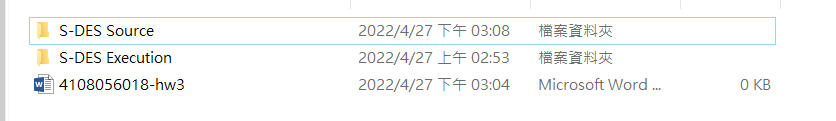
說明:

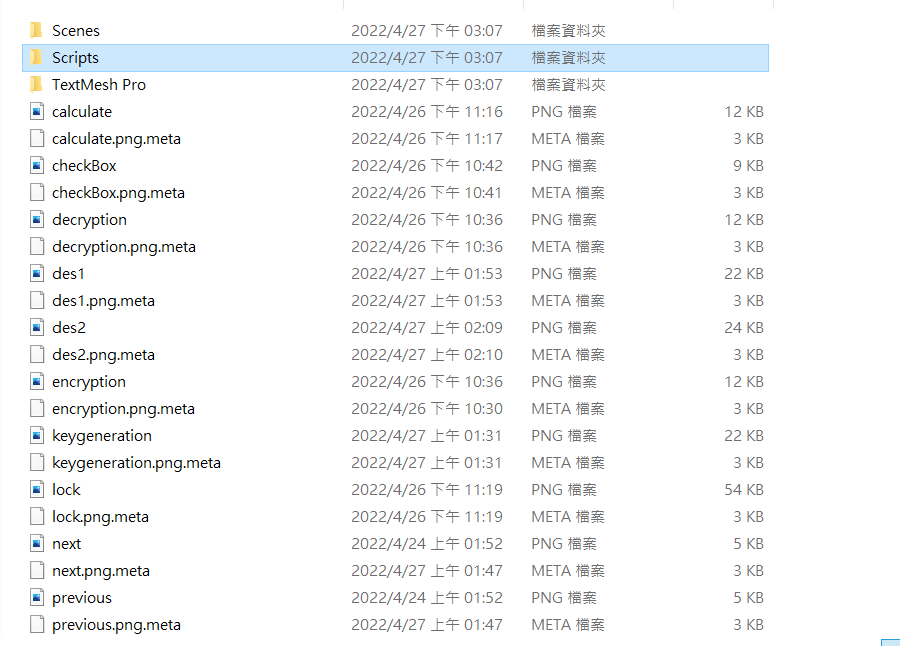
本次S-DES的實作是使用Unity來製作介面，內部演算法則透過c#來完成。



其中S-DES Source為source code與素材

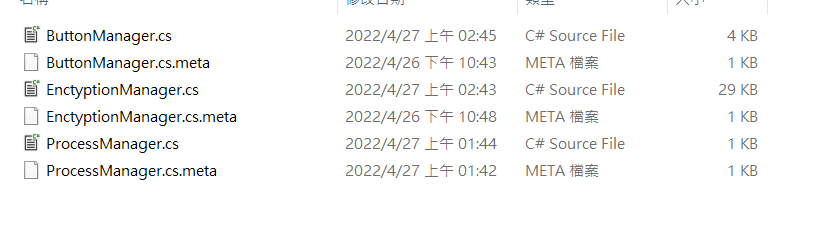


點進去後再點Assets

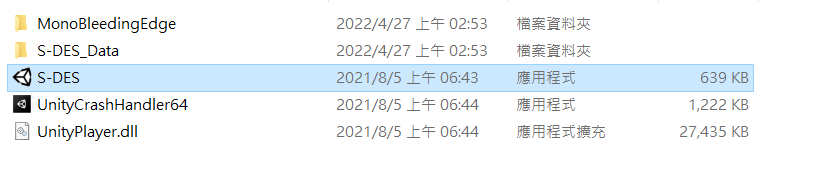


裡面會有用到的素材

之後再點入Scripts

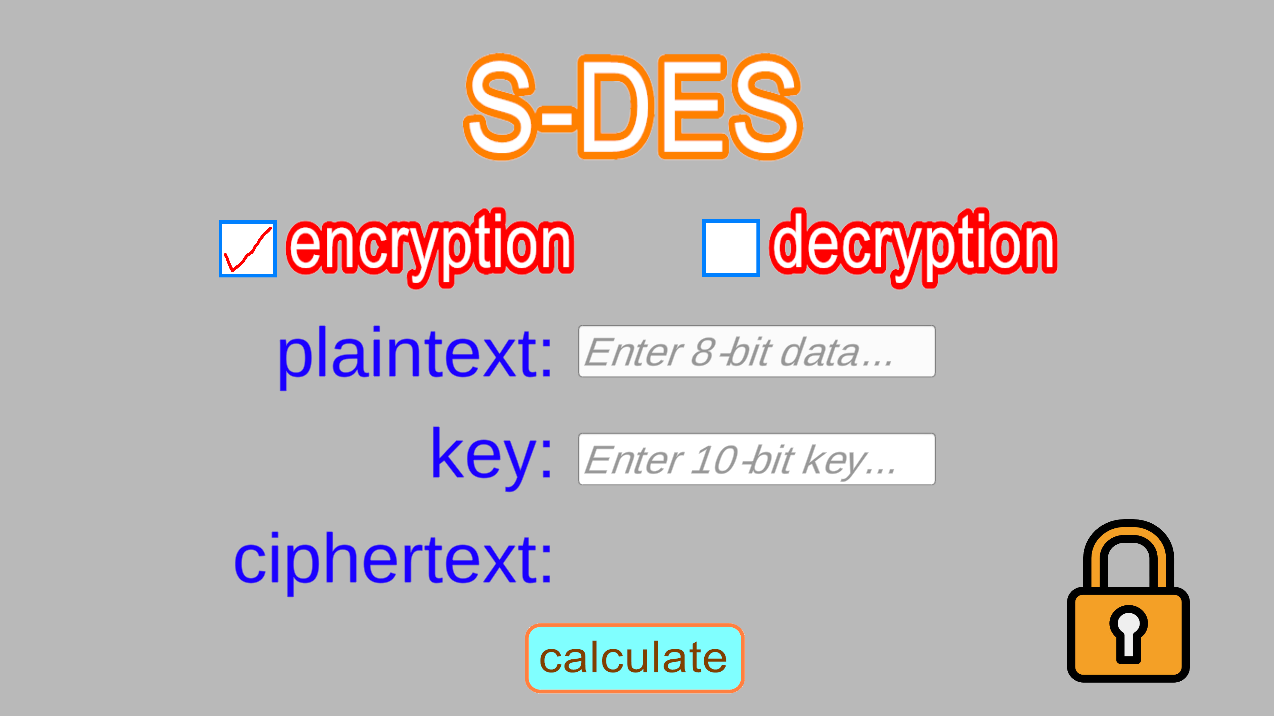


這邊則是實際撰寫的程式碼，其中EncryptionManager.cs是實際加解密的程式，而另外兩個則是介面互動的程式。

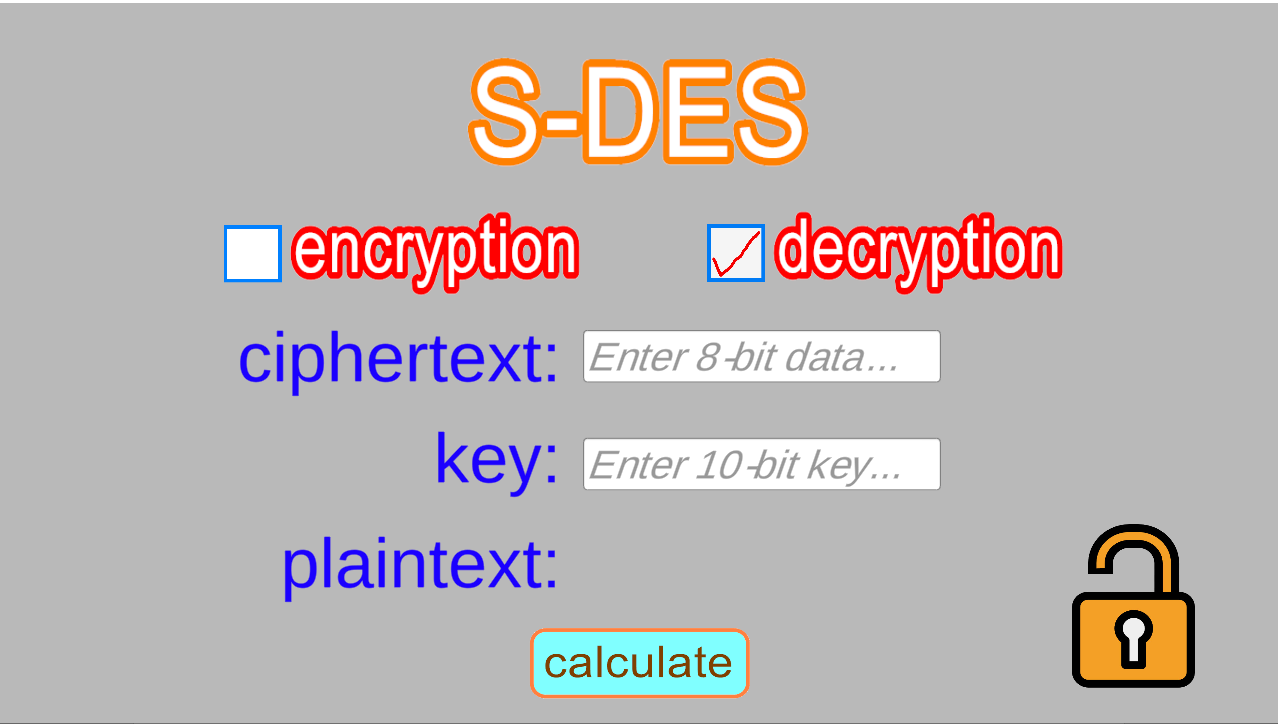


而S-DES Execution點進去後會看這些東西，點入S-DES.exe即可執行。

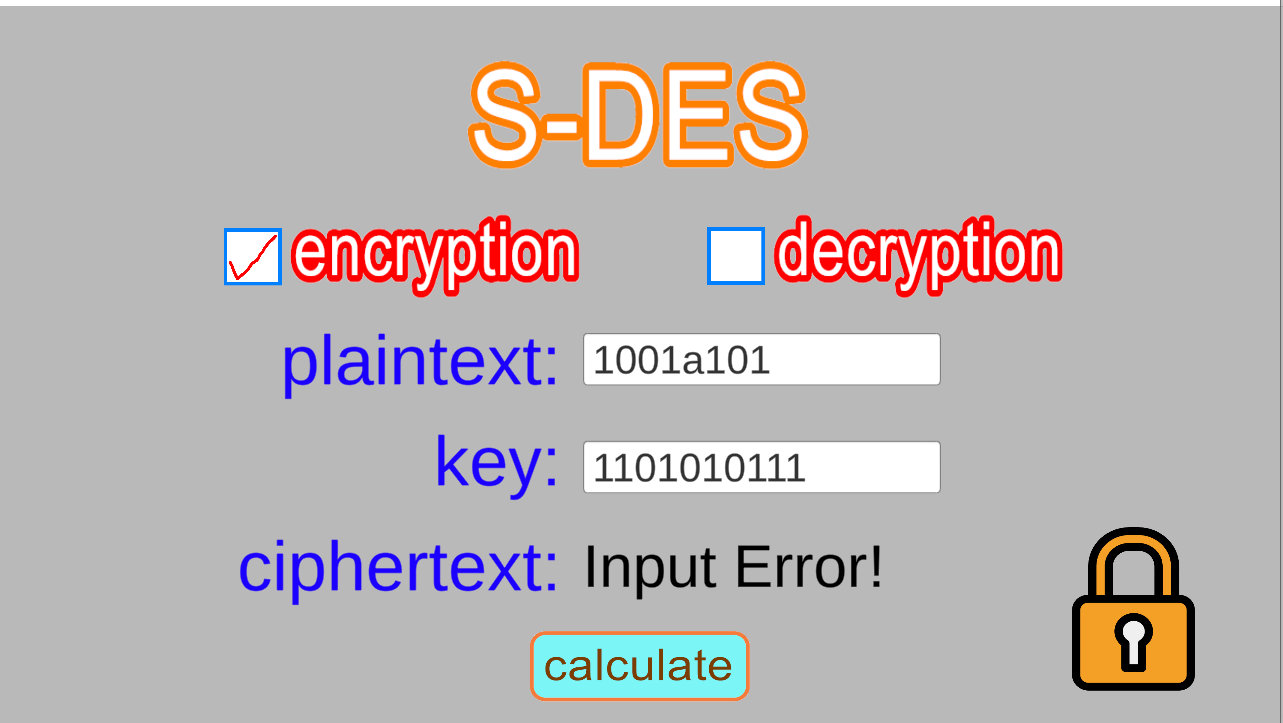
使用方式:



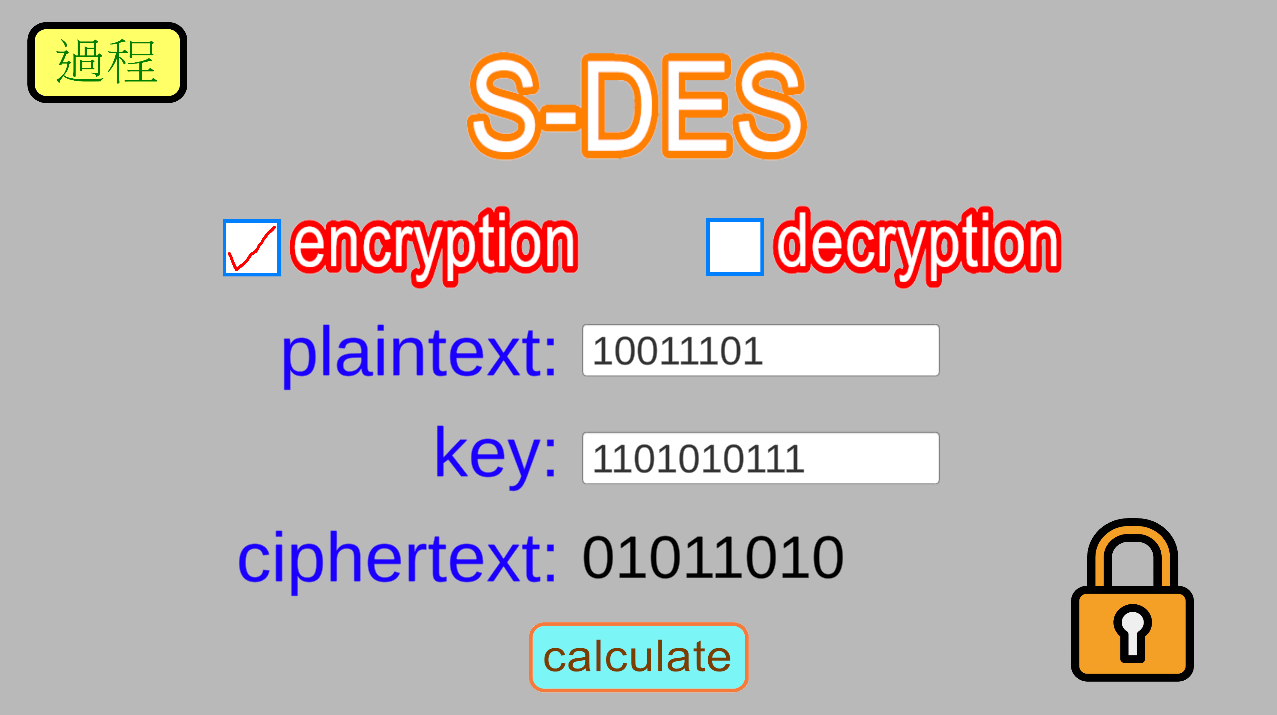
我們能夠選擇要使用加密或是解密，如果要使用加密的話，就會要求輸入plaintext與key，並輸出ciphertext。



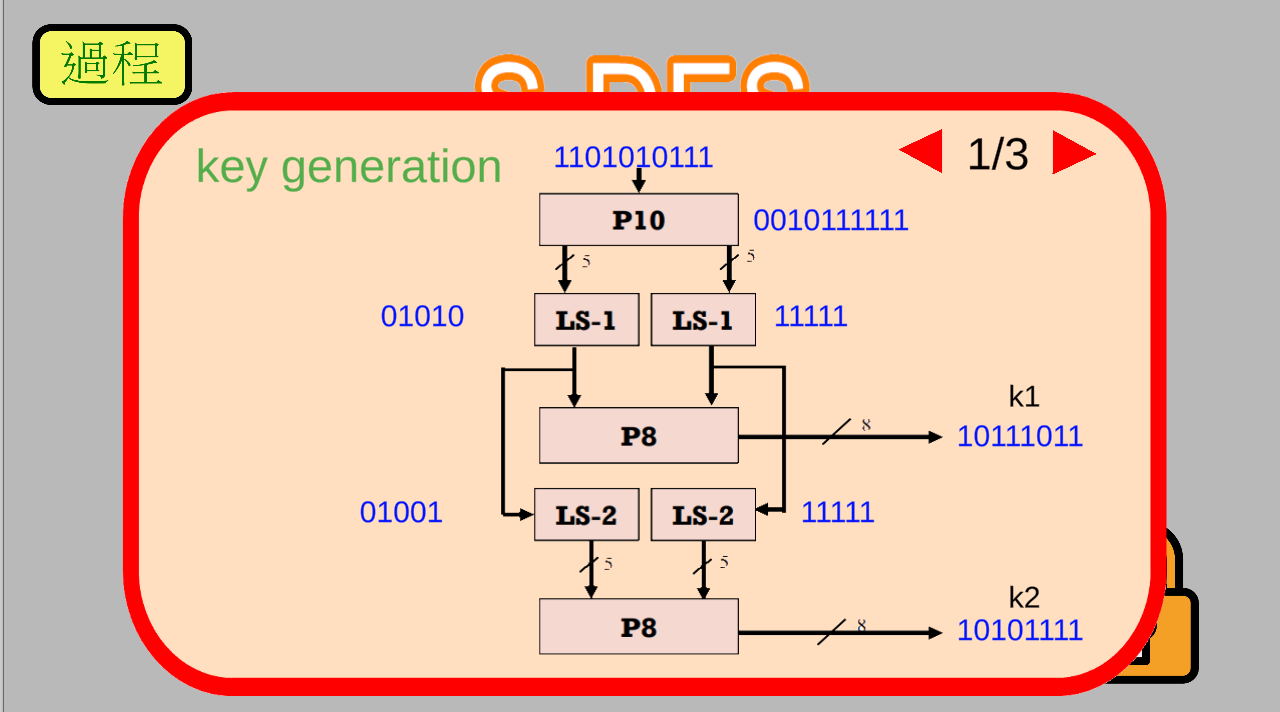
如果要使用解密的話，就會要求輸入ciphertext與key，並輸出plaintext。



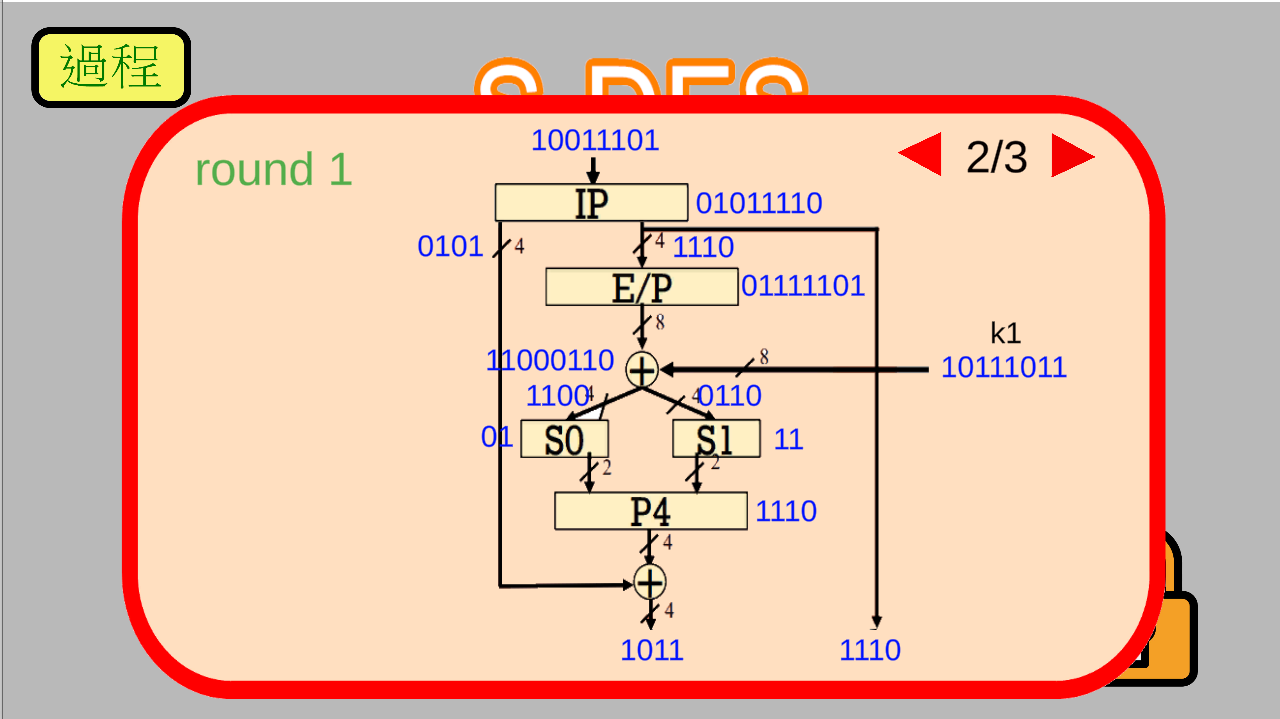
這邊有做輸入的防呆，若字串長度不夠且不是binary string的話就會產生錯誤。



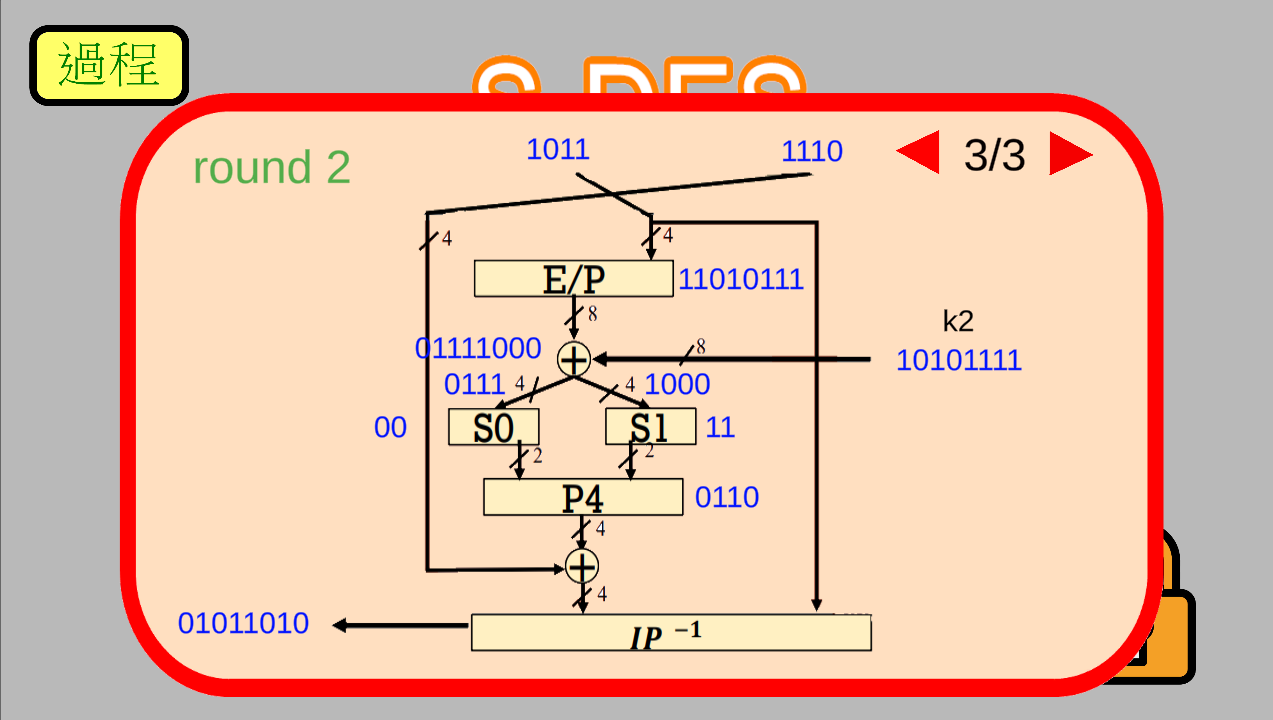
輸入完成後按下下方的calculate按鈕即可獲得輸出，並且左上角會彈出過程的按鈕。



點按鈕後就會跳出一個視窗，第1頁為key的generation。

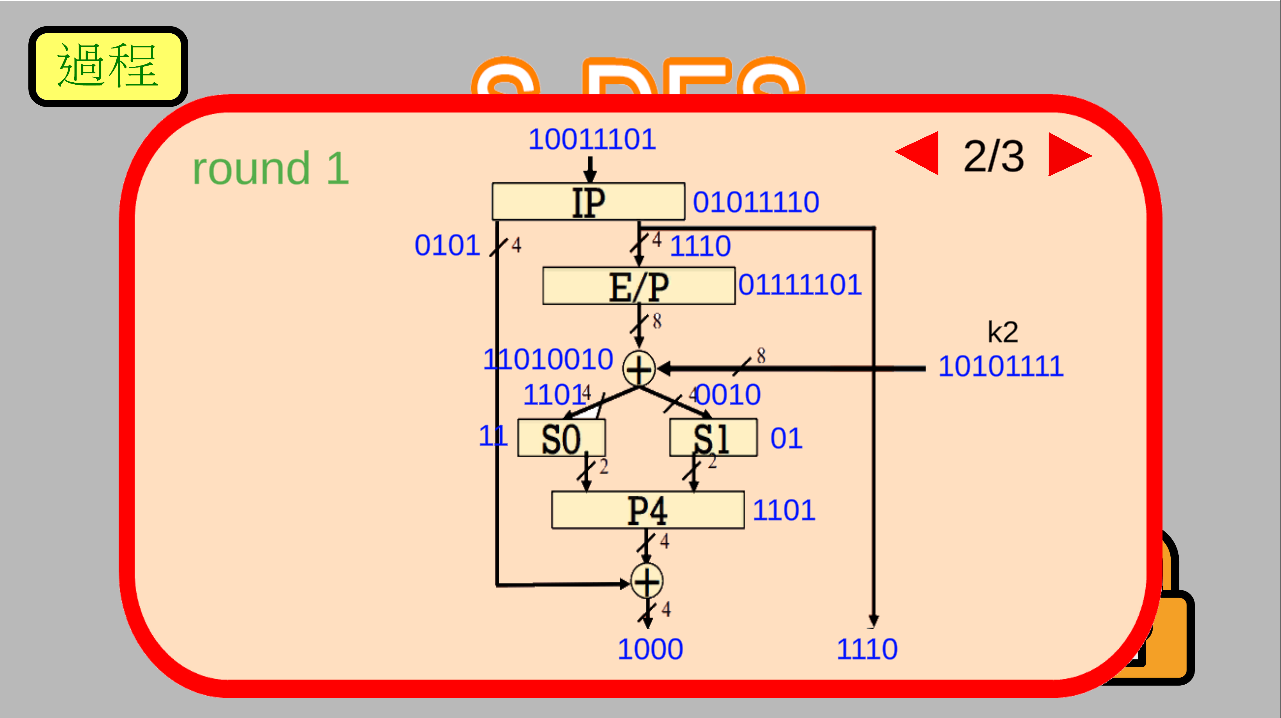


第2頁為round 1的加解密過程，因為是加密因此round 1使用k1。

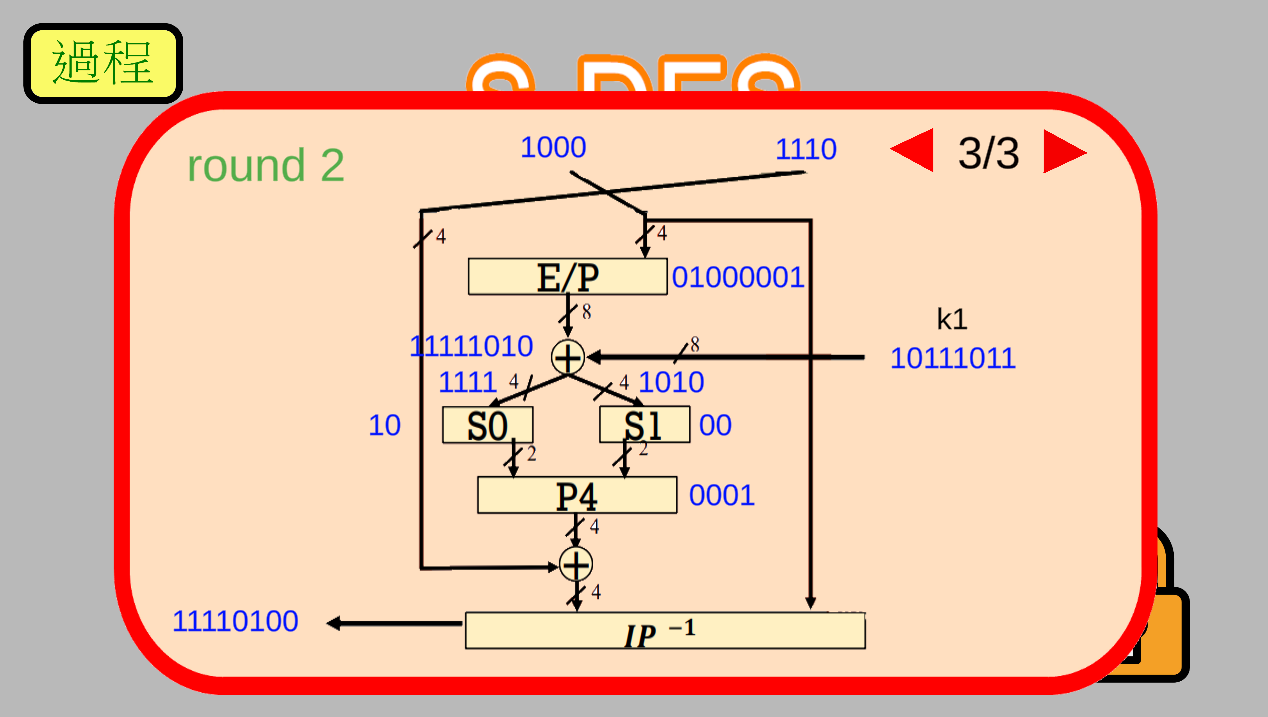


第3頁則是round 2的加解密過程，因為是加密因此round 2使用k2。

上述的藍色binary string都會隨著輸入而改變。

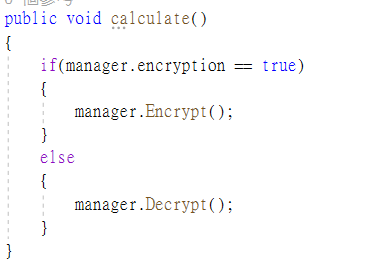


如果是解密的話round 1就會使用k2，而不是k1。

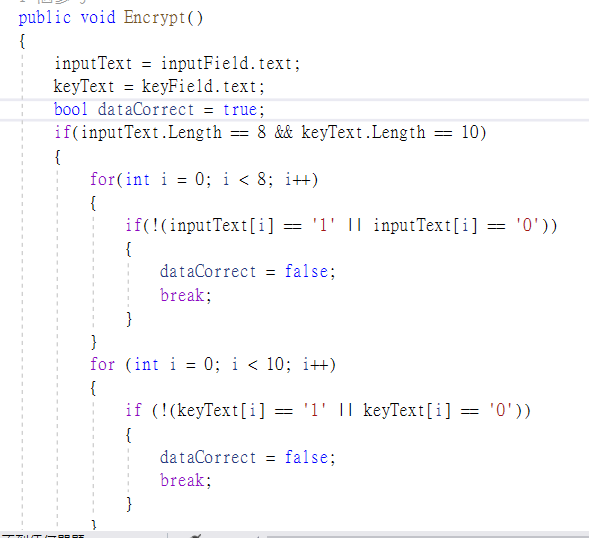


解密的round 2也是使用k1，而不是k2。

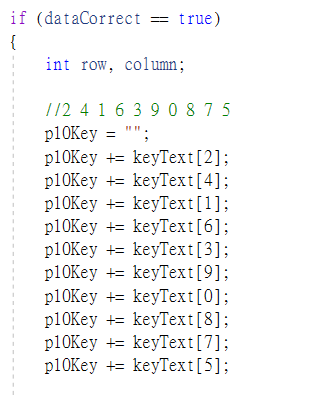
程式運作流程:



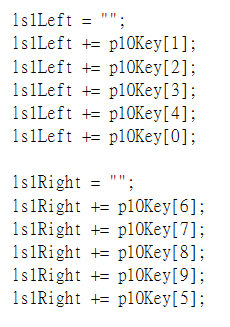
當按下claculate按鈕時會根據目前的狀況來選擇要執行加密或是解密。



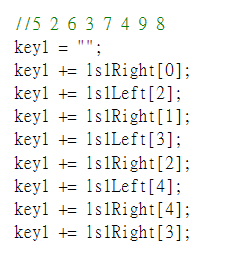
一開始會先進行輸入防呆的偵測，看長度是否符合，以及是否只有0跟1。



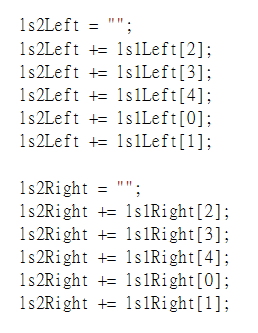
如果輸入正確就開始執行，一開始會先對key進行p10。



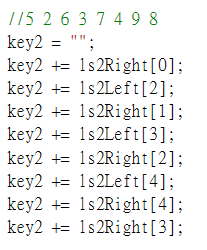
再來做LS-1。



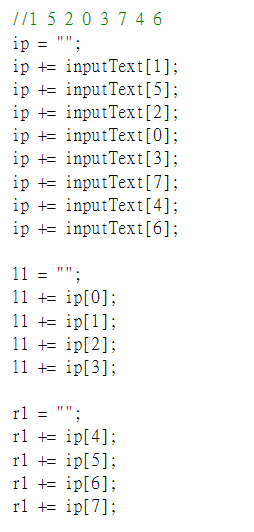
再來做p8產生key1。



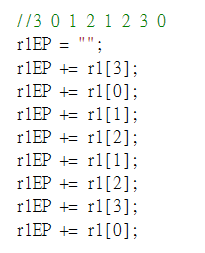
之後做LS-2。



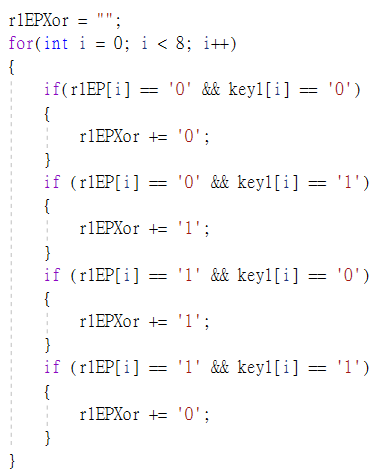
再做P8產生key2。



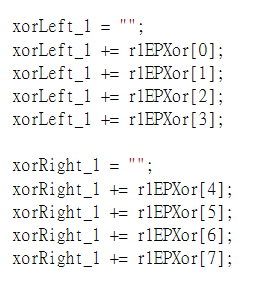
一開始先做IP並切成左半與右半。



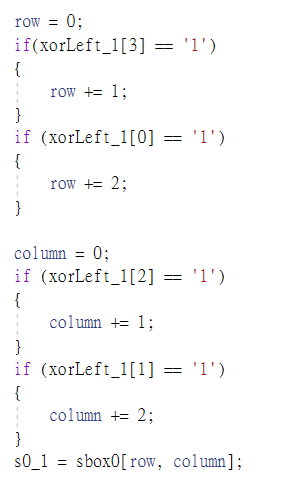
右半做EP。



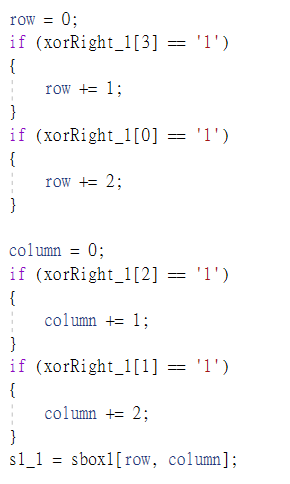
EP的結果與key1進行xor。



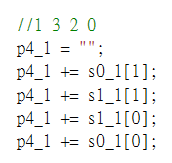
結果再切成左半與右半。



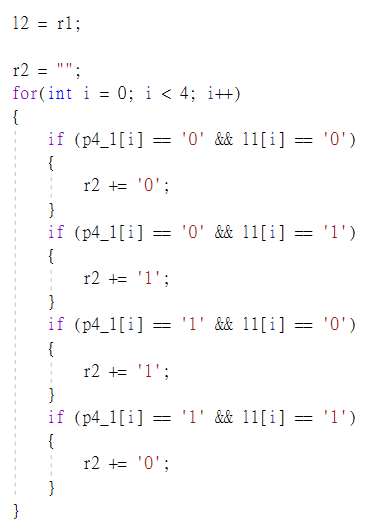
左半做S0。



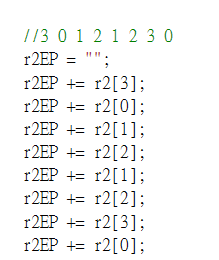
右半做S1。



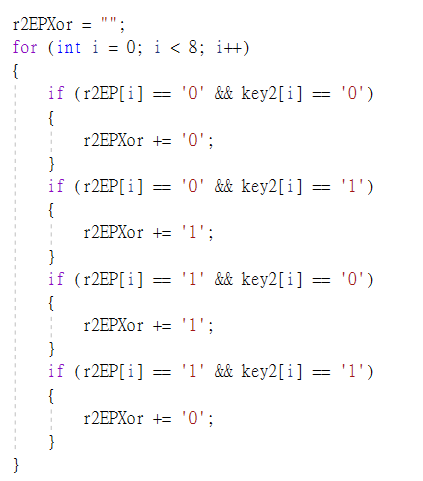
之後再做p4。



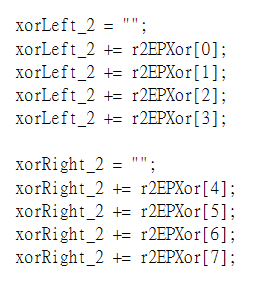
準備進行第2輪，l2會等於r1，r2會等於p4 xor l1。



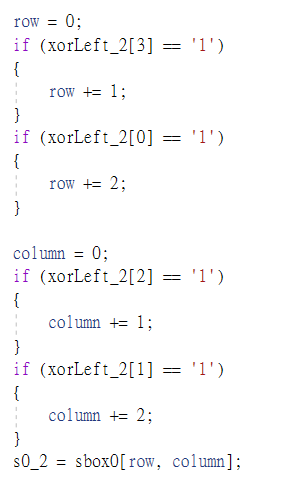
開始第2輪，r2先做EP。



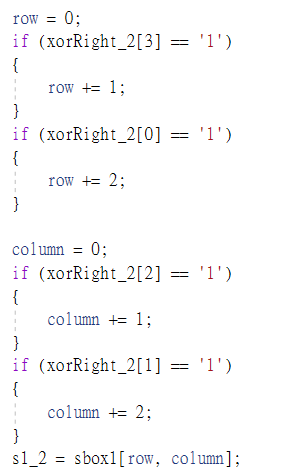
之後將EP的結果與key2做xor。



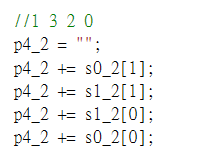
將結果分成左半與右半。



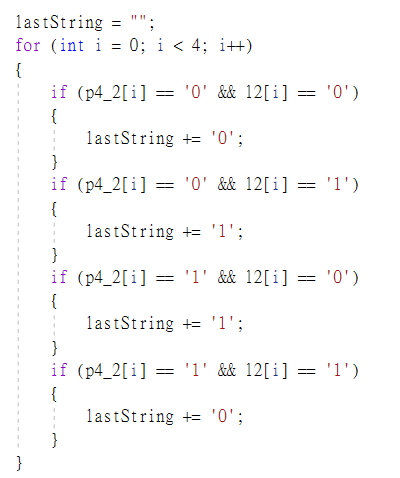
左半做S0。



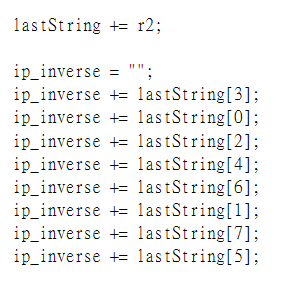
右半做s1。



之後做p4。

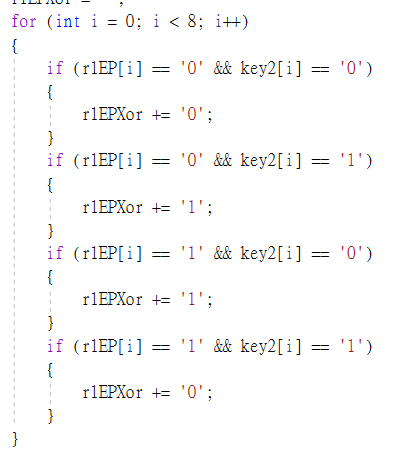


再將p4與l2做xor。



最後將結果串上r2，並再做IP inverse即可獲得最終結果。

解密基本上也是一樣的流程，只是改成第一次先使用k2，第二次再使用k1。



最後則是把流程圖的變數更新

