# Computer Organization and Assembly Language

## Final Project

# War of Tank

B96902007 黃柏翰

B96902114 陳世穎

# 目錄

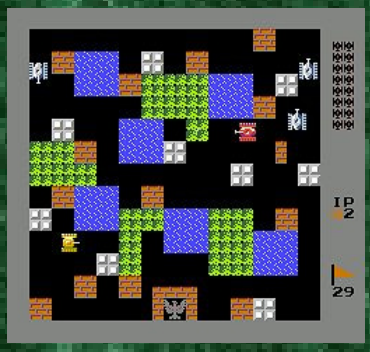
# 四. 參考資料,軟體

## 一. 動機:<<坦克大戰>>一直是從小以來僅次於俄羅斯方塊的經典紅白機遊戲,相信很多大家很多兒時的美好回憶中有這段,所以我們想要藉由實作出這個遊戲去深入了解這款經典遊戲的迷人所在之處,一方面坦克大戰也是很多遊戲類型的始祖,了解這遊戲,想必往後對於製作其他遊戲幫助一定很大。

## 二. 遊戲簡介

### 1. 原版遊戲歷史:<<坦克大戰>>是一款平面射擊遊戲。此紅白機遊戲是 1985 出版於日本開發的南夢宮。此後，這遊戲又在 GMEBOY 出版。這遊戲是模仿 1980 年街機

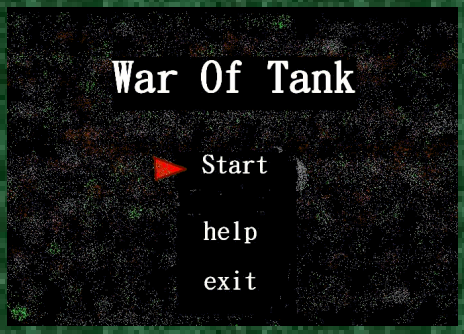
遊戲<<TANK Battlion>>而製作的。



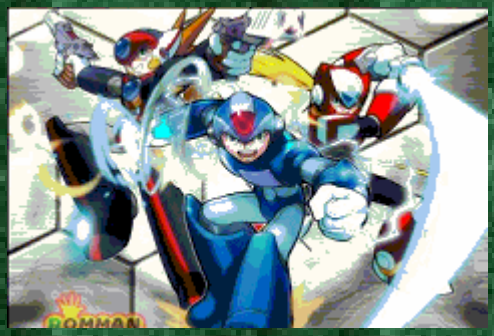
^^原版的坦克大戰。

## 2. 操作介面：

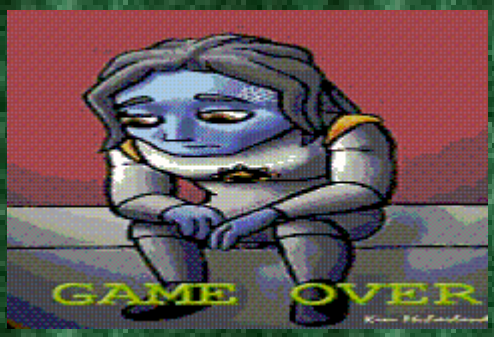
### (1) 操作介面：主要操作介面由小畫家與載入圖檔完成。




^^小畫家製作的遊戲開始畫面。

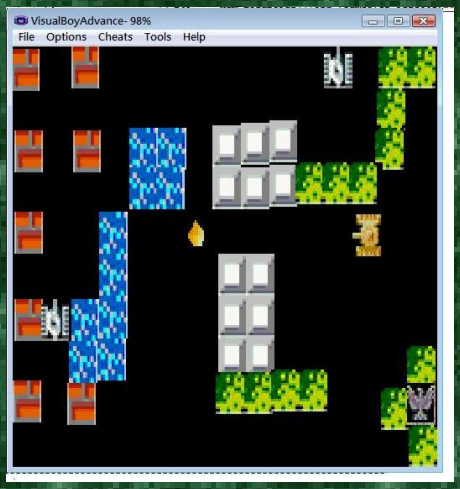


^^關卡全破的勝利畫面。




^^打輸時的畫面。

(2) 遊戲介面:主要的遊戲介面大致上與原版坦克大戰相似，而我們主要是採用 240*160 像素為整個遊戲畫面，並以 16*16 作為遊戲中的方格。




^^遊戲畫面

**(3)遊戲設定:**比較大的不同點在於原版坦克子彈不能連發，而我們的可以依照使用者連點的速度連續發射。另外子彈並沒有對射抵銷的狀況，且由於許多程式設計上的困難，我們並沒有設定子彈能打破磚塊，也因此少了許多坦克大戰的遊戲樂趣，這是我們遊戲設計上比較大的缺憾。

**3 .勝敗條件:**將敵方坦克擊毀數台後，到達敵方基地即可獲勝。

而在任何情況之下只要被敵方坦克碰觸到或是被子彈擊中即 **GAMEOVER**

**4.遊戲流程:**一開始進入遊戲之後，有三個，選項分別是開始，幫助，離開，開始則馬上進入遊戲，幫助則進入操作說明與勝負條件進入主要遊戲之後，就開始一一擊破敵人坦克

並進入敵方基地到達下一關藉以獲勝，若失敗則會進入 GAMEOVER 畫面。

## 三. 實作過程

### 1. 重要函式:

**ham_StartIntHandler(INT_TYPE_VBL,(void*)&my_func);**

這是主要能讓我們以每 1/k 秒為物件的執行時間單位的重要函數，他會每 1/60 秒直行一次第二個參數的 function pointer，而我們就可以控制變數來決定物件動作的循環時間。

**ham_CreateObj((void*)&buffer_Bitmap[2048], 0,1,**

**OBJ_MODE_NORMAL,1,0,0,0,0,0,0,ppx,ppy);**

這個 function 就是我們為什麼能讓物件在遊戲中移動，因為這個 function

會創造 Sprite，（精靈），讓我們可以控制她的位置，進而能移動等等，

第一個參數是圖片的 pointer,而後面兩個 0，1 是代表很重要的圖形大小，

以 0,1 來說，就是 16x16,而我們就可以創造一個 16*16*10 的 10 格圖片一次存取

而為什麼要一次讀 16*16*10 呢，因為如果一個個 sprite 分開來存，在 load palette（調色盤）的時候會把之前的精靈給洗掉。

**ham_UpdateObjGfx(id , (void*)graph)**

在坦克車移動的時候，左轉右轉必須要更動坦克的方向，但是在遊戲中並不能那麼直觀，必須要變動我們坦克的圖，所以這個函數可以讓我們改動精靈所代表的圖片。

**random_enemy ()**

這個函數是我們自訂的函數，其作用為用來生出敵方坦克，而為了使遊戲簡單一點，如果一個位置已經生出過坦克，將不會再度將他生在那個地方，

其中我們用到了 C language 中的 rand()來生出來 Winner() and Gameover()

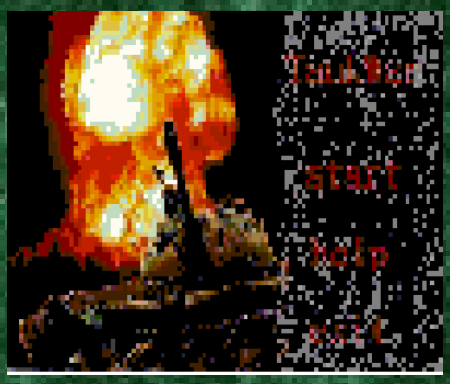
這兩個自訂函數都很簡單，當玩家達到勝利或失敗條件之後，

將會進入這兩個之中一個，然後按 A(for keyboard Z)可以重新開始。

## 2.遭遇的困難與解決辦法:實際下去做

才知道其實坦克大戰需要的設定量比想像中的大，比如說判定坦克能否穿越某種地形 子彈能否打穿磚塊，牆壁等等，另外一個很大的問題就是由於載入圖片，由於某些遊戲中需求的圖片只能以 240*160 像素的 256 色 BMP 檔案。載入 所以如果載入太過於複雜的圖案 就會造成嚴重的失真現象

^^原來預定的開始畫面




^^實際載入到 GBA 的畫面

而我們針對此問題的解決辦法就是盡量別用複雜的畫面，且製作時字體盡量大且清楚，如此一來在縮放的時候才能夠比較看得清楚。

而在坦克與子彈的存取上，我是使用 circular queue 的資料結構來做存取，

如此一來，不至於浪費太多陣列空間，然後以每 300*(1/60)秒，讓敵方坦克移動與發射子彈。而我遇到的第一個問題，就是如果我將兩種坦克的圖分開讀取時，palette 會產生錯亂，而幸虧後來在製作成 16*16*10 的圖一次讀取之後，就能正常的顯示我方坦　克，敵方坦克，與發射的子彈。在地圖上，我們使用的是一個 10*15 的 char　陣列來表示能否通過此

地，為什麼是 10*15 呢，因為 160*240/(16*16) = 10 * 15，其中常常會有
沒有轉換好座標而發生 debug 困難的時候，因為移動時的座標是放大到 160*240,而陣列則只有 10*15，這是很需要注意的地方。

附注：在第一關與第二關中間，由於時間上的疏忽，並沒來的及把 stage 的圖案顯示好，
造成使用者會有誤以為當機重來的影響。

3. 組語實作:我們主要是用 C 語言完成大部分的工作 再將 C 語言經由組譯器組譯過後再去最佳化程式效率 藉此達成組語實作

## 四. 參考資料,軟體


1. http://www.aaronrogers.com/ham/Day8/day8.php
2. http://zh.wikipedia.org/w/index.php?title=%E5%9D%A6%

E5%85%8B%E5%A4%A7%E6%88%98&variant=zh-tw

3. **軟體:**程式主要設計環境是用 DEV C++ 與作業 3 提供的 Ham 完成的，而繪圖是使用小畫家配合插圖完成。