Compétition de robotique *FIRST*® 2024

Guide de programmation LabVIEW pour KitBot

1 Table des matières

2 Présentation du document

Ce document vous guidera à travers la façon de préparer votre KitBot 2024 et de le faire fonctionner en utilisant l'exemple de code LabVIEW fourni. Pour éviter la duplication de contenu, ce document fait fréquemment référence à la documentation WPILib [NdT La documentation WPILib est disponible en français] pour des étapes spécifiques du développement. En plus de vous rendre fonctionnel avec le code fourni, ce document passera en revue la structure de ce code afin que vous puissiez comprendre comment il fonctionne. Enfin, nous passerons en revue certains des changements les plus probables que vous voudrez peut-être apporter au code et fournirons des exemples concrets de la façon d'apporter ces modifications.

Pour commencer avec l'exemple de code, ou pour y apporter certaines modifications, une compréhension minimale de LabVIEW est nécessaire. Le code et les exemples de modification fournis vous permettront d'avancer avec suffisamment de structure. Pour comprendre la procédure pas à pas, ou pour apporter des modifications non décrites dans ce document, une compréhension plus complète de LabVIEW est probablement requise. Le didacticiel NI LabVIEW et les pages NI LabVIEW pour FRC sont des ressources pour vous aider à démarrer.

Ce document, et l'exemple de code fourni, suppose l'utilisation des contrôleurs SPARK MAX fournis dans le kit de lancement des recrues.

3 Prise en main de votre code KitBot

3.1 Câblage de votre robot

Utilisez le document Introduction au câblage d'un robot FRC sur WPILib pour vous aider à câbler votre robot. Quelques notes spécifiques au KitBot 2024 :

- Le KitBot 2024 n'utilise pas de pneumatique. Vous pouvez ignorer les instructions concernant le concentrateur pneumatique ou le module de commande pneumatique, sauf si vous ajoutez de la pneumatique à votre design.
- Afin d'utiliser les mêmes ID pour le fonctionnement avec PWM et CAN, le code KitBot 2024 n'utilise pas le port PWM 0. Câblez les ports PWM en fonction des ID dans Constants.java (Gauche = 1,2; Droite = 3,4) ou modifiez les constantes pour refléter votre câblage.
- Le KitBot 2024 contient deux moteurs supplémentaires non inclus dans le document de câblage de base. Câblez-les de la même manière que les moteurs du système motopropulseur. Si vous utilisez les PWM, connectez le moteur de l'admission (le plus près du centre du robot) au port PWM 5 et le moteur du lanceur (le moteur plus proche de l'extérieur du robot) au port PWM 6.

3.2 Configuration du matériel et de l'environnement de développement

Avant de pouvoir charger du code et tester votre robot, vous devrez configurer votre matériel (roboRIO, radio, etc.) et configurer votre environnement de développement. Suivez les étapes 2 à 4 du guide d'Installation des composants logiciels de WPILib pour tout configurer et vous assurer que vous pourrez déployer un projet de robot de base.

Si vous utilisez les PWM, assurez-vous que les 6 MAX SPARK sont en mode « brushed » (voa). Lorsqu'alimenté, la DEL doit clignoter en jaune ou en bleu, pas en magenta ou en cyan. Pour changer le mode, vous pouvez soit maintenir le bouton Mode enfoncé pendant 3 secondes, soit utiliser la connexion USB et le client matériel REV. Vous pouvez également vérifier les modes neutres, frein ou libre (voa). Il est recommandé de régler les moteurs de l'admission et du lanceur en mode libre (jaune clignotant). Pour modifier le mode, appuyez brièvement sur le bouton Mode (moins de 3 secondes) ou utilisez la connexion USB et REV Hardware Client. Il n'y a pas de recommandation spécifique pour les moteurs de la propulsion, mais vous voudrez probablement que les 4 moteurs de propulsion soient sur le même mode, vous voudrez peut-être essayer de piloter avec chaque réglage pour décider ce que vous préférez.

3.3 Ouverture de l'exemple KitBot 2024

L'exemple de code pour KitBot 2024 est fourni dans des fichiers zip individuels pour chaque langage sur la page Web du KitBot (voa). Pour ouvrir le code LabVIEW :

1. Téléchargez et décompressez l'exemple de code LabVIEW. Assurez-vous de décompresser ou de copier vers un emplacement permanent, pas dans un dossier temporaire.

2. Le moyen le plus simple d'ouvrir le code est de tirer parti des associations de fichiers par défaut configurées par LabVIEW. Ouvrez le dossier décompressé et localisez le fichier « 2024 Kitbot » et double-cliquez dessus. Ce fichier est le fichier Projet et double-cliquez dessus pour lancer LabVIEW 2023 avec le projet ouvert.
3. Si vous souhaitez ouvrir le projet à partir de LabVIEW à la place (par exemple, si plusieurs versions de LabVIEW sont installées sur votre ordinateur), lancez LabVIEW 2023 et sélectionnez File->Open Project. Accédez au dossier dans lequel vous avez décompressé le code et sélectionnez le fichier 2024 KitBot, puis cliquez sur OK.

3.4 Passage au contrôle via CAN

Si vous avez câblé vos contrôleurs de moteur SPARK MAX à l'aide du filage CAN, vous devrez effectuer d'autres modifications de configuration et de code avant de continuer. Si vous utilisez les PWM, passez à la section 3.5 pour déployer et tester le code.

3.4.1 Configuration des SPARK MAX

Avant d'utiliser les SPARK MAX avec contrôle CAN, chacun d'eux doit se voir attribuer un ID unique. Étant donné que vos SPARK commencent tous avec le même ID, vous pouvez débrancher le bus CAN de chaque dispositif pendant que vous mettez à jour et attribuez un ID.

1. Installer le client matériel REV Hardware Client (voa)
2. Avec le robot éteint, connectez un câble USB entre l'ordinateur et le port USB du SPARK MAX. Laissez le robot éteint garantit que seul le SPARK MAX est alimenté et évite de changer les ID sur des appareils non désirés.
3. Installer la mise à jour du firmware on the SPARK MAX (voa)
4. Sélectionnez le ID CAN et le type de moteur (CAN ID and Motor Type -voa). Sauvegardez les changements.
    a. Vous pouvez définir les ID comme vous le souhaitez (certaines équipes définissent ID CAN = le numéro de canal auquel l'appareil est connecté sur le PD), puis mettez à jour ces constantes.
    b. Remarque : Si vous souhaitez « Faire tourner le moteur » comme décrit sur cette page, assurez-vous que le robot est dans un état sûr pour le faire (roues ne touchant pas le sol ou la table).
5. Répétez l'opération pour les 6 appareils du robot.
6. Bien que cela ne soit pas nécessaire, si vous utilisez le PDH de REV, vous voudrez peut-être vérifier maintenant qu'il a également le dernier firmware. Ne changez pas l'ID du PDH hors de la valeur par défaut, chaque type d'appareil a un espace distinct d'identification et votre PDH n'entrera pas en conflit avec votre SPARK MAX même s'il est placé au même ID.

Maintenant que tous vos appareils sont configurés, vous pouvez effectuer une vérification préliminaire que votre bus CAN est câblé correctement à l'aide du client matériel REV. Lorsque vous êtes branché avec un câble USB sur n'importe quel appareil REV de votre bus CAN, mettez le robot sous tension et vous devriez voir tous les autres appareils répertoriés dans le volet gauche du client matériel REV, sous

l'en-tête CAN Bus. Si vous ne voyez pas tous les appareils, vous avez probablement un ou plusieurs problèmes avec le câblage de votre bus CAN :

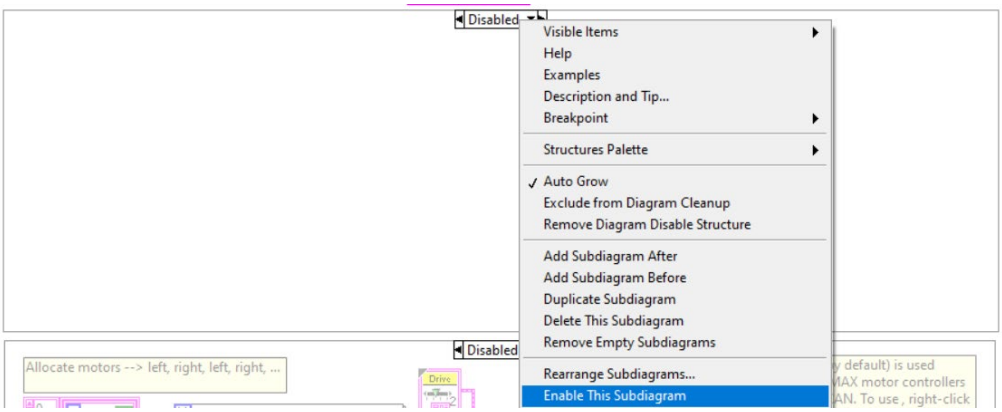
1. Vérifiez que votre bus CAN commence par le roboRIO et se termine par une résistance de 120 ohms, ou le terminateur intégré d'un concentrateur de distribution d'énergie ou d'un panneau de distribution électrique (avec la terminaison réglée sur ON à l'aide du cavalier ou du commutateur approprié.
2. Vérifiez que vos connexions de bus CAN correspondent toutes à jaune-jaune et vert-vert.
3. Vérifiez que toutes les connexions de fil CAN sont sécurisées les unes aux autres et que les connecteurs sont installés en toute sécurité dans chaque SPARK Max
4. Si vous rencontrez toujours des problèmes, déplacer la connexion USB vers différents appareils. Vérifier ce que chaque appareil peut « voir » dans le bus peut aider à localiser l'emplacement d'un problème.

3.4.2 Installation de REVLib

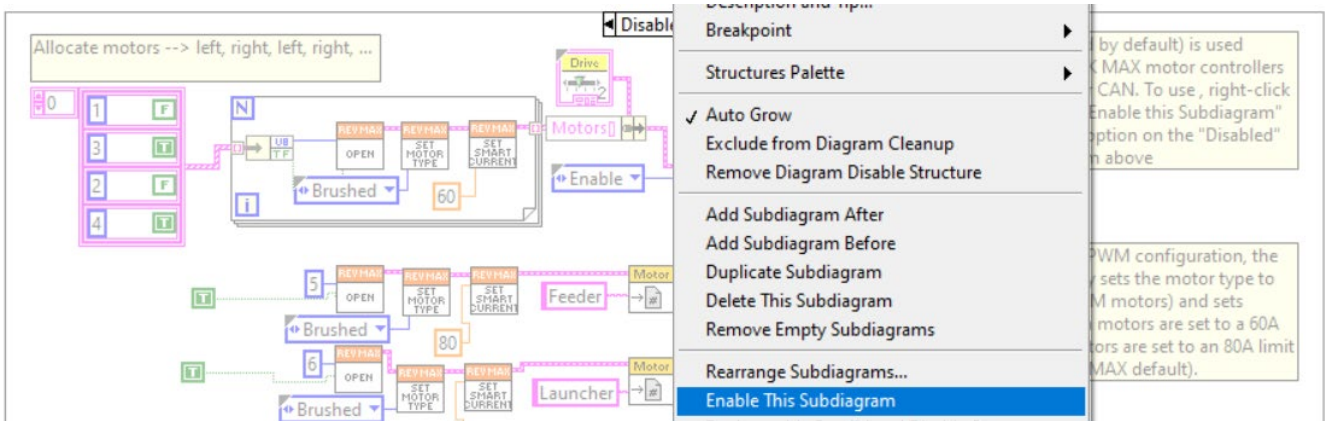
La bibliothèque logicielle du SPARK MAX en mode CAN est fournie par le fournisseur (REV Robotics). Vous devrez installer cette bibliothèque, sinon les VI CAN pour le SPARK MAX apparaîtront avec des icônes « ? ». Pour ce faire, suivez les instructions de la page de documentation REVLib (voa).

3.4.3 Mise à jour du code

Le code est principalement en place dans le projet pour passer du contrôle PWM au contrôle CAN, il vous suffira de faire un changement dans Begin.VI pour basculer.



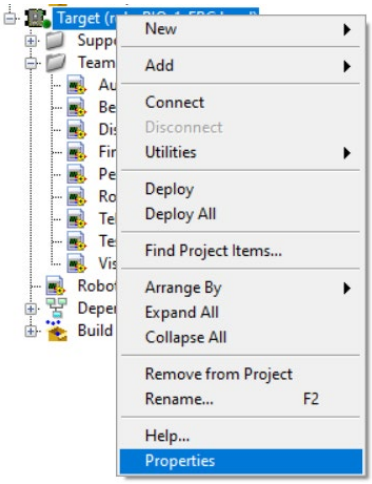
Le bloc supérieur de code pour le contrôle PWM est activé dans le projet par défaut. Cliquez sur la flèche à côté de « Enabled » pour passer au cas désactivé vide, puis cliquez avec le bouton droit de la souris et sélectionnez « Enable This Subdiagram». Cela désactivera le code PWM et activera le diagramme vide.



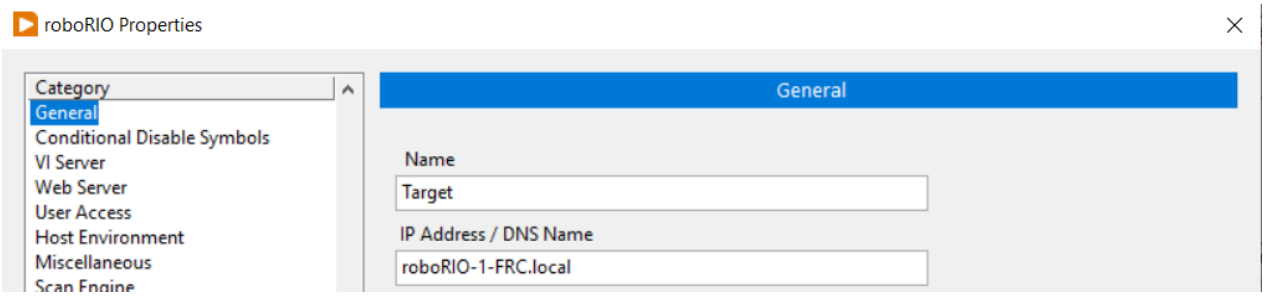
Ensuite, activez le diagramme CAN en cliquant avec le bouton droit de la souris et en sélectionnant « Enable This Subdiagram ».

3.5 Déploiement et test de l'exemple pour KitBot

Pour déployer l'exemple sur votre robot, vous devez définir le numéro d'équipe sur le projet.



Dans l'Explorateur de projets, faites un clic droit sur la ligne qui indique Target(...) et sélectionnez Properties.



Dans la case « IP Address / DNS Name », tapez **roborio-#####-FRC.local** où #### est votre numéro d'équipe (exemple pour l'équipe 1 illustré). Si vous rencontrez des difficultés à déployer, vous pouvez

également essayer de définir ceci sur **172.22.11.2** pour la connexion USB ou **10.TE. AM.2** pour la connexion Ethernet ou sans fil à un roboRIO connecté à une radio programmée où TE.AM est votre numéro d'équipe à 4 chiffres (par exemple 10.0.1.2 pour l'équipe 1, 10.3.12.2 pour l'équipe 312 et 10.75.43.2 pour l'équipe 7543).

Vous êtes maintenant prêt à exécuter ou à déployer l'exemple KitBot (rappelez-vous que seul Set as Startup persistera après un redémarrage du roboRIO !) tout comme vous avez déployé le projet de test à l'étape 4 du guide Zéro à Robot.

Avertissement: Assurez-vous d'avoir de l'espace dans toutes les directions lorsque vous faites fonctionner un robot. Même avec un code connu, le robot peut se déplacer à une vitesse ou dans des directions inattendues. Soyez prêt à désactiver le robot (Enter) ou à faire un arrêt d'urgence E-stop (barre d'espace) si nécessaire. Le code pour KitBot 2024 contient une routine autonome très simple qui déplacera le robot vers l'arrière à une vitesse de 1/2 pendant 1 seconde lorsque le robot est activé en mode autonome.

3.6 Configuration des manettes de jeu

Le code est configuré pour utiliser la classe de manette Xbox. Les manettes de jeu Logitech F310 fournies dans le kit de pièces apparaîtront comme des manettes Xbox dans le logiciel WPILib si elles sont configurées dans le mode correct. Pour configurer les contrôleurs, vérifiez que le commutateur à l'arrière du contrôleur est défini sur le paramètre « X ». Ensuite, lorsque vous utilisez le contrôleur, assurez-vous que le voyant à côté du bouton Mode est éteint, s'il est allumé, appuyez sur le bouton Mode pour le basculer. Lorsque le bouton Mode est allumé, le contrôleur permute la fonction du bâton analogique gauche et du D-pad.

3.7 Que fait le code?

Le code fourni implémente les commandes de robot suivantes dans le mode téléopéré :

- La manette de contrôle est une manette Xbox dans l'emplacement 0 de DriverStation
    - Contrôle la propulsion du robot à l'aide de Split-stick Arcade Drive
        - L'axe Y (vertical) du manche gauche contrôle le mouvement avant-arrière de la plateforme pilotable
        - L'axe X (horizontal) du manche droit contrôle la rotation de la plateforme pilotable
- La manette d'opérateur est une manette Xbox dans l'emplacement 1 de la station de pilotage
    - Gâchette avant gauche - Active les deux roues du lanceur de pièce de jeu vers l'intérieur à des vitesses différentes pendant que le bouton est maintenu. Cela permet au robot d'admettre une pièce de jeu
    - Bouton A – Exécute une courte séquence pour lancer une pièce de jeu pendant que le bouton est maintenu
        - Démarre la roue avant pour atteindre sa pleine vitesse
        - Attend 1 seconde

- Active la roue arrière pour transmettre une pièce de jeu vers la roue avant qui tourne

4 Structure globale du code

Le code fourni utilise la structure de code LabVIEW par défaut. Il s'agit des VI suivants. Ces VI se trouvent dans le dossier Team Code de l'Explorateur de projets, répertorié par ordre alphabétique. Les explications ici sont arrangées dans un sens plus chronologique.

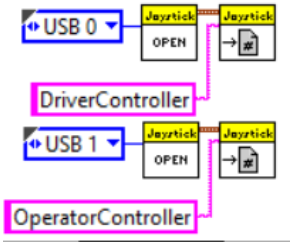
- **Begin.VI** –Utilisé pour initialiser des objets matériels et des données. Le code KitBot utilise ce VI pour la configuration matérielle.
- **Periodic Tasks.VI** – Ce VI est appelé une fois, mais contient des boucles qui s'exécutent périodiquement indépendamment du mode du robot. Pour le KitBot, le contrôle du lanceur est géré ici. Les équipes choisissent souvent de mettre un contrôle de mécanisme avancé ici afin qu'il puisse être exploité dans les modes autonome et Teleop.
- **Disabled.VI** – Ce code est appelé périodiquement (~ 50hz) pendant que le robot est désactivé. Vous pouvez l'utiliser pour réinitialiser l'état des choses, contrôler les lumières du robot, réagir à la sélection du mode autonome ou à toute autre action que vous souhaitez que le code fasse pendant que le robot est désactivé (rappelez-vous, vous ne pouvez pas contrôler les actionneurs en mode Désactivé !). Le code KitBot ne modifie pas le contenu par défaut de ce VI.
- **Autonomous Independent.VI** – Ce VI est appelé une fois lorsque le mode autonome commence et est automatiquement annulé lorsque le mode auto se termine. Vous pouvez soit structurer votre mode auto comme une séquence, ou ajouter une boucle à l'intérieur si vous souhaitez configurer une machine d'état. Le code KitBot contient un code auto de base pour reculer à 50% de vitesse pendant 1 seconde en plus du code par défaut dans ce VI.
- **Robot Global Data.vi** – Ce VI contient des données auxquelles vous souhaitez accéder à partir de différentes parties du programme du robot. Le code KitBot ajoute une énumération pour transmettre l'état du lanceur souhaité aux valeurs par défaut.
- **Test.VI** – Ce VI est appelé une fois lorsque le robot est activé en mode Test et est automatiquement annulé lorsque le robot est désactivé ou que le mode change. Vous pouvez écrire du code dans ce VI pour tester la fonctionnalité du robot, soit automatiquement, soit en utilisant les entrées du pilote. Le code KitBot ne modifie pas le contenu par défaut de ce VI.
- **Finish.VI** – Ce VI est appelé lorsque le bouton Terminer est enfoncé lors de l'exécution du code à partir du PC. Ce VI n'est jamais appelé lorsque le code est déployé sur le robot. La plupart des équipes n'utilisent pas ce VI. Le code KitBot ne modifie pas le contenu par défaut de ce VI.
- **Vision Processing.VI** – Ce VI peut être utilisé pour ajouter du code de traitement de la vision. Ce VI n'est pas exécuté par défaut, si vous choisissez de l'utiliser, vous devrez l'ajouter à Robot Main.VI vous-même, en utilisant les 4 autres VI dans la section inférieure (comme l'icône Timed Tasks) comme exemples. Le code KitBot ne modifie pas le contenu par défaut de ce VI.

5 Présentation du code

5.1 Begin.VI

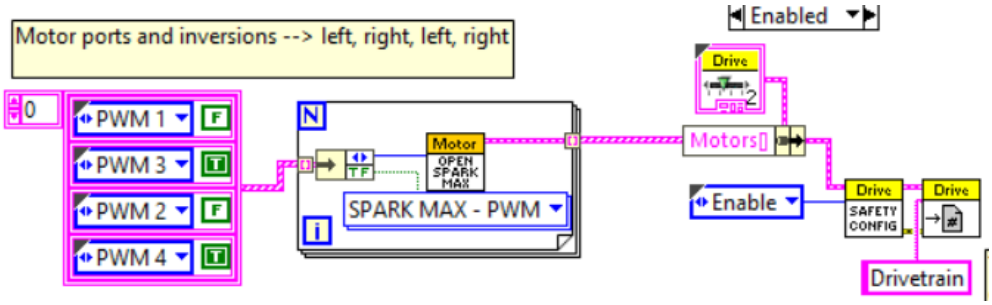
Comme décrit dans la section 4, Begin.VI est généralement utilisé pour ouvrir des références pour tout le matériel du robot et pour définir des configurations. Le code KitBot contient quelques éléments du modèle de projet LabVIEW par défaut, omis ici, et un code spécifique à KitBot décrit ci-dessous.

5.1.1 Références aux contrôleurs



Ce code du haut au centre du diagramme ouvre une référence à un contrôleur connecté aux ports 0 et 1 sur la station de pilotage et enregistre ces références à RefNums appelés DriveController et OperatorController respectivement.

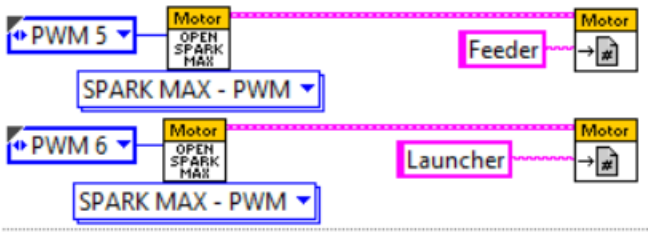
5.1.2 Base pilotable PWM



Cette section de code en haut du diagramme d'activation/désactivation supérieur est destinée à configurer les contrôleurs de moteur pour la base pilotable avec contrôle PWM. Comme indiqué dans le commentaire, l'ordre des constantes dans le tableau est gauche, droite, gauche, droite. Ce qui signifie que ce code met en place les contrôleurs de gauche sur les ports 1, et 2 sans inversion et les contrôleurs de droite sur les ports 3 et 4 et les inverse. Cette inversion permet à la base pilotable de fonctionner correctement avec les valeurs de l'axe du joystick transmises directement. Si, au lieu de cela, vous voulez que les deux côtés de la transmission avancent lorsque les valeurs positives sont passées, inversez plutôt le côté droit (et annulez les valeurs de l'axe du joystick dans Teleop.VI).

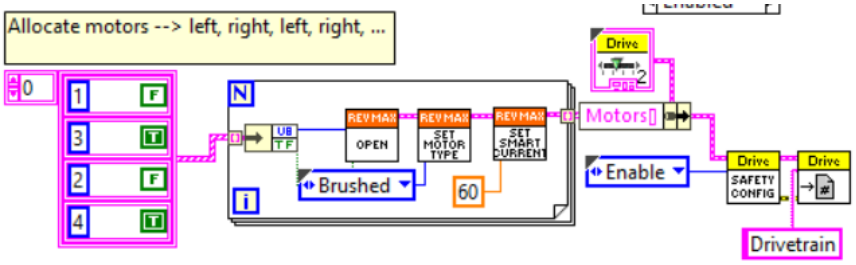
La section centrale de ce code spécifie que les contrôleurs de moteur sont des contrôleurs SPARK MAX connectés par PWM. Ensuite, le code active la fonction de sécurité du moteur sur la base pilotable, ce qui désactivera les moteurs si vous ne leur fournissez pas une commande mise à jour toutes les 100 ms. Enfin, nous stockons la référence dans un RefNum appelé Drivetrain.

5.1.3 Lanceur PWM



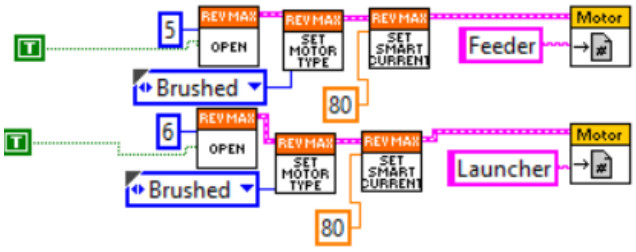
Sous le code de la base pilotable PWM se trouve ce code pour le lanceur PWM. Ce code est assez simple, il ouvre une référence aux deux SPARK MAX, sur les ports 5 et 6, et stocke la référence dans des RefNums.

5.1.4 Base pilotable CAN



Dans la prochaine structure Activée/Désactivée (désactivée par défaut) se trouve le code de configuration de la base pilotable si vous avez connecté vos contrôleurs de moteur avec CAN. Si vous n'avez pas installé REVLib correctement ou si vous n'avez pas redémarré LabVIEW depuis l'installation, ces icônes s'afficheront sous forme de symboles « ? ». Ce code fonctionne de façon très similaire au code de base pilotable PWM décrit ci-dessus, mais il définit explicitement le type de moteur sur « Brushed » pour correspondre aux moteurs CIM utilisés sur le KitBot et définit une limite de courant de 60 Ampères sur les moteurs.

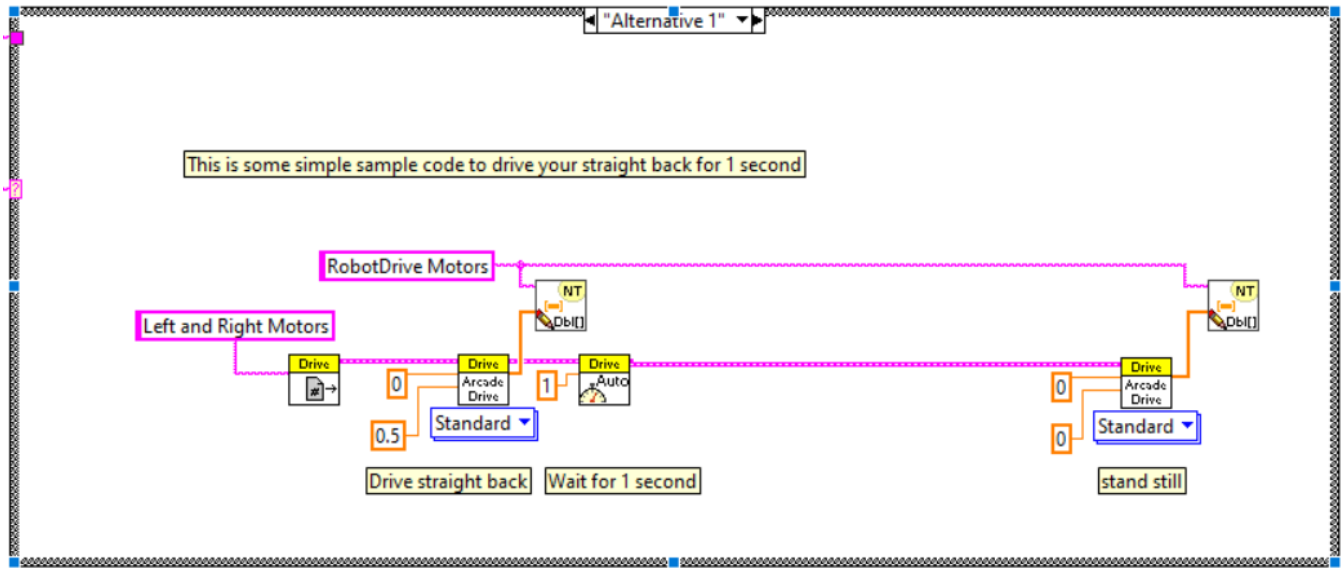
5.1.5 Lanceur CAN



Le code CAN du lanceur ouvre une référence aux SPARK MAX et les définit pour qu'ils soient inversés. Cela était nécessaire sur notre KitBot pour faire tourner les roues vers l'extérieur (lancement) avec une commande positive et vers l'intérieur (admission) avec une commande négative. Si votre robot se comporte différemment, vous pouvez soit modifier ces constantes d'inversion, soit inversez le signe des valeurs utilisées dans le contrôle du lanceur. Ce code définit également la limite de courant sur les moteurs à la valeur par défaut de 80 Ampères.

5.2 Autonomous Independent.VI

La majorité du code dans l'Autonomous Independent VI est le code par défaut fourni par le modèle de projet LabVIEW. Consultez les commentaires et certaines des ressources sur la page LabVIEW FRC dans la section 2 pour plus d'informations.



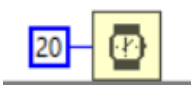
Le code dans le cas « Alternative 1 » fait reculer le robot à mi-vitesse pendant 1 seconde. Utilisez le tableau de bord LabVIEW afin de sélectionner ce mode autonome à exécuter. La vitesse fournie est positive afin de faire reculer le robot en raison des inversions utilisées dans Begin.VI. Parce que l'axe Y d'un joystick est négatif lorsque vous le poussez (vers l'avant), une valeur positive ici est vers l'arrière. Si vous préférez que les valeurs positives fassent avancer le robot, changez les inversions dans Begin et inversez le signe des axes joystick dans Teleop.

Le VI après l'ajustement de la vitesse est un VI spécial utilisé dans Autonomous qui alimente la sécurité du moteur de la base pilotable tout attendant pendant un temps spécifié.

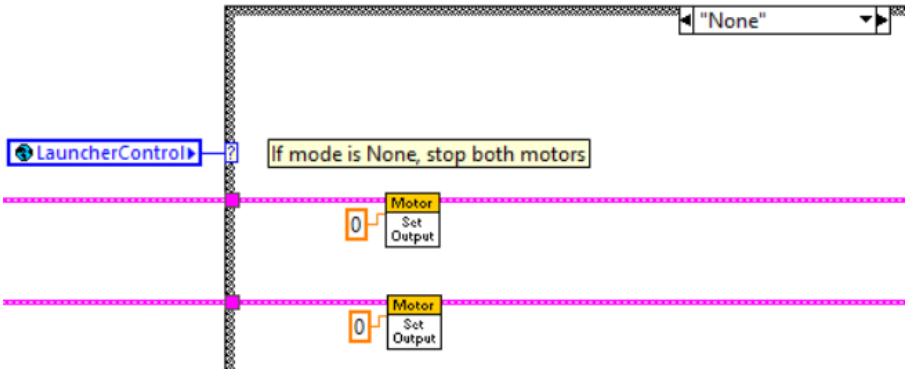
5.3 Periodic Tasks

Le Periodic Tasks VI peut être utilisé pour exécuter des actions de robot à un rythme fixe quel que soit le mode du robot. Le KitBot place le contrôle du lanceur ici afin que vous puissiez facilement étendre le code pour utiliser les contrôles du lanceur en mode autonome. Le code fourni définit uniquement la variable globale utilisée pour contrôler le lanceur à partir du code Teleop, mais vous pouvez l'utiliser comme exemple pour ajouter un contrôle du lanceur dans Autonomous si vous le souhaitez.

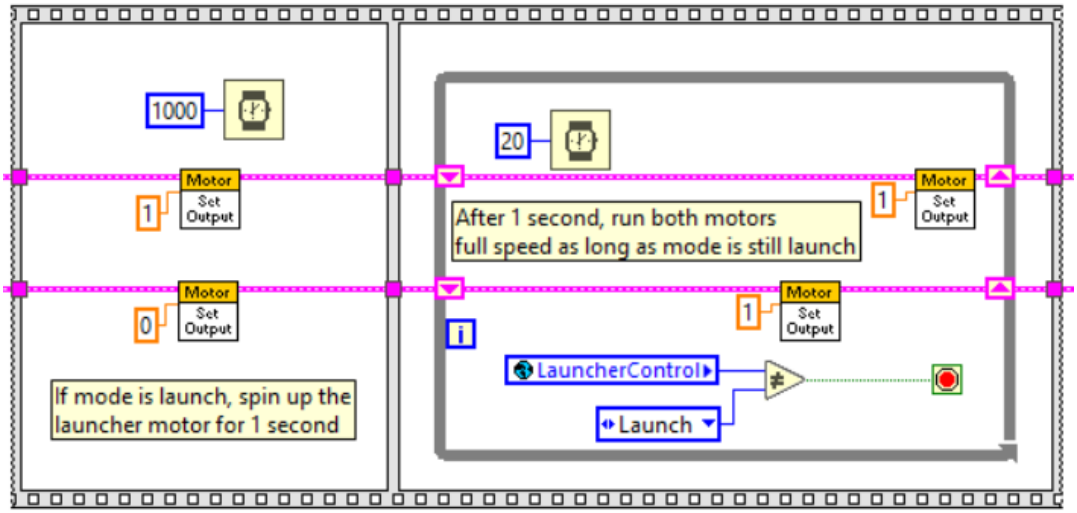
Les commandes du lanceur sont placées à l'intérieur d'une structure de boucle. Le Periodic Tasks VI n'est appelé qu'une seule fois, après la fin de Begin.VI, donc si nous voulons que notre code continue à fonctionner, nous devons fournir nos propres boucles.



Au bas de la boucle se trouve ce retard de 20 ms. Le code LabVIEW qui n'est pas connecté dans des connecteurs indiquant une dépendance de données s'exécute en parallèle, donc ce retard signifie que notre boucle prendra au moins 20 ms pour fonctionner, mais peut prendre plus de temps si d'autres éléments de boucle s'exécutent sur plus longtemps que cela. Pour le code KitBot, la boucle fonctionnera au taux de 20 ms si le LauncherControl est None ou Intake, mais la boucle extérieure ne sera pas en boucle du tout si le mode est Launch.

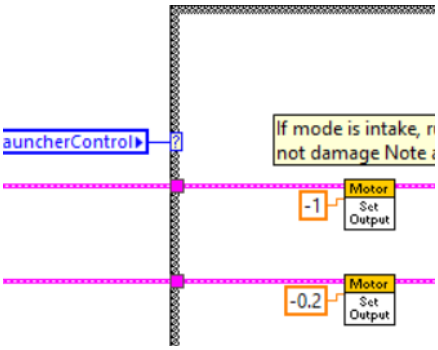


Le code Launcher utilise une structure de cas, alimentée par la variable globale LauncherControl, afin d'exécuter différents comportements en fonction du mode de lancement souhaité. Si le mode souhaité est défini sur None, le code définit les deux roues à 0 pour les arrêter.



Si le mode est défini sur Launch (cliquez sur les flèches en haut de la structure de casse pour parcourir les diagrammes), le code exécute cette structure séquentielle. Le premier cadre de cette structure définit la roue du lanceur à la vitesse maximale et maintient la roue d'admission réglée à 0 tout en retardant pendant 1 seconde. Cela donne à la roue du lanceur le temps d'atteindre sa pleine vitesse avant que le chargeur ne pousse l'anneau vers l'intérieur. Parce que ce cadre utilise un délai, la roue fonctionnera pendant au moins 1 seconde même si vous appuyez sur le bouton. En général, ce n'est pas un gros problème, donc nous n'avons pas ajouté de complexité ici pour empêcher cela.

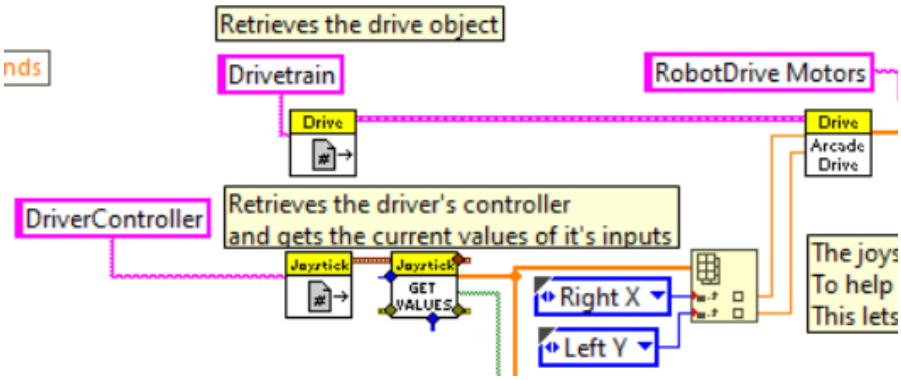
Le cadre suivant de la séquence contient une boucle qui s'exécute toutes les 20 ms. Cette boucle met les deux moteurs à pleine vitesse pour lancer l'anneau. La section inférieure de la boucle vérifie pour s'assurer que le mode est toujours réglé sur Launch. Dès que le mode n'est plus Launch, la valeur passée dans l'indicateur d'arrêt sera True et la boucle prendra fin.



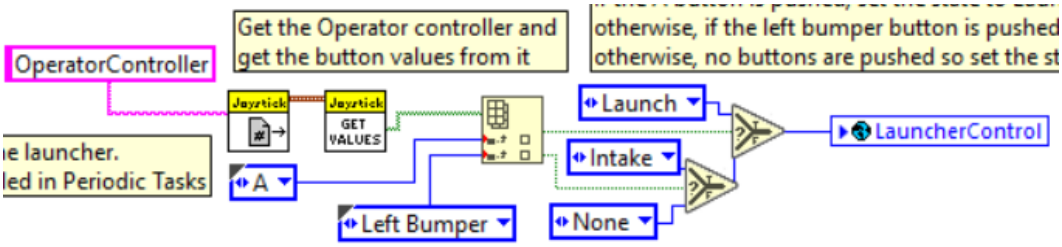
Dans la partie Intake de la structure de cas, la roue du lanceur est réglée pour rouler vers l'intérieur à la vitesse maximale et la roue d'admission est réglée pour rouler vers l'intérieur à basse vitesse. Cela tire complètement la pièce de jeu, mais évite de l'endommager par l'action de la roue une fois qu'elle est complètement à l'intérieur du robot.

5.4 Teleop.VI

Certains aspects du code Teleop sont des parties par défaut du code du modèle LabVIEW. Ces éléments ne sont pas décrits ici.



Cette section de code accède aux valeurs du joystick à partir du DriverController, sélectionne la valeur d'axes spécifiques dans le tableau d'axes et utilise ces valeurs pour contrôler la base pilotable en propulsion Arcade Drive. Arcade Drive contrôle la rotation du robot avec une valeur (ici la valeur de l'axe X droit) et la vitesse du robot avec une valeur distincte (ici la valeur de l'axe Y gauche).



Cette section de code accède aux valeurs de bouton à partir de OperatorController, sélectionne des valeurs de bouton spécifiques dans le tableau et utilise une paire de Select VI pour mapper ces états de bouton aux états du lanceur souhaités. Le Select VI le plus à droite a priorité. Étant donné qu'il s'agit du dernier Select exécutée, si on appui sur le bouton A, le mode sera défini sur Launch, quel que soit l'état de la gâchette avant gauche. L'autre Select VI est utilisé pour définir le mode sur Intake si la gâchette avant gauche est enfoncée et le bouton A ne l'est pas. Ce mode souhaité est stocké dans la variable globale et accessible à partir du Periodic Tasks VI, où le contrôle du lanceur se produit réellement.

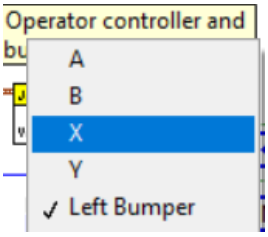
6 Apporter des modifications

Cette section détaille quelques modifications possibles courantes que vous voudrez peut-être apporter au code KitBot et fournit quelques références sur la façon d'aborder ces modifications.

6.1 Modification des boutons d'action

L'un des changements les plus faciles à apporter est de changer quels boutons contrôlent le lanceur. Pour modifier cela, sélectionnez simplement une autre option de bouton sur l'une des énumérations en cliquant sur la valeur et en sélectionnant le nouveau choix dans le menu déroulant.

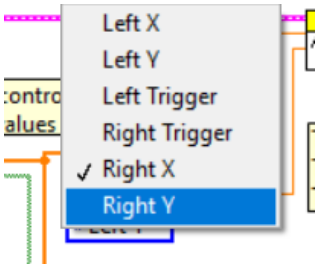
Par exemple, pour changer la commande d'admission de la gâchette avant gauche au bouton **X** :



6.2 Modification du comportement des sticks de pilotage

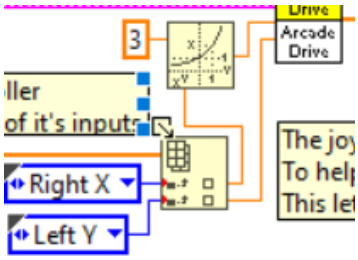
Un autre changement facile à faire est de modifier quels axes du contrôleur sont utilisés au pilotage du robot et comment. Pour modifier l'axe, cliquez simplement sur l'énumération et sélectionnez un nouvel axe dans la liste déroulante.

Exemple : changer le mouvement avant-arrière sur l'axe Y du stick droit et laisser la rotation sur l'axe X droit



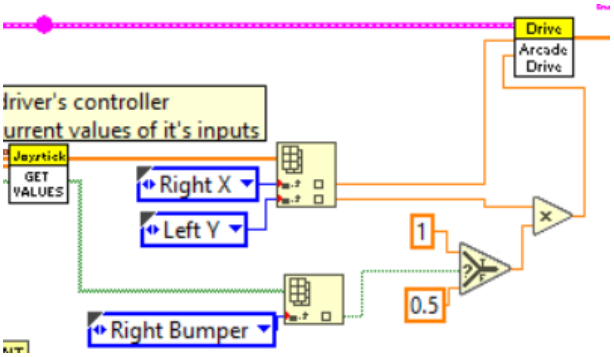
Vous pouvez également modifier les valeurs d'axe. Une modification courante consiste à porter au cube les valeurs. Cela préserve le signe de la valeur (positif reste postif, négatif reste négatif) et de la valeur maximale (ne réduit pas la vitesse maximale du robot) tout en fournissant moins de sensibilité à de faibles entrées, permettant potentiellement un contrôle plus précis à basse vitesse. Pour effectuer ce type de modification, vous pouvez appliquer la modification à l'axe entre l'endroit où il est dévelopé du tableau et l'endroit où il entre dans Arcade Drive VI.

Exemple : changer uniquement l'axe de rotation à mettre au cube (accompli ici en utilisant le 'x^y' VI de la palette Mathematics-Exponential) :



Une autre modification courante consiste à réduire les valeurs par défaut, mais à autoriser la valeur maximale si vous appuyez sur un bouton (mode turbo).

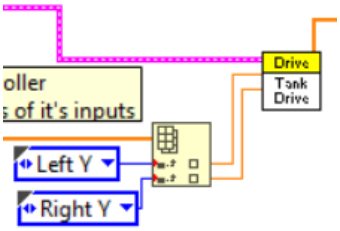
Exemple : Réduire la conduite avant-arrière de 50 % à moins que la gâchette avant droite ne soit enfoncée.



6.3 Modification du type de propulsion

Le dernier changement probable que nous couvrirons est le passage d'Arcade Drive à Tank Drive. Contrairement à la propulsion Arcade qui mappe un axe à la rotation et un au mouvement avant / arrière du robot, la propulsion de type Tank mappe un axe (généralement l'axe Y) de chaque côté d'une base pilotable différentielle. Pour effectuer cette modification, faites un clic droit sur ArcadeDrive VI, sélectionnez Replace et accédez à la palette WPI Robotics Library -> Robot Drive pour sélectionner le Tank Drive VI. Modifiez ensuite les axes du joystick comme désiré (généralement l'axe Y de chaque stick si vous utilisez un seul contrôleur et non une paire de joystick).

Exemple :



6.4 Développer des routines autonomes

Le code fourni contient un mode autonome très basique qui roule vers l'arrière à 1/2 de puissance pendant 1 seconde. Vous pouvez modifier ce cas dans la structure pour changer ou améliorer la routine et des modes autonomes supplémentaires peuvent être développés en ajoutant des cas supplémentaires à la structure.