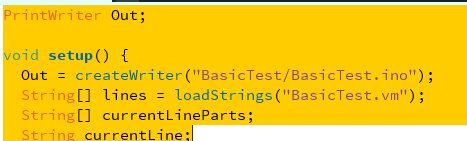
Documento De Funcionamiento Para BasicTest

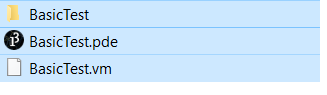
Este trabajo consiste en leer un archivo .VM y hacer una traducción por medio de processing para que el archivo pueda ser leído en la plataforma de Arduino.

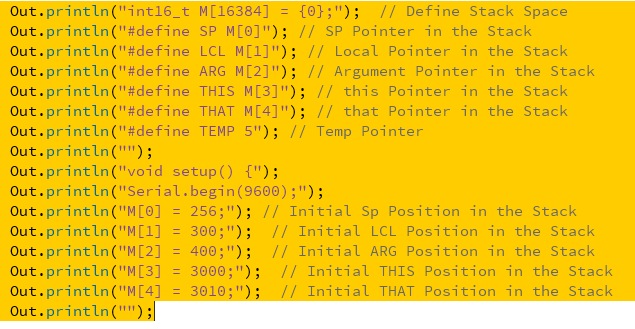
Para esto empezamos a utilizar algunas funciones específicas del lenguaje de processing( Java) para facilitar el proceso de lectura y traducción del archivo



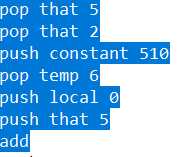
Primero creamos una variable PrintWriter que nos permitirá escribir dentro de un archivo de salida, Luego en el Setup del programa definimos el nombre del archivo de salida y la dirección, en este caso el archivo de salida llevará extensión. ino y se almacenará en dentro de la carpeta “Basictest” con el nombre de BasicTest.ino

Luego se crea un arreglo llamado lines que se encargara de leer cada una de las líneas de el archivo .vm que se encuentra dentro de la carpeta donde está ubicado el programa de modo tal que tendríamos algo así :

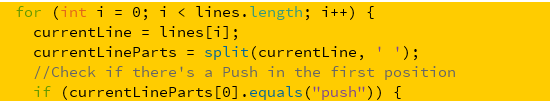




Antes de iniciar la traducción nos aseguramos de declarar las variables que necesitaremos en Arduino para el correcto funcionamiento del programa, en este caso necesitamos definir el tamaño del Stack, el Puntero de Sp y de cada uno de los diferentes segmentos de la memoria en el Stack, además de esto debemos también definir la posición inicial de cada uno de estos segmentos a excepción del Temp, también inicializaremos el serial de Arduino para poder visualizar los resultados del vector de prueba y compararlos con los del archivo.vm.



Para este ejemplo especifico debemos hacer varios push y pop en los diferentes segmentos de la memoria además de algunas operaciones logicas y aritmeticas.



Para esto tendremos el arreglo currentLineParts que nos servirá para almacenar por aparte cada parte de las diferentes líneas del archivo.Vm utilizando la función Split que se encargara de separar las cadenas cada vez que exista un espacio y la string, currentLine almacenará las diferentes posiciones de las lines del archivo.vm.

Luego lo primero que va a verificar es que exista un push en la primera posición de una de las tantas líneas que hay en el archivo .vm

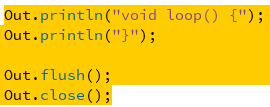


A continuación lo que el programa se encargara de hacer es asignarle a ese push un segmento determinado que estará escrito en el archivo .vm , eso incrementara el stack del segmento determinado con el numero que hay en la posición 3 de las líneas del archivo ( CurrentLineParts[2] ) .



De la misma forma como lo vimos anteriormente se hará el Pop, la gran diferencia es que ahora no estaremos incrementando el stack de cada uno de los segmentos sino que lo estaremos Decrementando, ya que el pop es una operación contraria al push.

Para las operaciones Aritmerico-Logicas se hará el mismo procedimiento realizado anteriormente en el documento de StackTest.



Finalmente abriremos y cerraremos la función loop para un correcto funcionamiento dentro del IDE de Arduino, y utilizaremos la función Flush para asegurarnos que no quede ningún dato en el buffer y la función close para cerrar el archivo una vez todo el proceso halla finalizado.