計算機組織暨組合語言期末專題

天線射擊遊戲

組員:B95701241　高新綠

B95209012　吳宜庭

一、遊戲介紹

start 畫面，按 B 鍵進入遊戲畫面 操作方式:方向左右控制方向 按 A 鍵發射

玩家扮演角色

玩家血格

敵人天線寶寶


VisualBoyAdvance-100%
File Options Cheats Tools Help

遊戲畫面


GAME OVER

遊戲結束畫面

二、 程式解析

我們是用之前作業用到的 HAM 來寫，而 HAM 有自己內建的 library—mygba.h 提供許多控制圖片、動畫的 function 使我們在寫的過程輕鬆不少。

以下介紹遊戲中主要的 code:

1. 主要參數、變數:

```
//gfx2gba -D -fsrc    -pbg.pal -t1 start.bmp bg.bmp gameover.bmp
#include "gfx/bg.pal.c"
#include "gfx/bg1.raw.c"
#include "gfx/start.raw.c"
#include "gfx/gameover.raw.c"
//gfx2gba -D -fsrc -pobject.pal -t8 t1.bmp t2.bmp t3.bmp t4.bmp anim.bmp sun.bmp
bullet.bmp heart.bmp num.bmp
#include "gfx/object.pal.c"
#include "gfx/t1.raw.c"
#include "gfx/t2.raw.c"
#include "gfx/t3.raw.c"
#include "gfx/t4.raw.c"
#include "gfx/anim.raw.c"
#include "gfx/sun.raw.c"
#include "gfx/heart.raw.c"
#include "gfx/bullet.raw.c"
#include "gfx/num.raw.c"
```

將遊戲要用的圖片載入進來

```
#define ANIM_COUNT 15          發射延遲
#define ANIM_BULLET 20         子彈數
#define HEART_NUM 5            生命值
#define T_NUM 8                一次出現的敵人次數
#define T_TIME 100             敵人出現間隔時間
#define    MIXER_FREQ 54471    音效
```

因爲遊戲需要許多參數控制爲了怕混淆以及利於修改，因此將一些重要的常數取代成有意義的名字

```
u8 anim;                                玩家指標
u8 anim_x=110;
u8 anim_y=130;                          玩家座標
int visual_count=0;                     玩家視覺暫留
u8    anim_hurt=0;                      玩家判定受傷
u8 a_press_count=0;                     攻擊延遲
u8 a_press_ava=0;                       有效攻擊
u8 bullet[ANIM_BULLET];                 子彈指標
u8 bullet_x[ANIM_BULLET]={110};
u8 bullet_y[ANIM_BULLET]={130};         子彈座標
u8 bullet_exist[ANIM_BULLET]={0};       子彈存在陣列
int bvisual_count=0;                    子彈視覺暫留
u8 sun;                                 魔王指標
u8 sun_x=5;
u8 sun_y=5;                             魔王座標
u8 sun_exist=0;                         魔王存在
int svisual_count=0;                    魔王視覺暫留
u8    sun_hurt=0;                       魔王受傷判定
int sun_life=0;                         魔王生命值
u8 heart[HEART_NUM];                    生命指標
int heart_count=0;                      生命延遲
u8 t[T_NUM];                            天線指標
u8 t_hurt[T_NUM]={0};                   天線受傷判定
u8 t_x[T_NUM]={0};                      天線座標 x
int t_count[T_NUM]={0};                 天線延遲
u8 t_y[T_NUM]={0};                      天線座標 y
u8 t_y_count=0;                         天線下降延遲
int t_exist[T_NUM]={0};                 天線存在
int number_pic[11];                     number picture
int base_score=0;                       分數
```

遊戲進行所用全域變數

```
extern const WaveData _binary_bg_raw_start;
extern const WaveData _binary_st_raw_start;
extern const WaveData _binary_attack_raw_start;
extern const WaveData _binary_hurt_raw_start;
sample_info *mysample[4];
```

主迴圈：

```
ham_Init();
ham_InitText(2);
ham_InitMixer(MIXER_FREQ);
```

程式一開始初使化

```
mysample[0]=ham_InitSample((u8*)_binary_bg_raw_start.data,_binary_bg_raw_star
t.size, _binary_bg_raw_start.freq>>10);
mysample[1]=ham_InitSample((u8*)_binary_st_raw_start.data,_binary_st_raw_start.
size,_binary_st_raw_start.freq>>10);
mysample[2]=ham_InitSample((u8*)_binary_attack_raw_start.data,_binary_attack_ra
w_start.size,_binary_attack_raw_start.freq>>10);
mysample[3]=ham_InitSample((u8*)_binary_hurt_raw_start.data,_binary_hurt_raw_s
tart.size,_binary_hurt_raw_start.freq>>10);
```

將音效文件載入

```
ham_SetBgMode(4);
```

HAM 有 0-5 的背景模式，我們挑選適用於 Bitmap 的模式 4

```
ham_LoadBGPal((void*)bg_Palette,256);
ham_LoadObjPal((void*)object_Palette,256);
ham_LoadBitmap((void*)start_Bitmap);
ham_FlipBGBuffer();
while(F_CTRLINPUT_B_PRESSED==0){}
```

無限迴圈使開始畫面停留按 B 鍵跳出

```
ham_PlaySample(mysample[1]);
ham_PlaySample(mysample[0]);
```

播放音效

```
anim=ham_CreateObj((void*)anim_Bitmap,OBJ_SIZE_32X32,OBJ_MODE_NORA
```

```
L,1,0,0,0,0,0,0,anim_x,anim_y);
ham_SetObjPrio(anim,0);
```

載入遊戲一開始即能看到的物件(ex:玩家、血量)

```
ham_StartIntHandler(INT_TYPE_VBL,(void*)&vblFunc);
```

HAM 內建重要的函式每 1/60 秒執行一次

```
    while(TRUE)
    {
    }
```

無限迴圈使遊戲一直進行

```
    return 0;
}
void vblFunc(void)
{
    ham_CopyObjToOAM();
    query_button();
    det_attack();
    det_hurt();
    update_t_gfx();
    heart_state();
    update_anim_gfx(); //update anim's state
    update_anim_pos(); //update anim's position
    update_bullet_pos();
    update_bullet_gfx();
    k+=1;
    if(k==1001) {
        k=1;
    }
    update_t_pos();
    if(base_score==200){
        sun_exist=1;
        ham_SetObjVisible(sun,1);
    }
        update_sun_pos();
```

```
        update_sun_gfx();
        ham_SyncMixer();
        ham_UpdateMixer();
        if(!mysample[0]->playing){
            ham_PlaySample(mysample[0]);
        }
}
```

vblFunc 爲所有主要的函式的驅動函式：

1.query_button(); 記錄輸入哪些動作

```
void query_button(void){
    if(a_press_count<ANIM_COUNT){
        a_press_count++;
    }else if(F_CTRLINPUT_A_PRESSED&&a_press_count==ANIM_COUNT){
        a_press_ava=1;
        a_press_count=0;
    }else if(a_press_count==ANIM_COUNT){
        a_press_count=0;
    }
    if(F_CTRLINPUT_LEFT_PRESSED){
        if(anim_x>0){
            anim_x--;
        }
    }else if(F_CTRLINPUT_RIGHT_PRESSED){
        if(anim_x<208){
            anim_x++;
        }
    }
}
```

F_CTRLINPUT_???_PRESSED:表示使用者按下哪個鍵

ANIM_COUNT:延遲攻擊

2. det_attack() 判定是否爲有效攻擊

```
void det_attack(void){
```

```
        int j=0;
        int m=0;
        int n=0;
for(i=0;i<ANIM_BULLET;i++){
         if(bullet_exist[i]==1){
              if(sun_exist){
                   m=(bullet_x[i]+sun_x)/2;
                   n=(bullet_y[i]+sun_y)/2;
      if(m<sun_x+16&&m>sun_x&&n<sun_y+16&&n>sun_y){
                     bullet_disappear(i);
                     sun_hurt=1;
                     sun_life+=1;
                     svisual_count=15;
                     base_score+=100;
                     break;
                  }
              }
      for( j=0;j<T_NUM;j++){
          if(t_exist[j]==1&&t_hurt[j]==0){
            m=(bullet_x[i]+t_x[j])/2;
            n=(bullet_y[i]+t_y[j])/2;
            if(m<t_x[j]+8&&m>t_x[j]&&n<t_y[j]+16&&n>t_y[j]){
            bullet_disappear(i);
            t_hurt[j]=1;
            t_count[i]=30;
            base_score+=10;
           }
        }
    }
    }
  }
  }
```

利用檢查 2 物件的終點是否落在被擊中物件的座標範圍判定是否爲有效攻擊如果攻擊有效則子彈消失且將敵人的受傷指標設爲 1

3. det_hurt() 判定是否有受傷

```
void det_hurt(void){
   int m=0;
   int n=0;
   for(i=0;i<T_NUM;i++){
        if(t_exist[i]==1&&t_hurt[i]==0){
             m=(t_x[i]+16+anim_x)/2;
             n=(t_y[i]+16+anim_y)/2;
            if(m>anim_x&&m<(anim_x+16)&&n>anim_y){

                 t_disappear(i);
                 anim_hurt=1;
                 break;
            }
        }
   }
   if(sun_exist){
          m=(sun_x+16+anim_x)/2;
          n=(sun_y+16+anim_y)/2;
           if(m>anim_x&&m<(anim_x+16)&&n>anim_y){
                 gameover();
           }
   }
}
```

與攻擊方法類似如果判定有受傷便使玩家受傷指標=1 如果被魔王打中直接結束遊戲

4. update_???_pos 更新物件座標

5. update_???_gfx 更新物件動畫圖片

以玩家爲例：

```
void update_anim_gfx(void){
      if(a_press_ava==1){
```

```
            visual_count=15;
            ham_UpdateObjGfx(anim,(void*)&anim_Bitmap[1024*2]);
      }else if(visual_count==-15){
            visual_count=15;
            ham_UpdateObjGfx(anim,(void*)&anim_Bitmap[1024]);
       }else if(visual_count==0){
            ham_UpdateObjGfx(anim,(void*)&anim_Bitmap[0]);
       }else if(anim_hurt==1){
            anim_hurt=0;
             visual_count=5;
             ham_UpdateObjGfx(anim,(void*)&anim_Bitmap[1024*3]);
        }
        visual_count--;
 }
```

一般情況下每 0.25 秒轉換一次圖片

而當攻擊或受傷時則會轉換成其他圖片

```
6.   if((base_score>200&&base_score%2000==0)||base_score==200){
            sun_exist=1;
            ham_SetObjVisible(sun,1);
      }
```

當分數達 200 以上每增加 2000 分魔王會出現

```
7. ham_SyncMixer();
            ham_UpdateMixer();
            if(!mysample[0]->playing){
                  ham_PlaySample(mysample[0]);
            }
ham_SyncMixer();
ham_UpdateMixer();  這兩個funciton每個frame都要執行一次
if(!mysample[0]->playing){
ham_PlaySample(mysample[0]);
}
```

每frame檢查一次，如果背景音樂播完了就再放一次

三、檢討

1.圖片：因爲當初使用的圖片是直接從網路上剪下來所以再轉成bmp時十分不清楚。

2.程式：遊戲的一些參數以及判定攻擊方式還有進步的空間因爲時間上的關係所以來不及參考一些遊戲的演算法以及架構，所以參數都是自己大概設再看結果去調整。

3.音效：HAM的轉檔程式只支援ACM(8位元單聲道)再加上播放的頻率也有限制，所以效果實在不好(有點理解爲什麼大家在用gba模擬器玩遊戲時都把音樂關掉)

四、分工

遊戲架構設計:高新綠、吳宜庭

HAM函式理解：高新綠、吳宜庭

報告：高新綠、吳宜庭

圖片製作:高新綠

程式除錯：吳宜庭