CS305 作業系統概論 Prog. #3 Process Coordination 說明報告.docx

作者 ：1043335 賴詩雨

版本	修改日期	說明	修改者
1.0.0.0	2017-05-29	初稿	賴詩雨
1.0.0.1	2017-05-31	修改	賴詩雨
1.0.0.2	2017-06-01	修改	賴詩雨

目錄

第1章　程式完成部分:

a. 基本功能

b. 進階功能

第2章　設計理念

利用 pthread 的 mutex 機制形成 critical section，使 dispatcher 和 producer 能正確利用共用變數溝通。

2.1 程式中特殊介紹

1. 有兩個結構，分別為 _S_Table 和 _S_Producer
2.
```
typedef struct _S_Table
{
    int         quantity;          // 空拍機組裝完成的數量
    int         count;             // 桌上擁有零件的數量
    int         generate[3][2];    // 各個 producer 產生的數量
    bool        dispatch;          // 為 1，代表 dispatcher 產生完成
    bool        A_or_B;            // 輪到 dispatcherA ，為 0
    bool        component[3];      // 放零件的地方
    int         DisCount[3][3];    // 計算 dispatcher 產生的個別零件
    pthread_mutex_t lock;
} S_Table, *P_Table;
```
3.
```
typedef struct _S_Producer
{
    int         number;            // 此 producer 擁有的零件號碼，對應 enum 定義
    P_Table     ST;
} S_Producer, *P_Producer;
```
4. 整個程式中有用到的結構宣告為

```
P_Table pT;           // 用於 dispatcher
P_Producer pPRO[3];   // 用於三個 producer
```

5. 程式中 enum 定義

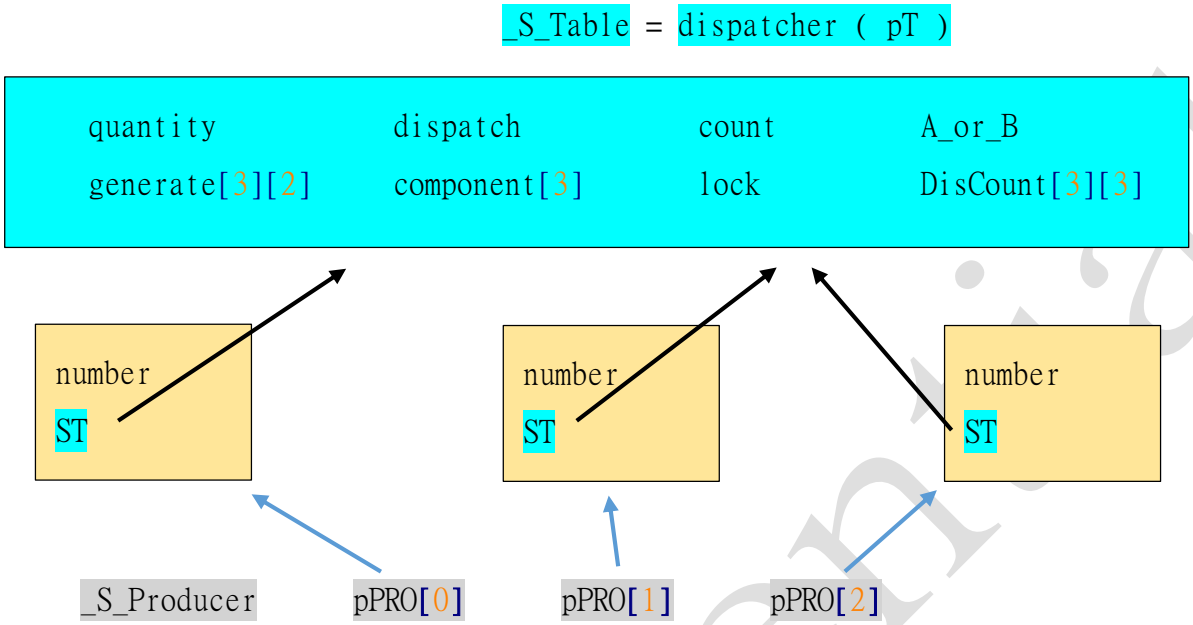
```
typedef enum {false, true} bool;
enum {battery, aircraft, propeller};
```

6. Function 說明

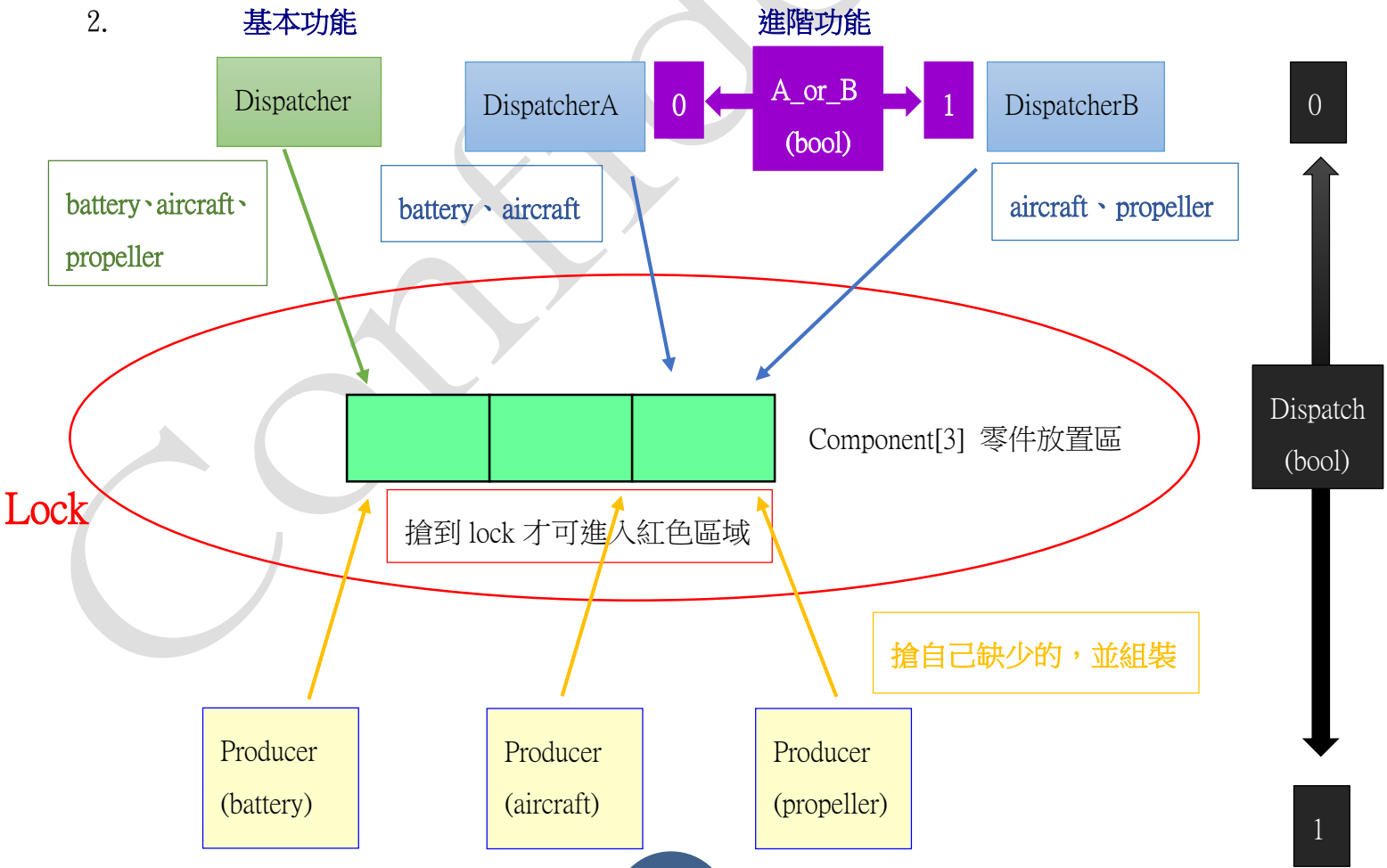
```
void producer( P_Producer ptr );  // producer
void dispatcher( P_Table ptr );   // 基本功能中的 dispatcher
void dispatcherA( P_Table ptr );  // 進階功能中的 dispatcherA
void dispatcherB( P_Table ptr );  // 進階功能中的 dispatcherB
```

2.2 結構圖

1.



2.

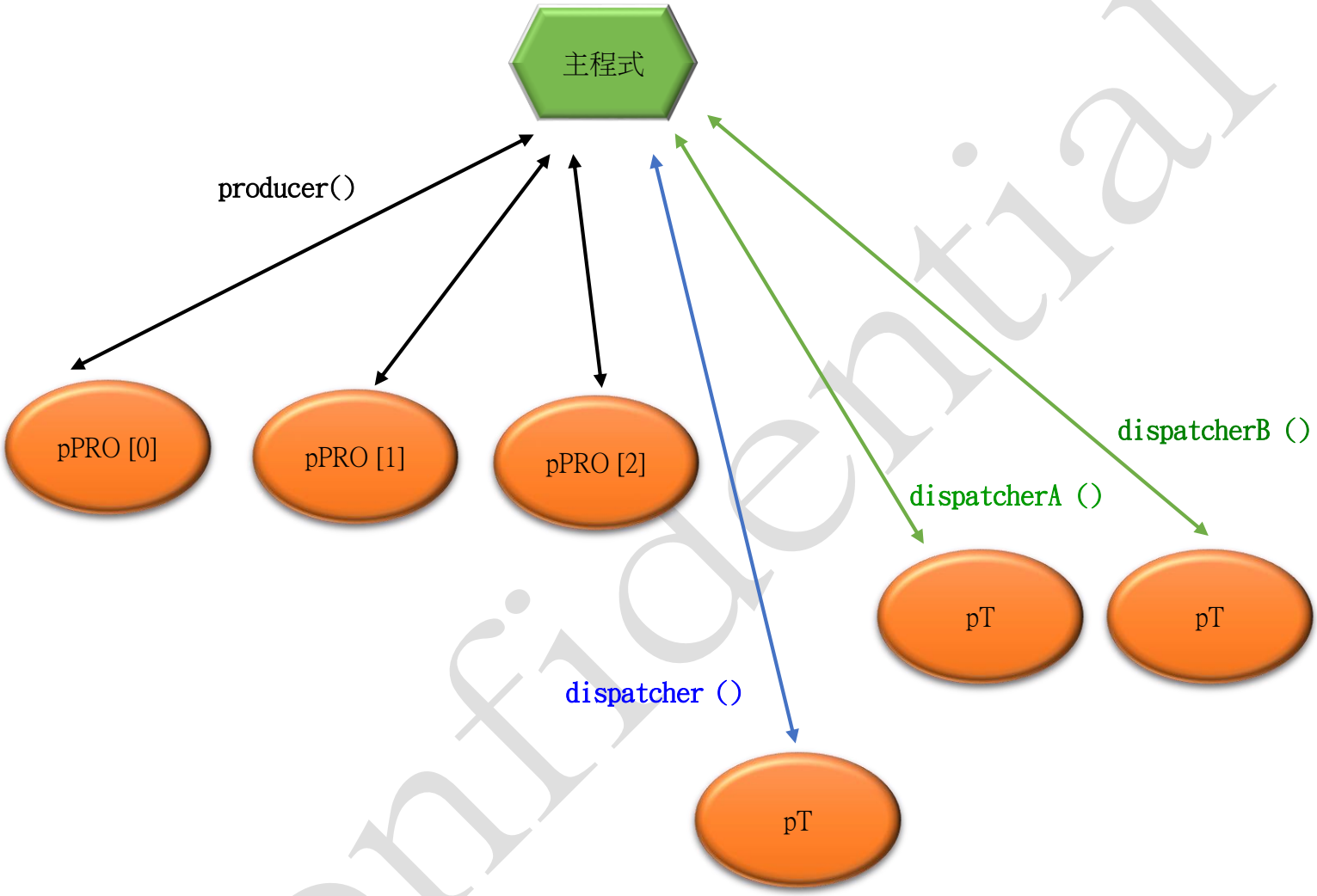


2.3 Thread 產生示意圖

結構

EX ：

```
pthread_create (&proThread1, NULL, (void *) &producer, pPRO[0]);
```



NOTE：pT 是指向共用結構區的指標

（p.3 的淺藍色區域 ）

第3章　程式流程簡介

1. 在 main 設定好結構的基本需求以及初始 lock 後，會開始 create 三個 producer (pPRO[0]~pPRO[2]) 。
2. 再來，根據命令列的數字(1 or 2)，決定要 create 一個(dispatcher)還是兩個 dispatcher(dispatcherA, dispatcherB)。
3. 都 create 好了以後，main 會等待 thread 執行。

[case 1] : 一個 dispatcher

4. 一開始 producer 還不可以搶 lock，因為桌上還沒有任何零件，所以 dispatcher 搶到 lock。
5. Dispatcher 會開始產生零件放到共用變數區(component[3])，若產生兩個零件了，則 producer 可以開始搶 lock。
6. 搶到 lock 的 producer 去判斷自己能否組成空拍機，若不行則把 lock 釋出給其他的 producer。
7. 當共用變數區的零件被消耗掉(零件少於兩個)，producer 就不可以搶 lock 了，dispatcher 搶到 lock 以後，可以繼續產生零件。
8. 重複執行 5~7 步驟，直到產生了 50 台空拍機。

[case 2] : 兩個 dispatcher

9. 一開始 producer 還不可以搶 lock，因為桌上還沒有任何零件，所以 dispatcherA 和 dispatcherB 去搶 lock 。

   Note: 為了確保公平競爭，布林值 A_or_B 會隨機獲得 0 or 1， 0 就讓 dispatcherA 搶到 lock。

10. 搶到的 Dispatcher 會開始產生零件放到共用變數區(component[3])，產生完一個零件後，dispatcherA 和 dispatcherB 再次去搶 lock，若產生兩個零件了，則 producer 可以開始搶 lock。
11. 搶到 lock 的 producer 去判斷自己能否組成空拍機，若不行則把 lock 釋出給其他的 producer。
12. 當共用變數區的零件被消耗掉(零件少於兩個)，producer 就不可以搶 lock 了，dispatcher 搶到 lock 以後，可以繼續產生零件。
13. 重複執行 9~12 步驟，直到產生了 50 台空拍機。
14. 擁有 50 台空拍機以後，main 會
    - 列印出 dispatcher 產生各種零件的數量
    - 根據每個 producer 組成空拍機的數量，由大到小印出。

第4章　程式輸出簡介

黑字 => 會印出的訊息

藍字 => 程式的動作

[case 1]

Dispatcher 搶到 lock，開始生產零件

Dispatcher: propeller

Dispatcher: battery

Producer 們可以去搶 lock，搶到的開始嘗試組裝

Producer (aircraft): OK, 1 drone(s)

Dispatcher: aircraft

Dispatcher: propeller

Producer (battery): OK, 2 drone(s)

...

...

Producer (propeller): OK, 50 drone(s)

Main 印出 dispatcher 生產的各種零件數量

Main 根據組裝數量由小排到大並印出

[case 2]

Dispatcher 們去搶 lock，搶到的開始生產零件

DispatcherA: propeller

DispatcherB: battery

Producer 們可以去搶 lock，搶到的開始嘗試組裝

Producer (aircraft): OK, 1 drone(s)

DispatcherB: aircraft

DispatcherB: propeller

Producer (battery): OK, 2 drone(s)

DispatcherB: aircraft

DispatcherA: propeller

Producer (battery): OK, 3 drone(s)

...

...

Producer (propeller): OK, 50 drone(s)

Main 印出 dispatcher 生產的各種零件數量

Main 根據組裝數量由小排到大並印出

第5章　程式如何編譯

1. 有附上 Makefile 檔提供操作

   OR

2. 在命令列輸入　　：gcc 1043335_03.c -lpthread -o 1043335_03
3. 編譯成功後，輸入：./1043335_03 1

NOTE：1 可改成 2，分別代表基本功能和進階功能

第6章　如何操作

1. 將 Makefile 中 run 的 1 改成 2，可從基本功能切換成進階功能
2. make main　=> 編譯
3. make run　=> 執行
4. make start => 編譯 & 執行
5. make clean => 清除編譯、執行後產生的檔案
6. make all　=> 清除檔案後編譯執行
7. 或是可以直接在命令列輸入 **第五章** 提供的指令