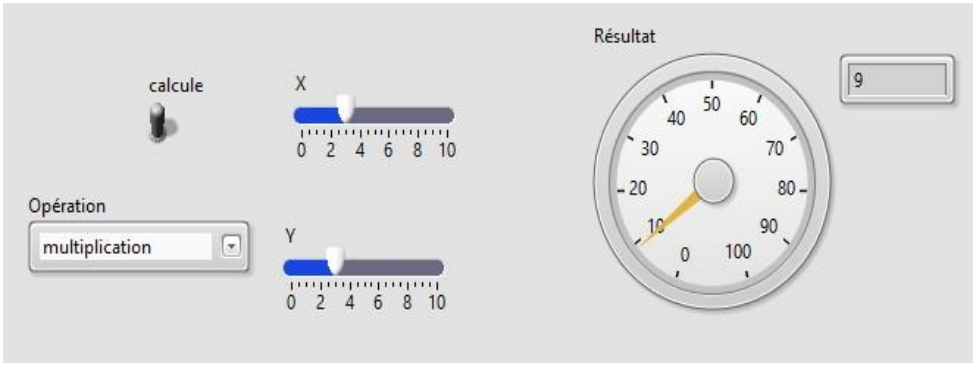
TP 4- Structure évènement, séquentielle et machine d'état

I. Objectif :

- Créer et manipuler les structures de types événement et séquentielle.
- Etudier et tester une structure de machine d'état

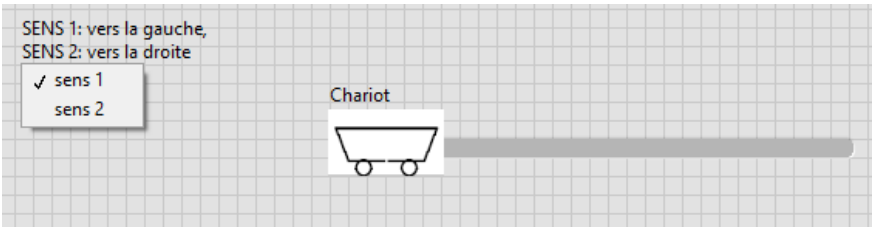
II. Travail demandé :

Manipulation 1 :



Ecrire un programme qui permet de réaliser, à chaque appui sur le bouton *calcule*, une opération mathématique (addition, soustraction, multiplication, division) et afficher le *Résultat*.

Manipulation 2 :



Ecrire un programme qui permet de déplacer un chariot dans deux sens, droite et gauche en utilisant une structure évènement.

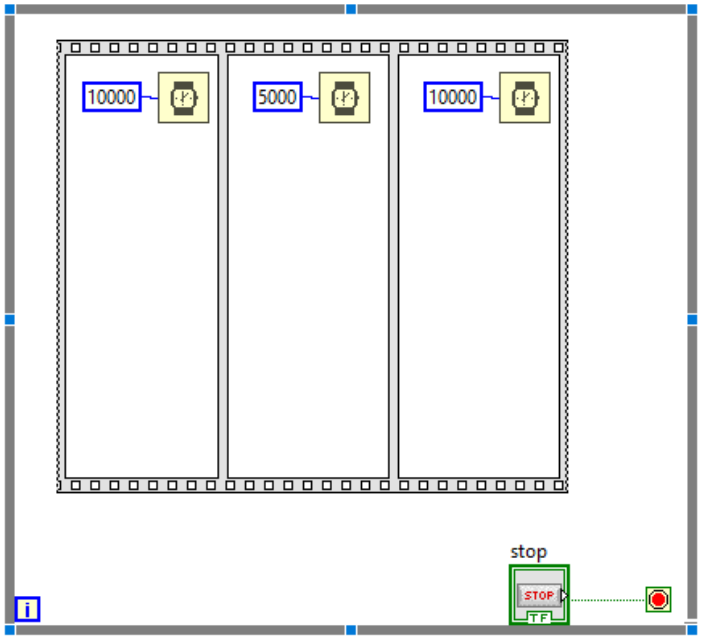
Manipulation 3 :

On se propose dans cette application de simuler et gérer un feu de signalisation d'un carrefour routier. Le panneau de signalisation est représenté avec des Leds colorées (rouge, orangé, vert) et il fonctionne sous comme suit :

- Le rouge est allumé pendant 10s
- Le feu orangé pendant 5s
- Le feu vert est allumé pendant 10s

1. Compléter le diagramme de bloc suivant en utilisant une structure séquentielle.

   N.B : On peut utiliser un nœud de priorité pour changer la valeur attribuée à un indicateur.



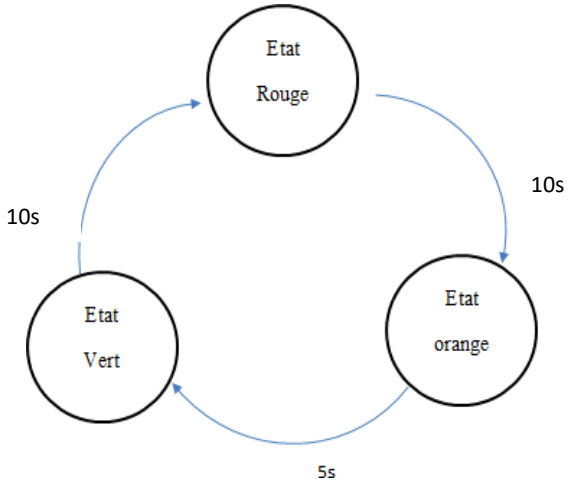
2. On souhaite maintenant créé une autre solution du feu de signalisation avec la structure de machine d'état.

La machine d'état est l'une des architectures principales dans le développement d'applications sous LabVIEW. En effet, elle permet de résoudre les difficultés de la programmation séquentielle pour les applications industrielles telles que la supervision et la commande des procédés. Principalement elle est décrite par :

- Une boucle While : permet l'exécution en continu du code
- Une structure Condition : chacun de ses états va reprendre par complète analogie un état du diagramme fonctionnel.
- La liste des états est représentée par les valeurs d'une énumération, qui sera déclarée comme une définition de type. Ce qui signifie :
  - ✓ Définition de type, la mise à jour du modèle sera répercutée sur toutes ses instances dans le code. Donc des états pourront être rapidement ajoutés et appliqués partout dans le code.
  - ✓ La commande de type énumération, ou *énum*, lorsqu'elle est câblée au terminal de sélection d'une structure Condition, fait correspondre les conditions à la liste des éléments de l'énum (ou énumération). Le code est donc auto-décrit. Si un état n'est pas présent dans la structure, le code ne sera pas exécutable.

- Un registre à décalage : permet d'implémenter « chaque état définit l'état suivant ». Il est câblé au terminal de sélection de la structure condition, soit le symbole « ? ». À chaque itération de la boucle While, le contenu du registre à décalage définira donc, via la structure condition, l'état courant à exécuter. À la fin de l'exécution de chaque état, une constante de l'énumération d'états est câblée sur la sortie du registre à décalage, définissant ainsi le prochain état à exécuter.

Le diagramme d'état décrivant le fonctionnement du feu de signalisation comporte 3 états (Rouge, Orange, Vert):



3. Compléter le diagramme de bloc pour la structure de machine d'état du feu de signalisation