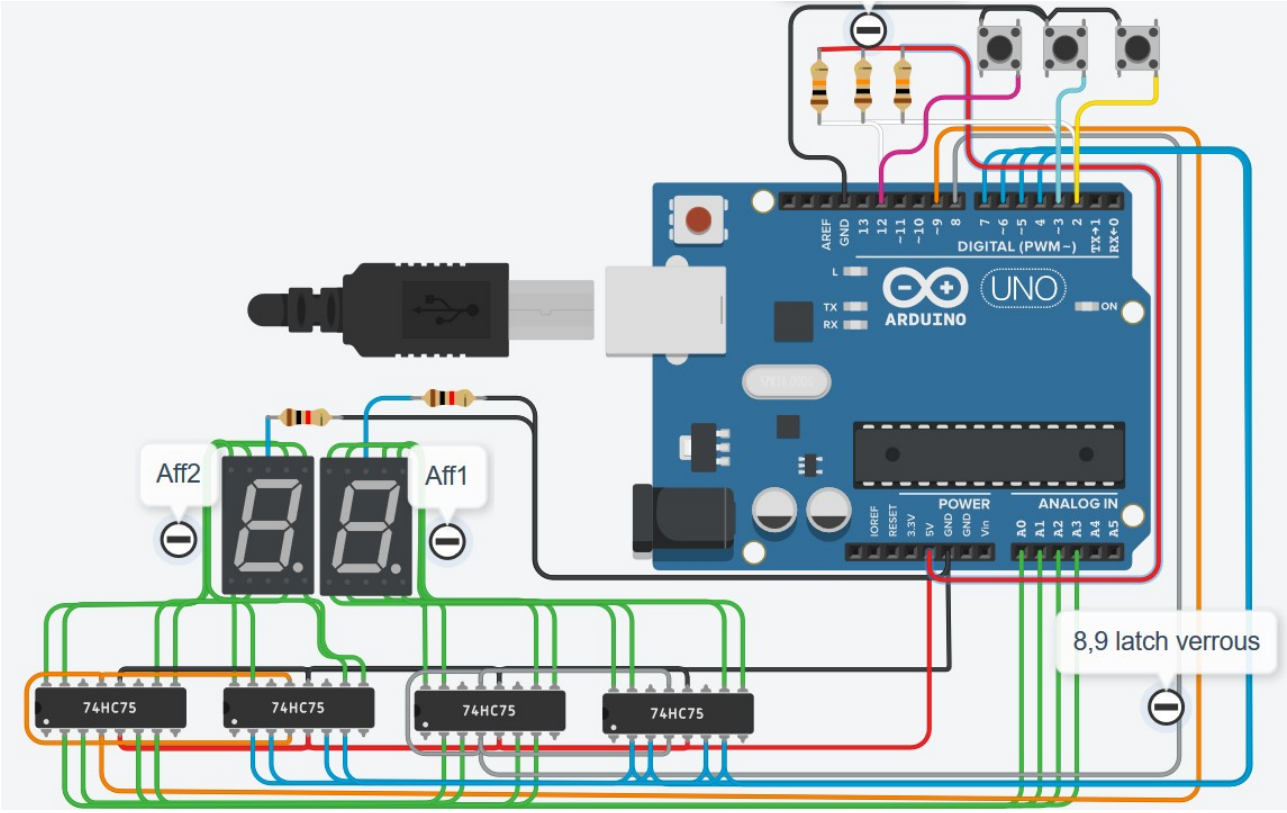
Circuit 1 : l'objectif est de faire un programme qui affiche un nombre à 2 chiffres sur les afficheurs 7S. L'appui sur les BP permet d'incrémenter/décrémenter/remettre à 0 la valeur. Un appui long (>0,75s) sur le BP incrément permet d'incrémenter la valeur tous les 0,25s.

Analyse du circuit :

Résistances pull-up sur les broches 2,3,12 (pas nécessaire d'activer les pull-up internes)
Les sorties A0,A1,A2,A3 contrôlent les segments A,B,C,D de l'afficheur 1 ou 2
Les sorties 4,5,6,7 contrôlent les segments E,F,G,PT de l'afficheur 1 ou 2
Il y a 4 verrous 74HC75 4 bits (8bits pour l'afficheur 1, 8bits pour l'afficheur 2) pour pouvoir mémoriser une valeur sur un afficheur. On gère les 2 afficheurs avec les mêmes 8 sorties.



1) Ecrire la partie configuration de toutes les entrées /sorties utilisées dans le setup()

2) Ecrire une fonction **`void Disp(byte chiffre,byte numAff)`** pour afficher un chiffre 0-9 sur l'un des 2 afficheurs. On pourra utiliser le tableau suivant pour faire l'association entre chiffre (indice) et segments à activer. Ex: pour afficher 5, il faut activer A,C,D,F,G, soit (0110 1101)b ou 0x6D. La valeur 0x6D est à l'indice 5.

```
// table de correspondance valeur <-> segments
// tab7S[10] éteint l'afficheur, tab7S[11] affiche le point seulement
byte tab7S[12]={0x3F,0x06,0x5B,  // à compléter};
```

Note: pour pouvoir afficher une valeur, il faut présenter la bonne valeur sur l'entrée des verrous (74HC75), puis verrouiller avec les commandes LATCH (sorties 8,9). La donnée présente est mémorisée quand la commande LATCH passe de 1 à 0. Notez bien que ceci permet de gérer les 2 afficheurs à l'aide de seulement 8 sorties du microcontrôleur.

3) Ecrire une fonction **`void Print7S(byte nombre)`** pour afficher un nombre 00-99 sur les 2 afficheurs. Si nombre<10, éteindre l'afficheur 2.

4) Exploiter les interruptions INT0 et INT1 pour pouvoir incrémenter (entrée 2) ou décrémenter (entrée 3) la valeur affichée sur appui sur les BP.

5) Exploiter l'interruption PinChange pour pouvoir remettre la valeur à 0 à l'aide du BP de l'entrée 12. Attention, l'interruption PinChange se produit sur tous les changements. Il faudra donc tester, dans l'ISR, la valeur de l'entrée pour savoir si c'est un front descendant (1->0) ou ascendant (0->1).

6) Détection Appui Long (entrée 2) : on souhaite qu'un appui prolongé (>0,75s) sur l'entrée 2 permette ensuite d'incrémenter la valeur tous les 0,25s tant que le BP est enfoncé.

On propose d'utiliser le Timer1 en mode CTC pour savoir si il y a un appui prolongé. Pour celà, vous pouvez utiliser la fonction **`SetTimer1CTC()`** fournie dans le polycopié pour la configuration. Il convient d'essayer de régler le Timer1 de sorte d'avoir une interruption COMPA tous les 0,25s (il faut choisir prescaler et valeur de comparaison). Si l'on détecte que le BP de l'entrée 2 est enfoncé pendant 3 ISR successives, on peut dès lors incrémenter la valeur affichée à chaque ISR suivante. Ceci jusqu'à ce que le bouton soit relâché.

Note: sur le site TinkerCad, l'exécution reste une simulation qui ne s'effectue pas en temps réel. Même si le programme fonctionne bien, il se peut que les incréments n'aient pas lieu à raison de 4 fois par seconde en temps réel. En revanche, le temps de simulation (qui défile parfois moins vite) est affiché.