Hace un tiempo que venía pensando en hacer algo con Arduino que uniera el mundo físico con el de la programación. Sabía que Arduino permite obtener información de diferentes sensores y generar eventos a partir de ellos. El primer ejemplo es hacer que cuando se presione un botón pulsador se pueda mantener encendido un led por algún tiempo.

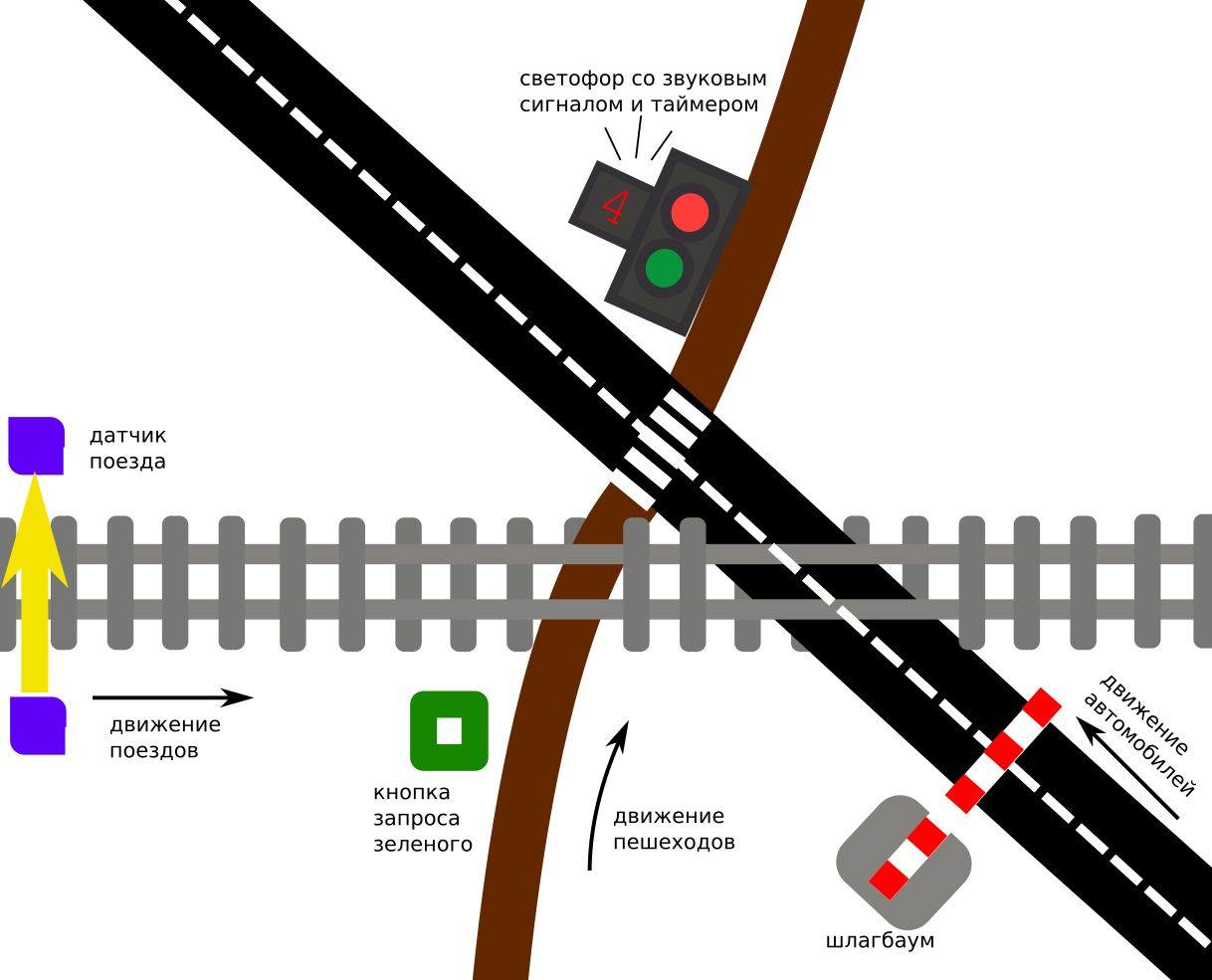
Comencé por hacer un curso en <https://www.coursera.org/> que brindaba la Universidad Autónoma de México, pero me pareció extremadamente básico (creo que el público objetivo eran chicos que apenas terminan la primaria). Luego encontré otro curso en la misma página pero brindado por el Instituto de Física y Tecnología de Moscú el realmente me pareció muy bueno.

**El desafío**

Este curso tenía como tarea crear un robot (en su definición más formal) el cual controlara un modelo que representara un cruce de las vías del tren con un cruce peatonal y una vía para autos.

El modelo debía cumplir con las siguientes reglas:

* Si aparece el tren, el semáforo se tiene que poner en rojo y bajar la barrera.
* Si un peatón quiere cruzar debe presionar el botón el cual baja la barrera y pone el semáforo en verde.
* Una vez que paso el tren se debe poner el semáforo en rojo y abrir la barrera.
* Si el peatón presiona el botón mientras pasa el tren, una vez que este termina de pasar el semáforo debe ponerse en verde y mantener la barrera baja.
* El peatón deberá esperar 18 seg desde la última vez que se le dio paso hasta que se le permita nuevamente.
* La luz verde del semáforo se mantendrá encendida durante 5 seg y después será intermitente por los próximos 4 seg.



Al parecer estas reglas son sencillas pero fue realmente desafiante lograrlas.

**Comiendo al elefante**

En este caso los pequeños bocados fueron:

1. Planeación. Hacer un diagrama del mundo físico, la señal eléctrica y el código.
2. Diseño. Hacer un diagrama de estados
3. Construcción. Construir el código
4. Implementación. Construir el modelo físico

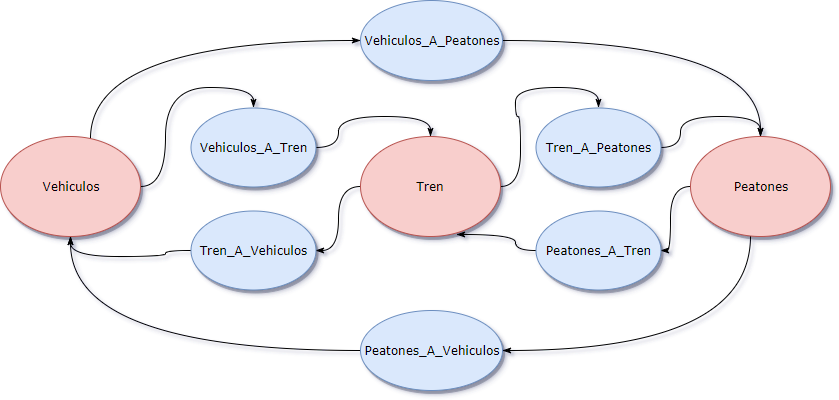
**Planeación**

Sabía que tenía ciertos requerimientos que se iban a convertir en componentes específicos, como por ejemplo para el semáforo necesitaba dos led uno rojo y otro verde, esto a nivel de la señal eléctrica requiere de un pin de la placa Arduino para cada uno. Para detectar el tren necesitaba algún tipo de sensor de presencia y decidí usar un medidor de distancia por ultrasonido (podría haber usado uno por IR, o un sensor de luz, etc.). En este caso el sensor requiere 2 pines más. También necesitaba el botón pulsador para los peatones y un servo para subir y bajar la barrera que habilita el tránsito de los autos. Otros componentes son: el servo para manejar la barrera, el buzzer para la señal sonora del aviso del tren y el 7 Segment display para mostrar la cuenta atras para habilitar nuevamente el botón de los peatones.

|  |  |  |  |
| --- | --- | --- | --- |
| Mundo Físico | Señal Eléctrica | Cant Pin | Código |
| Led Rojo | Digital | 2 | greenTrafficLight() |
| Led Verde | redTrafficLight() |
| Servo | PWM | 1 | openBarrier() |
| closeBarrier() |
| Press Button | Digital | 1 | digitalRead() |
| Ultrasonido | Analógico | 2 | measure() |
| Buzzer | Digital | 1 | beep() |
| 7 Seg Display | Digital | 7 | clearDisplay() |
| display() |

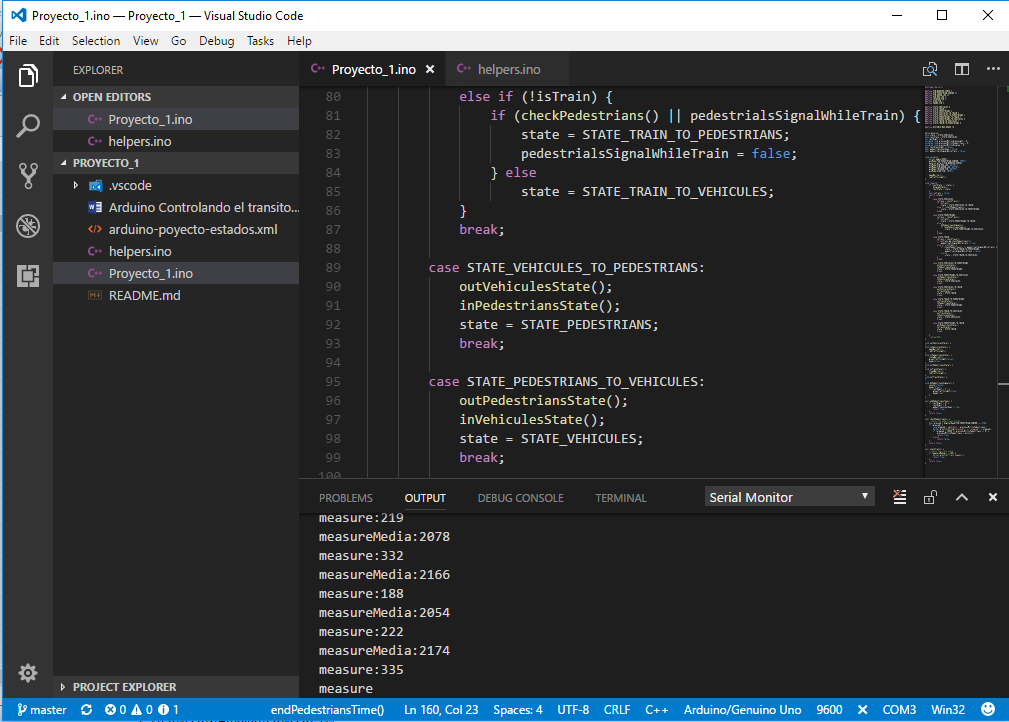
**Diseño**

Definí una máquina de estados para representar sus estados y sus transiciones ya que son muy marcados, por ejemplos no hay manera de que cuando este habilitado el paso del tren se habilite a los peatones o los vehículos. Por otro lado, las transiciones también son muy representativas, o sea, si estaba pasando el tren y fue presionado el botón de los peatones la transición es TREN\_A\_PEATONES con los cual es más fácil decidir mantener baja la barrera y poner el semáforo en verde.



**Construcción**

Comencé usando el IDE propio de Arduino, pero realmente no es más que un notepad con los botones de validar y subir, por lo que casi lo descarte al momento. Buscando un IDE mejor para trabajar encontré un plugin para Visual Studio Code que integra con Arduino el cual me pareció bastante aceptable. Tener en el IDE completador de código, descripción de las variables, ayuda sobre las funciones, poder engancharlo con el GIT, etc. Todas funcionalidades que no tiene el IDE propio de Arduino. Este plugin además de permitir validar (Ctrl-Alt-R) y subir (Ctrl-Alt-U) el código al Arduino, permite ver la consola del puerto serie y hacer un debug del programa (algo que no probé). Otra decisión de código fue separar en un segundo archivo (Helpers.Ino) las funciones que no son específicas de este proyecto, por ejemplo el que hace la medición del sensor o el que sube o baja la barrera, dejando en el archivo principal (Proyecto.Ino) las funciones que tienen lógica de negocio propia del proyecto.



**Implementación**

Arranque por poner los leds y el botón para generar el cambio de estado para habilitar a los peatones. Después puse la barrera y el estado de los vehículos y por último el sensor del tren con sus cambios de estados y transiciones.

En ese momento me di cuenta que el sensor a veces tiraba un valor muy bajo y que por lo que llegue a leer puede deberse a la baja calidad de los componentes, la interferencia en los cables o los cambios de voltajes generados posiblemente por el servo de la barrera. Para solucionarlo redefiní el método que mide la distancia para que devuelva el promedio entre 10 lecturas, de esa manera se filtran los valores extremos.

**LINKS**

* Repositorio del proyecto: <https://github.com/MarceloMosquera/ArduinoProyecto>
* Curso: https://www.coursera.org/learn/arduino/home/welcome
* TinkerCad: <https://www.tinkercad.com/things/6sMedEaTgCl-proyecto>
* Diagrama: https://easyeda.com/editor