行政院原子能委員會

委託研究計畫研究報告

核能級作業系統研究

Nuclear grade operating system

計畫編號：1002001INER023

受委託機關(構)：台灣科技大學電機所

計畫主持人：陳雅淑 助理教授

聯絡電話：0939-000953

E-mail address：yschen@mail.ntust.edu.tw

核研所聯絡人員：陳昌國

報告日期： 100 年 12 月 6 日

摘要

本計畫基於核能級控制器現發展之即時作業系統，評估其執行安全功能的時間確定性。本論文針對既有之TaiNICS應用程式建立系統化評估機制與時間分析之理論模型，分析TaiNICS應用程式之可排程性，進而限制該程式之非安全功能項目，並提出重新規劃應用程式規格的因應方案。

Abstract

This research project is to evaluate the response time predictability of the TaiNICS application which is executed on the real time operating system in the developed nuclear grade controller. The application schedulability analysis and improvement, unsafe function unit remove, processor resource management are explored in this project. The systematic analysis framework and mathematical formulas are proposed to improve the timing predictability of the real time operating system in a nuclear grade controller.

目　　錄

1. 前　言 1

2. 系統架構 2

2.1 核能級數位控制器架構 2

2.2 作業系統 3

3. TaiNICS 應用程式分析 7

3.1 TaiNICS 應用程式規格功能列表 7

3.1.1 優先權及排程方式 9

3.1.2 資源共用 9

3.2 時間分析理論模型 9

3.3 系統化評估機制 15

4. 時間行為塑模 16

4.1 執行時間(Execution time) 16

4.2 睡眠時間(Sleep time) 17

5. 系統模擬 18

5.1 系統模擬架構 18

5.2 線程產生器 18

5.3 事件觀察器(Watcher) 19

5.4 系統模擬結果 19

6. 結論 21

參考文獻 22

附　圖　目　錄

圖 2- 1：數位控制器架構 2

圖 2- 2：準備佇列 3

圖 2- 3：先進先出排程 4

圖 2- 4：調度排程 5

圖 2- 5：調度排程範例 5

圖 2- 6：先進先出排程範例 5

圖 3- 1：理論分析流程結果 14

圖 3- 2：系統化評估機制流程圖 15

圖 4- 1：線程運作模型 16

圖 5- 1：系統模擬架構 18

圖 5- 2：事件觀察器 19

圖 5- 3：NCL(21r)理論分析過程 20

圖 5- 4：系統模擬結果 20

附　表　目　錄

表 3- 1：核能級控制器之應用程式規格功能列表 8

1. 前　言

數位儀控技術於核電廠之應用為目前世界的趨勢，掌握自主關鍵技術將有助於核電廠的申照過程及長期之運轉與維護策略。有別於一般儀控系統，核能儀控系統對於即時反應時間的需求必須受到規範，才能保證該控制器執行安全功能時的正確性與安全性。然而，該控制器為多工執行環境，使得作業系統的排程設計、應用程式優先權設計與系統硬體中斷行為皆成為反應時間保證的重要關鍵。因此，如何分析核能級數位控制器上應用程式之反應時間需求與系統即時行為需求，並選擇適當之即時排程機制，有效地管控核能級數位控制器的系統資源，進而保證軟體執行的安全性成為一門重要課題[1-5]。

為確認核能級數位控制器執行期間的安全性，本計畫針對核能級數位控制器目前開發使用之 QNX 即時作業系統與 TaiNICS 應用程式，分析應用程式反應時間與系統時間行為的要求，建立時間確定性理論模型，進而評估其應用程式執行安全功能的時間確定性。為探索軟體反應時間確定性，本研究亦發展系統化評估機制，提供系統開發初期重新分配系統資源的準則以及限制非安全性功能的執行，以改善排程效能與安全性，期能對我國核電廠核能級控制器之軟體安全性設計有所助益，並協助核能級控制器的自行開發技術。本研究報告架構如下：第二章介紹系統架構，應用程式分析於第三章說明，時間行為塑模於第四章呈現，系統模擬流程於第五章解說，最後第六章為本研究報告的結論。

2. 系統架構

2.1 核能級數位控制器架構

本計畫主要目的為核能儀控系統上 TaiNICS 應用程式之安全性驗證。TaiNICS 應用程式主要負責處理、監控、傳遞週邊事件；例如從反應爐內接收感測器所偵測資料，並且將該資料進行搬移與運算比較。TaiNICS 應用程式依據功能內容分散成許多線程週期性執行於數位控制器；為有效管理多線程執行，本計畫考慮該控制器上利用 QNX 即時作業系統進行 CPU 資源利用與線程排程(圖 2-1)。有別於一般應用程式，TaiNICS 各線程的執行時間確定性將影響核能級控制器的安全性。因此，TaiNICS 應用程式執行期間的安全性驗證成為核能級數位控制器設計非常重要的一環。

圖 2- 1：數位控制器架構

2.2 作業系統

數位控制器採用的作業系統 QNX Neutrino [6-8]，QNX 作業系統為一個以 POSIX 為標準的即時(Real-time)作業系統，該作業系統只使用少量記憶體空間，並且可支援多任務(Multitasking)和線程(Thread)，提供優先權導向可搶佔式排程(Priority-driven preemptive scheduling)，一旦發生搶佔，可提供快速的本文交換(Context switch)，有效地減少額外時間的耗損。為了解應用程式於 QNX 作業系統上反應時間的確定性，以下將探討 QNX 作業系統所提供的排程策略，進而分析應用程式的執行順序與相互影響。

QNX 作業系統提供 0~255 共 256 層之準備佇列，該數值大小與優先權高低相應，例如：0 表示最低優先權，如下圖 2-2 所示:

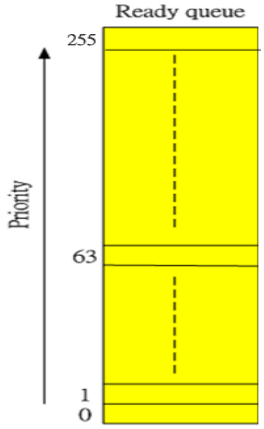

圖 2- 2：準備佇列

第 0 層為空閒(Idle)線程專用，第 1~63 層為非管理者(Non-root)線程使用，其餘 192 層為管理者(Root)線程使用，處理器處理線程之原則為優先權由高至低依序執行，若在同一優先權層底下存在多個線程需要被執行，則依照 QNX 作業系統所提供之三種排程策略先進先出排程(First In First Out，簡稱 FIFO)、調度排程(Round-Robin，簡稱 RR)與分散排程(Sporadic)決定執行順序，由於與分散排程為非週期性排

程，無法達到反應時間的確定性，故本研究使用先進先出與調度排程兩種策略，分別敘述如下:

(1) 先進先出(FIFO)排程：

在同一優先權層底下，由線程 A、B、C 抵達順序決定執行順序；優先抵達之線程優先執行，如下圖 2-3 所示。

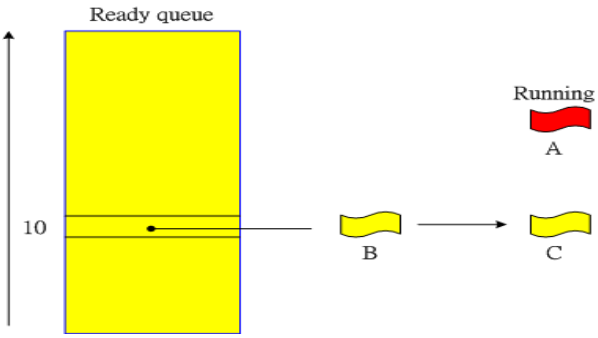

圖 2- 3：先進先出排程

(2) 調度(RR)排程：

與先進先出相似，在同一優先權層底下，由線程 A、B、C 抵達之順序決定執行順序，相異點是使用調度排程法之線程於單次執行中只被允許執行 4 個時脈週期(Tick)；若發生超出限制之情況，則該線程將被暫停執行，並強制移動至該佇列之最後端等待下次執行機會，依據 QNX 作業系統使用手冊[7]，每個時脈周期依處理器的 CPU 時脈不同而有所差異，CPU 時脈大於 40MHz，預設值為 1 毫秒(Millisecond)；CPU 時脈小於 40MHz，預設值為 10 毫秒。此外，時脈周期可用 QNX 核心指令 ClockPeriod()更改，最小可改為 10 微秒(Microsecond)，本研究之時脈周期是預設值 1 毫秒。調度排程法如下圖 2-4 所示。

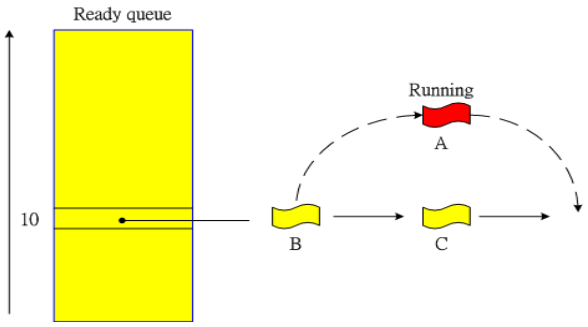

圖 2- 4：調度排程

調度排程與先進先出之差別以下例表示(圖 2-5、圖 2-6)。假設目前有兩組線程，其週期、執行時間、優先權和排程策略如下:

ID	Priority	Execution Time	Period
T_1	15r	6 ms	14 ms
T_2	15r	2 ms	13 ms
T_3	15r	3 ms	15 ms

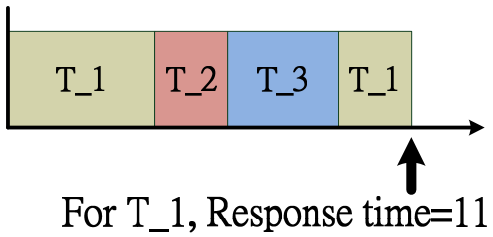

圖 2- 5：調度排程範例

ID	Priority	Execution Time	Period
T_1	15f	6 ms	14 ms
T_2	15f	2 ms	13 ms
T_3	15f	3 ms	15 ms

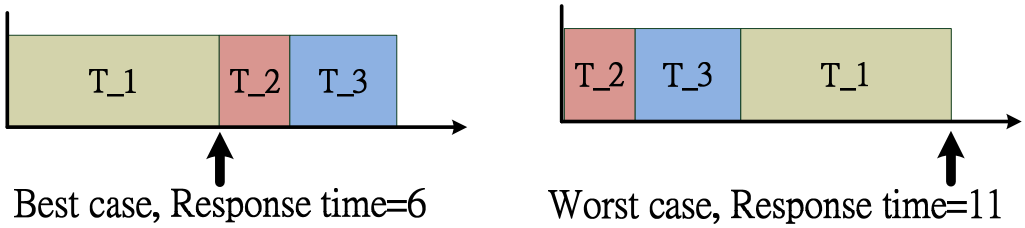

圖 2- 6：先進先出排程範例

以圖 2-5 為例，當所有線程之優先權相同且使用調度排程，假設線程 T_1 排在準備佇列最前端，由於受到系統限制執行時間(4 毫秒)，T_1 執行時間一旦執行到 4 個時脈週期(4 毫秒)，將被暫停執行並排至佇列最尾端，並依序執行 T_2 及 T_3，直到 T_3 執行完 T_1 才可繼續執行，故由本例可知，在使用調度排程下，即使排在準備佇列最前端，其執行時間超過時間限制之線程依然不會有較佳的回應時間。

以圖 2-6 為例，若所有線程改以先進先出排程，同樣 T_1 排在準備佇列最前端之情況下，由於沒有執行時間之限制，故 T_1 可完整執行至結束，則 T_1 有最短的回應時間；若假設情況為 T_1 排在準備佇列之最尾端，則其將造成 T_1 之回應時間為最長。從本例可觀察出，為考慮最差情況以確保系統安全性，因此在本研究中理論分析將假設每個線程皆有機會排在準備佇列之最尾端。

3. TaiNICS 應用程式分析

本計畫考慮之數位控制器的 TaiNICS 應用程式分為三大功能以及一 I/O 服務：(一) 新型控制語言(New Control Language, 簡稱 NCL)負責系統初始化、I/O 資料交換、溝通管理、錯誤偵測及處理。(二) 新型遠端輸出入匯流排接合器(New Remote I/O Bus Adapter, 簡稱 NBA)負責連接主處理器模組及次處理器(coprocessor)模組，並且將 I/O 資料從 I/O 模組搬移至共用記憶體。(三) 新型專業服務(New Engineer Service, 簡稱 NES)負責利用軟體看門狗技術監測上述的 NCL 及 NBA。(四) I/O 網路(IO-Net)負責接收外部感測元件回傳資料。本計畫針對 TaiNICS 應用程式於 QNX 作業系統上執行的時間安全性進行分析與討論。

3.1 TaiNICS 應用程式規格功能列表

數位控制器中的應用程式皆由 QNX 作業系統控管，必須由作業系統角度觀察與分析應用程式的反應時間。本計畫將 TaiNICS 應用程式建立為相對應作業系統的系統模型，首先將功能細分為各線程，並對各線程定義使用的排程策略、最長執行時間與優先權；討論其週期性與資源共用與否，定義其週期長度與資源種類。

經過分析與討論，TaiNICS 的應用程式之系統行為被定義為表 1 的系統模型，包括:識別碼 (ID)、優先權(Priority)與排程策略、程式功能(Process)、種類(Interrupt/ Thread)、週期性(Period)、執行時間(Execution time)及共享之資源 (Shared resource)。識別碼為 T_1 至 T_11，優先權所使用之數目越大代表優先權愈高，排程策略為先進先出或調度排程，分別表示為 f 或 r，例如:21r 代表優先權為 21，使用調度排程；18r 的優先權比 21r 要低；18f 之優先權較 18r 要高(QNX

作業系統中預設在同優先權下先進先出排程優先於調度排程)。程式功能代表該執行緒主要執行的功能，週期性與執行時間分別為執行緒發生時機以及最長預估時間，共享資源以 R1、R2 表示，Null 代表無使用資源。TaiNICS 之應用程式規格功能列表，如表 3-1 所示。

表 3- 1：核能級控制器之應用程式規格功能列表

Priority/ Scheduling (f:FIFO, r:RR)	Process	Interrupt/ Thread	Cycle time (ms)	Execution Time	Common Resource
21r (T2)	NCL	Thread	1	732 μs	R1
20f (T3)		Thread	100	154.54 μs	Null
9r (T9)		Thread	100	136 μs	Null
18f (T5)		Thread	20	246 μs	Null
21r (T1)	io-net	Interrupt	6	100 μs	R2
18r (T8) without priority inheritance	NBA	Thread	6	10.28 μs	R1
18f (T4) without priority inheritance		Thread	6	1.187ms	R2
18f (T6)		Thread	500	660 μs	Null
18f (T7)		Thread	120	2.64 μs	Null
9r (T10)		Thread	100	7.27 μs	Null
9r (T11)	NES	Thread	100	392 μs	Null

3.1.1 優先權及排程方式

在 QNX 作業系統中，單一優先權層可被多個線程註冊，另外，週期性工作可使用先進先出及調度排程兩種排程策略，當遇到兩線程優先權相同時，則假設使用先進先出之線程不可搶佔使用調度排程之線程，但是執行優先順序將以使用先進先出之線程優先。

3.1.2 資源共用

考慮資源共用之線程模型相較獨立線程模型之排程更為複雜，因一旦線程間發生資源共用關係，將有造成資料危障(Data hazard)之機會，影響運算結果的正確性，進而導致系統錯誤。

為有效解決資源共用所產生問題，因此在其線程使用共用資源時，必須將該資源引進保護資源機制(Resource protocol)，該保護方式主要原理為互斥(Mutual exclusion)機制，避免兩個或以上之線程在未完成使用資源之情況下重複使用資源；此外還須使用特定資源管理解決其衍生之問題[9-10]。

3.2 時間分析理論模型

以往即時可排程性之相關研究大多著重於單一優先權只允許單一線程可註冊或獨立線程[13-14]，而本研究之系統為單一優先權可註冊多線程，且線程間存在共用資源關係，故其可排程性分析更為複雜。本研究針對此特殊系統，提出一針對優先權同時對多線程進行時間分析理論模型ϕ如下:

$$\phi = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nk} \end{bmatrix} \qquad (1)$$

其中矩陣ϕ的行(Column)表示資源類別，列(Row)表示線程類別並以優先權高低遞減型態排列，$a_{ij} \in [0,1]$當中 1 表示第 j 線程有使用第 i 資源，反之則為 0，$i = 1,2,\ldots,n$與$j = 1,2,\ldots,k$。由表 3-1 可簡化資源共用情形，如下表所示：

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11
*R1	0	1	0	0	0	0	0	1	0	0	0
R2	1	0	0	1	0	0	0	0	0	0	0

*此例中，R1 表示為 T2 與 T8 共用的資源，雖 T8 優先權較 T2 低，但可能 R1 會先被 T8 占用，則最壞情況是 T2 要等 T8 執行完放掉 R1 資源。
為了避免發生優先權反轉(Priority inversion)發生，QNX 作業系統是使用優先權繼承(Priority inheritance)，以此例來說，T8 的優先權被暫時調整為 21(與 T2 相同)，可以避免 T8 占用 R1 資源時，被比它優先權高(如：T3)但比 T2 低的線程搶到執行權，而造成 T3 比 T2 還早執行完的優先權反轉現象。

則表示結果為：$\phi = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$

在分析最差情況之反應時間中，須將最長阻擋時間(blocking time)列入考慮；阻擋之定義為高優先權之線程因低優先權之線程使用資源而無法優先被執行的情況。第 i 線程之最長阻擋時間如式(2)所示。

$$B_{max,\tau_i}(\phi) = max\{(\bigcup_{\tau_i}^{\tau_h} a_{ij}) max\{C_{mj} a_{mj} | \forall \tau_i > \tau_m; m = i+1,\ldots,n\} | j = 1,\ldots,k\} \quad (2)$$

其中C_{mj}為第 m 線程使用第 j 個資源之時間，反應時間之計算模型由傳統計算方式[11]改良，將各線程之最長阻擋時間加入模型並以疊代方式計算至收斂為止，其計算式如下：

$$r_{0,\tau_i} = B_{max,\tau_i}(\phi) + \sum_{h=1}^{u} \{C_h | \forall \tau_h \geq \tau_i\} \quad (3)$$

與

$$r_{n,\tau_i} = B_{max,\tau_i}(\phi) + \sum_{h=1}^{u} \{C_h \lceil \frac{r_{n-1,\tau_i}}{P_h} \rceil | \forall \tau_h \geq \tau_i\} \quad (4)$$

其中 P_h 為第 h 線程的週期時間，τ_h 為第 h 線程的優先權。式(3)之 r0 表示為初始反應時間加入阻擋時間之計算值，式(4)為考慮上回合所計算之回應時間內其他線程在時間內的執行時間總量。若式(4)無法收斂或疊代過程中所計算結果超出該線程之週期時間，則表示該線程一旦被排程將有可能造成系統崩潰。

以一組未有共用資源之線程為例，說明本研究之理論分析部分與相關研究[13-14]之差異性，如下所示：

$$R_i^0 = C_i$$

$$R_i^{l+1} = C_i + B_i + \sum_{j=1}^{i-1} \lceil \frac{R_i^l}{T_j} \rceil C_j \qquad l = 0,1,2,\ldots$$

相關研究反應時間之分析如上兩式為針對單一線程逐個驗證，因此在驗證上所花費之時間成本相對較大。且在本系統中所存在之優先權註冊與以往架構有別，故所需驗證步驟更加繁瑣。

ID	Priority	Execution	Period
T1	21	5	25
T2	18	2	10
T3	18	6	12
T4	9	1	20

For T1: $R_1^0 = 5 \le 25$
$R_1^1 = 5 \le 25$

For T2: $R_2^0 = 2 \le 10$
$R_2^1 = 2 + 5 = 7 \le 10$
$R_2^2 = 2 + 5 = 7 \le 10$

For T3: $R_3^0 = 6 \le 12$
$R_3^1 = 6 + 5 + 2 = 13 \ge 12$

For T1: $R_1^0 = 5 \le 25$
$R_1^1 = 5 \le 25$

For T3: $R_2^0 = 6 \le 12$
$R_2^1 = 6 + 5 = 11 \le 12$
$R_2^2 = 6 + 5 = 11 \le 12$

For T2: $R_3^0 = 2 \le 10$
$R_3^1 = 2 + 6 + 5 = 13 \ge 10$

上例為使用相關研究分析方式在本研究系統架構下之結果，由結果可知若使用其方式需要耗費較長時間反覆驗證各種情況個線程之反應時間。若以本研究所提出之分析方式驗證，其結果如下：

For 21: $R_{0,21} = 5 \leq 25$

For 18: $R_{0,18} = 5 + 2 + 6 = 13 > 10\, and > 12$

兩者分析方式比較可發現本研究之分析方式之優點不僅在於可迅速分析，並且適用於是以往即時系統或是本研究之系統架構。

由表 3-1 可進行初步觀察，其中優先權為 21r 之 NCL 底下線程週期為 1ms，而優先權為 18f 之 NBA 底下線程最常執行時間為 1.187ms，此情況將造成每次 18f 線程被執行，則 21r 線程須額外多執行一次，且 21r 之線程執行時間長達 732us，導致無法通過排程測試。以下為 TaiNICS 應用程式驗證過程：

IO-net (21r):

$r_0 = T1$

$r_0 = 100\mu s \leq 6ms$

$r_1 = 100\mu s \leq 6ms$

NCL (21r):

$r_0 = T4 + T1 + T2$

$r_0 = 1.187ms + 100\mu s + 732\mu s = 2019\mu s > 1ms$

結果爲不可排程，需重新假設週期爲2019μs以上

$$r_1 = 1.187ms + 100\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil + 732\mu s\left\lceil \frac{r_0}{2.019\times10^3} \right\rceil$$

$= 2019\mu s \leq 2.019ms$

All (20f):

$r_0 = T4 + T1 + T2 + T3$

$r_0 = 1.187ms + 100\mu s + 732\mu s + 154.54\mu s = 2173.54\mu s \leq 100ms$

$$r_1 = 1.187ms + 100\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil + 732\mu s\left\lceil \frac{r_0}{2.019\times10^3} \right\rceil + 154.54\mu s\left\lceil \frac{r_0}{10^5} \right\rceil$$

$= 2905.54\mu s \leq 100ms$

$$r_2 = 1.187ms + 100\mu s\left\lceil \frac{r_1}{6\times10^3} \right\rceil + 732\mu s\left\lceil \frac{r_1}{2.019\times10^3} \right\rceil + 154.54\mu s\left\lceil \frac{r_1}{10^5} \right\rceil$$

$= 2905.54\mu s \leq 100ms$

All(18f):

$$r_0 = T8 + T1 + T2 + T3 + T4 + T5 + T6 + T7$$

$$r_0 = 10.28\mu s + 100\mu s + 732\mu s + 154.54\mu s + 246\mu s + 1187\mu s + 660\mu s + 2.64\mu s$$
$$= 3092.46\mu s \le 6ms$$

$$r_1 = 10.28\mu s + 100\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil + 732\mu s\left\lceil \frac{r_0}{2.019\times10^3} \right\rceil + 154.54\mu s\left\lceil \frac{r_0}{10^5} \right\rceil + 1187\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil + 246\mu s\left\lceil \frac{r_0}{2\times10^4} \right\rceil$$
$$+660\mu s\left\lceil \frac{r_0}{5\times10^5} \right\rceil + 2.64\mu s\left\lceil \frac{r_0}{1.2\times10^5} \right\rceil$$
$$= 3824.46\mu s \le 6ms$$

$$r_2 = 10.28\mu s + 100\mu s\left\lceil \frac{r_1}{6\times10^3} \right\rceil + 732\mu s\left\lceil \frac{r_1}{2.019\times10^3} \right\rceil + 154.54\mu s\left\lceil \frac{r_1}{10^5} \right\rceil + 1187\mu s\left\lceil \frac{r_1}{6\times10^3} \right\rceil + 246\mu s\left\lceil \frac{r_1}{2\times10^4} \right\rceil$$
$$+660\mu s\left\lceil \frac{r_1}{5\times10^5} \right\rceil + 2.64\mu s\left\lceil \frac{r_1}{1.2\times10^5} \right\rceil$$
$$= 3824.46\mu s \le 6ms$$

All (18r):

$$r_0 = 0 + T1 + T2 + T3 + T4 + T5 + T6 + T7 + T8$$

$$r_0 = 0 + 100\mu s + 732\mu s + 154.54\mu s + 246\mu s + 1187\mu s + 660\mu s + 2.64\mu s + 10.28\mu s$$
$$= 3092.46\mu s \le 6ms$$

$$r_1 = 0 + 100\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil + 732\mu s\left\lceil \frac{r_0}{2.019\times10^3} \right\rceil + 154.54\mu s\left\lceil \frac{r_0}{10^5} \right\rceil + 1187\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil + 246\mu s\left\lceil \frac{r_0}{2\times10^4} \right\rceil$$
$$+660\mu s\left\lceil \frac{r_0}{5\times10^5} \right\rceil + 2.64\mu s\left\lceil \frac{r_0}{1.2\times10^5} \right\rceil + 10.28\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil$$
$$= 3824.46\mu s \le 6ms$$

$$r_2 = 0 + 100\mu s\left\lceil \frac{r_1}{6\times10^3} \right\rceil + 732\mu s\left\lceil \frac{r_1}{2.019\times10^3} \right\rceil + 154.54\mu s\left\lceil \frac{r_1}{10^5} \right\rceil + 1187\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil + 246\mu s\left\lceil \frac{r_1}{2\times10^4} \right\rceil$$
$$+660\mu s\left\lceil \frac{r_1}{5\times10^5} \right\rceil + 2.64\mu s\left\lceil \frac{r_1}{1.2\times10^5} \right\rceil + 10.28\mu s\left\lceil \frac{r_1}{6\times10^3} \right\rceil$$
$$= 3824.46\mu s \le 6ms$$

All (9r):

$$r_0 = T1+T2+T3+T4+T5+T6+T7+T8+T9+T10+T11$$

$$r_0 = 0+100\mu s+732\mu s+154.54\mu s+246\mu s+1187\mu s+660\mu s+2.64\mu s+10.28\mu s+136\mu s+7.27\mu s+392\mu s$$
$$= 3627.73\mu s \le 100ms$$

$$r_1 = 0+100\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil +732\mu s\left\lceil \frac{r_0}{2.019\times10^3} \right\rceil +154.54\mu s\left\lceil \frac{r_0}{10^5} \right\rceil +1187\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil +246\mu s\left\lceil \frac{r_0}{2\times10^4} \right\rceil$$
$$+660\mu s\left\lceil \frac{r_0}{5\times10^5} \right\rceil +2.64\mu s\left\lceil \frac{r_0}{1.2\times10^5} \right\rceil +10.28\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil +136\mu s\left\lceil \frac{r_0}{10^5} \right\rceil +7.27\mu s\left\lceil \frac{r_0}{10^5} \right\rceil +392\mu s\left\lceil \frac{r_0}{10^5} \right\rceil$$
$$= 4359.73\mu s \le 100ms$$

$$r_2 = 0+100\mu s\left\lceil \frac{r_1}{6\times10^3} \right\rceil +732\mu s\left\lceil \frac{r_1}{2.019\times10^3} \right\rceil +154.54\mu s\left\lceil \frac{r_1}{10^5} \right\rceil +1187\mu s\left\lceil \frac{r_1}{6\times10^3} \right\rceil +246\mu s\left\lceil \frac{r_1}{2\times10^4} \right\rceil$$
$$+660\mu s\left\lceil \frac{r_1}{5\times10^5} \right\rceil +2.64\mu s\left\lceil \frac{r_1}{1.2\times10^5} \right\rceil +10.28\mu s\left\lceil \frac{r_1}{6\times10^3} \right\rceil +136\mu s\left\lceil \frac{r_1}{10^5} \right\rceil +7.27\mu s\left\lceil \frac{r_1}{10^5} \right\rceil +392\mu s\left\lceil \frac{r_1}{10^5} \right\rceil$$
$$= 5091.73\mu s \le 100ms$$

$$r_3 = 0+100\mu s\left\lceil \frac{r_2}{6\times10^3} \right\rceil +732\mu s\left\lceil \frac{r_2}{2.019\times10^3} \right\rceil +154.54\mu s\left\lceil \frac{r_2}{10^5} \right\rceil +1187\mu s\left\lceil \frac{r_2}{6\times10^3} \right\rceil +246\mu s\left\lceil \frac{r_2}{2\times10^4} \right\rceil$$
$$+660\mu s\left\lceil \frac{r_2}{5\times10^5} \right\rceil +2.64\mu s\left\lceil \frac{r_2}{1.2\times10^5} \right\rceil +10.28\mu s\left\lceil \frac{r_2}{6\times10^3} \right\rceil +136\mu s\left\lceil \frac{r_2}{10^5} \right\rceil +7.27\mu s\left\lceil \frac{r_2}{10^5} \right\rceil +392\mu s\left\lceil \frac{r_2}{10^5} \right\rceil$$
$$= 5091.73\mu s\mu s \le 100ms$$

圖 3- 1：理論分析流程結果

由驗證過程 NCL(21r)可知(圖 3-1)，在計算初始反應時間就已明顯超出其執行週期 1 毫秒，換言之，當 NCL(21r)之線程執行時，將有發生錯失期限(Miss deadline)的疑慮，故往上調整其執行週期為 2.019 毫秒，並重新計算，驗證其他線程之結果。由最後結果可知，在調整 NCL(21r)之執行週期後，其餘線程並無錯失期限發生的可能。

經由以上計算分析可知，優先權為 NCL(21r)之線程週期必需必須大於 2.019 毫秒，才可使計算收斂並且避免發生錯失期限而導致系統崩潰。

3.3 系統化評估機制

由理論模型可初步分析系統應用程式的可排程性，為進一步協助快速分析系統的行為，以重新規畫應用程式的規格，我們建構一系統評估機制工具模擬該系統執行狀況，進而驗證時間分析模型的可靠性。

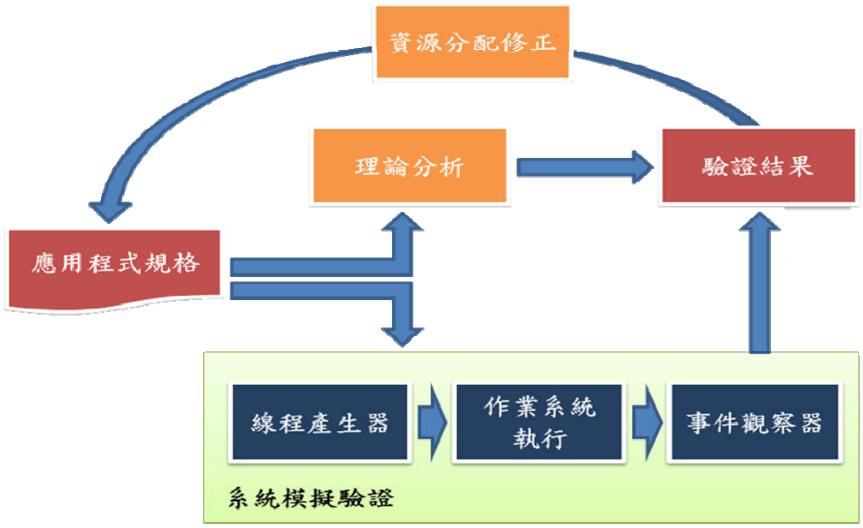

圖 3- 2：系統化評估機制流程圖

圖 3-1 為系統化評估機制流程圖，當取得應用程式規格後可同時進行理論及系統模擬部分，系統模擬部分包含依照線程規格在 QNX 作業系統中產生線程並實際執行，最後輸出實際排程結果， 理論部分將分析完結果與模擬輸出結果作比較。若理論結果與實際情況相符，並且確實有錯失期限的情況發生，該系統將啟動修正規格機制。

4. 時間行為塑模

由於操作在實際環境中會受到外在因素影響而產生時間誤差，其因素包含處理器架構、處理器頻率、作業系統背景線程(Background thread)…等，為精確實現各線程之時間行為模擬，本章將討論時間行為實現之方法。線程之運作模型，如圖 4-1.所示，可分為兩部分：(一)執行時間(Execution time)：該線程完成其工作所需耗費時間。(二)週期性(Period)：每間隔一段時間，線程將會被喚醒以執行其所負責之工作。為實現該運作模型時間行為，執行時間將由執行一單調迴圈數次以模擬該線程之時間行為；週期性行為則經由在執行時間結束時，強制該線程放棄處理器使用權一段時間(Sleep time)直到期限(Deadline)為止實現之。

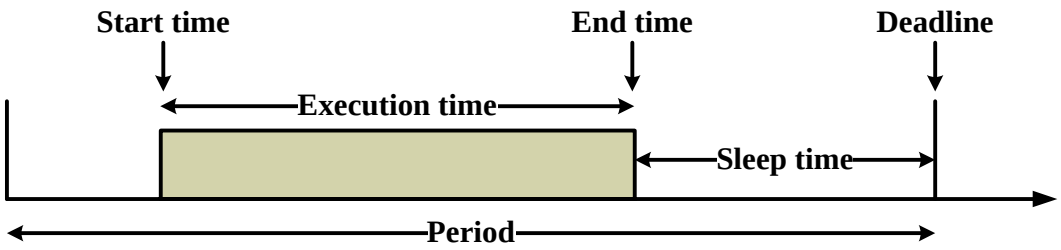

圖 4- 1：線程運作模型

4.1 執行時間(Execution time)

當線程獲得處理器使用權，線程即可執行指定之工作內容，然而，考慮系統發展初期尚未有實際線程可執行，因此為了模擬線程時間行為，線程執行時間由執行一單調迴圈數次模擬。以執行一次單調迴圈之時間為最小單位，經由式(5)將執行時間換算成等量之迴圈執行次數。

$$number\_of\_loop = \frac{execution\ time}{loop\ time} = \frac{execution\ time}{\frac{1}{f} \times number\_of\_cycles} = \frac{execution\ time \times f}{number\_of\_cycles} \quad (5)$$

其中 execution time(E)為線程執行時間，number_of_cycles 為單次執行函式所需之時間週期(Clock cycle)。利用 QNX 作業系統所提供之函式獲得系統時間週期，於迴圈執行前後獲得時間週期，相減即可得到執行迴圈所需之時間週期，不同處理器頻率會有不同之週期差。本研究之實驗環境處理器頻率為 2.8GHz，經實驗獲得執行迴圈所需之時間週期為 11~15 個週期，若取中間值則表示單次執行迴圈需花費 13 個時間週期，故執行時間誤差範圍為 $Eexcution\ Time \pm 6\%$ 。

4.2 睡眠時間(Sleep time)

當線程執行結束至下次執行有一段時間，系統會發出中斷，線程必須讓出處理器使用權給在等待佇列裡最高優先權之線程執行，直到時間消耗完後此線程才會再次執行。為了模擬此行為，線程執行結束進入睡眠時間，如圖 4-1，睡眠時間可經由下式得到：

$$Sleep\ time = Deadline - End\ time$$

將睡眠時間以微秒為單位將傳入 QNX 作業系統所提供之函式，此函式使得線程放棄處理器使用權，於睡眠時間消耗完後再次執行，即到達線程之週期(P_i)。本實驗所使用之處理器頻率大於 40MHz，根據作業系統預設之時脈週期為 1 毫秒，造成線程週期的誤差範圍為 $P_i \pm 2\%$ 。

5. 系統模擬

5.1 系統模擬架構

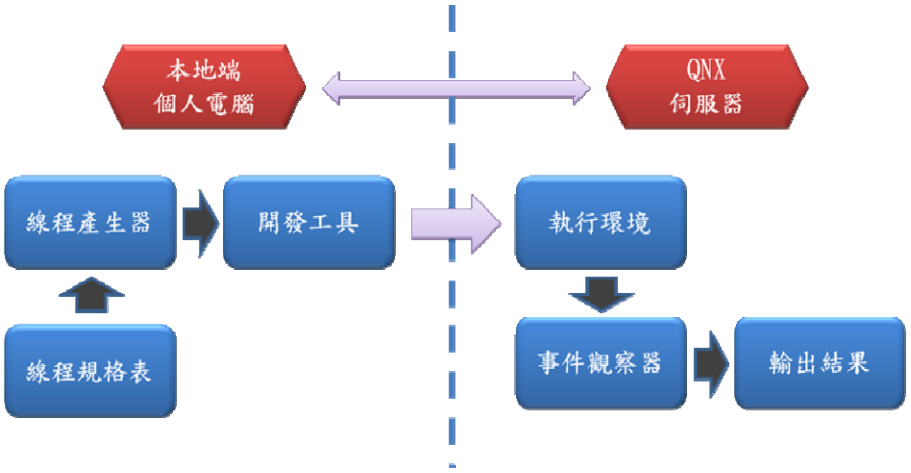

圖 5- 1：系統模擬架構

圖 5-1 為本研究計畫的系統模擬架構。首先，在本地端(Host)的電腦上使用線程產生器編譯完成輸出源碼檔；透過網路將檔案傳送至目的端(QNX 伺服器)進行模擬，最後產生出實測結果。

系統評估工具分為兩大部分：線程產生器與事件觀察器。 針對系統開發初期，尚未有完整應用程式檔案，可利用線程產生器針對應用程式規格產生相對應的線程之運作時間行為，並符合 QNX 作業系統可運行規範，接著將線程產生器產生的模擬線程在 QNX 作業系統中執行，再利用事件觀察器以觀察系統整體行為。

5.2 線程產生器

線程產生器內部可概括分為三個部分：(一) 建立樣板：將所有線程之各項規格經過編排後寫入樣板檔內。(二) 讀取樣板檔：依照該線程之各項參數讀取該參數至線程運作模型內適合之位置。(三) 輸出線程執行檔：將參數與線程模型合成後，產生一符合 QNX 作業系

統執行標準之執行檔。接著將線程產生器產生的模擬線程在 QNX 作業系統中執行，再利用事件觀察器以觀察系統整體行為。

5.3 事件觀察器(Watcher)

事件觀察器將會紀錄線程排程過程中的重要事件，包括起始時間、結束時間、搶佔(Preempt)時間、獲得(get)資源與釋放(Release)資源，開發者進而可由事件發生情況判斷如何重新規劃控制器資源。其利用標籤(Flag)記錄發生事件種類與時間點，如圖 5-2，將資訊一一存入，於所有線程執行完後，依據時間先後順序重新排序並印出結果。

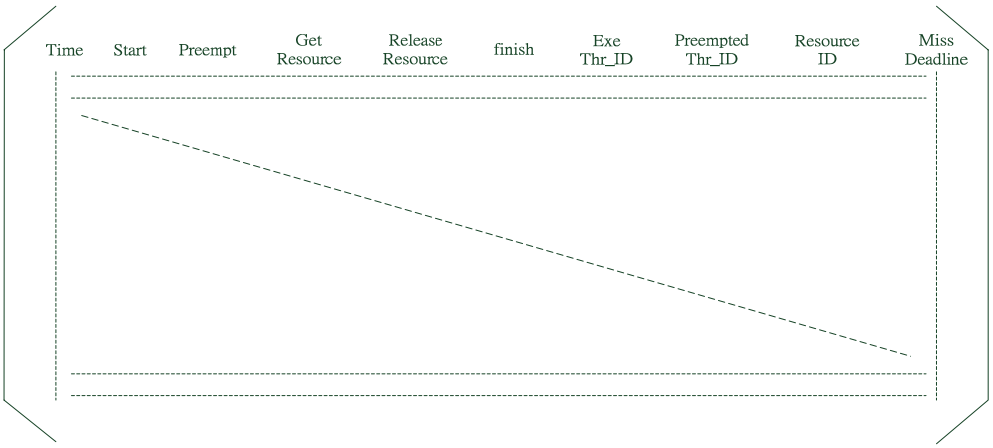

圖 5- 2：事件觀察器

5.4 系統模擬結果

本計畫針對表 3-1 的 TaiNICS 系統模型利用第三章節中的數學模型進行最差反應時間分析，發現 NCL(21r) 線程之反應時間已超出其執行週期 1 毫秒，換言之，當 NCL(21r)之線程執行時，將會有發生錯失期限(Miss deadline)可能，造成數位控制器的非安全行為，其反應時間分析如圖 5-3 所示(截自圖 3-1)。由於數學模型以最差情況進行考慮，本計畫同時利用上述之系統評估工具(線程產生器與事件觀察器)進行模擬驗證，仍發現 NCL(21r) 在部份情況下會發生錯失期

限，如圖 5-4 所示。

NCL (21r):

$r_0 = T4 + T1 + T2$

$r_0 = 1.187ms + 100\mu s + 732\mu s = 2019\mu s > 1ms$

結果爲不可排程，需重新假設週期爲$2019\mu s$以上

$$r_1 = 1.187ms + 100\mu s\left\lceil \frac{r_0}{6\times10^3} \right\rceil + 732\mu s\left\lceil \frac{r_0}{2.019\times10^3} \right\rceil$$

$= 2019\mu s \le 2.019ms$

圖 5- 3：NCL(21r)理論分析過程

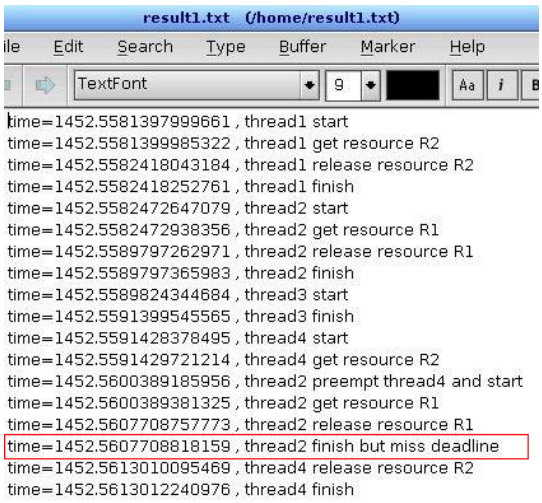

圖 5- 4：系統模擬結果

由理論分析及模擬結果的觀察中發現表 1 的 TaiNICS 應用程式規格將會有部分非安全行為發生。要避免該狀況，可重新分配控制器運算資源或重新定義系統規格。本計畫經由數學模型發現，若將 NCL(21r)之執行週期延長為 2.019ms 以上(圖 5-3)，即可使所有線程在時限內完成，保持系統的安全性。

6. 結論

本計畫旨在分析與評估核能級控制器內使用之TaiNICS應用程式的時間確定性，首先藉由作業系統的優先權設定與排程方式定義應用程式與系統行為之規格，進而探索並建立適合該系統的時間確定性分析之理論模型，並發展相關之系統評估工具，建立初期線程排程提供開發者重新定義系統規格或分配系統資源的參考，以確保系統能達到時間確定性的要求。

參考文獻

1. Asian Nuclear Safety Network, Technical Report, “Digital Instrumentation and Control Systems for Safety System and Main Control Room Design in Japan Nuclear Power Station,” December 2007.

2. International Atomic Energy Agency Vienna, Technical Report, “Implementing Digital Instrumentation and Control Systems in the Modernization of Nuclear Power Plants,” 2009.

3. National Academy Press, “Digital Instrumentation and Control Systems in Nuclear Power Plants, ” 1997

4. Lawrence, J.D.; Persons, W.L.; Preckshot, G.G.; Gallagher, J., “Evaluating software for safety systems in nuclear power plants, Computer Assurance,” Proceedings of the Ninth Annual Conference on Safety, Reliability, Fault Tolerance, Concurrency and Real Time, Security, 1994.

5. Jang-Soo Lee, Arndt Lindner, Jong-Gyun Choi, Horst Miedl and Kee-Choon Kwon “Software Safety Lifecycles and the Methods of a Programmable Electronic Safety System for a Nuclear Power Plant,” Lecture Notes in Computer Science, Volume 4166, 2006

6. QNX Realtime Operating System, “QNX Neutrino RTOS: Adaptive Partitioning User’s Guide,”http://www.qnx.com/

7. QNX Realtime Operating System, “QNX Neutrino Realtime Operating System: Library Reference,”http://www.qnx.com/

8. QNX Realtime Operating System, “QNX Neutrino RTOS:System Architecture,”http://www.qnx.com/

9. L. Sha, R. Rajkumar, and J.P. Lehoczky, “Priority Inheritance Protocols: An Approach to Real-Time Synchronization,” IEEE Trans. Computers, vol. 39, no. 9, pp. 1175-1185, Sept. 1990.

10. T.P. Baker, "A Stack-Based Resource Allocation Policy for Real Time Processes," IEEE I1th Real-Time Systems Symposium, December 4-7, 1990.

11. C.L. Liu and J.W. Layland, “Scheduling Algorithms for Multiprograming in a Hard Real-Time Environment,” J. ACM, vol. 20, no. 1, pp. 40-61, 1973.

12. POSIX Threads Programming, “POSIX Programmer's Manual, ”https://computing.llnl.gov/tutorials/pthreads/

13. Joseph, M., P. Pandya, “*Finding Response Times in a Real-Time System*,” BCS Computer Journal (Vol. 29, No. 5, Oct 86) pp.390-395.

14. Audsley, N., Burns, A., Richardson, M., Tindell, K. and Wellings, A., “*Applying New Scheduling Theory to Static Priority Pre-emptive Scheduling*,” Software Engineering Journal (September 1993).