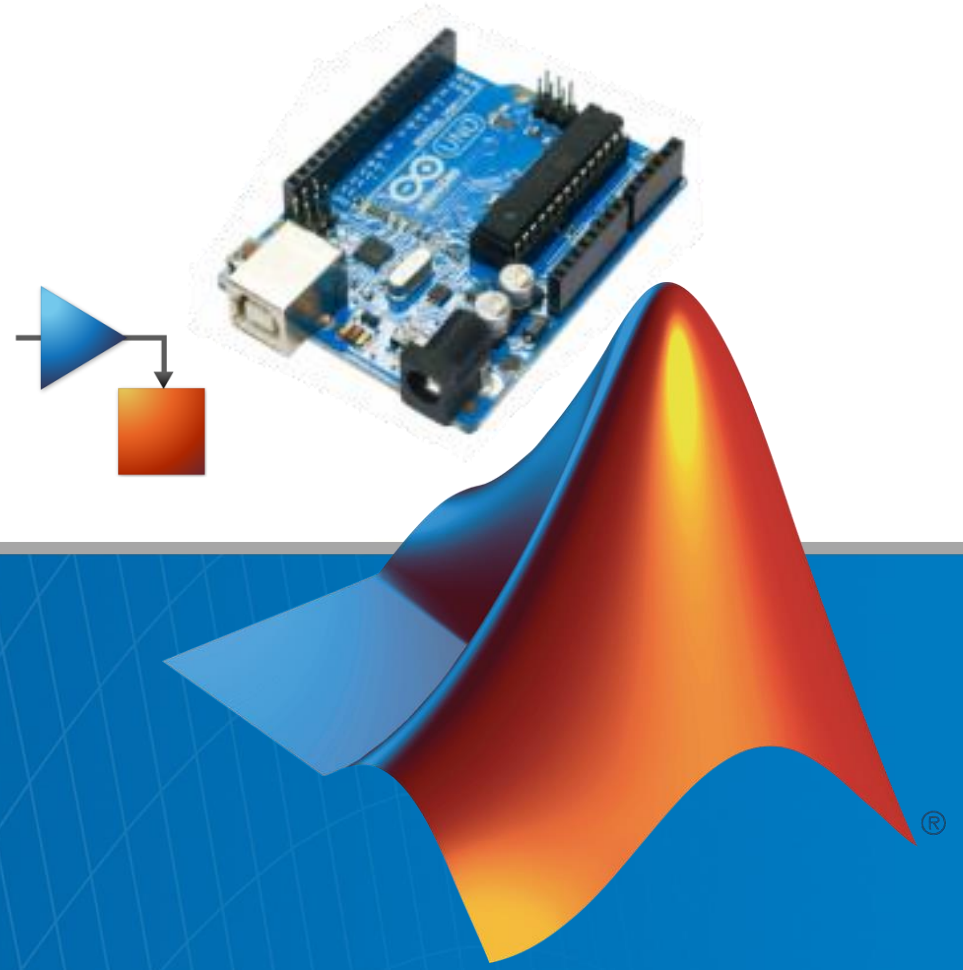
Workshop Arduino
PMClab



Démarche

- **Modéliser**
- Simuler
- Implémenter

Modéliser

1. Ouvrir la Simulink Library

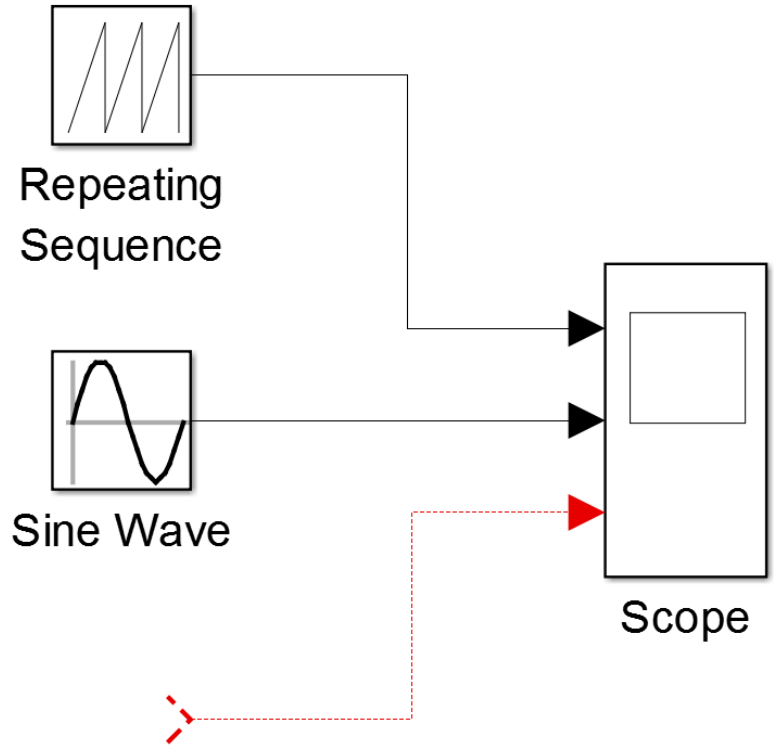
Elle contient des blocs permettant de réaliser des opérations mathématiques, logiques plus ou moins complexes.

Par exemple:

- Simulink>Sources permet de créer des signaux (carré, sinusoïde, échelon…)
- Simulink>Sinks permet d'afficher des signaux (scope, display)
- Simulink>Math Operations pour les opérateurs mathématiques

2. Ouvrir le modèle Signaux.slx et ajouter un bloc pour visualiser un signal carré sur le 3ème port du scope

Générer et visualiser des signaux



Démarche de l’ingénieur

- Modéliser
- **Simuler**
- Implémenter

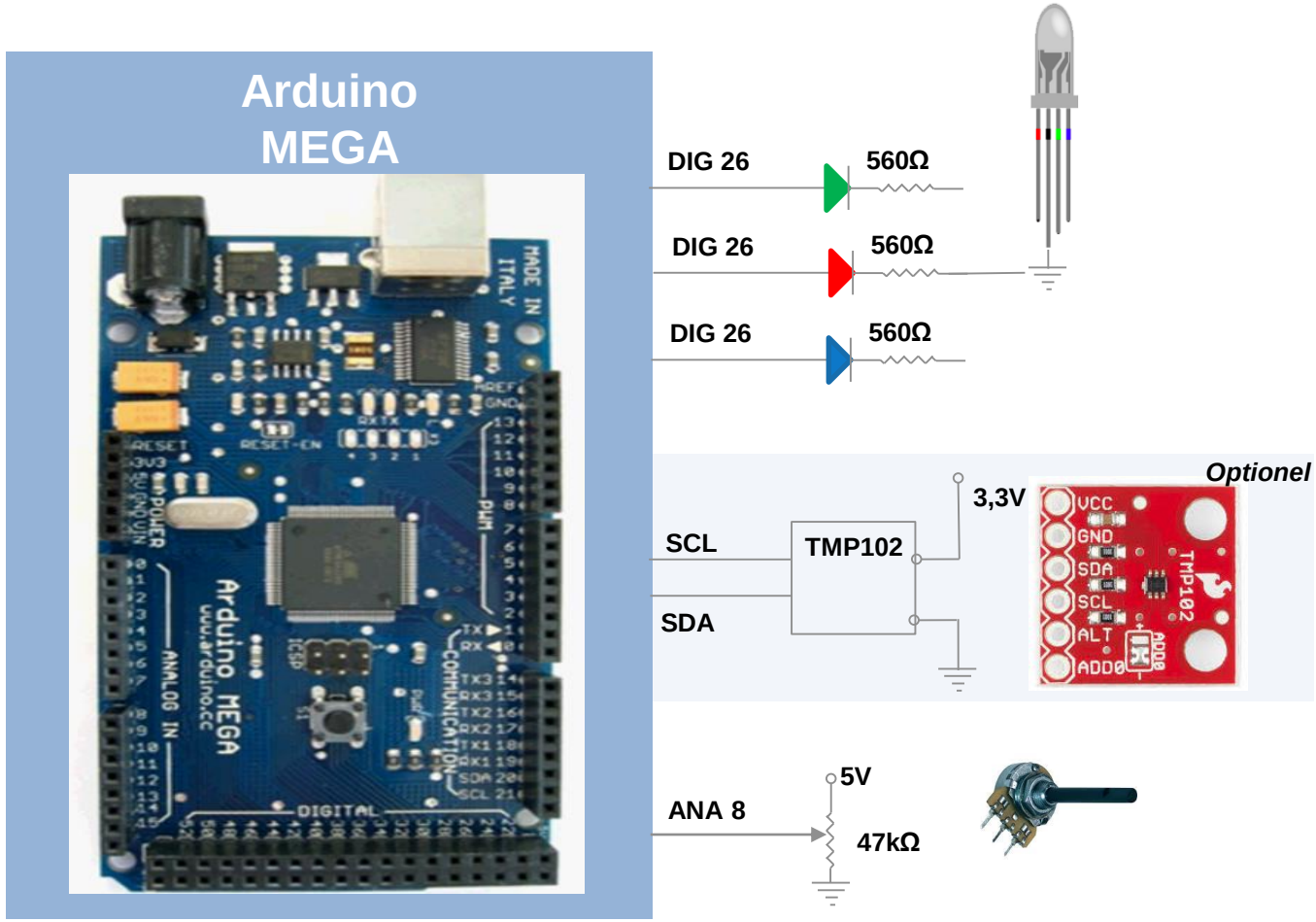
Simuler le Modèle

- Simuler le modèle **Signaux.slx** en cliquant sur
- Double-cliquer sur le bloc **Scope** pour visualiser les signaux.
- Modifier l'amplitude de la sinusoïde en changeant les paramètres du bloc puis relancer la simulation
- Cliquer sur pour arrêter une simulation

Démarche de l’ingénieur

- Modéliser
- Simuler
- **Implémenter**

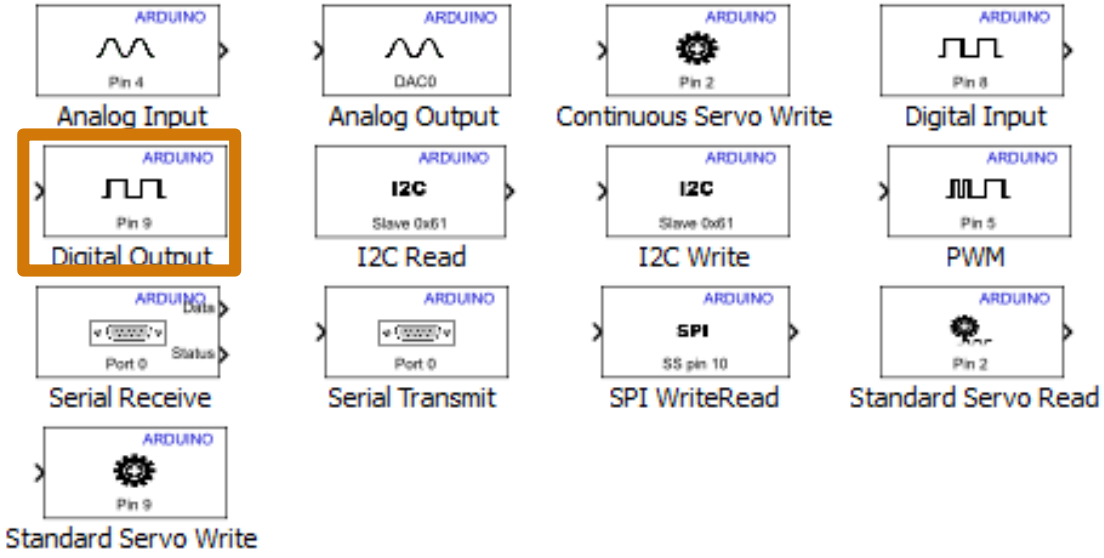
Schéma électronique



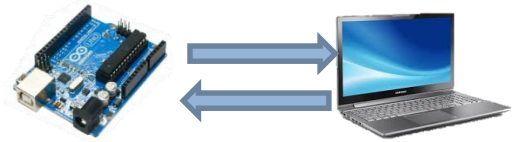
Ex1 : Faire clignoter la LED

Nous allons maintenant passer de la simulation à l'implémentation du modèle sur la carte Arduino. Il existe une librairie de blocs dédiés à la carte Arduino: ***Simulink Support Package for Arduino Hardware***

- Sauvegarder le modèle **Signaux.slx** sous un nouveau nom (ex: LED.slx)
- Modifier le modèle pour faire clignoter la LED:
  - Supprimer les signaux inutiles
  - Utiliser un bloc **Digital Output** pour envoyer le signal vers la LED et double-cliquer sur le bloc pour changer le numéro de port en fonction de la connexion de la maquette
  - Conserver le scope

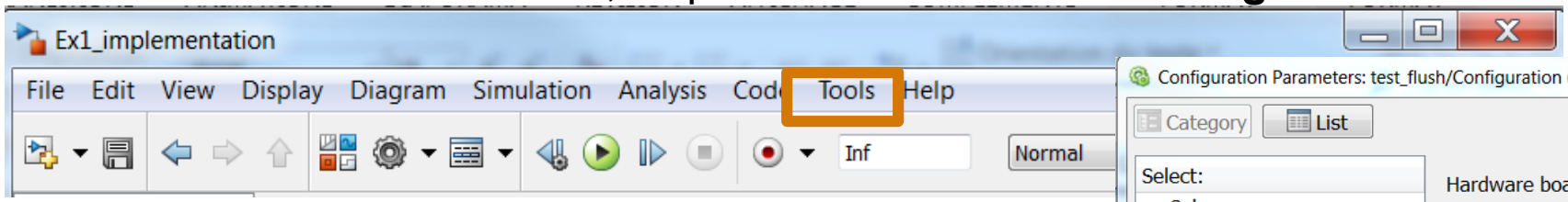


Préparer le modèle pour l'implémenter sur la carte Arduino (1/2)

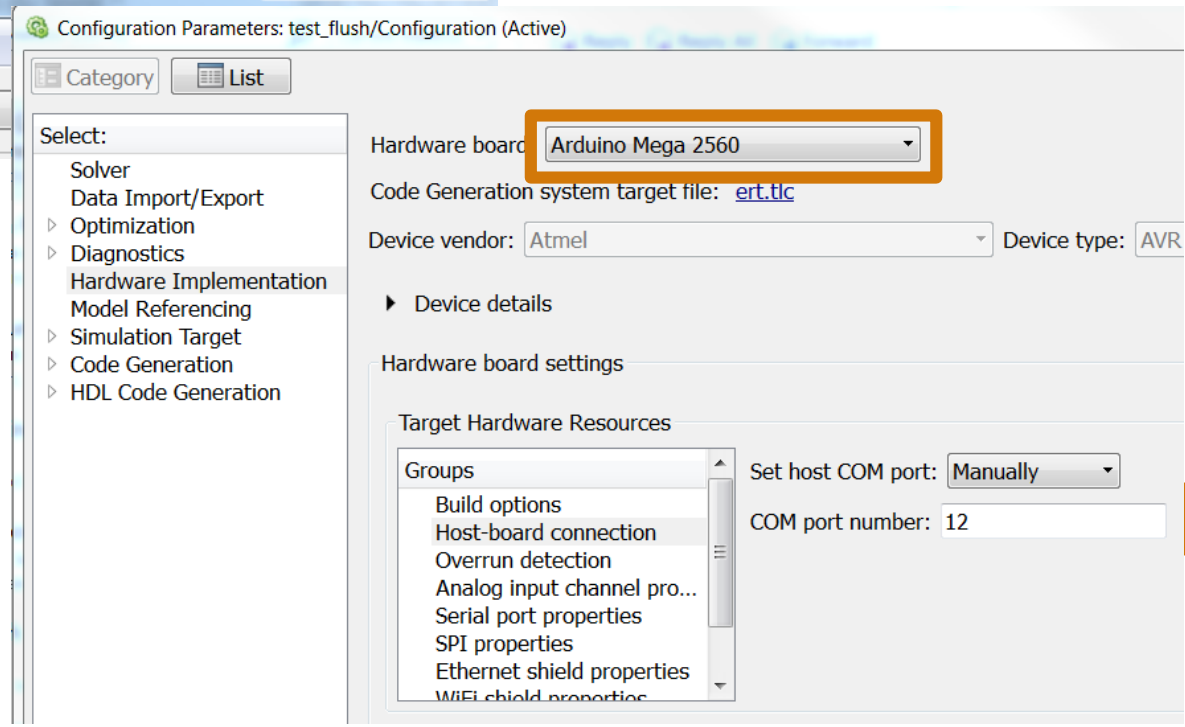


Préparer le modèle pour être embarqué sur la cible Arduino :

- Sur la barre d'outil du modèle, cliquer sur **Tools > Run on Target Hardware > Prepare to run…**

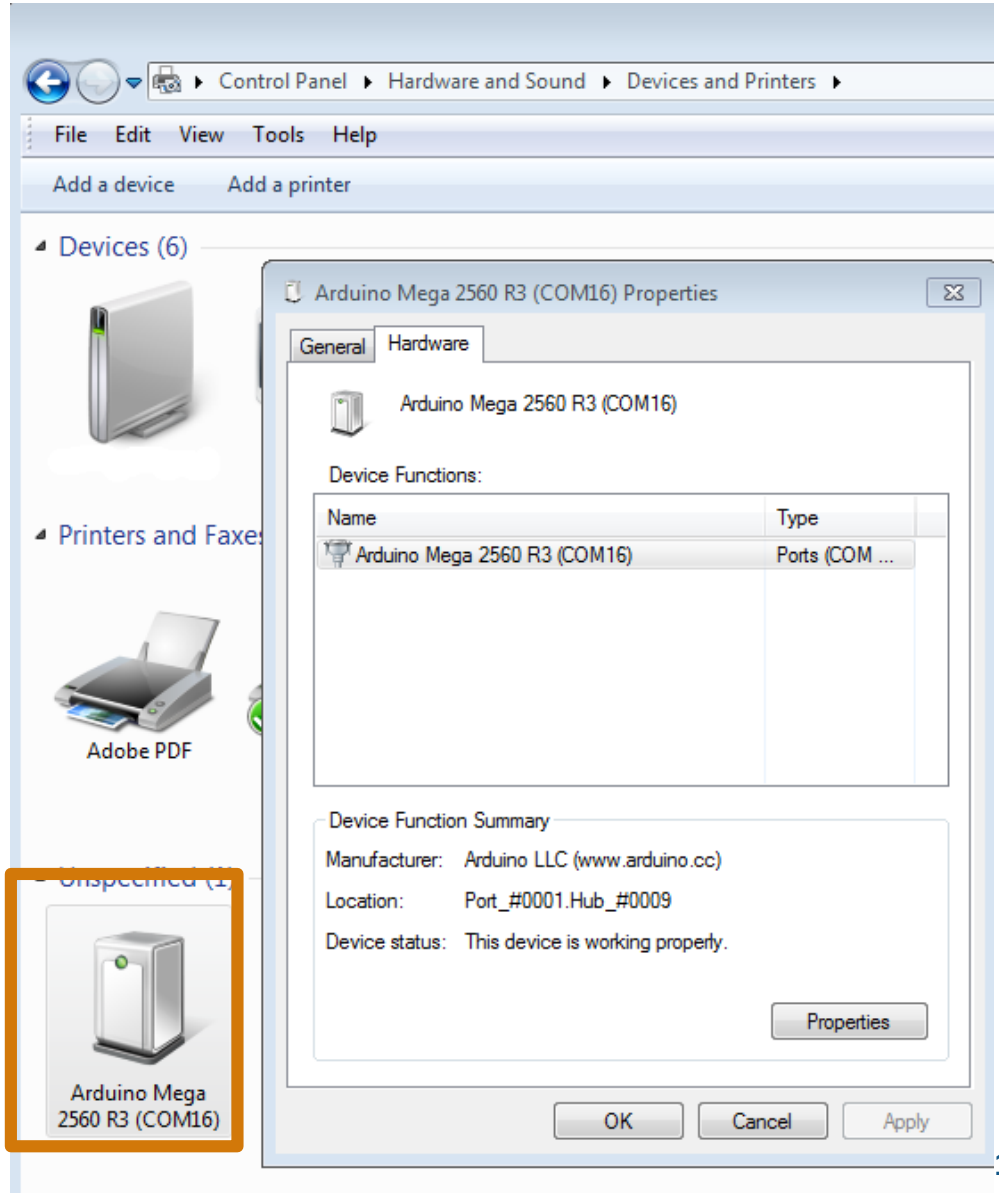


- Attendre que la fenètre suivante s'ouvre
- Choisir la cible (**Hardware Board)** :
  - Arduino Uno ou Leonardo
- Changer Set host COM port à **Manually**
- Renseigner le numéro du port COM (voir Annexe page suivante)



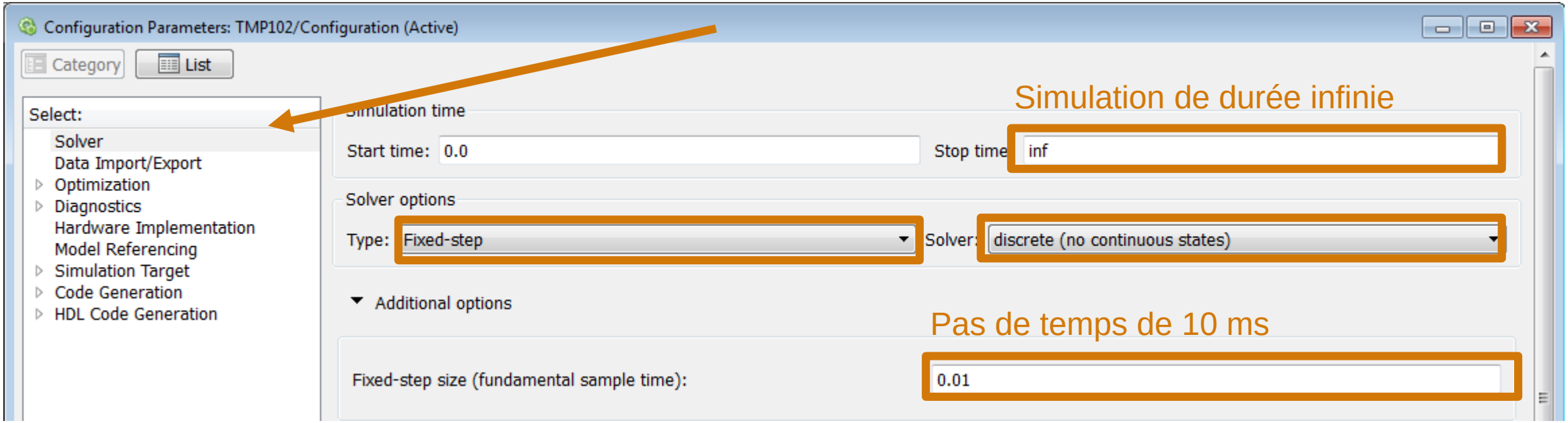
Annexe: Comment obtenir le numéro du port com ?

Aller dans
Control Panel>Printers and devices.
Relever le numéro du port Com affiché sur le périphérique **ArduinoMega2560**



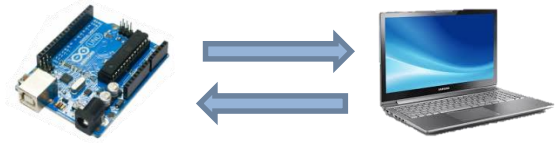
Préparer le modèle pour l'implémenter sur la carte Arduino (2/2)

- Avant de cliquer sur OK, aller sur **Solver** et le paramétrer comme suit:

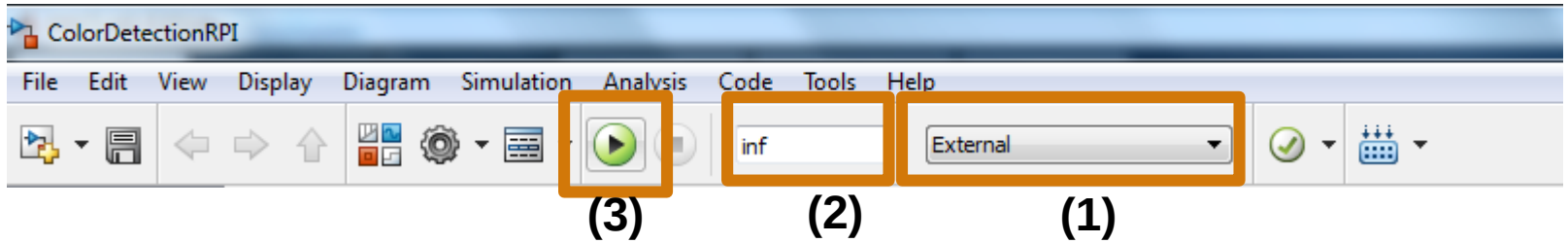


- Appuyer sur OK.

Implémenter le modèle sur la carte Arduino

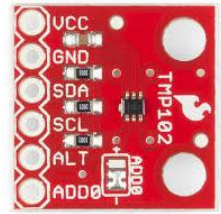


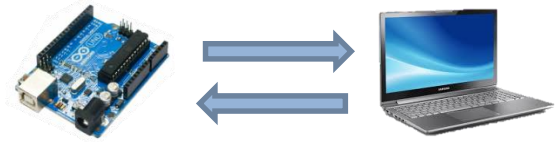
1. Vérifier que la Arduino est allumée et connectée au PC par le cable USB
2. Dans le modèle, sélectionner le mode “**External**” (1), verifier que le **temps d’exécution** est à Inf (2) et lancer **l’exécution** du modèle (3)



3. Le code C est généré, compilé, téléchargé et executé sur la carte. La communication entre la carte et le PC permet de modifier les paramètres du modèle et de visualizer les signaux en temps réel pendant l’exécution.
4. Vous pouvez modifier le temps d’allumage de la LED en modifiant les paramètres du bloc **Pulse Generator**
5. Appuyer sur pour arrêter le programme
6. Modifier le modèle pour créer le motif suivant: led rouge 0.5s, led verte 0.1s, led bleue 0.1s, led rouge 0.5s…

Ex2 : Lire les données d'un capteur

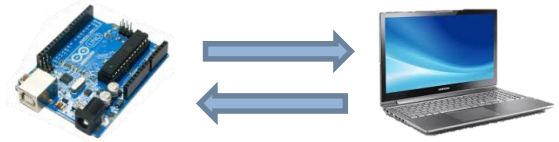




Le capteur TMP102 est un capteur de température. Il communique avec la carte Arduino par le bus de communication I²C.

- Ouvrir le modèle **TMP102.slx**
- Double-cliquer sur le bloc **I2CRead** qui permet de lire depuis un périphérique I²C connecté à la carte Arduino. Et configurer le bloc sachant que:
  - L'adresse du périphérique est 0x48 en hexadecimal ou 72 en decimal
  - La température est codée sur 12 bits et Big Endian (bits de poids fort en premier)
  - On souhaite lire une nouvelle valeur tous les 100 ms.
  - La résolution du capteur est de 0,0625.
- A quoi sert le bloc Gain=0,0625/16?
- Ajouter un bloc pour visualiser la température
- Suivre la même procédure (pages 9-11) pour exécuter le modèle sur la carte Arduino
- Mesurer la température du bout de votre doigt. De celui de votre voisin

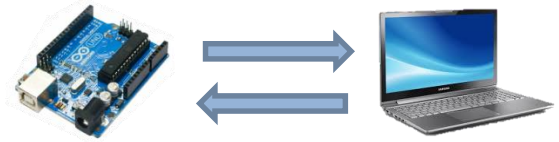
Ex2 : Lire les données du potentiomètre

Le potentiomètre est connecté à une entrée analogique de la carte Arduino

- Ouvrir le modèle **Potentiometre.slx**
- Double-cliquer sur le bloc **Analog Input** qui permet de lire la valeur du potentiomètre. Et configurer le bloc :
  - Modifier le port en fonction de votre breadboard
  - On souhaite lire une nouvelle valeur tous les 100 ms.
- A quoi sert le bloc Gain=100/1024? Comment changer le gain si l'on souhaite normaliser la valeur lue entre 0 et 10?
- Ajouter un bloc pour visualiser la valeur lue
- Suivre la même procédure (pages 9-11) pour exécuter le modèle sur la carte Arduino
- Faire varier la valeur du potentiomètre et visualiser la valeur lue sur le scope.

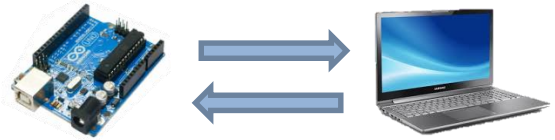
Ex3 : Détecteur



Nous allons utiliser le potentiomètre comme un capteur. Si la valeur lue dépasse une certaine valeur, la LED s'allume.

- Ouvrir le modèle **Detecteur.slx**
- Compléter le modèle en copiant des parties des modèles **LED** et **Potentiomètre**
- A quoi sert le bloc **Switch**? Mettre un seuil approprié.
- Suivre la même procédure (pages 9-11) pour exécuter le modèle sur la carte Arduino
- Vérifier que la LED s'allume lorsque vous dépassez la valeur de votre capteur. Vous pouvez tester différentes valeurs de seuil pendant que le modèle s'exécute sans avoir besoin de l'arrêter et le relancer.

Ex4 : Un robinet éclairé:
Adapter la couleur en fonction de la température

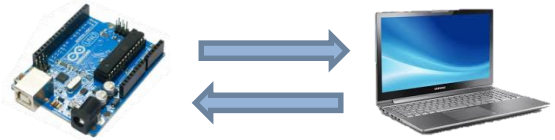


Nous souhaitons maintenant modifier la couleur de la LED en fonction de la température. On utiliseras ici le potentiomètre pour simuler un capteur de température.

Nous aurions pu ajouter un autre switch mais le modèle deviendrait plus complexe. Pour tester différents cas, il est plus facile de le faire avec du code en utilisant la structure if, elseif, else…

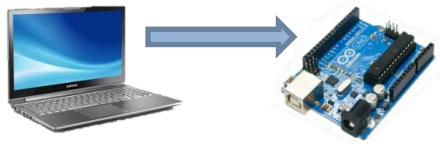
- Ouvrir le modèle **Robinet.slx**. On a remplacé le bloc **Switch** par un bloc **MATLAB function** qui contient une fonction en code MATLAB. Compléter le code pour obtenir le fonctionnement du schéma ci-dessus. Et ajouter les blocs arduino pour allumer la LED.
- Suivre la même procédure (pages 9-11) pour exécuter le modèle sur la carte Arduino
- Maintenant nous ne pouvons plus modifier les seuils de couleur pendant l'exécution du modèle (ex: vert si T>33°). Comment modifier le modèle pour que ça soit de nouveau possible?

D’autres exercices…



- L’intensité de la LED varie en fonction de la valeur du potentiomètre.
- L’intensité de la LED croit puis décroit avec une périodicité de 2s

Ex6 : Un modèle autonome sur la carte Arduino



Dans tous les exemples précédents, nous avons utiliser le mode external qui maintient une communication entre le PC et la carte Arduino pour afficher des signaux, modifier des paramètres…

Ce mode est très utile pour améliorer son modèle rapidement, régler ses paramètres et trouver d'éventuels bugs.

Une fois que l'on est satisfait, on peut charger ce modèle sur le carte pour qu'il fonctionne de manière autonome. Plus besoin du PC, juste d'alimenter la carte.

Ouvrez le modèle de votre choix et cliquez sur Deploy to Hardware 

Une fois le modèle chargé, vous pouvez débrancher la carte, fermer MATLAB, rebrancher la carte. Le programme est toujours chargé sur la carte.