行政院及所屬各機關出國報告

(出國類別：實　習)

西門子電力技術服務國際公司
動態分析短期課程

服務機關：台電系統規劃處

出國人職　　稱：電機工程師

姓　　名：呂天桂

派赴國家：美國

出國期間：105.10.15~105.10.24

報告日期：105.12.9

出國報告審核表

出國報告名稱：西門子電力技術服務國際公司動態分析短期課程		
出國人姓名	職稱	服務單位
呂天桂	電機工程師	台灣電力公司系統規劃處
出國類別	□考察 □進修 □研究 ■實習 □其他_____(例如國際會議、國際比賽、業務接洽等)	
出國期間：105 年 10 月 15 日 至 105 年 10 月 24 日		報告繳交日期：105 年 12 月 9 日

出國人員自我審核	計畫主辦機關審核	審 核 項 目
■	□	1.依限繳交出國報告
■	□	2.格式完整(本文必須具備「目地」、「過程」、「心得及建議事項」)
■	□	3.無抄襲相關資料
■	□	4.內容充實完備.
■	□	5..建議具參考價值
■	□	6.送本機關參考或研辦
□	□	7.送上級機關參考
□	□	8.退回補正，原因：
□	□	(1)不符原核定出國計畫
□	□	(2)以外文撰寫或僅以所蒐集外文資料為內容
□	□	(3)內容空洞簡略或未涵蓋規定要項
□	□	(4)抄襲相關資料之全部或部分內容
□	□	(5)引用相關資料未註明資料來源
□	□	(6)電子檔案未依格式辦理
□	□	9.本報告除上傳至出國報告資訊網外，將採行之公開發表：
□	□	(1)辦理本機關出國報告座談會(說明會)，與同仁進行知識分享。
□	□	(2)於本機關業務會報提出報告
□	□	(3)其他 _____
□	□	10.其他處理意見及方式：

報告人：　　單位主管：　　主管處主管：　　莊副總經理：

說明：
一、 各機關可依需要自行增列審核項目內容，出國報告審核完畢本表請自行保存。
二、 審核作業應儘速完成，以不影響出國人員上傳出國報告至「公務出國報告資訊網」為原則。

行政院及所屬各機關出國報告提要

出國報告名稱：西門子電力技術服務國際公司動態分析短期課程

頁數 47 含附件：□是 ■否

出國計畫主辦機關/聯絡人/電話：台灣電力公司/陳德隆/2366-7685

出國人員姓名/服務機關/單位/職稱/電話：

呂天桂/台灣電力公司/系統規劃處/電機工程師/2366-6913

出國類別：□1.考察 □2.進修□3.研究 ■4.實習 □5.其他

出國期間：105 年 10 月 15 日~105 年 10 月 24 日 出國地區：美國

報告日期：105 年 12 月 9 日

分類號/目

關鍵詞：自建模型(User-Defined Model)、動態模擬(Dynamic Simulation)、副程式(Subroutine)、程式碼雛型(Code Template)

內容摘要：(二百至三百字)

本次參加西門子電力技術服務國際公司動態分析短期課程主題為電力系統分析軟體(PSS/E)之動態自建模型撰寫，課程包含 PSS/E 動態模擬架構及程序、動態自建模型撰寫及範例練習等兩大部分。

本課程首先詳細介紹 PSS/E 動態模擬之內部運作流程及使用之陣列等資料結構，再依 PSS/E 程式運作所需導入電力設備動態模型

撰寫程式之架構及撰寫技巧介紹，最後藉由課程中實際撰寫簡易型激磁系統模型以深入了解各程式細節。藉由學習 PSS/E 程式內部運作流程及應用模型之撰寫，有助於增強電力系統動態模擬檢討能力。

本文電子檔已傳至出國報告資訊網

(http://report.nat.gov.tw/reportwork）

報 告 內 容

圖 目 錄

表 目 錄

一、出國緣由與目的

由於未來臺灣大型離岸風場及太陽光電專區的加入，如此間歇性之再生能源對電網系統的影響將與日俱增。在電力系統規劃階段，除了電力系統穩態潮流分析外，動態穩定度分析亦是相當重要的課題。考量未來再生能源規模龐大，於動態模擬時需使用更為精確之再生能源動態模型，使得模擬分析更為精準，以利電力系統的規劃。

此訓練課程除可提供本處派訓人員建構更完善之電力系統分析軟體 PSS/E 之動態分析能力，亦強化動態模型分析及撰寫能力，並吸取國外之經驗及技術，以期可應用於台電系統中。

二、出返國行程

本出國計畫，自 105 年 10 月 15 日起，至 105 年 10 月 24 日止，前後共 10 天，詳細行程及地理位置圖如下所示：

2-1 去程

表 2-1 去程行程表

日期	出發地點	出發時間	抵達地點	抵達時間
105.10.15	台北(桃園機場) TPE	7:50	紐約甘迺迪機場 JFK	10:45
105.10.16	紐約賓州車站 Penn Station	13:20	斯堪那特提 Schenectady	16:23

2-2 受訓

105.10.17 ～ 105.10.21

西門子電力技術服務國際公司(SIEMENS PTI)動態分析短期課程-PSS/E 使用者自建模型撰寫

2-3 返程

表 2-2 返程行程表

日期	出發地點	出發時間	抵達地點	抵達時間
105.10.22	斯堪那特提 Schenectady	9:23	紐約 New York	12:45
105.10.23	紐約甘迺迪 JFK	12:45		
105.10.24			台北(桃園機場) TPE	16:35

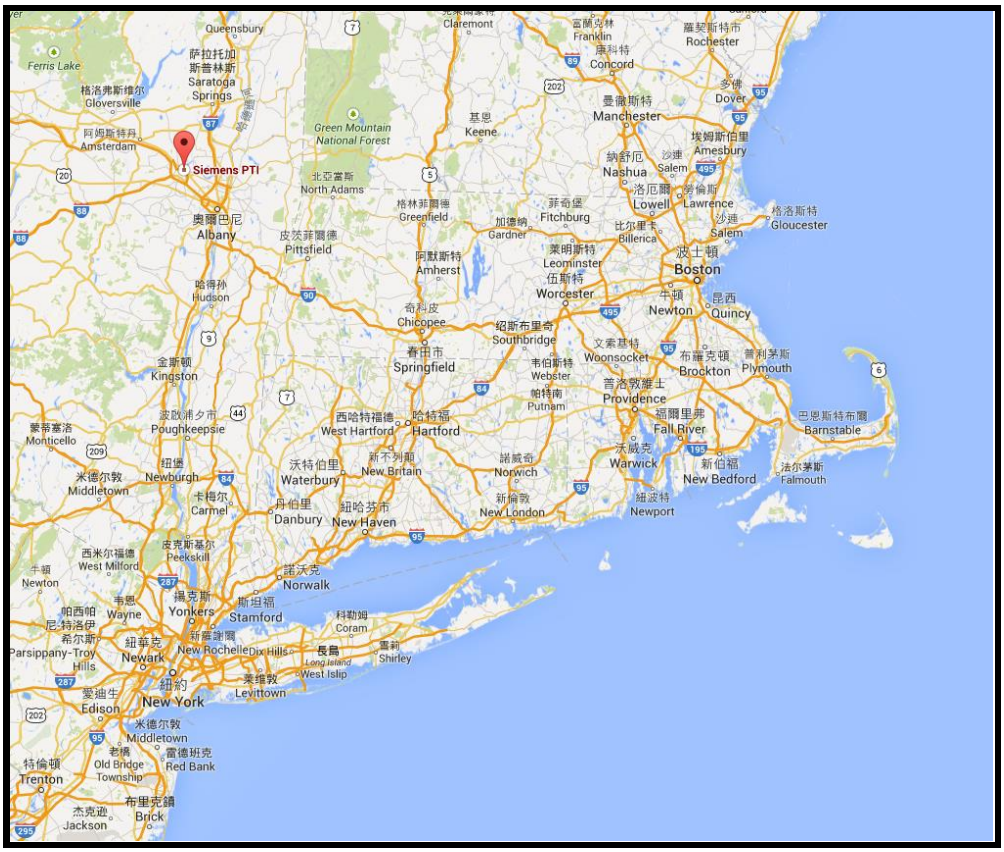


圖 2-1 美國紐約州斯堪那特提(SCHENECTADY)西門子公司地理位置

圖 2-2 美國紐約州斯堪那特提(SCHENECTADY)西門子公司

(Google 街景照片)

三、心得與建議

(一) 感謝各級長官給予此次赴美國西門子電力技術服務國際公司(SIEMENS PTI)動態分析短期課程學習機會，本次課程共有 7 位學員參加，其中學員僅 2 位(本人及 1 位韓電員工)為非美國籍，其餘學員皆在美國工作，包含電力公司、獨立調度中心、顧問公司及國家級研究單位研究人員，職務涵蓋規劃、運轉、發輸電及 ISO 準則等領域，學習期間除互相瞭解各公司電力系統架構、規模及發展方向，亦可透過 PSS/E 模擬問題探討有更多面向的思考。

圖 3-1 西門子公司用餐休息區一景

(二) 本次由於預算不足及西門子公司不同課程時間相隔數週等原因，僅參與西門子電力技術服務國際公司(SIEMENS PTI)動態分析一週短期課程。課程內容較屬於 PSS/E 進階動態模擬的部分，主題係撰寫 PSS/E 使用者自建模型，與其他課程相比，屬難度較高的課程，需具有 PSS/E 動態模擬經驗者較適合參加。來參加此課程之多數學員在其工作職務不需或從未撰寫使用者自建模型(據課程講師指出全世界僅有 5%之 PSS/E 使用者撰寫動態模型)，但藉由此課程的學習，幫助各學員瞭解 PSS/E 執行動態模擬時軟體內部資源的運作細節，增進動態模擬能力，亦深入了解動態模型之架構。

(三) 從台灣搭乘班機直飛紐約甘迺迪機場所需時間長達 15~16 小時，抵達紐約時身體已相當疲累，抵達美國機場後，辦理安檢及相關入境程序需花上 1~2 小時，若是班機飛抵美國時間為晚上，辦理完成入境程序時之時間將更晚。建議於訂購機票時選擇美國上午時間抵達，抵達後之路程交通若遇上狀況，較為容易且有足夠時間可處理應付，並且白天時間上路也較為安全。

(四) 受訓學員身處他鄉異國，同仁結伴受訓，無論是學習效果及互相照顧等方面都可互相幫忙，除了學習效果倍增外，對於美國高物價之生活花費(平均午、晚餐一頓平常餐點含稅及小費後價

格約 400 元~1000 元台幣不等)及住宿等費用，亦可互相分擔，減輕出國個人經費負擔。

(五) 美國火車容易誤點，由於美國境內幅員廣大，火車單程總小時數可能長達 20 小時左右行程，既使短程交通可能也容易受影響，建議若有其他公車(如灰狗巴士)路線可到達之處，除假日前後可能受到塞車因素耽擱時程，在總搭乘時間不長的狀況下，不失為一個理想的交通工具。

(六) 有智慧型手機之同仁，建議可於出發前在台灣先行辦理美國當地易付卡（T-Mobile），並選擇合適之通話即上網費率，因為抵達美國之後即須處理交通及住宿等生活瑣事，此外迷路或有急事時須與當地或台灣人員聯絡時，皆甚為便利，並可降低在國外生活之不安全感。

四、PSS/E 動態模擬介紹

4-1 PSS/E 軟體發展歷程

PSS/E (Power System Simulator for Engineering)軟體為美國電力技術公司(Power Technologies International ,PTI，已於 2005 年被西門子(Siemens)公司併購)於 1976 年開發之電力系統模擬軟體，四十年期間，PSS/E 軟體隨者電力系統技術的演進不斷持續更新及升級。PSS/E 具有強大的計算能力，PSS/E 是電力系統模擬軟體中，最廣泛為世界各電力公司及電力相關研究單位所使用之電力系統分析軟體之一，目前約有 115 個國家、超過 600 家不同的公司及組織使用 PSS/E 軟體進行電力系統模擬分析。

PSS/E 軟體在技術領域上提供許多先進及成熟的方法，以下簡述幾種主要功能：

- ✓ 電力潮流及優化(Optimal)電力潮流
- ✓ 平衡或不平衡故障分析
- ✓ 動態模擬
- ✓ 負載模型
- ✓ 傳輸限制分析
- ✓ 彈性交流輸電系統(FACTS)特性模擬

✓ PV 及 QV 分析

PSS/E 除了內建常用的大型發電系統之發電機、激磁機、調速機及電力系統穩定器(PSS)模型外，近年來國內外再生能源(太陽能、風力發電)裝置容量遽增，西門子公司亦針對太陽能及風力機組動態參數建立許多模型，可直接於 PSS/E 所建置之電力系統中模擬分析。

本課程使用 PSS/E v.33，此版本於 2011 年 5 月至 2013 年 10 月間發行，期間經歷數次小規模的升級，PSS/E v.33 在模擬事故分析上增加了許多功能，亦簡化部分模擬程序，諸如：

✓ 故障分析可使用多元處理

✓ 新式 N-1-1 偶發事故解

✓ 設定匯流排於正常時與故障時刻之電壓限制

✓ 敏感性分析

✓ 安全約束下之優化電力潮流（SCOPF）

將動態 CONEC 和 CONET 模型標準化

4-2 PSS/E 動態模擬之原理與應用

PSS/E 具有電力系統動態模擬分析之功能，可模擬電力系統發生偶發事故時，計算系統對此事故之響應。偶發事故可包含線路或

匯流排事故、線路跳線、發電機跳機、卸載及馬達啟動等。舉簡單之電阻-電感(R-L)串聯電路為例，如圖 4-1 所示，當電路中開關於某一時間點投入(ON)時，R-L 串聯電路之電流值 i 將隨時間動態增加至穩態電流值。為描述電流動態行為，使用一階微分方程式表示

$$E = Ri + L\frac{di}{dt}$$

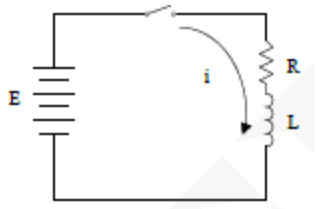


圖 4-1 電阻-電感串聯電路(課程講義提供)

將此方程式透過拉普拉斯轉換(Laplace Transform)，將電路從時域(time-domain)轉換至頻域(s-domain)，可整理成下圖 4-2 之轉移方程式(Transfer Function)，轉移方程式乃以輸出-輸入比型式呈現，其中輸出為電流 i(t)，輸入為電壓源 E(t)。R-L 串聯電路之電流 i(t)動態響應結果如圖 4-3 所示

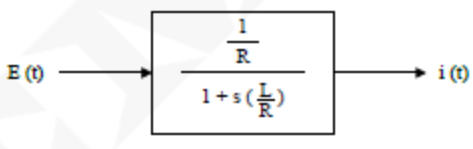


圖 4-2 電阻-電感串聯電路動態方塊圖(課程講義提供)

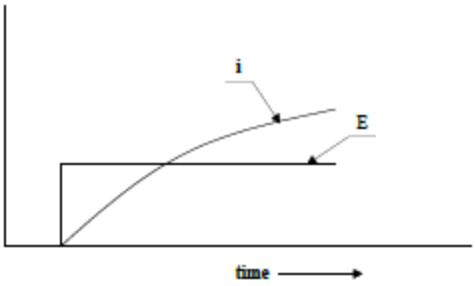


圖 4-3 電阻-電感串聯電路動態電流響應(課程講義提供)

PSS/E 動態模擬分析使用之機組動態模型即以此形式之方塊圖建構。電力系統元件如發電機、激磁機、調速機、穩定器、變壓器、彈性交流輸電設備等皆可以微分方程等數學式描述，並加以轉換成模型方塊圖，搭配 PSS/E 解潮流程式計算得到系統初始狀態，即可分析整體電力系統遇偶發事故時，計算系統之動態響應。

分析電力系統各類事故或用途，依其物理特性皆有不同之觀察時間範圍，進而所使用之分析工具及模型亦有所不同。若要執行系統暫態穩定度模擬，分析電力系統發生事故時，系統中發電機轉子角因發電機組與電力系統間之動能轉換變動之關係而有所變動，進而發電機之出力亦隨之變動。發電機遇偶發事故時，搖擺(Power Swing)現象之時間範圍依其物理特性約為 1 週波~10 秒間，故選擇動態模擬時間為 10 秒。電力系統各類型事故及用途之時間範圍比較如圖 4-4 所示。

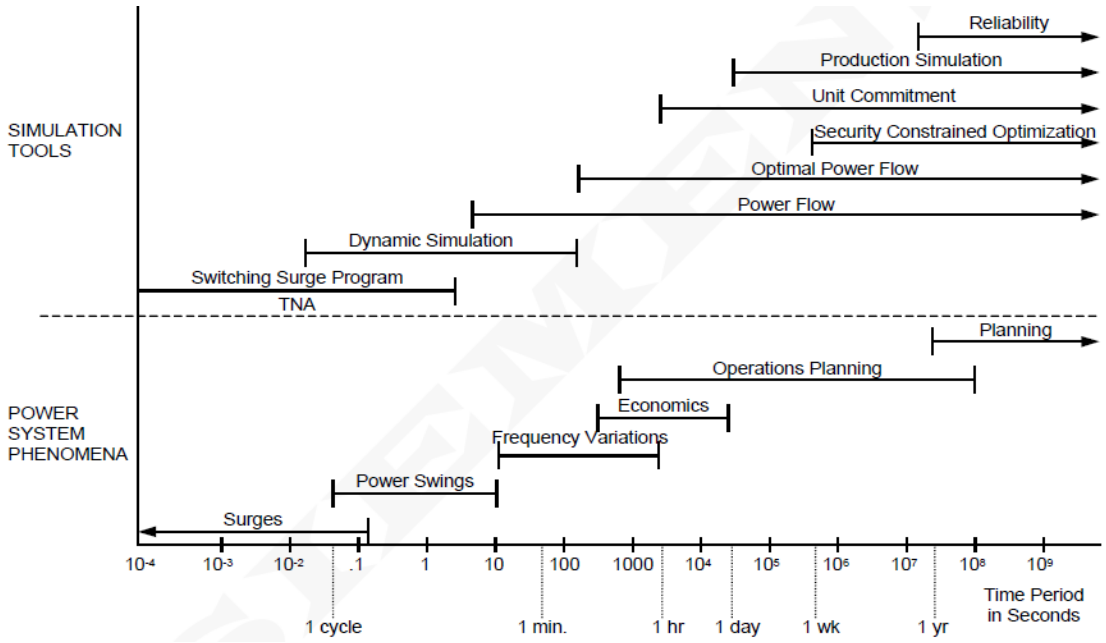


圖 4-4 電力系統各類事故及用途之時間範圍比較(課程講義提供)

4-3 PSS/E 動態模擬流程

電力系統動態模擬分析使用之電力系統網路求解(Network Solution)及動態模型注入電流(Current Injections)間相互關係可使用電路學(Circuit Theory)之結點方程式(Node Equation) Y • V = i(x,v)說明，其中結點方程式中電流 i(x,v)，其可代表機組設備模型(In Service)注入結點之電流，受到狀態變數 x 及結點(匯流排)電壓 v 影響。使用求解電力潮流程式對進行電力系統網路進行疊代(Iteration)運算，求得電力系統中各結點的電壓值作為初始電壓值，假設已知狀態變數初始值 x，如此可計算得流進結點的電流 i，將電流 i 代入結

點方程式 Y・V = i(x,v)，經過數值計算，可求得結點之新電壓值；將此新電壓值代入 i(x,v)，即可求得新的電流值。

執行電力系統暫態穩定度模擬時，依其物理特性，由於頻率響應的關係，系統動態響應過程中，輸電網路參數皆訂為常數值(正序值)，而不考慮其動態響應。若需分析更高頻率之動態響應，如突波分析，需考慮輸電線路之動態效應(特別是電容效應)，則須使用電磁暫態分析軟體 EMTP 來分析。

PSS/E 動態模擬流程，如圖 4-5 所示，模擬流程主要包含初始化(Initialization)及 RUN 模擬程序，詳細流程分別如圖 4-6 及 4-7 所示。動態模擬包含四大重點:

1. 機組模型狀態變數初始化。
2. 機組模型注入電流計算: 機組模型注入電流主要依網路求解得到之匯流排電壓計算而得。使用之設備模型只要有匯流排電壓與設備流入電流之代數關係之模型(Current-Injection Model)，皆需計算注入電流。這些模型設備包含:發電機、感應電動機、SVC、HVDC 及負載等。
3. 狀態變數時間導數計算(DSTATE): 在給定所有機組模型之狀態變數初始值及發電機定子電流情況下，計算模型內部每一狀態變數之時間導數(Time Derivative)。此階段在計算獲得狀態變數

時間導數值期間，亦計算模型中所有代數變數(Algebraic Variables)。

4. 計算模型輸出變數(Output Variables)。

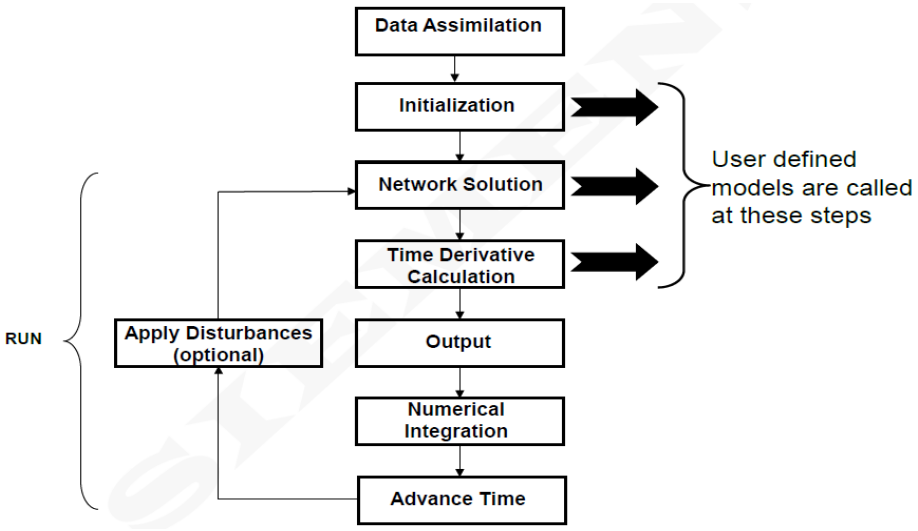


圖 4-5 PSS/E 動態模擬流程圖(課程講義提供)

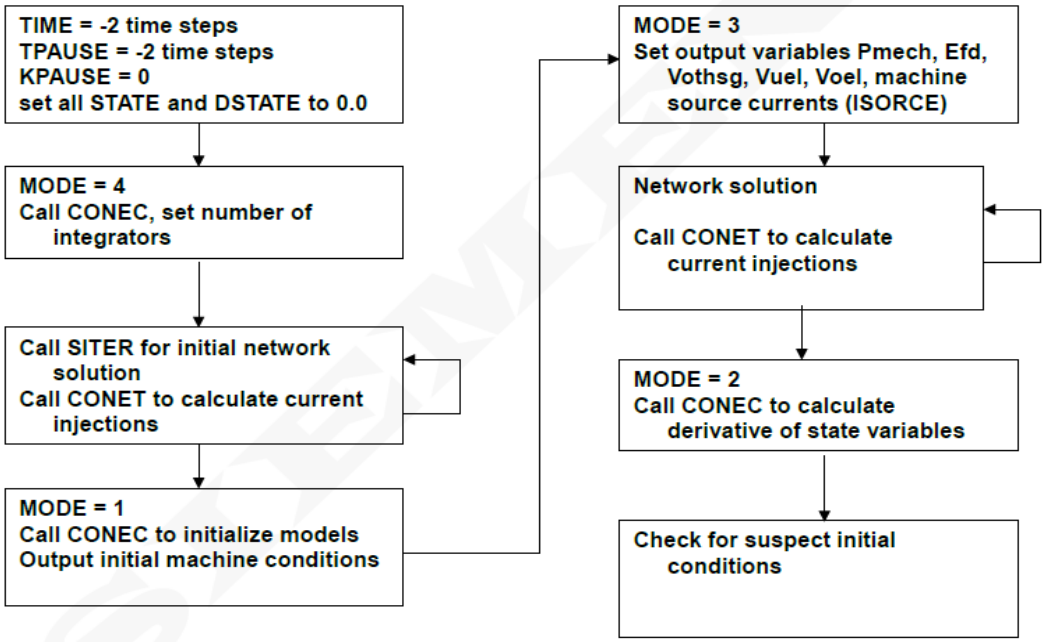


圖 4-6 初始化流程圖(課程講義提供)

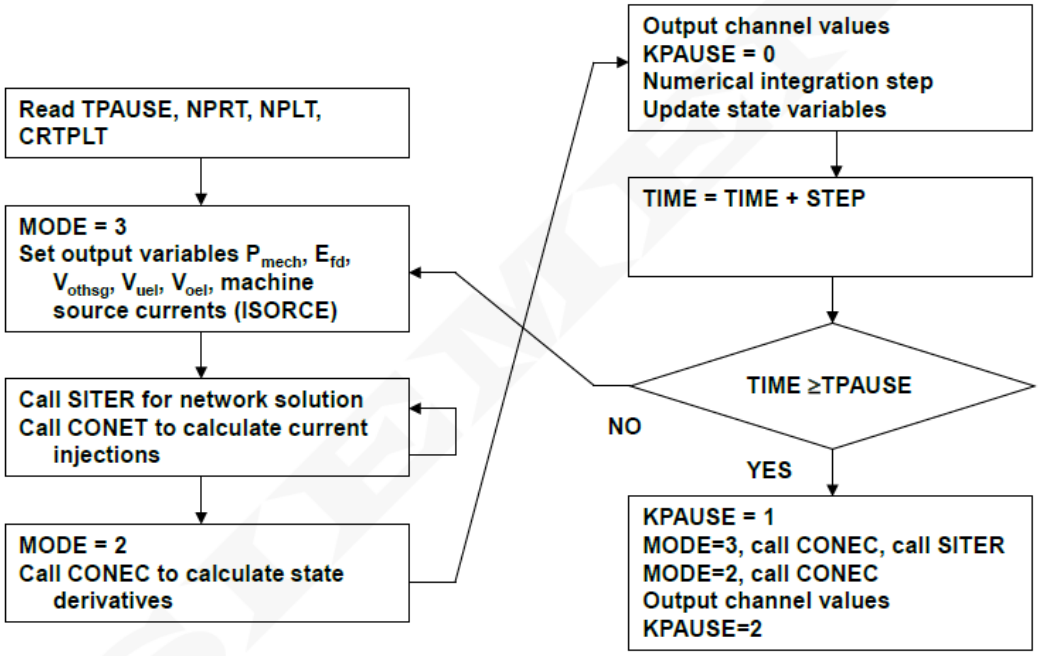


圖 4-7 RUN 模擬程序流程圖(課程講義提供)

4-4 連結副程式 CONEC 及 CONET

選定參與動模擬之設備模型後，使用者必須提供連結副程式(Connection Subroutines)，將這些模型連接於電網網路中，連結關係示意圖如圖 4-8 所示。

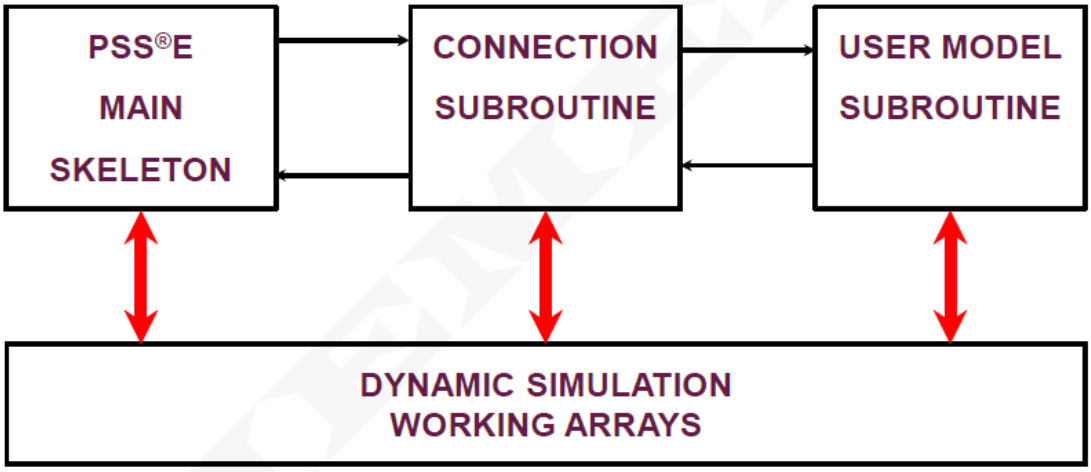


圖 4-8 PSS/E 動態模擬之基本程式架構(課程講義提供)

連結副程式(Connection Subroutines)有 CONEC 及 CONET，副程式 CONEC 主要功能為將包含狀態變數及微分方程式之設備模型(如激磁機模型)與 PSS/E 主程式連接。副程式 CONEC 主要負責計算模型中所有狀態變數之時間導數(存於陣列 DSTATE)；亦可用於模擬之進階控制。另外，副程式 CONET 於求解網路(指令 SITER)過程中主要呼叫(Call)設備模型來計算模型注入電流(Current Injections)，其計算過程中須根據設備模型連接之匯流排電壓值；亦可用於僅監視網路狀態或根據網路狀態值執行之設備模型(如 Relay 模型)。

在 31 版 PSS/E 以前，副程式 CONEC 中包含呼叫自建模型之呼叫程式碼描述(CALL Statements)，但在 31 版(含)及之後的版本則不需自建模型呼叫之描述。PSS/E 32 版(含)及更早版本，副程式 CONET 中包含模型注入電流計算及網路狀態監視(如電壓、頻率等)等模型呼叫描述；但從 33 版開始，副程式 CONET 中則無任何相關之模型呼叫敘述。PSS/E 33 版(含)之後之模型呼叫工作直接由 PSS/E 主程式內部(Table-Driven Form)執行，而模型不再透過副程式 CONEC 及 CONET 呼叫，因此使用者可選擇是否需要使用副程式 CONEC 及 CONET，連結關係示意圖如圖 4-9 所示。

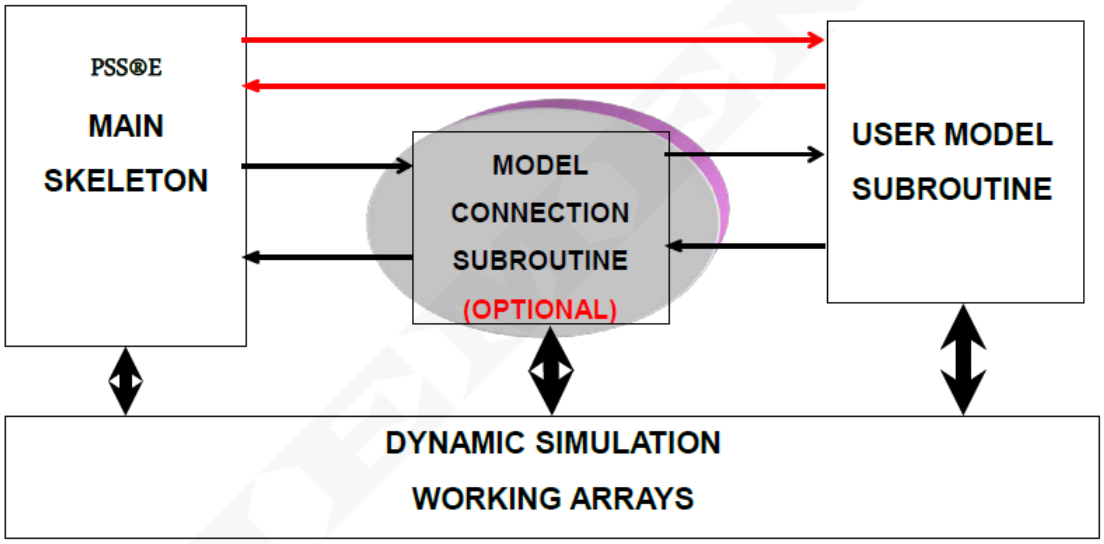


圖 4-9 PSS/E 主程式及自建模型連接關係圖(課程講義提供)

4-5 PSS/E 模擬資料類型

使用於 PSS/E 動態模擬之資料類型(Data types)可分為 4 類:

1. **常數**:於模擬中，數值不會變動的參數。
2. **狀態變數**:以微分方程式決定之瞬時值的變數。
3. **代數變數**:若已知所有狀態變數值與常數值，任意模擬時間點下可被決定的變數。
4. **輸入變數**:任何模擬時間點下可受動態模擬之邏輯外部(logic outside)指定的變數。

此外，PSS/E 有 4 類動態模擬儲存陣列(Dynamic Simulation Arrays):

1. **CON**:包含模型使用之各常數；
2. **STATE**:包含模型使用之各狀態變數；

3. **VAR**:包含模型使用之各代數變數；

4. **ICON**:包含模型使用之整數值，可能為常數或代數變數。

PSS/E 所使用之動態模擬陣列如表 4-1~表 4-4 所示。

表 4-1 動態模擬陣列-常數及狀態變數(課程講義提供)

CONSTANTS

Array	Contents	Indexed by
CON	General constants (real)	CON number
ICON	General constants (integer)	ICON number
CHRICN	General constants (character)	ICON number
MBASE	Machine base MVA	machine index
ZSORCE	Machine impedance (complex)	machine index
XTRAN	Step-up transformer impedance (complex)	machine index
GENTAP	Step up transformer tap ratio	machine index

STATE VARIABLES

Array	Contents	Indexed by
STATE	General state variable array	STATE number

表 4-2 動態模擬陣列-代數變數(課程講義提供)

ALGEBRAIC VARIABLES

Array	Contents	Indexed by
VAR	General algebraic variable array	VAR number
VOLT	Bus p.u. voltages (complex)	bus sequence number
CURNT	Network inflow current (p.u. on SBASE)(complex)	bus sequence number
BSFREQ	Bus p.u. frequency deviations	bus sequence number
ANGLE	Machine relative rotor angle (degrees)	machine index
PELEC	Machine electrical power (p.u. on SBASE)	machine index
QELEC	Machine reactive power (p.u. on SBASE)	machine index
ETERM	Machine terminal voltage (p.u.)	machine index
EFD	Generator main field voltage (p.u.)	machine index
PMECH	Turbine mechanical power (p.u. on MBASE)	machine index
SPEED	Machine speed deviation from nominal (p.u.)	machine index
XADIFD	Machine field current (p.u.)	machine index
ECOMP	Voltage regulator compensated voltage (p.u.)	machine index
VOTHSG	Stabilizer output signal (p.u.)	machine index
VUEL	Minimum excitation limiter output signal (p.u.)	machine index
VOEL	Maximum excitation limiter output signal (p.u.)	machine index
ISORCE	Machine current (p.u. on SBASE) (complex)	machine index
TPLOAD	Effective MW load (pu on SBASE)	load index
QPLOAD	Effective Mvar load (pu on SBASE)	load index

表 4-3 動態模擬陣列-輸入變數(課程講義提供)

INPUT VARIABLES

Array	Contents	Indexed by
VREF	Voltage regulator voltage setpoint (p.u.)	machine index
SETVAL	DC transmission power/current setpoint (MW/Amps)	DC line number

表 4-4 動態模擬陣列-內部變數(課程講義提供)

INTERNAL ARRAYS

Array	Contents	Indexed by
DSTATE	General state variable time derivatives (STATE space)	STATE number
STORE	General state variable integrator memory	STATE number
STORMT	General memory (extended term)	2*STATE number-1 and 2*STATE number
BSFMEM	Memory for frequency calculation	bus sequence number
STRTIN	Starting array indices for plant related models	array allocation table index
NUMTRM	Pointer to bus sequence number	machine index
NUMBUS	External bus number	bus sequence number
MACHID	Machine identifier	machine index
NUMLOD	Pointer to bus sequence number	load index
LOADID	Load identifier	load index
INTICN	Integer memory array	ICON number

4-6 PSS/E 動態模擬控制旗標

PSS/E 執行動態模擬時，PSS/E 運用之記憶體中，有部分純量變數格式之空間用於作為控制旗標(Control Flags)。控制旗標為PSS/E 動態模擬指令(Activities)與設備模型間之溝通橋樑。在呼叫機組模型前，PSS/E 指令(包括 STRT、RUN、MSTR 及 MRUN)即設定各旗標值，以決定需執行動態模型程式之區塊及執行狀態。旗標 MODE、MIDTRM、IBDOCU、IFLAG 及 KPAUSE 等變數之功能與用途簡述如下:

1. MODE (MODE = 1 ~ MODE = 8)

(1) **MODE = 1:**在電力系統網路為穩態之假設下，計算所有狀態變數之初始值。

(2) **MODE = 2:**計算所有狀態變數對時間的微分，且將結果放在 DSTATE 陣列內，所有穩定器的輸出值必需計算出來，並且將其放在 VOTHSG 陣列內；每一個激磁機的極小值及極大值限制器的輸出值必須計算出來，並分別放置在 VUEL 及 VOEL 陣列內。

(3) **MODE = 3:**設定模型輸出值並放置於相應的陣列。若為穩定器，則將其輸出值計算出來，並且放在 VOTHSG 陣列內；若模型為激磁機時必須計算出其輸出電壓標么值，並將其結果放在 EFD 陣列內，且每一個激磁機的極小值及極大值限制器的輸出值必須計算出來，並分別放置在 VUEL 及 VOEL 陣列內；若為汽輪機之調速器模型，則必須計算出其輸出的機械功率，並將結果放在 PMECH 陣列內，若為其它的模型就不需要此模式。

(4) **MODE = 4:**模型必需更新最大被使用到的狀態變數值，放入變數 NINTEG 內。

(5) **MODE = 5:**當執行 DOCU 列表指令時，會呼叫此模式，在此模式下，必須將此模型的一些參數列印出來，DOCU 指令會

將所選擇機組的全部模型參數列出，列出內容包含模型名稱、所使用之狀態變數及其陣列編號範圍、常數參數陣列編號範圍，以及各模型所用到的參數名稱及其內含值。

(6) **MODE = 6:**當執行 DYDA 指令時，會呼叫此模式，並且列出此模型的一些參數記錄，DYDA 指令會將所選擇機組的全部模型參數列出，列出內容包含模型名稱，以及各模型所用到模型的常數參數值。

(7) **MODE = 7:**當執行 DOCU 的檢查指令時，會呼叫此模式，在此模式下，必需檢查所有常數參數值，是否超出定義的範圍，若有超出時，則會將結果列於各模型的最上方。

(8) **MODE = 8:**當執行 Add/Edit Constants 指令時，會呼叫此模式，方便讓使用者能夠直接在動態模擬前，以列表方式修改常數參數值，而不必從機械參數檔案中修改。

2. MIDTRM

邏輯變數 MIDTRM 於執行模型程式 MODE1~MODE4 中具有意義，並指出為執行 State-space 模擬或 Extended term 模擬:

(1) MIDTRM=.FALSE 為 State-space 模擬

(2) MIDTRM=.TRUE 為 Extended term 模擬

3. IBDOCU

變數 IBDOCU 僅於執行模型程式 MODE5~MODE7 中具有意義，並指出指令 DOCU 或 DYDA 之運作之模式(MODE):

(1) IBDOCU=0 處理所有受呼叫之模型。

(2) IBDOCU>0 外部匯流排號碼;只處理在匯流排 IBDOCU 上受呼叫之模型。只有受副程式 CONEC 或 CONET 呼叫之模型需要偵測 IBDOCU 值。

4. IFLAG

變數 IFLAG 值可指出網路解(Network Solution)是否收斂。

(1) IFLAG=.FALSE 為網路求解尚未完成收斂。

(2) IFLAG=.TRUE 為網路求解已收斂，或達到其疊代最大次數。

5. KPAUSE

變數 KPAUSE 用於指出模擬切換時間點。

(1) KPAUSE=0 代表模型於正常模擬時間(Normal Time Step)時(Time=t)受呼叫。

(2) KPAUSE=1 為代表模型於時前切換時間(Pre-switching Time Step)時(Time=t^-)受呼叫。

(3) KPAUSE=2 為代表模型於暫停時間點後之第一個時間點(Time=t^+)受呼叫。

4-7 數值積分方法

為執行動態模擬時求解機組模型之微分方程式，以得到機組模型之動態行為，PSS/E 使用之數值積分(Numerical Integration)法為修改型尤拉數值積分法(Modified Euler Method)求解。模擬設定時，須將動態模擬時間間隔值(Time Step)設定為小於全系統機組模型之最小時間常數(Smallest Time Constant)，通常建議將時間間隔值設定為最小時間常數之一半長度，以避免求解時發生數值不穩定(Numerical Instability)的現象。

五、使用者自建模型程式撰寫

5-1 使用者模型程式

5-1-1 使用者模型撰寫需求

為撰寫使用者自建模型(User-Defined Model)程式，PSS/E 使用者必須:

1. 有欲撰寫模型之模型方塊圖(Block Diagram)與微分及代數方程式。
2. 有將描述模型之數學方程式轉換成 PSS/E 模型可接受之模型方塊圖格式之能力。
3. 熟悉 PSS/E 動態模擬流程、動態模擬資料陣列結構(Dynamic Simulation Data Arrays)及控制旗標(Control Flags)。
4. 熟悉 FORTRAN 或 FLECS 等程式語言。

5-1-2 使用者模型動態檔格式

執行 PSS/E 動態模擬時，會使用指令 DYRE，從動態模型原始資料檔(Dynamics Model Raw Data File (*.dyr))讀取動態模型資料(Model Data)並存入 PSS/E 工作記憶體之動態資料陣列中。

動態模型原始資料檔(*.dyr)包含許多組具邏輯結構之動態模型參數記載(logical records)，每一個模型參數記載內容包括 PSS/E 模型資料庫(PSS/E Model Library)中動態設備模型在電網網路中的位置(藉由匯流排、機組、負載、dc line 等)，以及動態模型之常數等參數。

動態模型原始資料檔(*.dyr)中除包括 PSS/E 模型資料庫中設備動態模型之動態模型參數記載外，亦包含使用者自撰模型(user-written models)之動態模型參數記載，其格式如下:

BUSID 'USRMDL' IM 'model name' IC IT NI NC NS NV data list /

其中

1. **model name** 與動態模型副程式名稱相同，最多使用 16 個字母。
2. **IC** 為使用者模型之類型碼(type code)，指出此為何種模型:

   IC = 0 表示從副程式 CONET 呼叫而不從副程式 CONEC 呼叫之模型；1 表示發電機模型；2 表示電流補償器模型；3 表示穩定器模型；4 表示激磁系統模型；5 表示渦輪調速機模型；7 表示雙終端 dc line 模型；8 表示其它副程式 CONEC 模型；9 表示最小激磁限制器模型；10 表示最大激磁限制器模型；其餘 11、12、13、14、17、18、19、20、21、23、24、101、102、103、104、105、106、107 所代表之模型請參考 34 版 PSS/E Manual 第 15 章。

3. **IT** 為受 CONEC 及/或 CONET 呼叫之使用者模型(當 IC 值為 0、7、8)，IT 值可以為 0、1 或 2，其中
   - IT = 0 代表非電流注入(Non-Current Injection)或量測(metering)模型；
   - IT = 1 代表電流注入(Current Injection)模型，在副程式 CONET 中，模型呼叫敘述(Model CALL Statement)放置於旗標 IFLAG 值(指出網路求解是否為收斂，使用 IF Statement 測試)測試程式碼之前；
   - IT = 2 代表為量測模型，在副程式 CONET 中，模型呼叫敘述放置於旗標 IFLAG 值測試程式碼之後。
4. **NI** 為模型中使用 ICONs 的數量(NI≦500)。
5. **NC** 為模型中使用 CONs 的數量(NC≦1000)。
6. **NS** 為模型中使用 STATEs 的數量。
7. **NV** 為模型中使用 VARs 的數量。
8. **NRI** 為模型中使用額外(保留)之 ICONs 的數量。
9. **Data list** 為 NI 個 ICONs，後接續為 NC 個 CONs。

在撰寫動態模型程式前，必須先清楚定義該模型使用之常數、狀態變數、代數變數等數量，並先行製作撰寫模型之資料表

(Datasheet)。相反地，若欲使用別人寫好的使用者自建模型，則須先得到該模型之資料表,以了解動態模型原始資料檔之資料值設定。

5-1-3 使用者模型程式碼雛型

PSS/E 使用者模型係以副程式(Subroutine)之型式撰寫，撰寫完成後，將此模型副程式透過編譯(Compile)及連結(Link)等程序，轉換成 PSS/E(主程式)可直接使用之檔案(.DLL)。撰寫模型程式碼時，程式碼敘述之必要項目，形成程式碼雛型(Code Template)，項目包括:

1. 副程式名稱敘述(Subroutine Statement)
2. 副程式引數(Subroutine Arguments)
3. 模式(MODE 1~MODE 8)判斷之 IF-THEN-ELSE 敘述
4. PSS/E 變數使用指令 COMON.INS 敘述
5. IMPLICIT NONE 敘述

在深入撰寫程式碼前，先完成程式碼雛型，再針對程式碼中每一模式(MODE)填入相對應功能之程式碼,將使程式撰寫更有效率。

5-1-4 程式起始位址陣列索引

執行動態模擬時，PSS/E 主程式必須得到各動態模型使用之各類動態模擬陣列(CON、STATE、VAR 及 ICON)之記憶體起始位址，

起始位址乃透過起始位址陣列索引(Starting Array Index)程式得到。執行指令 DYRE 時，每一使用中動態模型之動態模擬陣列配置表登記(Array Allocation Table Entries)將藉由下列程式設定:

1. 若為電廠相關模型(Plant-Related Models)，則
   - STRTIN(1,ISLOT):動態模型使用 NC 個 CONs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - STRTIN(2,ISLOT):動態模型使用 NS 個 STATEs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - STRTIN(3,ISLOT):動態模型使用 NV 個 VARs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - STRTIN(4,ISLOT):動態模型使用 NI 個 ICONs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
2. 若為負載相關模型(Load-Related Models)，則
   - LDSTRT(1,ISLOT):動態模型使用 NC 個 CONs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - LDSTRT (2,ISLOT):動態模型使用 NI 個 ICONs 中第一筆索引值，若 NI 為 0，則此程式回傳值為 0。
   - LDSTR2(1,ISLOT2):動態模型使用 NS 個 STATEs 中第一筆索引值，若 NS 為 0，則此程式回傳值為 0。

- LDSTR2(2,ISLOT2):動態模型使用 NV 個 VARs 中第一筆索引值，若 NV 為 0，則此程式回傳值為 0。

3. 若為電驛模型(Line Relay Models)，則
   - RLSTRT(1,ISLOT):動態模型使用 NC 個 CONs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - RLSTRT(2,ISLOT):動態模型使用 NS 個 STATEs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - RLSTRT(3,ISLOT):動態模型使用 NV 個 VARs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - RLSTRT(4,ISLOT):動態模型使用 NI 個 ICONs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
4. 若為風機相關模型(Wind-Related Models)，則
   - WSTRTIN(1,ISLOT):動態模型使用 NC 個 CONs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - WSTRTIN(2,ISLOT):動態模型使用 NS 個 STATEs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - WSTRTIN(3,ISLOT):動態模型使用 NV 個 VARs 中第一筆索引值，若 NC 為 0，則此程式回傳值為 0。
   - WSTRTIN(4,ISLOT):動態模型使用 NI 個 ICONs 中第一筆索

引值，若 NC 為 0，則此程式回傳值為 0。

5-1-5 轉移方程式及基本控制方塊

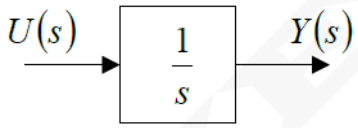
動態模型係由微分方程式描述其狀態變數值動態行為之數學模型，將這些描述設備動態行為之方程式透過拉式轉換，將方程式轉換至 s-domain，並經整理成所定義輸出-輸入比之形式，形成轉移方程式(Transfer Function)。轉移方程式可透過控制方塊圖方式呈現，將各方塊圖按數學方程式之關係連接，形成完整之設備數學模型。

方塊圖依運算特性的不同，可分為積分器(Integrator)、一階落後(First order Lag)、Washout、領先-落後(Lead-Lag)、比例-積分(Proportional Integral (PI))、比例-積分-微分(Proportional Integral Derivative (PID))、二階(Second Order block)、輸出端上下限(Windup Limit)、狀態變數上下限(Non-Windup Limit)等基本方塊，以下分別列出各別方塊圖之 MODE1 及 MODE2 程式碼。再來，為簡化 MODE1~MODE3 程式撰寫之複雜度及考慮一致性，使用 PSS/E 內建之基本方塊程式(Elementary Block Functions)(參考 34 版 PSS/E Program Application Guide Volume 2 第 24 章)執行數學模型中各方塊圖之狀態變數初始化、狀態變數微分值及輸出值等計算。若須從 PSS/E 程式庫(Function Library)引用基本方塊程式，則須在模型程式中撰寫 INCLUDE‘COMON4.INS’描述。

1. 積分器(INTEGRATOR)

積分器 MODE 1、MODE 2 程式碼及基本方塊程式碼(及其變數宣告)，如表 5-1 及表 5-2 所示。

表 5-1 積分器方塊圖程式碼

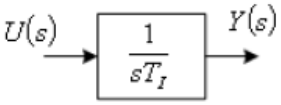
<table>
<tr><th colspan="2">積分器方塊圖程式碼</th></tr>
<tr><th>MODE1(初始化)</th><th>MODE2(模擬)</th></tr>
<tr><td>DSTATE = 0
STATE = Y</td><td>DSTATE = U
Y = STATE</td></tr>
<tr><td colspan="2"> 

(課程講義提供)</td></tr>
</table>

表 5-2 積分器基本方塊程式

<table>
<tr><th colspan="2">積分器基本方塊程式(整理自 PSS/E 使用手冊)</th></tr>
<tr><td>MODE1
(初始化)</td><td><pre>
 INTEGER IERR
 REAL VINP
 REAL VOUT

C_
C____ UNLIMITED INTEGRATOR (INITIALIZED WITH OUTPUT = Y)
C_
        VOUT = Y   ! block output
        VINP = INT_MODE1(TI,   !   time constant (real)
     #   VOUT,   !   block output variable (real)
     #   K   ,   !   index for state variable (integer)
     #   IERR)   !   error code (integer)

C IERR = 0 => no error
C IERR = 4 => TI is zero
</pre></td></tr>
<tr><td>MODE2
(模擬)</td><td><pre>
C_
C____ UNLIMITED INTEGRATOR
C_
        VINP = U
        VOUT = INT_MODE2(TI,   !   time constant (real)
     #   VINP,   !   block input variable (real)
     #   K)   !   index for state variable (integer)
</pre></td></tr>
<tr><td>MODE3
(輸出)</td><td><pre>
C_
C____ UNLIMITED INTEGRATOR
C_
        VINP = U
        VOUT = INT_MODE3(TI,   ! time constant (real)
     #   VINP,   ! block input variable (real)
     #   K)   ! index for state variable (integer)
</pre></td></tr>
<tr><td colspan="2"> 
</td></tr>
</table>

2. 一階落後(First Order Lag)

一階落後 MODE1、MODE2 程式碼及基本方塊程式碼(及其變數宣告)，如表 5-3 及表 5-4 所示。

表 5-3 一階落後方塊圖程式碼

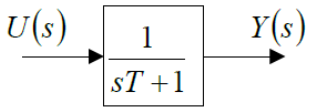
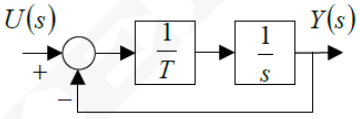
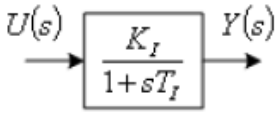
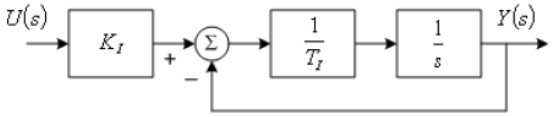
一階落後方塊圖程式碼	
MODE1(初始化)	**MODE2(模擬)**
DSTATE = 0 STATE = Y U = Y	DSTATE = (U-STATE)/T Y = STATE
  (課程講義提供)	

表 5-4 一階落後基本方塊程式

一階落後基本方塊程式(整理自 PSS/E 使用手冊)	
MODE1 (初始化)	INTEGER IERR REAL VINP REAL VOUT C_ C____ UNLIMITED FIRST ORDER BLOCK (INITIALIZED WITH OUTPUT = Y) C_ VOUT = Y VINP = LAG_MODE1(KI, ! gain (real) # TI, ! time constant (real) # VOUT, ! block output variable (real) # K, ! index of state variable (integer) # IERR) ! error code (integer) C C IERR = 0 => no error C IERR = 3 => Gain KI = 0 (fatal error at initialization) C IERR = 4 => TI = 0
MODE2 (模擬)	C_ C____ UNLIMITED FIRST ORDER LAG C_ VINP = U VOUT = LAG_MODE2(KI, ! gain (real) # TI, ! time constant (real) # VINP, ! block input variable (real) # K) ! index of state variable (integer)
MODE3 (輸出)	C_ C____ UNLIMITED FIRST ORDER LAG C_ VINP = U VOUT = LAG_MODE3(KI, ! gain (real) # TI, ! time constant (real) # VINP, ! block input variable (real) # K) ! index of state variable (integer)
  	

3. Washout

Washout MODE1、MODE2 程式碼及基本方塊程式碼(及其變數宣告)，如表 5-5 及表 5-6 所示。

表 5-5 Washout 方塊圖程式碼

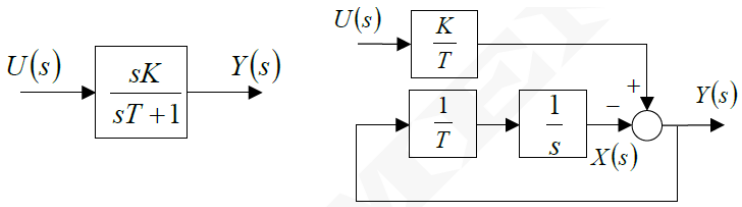
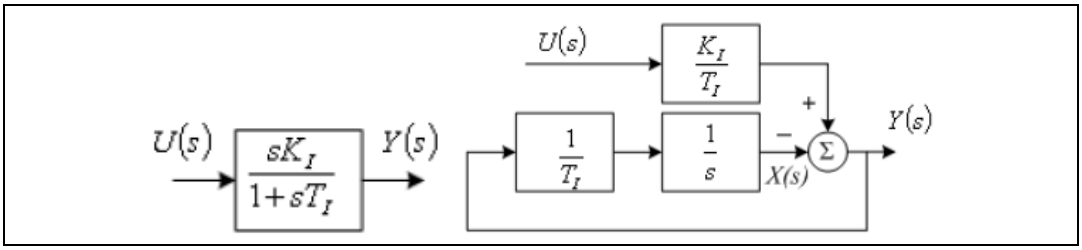
Washout 方塊圖程式碼	
MODE1(初始化)	**MODE2(模擬)**
DSTATE = 0 STATE = K*U/T Y = 0	Y = K*U/T-STATE DSTATE = (K*U/T-STATE)/T DSTATE = Y/T
 (課程講義提供)	

表 5-6 Washout 基本方塊程式

Washout 基本方塊程式(整理自 PSS/E 使用手冊)	
MODE1 **(初始化)**	INTEGER IERR REAL VINP REAL VOUT C_ C____ WASHOUT BLOCK C_ VINP = U VOUT = WSHOUT_MODE1(KI, ! gain (real), # TI, ! time constant (real) # VINP, ! block input variable (real) # K, ! index of state variable (integer) # IERR) ! error code (integer) C C IERR = 0 => no error C IERR = 4 => TI = 0
MODE2 **(模擬)**	C_ C____ WASH-OUT C_ VINP = U VOUT = WSHOUT_MODE2(KI, ! gain (real) # TI, ! time constant (real) # VINP, ! block input variable (real) # K) ! index of state variable (integer)
MODE3 **(輸出)**	C_ C____ WASH-OUT C_ VINP = U VOUT = WSHOUT_MODE3(KI, ! gain (real) # TI, ! time constant (real) # VINP, ! block input variable (real) # K) ! index of state variable (integer)



4. 領先-落後(Lead-Lag)

領先-落後 MODE 1、MODE 2 程式碼及基本方塊程式碼(及其變數宣告)，如表 5-7 及表 5-8 所示。

表 5-7 領先-落後方塊圖程式碼

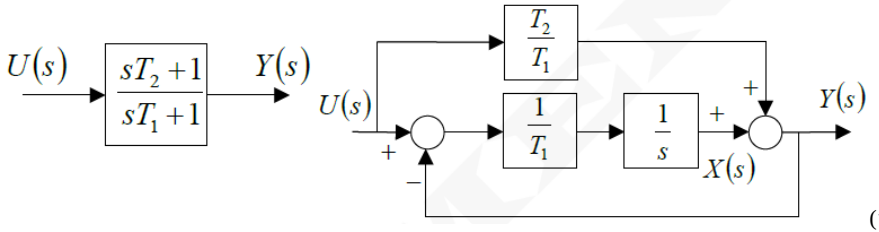
領先-落後方塊圖程式碼	
MODE1(初始化)	**MODE2(模擬)**
DSTATE = 0 U = Y STATE = (1-T2/T1)*Y	Y = STATE+T2*U/T1 DSTATE = (U-Y)/T1
 (課程講義提供)	

表 5-8 領先-落後基本方塊程式

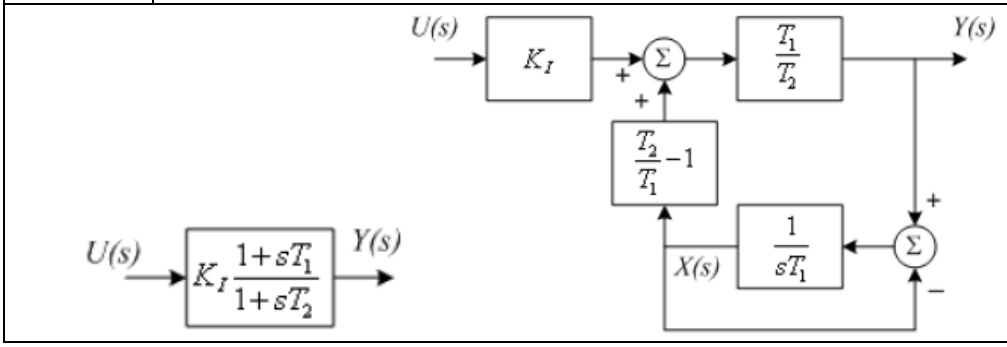
領先-落後基本方塊程式(整理自 PSS/E 使用手冊)	
MODE1 **(初始化)**	INTEGER IERR REAL VINP REAL VOUT C_ C____ UNLIMITED LEAD-LAG BLOCK C_ VOUT = Y VINP = LDLG_MODE1(KI, ! gain (real) # T1, ! numerator (lead) time constant (real) # T2, ! denominator (lag) time constant (real) # VOUT, ! block output variable (real) # K, ! index of state variable (integer) # IERR) ! error code (integer) C C IERR = 0 => no error C IERR = 3 => Gain KI = 0 C IERR = 4 => T2 = 0 (Dynamics of the block is ignored) C IERR = 5 => T1 = 0 (Block treated as first order lag)

MODE2 (模擬)

```
C_
C____ UNLIMITED LEAD-LAG BLOCK
C_
        VINP = U
        VOUT = LDLG_MODE2( KI,              ! gain (real)
     #                     T1,              ! numerator (lead) time constant (real)
     #                     T2,              ! denominator (lag) time constant (real
     #                     VINP,            ! block input variable (real)
     #                     K  )             ! index of state variable (integer)
```

MODE3 (輸出)

```
C_
C____ UNLIMITED LEAD-LAG
C_
        VINP = U
        VOUT = LDLG_MODE3( KI,              ! gain (real)
     #                     T1,              ! numerator (lead) time constant (real)
     #                     T2,              ! denominator (lag) time constant (real
     #                     VINP,            ! block input variable (real)
     #                     K  )             ! index of state variable (integer)
```

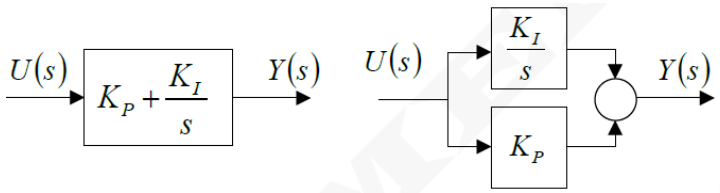


5. 比例-積分(Proportional Integral (PI))

比例-積分 MODE 1、MODE 2 程式碼及基本方塊程式碼(及其變數宣告)，如表 5-9 及表 5-10 所示。

表 5-9 比例-積分方塊圖程式碼

比例-積分方塊圖程式碼	
MODE1(初始化)	**MODE2(模擬)**
DSTATE = 0 U = 0 STATE = Y	DSTATE = KI*U Y = (KP*U)+STATE



(課程講義提供)

表 5-10 比例-積分基本方塊程式

比例-積分基本方塊程式(整理自 PSS/E 使用手冊)

MODE1
(初始化)

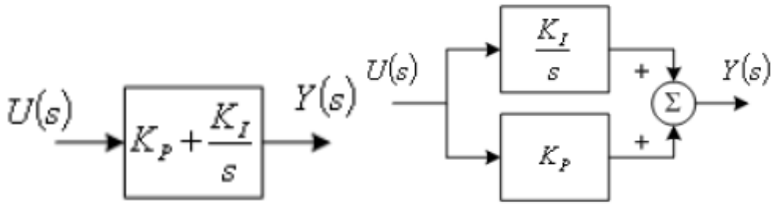
```
INTEGER IERR
REAL VINP
REAL VOUT
C_
C____ UNLIMITED PI BLOCK (INITIALIZED WITH OUTPUT = Y)
C_
        VOUT = Y
        VINP = PI_MODE1( KP,              ! proportional gain (real)
     #                   KI,              ! integral gain (real)
     #                   VOUT,            ! block output variable (real)
     #                   K,               ! index of state variable (real)
     #                   IERR  )          ! error code (integer)
C
C IERR = 0 => no error
C IERR = 3 => Gain KP = 0 (fatal error at initialization)
```

MODE2
(模擬)

```
C_
C____ UNLIMITED PI
C_
        VINP = U
        VOUT = PI_MODE2( KP,              ! proportional gain (real)
     #                   KI,              ! integral gain (real)
     #                   VINP,            ! block input variable (real)
     #                   K,               ! index of state variable (real)
```

MODE3
(輸出)

```
C_
C____ UNLIMITED PI
C_
        VINP = U
        VOUT = PI_MODE3( KP,              ! proportional gain (real)
     #                   KI,              ! integral gain (real)
     #                   VINP,            ! block input variable (real)
     #                   K,               ! index of state variable (real)
```



6. 比例-積分-微分(Proportional Integral Derivative (PID))

比例-積分-微分 MODE 1、MODE 2 程式碼及基本方塊程式碼(及其變數宣告)，如表 5-11 及表 5-12 所示。

表 5-11 比例-積分-微分方塊圖程式碼

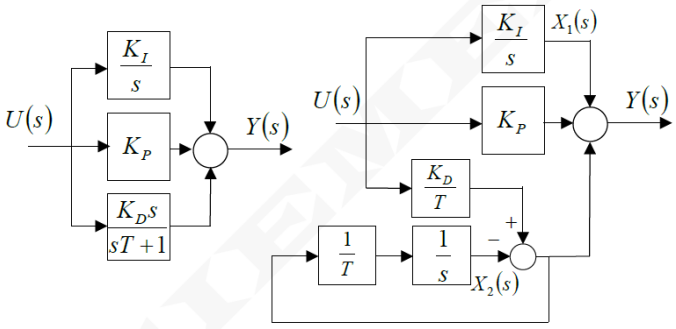
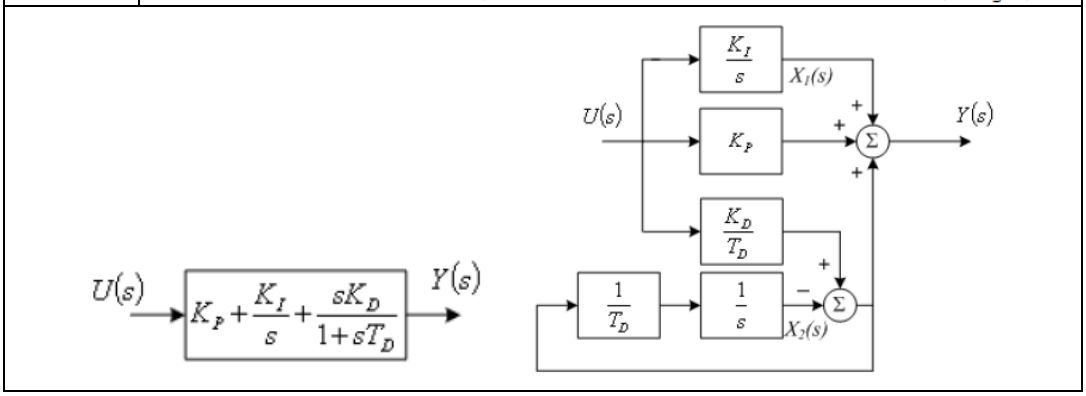
比例-積分-微分方塊圖程式碼	
MODE1(初始化)	MODE2(模擬)
DSTATE1 = 0 DSTATE2 = 0 U = 0 STATE1 = Y STATE2 = 0	DSTATE1 = KI*U DSTATE2 = (KD*U/T-STATE2)/T Y = STATE1-STATE2+(KP+KD/T)*U
 (課程講義提供)	

表 5-12 比例-積分-微分基本方塊程式

比例-積分-微分基本方塊程式(整理自 PSS/E 使用手冊)	
MODE1 (初始化)	INTEGER IERR REAL VINP REAL VOUT C_ C____ UNLIMITED PID BLOCK (INITIALIZED WITH OUTPUT = Y) C_ VOUT = Y VINP = PID_MODE1(KP, ! proportional gain (real) # KI, ! integral gain (real) # KD, ! derivative gain (real) # TD, ! time constant of the derivative channel (real) # VOUT, ! block output variable (real) # K, ! index of integral state variable (integer) # K1, ! index of derivative state variable (integer) # IERR) ! error code (integer) C IERR = 0 => no error C IERR = 3 => Gain KP = 0 (fatal error at initialization)
MODE2 (模擬)	C_ C____ UNLIMITED PID C_ VINP = U VOUT = PID_MODE2(KP, ! proportional gain (real) # KI, ! integral gain (real) # KD, ! derivative gain (real) # TD, ! time constant of the derivative channel (real) # VINP, ! block input variable (real) # K, ! index of integral state variable (integer) # K1) ! index of derivative state variable (integer)

MODE3 (輸出)	C_ C____ UNLIMITED PID C_ VINP = U VOUT = PID_MODE3(KP, ! proportional gain (real) # KI, ! integral gain (real) # KD, ! derivative gain (real) # TD, ! time constant of the derivative channel (real) # VINP, ! block input variable (real) # K, ! index of integral state variable (integer) # K1) ! index of derivative state variable (integer)

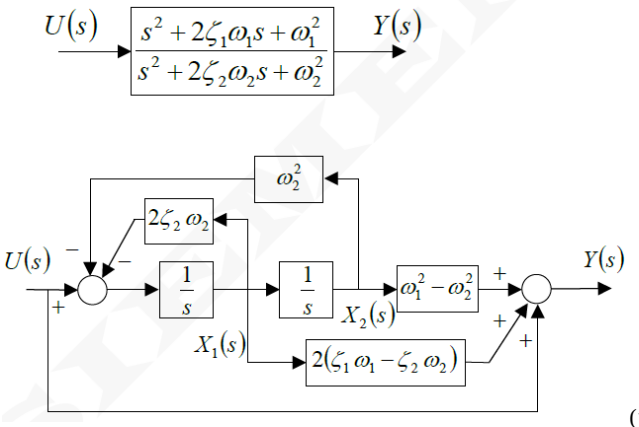


7. 二階(Second Order Block)

二階 MODE 1、MODE 2 程式碼及基本方塊程式碼(及其變數宣告)，如表 5-13 及表 5-14 所示。

表 5-13 二階方塊圖程式碼

二階方塊圖程式碼	
MODE1(初始化)	**MODE2(模擬)**
DSTATE1 = 0 STATE1 = 0 U = Y*(W2*W2)/(W1*W1) DSTSTE2 = 0 STATE2 = Y/(W1*W1)	Y = U+(W1*W1-W2*W2)*STATE2 +2*(ZETA1*W1-ETA2*W2)*STATE1 DSTATE1 = U-2*ZETA2*W2*STATE1 -(W2*W2)*STATE2 DSTATE2 = STATE1



(課程講義提供)

表 5-14 二階基本方塊程式

二階基本方塊程式(整理自 PSS/E 使用手冊)

MODE1
(初始化)

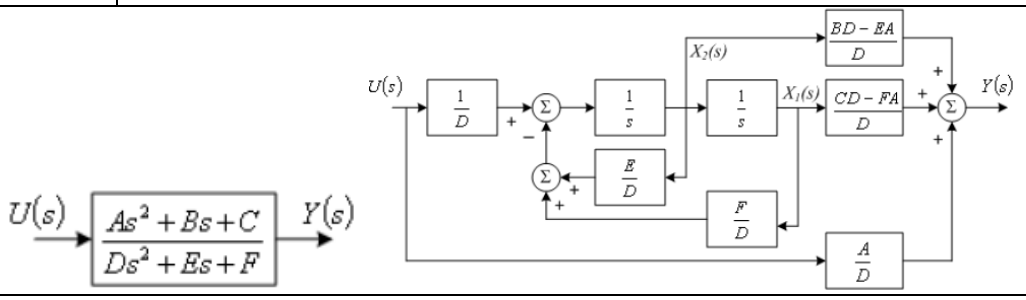
```
INTEGER IERR
REAL VINP
REAL VOUT

C_
C____ 2ND ORDER BLOCK (INITIALIZED WITH OUTPUT = Y)
C_
        VOUT = Y
        VINP = ORD2_MODE1( A,           ! parameter A (real)
     #                     B,           ! parameter B (real)
     #                     C,           ! parameter C (real)
     #                     D,           ! parameter D (real)
     #                     E,           ! parameter E (real)
     #                     F,           ! parameter F (real)
     #                     VOUT,        ! block output variable (real)
     #                     K,           ! index of first state variable (integer)
     #                     K1,          ! index of second state variable (integer)
     #                     IERR  )      ! error code (integer)
C
C IERR = 0 => no error
C IERR = 3 => D = 0 (dynamics of block is ignored)
C IERR = 4 => C = 0
```

MODE2
(模擬)

```
C_
C____ SECOND ORDER BLOCK
C_
        VINP = U
        VOUT = ORD2_MODE2( A,           ! parameter A (real)
     #                     B,           ! parameter B (real)
     #                     C,           ! parameter C (real)
     #                     D,           ! parameter D (real)
     #                     E,           ! parameter E (real)
     #                     F,           ! parameter F (real)
     #                     VINP,        ! block input variable (real)
     #                     K,           ! index of first state variable (integer)
     #                     K1)          ! index of second state variable (integer)
```

MODE3
(輸出)

```
C____ SECOND ORDER BLOCK
C_
VINP = U
        VOUT = ORD2_MODE3( A,           ! parameter A (real)
     #                     B,           ! parameter B (real)
     #                     C,           ! parameter C (real)
     #                     D,           ! parameter D (real)
     #                     E,           ! parameter E (real)
     #                     F,           ! parameter F (real)
     #                     VINP,        ! block input variable (real)
     #                     K,           ! index of first state variable (integer)
     #                     K1)          ! index of second state variable (integer)
```



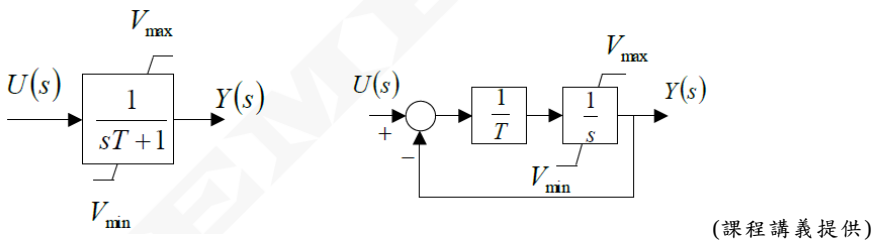
8. 一階狀態變數上下限(First Order with Non-Windup Limit)

一階狀態變數上下限 MODE 1、MODE 2 程式碼及基本方塊程式碼(及其變數宣告)，如表 5-15 及表 5-16 所示。

表 5-15 一階狀態變數上下限方塊圖程式碼

一階狀態變數上下限方塊圖程式碼	
MODE1(初始化)	**MODE2(模擬)**

STATE = Y
IF(Y .GT. VMAX) THEN
WRITE() 'INITIALIZATION ERROR'
ELSE IF(Y .LT. VMIN) THEN
WRITE() 'INITIALIZATION ERROR'
ENDIF
U = STATE
	IF(STATE .GT. VMAX) THEN
	STATE = VMAX
	STORE = VMAX
	DSTATE = (U-STATE)/T
	IF(DSTATE .GT. 0.) DSTATE = 0.
	ELSE IF(STATE .LT. VMIN)
	STATE = VMIN
	STORE = VMIN
	DSTATE = (U-STATE)/T
	IF(DSTATE .LT. 0.) DSTATE = 0
	ELSE
	DSTATE = (U-STATE)/T
	ENDIF

(課程講義提供)

表 5-16 一階狀態變數上下限基本方塊程式

一階狀態變數上下限基本方塊程式(整理自 PSS/E 使用手冊)

MODE1
(初始化)

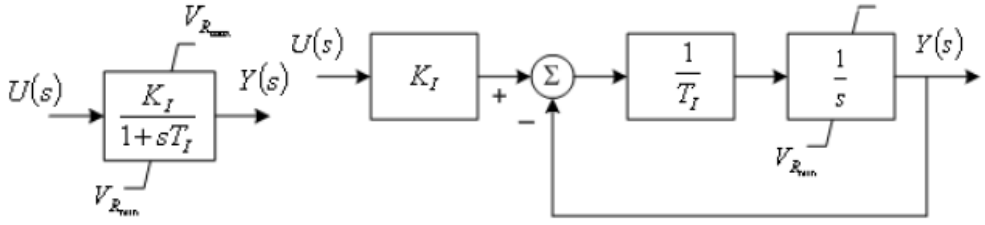
```
INTEGER IERR
REAL VINP
REAL VOUT
C_
C____ NON-WINDUP FIRST ORDER LAG BLOCK
C_
        VOUT = Y
        VINP = NWLAG_MODE1( KI,             ! gain (real)
     #                      TI,             ! time constant (real)
     #                      VRMAX,          ! max. limit (real)
     #                      VRMIN,          ! min. limit (real)
     #                      VOUT,           ! block output variable (real)
     #                      K,              ! index of state variable (integer)
     #                      IERR  )         ! error code (integer)
C
C IERR = 0 => no error
C IERR = 1 => initialization above VRMAX
C IERR = 2 => initialization below VRMIN
C IERR = 3 => Gain KI = 0 (fatal error at initialization)
C IERR = 4 => TI = 0
```

MODE2
(模擬)

```
C_
C____ NON-WINDUP FIRST ORDER LAG
C_
        VINP = U
        VOUT = NWLAG_MODE2(KI,              ! gain (real)
     #                     TI,              ! time constant (real)
     #                     VRMAX,           ! max. limit (real)
     #                     VRMIN,           ! min. limit (real)
     #                     VINP,            ! block input variable (real)
     #                     K  )             ! index of state variable (integer)
```

MODE3
(輸出)

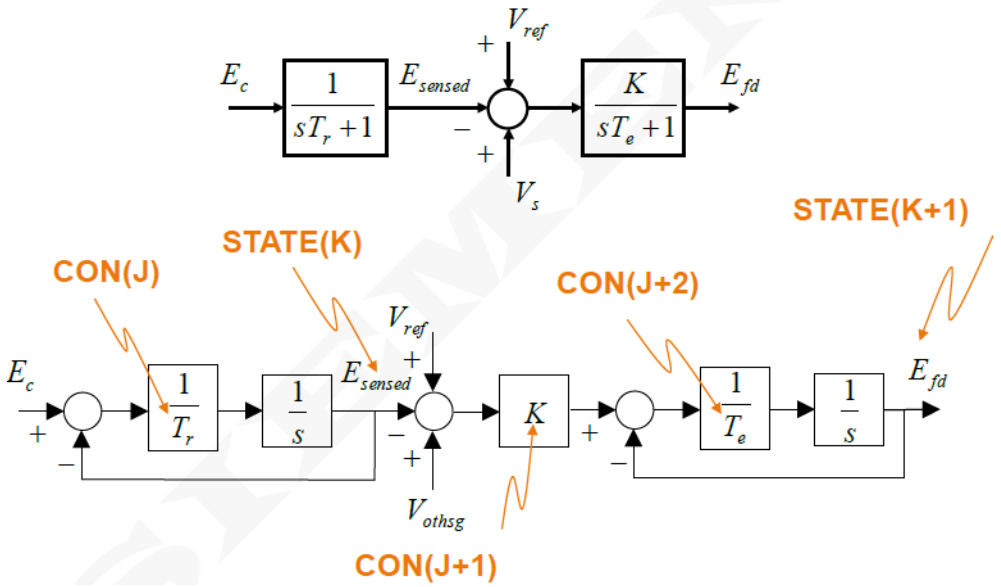
```
C_
C____ NON-WINDUP FIRST ORDER LAG
C_
        VINP = U
        VOUT = NWLAG_MODE3(KI,              ! gain (real)
     #                     TI,              ! time constant (real)
     #                     VRMAX,           ! max. limit (real)
     #                     VRMIN,           ! min. limit (real)
     #                     VINP,            ! block input variable (real)
     #                     K  )             ! index of state variable (integer)
```



5-2 使用者模型程式撰寫練習

本報告模型程式撰寫範例以課程中練習之簡化型激磁系統(Simplified Excitation System)UEXS 為例，撰寫動態模型前，需先給定數學模型方塊圖，並表列該模型使用之常數(CONs)、狀態變數

(STATEs)、代數變數(VARs)及整數常數(ICONs)等，簡化型激磁系統 UEXS 方塊圖及資料表(Datasheet)如下圖 5-9 及圖 5-10 所示。

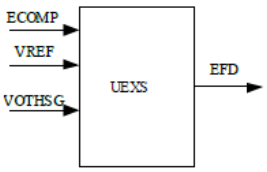


(課程講義提供)

圖 5-1 簡化型激磁系統 UEXS 模型方塊圖

UEXS
Simplified Excitation System

This model is located at system bus #_____ IBUS,
machine #_____ ID,
This model uses CONs starting with #_____ J,
and STATEs starting with #_____ K.



CONs	#	Value	Description
J			Tr (sec) > 0
J+1			K
J+2			Te (sec) > 0

STATEs	#	Description
K		Sensed Vc
K+1		E_{fd}

BUSID 'USRMDL' ID 'UEXS' 4 0 0 3 2 1 T_R, K, T_E /

(課程講義提供)

圖 5-2 簡化型激磁系統 UEXS 模型資料表

自建模型程式碼(FORTRAN)如下所示:

```
C[UEXS]    10/18/16   This is a sample excitation system model which was
C                     developed during the model writing class
C  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C++
C     This model is the model of an excitation system.
C--
C++
C     CONs
C     ----
C     J         TR, Measurement time constant
C     J+1       K, exciter gain
C     J+2       Te, exciter time constant
C--
      SUBROUTINE UEXS(MC,SLOT)
C
      INCLUDE 'COMON4.INS'
C
      IMPLICIT NONE
C
      INTEGER IB    ! bus sequence index
      INTEGER IBUS  ! exernal bus number
      INTEGER IERR  ! error code
      INTEGER J     ! starting CON index
      INTEGER K     ! starting STATE index
      INTEGER MC    !  machine index
      INTEGER SLOT  ! allocation index for getting the starting indices
C
      REAL VINP     ! block input
      REAL VOUT     ! block output
C
      LOGICAL ERRFLG ! error flag used in DOCU,check
C
      IF (MODE == 8) THEN
        CON_DSCRPT(1) = 'TR, Measurement time constant (s)'
        CON_DSCRPT(2) = 'K, exciter gain (pu)'
        CON_DSCRPT(3) = 'Te, exciter time constant (s)'
C
        RETURN
      ENDIF
C++
C     Add logic to derive the starting CON, STATE, VAR and ICON indices
C--
      J = STRTIN(1,SLOT)   ! starting CON index
      K = STRTIN(2,SLOT)   ! starting state index
C
      IF (MODE > 4) GO TO 1000
C++
C      Here MODE can be any value 1 trough 4, which are the simulation modes.
C      Because these are simulation modes, check if NUMTRM is negative and
C      RETURN out of the model.
C--
      IF (NUMTRM(MC) < 0) RETURN
C
      IF (MODE == 1) THEN
C
        IF (MIDTRM) RETURN
C
        VOUT = EFD(MC)
        VINP = LAG_MODE1(CON(J+1), CON(J+2), VOUT, K+1, IERR)
C
        VOUT = ECOMP(MC)
        VREF(MC) = VINP + VOUT - VOTHSG(MC)
C
        VINP = LAG_MODE1(1.0, CON(J), VOUT, K, IERR)
C
        RETURN
```

```
      ENDIF

      IF (MODE == 2) THEN
C
        IF (MIDTRM) RETURN
C
         VINP = ECOMP(MC)    ! block input
         VOUT = LAG_MODE2(1.0, CON(J), VINP, K)
C
         VINP = VREF(MC) - VOUT + VOTHSG(MC)
         VOUT = LAG_MODE2(CON(J+1), CON(J+2), VINP, K+1)
C
         RETURN
      ENDIF

      IF (MODE == 3) THEN
C
        IF (MIDTRM) RETURN
C
         VINP = ECOMP(MC)    ! block input
         VOUT = LAG_MODE3(1.0, CON(J), VINP, K)
C
         VINP = VREF(MC) - VOUT + VOTHSG(MC)
C
         VOUT = LAG_MODE3(CON(J+1), CON(J+2), VINP, K+1)
C
         EFD(MC) = VOUT     ! this is the model output
C
         RETURN
      ENDIF

      IF (MODE == 4) THEN
C
         IF (MIDTRM) THEN
C++
C     Since the model is not coded in MIDTRM, put out a message. The NOTMID
C     function will put that message out.
C--
            CALL NOTMID
            RETURN
         ELSE
C++
C     Set the value of NINTEG
C--
            IF (K+1 > NINTEG) THEN
               NINTEG = K+1
            ENDIF
            RETURN
         ENDIF
      ENDIF
C
 1000 IF (MODE == 6) THEN
C
         IB = ABS(NUMTRM(MC))      ! bus sequence number
         IBUS = NUMBUS(IB)         ! IBUS is the external bus number

         WRITE(DBUF01,507) IBUS, MACHID(MC), CON(J:J+2)
         CALL REPORTS(IPRT,DBUF01,2)
C
         RETURN
      ENDIF
C
      IF (MODE==5 .OR. MODE==7) THEN
C
         IF (MODE==5) THEN
            CALL DOCUHEADING
         ELSE        ! MODE is actually equal to 7
C++
C     Add model data checks
```

```
C--
      CALL DOCUCHK(2,                          ! CON number
     &             'K',                        ! CON name
     &              conMsgWARN,                ! type (error / warning)
     &              docuChkErr_InclusiveRange, ! range of check
     &              1.0,200.0,                 ! lower range, upper range
     &              ERRFLG)                    ! flag
C
      CALL DOCUCHK(3,                          ! CON number
     &             'Te',                       ! CON name
     &              conMsgError,               ! type (error / warning)
     &              docuChkErr_Equal,          ! range of check
     &              0.0,0.0,                   ! lower range, upper range
     &              ERRFLG)                    ! flag
C
      IF (.NOT. ERRFLG) THEN
C++
C     Since ERRFLG is FALSE, it means that Te is not equal to zero. Now check
C     if Te is > 0 and <= 0.5. This means Te is in not in range if Te<0 or
C     Te > 0.5
C--
        CALL DOCUCHK(3,'Te',conMsgWARN,   docuChkErr_InclusiveRange,
     &              0.0,0.5, ERRFLG)
C
C
      ENDIF
C++
C     If DOCU,check does not find any data errors, then RETURN out of MODE 7
C     logic
C--
      IF (.NOT. ERRORSFOUND()) RETURN
C
    ENDIF
C
    CALL SHOW_MODEL_INDICIES(0,     ! starting ICON index
     &                       0,     ! ending ICON index
     &                       J,     ! starting CON index
     &                       J+2,   ! ending CON index
     &                       K,     ! starting state index
     &                       K+1,   ! ending state index
     &                       0,     ! starting VAR index
     &                       0,     ! ending VAR index
     &                       0,     ! starting reserved ICON index
     &                       0)     ! ending reserved ICON index
C
    WRITE(DBUF01,7) CON(J:J+2)
    CALL REPORTS(IPRT,DBUF01,2)
C
    RETURN
  ENDIF
C++
C     Format statements
C--
 7    FORMAT(3X,'  TR       K         Te   '/
     &       1X,3G12.5)
 507  FORMAT(I7,'  ''USRMDL''  ',A2,'  ''UEXS''  ',' 4   0  0   3   2   0  ',/
     &       7X,3G12.5,'   /')
C
      END SUBROUTINE UEXS
```