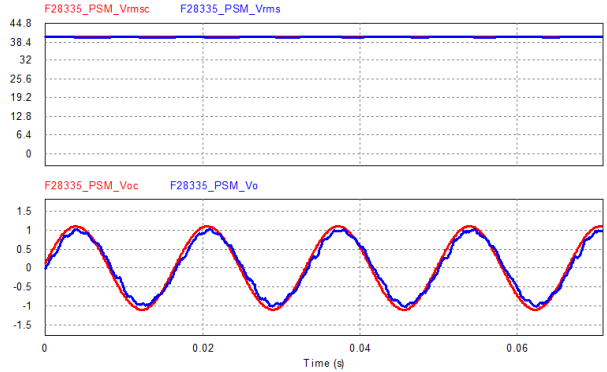
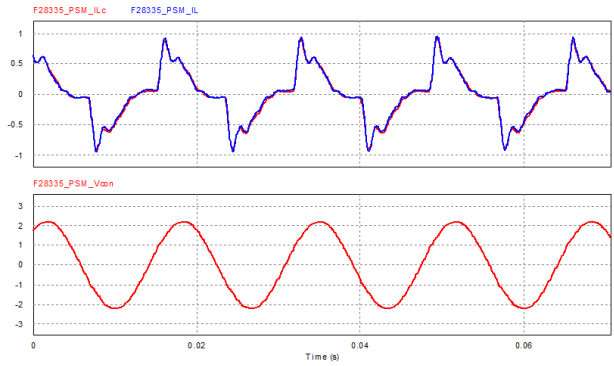


圖 5.20

非線性負載下 RS232 傳回之量測結果(b)



實驗 3 單相市電並聯變流器

實驗目的

學習單相市電並聯變流器之鎖相迴路方法、電流迴路及電壓迴路控制器設計、硬體規劃及並網之 SimCoder 程式撰寫等。

實驗原理

市電並聯變流器電路工作原理與控制器設計

單相市電並聯變流器之控制架構如圖 6.1 所示，控制採用雙迴路，外迴路為直流電壓控制器用以維持直流鏈電壓(V_d)，內迴路為電感電流控制用以控制市電輸入電流使輸入功率因數為一。

電流迴路控制方塊圖設計如圖 6.2 所示，其如同實驗 2 之變流器，利用迴授與前向(feedforward)控制並用，假設輸入電流可以與輸入電壓同相達到單位功因，前向控制信號 v_o^*可用以消除 V_s對電流迴路之擾動。如此電流迴路可以回授迴路求得為：

$$\frac{i_o}{i_o^*}=\frac{\frac{k_s k_1 k_{pwm}}{L}}{s+\frac{k_s k_1 k_{pwm}}{L}}=\frac{u_R}{s+u_R},\ u_R=\frac{k_s k_1 k_{pwm}}{L} \tag{6.1}$$

其中 u_R即相當於輸入電流迴路之頻寬，其可以由增益 k_1設定。如圖 6.3 所示，變流器之電流命令(i_o^*)乃由外迴路之直流電壓控制器 G_v所產生，其利用電壓調整之誤差乘上一與輸入電壓同步之單位正弦波 $sin(\omega t)$所產生。G_v之設計必須推導直流電壓迴路之模型，可根據圖 6.4(a)在單位功因下之等效電路來推導，由於直流鏈尚連接其他轉換

器將承接(或提供)由市電提供 (或吸收)之電力，因此直流側之模型僅為直流電容即可。交流側之輸入功率為：

$$P_{ac} = V_{s(p)} \sin \omega t \cdot I_m \sin \omega t = \frac{V_{s(p)} I_m}{2} - \frac{V_{s(p)} I_m}{2} \cos 2\omega t \qquad (6.2)$$
$$= \overline{P}_{ac} + \tilde{P}_{ac2}$$

其除了一直流項之外亦包含一二次之諧波項，此二次諧波項將造成直流電壓二次之漣波成份。直流側之平均功率等於交流側功率之直流項

$$\overline{P}_{ac} = P_{dc} \qquad (6.3)$$

將交流電流源反應至直流側可得圖 6.4(b)之等效電路，根據(6.3)可得：

$$\frac{V_{s(p)} I_m}{2} = V_d I_d \qquad (6.4)$$

$$I_d = \frac{V_{s(p)} I_m}{2V_d} = k_{dc} I_m \qquad (6.5)$$

由直流電流源 I_d 對 C_d 電容充電可得電壓迴路之模型為：

$$\frac{V_d}{I_m} = \frac{k_{dc}}{sC_d}, \quad k_{dc} = \frac{V_{s(p)}}{2V_d} \qquad (6.6)$$

電壓控制器 G_v 可根據圖 6.5 來設計，其中由於電流迴路頻寬較電壓迴路頻寬要寬出非常多，因此可以將(6.1)之電流迴路響應簡化為等於 1，因此 I_m^* 至實際 I_o 之電流振幅 I_m 之增益即為電流感測比例 k_s 之倒數，電壓迴路 H_{dc} 之波德圖可繪出如圖 6.6 所示。考慮直流電壓具有二次漣波，為使市電電流命令為低失真，電壓迴路之頻寬必須遠低於 120Hz 以衰減電壓之二次漣波，因此 G_v 採用 type II 補償器(即 PI + Low-Pass)之設計方式，其波德圖如圖 6.6 所示，閉迴路之增益(loop response)以及若僅採用 PI 控制器之響應亦繪於圖 6.6 中以茲比較。

圖 6.1

單相市電並聯變流器之控制架構

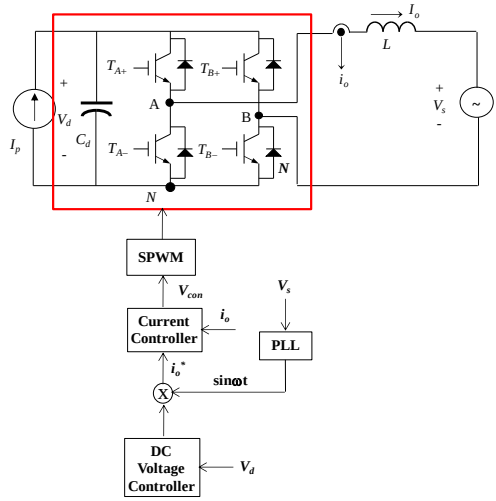


圖 6.2

電流控制迴路

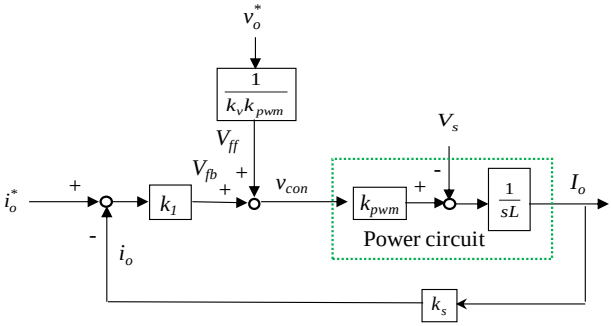


圖 6.3

電壓控制迴路

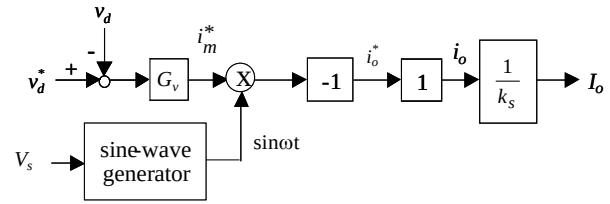


圖 6.4

變流器在單位功因下之等效電路

(a)等效電路

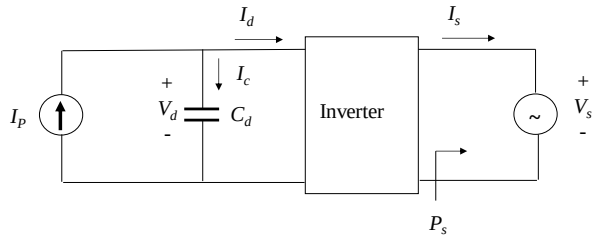


圖 6.4

變流器在單位功因下之等效電路

(b)直流側小信號等效電路

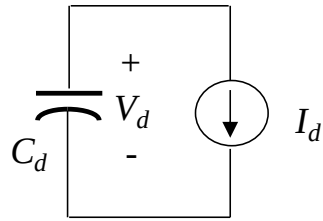


圖 6.5

電壓迴路控制器設計

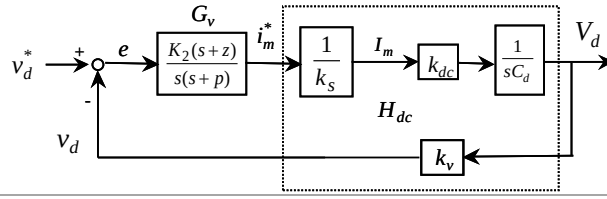
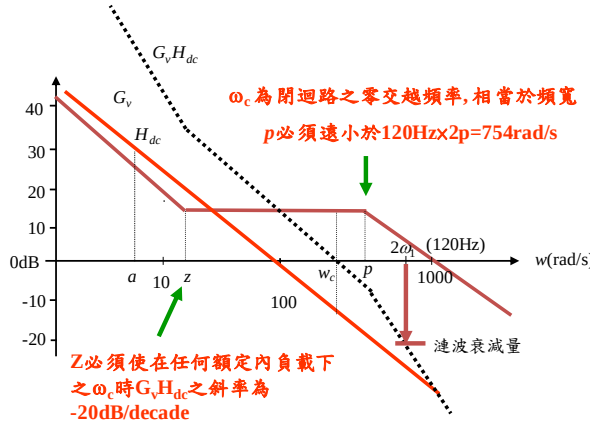


圖 6.6

電壓迴路控制波德圖



鎖相迴路設計

鎖相迴路架構如圖 6.7 所示，其利用市電電壓(V_s)取樣(*S/H*)後得到一信號 $V_m sin(wt)$，再經過一 1/4 市電週期之延遲得到一信號-$V_m cos(wt)$。二信號再與其後產生之同步信號 $cos(\omega_1 t)$與 $sin(\omega_1 t)$分別相乘後相加得到：

$$e = V_m\{\sin(\omega t)\cos(\omega_1 t) - \cos(\omega t)\sin(\omega_1 t)\} \quad (6.7)$$

信號 *e* 再經過一比例積分器(PI)後得到一頻率修正信號 $\Delta\omega$，再與原設定頻率 ω_0(=377)相加後得到一頻率 ω_1，ω_1 再經過積分後得到一角度信號θ，θ再經過一 0~2π之區間限制器後查閱 Sin()表(Sine table)與 Cos()表(Cosine table)得到 $cos(\omega_1 t)$與 $sin(\omega_1 t)$信號。藉由比例積分調整可以使誤差 *e* 為零，達到鎖相目地，亦即 $\omega = \omega_1$。

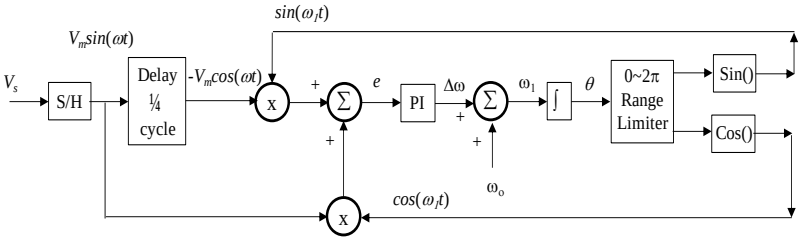


圖 6.7 鎖相迴路

電路模擬

整流器規格

- P_o=115W, V_o=40Vac/60Hz,
- V_d = 70V, v_{tri}=18kHz/5V_{pp}, k_s = 1/3.472V/A, k_v = 1/162.2,
- 採用單電壓極性切換, 空白時間 2μs,
- L = 1.323mH, C_d= 330μF, Co=10μF

電流迴路設計

$$k_{pwm} = 70/2.5 = 28$$
$$f_{ci} = 18k/10 = 1.8kHz$$
$$u_R = 1.8k \times 2\pi = 11310rad/s$$

- $k_1 = 1.68$

電壓迴路設計

- $H_{dc}(s) = \dfrac{26.21}{s}$
- 設定 $f_c = 20Hz, \omega_c = 125rad/s$
- 選擇 G_v之 p=180rad/s, z=30rad/s，則
  $G_v(s) = \dfrac{1024(s+30)}{s(s+180)}$

PSIM 模擬

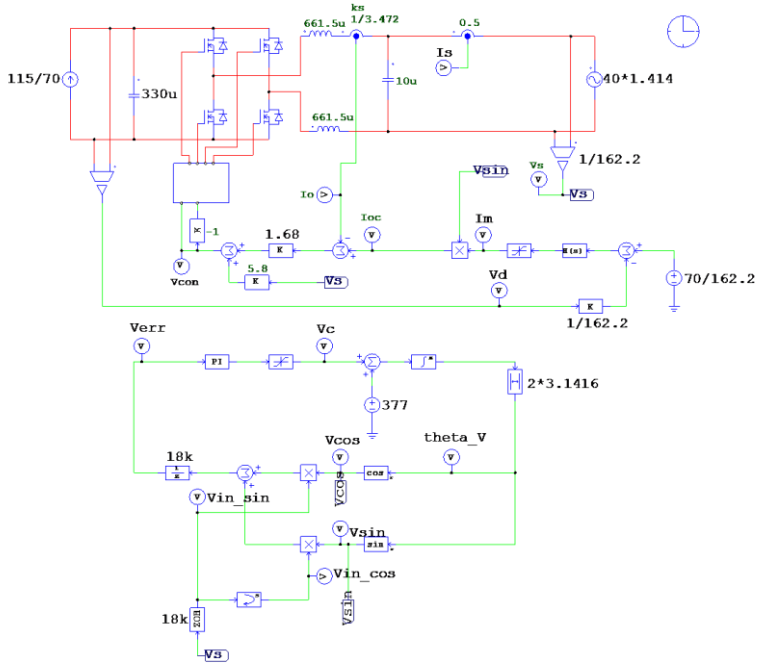


圖 6.8 市電並聯變流器 PSIM 模擬電路

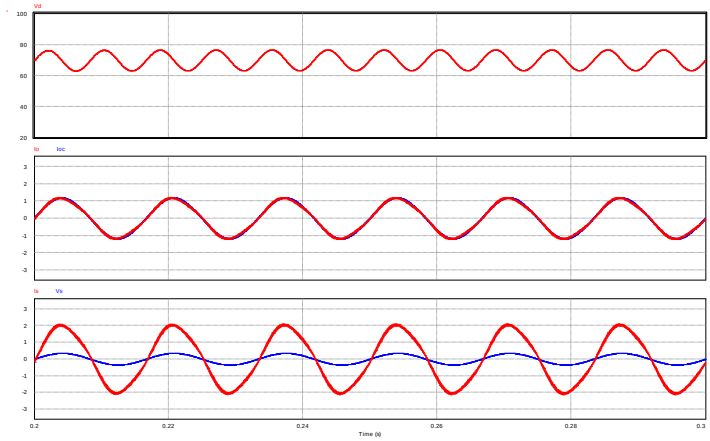


圖 6.9 市電並聯變流器 PSIM 模擬結果

SimCoder 程式規劃及電路模擬

以 SimCoder 建立之市電並聯變流器電路如圖 6.10 所示，除了上述電壓及電流控制器，鎖相電路外，尚包含：(1)一 Relay 控制器，其乃偵測輸入電壓(V_d)及市電電壓是否正常，才開啟 Relay 並上市電；(2)一 PWM 啟動電路，其除偵測 Relay 外之市電電壓外，亦檢查 Relay 並上後變流器之電壓，當二者均正常後，才允許 PWM 開始觸發，以及允許電壓積分器開始動作，模擬結果如圖 6.11 所示。

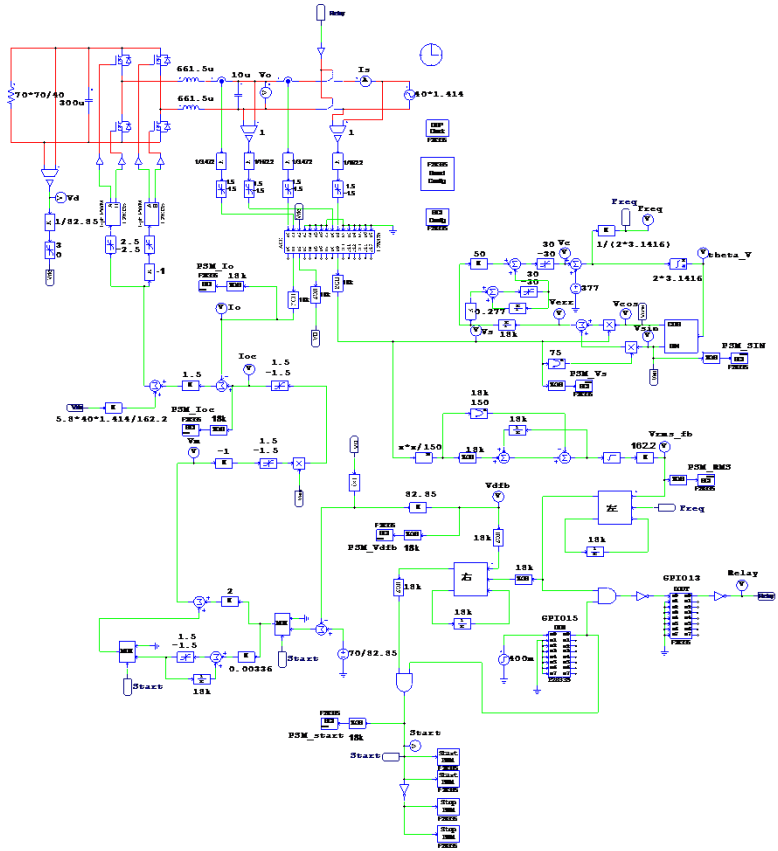


圖 6.10 以 SimCoder 建立之市電並聯變流器電路

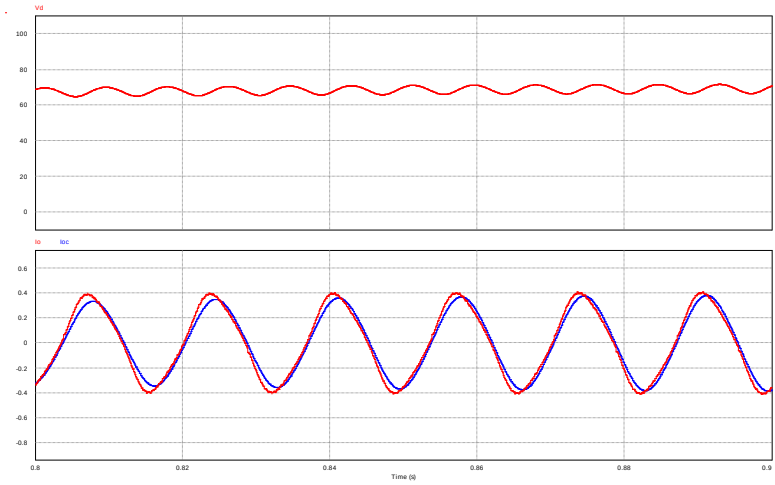


圖 6.11 以 SimCoder 建立之市電並聯變流器電路模擬結果

實驗量測

實驗設備與教具配置如圖 6.12，直流電源供應器 PSW160-7.2 (調至 75V, CC 準位調至 1A)與端子 J1 連接，端子 J3 經過交流功率表 GPM-8213 後與被動式交流負載 GPL-100 連接，J7 與交流電源供應器 APS-7050(調至 40Vac,限流 CC 設定至少 4A)連接，以此來模擬市電，被動式交流負載(GPL-100)連接於交流側之目地乃使變流器送出之功率均可消耗於交流負載上，請將交流負載之三段電阻均調為 ON(40Vac 下至少 150W)。開電前請確認變流器之啟動開關為 OFF 狀態，再開啟直流及交流電源，接著再起動變流器。AC 及 DC 電源圖 6.13 為在鎖相迴路時 DSP 示波器量測的輸出電壓與市電電壓波形，圖 6.14 為市電電壓與電感電流實際波形，6.15 則為 DSP 示波器上看到的波形。饋入市電之電流可以利用 DC 電源之 CC 準位來調整。

圖 6.12

實驗設備配置圖

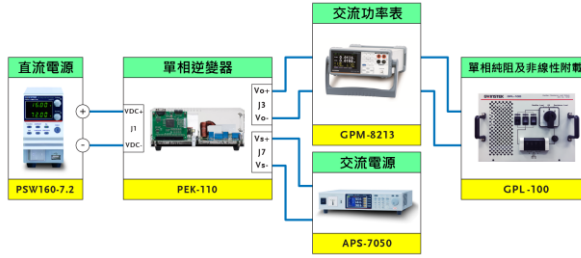


圖 6.13

鎖相迴路 DSP Oscilloscope 量測波形

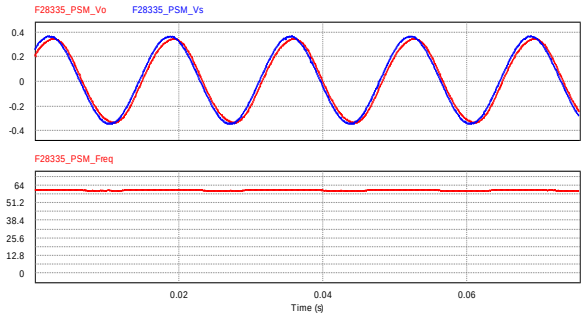


圖 6.14

市電電壓及電感電流量測波形

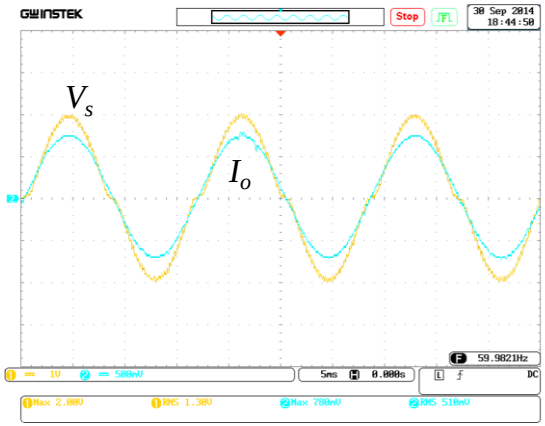
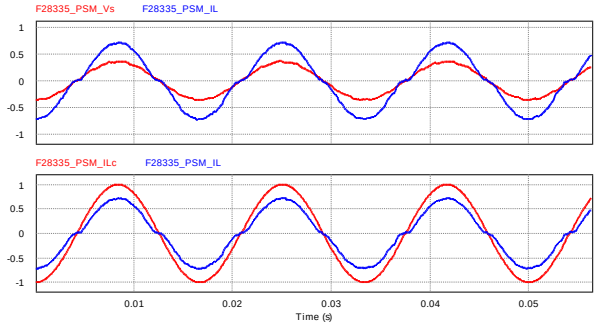


圖 6.15

DSP Oscilloscope 量測之電壓及電流波形



實驗 4 無橋式 PFC AC-DC 轉換器

實驗目的

學習 CCM PFC 原理、電流迴路及電壓迴路控制器設計、Totem ploe 無橋式 PFC 轉換器之設計、硬體規劃及 PFC 之 SimCoder 程式撰寫等。

實驗原理

單相 CCM 升壓式 PFC 轉換器之控制方法

CCM 升壓式 PFC 轉換器最常採用之控制架構為如圖 7.1 所示之雙迴路平均電流控制架構，其中 K_v 及 K_s 分別為電壓及電流之感測比例，其迴授輸出電壓(V_{df})，與參考電壓(V_d^*)比較後經由電壓誤差放大器(G_v)放大後得到 V_{ea} 信號，V_{ea} 信號再乘上感測之輸入電壓信號 K_vV_{in} 後得到電感電流命令 I_s^*，I_s^*再與迴授之電感電流信號(I_s)比較並經由電流誤差放大器(G_{CA})調整後得到 PWM 之控制電壓 V_{con}，再與 PWM 之鋸齒波(V_t)比較後得到開關之責任週期。如輸入電壓表示如下：

$$V_{in}=V_m sin\omega t \tag{7.1}$$

由圖 7.1 可得：

$$I_s^* = K_v V_{in} V_{ea} \tag{7.2}$$

若電流迴路能確實使電流追蹤其命令，則 $I_s=I_s^*$，輸入電感電流可表示為：

$$I_{in} = I_S / K_S = I_S^* / K_S = K_v V_{in} V_{ea} / K_S \tag{7.3}$$

由功率平衡可得：

$$\begin{aligned} P_{chg} &= V_d I_o = P_{in} = I_{in} V_{in} = K_v V_{in}^2 V_{ea} / K_s \\ &= K_v \frac{V_m^2}{2}(1-\cos 2\omega t) V_{ea} / K_s \\ &= P_o + P_{o2} \end{aligned} \tag{7.4}$$

忽略二次虛功率，由(7.4)可得：

$$I_o = \frac{K_v V_m^2}{2 K_s V_d} V_{ea} \tag{7.5}$$

(7.5)指出(I_o / V_{ea})之增益與輸入電壓的平方成正比，由於在全球通用電壓範圍下市電電壓有 3 倍左右的變化(90~264 V_{ac})，因此此增益變化達到 9 倍，不利電壓迴路誤差放大器之設計。

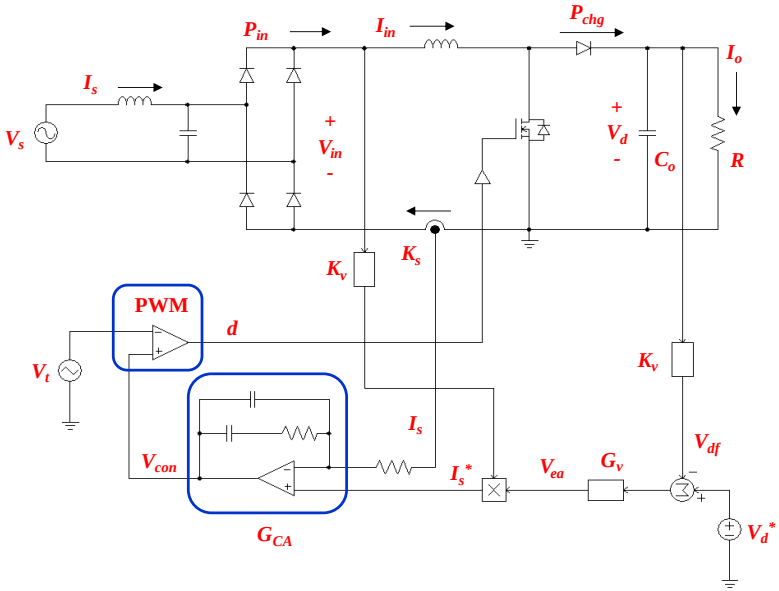


圖 7.1 採用雙迴路平均電流控制架構之 CCM 升壓式 PFC 轉換器

為了改善上述輸入電壓變化對電壓迴路增益之影響，圖 7.2 加入了電壓前饋控制(feedforward control)，其 V_{ea} 信號乘上 $K_v V_{in}$ 後再除以輸入電壓之平方後再得到電感電流命令 I_s^*：

$$I_s^* = \frac{K_v V_{in} V_{ea}}{K(K_v V_{in})^2} \tag{7.6}$$

藉由此安排，可得：

$$\begin{aligned} P_{chg} &= P_{in} = I_{in} V_{in} = V_d I_o \\ &= V_{in} I_s^* / K_s \\ &= K_v V_{in}^2 V_{ea} / (K_s K (K_v V_{in})^2) \\ &= V_{ea} / (K_s K_v K) \end{aligned} \tag{7.7}$$

因此

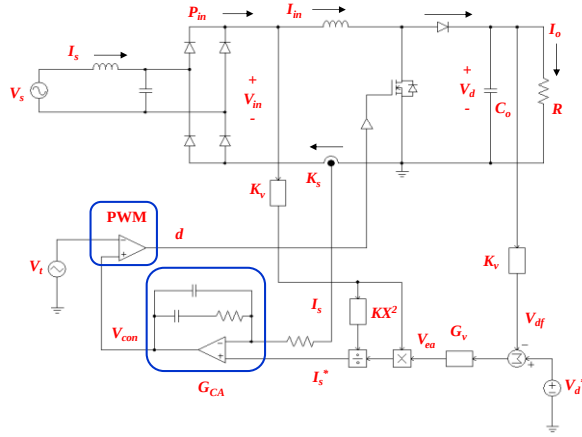
$$I_o = \frac{1}{K_s K_v K} V_{ea} \tag{7.8}$$

(7.8)指出(I_o / V_{ea})之增益不再與輸入電壓的平方成正比，而為一定值，即便市電電壓有 3 倍左右的變化，電壓迴路增益仍維持不變，因此易於設計電壓迴路之誤差放大器。

圖 7.2

具有前向控制之雙迴路平均電流控制之 CCM 升壓式

PFC 轉換器



單相 CCM 升壓式 PFC 轉換器之控制器設計

A. 電流迴路設計

對於外迴路控制器產生內迴路控制器命令之雙迴路控制架構，一般內迴路頻寬要較外迴路頻寬寬廣許多(通常設定 4 倍以上)，因此控制迴路之設計應先設計內迴路，在設計外迴路控制器時將內迴路控制之響應視為理想即可。以下先進行電流內迴路之設計，在設計電流誤差放大器之前，需先求出從電流誤差放大器輸出到迴授電感電流之小信號模型。由圖 7.1 之電感電流控制迴路並利用狀態平均法可得：

$$L\frac{dI_{in}}{dt} = V_{in} - (1-d)V_d \tag{7.9}$$

$$\frac{\tilde{I}_{in}}{\tilde{d}} = \frac{V_d}{sL} \tag{7.10}$$

考慮 PWM 及電流感測增益由(6.22)可得電流迴路之小信號模型：

$$H_i(s) = \frac{\tilde{I}_s}{\tilde{V}_{con}} = \frac{\tilde{I}_{in}K_s}{\tilde{d}V_s} = \frac{K_sV_d}{sLV_s} \tag{7.11}$$

電流迴路頻寬之限制由 PWM 控制電壓(V_{con})之變化率不得大於鋸齒波電壓變化率決定，亦即 V_{con} 之上升斜率必需小於鋸齒波之上升斜率($=V_sf_s$)，由於 V_{con} 乃感測之電感電流下降斜率經由 G_{CA} 反向放大獲得，因此

$$G_{CA}(\omega_{ci})K_s(V_d - V_{in})/L \leqq V_sf_s \tag{7.12}$$

其中 ω_{ci} 為電流迴路之零交越點(頻寬)。由於輸入電壓(V_{in})隨著正弦波變化，電感電流下降斜率出現在輸入電壓為零時，因此由(6.24)可得電流迴路頻寬之極限為：

$$G_{CA}(\omega_{ci})K_sV_d/L = V_sf_s \tag{7.13}$$

因此：

$$G_{CA,\max} = \frac{\tilde{V}_{con}}{\tilde{I}_s} = \frac{V_s f_s L}{V_d K_s} \tag{7.14}$$

由(7.11)及(7.14)並令$\omega_{ci}=2\pi f_{ci}$可得：

$$G_{CA,\max}(\omega_{ci})H_i(\omega_{ci}) = \frac{f_s}{2\pi f_{ci}} = 1 \tag{7.15}$$

由(7.15)可得電流迴路之最大頻寬為：

$$f_{ci} = \frac{f_s}{2\pi} \tag{7.16}$$

若 G_{CA}採用如圖 7.3(a)之二類誤差放大器：

$$G_{CA}(s) = \frac{1+sR2C1}{sR1(C1+C2)(1+sR2C1C2/C1+C2)} = \frac{K(s+Z)}{s(s+P)} \tag{7.17}$$

$$P = \frac{1}{R_2C_2},\ Z = \frac{1}{R_2C_1} \tag{7.18}$$

其設計如圖 7.3(b)所示，先將 H_i之波德圖繪出，設定 $f_{ci} < f_s/2$，求得 $|G_{CA}(f_{ci})|=1/H_i(f_{ci})$。指定 R_1，可得 $R_2=R_1|G_{CA}(f_{ci})|$。設定 P 及 Z 使 $P=3\omega_{ci}$及 $Z=\omega_{ci}/3$，$\omega_{ci}=2\pi f_{ci}$，再由(7.18)可求得 C_1及 C_2。

圖 7.3 G_{CA}之設計 (a)誤差放大器電路

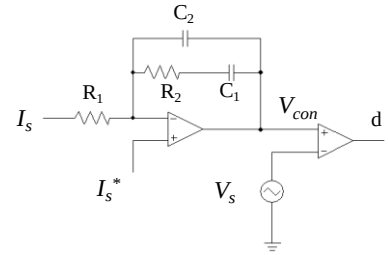
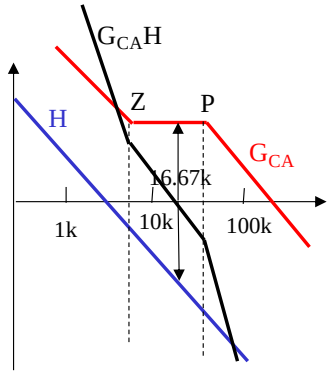


圖 7.3 G_{CA}之設計 (b)波德圖



B. 電壓迴路設計

如圖 7.4 所示，在單位功因下，單相 CCM 升壓式 PFC 轉換器之輸入功率因數可以表式為：

$$\begin{aligned} P_{ac} &= V_m \sin\omega t \cdot I_m \sin\omega t \\ &= \frac{V_m I_m}{2} - \frac{V_m I_m}{2}\cos 2\omega t \\ &= \overline{P}_{ac} + \tilde{P}_{ac2} \end{aligned} \tag{7.19}$$

其包含一實功項$\overline{P}_{ac}$與一二次之虛功項$\tilde{P}_{ac2}$。如同前述，直流輸出電壓之調整僅由實功項$\overline{P}_{ac}$負責，而二次之虛功項$\tilde{P}_{ac2}$對電壓之調整無作用，但會造成輸出電壓之二次漣波。因此此處電壓迴路之調整僅考慮$\overline{P}_{ac}$，基於功率平衡可得：

$$\overline{P}_{ac} = P_{dc} \tag{7.20}$$

亦即

$$\frac{V_m I_m}{2} = V_d I_d \tag{7.21}$$

若採用圖 7.1 之控制架構，利用(7.21)及由(7.5)可得小信號模型：

$$\tilde{I}_d = \frac{V_m \tilde{I}_m}{2V_d} = \frac{V_m \tilde{V}_{ea}(K_v V_m)}{2V_d K_s} = k_{dc}\tilde{V}_{ea} \tag{7.22}$$

其中

$$k_{dc} = \frac{K_v V_m^2}{2V_d K_s} \tag{7.23}$$

針對兩級式電路其小信號模型可以表示為圖 7.4(b)，由(7.23)及圖 7.4(b)可得：

$$\frac{\tilde{V}_d}{\tilde{V}_{ea}} = \frac{k_{dc}}{sC} \tag{7.24}$$

圖 7.4

單相 CCM 升壓式 PFC 轉換器之電壓迴路模型推導：

(a)在單位功因下之等效電路

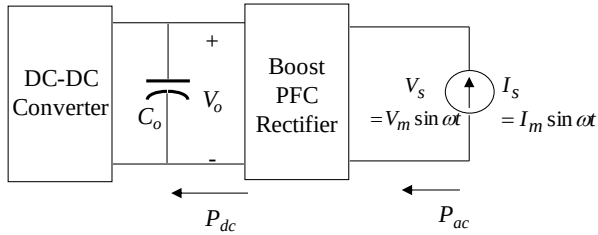
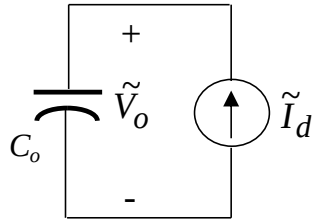


圖 7.4

單相 CCM 升壓式 PFC 轉換器之電壓迴路模型推導：

(b)小信號等效電路



利用(7.24)可繪出電壓迴路之控制方塊圖如圖 7.5(a)所示，其中：

$$H_v(s) = \frac{\tilde{v}_d}{\tilde{V}_{ea}} = \frac{k_{dc}k_v}{sC_o} \tag{7.25}$$

電壓迴路波德圖如圖 7.5(b)所示，其零交越頻率考慮前述輸出電壓由於二次虛功所造成之二次漣波，因此電壓誤差放大器 G_v 在此採用二

類之補償電路，其與 H_v 組成之迴路增益(loop gain) G_vH_v 的零交越頻率(f_c)必需遠低於二次漣波頻率($2f_o$=120Hz)，以衰減 V_{ea} 上之二次漣波，才能使輸出電流為低失真。另一方面此零交越頻率(f_c)相當於電壓迴路之頻寬，因此若太窄將使電壓之動態響應太差。因此最佳之零交越頻率一般定在 20Hz，使之兼顧電壓響應速度與輸入電流失真。

對於一般通用(universal)輸入電壓 90~264V_{ac}，在輸入電壓有將近 3 倍變化下，由於電壓迴路 H_v 之增益(k_{dc})受輸入電壓之平方變化影響如(6.33)所示，H_v 增益將有 9 倍變化(≈20dB)，因此圖 7.5 所示之迴路增益(loop gain) G_vH_v 在輸入電壓最高和最低時所產生之零交越頻率 $f_c(H)$和 $f_c(L)$將有 10 倍左右的差距。例如當電壓迴路控制器以較高輸入電壓參數來設計使 f_c(H)=20Hz，則在低輸入電壓時將造成電壓迴路的頻寬太窄 $f_c(L)$=2Hz，使電壓響應太慢。反之，若以低輸入電壓之參數來設計使 $fc(L)$=20Hz，則將造成高輸入電壓時頻寬太寬 $f_c(H)$=200Hz，無法衰減輸出電壓之二次漣波成分，造成電流失真過大等問題。

圖 7.5

根據圖 6.3 控制架構之電壓迴路設計：

(a)控制方塊圖

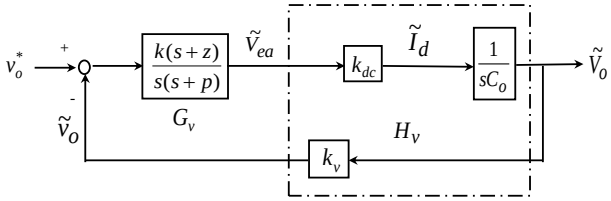
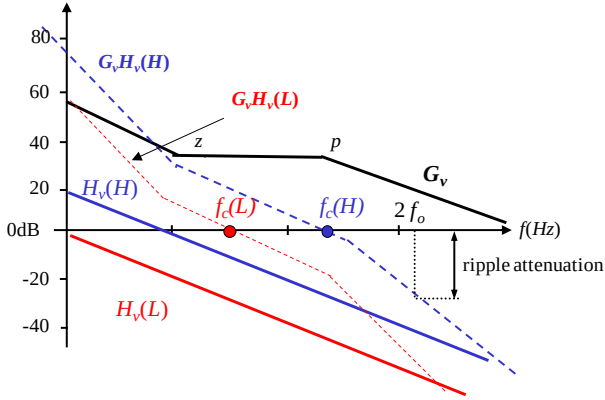


圖 7.5

根據圖 6.3 控制架構之電壓迴路設計：

(b)波德圖



C. 加入前向控制

為克服輸入電壓 3 倍變化所造成迴路增益 9 倍之變化問題，可以採用如圖 7.2 所示具有前向控制之雙迴路平均電流控制架構，利用(7.21)及由(7.8)可得小信號模型：

$$I_d = \frac{V_m I_m}{2V_d} = \frac{V_m V_{ea}(K_v V_m)}{2V_d K_s K(K_v V_m)^2} = k_{dcf} V_{ea} \tag{7.26}$$

其中

$$k_{dcf} = \frac{1}{2V_d K_s K_v K} \tag{7.27}$$

利用(7.27)重新繪出電壓迴路之控制方塊圖如圖 7.6(a)所示，其中：

$$H_v(s) = \frac{\tilde{v}_d}{\tilde{V}_{ea}} = \frac{k_{dcf} k_v}{sC_o} \tag{7.28}$$

電壓迴路波德圖如圖 7.6(b)所示，由於對於各個輸入電壓而言，H_v之波德圖為固定，因此 G_v誤差放大器之設計僅需針對單一 H_v來設計即可，並將其零交越頻率 f_c置於 20Hz(=125rad/s)以衰減 120Hz($2f_o$)之電壓漣波。二類誤差放大器 G_v之零點及極點一般置於 z=20~30rad/s, p=180~250rad/s。

圖 7.6

根據圖 7.2 控制架構之電壓迴路設計：

(a)控制方塊圖

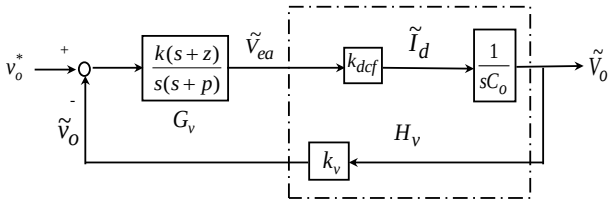
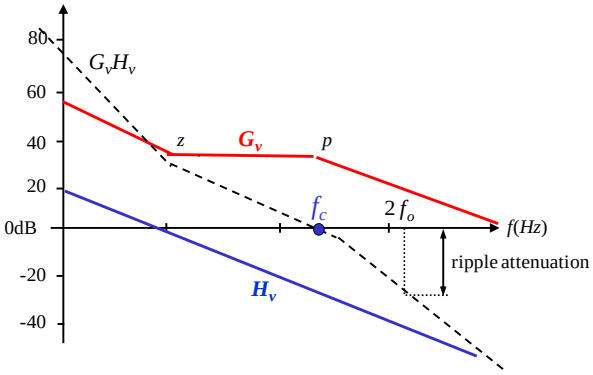


圖 7.6

根據圖 7.2 控制架構之電壓迴路設計：

(b)波德圖



單相 CCM 無橋式升壓式轉換器工作原理

傳統升壓式 PFC 轉換器，其輸入全橋式整流器之導通損占有轉換器相當比重，為了提升 PFC 前級之效率，許多無橋式 PFC 電路架構被提出，圖 7.7 所示為最早被提出之無橋式升壓式 PFC 轉換器，其動作如圖 7.8 所示，正半週時由 S_1 及 D_1 動作，S_2 之旁路二極體當成整流二極體，在正半週時一直維持導通。在負半週時則由 S_2 及 D_2 動作，S_1 之旁路二極體當成整流二極體，在負半週時一直維持導通。由於低頻切換，因此 S_1 及 S_2 之 MOSFET 開關旁路二極體足敷使用。相較於傳統架構，無論是在交流之正半週亦或負半週，其在開關導通時與交流輸入電壓形成導通路徑上之元件與傳統架構相較少了一二極體，確實可以降低導通損。但其缺點為直流輸出電壓之負端與交流電壓之相對電壓為高頻變動且大小為直流輸出電壓準位，此高頻電壓變化將對輸出負端與市電接地端之雜散電容充放，造成較嚴重之共模雜訊，因而相對需要在 EMI 的防制電路上付出代價。

圖 7.7

無橋式升壓式 PFC 轉換器

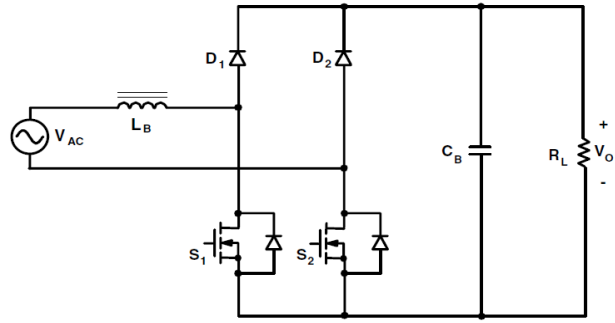


圖 7.8

無橋式升壓式 PFC 轉換器之動作：

(a)正半週

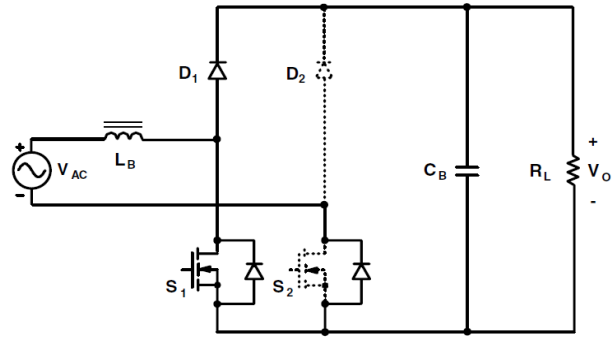
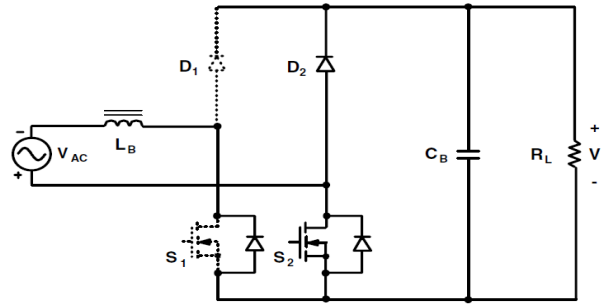


圖 7.8

無橋式升壓式 PFC 轉換器之動作：

(b)負半週



為改良圖 7.7 之高頻共模電流缺點，圖 7.9 進一步使用雙向開關之無橋式升壓式 PFC 轉換器，無論在正半週或負半週，開關導通時導通路徑上僅有二關關，開關截止時則由 D_1 及 D_2 作高頻切換將能量轉移至直流鏈，二極體 D_3 及 D_4 則作低頻切換，因此直流輸出電壓之負端與交流電壓之相對電壓為低頻變動。與圖 7.7 相較，無論是導通損亦或共模電流均有相當改善，此外二開關之觸發為共地，觸發電路設計較為簡單。

圖 7.9

使用雙向開關之無橋式升壓式 PFC 轉換器

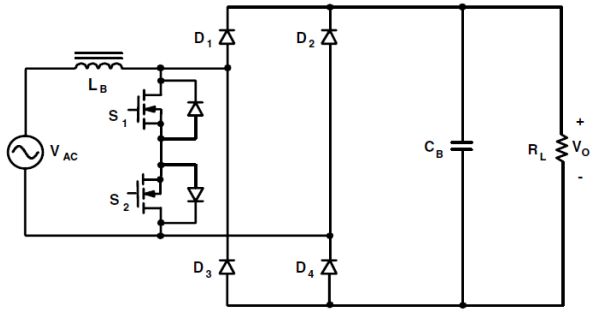


圖 7.10 所示為採用雙升壓式電路之無橋式升壓式 PFC 轉換器，L_{B1}-S_1-D_1-D_4 與 L_{B2}-S_2-D_2-D_3 分別在交流之正半週與負半週形成二獨立之升壓式轉換器，D_3 及 D_4 為低頻切換，因此直流輸出電壓之負端與交流電壓之相對電壓為低頻變動。其導通路徑之元件數與圖 7.7 之電路相當，但使用二電感，電感損耗分散且電感較易扁平及縮小化設計，二開關之觸發為共地且可以使用相同之 PWM 信號，觸發電路設計較為簡單。

圖 7.10

採用雙升壓式電路之無橋式升壓式 PFC 轉換器

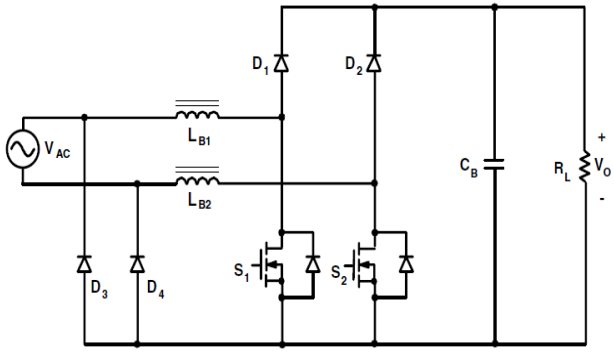


圖 7.11 所示為虛擬圖騰級(Pseudo totem-pole)無橋式升壓式 PFC 轉換器，L_{B1}-S_1-D_1-D_4 與 L_{B2}-S_2-D_2-D_3 分別在交流之正半週與負半週形成二獨立之升壓式轉換器，D_3 及 D_4 為低頻切換，因此直流輸出電壓之負端與交流電壓之相對電壓為低頻變動。其導通路徑之元件數與圖 7.10 之雙升壓式電路相當，但二開關之觸發為不共地且必須使用兩組分離之 PWM 信號，觸發電路設計較為複雜。

圖 7.11

虛擬圖騰級(Pseudo totem-pole)無橋式升壓式 PFC 轉換器

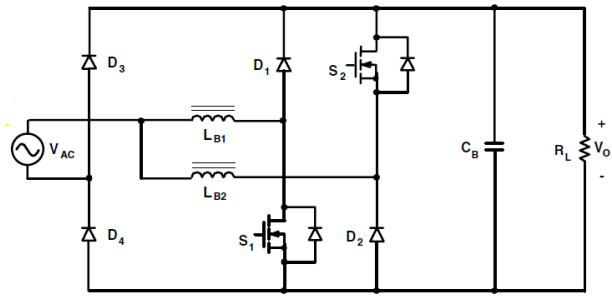
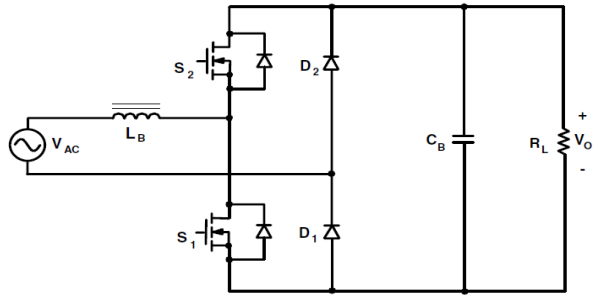


圖 7.12 所示為圖騰級(Totem-pole)無橋式升壓式 PFC 轉換器，包含一開關臂與一二極體臂，L_B-S_1-D_1-S_2 之旁路二極體與 L_B-S_2-D_2-S_1 之旁路二極體分別在交流之正半週與負半週形成二升壓式轉換路徑，D_1 及 D_2 為低頻切換，因此直流輸出電壓之負端與交流電壓之相對電壓為低頻變動。其導通時路徑上之元件數與圖 7.10 之雙升壓式電路相當，優點為使用元件數量較少，二開關之觸發可使用 bootstrap 驅動電路，亦可使用同步整流方式觸發，使效率進一步提升。

圖 7.12

圖騰級(Totem-pole)無橋式升壓式 PFC 轉換器



電路模擬

無橋式升壓式 PFC 轉換器規格

- P_o=115W, V_o=40Vac/60Hz,
- V_d = 70V, v_{tri}=18kHz/5V_{pp}, k_s = 1/3.472V/A, k_v = 1/162.2,
- PWM 空白時間 2μs,
- L = 1.323mH, C_d= 330μF

電流迴路設計

採用二類控制器，但將低通部份置於感測迴路(f_c =20kHz)，控制器本身採用比例積分控制($G_{ca} = \frac{s+1000}{s}$)，另外 PWM 之控制電壓亦加入一前向控制信號(= $\frac{V_d - V_{in}}{k_{pwm}}, k_{pwm} = \frac{V_d}{v_t}$)

電壓迴路設計

- 以滿載電阻來設計，$R = 49\Omega$，因此
  $$H_{dc}(s) = \frac{26.21}{s+61.84},$$
- 設定 $f_c = 20Hz, \omega_c = 125rad/s$
- 選擇 G_v之 p=180rad/s, z=30rad/s，則
  $$G_v(s) = \frac{30(s+30)}{s(s+180)}$$

PSIM 模擬

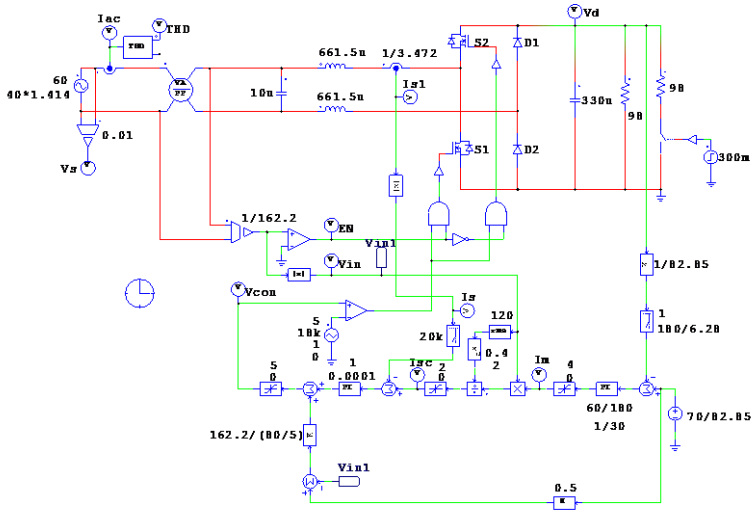


圖 7.13 圖騰級無橋式升壓式 PFC 轉換器之模擬電路

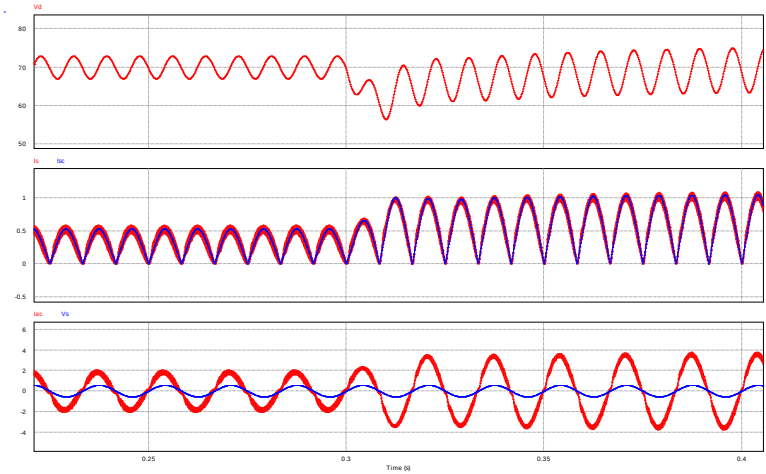


圖 7.14 圖騰級無橋式升壓式 PFC 轉換器之模擬結果

SimCoder 程式規劃及電路模擬

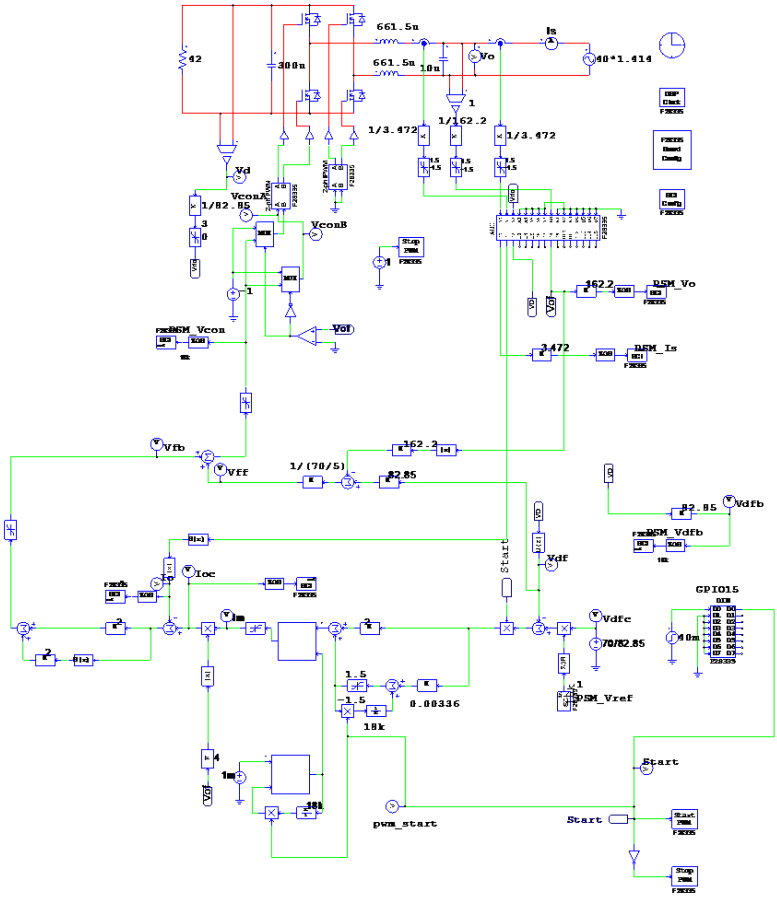


圖 7.15 SimCoder 所建立之圖騰級無橋式升壓式 PFC 轉換器之模擬電路

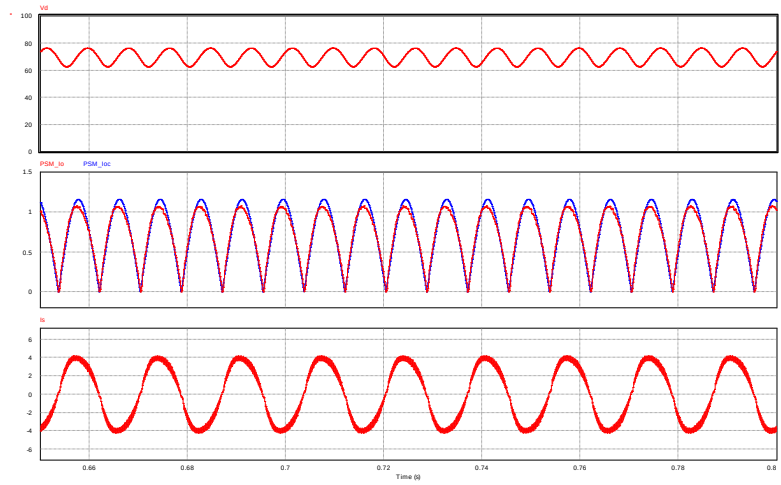


圖 7.16 SimCoder 所建立之圖騰級無橋式升壓式 PFC 轉換器之模擬結果

實驗量測

實驗設備與教具配置如圖 7.17，交流電源供應器 APS-7050 連接到 PEK-110 的端子 J7，端子 J1 連接到直流負載 PEL-2040A。圖 7.19 為 DSP 示波器中 RS232 傳回之波形。

7.17

實驗設備配置圖

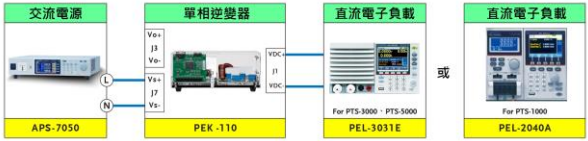


圖 7.18

圖騰級無橋式升壓式 PFC 轉換器電路量測結果

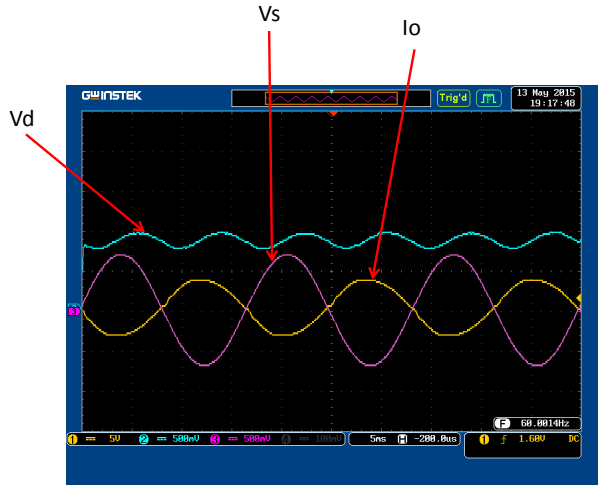


圖 7.19 (a)

DSP 示波器中 RS232 傳回之波形

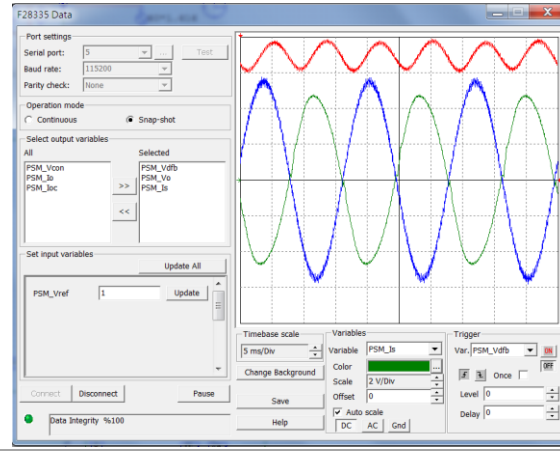
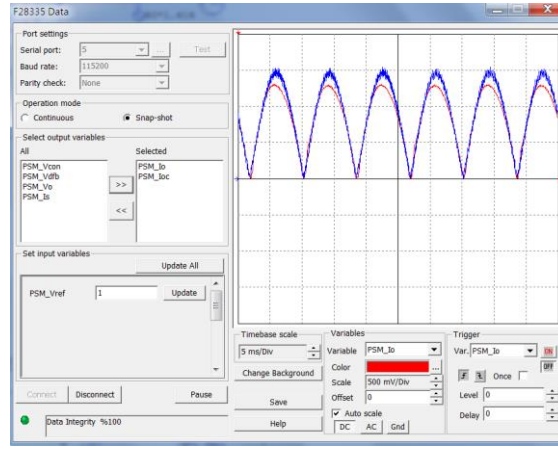


圖 7.19 (b)

DSP 示波器中 RS232 傳回之波形



實驗 5 全橋式 AC-DC 切換式整流器

實驗目的

學習全橋式 AC-DC 切換式整流器之原理、電流迴路及電壓迴路控制器設計、硬體規劃及 SimCoder 程式撰寫等。

實驗原理

單相切換式整流器之原理與設計

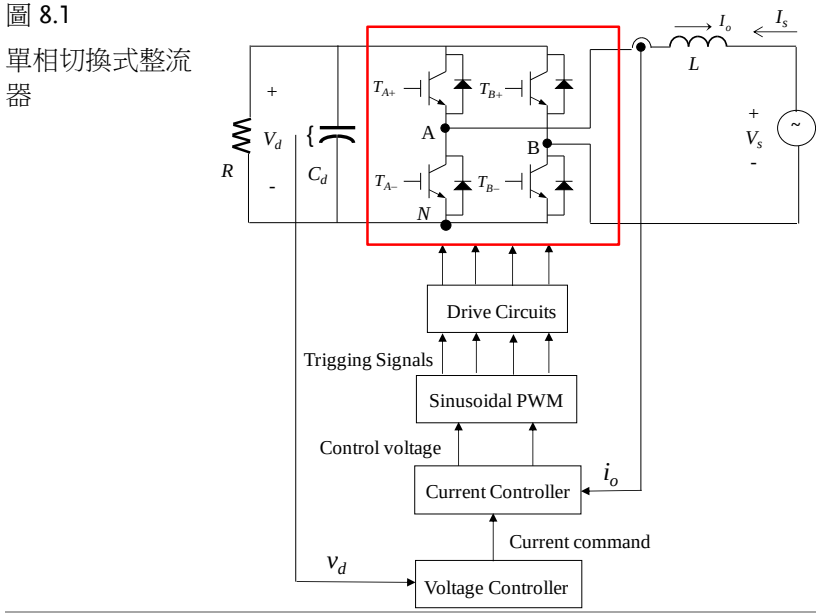
單相切換式整流器如圖 8.1 所示，其輸入為交流電壓 V_s，輸出為直流電壓 V_d，採用雙迴路控制，外迴路為直流電壓控制迴路，用以調整 V_d，內迴路為電流迴路，用以調整使輸入電流(I_s = -I_o)為與輸入電壓同相且低失真之正弦波，達到單位功率因數之目的。

單相切換式整流器之電流迴路模型仍然適用前述變流器之模型，其控制方塊圖設計如圖 8.2 所示，其中 k_s 及 k_v 分別為電流及電壓之感測增益，電流控制器乃利用迴授與前向(feedforward)控制並用，假設輸入電流可以與輸入電壓同相達到單位功因，前向控制信號 v_o^*可用以消除 V_s 對電流迴路之擾動，v_o^*為輸出電壓之命令。如此電流迴路可以回授迴路求得為：

$$\frac{i_o}{i_o^*} = \frac{\frac{k_s k_1 k_{pwm}}{L}}{s + \frac{k_s k_1 k_{pwm}}{L}} = \frac{u_R}{s + u_R}, \quad u_R = \frac{k_s k_1 k_{pwm}}{L} \tag{8.1}$$

其中 u_R 即相當於輸入電流迴路之頻寬，其可以由增益 k_1 設定，一般頻寬設定為切換頻率之 1/10~1/5。

圖 8.1
單相切換式整流器



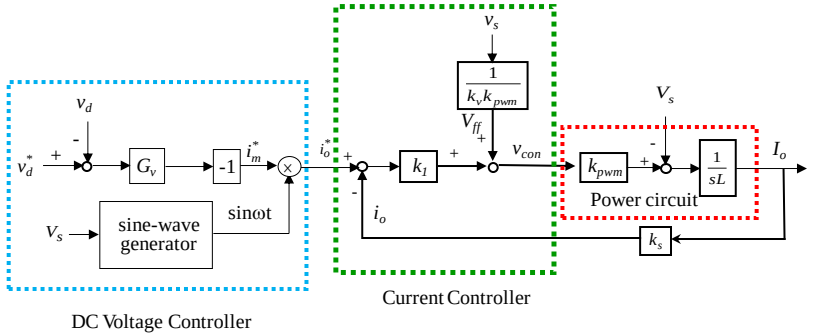


圖 8.2 控制電路設計

整流器之電流命令(i_o^*)則由外迴路之直流電壓控制器 G_v所產生，其利用電壓調整之誤差乘上一與輸入電壓同步之單位正弦波($sin\omega t$)所產生。G_v之設計必須推導直流電壓迴路之模型，可根據圖 8.3(a)在單位功因下之等效電路來推導，交流側之輸入功率為：

$$P_{ac} = V_{s(p)} \sin \omega t \cdot I_m \sin \omega t = \frac{V_{s(p)} I_m}{2} - \frac{V_{s(p)} I_m}{2} \cos 2\omega t \qquad (8.2)$$

$$= \overline{P}_{ac} + \tilde{P}_{ac2}$$

其除了一直流項之外亦包含一二次之諧波項，此二次諧波項將造成直流電壓二次之漣波成份。直流側之平均功率等於交流側功率之直流項

$$\overline{P}_{ac} = P_{dc}$$

將交流電流源反應至直流側可得圖 8.3(b)之等效電路，根據(8.2)可得：

$$\frac{V_{s(p)} I_m}{2} = V_d I_d \tag{8.3}$$

$$I_d = \frac{V_{s(p)} I_m}{2V_d} = k_{dc} I_m \tag{8.4}$$

由直流電流源 Id 對 Cd 電容充電可得電壓迴路之模型為：

$$\frac{V_d}{I_m} = \frac{k_{dc} R}{1 + sC_d R} = \frac{\frac{k_{dc}}{C_d}}{s + \frac{1}{RC_d}} = \frac{k_{dc} / C_d}{s + a}，\ k_{dc} = \frac{V_{s(p)}}{2V_d}，a = \frac{1}{RC_d} \tag{8.5}$$

電壓控制器 G_v 可根據圖 8.4(a)電壓控制迴路方塊圖來設計，其中由於電流迴路較電壓迴路頻寬要寬出非常多，因此可以將(8.1)之電流迴路響應簡化為等於 1，因此 I_m^* 至實際 I_o 之電流振幅 I_m 之增益即為電流感測比例 k_s 之倒數，電壓迴路 H_{dc} 之波德圖可繪出如圖 8.4(b)所示。考慮直流電壓具有二次漣波，為使市電電流命令為低失真，電壓迴路之頻寬必須遠低於 120Hz 以衰減電壓之二次漣波，因此 G_v 採用 type II 補償器(即 PI + Low-Pass)之設計方式，其波德圖如圖 8.4(b)所示，閉迴路之增益(loop response)以及若僅採用 PI 控制器之響應亦繪於圖 8.4(b)中以茲比較。

圖 8.3

單相切換式整流器電壓迴路之等效電路：

(a)在單位功因下之等效電路

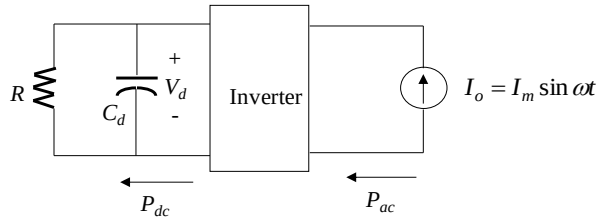


圖 8.3

單相切換式整流器電壓迴路之等效電路：

(b)將交流電流源反應至直流側之等效電路

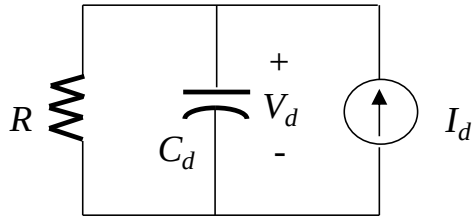


圖 8.4

直流電壓控制器設計：

(a)電壓迴路控制方塊圖

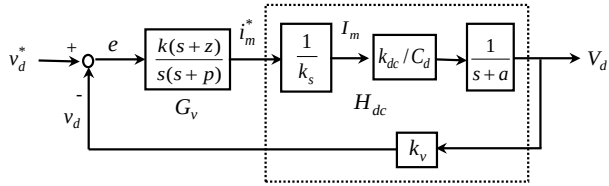
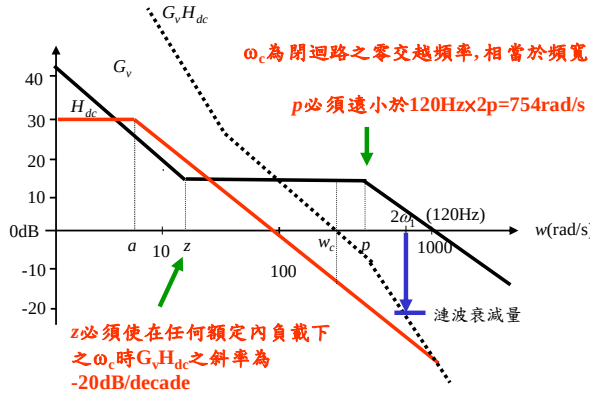


圖 8.4

直流電壓控制器設計：

(b)電壓迴路波德圖



電路模擬

整流器規格

- P_o=115W, V_o=40Vac/60Hz,
- V_d = 70V, v_{tri}=18kHz/5V_{pp}, k_s = 1/3.472V/A, k_v = 1/162.2,
- 採用單電壓極性切換, 空白時間 2μs,
- L = 1.323mH, C_d= 330μ

電流迴路設計

$$k_{pwm} = 70/2.5 = 28$$

- $f_{ci} = 18k/10 = 1.8kHz$

$$u_R = 1.8k \times 2\pi = 11310 rad/s$$

$$k_1 = 1.68$$

電壓迴路設計

- 以滿載電阻來設計， $R = 49\Omega$ ，因此

$$H_{dc}(s) = \frac{26.21}{s + 61.84},$$

- 設定 $f_c = 20Hz, \omega_c = 125 rad/s$
- 選擇 G_v 之 p=180rad/s, z=30rad/s，則

$$G_v(s) = \frac{1141(s+30)}{s(s+180)}$$

PSIM 模擬

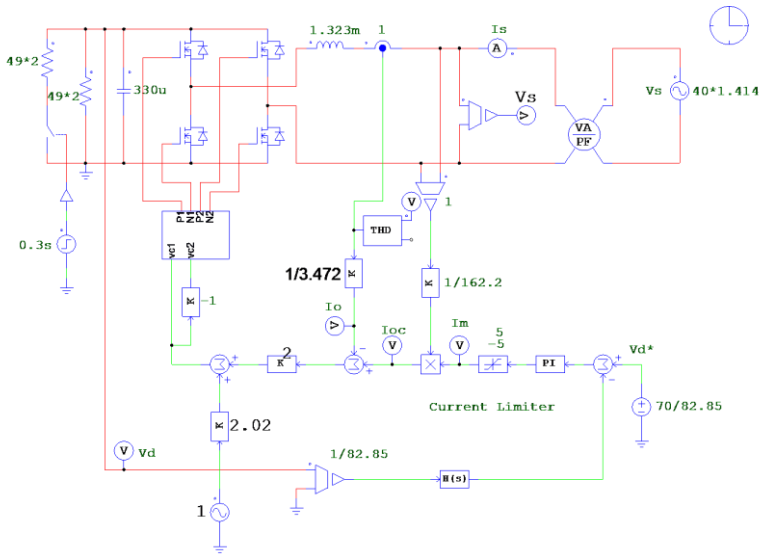


圖 8.5 負載瞬間從 98Ω變化到 49Ω之模擬電路(電流迴路為 P 控制)

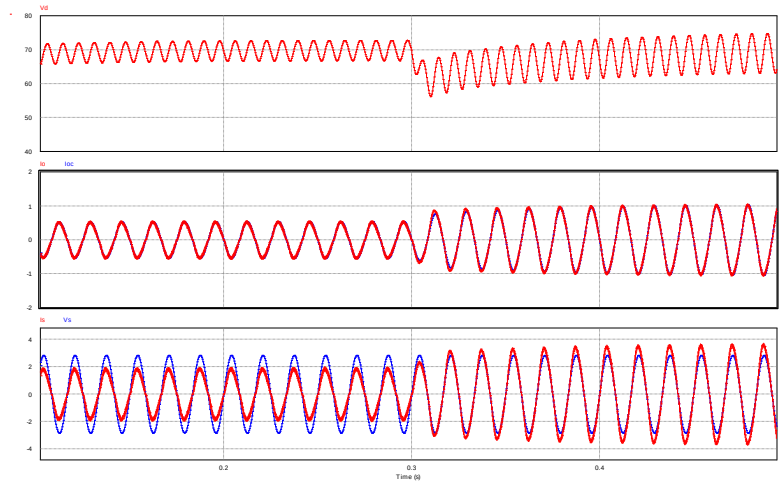


圖 8.6 負載瞬間從 98Ω變化到 49Ω之模擬結果

SimCoder 程式規劃及電路模擬

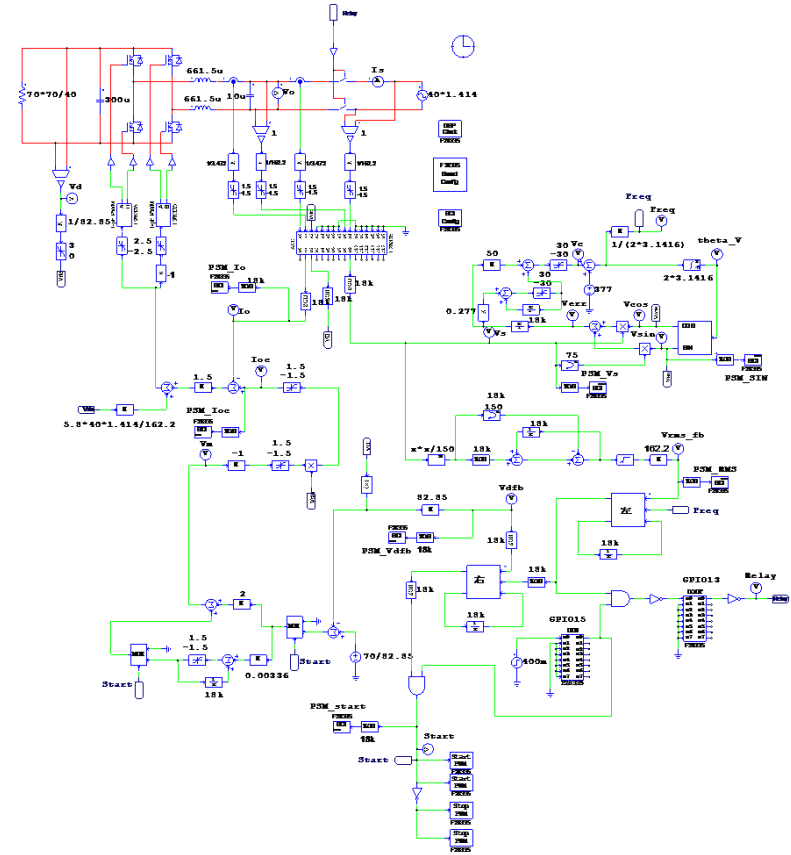


圖 8.7 以 SimCoder 建立之單相切換式整流器模擬電路

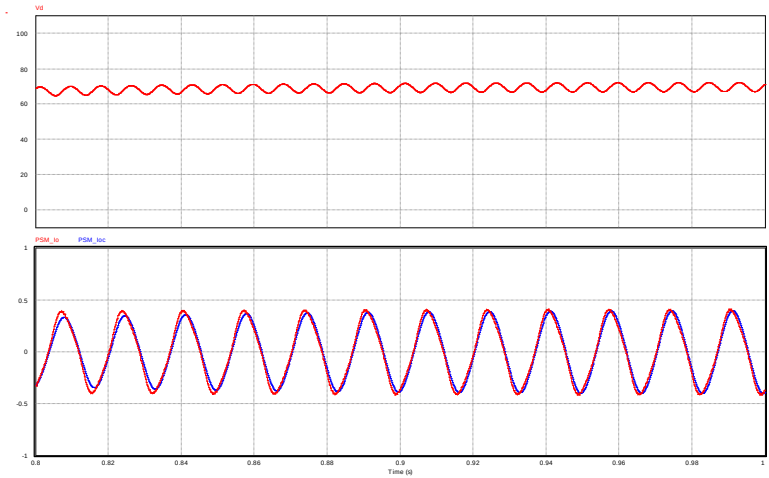


圖 8.8 以 SimCoder 建立之單相切換式整流器電路模擬結果

實驗量測

實驗設備與教具配置如圖 8.9，交流電源供應器 APS-7050 連接到 PEK-110 的端子 J7，端子 J1 連接到直流負載 PEL-2040A。圖 8.10 為實際量測波形。

圖 8.9

實驗設備配置圖

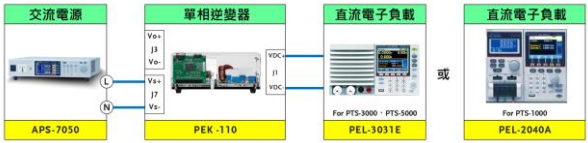
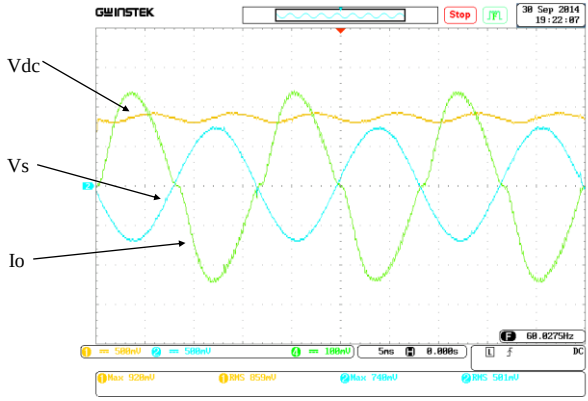


圖 8.10

單相切換式整流器電路量測結果

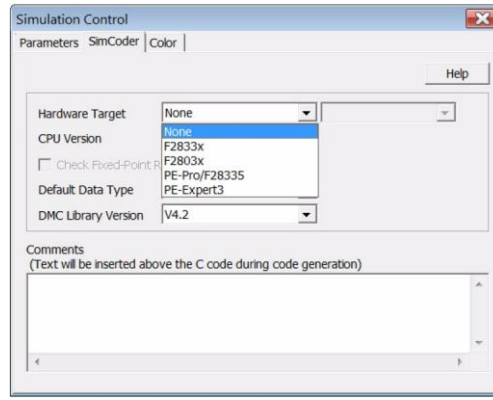


附錄 A SimCoder 概述

SimCoder 為 PSIM 軟體的一個附加工具模組。可由 PSIM 的電路圖產生 C 程式碼(code)，藉由特定的硬體標的資料庫，SimCoder 產生的 C 程式碼可直接在 DSP 硬體平台上執行。本手冊將說明如何使用 SimCoder。

SimCoder 於模擬控制上設定

SimCoder 在模擬控制方塊的 SimCoder 表格中完成參數設定，如下圖，須適當地設定參數以便能正確地產生程式碼。



設定過程說明如下

所支援的硬體標的(Hardware Targets)	• SimCoder 支援下表的硬體標的： • None 產生的程式碼只能用於模擬，無法用於任何特定硬體標的。 • F2833x 產生的程式碼可用於 TI 的浮點 TMSF2833x 系列 DSP。 • F2803x 產生的程式碼可用於 TI 的定點 TMSF2803x 系列 DSP。 • PE-Pro/F28335 產生的程式碼可用於 PE-Pro/F28335 DSP 板，為一塊由 Myway 公司所生產的 DSP 板，它使用 TI 的浮點 DSP TMSF28335 和 Myway 的 PE-OS 資料庫。 • PE-Expert3 產生的程式碼可用於 PE-Expert3 DSP 硬體，為一塊由 Myway 公司所生產的 DSP 開發平台，它使用 TI 的浮點 DSP TMS320C6713 和 Myway 的 PE-OS 資料庫。
專案組態 (Project Configuration)	針對 F2833x 與 F2803x 標的，專案組態可設定為 RAM Debug, RAM Release, Flash Release 或 Flash RAM Release。對於 PE-Expert3 標的，專案組態可設定為 PE-View9 或 PE-View8。

CPU 版本	對於 F2803x 與 2833x，隨後的 CPU 版本可選擇。 • 針對 *F2833x*，有 F28335、F28334 和 F28332 可供選擇。 • 針對 *F2803x*，各種腳位配置的 F28030 到 F28035 可供選擇。
檢查定點範圍	這只針對 F2803x 系列，如果選取此項目，SimCoder 在模擬時會檢查數據並提供數據範圍的列表。在此表中，接近或超過此範圍的資料會被特別標注。
預設資料形式 (Default Data Type)	當硬體標的為浮點形式，此部分會自動被選取。當沒有硬體標的或其為定點形式，則必須在下拉式選單中選擇一個可用的初始資料形式。 • 針對 *F2803x*，有如下可供選擇：整數、IQ1、IQ2、…、IQ30。 • 當沒有硬體標的，其選擇如下：浮動、整數、IQ1、IQ2、…、IQ30。
DMC 元件資料庫版本	德州儀器的數位馬達控制(Digital Motor Control, DMC)資料庫是由 C 函數(或巨集)組成，針對 C2000 DSP 器件的馬達控制使用者所開發。為了善用這項資源，SimCoder 整合了 DMC 資料庫函數到 PSIM 的元件資料庫以便產生程式碼。 TI 在某些巨集已經發佈不同的版本，SimCoder 支援 V4.0、V4.1、和 V4.2。
⚠注意	一旦選用某個版本，其他版本將會被關閉。
注解	注解區在 SimCoder 表格的下方可供使用者在 SimCoder 產生的程式碼中加入注解，此區的所有文字會被加入在 C 程式碼的開頭當作注解。

產生程式碼的元件(Elements)

所有在 Elements>>Event Control 和 Elements>>SimCoder 中的元件皆可用於產生程式碼。

每個硬體標的元件可在 Elements>>SimCoder 的子選單 F2833x Target, F2803x Target, PE-Pro/F28335 Target 和 PE-Expert3 Target 中找到。

許多在 PSIM 標準資料庫中的元件也可用於產生程式碼，為了區別在標準資料庫中可用及不可用於產生程式碼的元件，在 Options>>Settings>>Advanced，如果選項框*"show image next to elements that can be used for code generation"*被選取，一個小圖案  會出現在這些可用於產生程式碼的元件前。

此外，當選取此選項框時，F2833x 和 F2803x Target 的所有元件前會出現  圖案，PE-Pro/F28335 Target 出現  ，PE-Expert3 Target 出現  。

在標準資料庫中可用於產生程式碼的元件列表，請參照 32 頁。

附錄 B 產生程式碼 – 逐步介紹

使用 SimCoder 自動產生程式碼的設計步驟如下：

1. 在 PSIM 下系統之控制部分使用連續模式進行設計與模擬。
2. 將系統的控制部分轉變為離散模式並重新模擬驗證。
3. 如無硬體標的亦可將控制部分置於子電路內並產生程式碼。
4. 如具有硬體標的，使用硬體標的元件去取代原系統之元件，並經由模擬驗證結果，最後產生程式碼。

然而，前兩步並非強制性，也可不經模擬直接建立線路圖並產生程式碼。

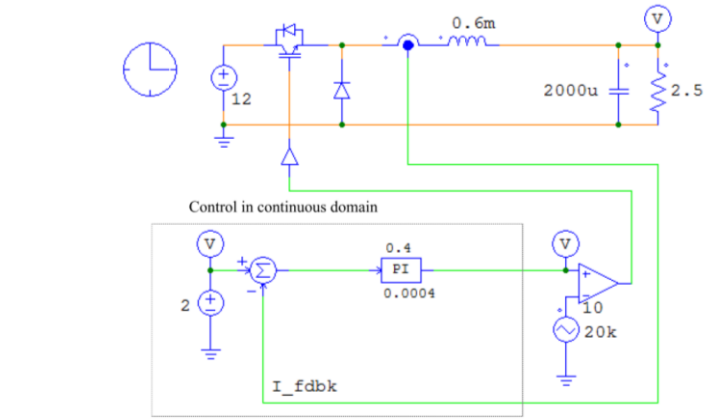
⚠注意 只有當控制部分在離散模式下才能產生程式碼，連續模式則不能。因此，對於 SimCoder 而言，Digital Control Module 是必備的。

下列章節將用簡單的例子說明產生程式碼的過程。

連續模式系統

通常系統會先在連續模式下設計與模擬，下面為一個簡單的電流回授直流轉換器電路，在連續模式下設計一個有 PI 控制器的控制電路，PI 的增益 *k* 為 0.4 與時間常數 *T* 為 0.0004，切換頻率為 20kHz。

此範例的目的為針對虛線框內的控制電路產生 C 程式碼，要產生程式碼，第一步為將 S 平面(S-domain)的類比 PI 控制器轉換成 Z 平面(Z-domain)的數位 PI 控制器。

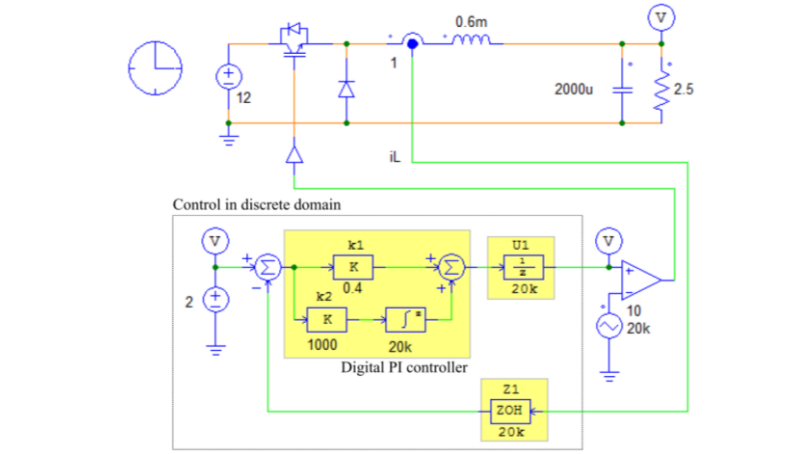


離散模式系統

要將類比控制器轉換成數位控制器，可使用 Digital Control Module 附帶的 *s2z Converter*，該程式可在 **Utilities>>s2z Converter** 啟動。

類比控制器可用不同的方法轉換成數位控制器，最常用的為 Bilinear(也可稱為 Tustin 或 Trapezoidal)法和 Backward Euler 法。

此範例中，取樣頻率與切換頻率同為 20kHz，使用 Backward Euler 法將類比 PI 控制器轉換成數位 PI 控制器。由轉換程式得知： PI 控制器的比例部分參數為 k_1=0.4，積分部分參數為 k_2=1000，數位 PI 控制器的電路如下圖

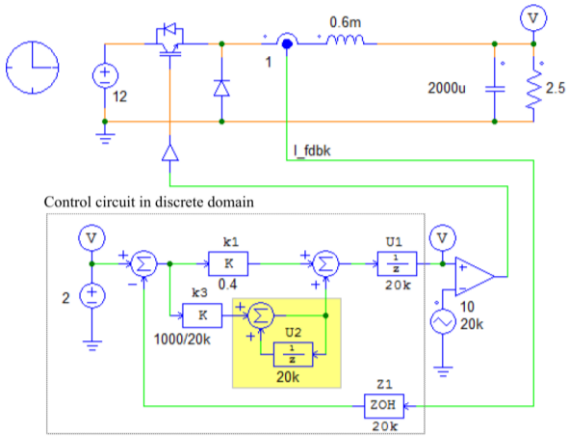


相較於連續模式下的控制電路，此電路有三處變更，以黃色框強調：(i)數位 PI 控制器取代類比 PI 控制器，數位積分器的演算法標誌設定為 1(針對 Backward Euler 法)且取樣頻率設為 20kHz，由轉換程式取得的增益 k_1 和 k_2 敘述如上。(ii)回授電流取樣 i_L 以 zero-order-hold(ZOH)方塊 Z_1 來模擬在數位硬體實現的 A/D 轉換器。(iii)數位控制內部一個週期延遲的實現以單位延遲(unit-delay)方塊 U_1 用來模擬。延遲的原因為，通常在一個週期開始時取樣，在週期內計算控制器參數，但因為完成計算需花時間，新的計算量通常等到下一個週期開始才使用。

請注意轉換後的數位控制器應產生穩定的控制迴路和所期望的性能，如果數位控制器模擬的結果不穩定或不如預期，則須回頭查看類比控制系統，重新設計類比控制器並重覆這個過程。

藉由 Backward Euler 法，也可在時域中將輸入輸出關係做如下描述：　$y(n) = y(n-1) + T_s * u(n)$

其中 *y(n)*和 *u(n)*為當時的輸出與輸入，*y(n-1)*為先前取樣週期的輸出，T_s 為取樣週期。使用上述的方程式，可用加法器(summer)和單位延遲(unity-delay)方塊取代上述電路的離散積分器，如下圖所示



注意

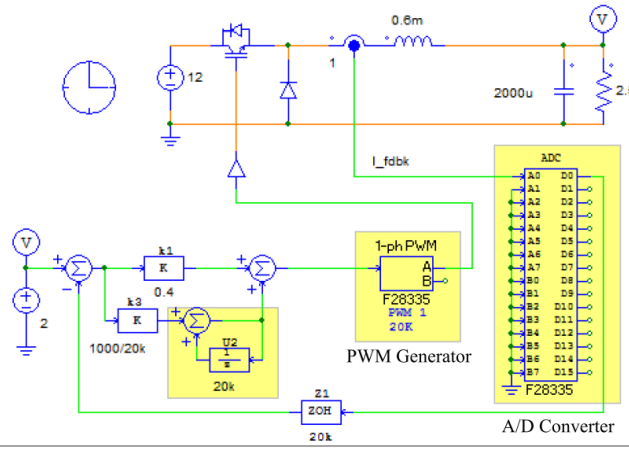
因為方程式中的 Ts 因子，比例方塊的增益 k2 須除以 20kHz 的取樣頻率，此電路的優點在於更容易啟動或終止積分器積分。

有了離散模式下的控制電路，可往下進行下一個步驟。

產生硬體標的之程式碼

為產生特定硬體標的程式碼，所設計電路須利用相關硬體標的元件來修改，變數須因應硬體元件的有效範圍做適當的調整。

下面為同樣的範例電路但增加 F2833x 的硬體標的元件，為了更清楚說明，硬體元件以黃色來標注。



此範例電路圖，做了如下變更：

1. 在電流感測器與控制子電路間加上 A/D 轉換器，須特別注意 A/D 轉換器的輸入範圍。如果電流感測器的輸出超過 A/D 轉換器的範圍，它必須做相對應的縮減，對於這個例子，A/D 轉換器的設定如下：

- ADC Mode : Start-stop (8-channel)。這意味著 A/D 轉換由 PWM 產生器觸發，且只使用半數的 ADC 轉換器。
- Ch A0 Mode : DC。設定輸入信號範圍為 0~+3V。
- Ch A0 Gain : 1.0。

2. 以硬體 PWM 產生器取代比較器與載波，此範例的硬體 PWM 產生器相關設定如下：

- PWM Source : PWM1。定義 F28335 處理器的 PWM 模組。
- Output Mode : Use PWM A。定義 PWM 的輸出端口。
- Sampling Frequency : 20k。定義取樣頻率為 20kHz。
- Carrier Wave Type : Sawtooth(start high)。此設定往後在附錄 E F2833x 硬體標的會說明。
- Trigger ADC : Trigger ADC Group A。此設定往後在附錄 E F2833x 硬體標的會說明。
- ADC Trigger Position : 0。此設定往後在附錄 E F2833x 硬體標的會說明。
- Peak to Peak Value : 10。定義 PWM 產生器的鋸齒波信號範圍。

3. 因為 PWM 產生器本身就包含一個取樣週期的延遲，故移除前一圖中之單位延遲方塊 U1。

4. 在 Simulation Control 中的 SimCoder 選項表：

- Hardware Type 設定 F2833x 以及 RAM Debug。
- CPU Version 設定 F28335。
- Default Fixed-Point Position 並不適用，因為它是浮點。

5. 使用者可在產生的程式碼前加上註解(comment)，進入 Simulation Control 的 SimCoder 選項表，便可輸入或編輯註解。

   為了在增加硬體元件後檢查這些變更是否有效，執行全系統模擬，其結果應該很接近 2.2 節數位控制系統所模擬的結果。
   一旦驗證了模擬結果，可點選 **Simulate>> Generate Code** 產生 C 程式碼，針對 F2833x 硬體所產生的程式碼可不須做任何變更便可直接燒錄。

對於上述系統，SimCoder 產生的程式碼如下

```
/*********************************************************************************
// This code is created by SimCoder Version 9.3.3 for TI F2833x Hardware Target
//
// SimCoder is copyright by Powersim Inc., 2009-2013
//
// Date: February 24, 2014 14:36:33
*********************************************************************************/
#include<math.h>
#include"PS_bios.h"
typedef float DefaultType;
#defineGetCurTime() PS_GetSysTimer()

interrupt void Task();

DefaultTypefGbliref = 0;
DefaultTypefGblU2 = 0;

interrupt void Task()
{
    DefaultTypefU2, fSUMP1, fSUMP3, fk3, fk1, fSUM1, fZ1, fTI_ADC1, fVDC2;

    PS_EnableIntr();
    fU2 = fGblU2;
```

```
    fTI_ADC1 = PS_GetDcAdc(0);
    fVDC2 = 2;
    fZ1 = fTI_ADC1;
    fSUM1 = fVDC2 - fZ1;
    fk1 = fSUM1 * 0.4;
    fk3 = fSUM1 * (1000.0/20000);
    fSUMP3 = fk3 + fU2;
    fSUMP1 = fk1 + fSUMP3;
    PS_SetPwm1RateSH(fSUMP1);
#ifdef DEBUG
    fGbliref = fVDC2;
#endif
    fGblU2 = fSUMP3;
    PS_ExitPwm1General();
}

void Initialize(void)
{
    PS_SysInit(30, 10);
    PS_StartStopPwmClock(0);
    PS_InitTimer(0, 0xffffffff);
    PS_InitPwm(1, 0, 20000*1, (4e-6)*1e6, PWM_POSI_ONLY, 42822);// pwnNo, waveType, frequency, deadtime,
outtype
    PS_SetPwmPeakOffset(1, 10, 0, 1.0/10);
    PS_SetPwmIntrType(1, ePwmIntrAdc0, 1, 0);
    PS_SetPwmVector(1, ePwmIntrAdc0, Task);
    PS_SetPwmTzAct(1, eTZHighImpedance);
    PS_ResetAdcConvSeq();
    PS_SetAdcConvSeq(eAdc0Intr, 0, 1.0);
    PS_AdcInit(1, !1);

    PS_StartStopPwmClock(1);
}

void main()
{
    Initialize();
    PS_EnableIntr();   // Enable Global interrupt INTM
    PS_EnableDbgm();
    for (;;) {
    }
}
```

產生的程式碼有著如下的結構：

- *Interrupt void Task()*：中斷服務程序為 20kHz，每 20kHz 被呼叫一次。
- *void Initialize()*：初始化程序，它將對硬體初始化。
- *void main()*：主程式，他呼叫初始化程序，並執行一個無限迴圈。

注意

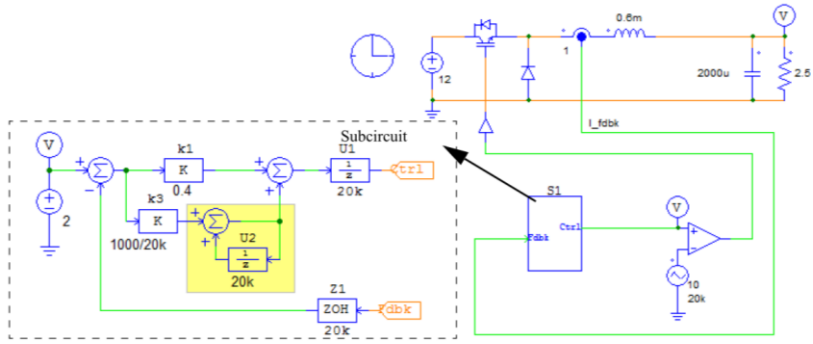
在此範例中，所有的控制方塊都在 20kHz 的取樣頻率下運作，如果有方塊在不同的取樣頻率下運作，將會建立另一個服務程序。一個中斷服務程序對應一個取樣頻率。對於那些沒有相對應取樣頻率的方塊，其對應的程式碼會被放在主程式內。

這個程式碼和所有必須的文件儲存在與原電路相同目錄的子資料夾下，使用者可在 TI Code Composer Studio(CCS)環境中載入這些文件，編譯程式碼，並燒錄到 DSP 硬體做即時操作。

子電路(Subcircuit)產生程式碼

在 PSIM 中，可以將電路的某部分做成一個子電路，SimCoder 可以針對每個子電路產生程式碼，前提是子電路中的每個元件都可支援產生程式碼，在 2.5.3 節會列出一些限制。

為了說明子電路產生程式碼，將持續使用 2.3 節的範例電路，虛線框內的控制電路將會置換成一個子電路，如下圖



選取虛線框內的電路以產生子電路。擊點滑鼠右鍵將會出現一個下拉式選單，選擇 **Create Subcircuit** 並設定子電路名稱。

子電路產生程式碼不包含比較器和 PWM 載波源。這樣安排的一個原因是，大部分的硬體設置這兩個功能由外部硬體實現或嵌入至微控制器的周邊介面。另一個原因是，程式碼每 20kHz 的取樣頻率才執行一次，但載波和比較器在每個模擬時間間隔(time step)都需計算。當使用 SimCoder 產生程式碼，每個元件的取樣頻率都需要定義。比較器的兩個輸入，一個是 20kHz 取樣頻率的控制器，另一個是未定義的載波源。在這種情形下，SimCoder 會假定比較器的兩個輸入都有相同的取樣頻率，由有定義取樣頻率者決定。

SimCoder 可針對子電路產生程式碼做為模擬或具有硬體標時使用，這兩種形式的程式碼不可交替使用。針對模擬產生的程式碼不可用在硬體標的，反之亦然。這兩種情形再下面章節或說明。

子電路為模擬產生程式碼

針對模擬產生程式碼，步驟如下

1. 在主電路中，在子電路方塊擊點滑鼠右鍵並選擇 Attributes。
2. 在 Attribute 對話視窗下，擊點按鈕 Generate Code，並選擇 Generate Code for Simulation。
3. 如果需要，使用者可在程式碼的開頭增加註解。在 Simulation Control 的 SimCoder 表，擊點 Generate Code 按鈕之前在對話視窗輸入或編輯註解。

產生的程式碼開頭如下：

```
/*********************************************************************
// This code is created by SimCoder Version 9.3.3
//
// SimCoder is copyright by Powersim Inc., 2009-2013
//
// Date: February 24, 2014 14:55:33
*********************************************************************/
#include    <stdio.h>
#include    <math.h>
#define    ANALOG_DIGIT_MID    0.5
#define INT_START_SAMPLING_RATE 1999999000L
#define NORM_START_SAMPLING_RATE 2000000000L

typedef void (*TimerIntFunc)(void);
typedef double DefaultType;
DefaultType    *inAry = NULL, *outAry;
DefaultType    *inTmErr = NULL, *outTmErr;

double fCurTime;
double GetCurTime() {return fCurTime;}

/* The input/output node definition for C/DLL block.
    The 2nd display node (outAry[1]): From element 'S1_iref'.
*/
/* The C block for the generated code has the following additional output port(s):
    2 - S1.iref
*/
void _SetVP6(int bRoutine, DefaultType fVal);
void InitInOutArray()
{... ...}
void FreeInOutArray()
{ ... ...}

void CopyInArray(double* in)
{ ... ...}

void CopyOutArray(double* out)
{ ... ...}

void Task();
void TaskS1(DefaultType fIn0, DefaultType *fOut0);

DefaultType    fGblS1_U1 = 0;
DefaultType    fGblS1_U2 = 0;
void TaskS1(DefaultType fIn0, DefaultType *fOut0)
{ ... ...}

void Task()
{
    TaskS1(inAry[0],&outAry[0]);
}

typedef struct {
    TimerIntFunc    func;
    long            samprate;
    double            tmLastIntr;
}    TimeChk;
#define NUM_TIMER_INTR    1
TimeChk    lGbl_TimeOverChk[NUM_TIMER_INTR] = {
    {Task, 20000, 0}};

void InitAllTaskPtr(void)
{
    lGbl_TimeOverChk[0].func = Task;
```

```
        val = fVal;
    }
    outAry[1] = val;
}
```

由子電路產生的程式碼結尾，包含 SimulationStep 函數、SimulationBegin 函數以及 SimulationEnd 函數，如下所示。這些函數可被用在能取代子電路的 C Block 中。

```
void SimulationStep(
            double t, double delt, double *in, double *out,
             int *pnError, char * szErrorMsg,
             void ** reserved_UserData, int reserved_ThreadIndex, void * reserved_AppPtr)
{ ... ...}

void SimulationBegin(
            const char *szId, int nInputCount, int nOutputCount,
             int nParameterCount, const char ** pszParameters,
             int *pnError, char * szErrorMsg,
             void ** reserved_UserData, int reserved_ThreadIndex, void * reserved_AppPtr)
{
    InitInOutArray();
}

void SimulationEnd(const char *szId, void ** reserved_UserData, int reserved_ThreadIndex, void *
reserved_AppPtr)
{
    FreeInOutArray();
}
```

以 C Block 取代子電路

在子電路屬性的對話視窗，一個叫 Replace subcircuit with generated code for simulation 的選擇框。如選取時，模擬完成時系統將以 C Block 取代子電路。使用者在系統中不需要經歷以 C Block 取代子電路的過程。

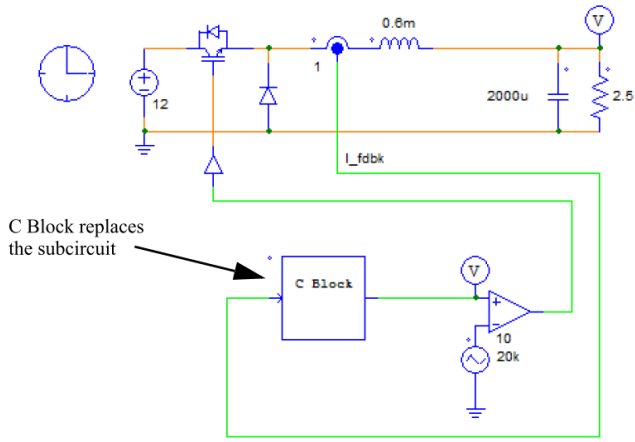
然而，如果使用者想隨意地修改 C 程式碼，可經由下列步驟將產生的程式碼放到 C Block。

在上面的例子，由子電路產生的程式碼包含四個部分：*SimulationStep*、*SimulationBegin*、*SimulationEnd* 以及餘下的部分。同樣地，C Block 也由這四部分組成，包含下列部分：

```
#include <Stdlib.h>
#include <String.h>
#include <math.h>
#include <Psim.h>

// PLACE GLOBAL VARIABLES OR USER FUNCTIONS HERE...
... ...

////////////////////////////////////////////////////////////////////////
// FUNCTION: SimulationStep
{
// ENTER YOUR CODE HERE...

}

////////////////////////////////////////////////////////////////////////
// FUNCTION: SimulationBegin
{
// ENTER INITIALIZATION CODE HERE...

}

////////////////////////////////////////////////////////////////////////
// FUNCTION: SimulationEnd
{

}
```

在主電路上建立一個 C Block，使用下拉式選單至 **Element>>Other>> Function Blocks>>C Block**，複製所產生程式碼的每一部分到 C Block，從 *SimulationStep* 到 C Block 的 *SimulationStep*，從 *SimulationBegin* 到 C Block 的 *SimulationBegin*，從 *SimulationEnd* 到 C Block 的 *SimulationEnd*，再將餘下的程式碼放到 *User Functions* 部份。

以 C Block 取代子電路的範例電路如下：



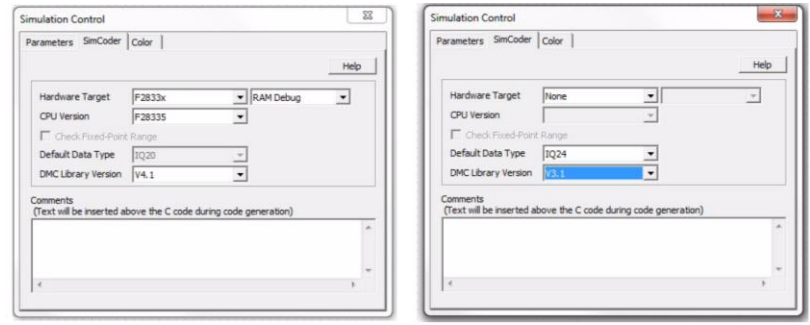
子電路為硬體標的產生程式碼

與 2.5.1 節同樣的範例電路也可用來為硬體電路產生程式碼。

在 Simulation Control 對話視窗的 SimCoder 表內設定所需的硬體標的類型，如下圖左，選定 *F2833x*，CPU 版本為 *F28335*，專案組態為 *RAM Debug*。

當硬體標的選擇 *None*，仍可針對特定的預設資料格式產生硬體標的程式碼。

下圖右的範例以 IQ24 的位置設定定點的資料格式。



開啟子電路屬性視窗，擊點 *Generate Code* 並選擇 Generate Code for Hardware Target。

SimCoder 用子電路針對 F28335 所產生的 C 程式碼如下，不同於針對整個系統所產生的程式碼，子電路產生的程式碼沒有主程式和初始化程序。它可插入使用者自己的 F28335 硬體標的程式碼。

子電路只有一個取樣頻率，正因如此，產生的程式碼只有一個 *TaskS1* 函數，變數 *fIn0* 表示為子電路輸入 *I_fdbk*，變數 *fOut0* 則對應子電路輸出 *Ctrl*。

```
#defineNULL (0)
#ifndef DSP28_DATA_TYPES
#define DSP28_DATA_TYPES
typedef int                int16;
typedef long               int32;
typedef long long          int64;
typedef unsigned int       Uint16;
typedef unsigned long      Uint32;
typedef unsigned long long Uint64;
typedef float              float32;
typedef long double        float64;
#endif
DefaultTypefGblS1_U1 = 0;
DefaultTypefGblS1_iref = 0;

// Parameter list for S1
void TaskS1(DefaultType fIn0, DefaultType *fOut0)
{
   DefaultTypefS1_U1, fS1_SUMP1, fS1_B4, fS1_k2, fS1_k1, fS1_SUM1, fS1_Z1;
   DefaultTypefS1_VDC2;

   *fOut0 = fGblS1_U1;

   fS1_VDC2 = 2;
   fS1_Z1 = fIn0;
   fS1_SUM1 = fS1_VDC2 - fS1_Z1;
   fS1_k1 = fS1_SUM1 * 0.4;
   fS1_k2 = fS1_SUM1 * 1000;
   {
            static DefaultType out_A = 0;
            fS1_B4 = out_A + 1.0/20000 * (fS1_k2);
            out_A = fS1_B4;
   }
   fS1_SUMP1 = fS1_k1 + fS1_B4;
   fGblS1_U1 = fS1_SUMP1;

#ifdef_DEBUG
    fGblS1_iref = fS1_VDC2;
#endif
}
```

子電路產生程式碼的限制

使用 SimCoder 子電路產生程式碼有幾個限制，條列如下：

- 所有在子系統的元件都需支援產生程式碼，要得知元件是否可用於產生程式碼，在 PSIM 中，至 **Options>>Settings**，選取選擇框 **Show image next to elements that can be used for code generation**。在 PSIM 元件庫中所有前端有標式符號的元件都可用來產生程式碼。
- 只可使用單方向的子電路端口，也就是說，輸入信號端口只可用於子電路輸入，輸出信號端口只可用於子電路輸出，雙向端口不可使用。
- 子系統中不可放置輸入/輸出元件(如 A/D 轉換器，數位輸入/輸出，編碼器，計數器和 PWM 產生器)和硬體中斷元件，這些都只能在最上層的主電路中。
- 如果子系統的輸入有一取樣率，且無法由子系統內部電路產生，則必須在輸入端連接一個 zero-order-hold 方塊以明確定義取樣率。如果這個範例不使用 zero-order-hold 方塊，輸入(以及之後連接到此輸入的方塊)將被視為沒有取樣率。
- 如果子系統的輸入沒有 zero-order-hold 方塊與之連接，SimCoder 將會在子系統中由連接到它的方塊獲得取樣率，然而，避免混淆，強烈建議在每一個輸入放置 zero-order-hold 方塊以明確定義取樣率。

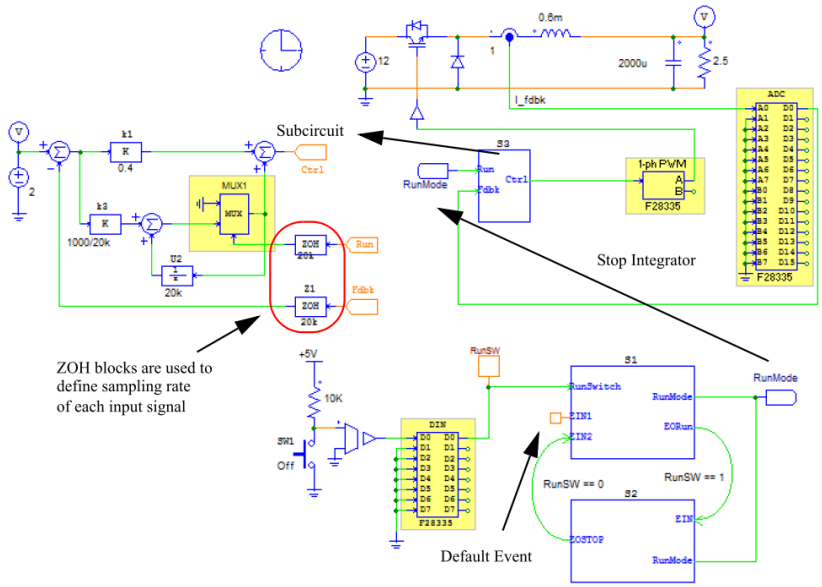
事件控制(Event Control)的系統

通常一個系統會包含事件轉換，當某些條件成立時，系統會從這個狀態轉換至另一個狀態，SimCoder 經由子電路處理事件控制，事件控制會在附錄 C 事件處理做更詳細地說明。

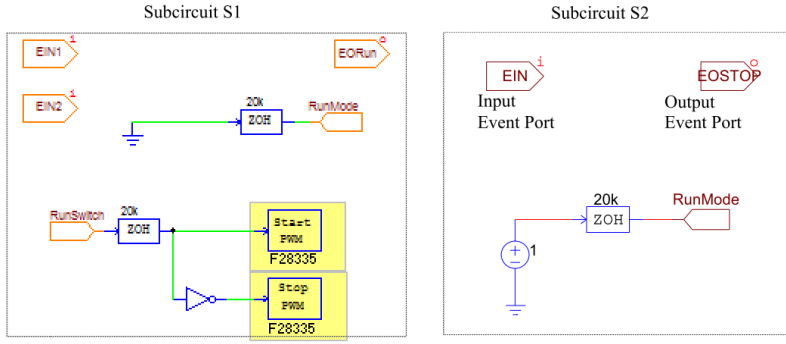
在這節的範例增加下列的情況以便說明事件控制如何執行：

- 系統的開始與停止由一個手動開關控制，因此，此系統會有兩個操作模式：停止模式與執行模式。當開關切換時，系統將由一個模式轉換到另一個模式。
- 在停止模式，為了避免積分器飽和，輸出將會重置為 0。事件控制系統範例如下：

事件控制系統範例如下：



如下圖表，S1 和 S2 為子電路，其內容如下：



相較於電路部分，改變如下：

- 增加兩個事件控制子電路以實現兩種操作模式：停止模式(子電路 S1)與執行模式(子電路 S2)。
- 預設的操作模式為停止模式。這是經由連接 Default Evemt 元件到子電路 S1 的端口 EN1 來定義。
- 子電路 S1 有兩個輸入事件端口為 EIN1 和 EIN2，一個輸出事件端口 EORun，一個輸入信號端口 RunSwitch，以及一個輸出信號端口 RunMode。子電路 S2 有一個輸入事件端口 EIN，一個輸出事件端口 EOSTOP，以及一個訊號端口 RunMode。
- 定義從停止模式轉換到執行模式的條件，反之亦然。變數 RunSW 在這種條件下為一全域變數(更詳盡請參照 60 頁)，且藉由全域變數元件連接到數位輸入元件的輸出腳位 D8 來定義。
- 硬體數位輸入元件用來量測按鈕開關 SW1 的位置，當開關截止，數位輸入電壓為高電位(1)，因此為全域變數 RunSW 與系統處於執行模式。當 RunSW 為低電位(0)，系統處於停止模式。

- 在停止模式下增加多工器 MUX1 以避免積分器積分，RunMode 信號為 0 且積分器不動作當系統不執行時。當 RunMode 信號為 1，積分器將開始動作。

系統如何動作如下：

- 手動開關的位置經由硬體數位輸入讀取，信號經由輸入信號端口 RunSwitch 傳送給子電路 S1(停止模式)，同樣的訊號也被指定為全域變數 RunSw。

- 系統初始在停止模式，當條件 RunSW==1(或 RunSW 為 1)成立，系統將從停止模式轉換成執行模式，由 S1 的輸出事件端口 EORun 與 S2 的輸入事件端口 EIN 的連接來定義。

- 當處於執行模式，如 RunSW==0(或 RunSw 為 0)的條件成立，系統將會從執行模式轉換成停止模式，由 S2 的輸出事件端口 EOSTOP 與 S1 的輸入事件端口 EIN2 的連接來定義。

- 在停止模式的子電路，RunMode 訊號會被設定為 0，只要 RunSwitch 的信號為 0，硬體 PWM 產生器將會停止，但當 RunSwitch 變為 1 時，PWM 開始動作，同時由停止模式切換到執行模式。

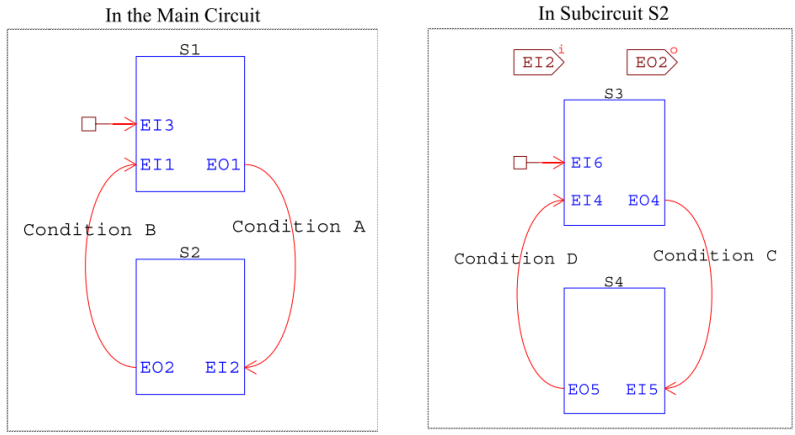
- 在執行模式的子電路，為了能使積分器動作，RunMode 信號將設定為 1。

在系統修正後，使用者可在針對硬體標的產生系統程式碼前藉由模擬來驗證變更是否正確。

附錄 C 事件處理

基本概念

事件用來描述系統由一個狀態轉變到另一個狀態。舉例來說，下圖顯示幾個操作狀態以及轉換如何發生。



在主電路有兩個狀態 S1 和 S2，都是子電路形式。每個狀態的原理圖都包含一個子電路。狀態 S1 有兩個輸入事件端口(port)*EI1* 和 *EI3*，一個輸出事件端口 *EO1*；狀態 S2 有一個輸入事件端口 *EI2*，一個輸出事件端口 *EO2*。一開始的預設狀態為 S1，其乃由預設事件元件連接到輸入事件端口 *EI3* 所定義。

藉由轉換條件 A，S1 的輸出事件端口 *EO1* 連接到 S2 的輸入事件端口 *EI2*，這意味著當條件 A 成立時，系統將從狀態 S1 轉換成狀態 S2。同樣地，S2 的輸出事件端口 *EO2* 連接到輸入事件端口 *EI1*，當條件 B 成立時，系統將由狀態 S2 轉換成 S1。

當兩個或多個狀態不能同時存在且在任何時間點只存在一個狀態，如這個例子的 S1 和 S2，稱這些狀態為專屬狀態。

右邊的系統顯示子電路 S2 內部，依序有兩個狀態 S3 和 S4，當系統轉換到 S2 時，預設由 S3 開始。當條件 C 成立時，將由 S3 轉換到 S4，當條件 D 成立時，又再回到狀態 S3。

一個系統包含的狀態數量沒有限制。

事件處理的元件

下列元件可用來定義事件和狀態轉換：

1. 輸入事件端口(Input event port)
2. 輸出事件端口(Output event port)
3. 初始事件元件(Default event element)
4. 方塊初次進入的旗標(Flag for block first entry)
5. 硬體中斷元件(Hardware interrupt element)
6. 全域變數(Global variable)

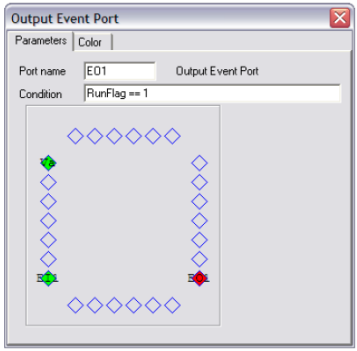
在子電路內放置輸入事件端口將創造一個允許轉入子電路的事件，同樣地，在子電路內放置輸出事件端口將創造一個允許轉出子電路的事件。

舉例來說，下圖為放置輸入事件端口和輸出事件端口後的子電路圖示。

事件端口的圖示為方形，不同於圓形的信號端口圖示。

連接到輸入事件端口的只能是輸出事件端口或硬體中斷元件並使用事件連接線功能。輸入/輸出事件端口和硬體中斷元件不可連接到其他形式的接點上。

必須定義每個輸出事件端口的條件。子電路 S3 的輸出事件端口 *EO1* 屬性視窗如下



"RunFlag==1"為觸發輸出事件發生的條件，條件敘述必須為有效的 C 程式碼表示式。舉例來說，條件敘述可以如下：

(RunFlag==1)&&(Flag>=250.) || (FlagB<Vconst)

注意	只有在參數文件中定義或通過主電路進入子電路的全域變數、數值和參數常數可使用在條件表示式。在上面的表示式，RunFlag、FlagA 和 FlagB 為全域變數，Vconst 為參數文件中定義或通過主電路進入子電路的常數。

連接全域變數元件到一個節點可建立一個全域變數。

具有事件的子電路之限制

包括輸入或輸出事件端口的子電路會被視為具有事件的子電路，具有事件子電路的內部所有元件將會繼承事件屬性。也就是說，如果子電路 A 在子電路 B 內，子電路 B 是具有事件的子電路，即使子電路 A 沒有任何輸入/輸出端口，仍被視為具有事件的子電路。

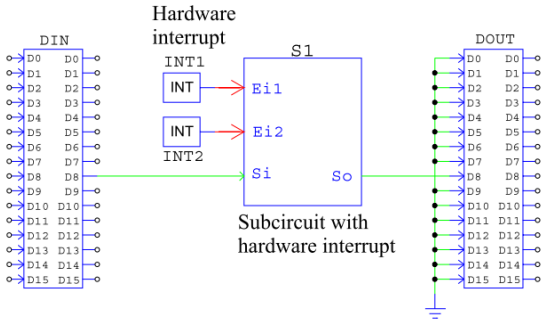
當用子電路來處理事件，在 PSIM 中有三種類型的子電路：

- 正規子電路(*Regular subcircuits*)：這種類型的子電路不包含任何事件端口且與先前常見的子電路相同。
- 具有事件的子電路(*Subcircuit with events*)：這種類型的子電路包含輸入/輸出事件端口，但輸入事件端口並沒有連接硬體中斷元件。
- 具有硬體中斷的子電路(*Subcircuit with hardware interrupt*)：這種類型的子電路只包含輸入事件端口並只連接到硬體中斷元件，子電路內無輸出事件端口，且輸入事件端口沒有連接到輸出事件端口。這是具有事件的子電路中的特例，這種類型的子電路指專門處理硬體中斷。

因為具有事件或硬體中斷的子電路只包含在產生程式碼中，針對這兩種類型的子電路，更多的限制如下：

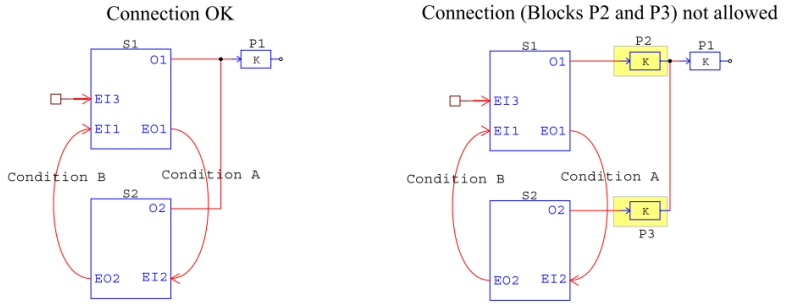
- 具有事件或硬體中斷的子電路內的所有元件必須支援產生程式碼。例如，這樣的子電路不能包括電阻和 RMS 方塊，因為這兩個都不支援產生程式碼。
- 具有硬體中斷的子電路可有多個輸入事件端口，但不能有輸出事件端口。也只有硬體中斷元件可以連接到輸入事件端口。此外，子電路的信號輸出/輸入端口只可連接到硬體元件，不可連到其他函數方塊。

下圖顯示一個具有硬體中斷的子電路如何連接：



具有硬體中斷的子電路 S1 有兩個硬體中斷元件 INT1 和 INT2 與之連接，一個信號輸入端口 Si 連接到硬體數位輸入，一個信號輸出端口 So 連接到硬體數位輸出。

- 如果具有硬體中斷的子電路包含有取樣率的 Z 平面方塊，當硬體中斷發生時，子電路被呼叫取樣率將被忽略。舉例來說，如果一個包含數位積分器的子電路，數位積分器的取樣率將會被忽略。在積分器計算時，中斷前的值將被保留成硬體中斷後的值。
- 如果有兩個子電路的輸出信號相連接，則必需直接連接，不可經由其他元件連接，參考下圖說明：



在左邊的電路，子電路 S1 和 S2 都有一個信號輸出端口 O1 和 O2，它們一起連接到比例方塊 P1 的輸入，這種電路動作的方式為方塊 P1 的輸入來自 O1 或 O2 端口，取決於哪個狀態動作，這種連接方式是允許的。

然而，在右邊的電路，O1 連接到 P2，O2 連接到 P3，P2 和 P3 的輸出連接在一起，這樣的連接是不被允許的。在此例子中，方塊 P2 應該移到子電路 S1 內，方塊 P2 則應該移動到子電路 S2 內。

附錄 D SimCoder 資料庫

SimCoder 有無硬體標的皆可使用，當沒有硬體標的時，將控制原理圖轉成 C 程式碼，此程式碼在 PSIM 中模擬時，並不針對特定硬體，另一方面，當有硬體標的時，SimCoder 會產生已準備好在特定硬體標的上執行或被特定硬體所採用的程式碼。

SimCoder 元件資料庫包含兩種類型的元件：一種為獨立於任何硬體標的或所有硬體標的皆可共用，另一種為針對特定的某個硬體標使用。

獨立於任何硬體標的之 SimCoder 元件包含如下：

- 某些 PSIM 標準資料庫的元件。
- 在 Elements>>Event Control 下的所有元件。
- 在 Elements>>SimCoder 下的全域變數(*Global Variable*)元件。

所有硬體標的皆可共用的 SimCoder 元件包含如下：

- 在 Elements>>SimCoder 下的中斷(*Interrupt*)元件。
- 在 Elements>>SimCoder>>TI DMC Library 下的 *TI DMC* 元件。

針對特定硬體使用的 SimCoder 元件包含如下：

- F2833x Target：在 Elements>>SimCoder>>F2833x Target 下的所有元件。
- F2803x Target：在 Elements>>SimCoder>>F2803x Target 下的所有元件。
- PE-Pro/F2833x Target：在 Elements>>SimCoder>> PE-Pro/F2833x Target 下的所有元件。
- PE-Expert3 Target：在 Elements>>SimCoder>> PE-Expert3 Target 下的所有元件。

所有獨立於硬體標的或硬體標的共用的 SimCoder 元件將在本章節說明，每個特定硬體標的之元件將在 5~8 章說明，TI DMC 資料庫則在第附錄 D 說明。

標準 PSIM 資料庫的元件

許多在 PSIM 標準資料庫的元件都可用來產生程式碼，在 **Options>> Settings>> Advanced**，如果選項框*"show image next to elements that can be used for code generation"*被選取，一個小圖案會出現在這些可用於產生程式碼的元件前。

⚠注意	所有的數學函數方塊和簡易 C 方塊，變數 t(針對時間)，delta(針對時間步長)不能用於 SimCoder 產生程式碼。

此外，參數文件元件和鋸齒電壓源元件在 SimCoder 中有特別的用法，將在這章節中說明。

在參數文件(Parameter File)中定義全域(Global)參數

參數文件元件可如同先前的用法，在 SimCoder，可用來定義全域參數。

為了使產生的程式碼更具可讀性與易於管理，有時候最好是在程式碼中以參數名稱取代實際數值。例如，如果控制器的增益是 1.23，定義參數 Kp=1.23，在程式碼中以參數 Kp 取代。

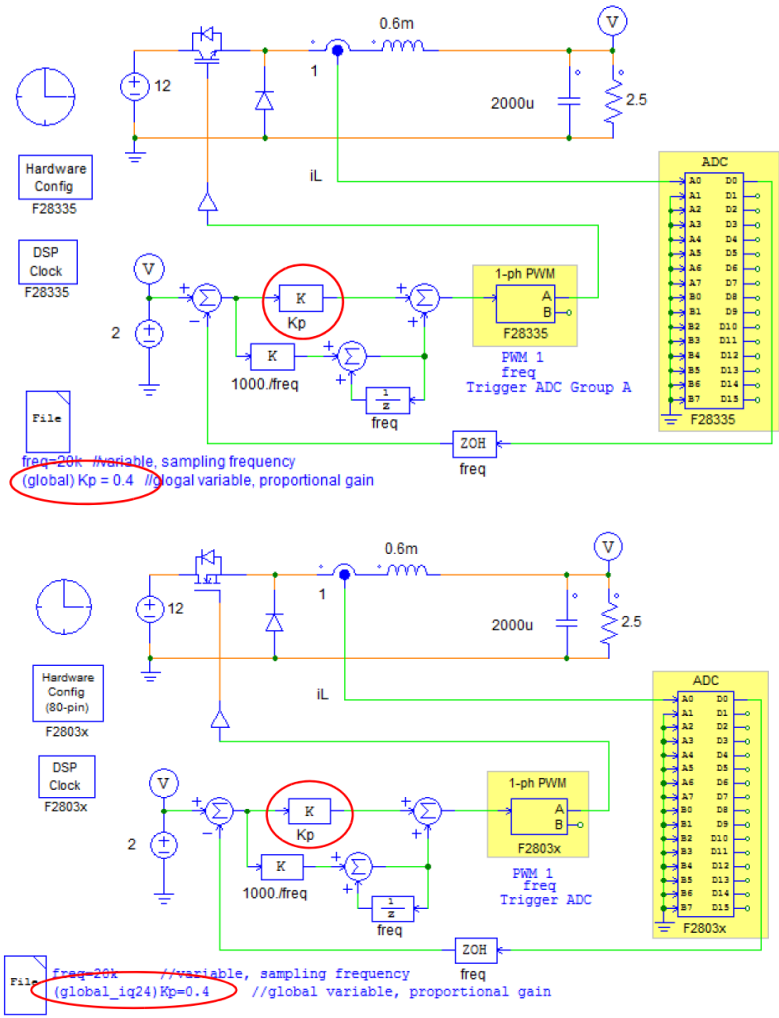
這種參數類型為全域的，可在程式碼中的任何地方使用。定義一個參數為全域參數，在參數文件內容中，參數名稱前使用"(global)"定義。

如下所示為一降壓式轉換器電路，比例控制器的增益定義為 Kp，在參數文件中，如果 DSP 標的為浮點 F28335，參數 Kp 定義為：

(global)Kp=0.4

如果 DSP 標的為在 IQ-24 的定點 F28035，參數 Kp 定義為：

(global_Iq24)Kp=0.4



注意

在程式碼中，參數 Kp 的初始值定義為 0.4，參數名稱 Kp 則用在計算中。

針對 F28335 產生的程式碼如下

```
/*********************************************************************************
// This code is created by SimCoder Version 9.1 for TI F28335 Hardware Target
//
// SimCoder is copyright by Powersim Inc., 2009-2011
//
// Date: November 08, 2011 11:26:37
*********************************************************************************/
#include      <math.h>
#include      "PS_bios.h"
typedef float DefaultType;
#define       GetCurTime() PS_GetSysTimer()

interrupt void Task();

DefaultType fGbliref = 0.0;
DefaultType fGblUDELAY1 = 0;

DefaultType Kp = 0.4;        The global parameter Kp is defined here.
interrupt void Task()
{
            DefaultType fVDC2, fTI_ADC1, fZOH3, fSUM1, fP2, fSUMP3, fUDELAY1, fP1, fSUMP1;
            PS_EnableIntr();
            fUDELAY1 = fGblUDELAY1;


            fTI_ADC1 = PS_GetDcAdc(0);
            fVDC2 = 2;
#ifdef      _DEBUG
            fGbliref = fVDC2;
#endif
            fZOH3 = fTI_ADC1;
            fSUM1 = fVDC2 - fZOH3;
            fP2 = fSUM1 * (1000./20000);
            fSUMP3 = fP2 + fUDELAY1;
            fGblUDELAY1 = fSUMP3;
            fP1 = fSUM1 * Kp;        The parameter Kp is used here.
            fSUMP1 = fP1 + fSUMP3;
            PS_SetPwm1RateSH(fSUMP1);
            PS_ExitPwm1General();
}
```

產生鋸齒波

鋸齒波在硬體使用中可當做系統的計時器，或產生其他的週期波(如正弦波)，在 **Elements>>Sources>>Voltage** 下的鋸齒電壓源可在硬體中以實際的計數器實現。

假設在硬體中存在一個 32 位元的計數器，每 20ns 增加一次以產生鋸齒波。對於 PE-Expert3 硬體標的，在 PEV 板使用 32-bit 的非同步計數器，每 20ns 增加一次以產生鋸齒波。

事件控制元件

下列元件用來實現事件控制：

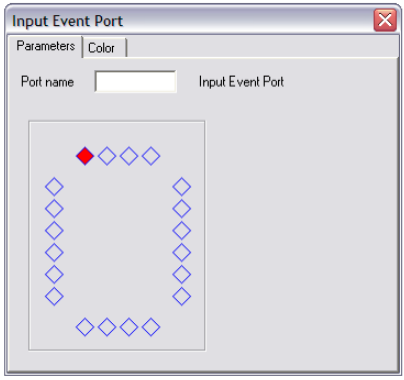
輸入事件

輸入事件元件圖示如下：

圖示 Input Event

圖示中的字母“ i ”代表“input”。

輸入事件元件為子電路界面端口的一種類型，只用在子電路內，連續擊點元件兩下後，即可定義端口的名稱及位置，如下：



在主電路中呼叫子電路，如果有事件連接線連接到這個端口，當事件連接的條件成立，系統將會經由輸入事件端口轉換到子電路。

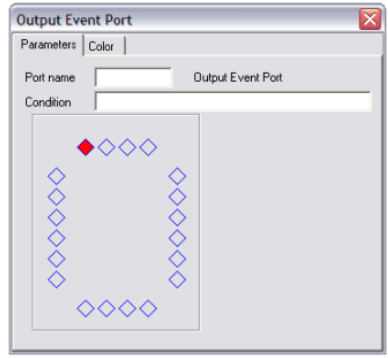
輸出事件

輸出事件元件圖示如下：

圖示 Output Event

圖示中的字母“ o ”代表“output”。

輸出事件元件亦為子電路界面端口的一種類型，只用在子電路內，連續擊點元件兩下後，即可定義端口的名稱及位置，如下：



當輸出事件端口定義的條件成立，系統將由這個子電路轉出至另一子電路。

條件敘述必須使用 C 語言所支援的格式及運算。例如，下面的敘述為可接受的條件敘述：

A==1

B>=1

(A>B)&&(C>D)

其中 A、B、C 和 D 為全域變數或常數。

初始事件

初始事件元件圖示如下：

圖示：

圖示	Default Event

當有多個專屬狀態時，初始事件元件用來定義哪個狀態為初始狀態，連接到這個子電路以外的子電路之輸入事件端口。

事件連結

事件連結元件為 SimCoder 中連接輸出事件端口或硬體中斷元件到輸入事件端口的元件。注意不要與一般用來連接其他 PSIM 元件的佈線工具混淆，事件連接只可用來連接事件。事件連結元件位於 **Elements>>Event Control>>Event Connection**。

在事件連接線連續擊點兩下，可編輯事件線所連接到的輸出事件端口之條件敘述。

除了起點和終點，事件連接線在此之間還有兩個點。藉由修正這兩點的位置來改變連接線的形狀，修改這兩點來以\凸顯事件接線。擊點右鍵並選擇"Modify handle 1"或"Modify handle 2"。

事件方塊首次進入的標誌

有時候當程式第一次進入一個事件子電路方塊時，需要執行某些操作。提供事件首次進入的標誌來辨別。

圖示

屬性

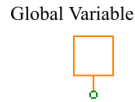
參數	描述
Event Subcircuit Block Name	標誌所給的事件子電路方塊名稱

標誌節點為一個輸出節點。當事件子電路方塊第一次進入時，節點值為 1，反之，為 0。舉例來說，當發現事件子電路方塊 S1 第一次進入時，設定 Event Subcircuit Block Name 為 S1。

全域變數

全域變數使用在條件敘述或特殊場合。

圖示



屬性

參數	描述
Name	全域變數名稱
Initial Value	全域變數初始值

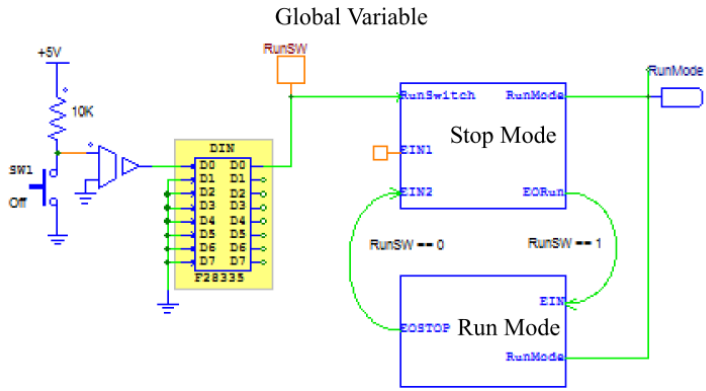
為了定義一個訊號為全域變數，連接全域變數元件到特定節點。注意只有用來產生程式碼的控制電路的信號可定義為全域變數。

正如其名，全域變數可全範圍使用。當全域變數的初始值改變，在電路中，包含子電路的所有全域變數的初始值在同時都會改變。

全域變數可為信號接收或信號輸出，當為信號接收端時，將從節點接收信號值，當為信號輸出端時，會將值放在節點。

一旦在事件條件敘述中使用全域變數，所有在條件敘述中的變數須皆為全域變數。

舉例如下：



在此範例中，全域變數 *RunSW* 連接到數位輸入的輸出腳位 D0。全域變數使用在這兩個操作模式間轉換的條件敘述。

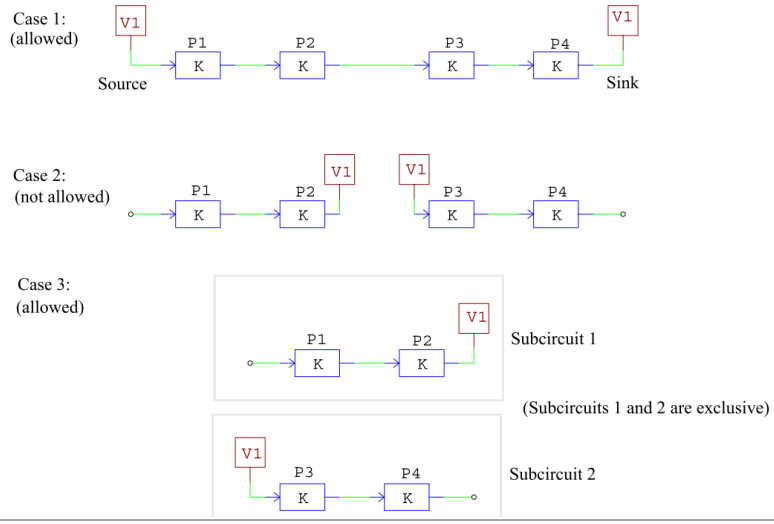
全域變數的另一用法為信號輸出，例如，全域變數可用來當作信號輸出與傳值至另一方塊。

注意

全域變數不應用來當作從一節點傳值到另一節點的標籤。當兩個節點可經由線實際連接，全域變數的使用有下列限制：

- 只要在同樣的傳輸路徑相同名稱的全域變數可多次使用。
- 如果不一樣的傳輸路徑，則不允許同樣名稱的全域變數，除非在不同的專屬狀態(不能同時發生的狀態為專屬狀態)。

為了說明這點，下面流程圖說明全域變數使用的場合。



在 Case 1，全域變數 V1 先為輸出源，並賦予方塊 P1 的輸入一個值。經過一連串計算，方塊 P4 的輸出也被賦予相同的全域變數 V1。因為兩個全域變數都在同一個傳輸路徑上，所以是可行的。

在 Case 2，全域變數 V1 用來當作從方塊 P2 的輸出傳值到方塊 P3 的輸入之標籤，這是不允許的。從一個節點傳值到另一個節點，應該用標籤取代或用線連接兩個節點。

在 Case 3，全域變數 V1 在子電路 S1 和 S2 都有使用，然而，子電路 S1 和 S2 為兩個專屬狀態，也就是說，系統將會執行子電路 1 或 2，並不會同時執行。在這個案例中全域變數的使用是允許的。

中斷

在硬體標的中，如數位輸入、編碼器、捕捉和 PWM 產生器(用於 F2833x 和 F2803x DSP)等元件可產生硬體中斷，中斷方塊允許使用者用對應的子電路聯繫產生中斷的元件來表示中斷服務程序。

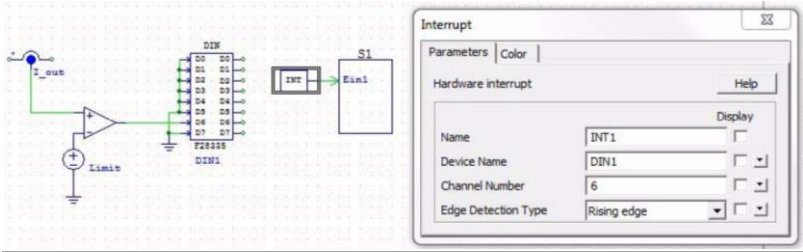
⚠注意	中斷元件不能置於子電路內，它只能位於在最上層的主電路。

圖示

屬性

參數	描述
Device Name	啟動硬體中斷的硬體設備名稱
Channel Number	啟動中斷的設備之輸入通道數 例如，數位輸入的通道 D0 產生中斷，通道數應設為 0。 注意此參數只使用在： 1.數位輸入 2.捕捉(只在 PE-Expert3 標的) 並不適用於解碼器與 PWM 產生器。
Trigger Type	只用於數位輸入與捕捉，可為下列其中之一： No edge detection：不會發生中斷。 Rising edge：輸入信號的上升緣產生中斷。 Falling edge：輸入信號的下降緣產生中斷。 Rising/Falling edges：輸入信號的上升與下降緣都會產生中斷。

下圖顯示如何使用中斷：



在這個電路中，量測電流 I_out 並與參考值 Limit 比較，如果電流 I_out 超過 Limit，比較器的輸出將由 0 變為 1。數位輸入方塊 DIN1 的通道 D6 將接收到一上升邊緣，中斷方塊參數設定如圖所示：

1. Device Name：*DIN1* 針對特定的數位輸入方塊
2. Channel Number：*6* 針對特定的數位輸入通道 *D6*
3. Edge Detection Type：針對 *I_out* > *Limit* 條件下的上升邊緣

比較器輸出的上升邊緣將會產生硬體中斷且此操作將會經由輸入事件端口 *EI1* 轉換到子電路 *S1*。

注意

中斷方塊 INT1 和事件子電路 S1 間的連接為事件連接，並非一條線的連結。

附錄 E F2833x 硬體標的

有了 F2833x 硬體標的，SimCoder 可產生已準備好在基於 TI F2833x 浮點 DSP 的任何硬體板上執行的程式碼。

F2833x 硬體標的可在所有的 F2833x 封裝上執行，後兩頁的圖為 low-profile flatpack(LQFP)封裝的 F2833x DSP 之腳位定義。圖中實現 F2833x 硬體標的之主要函數以不同顏色來標注。

F2833x 硬體標的資料庫包含下列函數方塊：

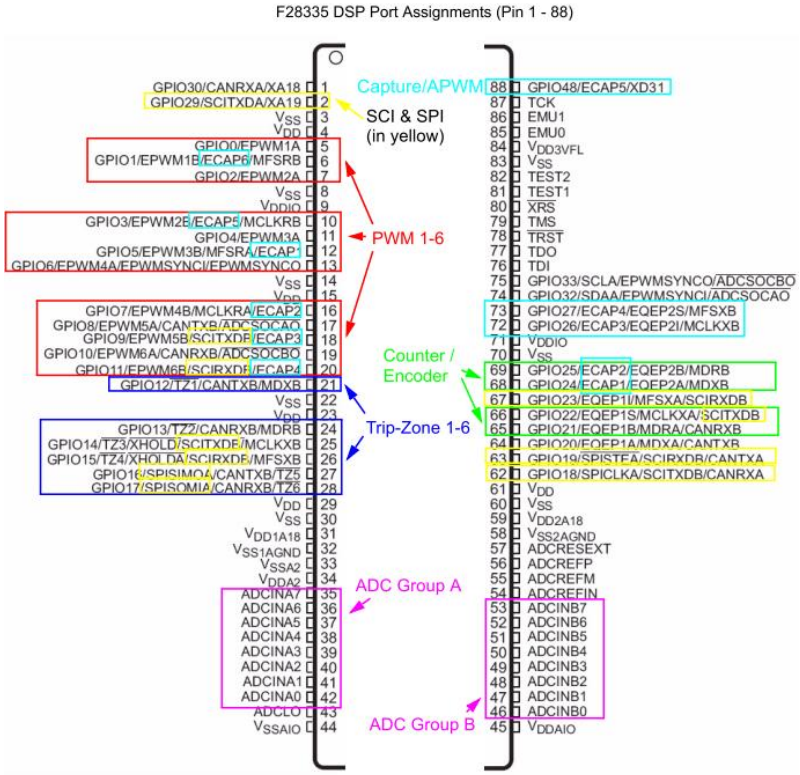
- PWM 產生器(PWM generators)：三相(3-phase)、雙相(2-phase)、單相(1-phase)和 APWM
- PWM 產生器的開始/停止函數(Start/Stop functions for PWM generators)
- Trip-zone 與其狀態(Trip-zone and trip-zone state)
- 類比/數位轉換器(A/D Converter, ADC)
- 數位輸入與輸出(Digital input and output)
- SCI 組態、輸入與輸出(SCI configuration, input, output)
- SPI 組態、設備、輸入與輸出(SPI configuration, device, input, output)
- 捕捉與其狀態(Capture and capture state)
- 編碼器與其狀態(Encoder and encoder state)
- 上/下計數器(Up/Down counter)
- DSP 時脈(DSP clock)
- 硬體組態(Hardware configuration)

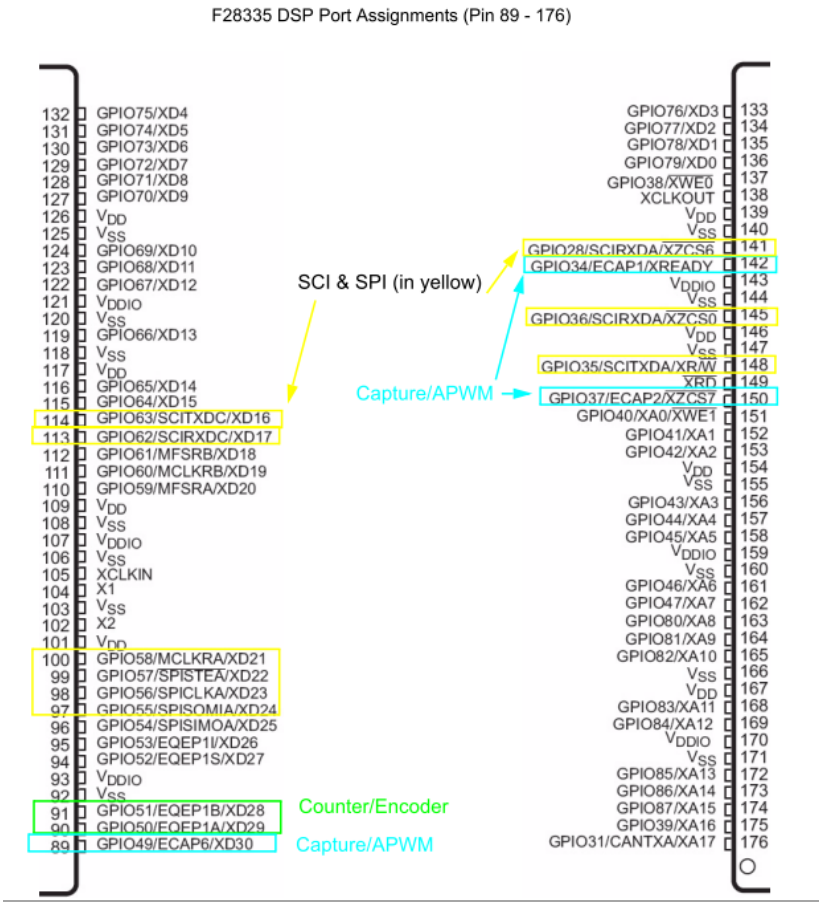
針對有多個取樣率的系統產生程式碼，SimCoder 使用 PWM 產生器的中斷為 PWM 取樣率。對於控制系統其他的取樣率，將先使用計數器 1 中斷，如需要再使用計數器 2 中斷，如果控制系統超過三個取樣率，相對應的中斷程序會在主程式中藉由軟體處理。

在 TI F2833x，PWM 產生器可以產生硬體中斷，SimCoder 將尋找與集結所有連接到 PWM 產生器並與其有相同取樣率的元件。這些元件將自動放置在所產生的程式碼的中斷服務程序中並執行。

此外，數位輸入、解碼器、捕捉和觸發區也可產生硬體中斷。每個硬體中斷必須有相對的中斷方塊(在 5.4 節中說明)，而每個中斷方塊則必須有相對應的中斷服務程序(表示中斷服務程序的子電路)。例如，PWM 產生器與數位輸入都產生中斷，它們各自都要有一個中斷方塊與中斷服務程序。

F2833x 硬體標的資料庫內元件定義將在此章說明。



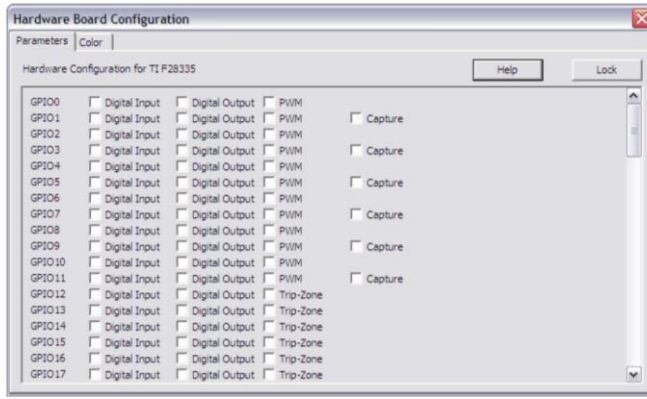

F28335 DSP Port Assignments (Pin 89 - 176)
132 GPIO75/XD4
131 GPIO74/XD5
130 GPIO73/XD6
129 GPIO72/XD7
128 GPIO71/XD8
127 GPIO70/XD9
126 VDD
125 VSS
124 GPIO69/XD10
123 GPIO68/XD11
122 GPIO67/XD12
121 VDDIO
120 VSS
119 GPIO66/XD13
118 VSS
117 VDD
116 GPIO65/XD14
115 GPIO64/XD15
114 GPIO63/SCITXDC/XD16
113 GPIO62/SCIRXDC/XD17
112 GPIO61/MFSRB/XD18
111 GPIO60/MCLKRB/XD19
110 GPIO59/MFSRA/XD20
109 VDD
108 VSS
107 VDDIO
106 VSS
105 XCLKIN
104 X1
103 VSS
102 X2
101 VDD
100 GPIO58/MCLKRA/XD21
99 GPIO57/SPISTEA/XD22
98 GPIO56/SPICLKA/XD23
97 GPIO55/SPISOMIA/XD24
96 GPIO54/SPISIMOA/XD25
95 GPIO53/EQEP1I/XD26
94 GPIO52/EQEP1S/XD27
93 VDDIO
92 VSS
91 GPIO51/EQEP1B/XD28
90 GPIO50/EQEP1A/XD29
89 GPIO49/ECAP6/XD30
SCI & SPI (in yellow)
Capture/APWM
Counter/Encoder
Capture/APWM
GPIO76/XD3 133
GPIO77/XD2 134
GPIO78/XD1 135
GPIO79/XD0 136
GPIO38/XWE0 137
XCLKOUT 138
VDD 139
VSS 140
GPIO28/SCIRXDA/XZCS6 141
GPIO34/ECAP1/XREADY 142
VDDIO 143
VSS 144
GPIO36/SCIRXDA/XZCS0 145
VDD 146
VSS 147
GPIO35/SCITXDA/XR/W 148
XRD 149
GPIO37/ECAP2/XZCS7 150
GPIO40/XA0/XWE1 151
GPIO41/XA1 152
GPIO42/XA2 153
VDD 154
VSS 155
GPIO43/XA3 156
GPIO44/XA4 157
GPIO45/XA5 158
VDDIO 159
VSS 160
GPIO46/XA6 161
GPIO47/XA7 162
GPIO80/XA8 163
GPIO81/XA9 164
GPIO82/XA10 165
VSS 166
VDD 167
GPIO83/XA11 168
GPIO84/XA12 169
VDDIO 170
VSS 171
GPIO85/XA13 172
GPIO86/XA14 173
GPIO87/XA15 174
GPIO39/XA16 175
GPIO31/CANTXA/XA17 176

硬體配置

F2833x DSP 提供 88 個通用輸出入(general-purpose-input-output, GPIO)端口(GPIO0 到 GPIO87)，每個端口可能配置不同的功能。然而，對於一個特定的 DSP 板，並非每個端口皆可由外部設定且通常某些端口的功能是固定的。硬體配置方塊提供針對特定 DSP 板配置 SimCoder 的方式。

圖示

對話視窗如下圖所示：



對於每個 GPIO 提供選取可用函數的複選框，如選擇某一函數，就只能使用該函數，其餘的則無法使用。例如，GPIO1 可當數位輸入、數位輸出、PWM 和捕捉使用，一旦選用 GPIO1 來當 PWM 輸出，只能勾選 PWM 的複選框而其他的則須空白。如果在電路中 GPIO1 用來當數位輸入，則 SimCoder 會出現錯誤。

DSP 時脈

DSP 時脈方塊定義 F2833x DSP 的速度與外部時脈頻率以及程式空間大小。

圖示

屬性

參數	描述
External Clock(MHz)	DSP 板外部時脈的頻率，單位 MHz。此頻率必須為整數，且最大允許頻率為 30MHz。
DSP Speed(MHz)	DSP 速度，單位 MHz。需為整數且為外部時脈的整數倍，從 1 到 12 倍。最大允許速度為 150MHz。

如電路中不使用 DSP 時脈方塊，則使用 DSP 方塊的預設值。

PWM 產生器

F2833x DSP 提供六組 PWM 輸出：PWM 1(GPIO0 和 GPIO1), PWM 2(GPIO2 和 GPIO3), PWM 3(GPIO4 和 GPIO5), PWM 4(GPIO6 和 GPIO7), PWM 5(GPIO8 和 GPIO9)和 PWM 6(GPIO10 和 GPIO11)。每組有兩個彼此互補的輸出。舉例來說，除了當 PWM 操作在特殊模式外，PWM 1 有一個正輸出 PWM 1A 和一個負輸出 PWM 1B。

在 SimCoder 中，六個 PWM 可如下列方式使用:

- 兩個三相 PWM 產生器：PWM 123(包含 PWM 1, 2 和 3)和 PWM 456(包含 PWM 4, 5 和 6)。
- 六個雙相 PWM 產生器：在特殊操作模式下有兩個不互補輸出的 PWM 產生器 PWM 1, 2, 3, 4, 5 和 6。
- 單相 PWM 產生器：有著兩個互補輸出的 PWM 1, 2, 3, 4, 5 和 6。
- 相位偏移單相 PWM 產生器：有著兩個互補輸出的 PWM 2, 3, 4, 5 和 6。

這些 PWM 產生器可觸發 A/D 轉換器並使用 trip-zone 信號。

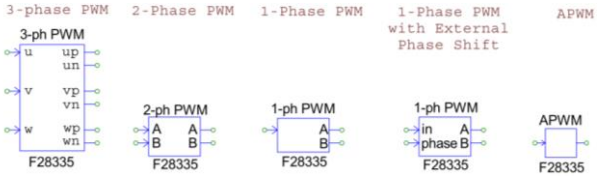
除了上述的 PWM 產生器，還有六個與捕捉使用同樣來源的 APWM 產生器。與六個 PWM 產生器(PWM1~6)相比，這些 PWM 產生器有不能觸發 A/D 轉換器與不能使用 trip-zone 信號的功能限制。因為共用來源，故端口做為捕捉使用時，就不能用來當 PWM 產生器。

注意

SimCoder 中所有 PWM 產生器內部都包含一個切換週期的延遲，也就是說 PWM 產生器的輸入值在用來更新 PWM 輸出前會先延期一個週期。此延遲是模擬 DSP 硬體實現時的內部延遲，故其是必要的。

PWM 產生器的圖示與參數如下：

圖示



三相 PWM 產生器

三相 PWM 產生器的圖示中，“ u ”,“ v ”和“ w ”為三個相位(或稱為相位“ a ”,“ b ”和“ c ”)。字母“ p ”代表正輸出，“ n”代表負輸出。例如，三相 PWM 123，“ up ”代表 PWM 1A，“ un ”代表 PWM 1B。

屬性

參數	描述
PWM Source	PWM 產生器的來源。 可為使用 PWM1~3 的"3-phase PWM 123" 或使用 PWM4~6 的"3-phase PWM 456"。
Dead Time	PWM 產生器的空白時間 Td，單位為秒。
Sampling Frequency	PWM 產生器的取樣頻率，單位 Hz。基於此頻率完成計算且更新 PWM 信號責任週期。
PWM Freq. Scaling Factor	PWM 頻率與取樣頻率間的比例因數，可為 1、2 或 3。亦即，PWM 頻率(用來控制開關切換的 PWM 輸出信號頻率)可為取樣頻率的倍數。例如，取樣頻率為 50kHz 且比例因數為 2，意謂著 PWM 頻率為 100kHz。開關以 100kHz 切換，但閘極信號每兩個切換週期 50kHz 更新一次。

Carrier Wave Type	載波類型和初始 PWM 輸出狀態，為下列其一： Triangular(start low)：三角波，初始 PWM 輸出狀態為低準位。 Triangular(start high)：三角波，初始 PWM 輸出狀態為高準位。 Sawtooth(start low)：鋸齒波，初始 PWM 輸出狀態為低準位。 Sawtooth(start high)：鋸齒波，初始 PWM 輸出狀態為高準位。
Trigger ADC	設定 PWM 產生器是否觸發 A/D 轉換器。為下列其一： Do not trigger ADC：PWM 不會觸發 A/D 轉換器。 Trigger ADC Group A：PWM 將觸發 A/D 轉換器的 A 群組。 Trigger ADC Group B：PWM 將觸發 A/D 轉換器的 B 群組。 Trigger ADC Group A & B：PWM 將觸發 A/D 轉換器的 A 與 B 群組。
ADC Trigger Position	A/D 觸發位置的範圍從 0 到一個小於 1 的值。當為 0 時，A/D 轉換器從 PWM 週期開始時被觸發。當為 0.5 時，A/D 轉換器在 PWM 週期為 180°時被觸發。
Use Trip-Zone i	定義 PWM 產生器使用第 i 個 trip-zone 信號與否，i 的範圍從 1 到 6，為下列其一： Disable Trip-Zone i：停用第 i 個 trip-zone 信號。 One shot：PWM 產生器在單次觸發模式下使用 trip-zone 信號。一旦觸發，PWM 須手動啟動。 Cycle by cycle：PWM 產生器在週期循環的基礎下使用 trip-zone 信號。trip-zone 信號在當週內有效，PWM 將在下一個週期自動重新啟動。

Trip Action	定義 PWM 產生器如何回應跳閘動作。為下列其一： High impedence：PWM 輸出為高阻抗。 PWM A high & B low：PWM 正輸出為高準位，負輸出為低準位。 PWM A low & B high：PWM 正輸出為低準位，負輸出為高準位。 No action：不採取任何動作。
Peak-to-peak Value	載波的峰對峰值 Vpp。
Offset Value	載波的直流偏移量 Voffset。
Initial Input Value u, v, w	三相輸入 u、v 和 w 的初始值。
Start PWM at Beginning	當設定為"Start"，開始時 PWM 立即啟動。如果設定為"Do not start"，則需使用"Start PWM"函數來啟動 PWM。

單相 PWM 產生器

帶有內部或外部相位偏移的單相 PWM 產生器具有相同的屬性，不同的是外部相位偏移的單相 PWM 有一個相位偏移的外部輸入(標記為“ phase ”)。相位的單位與沒有外部相位偏移的單相 PWM 相同都為角度。

屬性

參數	描述
PWM Source	PWM 產生器的來源。 沒有相位偏移，可為 PWM 1~PWM 6。 有相位偏移，可為 PWM 2~PWM 6。

Output Mode	PWM 產生器的輸出模式，可為下列其一： Use PWM A&B：使用 PWM 輸出 A 和 B，為互補。 Use PWM A：只使用 PWM 輸出 A。 Use PWM B：只使用 PWM 輸出 B。
Dead Time	PWM 產生器的空白時間 Td，單位為秒。
Sampling Frequency	PWM 產生器的取樣頻率，單位 Hz。基於此頻率完成計算且更新 PWM 信號責任週期。
PWM Freq. Scaling Factor	PWM 頻率與取樣頻率間的比例因數，可為 1、2 或 3。亦即，PWM 頻率(用來控制開關切換的 PWM 輸出信號頻率)可為取樣頻率的倍數。例如，取樣頻率為 50kHz 且比例因數為 2，意謂著 PWM 頻率為 100kHz。開關以 100kHz 切換，但閘極信號每兩個切換週期 50kHz 更新一次。
Carrier Wave Type	載波類型和初始 PWM 輸出狀態，為下列其一： Triangular(start low)：三角波，初始 PWM 輸出狀態為低準位。 Triangular(start high)：三角波，初始 PWM 輸出狀態為高準位。 Sawtooth(start low)：鋸齒波，初始 PWM 輸出狀態為低準位。 Sawtooth(start high)：鋸齒波，初始 PWM 輸出狀態為高準位。
Trigger ADC	設定 PWM 產生器是否觸發 A/D 轉換器。為下列其一： Do not trigger ADC：PWM 不會觸發 A/D 轉換器。 Trigger ADC Group A：PWM 將觸發 A/D 轉換器的 A 群組。 Trigger ADC Group B：PWM 將觸發 A/D 轉換器的 B 群組。 Trigger ADC Group A&B：PWM 將觸發 A/D 轉換器的 A 與 B 群組。

ADC Trigger Position	A/D 觸發位置的範圍從 0 到一個小於 1 的值。當為 0 時，A/D 轉換器在 PWM 週期開始時被觸發。當為 0.5 時，A/D 轉換器在 PWM 週期為 180°時被觸發。
Use Trip-Zone i	定義 PWM 產生器使用第 i 個 trip-zone 信號與否，i 的範圍從 1 到 6，為下列其一： Disable Trip-Zone i：停用第 i 個 trip-zone 信號。 One shot：PWM 產生器在單次觸發模式下使用觸 trip-zone 信號。一旦觸發，PWM 須手動啟動。 Cycle by cycle：PWM 產生器在週期循環的基礎下使用 trip-zone 信號。trip-zone 信號在當週內有效，PWM 將在下一個週期自動重新啟動。
Trip Action	定義 PWM 產生器如何回應跳閘動作。可為下列其一： High impedence：PWM 輸出為高阻抗。 PWM A high & B low：PWM 正輸出為高準位，負輸出為低準位。 PWM A low & B high：PWM 正輸出為低準位，負輸出為高準位。 No action：不採取任何動作。
Peak-to-peak Value	載波的峰對峰值 Vpp。
Offset Value	載波的直流偏移量 Voffset。
Phase Shift	相對於 PWM 產生器參考輸出的輸出相位偏移，單位度(°)。單相相移 PWM 針對此特性會多個輸入。
Initial Input Value	輸入的初始值。
Start PWM at Beginning	當設定為"Start"，開始時 PWM 立即啟動。如果設定為"Do not start"，則需使用"Start PWM"函數來啟動 PWM。

相位偏移

單相 PWM 產生器可以產生一個相對另一 PWM 信號有相位偏移的 PWM 信號。兩個 PWM 系列：PWM 1, 2, 3 與 PWM 4, 5, 6, 敘述如下：

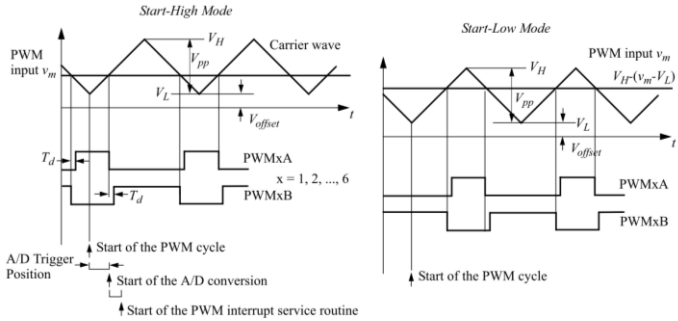
- 參考 PWM 與相位偏移 PWM 需來自同一系列，也就是 PWM 1 為參考，PWM 2 和 3 或 PWM 4, 5 和 6 可相對 PWM 1 做相位偏移。或 PWM 2 為參考，PWM 3 可相對 PWM 2 做相位偏移。相同的，PWM 4(或 5)為參考，PWM 5(或 6)相對 PWM 4(或 5)做相位偏移。但使用 PWM 2(或 3)當 PWM 4, 5 和 6 相位偏移的參考則不允許。
- 參考 PWM 和相位偏移 PWM 在系列中需是連續的，不能使用 PWM1 當 PWM3 的相位偏移參考且不讓 PWM2, 5 或 6 知道。

相位偏移值的單位為角度，當值為-30°，就會相對參考 PWM 產生器的輸出在一個切換週期中向右(落後)偏移 30°，其等效於 PWM 載波向右偏移 30°。當相位值為 30°，則輸出為向左(領先)偏移 30°。

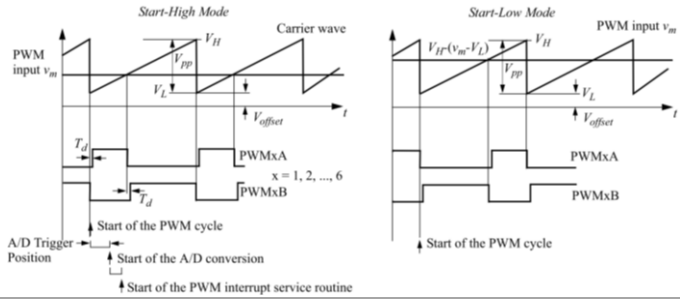
載波

載波波形有兩種類型：三角波(有等效上升和下降斜率區間)和鋸齒波。此外，也有兩種操作模式：低啟動和高啟動模式，說明如下：

三角波載波的 PWM 產生器輸入和輸出波形如下：



鋸齒波載波的 PWM 產生器輸入和輸出波形如下：



上圖說明如何定義空白時間和當 PWM 產生器觸發 A/D 轉換器時的時間序列。如果選擇觸發 A/D 轉換器，從 PWM 週期開始，在某個由 A/D 轉換器觸發位置定義的延遲時間後，A/D 轉換將開始動作。在 A/D 轉換完成後，PWM 中斷服務程序將會開始。

如果 PWM 產生器不觸發 A/D 轉換器，PWM 中斷服務程序將在 PWM 週期開始時執行。

上圖顯示高啟動與低起動模式如何工作。假定 PWM 輸入 v_m 與載波最低值 V_L 和最高值 V_H。在高啟動模式，PWM 正輸出 PWMA 在切換週期開始時為高準位，只要輸入 vm 大於載波則一直維持在高準位。例如，當載波由 0 到 1，V_L=0 且 V_H=1，假設 v_m=0.2，只要載波小於 0.2 則 PWM 輸出 PWMA 將維持在高準位。

另一方面，在低啟動模式，PWM 正輸出 PWMA 在切換週期開始時為低準位，當載波大於 V_H-(v_m-V_L)時則為高準位。例如，當載波由 0 到 1，V_L =0 且 V_H=1，

假設 v_m =0.2，只要載波大於 0.8 則 PWM 輸出 PWMA 將維持在高準位。

注意

低起動模式中，在與載波比較產生 PWM 信號之前，PWM 輸入 v_m 在內部被轉換到 V_H-(v_m- V_L)。在這轉換中，高啟動和低起動模式有著相同的責任週期表示。例如，對於一個 V_L =0 以及 V_H=1 的鋸齒波或是一個 V_L =-V_H 的三角波，高啟動和低起動的 PWMA 輸出的責任週期 D 皆為 D= v_m /V_H。

雙相 PWM 產生器

屬性

參數	描述
PWM Source	PWM 產生器的來源。可為 PWM 1~PWM 6。
Mode Type	PWM 產生器的操作模式。為 6 個模式之一，6 個操作模式的波形說明如下：
Sampling Frequency	PWM 產生器的取樣頻率，單位 Hz。基於此頻率完成計算且更新 PWM 信號責任週期。
PWM Freq. Scaling Factor	PWM 頻率與取樣頻率間的比例因數，可為 1、2 或 3。亦即，PWM 頻率(用來控制開關切換的 PWM 輸出信號頻率)可為取樣頻率的倍數。例如，取樣頻率為 50kHz 且比例因數為 2，意謂著 PWM 頻率為 100kHz。開關以 100kHz 切換，但閘極信號每兩個切換週期 50kHz 更新一次。
Trigger ADC	設定 PWM 產生器是否觸發 A/D 轉換器。為下列其一： Do not trigger ADC：PWM 不會觸發 A/D 轉換器。 Trigger ADC Group A：PWM 將觸發 A/D 轉換器的 A 群組。 Trigger ADC Group B：PWM 將觸發 A/D 轉換器的 B 群組。 Trigger ADC Group A&B：PWM 將觸發 A/D 轉換器的 A 與 B 群組。
ADC Trigger Position	A/D 觸發位置的範圍從 0 到一個小於 1 的值。當為 0 時，A/D 轉換器在 PWM 週期開始時被觸發。當為 0.5 時，A/D 轉換器在 PWM 週期為 180°時被觸發。

Use Trip-Zone i	定義 PWM 產生器使用第 i 個 trip-zone 信號與否，i 的範圍從 1 到 6，為下列其一： Disable Trip-Zone i：停用第 i 個 trip-zone 信號。 One shot：PWM 產生器在單次觸發模式下使用 trip-zone 信號。一旦觸發，PWM 須手動啟動。 Cycle by cycle：PWM 產生器在週期循環的基礎下使用 trip-zone 信號。trip-zone 信號在當週內有效，PWM 將在下一個週期自動重新啟動。
Trip Action	定義 PWM 產生器如何回應跳閘動作。可為下列其一： High impedence：PWM 輸出為高阻抗。 PWM A high & B low：PWM 正輸出為高準位，負輸出為低準位。 PWM A low & B high：PWM 正輸出為低準位，負輸出為高準位。 No action：不採取任何動作。
Peak Value	載波的峰值 Vpk。
Initial Input Value A, B	輸入 A 和 B 的初始值。
Start PWM at Beginning	當設定為"Start"，開始時 PWM 立即啟動。如果設定為"Do not start"，則需使用"Start PWM"函數來啟動 PWM。

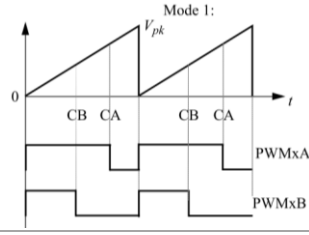
對於雙相 PWM 產生器，輸出是依據操作模式來決定，如下所示。載波依據操作模式可為鋸齒波或三角波，從 0 增加到峰值 V_{pk}，且無偏移。

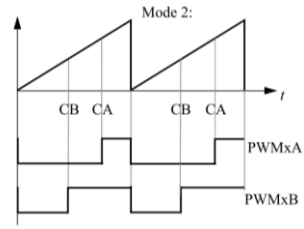
操作模式 1

下圖左為模式 1 的波形。如圖" CA "和" CB "指雙相 PWM 產生器的兩個輸出 A 和 B。每個輸入控制每個輸出的截止時間。

操作模式 2

下圖右為模式 2 的波形。不像模式 1，每個輸入控制每個輸出的導通時間。



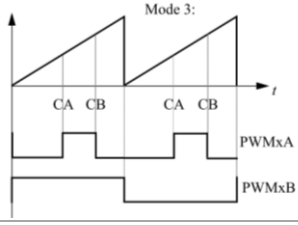


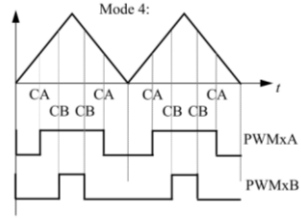
操作模式 3

下圖左為模式 3 的波形。輸入 A 控制 PWM 輸出 A 的導通時間且輸入 B 控制截止時間。PWM 輸出 B 導通時為一完整的 PWM 週期，在下一個週期截止。

操作模式 4

下圖右為模式 4 的波形。載波為三角波，每個輸入控制其輸出的導通與截止。



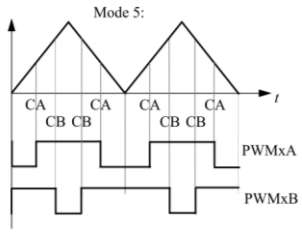


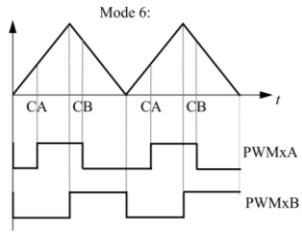
操作模式 5

下圖左為模式 5 的波形。載波為三角波，與模式 4 相似，每個輸入控制其輸出的導通與截止，注意在這例子中 PWM 輸出 B 為交錯。

操作模式 6

下圖右為模式 6 的波形。輸入 A 控制 PWM 輸出 A 的導通時間且輸入 B 控制截止時間，PWM 輸出 B 在前半週為導通，在後半週為截止。





APWM 產生器

屬性

參數	描述
PWM Source	APWM 產生器與捕捉使用相同來源。PWM 來源可為使用 14 個指定 GPIO 端口的 6 個 APWM 的其中一個，如下所列： APWM1(GPIO5, GPIO24, GPIO34) APWM2(GPIO7, GPIO25, GPIO37) APWM3(GPIO9, GPIO26) APWM4(GPIO11, GPIO27) APWM5(GPIO3, GPIO48) APWM6(GPIO1, GPIO49)
PWM Frequency	PWM 產生器的頻率，單位 Hz。
Carrier Wave Type	載波類型和初始 PWM 輸出狀態，為下列其一： Sawtooth(start low)：鋸齒波，初始 PWM 輸出狀態為低準位。 Sawtooth(start high)：鋸齒波，初始 PWM 輸出狀態為高準位。

Stop Action	當 PWM 產生器停止時的輸出狀態。可為下列其中之一： Output low：PWM 輸出設定為低準位。 Output high：PWM 輸出設定為高準位。
Peak-to-peak Value	載波的峰對峰值。
Offset Value	載波的直流偏移量。
Phase Shift	相對於 PWM 產生器參考輸出的輸出相位偏移，單位度(°)。
Initial Input Value	輸入初始值。
Start PWM at Beginning	當設定為"Start"，開始時 PWM 立即啟動。如果設定為"Do not start"，則需使用"Start PWM"函數來啟動 PWM。

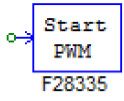
相似於單相 PWM 產生器，APWM 產生器可產生一個參照另一個 PWM 產生器的相位偏移 PWM 信號。相位偏移規則如同單相 PWM 產生器。

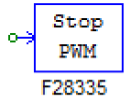
如先前提醒，APWM 產生器相較於單相 PWM 產生器減少一些功能，不能觸發 A/D 轉換器也無法使用 trip-zone 信號。

PWM 開始與停止

PWM 開始與 PWM 停止方塊提供開始或停止 PWM 產生器的功能，圖示與屬性如下所示：

圖示





屬性

參數	描述
PWM Source	PWM 產生器的來源。可為 PWM 1-6、三相 PWM 123 和 PWM 456 以及捕捉 1-6。

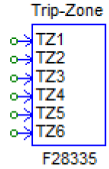
觸發區與其狀態

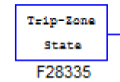
F2833x DSP 內提供了 6 個 trip-zone，trip-zone 1 到 6 使用 GPIO12 到 GPIO17 端口，trip-zone 是用來處理外部故障或是跳閘條件，相對應的 PWM 輸出可以透過編程做出對應的動作。

一個 trip-zone 信號可以被多個 PWM 產生器使用，一個 PWM 產生器可使用任何一個或是 6 個信號信號，信號信號產生的中斷可由中斷方塊處理。

當輸入信號為低準位時，trip-zone 信號會觸發跳閘動作。

圖示





屬性(trip-zone)

參數	描述
Port GPIO12 as Trip-Zone 1	定義若使用 GPIO12 端口為 trip-zone 1。
Port GPIO13 as Trip-Zone 2	定義若使用 GPIO13 端口為 trip-zone 2。
Port GPIO14 as Trip-Zone 3	定義若使用 GPIO14 端口為 trip-zone 3。
Port GPIO15 as Trip-Zone 4	定義若使用 GPIO15 端口為 trip-zone 4。
Port GPIO16 as Trip-Zone 5	定義若使用 GPIO16 端口為 trip-zone 5。
Port GPIO17 as Trip-Zone 6	定義若使用 GPIO17 端口為 trip-zone 6。

屬性(trip-zone 狀態)

參數	描述
PWM Source	PWM 產生器的來源。可為 PWM1-6、三相 PWM 123 和 PWM 456。

可在單次觸發模式或週期循環模式下產生 trip-zone 中斷，在 PWM 產生器參數輸入下定義。週期循環模式下中斷只會影響當前的 PWM 輸出，另一方面，單次觸發模式下，當輸入信號為低準位時中斷觸發跳閘動作。會將 PWM 永久設置，PWM 必須重新啟動才能恢復操作。

當觸發 PWM 產生器產生中斷時，trip-zone 狀態元件表示 trip-zone 信號處於單次觸發模式或週期循環模式下，當輸出為 1，意謂著 trip-zone 信號處於單次觸發模式下，當輸出為 0，trip-zone 信號處於週期循環模式下。

在這例子中 PWM 輸出 B 為交錯。

注意

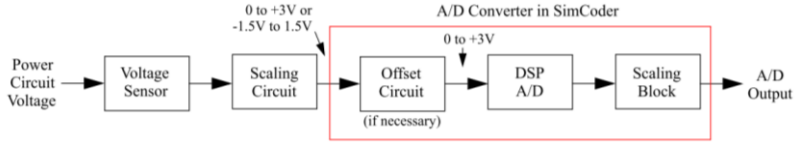
當定義對應 trip-zone 的中斷方塊，中斷方塊的參數“Device Name ”應為 PWM 產生器的名稱，而非 trip-zone 方塊的名稱。例如，若 PWM 產生器“ PWM_G1”使用 trip-zone 方塊“ TZ1 ”的 trip-zone1。對應中斷方塊的名稱“Device Name ”應為“ PWM_G1”而非“ TZ1 ”。中斷方塊的參數“ Channel Number ”在這個例子中沒有使用。

數位/類比轉換器

F2833x DSP 的 12 位元、16 個通道的數位/類比轉換器(ADC)可分成兩個區塊：Group A 及 Group B。在 DSP 上 A/D 轉換器之電壓輸入範圍為 0~+3V。

通常功率級電路的物理量(電壓、電流、速度等)會先經過幾級電路後再送入 DSP，例如：功率級電路上的電壓準位通常比較高，首先使用電壓感測器轉換成控制信號，再由比例運算電路將信號縮放，如有需要再由一個偏移量電路提供信號直流偏移量，所以進到 DSP A/D 輸入的信號都在 0~+3V 間。此信號在 DSP 內轉換成數位值且運用比例運算方塊還原回其原始值。

完整過程如下圖所示：



如上圖，SimCoder 中的 A/D 轉換器元件不完全與實際 DSP 的相同。更明確的說，它包含偏移量電路、A/D 轉換器與比例運算方塊。這樣的設計便於交流系統的應用。

SimCoder 資料庫中 A/D 轉換器的圖示與屬性說明如下。往後說明 " A/D 轉換器 "指的是 SimCoder 資料庫中的 A/D 轉換器，而非 DSP 的，除非另有說明。

圖示

屬性

參數	描述
ADC Mode	定義 A/D 轉換器的操作模式。為下列其一： Continuous：A/D 轉換器連續地執行轉換。當讀取轉換器的值時，其為最後一次轉換的結果。 Start/stop(8-channel)：A/D 轉換器只依據請求執行轉換，只在一個 8 通道群組。 Start/stop(16-channel)：A/D 轉換器只依據請求執行轉換，在所有的 16 個通道。
Ch Ai or Bi Mode	A/D 轉換器通道 Ai 或 Bi 的輸入模式，i 從 0 到 7。輸入模式可為下列其一： AC：輸入範圍從-1.5V 到+1.5V。此選項包含進入 A/D 轉換器的偏移電路，當需要外部準位調節器調整交流信號至 0~+3V 的範圍之情形時是方便的。 DC：輸入為一直流值，範圍為 0~+3V。
Ch Ai or Bi Gain	A/D 轉換器通道 Ai(或 Bi)的增益 k, i 從 0 到 7。

操作模式

當設定為連續模式時，A/D 轉換器可自主地執行轉換。PWM 產生器觸發“ 啟動/停止 ”模式的轉換。

注意

PWM 產生器觸發 A/D 轉換器有下列的限制：

- A/D 轉換器只能被一個 PWM 產生器觸發。也就是如果有多個 PWM 產生器，只有一個可設定用來觸發 A/D 轉換器，其餘的應設定不能觸發 A/D 轉換器。
- 當群組內某些信號也在電路內但與 PWM 產生器的頻率有不同的取樣率，則不允許 A/D 轉換器由一個 PWM 產生器觸發。
  在這樣情況下，建議 A/D 轉換器設為連續模式。

輸出縮放

輸出依據下列方程式作縮放：

$$V_o = k * V_i$$

其中 *Vi* 為 A/D 轉換器輸入端口的值。

輸入偏移與縮放

注意 A/D 轉換器的輸入必須保持在輸入範圍內，當輸入超出範圍時，會被箝位在極限值並給一個警告信息。

A/D 轉換器輸入端口的信號也必須如此縮放，當輸入通道模式為直流時，最大輸入電壓將被縮放到+3V；當輸入通道模式為交流時，最大輸入峰值電壓將被縮放到+1.5V。

在許多應用中，其電路監控的變數為交流信號，特別是交流馬達驅動系統。對於每個這類的交流信號，為了要調整信號準位到可被接受的 0 到+3V 範圍，硬體電路須在 DSP 類比輸入端加入偏移電路。

對於這種情況，F2833x DSP 的 SimCoder A/D 轉換器提供一個便利的方式。取代準位調整與 A/D 輸出信號縮放，使用者可在 SimCoder A/D 轉換器中選擇偏移選項與比例因數，將會產生相對應的標的程式碼。

下列有兩個範例說明如何使用 A/D 轉換器：一個直流輸入而另一個交流輸入。

A/D 轉換器通道直流模式範例

假設功率級迴路電壓為一直流量，範圍如下：

$V_{i\_mim} = 0V$, $V_{i\_\max} = 150V$

A/D 轉換器的輸入模式設定為直流，範圍從 0 到 3V。假設在某一個點電壓的實際值為：

$V_i = 100V$

設定電壓取樣增益為 0.01，經過電壓取樣後，輸入的最大值與實際值變為：

$V_{i\_\max\_s} = 150 * 0.01 = 1.5V$

$$V_{i\_s} = 100 * 0.01 = 1V$$

為了運用 DSP 的全部範圍，使用一個增益為 2 的調節電路。結合電壓取樣的增益與調節電路後變成：0.01*2=0.02。經過調節電路後，DSP A/D 轉換器的輸入最大值和實際值變為：

$$V_{i\_\max\_s\_c} = 1.5 * 2 = 3V$$

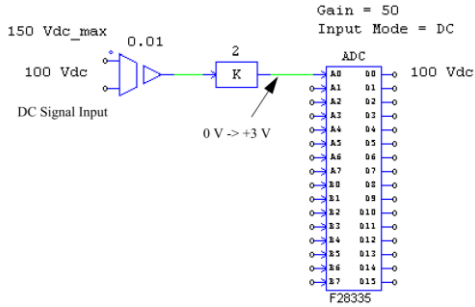
$$V_{i\_s\_c} = 1 * 2 = 2V$$

選擇在 DSP A/D 轉換器後的縮放方塊可恢復到原本功率級電路的數值，在此範例中，增益為 50，為調節電路與電壓感測器增益組合的倒數，A/D 轉換器的輸出最大值與實際值為：

$$V_{o\_\max} = 50 * 3 = 150V$$

$$V_o = 50 * 2 = 100V$$

PSIM 中 A/D 通道的增益設定為 50，電路連接與設定如下圖所示：



注意

此範例中如果比例方塊的增益由 2 變為 1，則 A/D 增益由 50 變為 100，模擬結果將會相同，但產生的硬體程式碼則不正確。這是因為硬體程式碼以為的最大輸入值會被縮放到+3V，但此範例只到 1.5V。因此，在直流模式中，使用者必須這樣設定電路使得最大的輸入值縮放到+3V。

A/D 轉換器通道交流模式範例

在另一個範例，假設功率級迴路電壓為一交流量，範圍如下：

$$V_{i\_\max} = +/-75V$$

A/D 轉換器的輸入模式設定為交流，範圍從-1.5V 到+1.5V。假設電壓的實際值其峰值為：

$$V_i = +/-50V$$

設定電壓取樣增益為 0.01，經過電壓取樣後，輸入的最大值與實際值變為：

$$V_{i\_\max\_s} = +/-0.75V$$

$$V_{i\_s} = +/-0.5V$$

因為 A/D 轉換器的輸入範圍從-1.5V 到+1.5V，在信號送進 DSP 前需要做縮放，需要一個增益為 2 的調節電路(亦即 1.5/0.75=2)。經過調節電路後，DSP A/D 轉換器的輸入最大值和實際值變為：

$$V_{i\_\max\_s\_c} = +/-1.5V$$

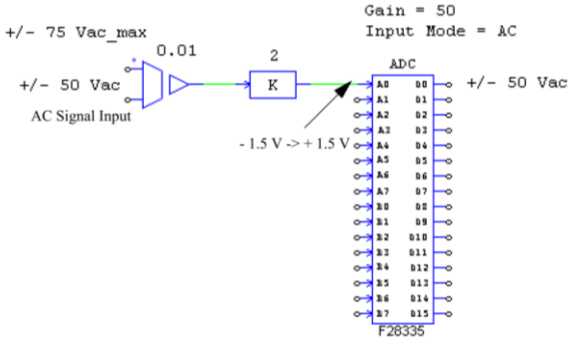
$$V_{i\_s\_c} = +/-1V$$

選擇在 DSP A/D 轉換器後的縮放方塊可恢復到原本功率級電路的數值，在此範例中，增益為 50，為調節電路與電壓感測器增益組合的倒數，A/D 轉換器的輸出最大值與實際值為：

$$V_{o\_\max} = +/-75V$$

$$V_o = +/-50V$$

PSIM 中 A/D 通道的增益設定為 50，電路連接與設定如下圖所示



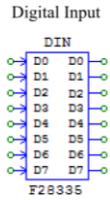
⚠注意	在此電路中，交流信號直接傳送給 A/D 轉換器。這是因為當 A/D 輸入設定為交流時，輸入範圍從-1.5V 到+1.5V，A/D 轉換器方塊已經包含執行直流偏移的調節電路之功能。在實際硬體電路，交流信號仍須經由縮放與偏移以達到 DSP A/D 轉換器所需的範圍 0V 到 3V 內。

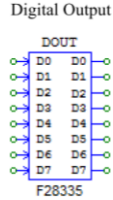
確保針對硬體所產生的程式碼之正確性，在 A/D 轉換器輸入端口的最大峰值必須縮放到 1.5V。

數位輸出與輸入

F2833x DSP 有 88 個可配置為數位輸入(Digital Input)或數位輸出(Digital Output)的通用輸出入(general-purpose-input-output, GPIO)端口。SimCoder 提供了八通道的數位輸出入方塊，多個八通道方塊可在同一個電路中使用。

圖示





屬性(數位輸入)

參數	描述
Port Position for Input i	輸入 i 的端口位置，i 從 0 到 7。為 88 個 GPIO 端口的其中之一，從 GPIO0 到 GPIO87。
Use as External Interrupt	顯示若此端口用來當外部中斷輸入。

屬性(數位輸出)

參數	描述
Port Position for Output i	輸出 i 的端口位置，i 從 0 到 7。為 88 個 GPIO 端口的其中之一，從 GPIO0 到 GPIO87。
⚠注意	當 GPIO 作為輸入端口時，同一端口就不可當外設端口使用。例如，GPIO1 設定為數位輸入也同時為 PWM1 輸出時，就會回報錯誤。

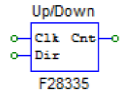
在 F2833x DSP 中，從 GPIO0 到 GPIO63 最多可定義七個外部中斷源 (具體而言，從 GPIO0 到 GPIO31 最多 2 個，GPIO32 到 GPIO63 最多 5 個)。GPIO0 到 GPIO31 的外部中斷比 GPIO32 到 GPIO63 的優先。

上/下計數器

F2833x DSP 有兩個上/下計數器。第一個可為 GPIO20-21 或 GPIO50-51，第二個則在 GPIO24-25。

⚠注意	在 GPIO20-21 與 GPIO50-51 的計數器 1 使用相同的內部函數方塊，但不可同時使用。

圖示



屬性

參數	描述
Counter Source	計數器的來源。為下列其一： Counter1 (GPIO20, 21)：計數器 1 使用 GPIO20 和 21。 Counter1 (GPIO50, 51)：計數器 1 使用 GPIO50 和 51。 Counter2 (GPIO24, 25)：計數器 1 使用 GPIO24 和 25。

在圖示中，“ Clk ”為輸入時脈信號，“ Dir ”為計數方向的信號。當 Dir 輸入為 1，計數器將正向計數，當輸入為 0 時，計數器將反向計數。

⚠注意	“ Clk ”輸入對應計數器的第一個端口，“ Dir ”輸入對應第二個。例如，當計數器 1 為 GPIO20-21 時，GPIO20 為“ Clk ”而 GPIO21 為“ Dir ”。

上/下計數器的輸出給予計數值。

⚠注意	上/下計數器與編碼器使用相同的來源。但同一個 GPIO 不能同時為計數器和編碼器。例如，同時使用編碼器 1 和計數器 1 會產生衝突解不被允許。

編碼器與其狀態

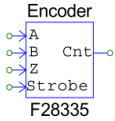
F2833x DSP 有兩個編碼器，編碼器 1 可為 GPIO20-21 或 GPIO50-51，編碼器 2 為 GPIOP24-25。

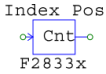
⚠注意	在 GPIO20-21 與 GPIO50-51 的編碼器 1 使用相同的內部函數方塊，但不可同時使用。編碼器的輸出給予計數值。

編碼器狀態方塊是用來表示哪個輸入信號(或指示信號或選通信號)產生中斷。也可藉由指示(Z)信號與選通信號產生硬體中斷而編碼器狀態的輸出將表示何種信號產生中斷。當輸出為 0 即指示信號產生中斷，當輸出為 1 即選通信號產生中斷。

編碼器指示/選通位置方塊用來鎖住編碼器的初始位置。當此方塊輸入為 0 時，編碼器計數將設定為 0，當輸入變為 1 時編碼器開始動作，當指示/選通事件發生時編碼器會鎖住計數值。

圖示





屬性(編碼器)

參數	描述
Encoder Source	編碼器的來源。為下列其一: Encoder1(GPIO20, 21)：編碼器 1 使用 GPIO20 和 21，GPIO22 為選通而 GPIO23 為指示(Z)。 Encoder1(GPIO50, 51)：編碼器 1 使用 GPIO50 和 51，GPIO52 為選通而 GPIO53 為指示(Z)。 Encoder2(GPIO24, 25)：編碼器 2 使用 GPIO24 和 25，GPIO27 為選通而 GPIO28 為指示(Z)。
Use Z Signal	定義若編碼器使用指示(Z)信號。

Use Strobe Signal	定義若編碼器使用選通信號。
Counting Direction	計數方向可為正向或反向。當設定為正向，計數器遞增計數，反之，計數器遞減計數。
Z Signal Polarity	定義指示信號的觸發極性： 高準位動作 低準位動作
Strobe Signal Polarity	定義選通信號的觸發極性： 高準位動作 低準位動作
Encoder Resolution	外部編碼器硬體的解析度。當為 0 時，編碼計數器會持續計數且不會重置，例如，解析度設置為 4096，當到達 4095 時，計數器將重置為 0。

屬性(編碼器狀態)

參數	描述
Encoder Source	定義哪個編碼器產生中斷。為下列其一，需為同一原理圖上的同一編碼器： Encoder1 (GPIO20, 21)：編碼器 1 使用 GPIO20 和 21。 Encoder1 (GPIO50, 51)：編碼器 1 使用 GPIO50 和 51。 Encoder2 (GPIO24, 25)：編碼器 2 使用 GPIO24 和 25。

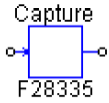
屬性(編碼器指示/選通位置)

參數	描述
Encoder Source	定義哪個編碼器產生中斷。為下列其一，需為同一原理圖上的同一編碼器： Encoder1 (GPIO20, 21)：編碼器 1 使用 GPIO20 和 21。 Encoder1 (GPIO50, 51)：編碼器 1 使用 GPIO50 和 51。 Encoder2 (GPIO24, 25)：編碼器 2 使用 GPIO24 和 25。
Latch Position	指定保持計數形式，由下列選擇： IndexPos，若編碼器設定"Use Z Signal"不為"No"。 StrobePos，若編碼器設定"Use Strobe Signal"不為"No"。
Type of Postion	可由下列做選擇： 第一次鎖住的位置，或 目前鎖住的位置

捕捉與其狀態

F2833x DSP 提供六個捕捉。捕捉可產生中斷，且中斷觸發模式將由中斷方塊定義。

圖示



捕捉狀態方塊輸出為 1 或 0, 1 表示為上升邊緣而 0 表示為下降邊緣。

屬性(捕捉)：

參數	描述
Capture Source	捕捉的來源。使用 14 個指定 GPIO 的六個捕捉，如下所列： Capture1 (GPIO5, GPIO24, GPIO34) Capture2 (GPIO7, GPIO25, GPIO37) Capture3 (GPIO9, GPIO26) Capture4 (GPIO11, GPIO27) Capture5 (GPIO3, GPIO48) Capture6 (GPIO1, GPIO49)
Event Filter	事件濾波器預分頻。輸入信號由選定的預分頻劃分。
Timer Mode	捕捉計數器定時模式。可為 Absolute time 或 Time Difference。

屬性(捕捉狀態)：

參數	描述
Capture Source	捕捉的來源。可為 6 個捕捉的其中之一。

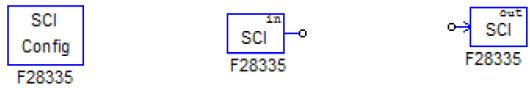
串列傳輸界面(SCI)

F2833x DSP 提供串列傳輸界面(SCI)的功能。透過 SCI，DSP 內部數據可由外部 RS-232 線傳送到電腦端。PSIM 提供 DSP 與電腦兩端所有傳送與接收數據必要的功能並在電腦端顯示。這是一個便於即時監控、除錯與調整 DSP 程式碼的方式。

對於 SCI 與監控功能更詳細的描述請參考"Tutorial - Using SCI for Real-time Monitoring in F2833x Target.pdf "文件。

SimCoder 提供三個 SCI 函數方塊：SCI 配置、SCI 輸入與 SCI 輸出說明如下：

圖示



SCI 配置

SCI 配置方塊定義 SCI 端口、傳輸速度、同位元檢查類型與數據緩衝區大小。

屬性

參數	描述
SCI Port	定義 SCI 端口，有 7 組 GPIO 端口可供 SCI 使用，如下所列： SCIA(GPIO28, GPIO29) SCIA(GPIO35, GPIO36) SCIB(GPIO9, GPIO11) SCIB(GPIO14, GPIO15) SCIB(GPIO18, GPIO19) SCIB(GPIO22, GPIO23) SCIC(GPIO62, GPIO63)

Speed(bps)	SCI 傳輸速度，單位 bps(每秒多少位元)，目前所提供的速度列表：200000, 115200, 57600, 38400, 19200 和 9600bps。或手動指定任何其他速度。
Parity Check	在傳輸中針對誤檢查的同位元檢查設定。可為無、奇或偶。
Output Buffer Size	針對 SCI 在 DSP 中配置的數據緩衝區大小，緩衝區位於 RAM 內，每個緩衝區元件儲存一包含三個 16 位元的字(6 位元組或 48 位元)的資料點。
⚠注意	緩衝區大小需適當選擇。一方面，為了收集更多數據以便長時間監控更多變數故偏好大的緩衝區，另一方面，DSP 內部記憶體有限，緩衝區太大則會干擾 DSP 的正常運作。

對於如何選擇緩衝區大小，更詳盡的資訊請參照"Tutorial - Using SCI for Real-time Monitoring in F2833x Target.pdf "文件。

SCI 輸入

SCI 輸入方塊用來定義一可變 DSP 程式碼中的變數。SCI 輸入變數的名稱會顯示在 DSP 示波器(在 Utilities 選單下)中，在執行時可透過 SCI 改變其數值。

SCI 輸入方塊提供一個便捷的方式來改變參考值或微調控制器參數。

屬性

參數	描述
Initial value	SCI 輸入變數的初始值

在原理圖中，SCI 輸入可視為常數。程式碼在 DSP 中執行時儘管其值可被改變，但模擬時還是被固定在初始值。

SCI 輸出

SCI 輸出方塊用來定義變數的顯示。當 SCI 輸出方塊連接到一個節點，SCI 輸出方塊的名稱會顯示在 DSP 示波器(在 Utilities 選單下)中，在執行期間，此變數的數據將透過 SCI 由 DSP 傳送到電腦，在 DSP 示波器上顯示波形。

SCI 輸出方塊提供一個方便監控 DSP 波形的方法。

屬性

參數	描述
Data Point Step	定義如何收集常見數據。如果 Data Point Step 為 1，每個數據將被收集與傳送。如果設定為 10，每 10 個點只有一個會被收集與傳送。
⚠注意	當數據點刻度太小時，或許會有過多的數據點且可能無法全部傳送。在這個例子中，部分的數據點在數據傳輸時會被捨棄。

數據點刻度參數只使用在連續模式的 DSP 示波器中，當在快照模式中，此參數會被忽略且每個點都會被收集與傳送。

在模擬中，SCI 輸出可視為電壓探測器。

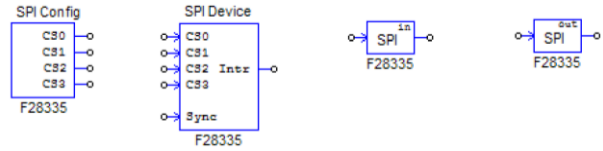
串列外設界面(SPI)

F2833x DSP 提供串列外設界面的功能。藉由 TI F2833x 標的資料庫中 SPI 方塊，可容易與方便地實現與外部 SPI 設備(如外部 A/D 與 D/A 轉換器)傳輸的功能。手寫 SPI 設備程式碼通常是一件耗時且不容易的任務。有了支援 SPI 功能，PSIM 大大地簡化並加速撰寫程式碼與硬體實現的過程。

對於如何使用 SPI 方塊更詳細的描述請參考"Tutorial - Using SCI for Real-time Monitoring in F2833x Target.pdf "文件。

SimCoder 提供四個 SPI 函數方塊：SPI 配置、SPI 設備、SPI 輸入與 SPI 輸出說明如下：

圖示



SPI 配置

SPI 配置方塊定義 SPI 端口、晶片選取引腳與 SPI 緩衝區大小。它必須存在於使用 SPI 的電路圖中且需位於主電路圖中。

屬性：

參數	描述
SPI Port	定義 SPI 端口，為 GPIO16-19 或 GPIO54-57。
Chip Select Pin0, 1, 2, and 3	晶片選擇引腳的 GPIO 端口。PSIM 最多支援 16 個 SPI 設備，其需要 4 個 GPIO 晶片選擇引腳，定義為晶片選擇引腳 1 到 3。GPIO 端口和 SPI slave transmit -enable(SPISTE)用來產生晶片選擇信號。

SPI Buffer Size	SPI 命令的緩衝區大小。每個記憶體單元保存 SPI 命令的引索。通常可在所有 SPI 輸出入元件指定緩衝區大小為 1 加上 SPI 命令數(即開始轉換命令、接收數據命令、傳送數據命令與同步命令)。

SPI 設備

SPI 設備方塊定義相對應的 SPI 硬體設備的資訊。電路圖中 SPI 設備方塊的數量需與 SPI 硬體設備數相同。

屬性：

參數	描述
Chip Select Pins	SPI 設備對應的晶片選擇引腳之狀態。當晶片選擇引腳為此狀態時，則此 SPI 設備被選取。
Communication Speed(MHz)	SPI 傳輸速度，單位 MHz。
Clock Type	SPI 時脈類型，由 SPI 硬體設備決定。可為下列其一： Rising edge without delay：時脈通常為低準位，且在上升緣時鎖住數據。 Rising edge with delay：時脈通常為低準位，且在有延遲的上升緣時鎖住數據。 Falling edge without delay：時脈通常為高準位，且在下降緣時鎖住數據。 Falling edge with delay：時脈通常為高準位，且在有延遲的下降緣時鎖住數據。
Command Word Length	SPI 傳輸命令的字長度或有效位元長度。可從 1 到 16 位元。
Sync. Active Mode	SPI 設備同步命令之觸發模式。可為上升緣或下降緣。

SPI Initial Command	初始化 SPI 設備的 SPI 命令。
Hardware Interrupt Mode	指定 SPI 設備產生的中斷信號類型。只有當 SPI 設備的中斷輸出節點連接到數位輸出元件的輸入時才有效。為下列其中之一： No hardware interrupt Rising edge Falling edge
Interrupt Timing	指定當 SPI 設備完成轉換時如何產生中斷。可為下列其一： No interrupt：無中斷產生。在這種情況，DSP 送命令給 SPI 輸入設備。此設備開始轉換並在相同命令下回傳結果。 Multiple interrupt in series：在每次轉換後產生多個串連中斷，針對有一個 A/D 轉換器與多個輸入通道的 SPI 設備。在這種情況中，DSP 送出第一個轉換命令且 SPI 設備開始轉換。當轉換完成，SPI 設備將產生中斷。在此中斷服務程序，DSP 將傳送一個命令以獲取轉換結果且相同 SPI 輸入設備的另個通道開始一個新的轉換。 One-time interrupt：在轉換結束時只產生一個中斷，SPI 設備可在一個請求中執行多個通道轉換。這種情況下，DSP 傳送命令到 SPI 輸入設備且 SPI 設備完成多個輸入通道的轉換。當轉換完成時，SPI 設備將產生中斷。
Command Gap(ns)	兩個 SPI 命令的間隔，單位奈秒(nsec)。
Conversion Sequence	定義 SPI 輸入元件的名稱，以逗號區隔並決定轉換順序。
⚠注意	只有當 SPI 設備產生多個串連中斷時變數才有效。

在電路圖中，所有 SPI 設備的晶片選取引腳連接到 SPI 配置方塊的晶片選取引腳，不須定義晶片選取邏輯如何實現。然而，在實際的硬體則需依據相對應的晶片選取邏輯實現。

SPI 命令為一連串藉由逗號隔開的 16 位元數字組成。在這 16 位元數字中，命令只使用較低位元為有效位元。例如，如果命令字長為 8，0~7 位元為命令，8~15 則未被使用。

SPI 設備可為輸出或輸入。例如，外部 A/D 轉換器為輸入設備，通常 DSP 會傳送一個或多個 A/D 轉換命令到此設備，然後設定一個同步信號開始轉換。同步信號在同一設備的下一個命令重置。

使用同步信號的 SPI 輸入設備通常需要一個中斷引腳去觸發 DSP 進入中斷服務程序。

另一方面，外部 D/A 轉換器為一輸出設備。通常 DSP 會傳送一個或多個 D/A 轉換命令到此設備，然後設定一個同步信號開始轉換。同步信號在同一設備的下一個命令重置。

SPI 輸入

SPI 輸入設備有多個輸入通道。SPI 輸入方塊用來定義 SPI 傳輸的輸入通道屬性且一個 SPI 輸入方塊對應一個輸入通道。

屬性：

參數	描述
Device name	SPI 輸入設備名稱
Start Conversion Command	開始轉換的命令，16 進制，以逗點區隔(例如，0x23, 0x43,0x00)。
Receiving Data Command	接收數據的命令，16 進制，以逗點區隔(例如，0x23, 0x43,0x00)。

Data Bit Position	定義在接收的數據字串中數據位元的位置。公式為：ElementName={Xn[MSB..LSB]} 其中 ElementName 為 SPI 輸入設備的名稱。如果是目前的 SPI 輸入設備，以 y 代替。 {}意味著括弧內項目重複多次。 Xn 為從 SPI 輸入設備接收的第 n 個字，n 從 0 開始。 MSB..LSB 定義字的有效位元位置。
Input Range	指定參數 Vmax 定義的輸入範圍。只有當 SPI 設備為 A/D 轉換器時此變數才有效。若設備轉換器模式為 DC，輸入範圍由 0 到 Vmax。若設備轉換器模式為 AC，輸入範圍由-Vmax/2 到 Vmax/2。
Scale Factor	輸入比例因數 Kscale。若比例因數為 0，SPI 設備不為 A/D 轉換器，其結果將與 DSP 從 SPI 傳輸接收的完全一樣。反之，若 SPI 設備為 A/D 轉換器，其結果與該因數和 A/D 轉換模式成比例。
ADC Mode	設備的 A/D 轉換模式。可為 DC 或 AC，注意只有當此設備為 A/D 轉換器時參數才有效。
Initial Value	輸入的初始值。

Data Bit Position 公式用來定義 SPI 輸入設備的數據長度。例如，y=x1[3..0]x2[7..0]，代表數據長度為 12，結果為第二個字的較低四個位元與第三個字的較低 8 個位元。如果接收到的字串為 0x12,0x78,0xAF，結果為 0x8AF。

如果比例因數不為 0，則輸出將與下列成比例：

在 DC 轉換模式
- 模擬：$Output = Input \cdot K_{scale}$
- 硬體：$Output = \frac{\mathrm{Re}\,sult \cdot V_{\max} \cdot K_{scale}}{2^{Data\_\mathrm{Length}}}$

在 AC 轉換模式
- 模擬：$Output = Input \cdot K_{scale}$
- 硬體：$Output = \frac{(\mathrm{Re}\,sult - 2^{Data\_Length-1}) \cdot V_{\max} \cdot K_{scale}}{2^{Data\_Length-1}}$

Data_Length 參數由 Data Bit Position Formula 計算得之。

SPI 輸出

SPI 輸出設備有多個 SPI 輸出通道。SPI 輸出方塊用來定義 SPI 傳輸的輸出通道屬性且一個 SPI 輸出方塊對應一個輸出通道。

屬性

參數	描述
Device name	SPI 輸出設備名稱
Scale Factor	輸出比例因數 Kscale。若比例因數為 0，SPI 設備不為 A/D 轉換器，其結果將與 DSP 從 SPI 傳輸接收的完全一樣。反之，若 SPI 設備為 A/D 轉換器，其結果與該因數和 A/D 轉換模式成比例。
Output Range	指定參數 Vmax 定義的輸出範圍。只有當 SPI 設備為 A/D 轉換器時此變數才有效。若設備轉換器模式為 DC，輸入範圍由 0 到 Vmax。若設備轉換器模式為 AC，輸入範圍由-Vmax/2 到 Vmax/2。
DAC Mode	設備的 D/A 轉換模式。可為 DC 或 AC，注意只有當此設備為 D/A 轉換器時參數才有效。
Sending Data Command	傳送輸出數據的命令，16 進制，以逗點區隔(例如，0x23,0x43,0x00)。
Data Bit Position	定義在傳送的數據字串中數據位元的位置。公式為：ElementName={Xn[MSB..LSB]} 其中 ElementName 為 SPI 輸出設備的名稱。如果是目前的 SPI 輸出設備，以 y 代替。 {}意味著括弧內項目重複多次。 Xn 為從 SPI 輸出設備傳送的第 n 個字，n 從 0 開始。 MSB..LSB 定義字的有效位元位置。

Sync. Command	SPI 輸出設備的同步輸出通道之命令，16 進制，以逗點區隔(例如，0x23,0x43,0x00)。當 SPI 輸出設備無同步信號時才使用此命令。

Data Bit Position 公式用來定義 SPI 輸出設備的數據長度。例如，y=x1[3..0]x2[7..0]代表數據長度為 12，結果為第二個字的較低四個位元與第三個字的較低 8 個位元。如果接收到的字串為 0x12,0x78,0xAF，結果為 0x8AF。

如果比例因數不為 0，則輸出將與下列成比例：

在 DC 轉換模式
- 模擬：$Output = Input \cdot K_{scale}$
- 硬體：$Output = \frac{\mathrm{Re}\,sult \cdot K_{scale} \cdot 2^{Data\_\mathrm{Length}}}{V_{\max}}$

在 AC 轉換模式
- 模擬：$Output = Input \cdot K_{scale}$
- 硬體：$Output = 2^{Data\_Length} + \frac{\mathrm{Re}\,sult \cdot K_{scale} \cdot 2^{Data\_\mathrm{Length\text{-}1}}}{V_{\max}}$

Data_Length 參數由 Data Bit Position Formula 計算得之。

專案設定與記憶體配置

當針對 F2833x 硬體標的產生程式碼時，SimCoder 也對於程式碼編譯、連結與上傳至 DSP 開發環境 TI CCS 創造一個完整的專案文件。

目前，支援 CCS 3.3 版。假定 PSIM 電路圖文件是“ test.sch ”，在產生程式碼後，電路圖文件的目錄下將產生“ test(C Code) ”的子資料夾並包含下列文件：

- test.c：產生的 C 程式碼
- PS_bios.h：SimCoder F2833x 資料庫的標題文件
- passwords.asm：說明 DSP 程式碼密碼的文件
- test.pjt：Code Composer Studio 的專案文件
- DSP28335_Headers_nonBIOS.cmd：外設暫存器連接器命令文件
- F28335_FLASH_Lnk.cmd：Flash memory 快閃記憶體連接器命令文件
- F28335_FLASH_RAM_Lnk.cmd：Flash RAM memory 連接器命令文件
- F28335_RAM_Lnk.cmd：RAM memory 連接器命令文件

⚠注意

如果硬體標的不是 F28335，連接器命令文件的名稱將被指定給對應的標的硬體。例如，如果標的硬體為 F28334，文件名稱相對地為 F28334 FLASH Lnk.cmd, F28334 FLASH RAM Lnk.cmd 與 F28334RAM Lnk.cmd。

除此之外，專案也需要下列文件：

- PS_bios.lib：SimCoder F2833x 資料庫，位於 PSIM 資料夾
- C28x_FPU_FastRTS_beta1.lib：TI 快速浮點資料庫，位於 PSIM\lib 子資料夾

當程式碼產生時，會自動複製"C28x_FPU_FastRTS_beta1.lib"和"PS_bios.lib"這兩個文件到專案資料夾。

每次產生程式碼時，也會創造.c 與.pjt 文件(此範例中為 test.c 和 test.pjt)。如果已手動更改這兩個文件，一定要複製更改的文件到不同的位置。否則，下次執行產生程式碼時，更改的文件將會被覆蓋。

專案設定

在 CCS 專案文件，需提供下列設定：

- *RAM Debug*：在 Debug 模式編譯程式碼並 RAM memory 在執行。
- *RAM Release*：在 Release 模式編譯程式碼並 RAM memory 在執行。
- *Flash Release*：在 Release 模式編譯程式碼並 Flash memory 在執行。
- *Flash RAM Release*：在 Release 模式編譯程式碼並 RAM memory 在執行。

當選擇 RAM Debug 或 RAM Release 設定時，CCS 使用連接器命令文件 F28335_RAM_Lnk.cmd 配置程式與數據空間。

當選擇 Flash Release 設定時，CCS 使用連接器命令文件 F28335_FLASH_Lnk.cmd 配置程式與數據空間。

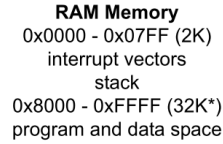
當選擇 Flash RAM Release 設定時，CCS 使用連接器命令文件 F28335_FLASH_RAM_Lnk.cmd 配置程式與數據空間。記憶體配置相同於 RAM Release 設定。

在 release 模式下編輯程式碼快過在 debug 模式，在 RAM Release 和 Flash RAM Release 最快，RAM Debug 則較慢，Flash Release 則是最慢。在開發中，通常會先在 RAM Debug 以易於 debug，然後切換到 RAM Release 隨後到 RAM Release 或 Flash RAM Release。

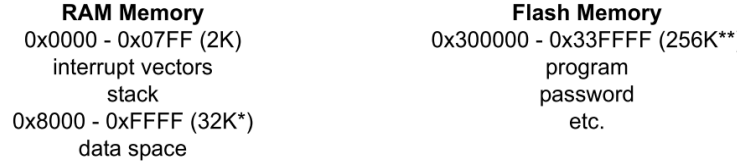
記憶體配置

在產生連結的文件，記憶體配置由下列方式定義：

RAM Debug、RAM Release 與 Flash RAM Release 的設置：



Flash Release 的設置



注意

1. SimCoder 預先定義了用於程式與數據空間的 RAM memory：

- 針對 F28335, F28334：從 0x8000 到 0xFFFF (32K)
- 針對 F28332：從 0x8000 到 0xDFFF (16K)
- 如果程序加上數據空間超過 RAM 空間的大小，則必須選擇 Flash Release 作為專案設定。

2. SimCoder 預先定義了用於程式空間的 flash memory：

- 針對 F28335：從 0x300000 到 0x33FFFF (256K)
- 針對 F28334：從 0x320000 到 0x33FFFF (128K)
- 針對 F28332：從 0x330000 到 0x33FFFF (64K)