4] LabWork conducting

\* difficulties encountered and resolving measures:

剛開始發現怎麼接都沒辦法顯示，以為是自己接錯了，之後拿switch那邊的去測七段顯示器也沒有顯示任何的東西，後來才發現我們忘記插電了

原本把P1接到七段顯示器那區的下排，把P2接到七段顯示器那區的上排，發現顯示怪怪的，去檢查一下code認為應該總要有個東西對才對，所以懷疑是接線的問題，後來把P1接到七段顯示器那區的上排，把P2接到七段顯示器那區的下排就成功了。

\* phenomena observed and explanations

Code上發現:

按照lab2.2給的code可以執行，而結果會是543210閃6次，接著是右邊3個5(XXX555)和左邊3個1(111XXX)輪著閃6次。透過code，發現R1 is position, R2 is the number。

其實一開始的543210並不是同時閃，而是先用R1設定是0，用R2設定位置為最右邊那一格，顯示0了之後，把R1從0變成1，再用R2設定位置為右邊第二格，以此類推，便能顯示543210。之所以能同時顯示是因為instruction執行的速度很快，因為數字間顯示的時間差的極短所以我們會認為是同時顯示。

Delay問題:

===III=== : 3個5(XXX555)會顯示的時間，要是沒有這一行，會幾乎看不到右邊的那3個5(XXX555)

===JJJ=== : 3個1(111XXX)會顯示的時間，要是沒有這一行，會幾乎看不到左邊的那3個1(111XXX)

===LLL=== : 543210與下次5432110中間間隔全暗的時間，要是沒有這一行，會幾乎看不到閃爍而是認為543210一直亮著

七段顯示器:

七段顯示器那區的上排的8bits中，只有7bits是控制位置，分別是控制小數點，第一個數字，第二個數字，.....，第六個數字。第幾個bit為0的時候第幾個位置才會亮，1為不亮，MSB似乎什麼都不影響，像是FFH(11111111)代表全暗

七段顯示器那區的下排的8bits中，只有4bits會控制數字

push-pop問題:

delay1的push-pop不可以省略，因為R1等等還會用到，而不是先被重新賦值，所以必須要特別先push之後再pop回來

delay2的push-pop可以省略，因為R3不會被用到，所以不需要特別先push之後再pop回來，而R1和R2因為很快就會被重新賦值，重新賦值完才會用到，所以不用特別用push-pop保存

delay問題2:

===KKK=== : 讓543210可以同時出現的關鍵，要是沒有這一行，會幾乎看不到543210因為閃太快

要是把===KKK===的delay1的時間乘上1000倍，543210看起來不太像是同時出現，而是像接續出現

5] self-evaluation

\* things learnt

學會最基本的流程，了解如何用code去控制七段顯示器(數字和位置)，還有要怎麼利用delay和視覺暫留以達到同時顯示543210，已及了解到更多push-pop的用法

\* retrospections raised to oneself

在開始用code前記得hardware已經都準備好了，不要在忘記接電源了。要debug的時候不要一次改太多東西，一次變動一點就好比較好之道bug發生在哪。還有有些事情不要太直覺，像是不要總認為1是亮0是暗，七段顯示器就是相反。記得要保存直的時候用push-pop