期末專題-太鼓達人

第六組：林銘宇、楊琇閔、賴宥瑋、曾唯丞

研究動機

我們組內討論過後，決定在原本太鼓達人三機版作業的基礎上，再加上額外的功能，作為這次期末的專題。我們將運用I2C連線作為連接三個Arduino板的方法，因為我們覺得相比UART，I2C較便於使用。

各Arduino板連接裝置

- Master Arduino :

  一個LCD板、兩個按鈕(選歌、開始遊戲)

- Slave Arduino 1 :

  一個LCD板、兩個按鈕(左右打擊)、一個蜂鳴器

- Slave Arduino 2 :

  一個LCD板、兩個按鈕(左右打擊) 、一個蜂鳴器

主要功能

- 雙人同時遊玩
- 選歌時試聽
- 隨機譜面
- 計算combo數
- 雙機即時分數更新
- 勝負判斷
- 重複遊玩

選歌時試聽

```
184  ISR(TIMER1_COMPA_vect)
185  {
186    TCNT1 = 0; // 重製計時器
187    if(now_song == 1 && is_song_select == 0) //當目前是第一首歌且還沒選歌時
188    { //播小蜜蜂
189      tone(speakerPin, frequencies[melody_bee[play_song++]], 125); //彈奏音符
190      if(play_song == scoreLen_bee) play_song = 0; //當播完小蜜蜂時，重新播放(重製play_song)
191    }
192    else if(now_song == 2 && is_song_select == 0) //當目前是第二首歌且還沒選歌時
193    { //播小星星
194      tone(speakerPin, frequencies[melody_star[play_song++]], 125); //彈奏音符
195      if(play_song == scoreLen_star) play_song = 0; //當播完小星星時，重新播放(重製play_song)
196    }
197    else if(now_song == 3 && is_song_select == 0) //當目前是第三首歌且還沒選歌時
198    { //播火車快飛
199      tone(speakerPin, frequencies[melody_train[play_song++]], 125); //彈奏音符
200      if(play_song == scoreLen_train) play_song = 0; //當播完火車快飛時，重新播放(重製play_song)
201    }
202  }
```

Master 按鈕1-選歌

```
216  void press_right() //右按鈕硬體中斷，選歌
217  {
218    unsigned long currentTime = millis(); //目前時間
219
220    // 防彈跳 和 顯示目前是在第幾首(顯示白點在歌曲前)
221    if (currentTime - lastDebounceTime > debounceDelay)
222    {
223      if(digitalRead(right_button_pin) == LOW) // 檢查按鈕是否仍然處於按下狀態
224      {
225        tft.fillCircle(10, 30 + 80 * (now_song - 1), 3, ILI9341_BLACK); //清除原本的歌
226        now_song += 1; //歌曲目+1
227        if (now_song == 4) now_song = 1; //因為沒有第四首，所以當now_song為4時，讓他變成第1首
228        tft.fillCircle(10, 30 + 80 * (now_song - 1), 3, ILI9341_WHITE); //顯示目前的歌
229
230        lastDebounceTime = millis();  // 儲存按下按鈕的時間
231      }
232    }
233    play_song = 0; //重新播放新的音樂
234  }
```

Master 按鈕2-開始遊戲

```
236  void press_left() //左按鈕硬體中斷，確定歌
237  {
238    unsigned long currentTime = millis(); //目前時間
239
240    // 防彈跳
241    if (currentTime - lastDebounceTime > debounceDelay)
242    {
243      if(digitalRead(left_button_pin) == LOW) // 檢查按鈕是否仍然處於按下狀態
244      {
245        is_song_select = 1;
246        tft.fillScreen(ILI9341_BLACK); //螢幕變黑
247        if(now_song == 1) //選第一首
248        {…
257        }
258        else if(now_song == 2) //選第二首
259        {…
268        }
269        else if(now_song == 3) //選第三首
270        {…
279        }
280        can_send = 1; //讓master可以傳給slave
281        lastDebounceTime = millis();  // 儲存按下按鈕的時間
282      }
283    }
```

Master-傳送訊號(I2C)

```
104   if(is_song_select == 1 && can_send == 1) //當選歌了，且master可以傳送給slave1
105   {
106     Wire.beginTransmission(slave1_address);
107     if(now_song == 1) Wire.write('1');
108     if(now_song == 2) Wire.write('2');
109     if(now_song == 3) Wire.write('3');
110     Wire.endTransmission();
111   }
112
113   if(is_song_select == 1 && can_send == 1) //當選歌了，且master可以傳送給slave2
114   {
115     Wire.beginTransmission(slave2_address);
116     if(now_song == 1) Wire.write('1');
117     if(now_song == 2) Wire.write('2');
118     if(now_song == 3) Wire.write('3');
119     Wire.endTransmission();
120     can_send = 0; //停止傳送
121   }
```

Master-接收訊號(I2C)

```
123  if(is_song_select == 1 && can_request_1 == 1) //當選歌了，且可以接受slave1傳來的訊息
124  {
125    Wire.requestFrom(slave1_address, 2); //跟slave1要數據
126    while(Wire.available())
127    {
128      int score = Wire.read(); //讀取分數
129      int combo = Wire.read(); //讀取combo
130      int pre_score; //儲存上個分數
131      if(score == 70) //當分數70時，代表遊戲結束
132      {
133        String text = "Song Complete";
134        int text_length = text.length() * 6 * 2; // "Song Complete"長度
135        int x = (tft_width - text_length) / 2; //文字置中
136        can_request_1 = 0; //停止要求slave1
137        tft.setTextSize(2);
138        tft.setCursor(x,220);
139        tft.setTextColor(ILI9341_WHITE);
140        tft.print("Song Complete");
141      }
142      else if(score != pre_score) //當分數有變化時
143      {
144        show_p1_score(score-1); //印出分數
145        pre_score = score; //儲存分數
146        score1 = score; //儲存分數
147      }
148      if(pre_combo_1 != combo) //當combo有變化時
149        show_p1_combo(combo); //印出combo
150    }
151  }
```

Slave-接收訊號(I2C)

```
196  void receiveEvent(int bytes) //接受到master傳來的訊息時執行
197  {
198    reset(); //因為可以重玩，所以每次近來都要重置
199    tft.setTextSize(2); //設定文字大小
200    tft.setCursor(10,140); //設定文字位置
201    tft.setTextColor(ILI9341_WHITE); //設定文字顏色
202    tft.print("Score 0"); //印出初始分數
203
204    tft.setTextSize(3); //設定文字大小
205    now_song = Wire.read(); //讀取master傳來的訊息
206    if(now_song == '1') //當收到 '1' 時，代表為小蜜蜂
207    {
208      tft.setCursor(70, 15);
209      tft.println("Little bee");
210    }
211    else if(now_song == '2') //當收到 '2' 時，代表為小星星
212    {
213      tft.setCursor(61, 15);
214      tft.println("Little star");
215    }
216    else if(now_song == '3') //當收到 '3' 時，代表為火車快飛
217    {
218      tft.setCursor(115, 15);
219      tft.println("Train");
220    }
221    set_drum(); //設定音符顏色
222
223    //繪製箭頭
224    tft.drawLine(15, 70, 15, 85, ILI9341_WHITE);
225    tft.fillTriangle(12, 85, 18, 85, 15, 90, ILI9341_WHITE);
226  }
```

2024/1/7

Slave-Time Interrupt

```
111  ISR(TIMER1_COMPA_vect) //硬體中斷
112  {
113    TCNT1 = 0; // 重製計時器
114    if(now_song == '1' || now_song == '2' || now_song == '3') count ++; //計算目前播放到哪
115    for(int i = 15; i <= tft_width; i += 40) //重製音符
116    {
117      tft.drawCircle(i, 120, 15, ILI9341_BLACK);
118    }
119    for(int i=0 ; i<song_len ; i++){
120      if(now_song == '1') //小蜜蜂
121      {
122        if(cir_position_bee[i] != 0 && cir_position_bee[i] <= tft_width) //顯示音符在LCD範圍(0 ~ 320)
123        {
124          if(cir_position_bee[i] == 15) //當出現在要彈奏的位置
125          {
126            tone(speakerPin, frequencies[melody_bee[i]], 125); //彈奏音符
127            now_color = drum_color_bee[i]; //同步該音符顏色
128            if(appear_bee[i]) is_there_note = 1; //判斷是否有音符(用以計算分數和combo)
129            else is_there_note = 0;
130            count_combo(); //計算combo
131          }
132          if(appear_bee[i]) //如果有音符(frequencies = 0)則繪製音符
133          {…
136          }
137        }
138        if(cir_position_bee[i] > 15) //遞減每個為彈奏過的音符
139        {
140          cir_position_bee[i] -= 40;
141        }
142        else cir_position_bee[i] = 0;
143      }
144      if(now_song == '2'){…
166      }
167      if(now_song == '3') //火車快飛
```

2024/1/7

Slave-硬體中斷

```
250  void press_right() //硬體中斷，當按下右鍵執行
251  {
252    if(now_color == 1 && is_there_note == 1) //顏色為紅色時(正確)
253    {
254      show_score(); //更新分數
255      now_color = -1; //當+1分後，馬上讓目前顏色變-1，以免分數重複計算
256      strike = 1; //打擊正確時strike為1
257    }
258  }
259
260  void press_left() //硬體中斷，當按下左鍵執行
261  {
262    if(now_color == 0 && is_there_note == 1) //顏色為藍色時(正確)
263    {
264      show_score(); //更新分數
265      now_color = -1; //當+1分後，馬上讓目前顏色變-1，以免分數重複計算
266      strike = 1; //打擊正確時strike為1
267    }
268  }
```

Slave-隨機譜面

```
277     int now = 15; //音符初始位置
278     for(int i=0 ; i<song_len ; i++)
279     {
280       int drum_color = random(0, 6); //取亂數，奇數為紅色，偶數為藍色
281       now += 40; //音符的間隔
282       if(now_song == '1') //小蜜蜂
283       {
284         cir_position_bee[i] = now; //音符位置設定
285         if(melody_bee[i] == 0) appear_bee[i] = false; //當為frequencies = 0，不顯示音符(設appear為false)
286         else appear_bee[i] = true; //其他要顯示(設appear為true)
287         if(drum_color % 2 == 0) drum_color_bee[i] = 0; //當drum_color為偶數時,音符為藍色
288         else drum_color_bee[i] = 1; //當drum_color為奇數時,音符為紅色
289       }
290       else if(now_song == '2') //小星星
291       {…
297       }
298       else if(now_song == '3') //火車快飛
299       {…
305       }
306     }
307   }
```

2024/1/7

Slave-計算combo

```
328  void count_combo() //計算combo數
329  {
330    if(is_there_note == 1) //如果有音符
331    {
332      if(strike == 1) //有打到
333      {
334        pre_combo = combo;
335        combo += 1;
336      }
337      else if(strike == 0) //沒打到
338      {
339        pre_combo = combo;
340        combo = 0;
341      }
342      else combo = combo;
343      strike = 0; //重置strike
344      print_combo(combo); //印出combo數
345    }
346  }
```

Slave-傳送訊號(I2C)

```
228  void requestEvent() //接受到master要求的訊息時執行
229  {
230    //當歌曲播放完畢時
231    if( (count==scoreLen_bee+2 && now_song == '1') || (count==scoreLen_star+2 && now_song == '2') || (count==scoreLen_train+2 && now_song == '3'))
232    {
233      Wire.write(70); //寫70給master(因為每首歌曲長度不超過70，所以分數<=70，當master收到70代表結束)
234      Wire.write(combo); //寫combo數給master
235      tft.setTextSize(2);
236      String text = "Song Complete";
237      int text_length = text.length() * 6 * 2; // "Song Complete"長度
238      int x = (tft_width - text_length) / 2; //文字置中
239      tft.setCursor(x,220);
240      tft.setTextColor(ILI9341_WHITE);
241      tft.print("Song Complete");
242    }
243    else //當歌曲還未播放完畢時，同步score和combo給master
244    {
245      Wire.write(score);
246      Wire.write(combo);
247    }
248  }
```

Master-決定勝負

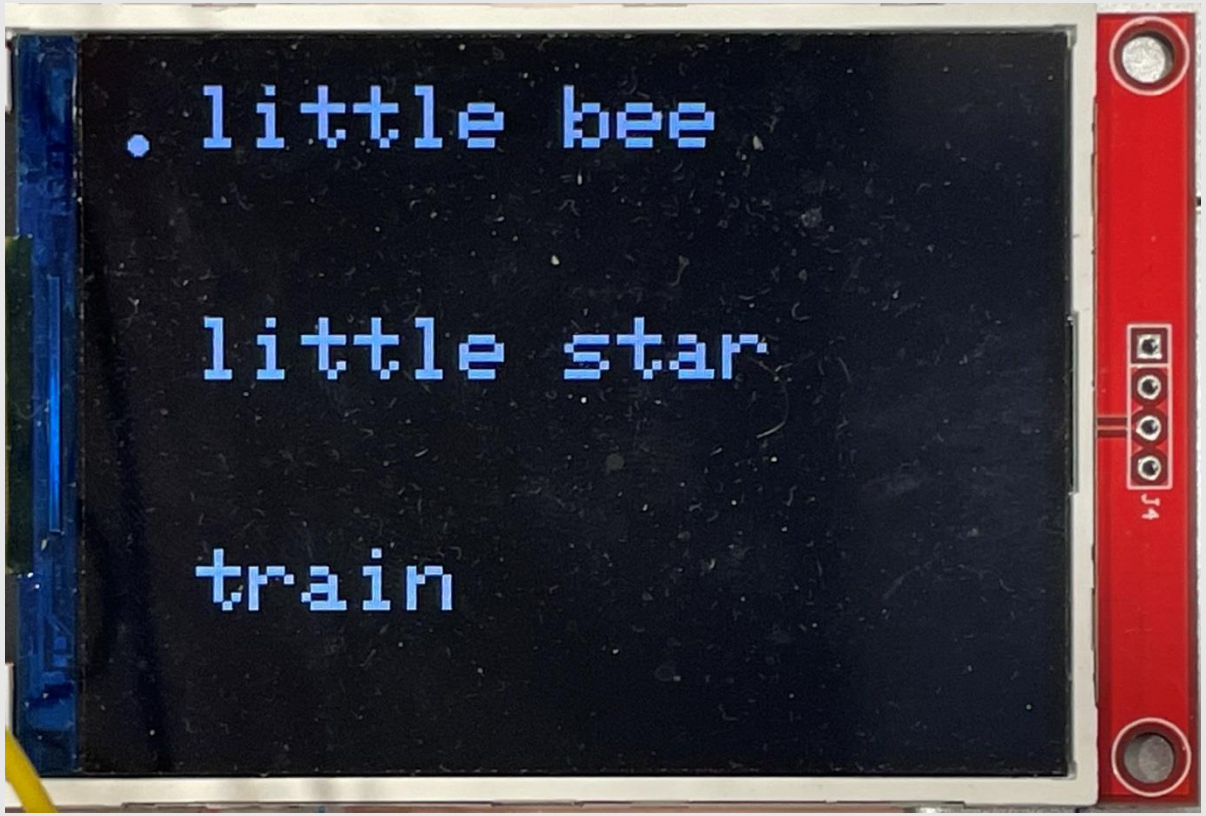
```
380  void pk() //決定勝負
381  {
382    if(score1 > score2) //p1分數大於p2
383    {
384      String text = "Player 1 win !"; //p1贏
385      int text_length = text.length() * 6 * 2; // "Little bee"長度
386      int x = (tft_width - text_length) / 2; //文字置中
387      tft.setCursor(x, 40);
388      tft.print("Player 1 win !");
389    }
390    else if(score1 < score2) //p1分數小於p2
391    {
392      String text = "Player 2 win !"; //p2贏
393      int text_length = text.length() * 6 * 2; // "Little bee"長度
394      int x = (tft_width - text_length) / 2; //文字置中
395      tft.setCursor(x, 40);
396      tft.print("Player 2 win !");
397    }
398    else//p1分數等於p2
399    {
400      String text = "Draw !"; //平手
401      int text_length = text.length() * 6 * 2; // "Little bee"長度
402      int x = (tft_width - text_length) / 2; //文字置中
403      tft.setCursor(x, 40);
404      tft.print("Draw !");
405    }
406    delay(3000);
407    reset(); //重置，可重玩
408  }
```

重置-master & slave

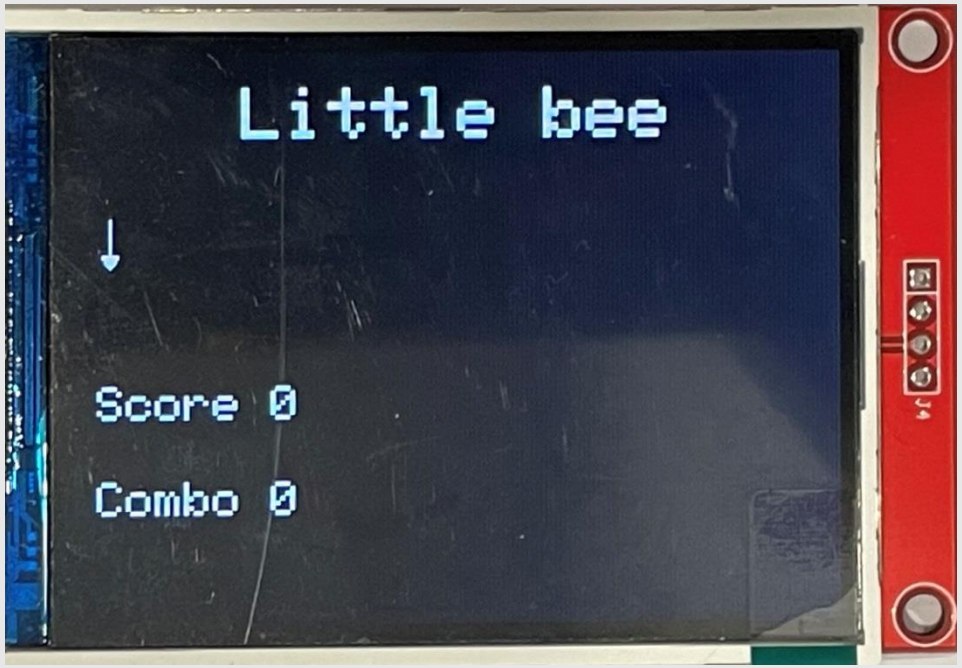
```
410  void reset() //重置
411  {
412    tft.fillScreen(ILI9341_BLACK); //螢幕變黑
413    show_song(); //顯示歌曲
414    tft.fillCircle(10, 30, 3, ILI9341_WHITE); //初始選歌為第一首
415    is_song_select = 0; //是否選歌(0還沒選，1選了)
416    can_request_1 = 1, can_request_2 = 1;
417    now_song = 1; //目前選歌曲目(共3首)
418    can_send = 0; //是否可傳送信號給slave(0不行 1可以);
419    play_song = 0; //重置目前播放到第幾個音
420  }
```

```
365  void reset() //重置
366  {
367    tft.fillScreen(ILI9341_BLACK); //螢幕變黑
368    score = 0; //分數歸0
369    combo = 0; //combo歸0
370    strike = 0; //打擊歸0
371    count = 0; //記數歸0
372    now_song = '0'; //回到沒選歌狀態
373  }
```

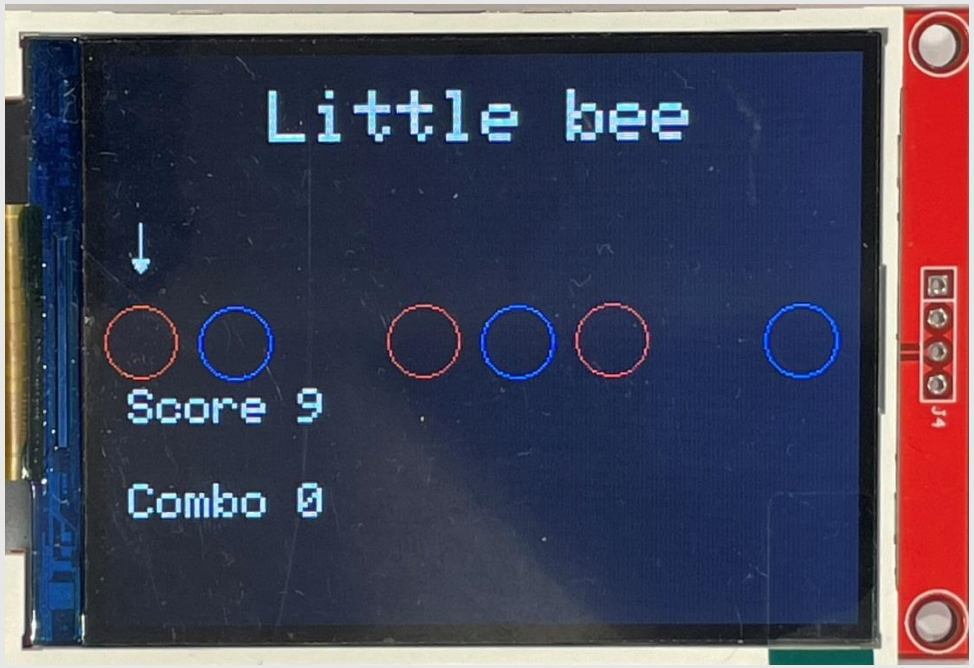
Master Arduino-初始畫面



Slave Arduino-遊戲畫面

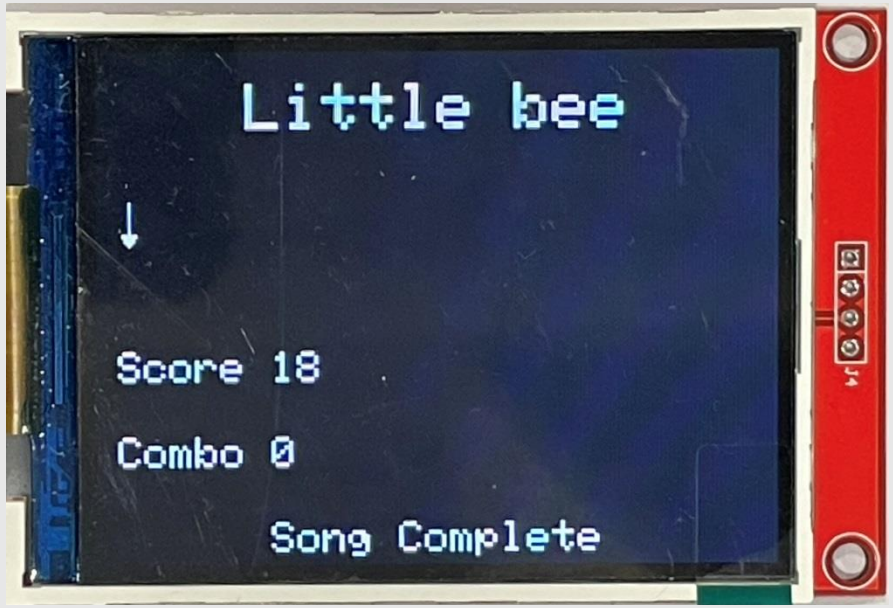


▲ 起始畫面

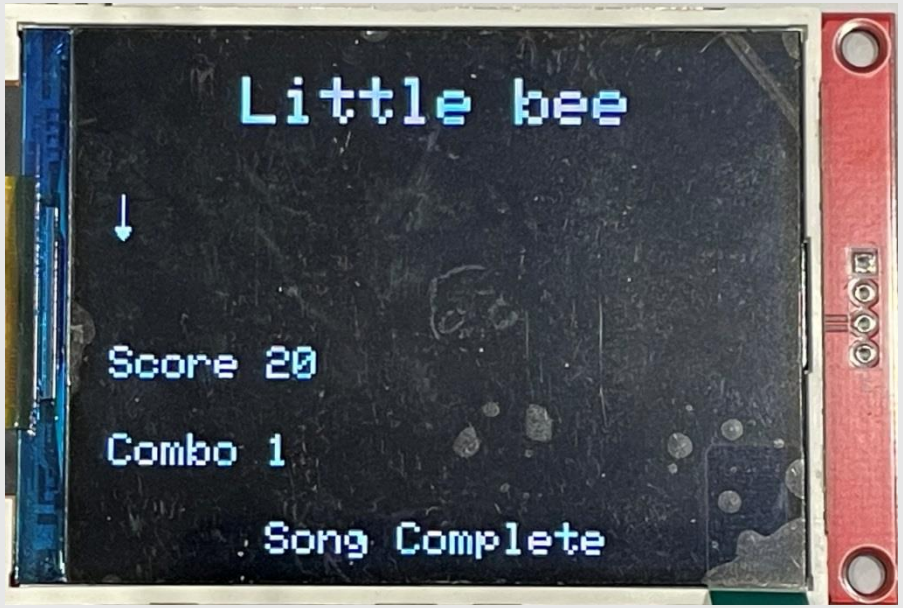


▲ 遊戲中畫面

Slave Arduino-結束畫面



▲ Slave 1



▲ Slave 2

Master Arduino-結算畫面

分工表

姓名	工作內容	占比
S1154007 賴宥瑋	歌曲畫面顯示（master和slave，包括選歌，播放音樂）	25%
S1154009 林銘宇	I2C傳輸及顯示譜面並讓譜面隨機	25%
S1154010 楊琇閔	硬體中斷（分數及combo數）	25%
S1154021 曾唯承	勝負判斷及重新整理畫面（讓整個遊戲可以進一直進行下去）	25%