UNIVERSIDAD AMERICANA DE ACAPULCO

"EXCELENCIA PARA EL DESARROLLO"

FACULTAD DE INGENIERÍA EN COMPUTACIÓN

INCORPORADA A LA UNIVERSIDAD NACIONAL
AUTONÓMA DE MÉXICO
CLAVE DE INCORPORACIÓN 8852-16

"PROTOTIPO INTERACTIVO DE APRENDIZAJE DE FRACCIONES PARA EDUCACIÓN PRIMARIA"

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

PRESENTAN
FÁTIMA GUTIÉRREZ ALEMÁN
ZULY KARLA RAMÍREZ ROJAS

DIRECTOR DE TESIS
M.C. DAVID JAIME GONZALEZ MAXINEZ

ACAPULCO, GUERRERO, ENERO 2020.

AGRADECIMIENTOS

En primera instancia agradecemos a Dios, por permitirnos culminar este propósito, brindándonos salud, sabiduría, serenidad, agilidad y visión.

A nuestro asesor; el Dr. David Maxinez, por darnos soporte y apoyo en cada paso de este proceso. Agradecemos con creces el ayudarnos a lograr esta meta, compartiéndonos parte de sus conocimientos que facilitaron la realización de este proyecto.

DEDICATORIA

Dedicado al soporte primordial de mi vida, mi familia. A mis padres, que su sacrificio se vea gratificado con la culminación de esta etapa profesional. A mis hermanos, que siempre han sido ejemplo, apoyo y motivación a lo largo de mi vida.

Este proyecto me ayudo a superar una difícil etapa personal que la vida me había presentado. Elevó las expectativas que tenía respecto a mí misma. Me brindó más razones para desarrollarme como profesionista y como persona.

Fátima Gutiérrez Alemán

DEDICATORIA

Primero que nada, quiero dedicar esta tesis a Dios, por permitirme llegar hasta este momento.

A mi Papá por impulsarme, apoyarme y estar a mi lado durante estas etapas de mi vida, a mi madre que, aunque no está conmigo en este momento siempre me alentó y me enseñó a ser independiente a trabajar para ser una profesionista y ser mejor persona cada día

A mis hermanos que me han apoyado con sus ocurrencias y su forma de ser para seguir avanzando.

A mi esposo porque me dio apoyo en los momentos más difíciles de mi vida.

A mi tía porque me enseñó a luchar por lo que uno quiere.

Y por último quiero agradecer a mi amiga por ser mi apoyo y realizar en conjunto esta tesis.

Zuly Karla Ramírez Rojas

INDICE DE CONTENIDO

CAPÍTULO 2.- DESCRIPCIÓN FUNCIONAL DEL LABORATORIO PARA EL APRENDIZAJE DE FRACCIONES

CAPÍTULO 3.- LABORATORIO DE FRACCIONES: MONTAJE

CAPÍTULO 4.- LABORATORIO DE FRACCIONES: PROGRAMACIÓN

INTRODUCCIÓN

Este proyecto surge como una necesidad de innovar con nuevas estrategias para enseñar el manejo y comprensión de las fracciones matemáticas. la propuesta considera la participación interactiva y no solo receptiva del alumno para comprender el significado de una fracción, para esto se diseña una estructura interactiva y visual donde el alumno participa introduciendo datos manualmente como respuesta a una pregunta realizada por el profesor, como estímulo a una respuesta correcta el proyecto considera una carrera de autos incorporados en la estructura y que se mueven a través de una etapa de potencia y un microcontrolador basado en la plataforma Arduino UNO.

PLANTEAMIENTO DEL PROBLEMA

Nuestro problema consiste en diseñar un laboratorio interactivo de manejo y uso de fracciones, que permita al alumno interactuar y crear su propio conocimiento “constructivismo” mediante la visualización de resultados en pantallas de tecnología led en las cuales el alumno puede observar si el resultado ingresado en el laboratorio es correcto o incorrecto el premio a un resultado correcto es hacer que un móvil “auto de juguete” avance unos centímetros sobre una pista dispuesta dentro del laboratorio para el desplazamiento del móvil.

JUSTIFICACIÓN

El aprendizaje, manejo y uso de las fracciones a nivel primaria es un problema a nivel nacional. El no existir materiales educativos para esta actividad fuera de los tradicionales ─gis y borrador─ hacen que el alumno se distraiga debido a que no es lo mismo un material con el que pueda interactuar a otro donde el conocimiento solo es abstracto. En este trabajo se desarrolla un prototipo electrónico interactivo para el aprendizaje de fracciones a nivel primaria específicamente de tercero a sexto grado.

HIPÓTESIS

Mostrar que las nuevas tecnologías interactivas incorporadas en la educación permiten que el alumno obtenga un conocimiento basado en sus propias experiencias de aprendizaje y en forma paralela dar al docente una ayuda pedagógica tecnológica que le permita transmitir de manera más fácil y agradable el conocimiento sobre el tema de fracciones.

Uno de los beneficios que tiene este prototipo interactivo es que de manera lúdica el alumno comprenderá, entenderá y resolverá problemas de fracciones más rápidamente y el aprendizaje en ellos será más significativo.

Por medio de este prototipo didáctico el docente:

- Motivará al alumno a aprender el tema de fracciones.
- Tendrá la facilidad de enseñar fracciones de manera interactiva.
- Resolverá problemas de fracciones estimulando el aprendizaje con un premio “movimiento del móvil hasta llegar a la meta”
- Fomentará el trabajo en equipo.

OBJETIVO GENERAL

Desarrollar un prototipo interactivo y visual para el aprendizaje y uso de fracciones matemáticas.

OBJETIVOS ESPECÍFICOS

Diseño y construcción ergonométrico de una estructura para el manejo interactivo de fracciones matemáticas.

Diseño, construcción y programación de pantallas de matriz de 8X8 para que el lector pueda visualizar los resultados.

Diseño, construcción y programación de pantallas LCD para que el lector pueda visualizar los resultados.

Diseño, construcción y programación de módulos de control interactivos utilizados para que el lector ingrese la respuesta solicitada a una pregunta específica.

Capítulo 1.

ESTUDIO DE LAS FRACCIONES A NIVEL PRIMARIA

INTRODUCCIÓN

La enseñanza de las fracciones numéricas tradicionalmente despierta un gran reto para los profesores que la imparten, en nuestro país el aprendizaje en los niveles de primaria resulta lento, difícil de aprender y en ocasiones aburrido para el estudiante, el poco interés mostrado por el alumno en esta área es de diversa índole por ejemplo, las operaciones y resultados son abstractos no son tangibles, el termino fracción no es entendido correctamente y no existen el material educativo especializado para esquematizar la fracción. El Abaco; figura 1.1, es uno de los pocos materiales educativos donde el alumno interactúa con el uso y manejo de fracciones.

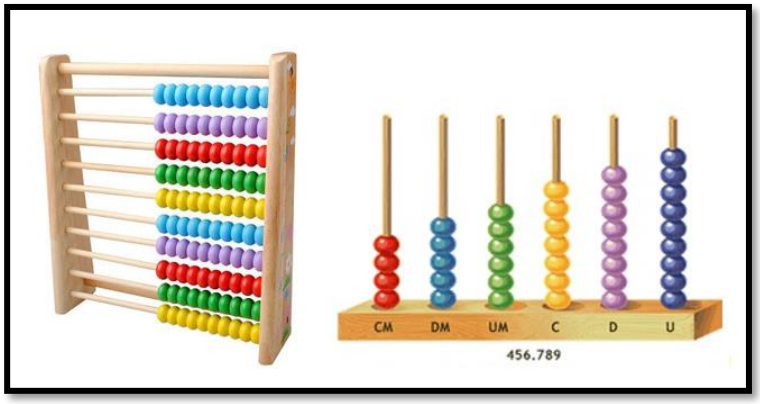

Figura 1.1. Abaco

La presente tesis propone el uso de tecnologías electrónicas y computacionales actuales para el diseño de materiales educativos que puedan provocar un revulsivo en el aprendizaje y uso de las fracciones. Este prototipo didáctico puede utilizarse en nivel primaria de primero hasta sexto grado.

1.1. Matemáticas y Fracciones

Las matemáticas son la ciencia que estudia las propiedades y relaciones entre entidades abstractas como números, figuras geométricas o símbolos. Es un conjunto de lenguajes formales que pueden ser usados

como herramienta para plantear problemas de manera ambigua en contextos específicos que permite resolver problemas en diversos ámbitos, como se muestra en la figura 1.2.

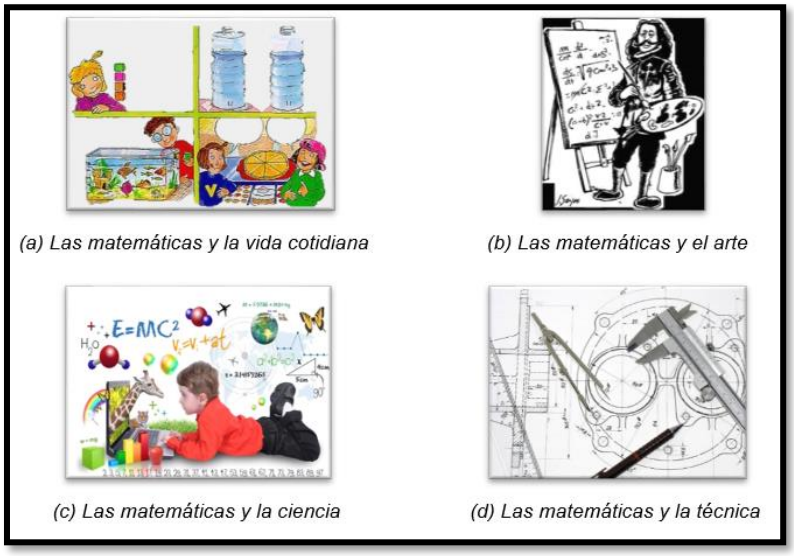

Figura 1.2. Las matemáticas en la vida del ser humano.

1.1.1 ¿Por qué y para qué enseñar matemáticas?

La matemática es una parte de la educación general que inicia en los primeros años de la educación a nivel primaria y que forma parte de la herramienta básica donde los alumnos puedan adquirir competencias numéricas, geométricas, estadísticas y de medida suficientes para desenvolverse en su vida diaria, así como para leer e interpretar información matemática que aparece en los medios de información.

1.1.2 ¿Que son las fracciones?

Una fracción es un número, que se obtiene al dividir un entero en partes iguales. Por ejemplo, cuando decimos comer una cuarta parte de una pizza, estamos dividiendo la pizza “el entero” en cuatro partes iguales y consideramos comer una de esas cuatro partes, como se muestra en la figura 1.3.

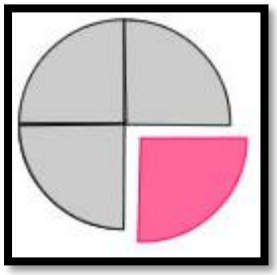

Figura 1.3. Representación gráfica de una fracción.

La fracción ¼ representa matemáticamente el entero dividido entre el número de partes.

La fracción ¼ está formada por dos términos: **el numerador y el denominador**, figura 4. El **numerador** es el número que está sobre la raya fraccionaria; es el número de partes que se considera de la unidad o total y el **denominador** es el que está debajo de la raya fraccionaria; es el número de partes iguales en que se ha dividido la unidad o total. Figura 1.44

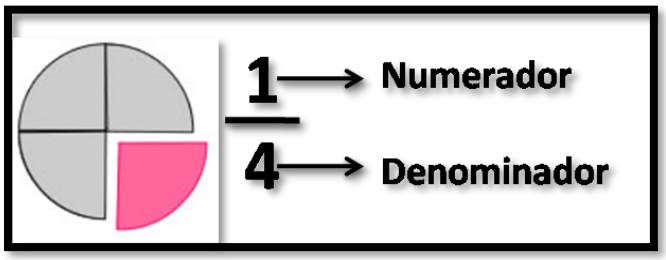

Figura 1.4. Partes de una fracción.

1.1.3. Tipo de fracciones.

- **Fracciones propias:** Las fracciones propias son aquellas cuyo numerador es menor que el denominador. Su valor debe ser menor que un entero. Por ejemplo, ¼ es menor que un entero, figura 1.5.

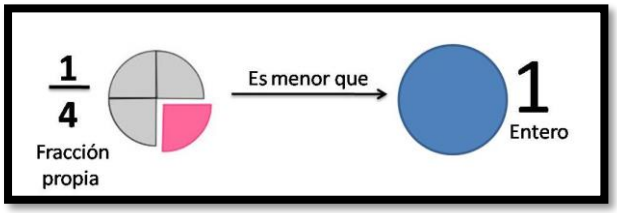

Figura 1.5. Ejemplo de fracción propia.

- **Fracciones impropias:** Las fracciones impropias son aquellas cuyo numerador es mayor que el denominador. Su valor es mayor que 1, figura 1.6

Figura 1.6. Ejemplo de fracción impropia

- **Fracciones unitarias:** Las fracciones unitarias tienen el numerador igual al denominador. Ejemplo figura 1.7.

Figura 1.7. Fracción unitaria

- **- Fracciones decimales:** Una fracción decimal es aquella que tiene por denominador la unidad seguida de ceros: 10, 100, 1000. Ejemplo en la figura 1.8.

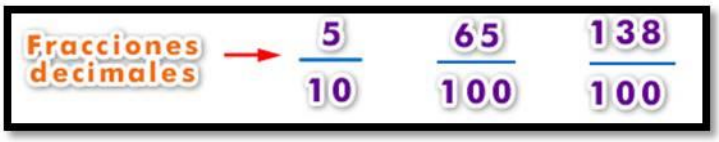

Figura 1.8. Ejemplo de fracciones decimales

- **Fracciones equivalentes:** Las fracciones equivalentes representan la misma parte de una unidad o entero, figura 1.9.

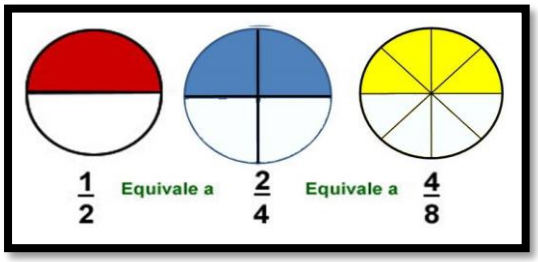

Figura 1.9. Ejemplo grafico de las fracciones equivalentes.

- **- Fracciones Mixtas:** Se caracterizan por ser una combinación de un número entero y una fracción. La fracción mixta se obtiene de dividir el numerador entre el denominador, siempre y cuando sea una fracción impropia. Se escribe el resultado de la división como un número entero, después se escribe lo que sobro de la división para poner la fracción. Como se muestra en la figura 1.10.

Figura 1.10. Ejemplo de convertir la fracción impropia a una fracción mixta.

1.2 Materiales educativos.

Generalmente, la idea de enseñar fracciones se basa en memorizar reglas y conceptos, pero una mejor forma para su aprendizaje es manipulando y comprobando los resultados utilizando materiales físicos, figura 1.11.

Figura 1.11. Enseñanza de fracciones mediante material educativo

1.2.1. ¿Qué son los materiales educativos?

El material educativo es un medio que sirve para estimular el proceso educativo, permitiendo al niño adquirir informaciones, experiencias, desarrollar actitudes y adoptar normas de conductas de acuerdo a las competencias que se quieren lograr.

Un material educativo tiene como objetivo, propiciar ambientes, experiencias de aprendizaje e interacciones humanas positivas que fortalezcan el proceso educativo. En general ayudan en el desarrollo de habilidades como:

- Creatividad
- Habilidad manual.
- Capacidad para aprender por cuenta propia.

- Permiten el aprendizaje de conceptos matemáticos básicos.
- Permite el desarrollo de la habilidad del pensamiento lógico matemático.
- Fomentan la atención y la concentración.
- Fomentan valores como la cooperación y la colaboración.
- Fomenta el desarrollo competitivo en el alumno.

1.3. Funciones de los materiales educativos.

Unas de las principales funciones de los materiales educativos es ayudar al docente en el proceso de la enseñanza-aprendizaje buscando con ello despertar en el estudiante el interés por una determinada actividad o buscando la comprensión de un problema de índole generalmente abstracto.

Los materiales educativos actualmente se pueden encontrar en forma física o en formato digital y dentro de ellos una serie de subcategorías, sin embargo, de manera general solo diferenciaremos entre un material educativo físico y uno virtual.

1.4 Materiales educativos digitales.

Los materiales educativos digitales son aquellos que están hechos mediante lenguajes de programación y manipulados mediante un sistema de cómputo. Estos programas se diseñan para una determinada función tal y como se muestra en la tabla 1.1.

FUNCIÓN	DESCRIPCIÓN	EJEMPLOS DE SOFTWARE
Informativa	A través de sus actividades presentan unos contenidos que proporcionan información estructuradora de la realidad a los estudiantes.	Programas tutoriales, Simuladores, Bases de datos.
Instructiva	Dirigen las actividades de los estudiantes en función de sus respuestas y progresos, para dar cumplimiento a los objetivos educativos.	Programas tutoriales.
Evaluadora	Permite responder inmediatamente a las respuestas y acciones de los estudiantes, les hace especialmente adecuados para evaluar el trabajo que se va realizando con ellos.	Los programas que incluyen un módulo de evaluación
Investigadora	Ofrecen a los estudiantes entornos donde investigar: buscar determinada información, cambiar los valores de las variables de un sistema, etc. Además, pueden proporcionar a los profesores y estudiantes instrumentos de gran utilidad para el desarrollo de trabajos de investigación que se realicen básicamente al margen de las computadoras.	Bases de datos, simuladores, Programas constructores y herramientas.

Expresiva	Son medios para representar conocimientos y formas de comunicación.	Procesadores de texto, editores gráficos, lenguajes de programación.
Metalingüística	Apoyan en el aprendizaje de los lenguajes propios de la Informática.	Sistemas operativos MS/DOS, Windows, lenguajes de programación.

Tabla 1.1 Clasificación de materiales educativos digitales.

1.4.1 Ejemplos de materiales educativos digitales.

A continuación, mostraremos algunos ejemplos de software que no solo es utilizado para el área de las matemáticas sino también para otras asignaturas.

- **GCompris**

Es una colección de juegos educativos compuesta por diferentes actividades para niños entre 2 y 10 años de edad, figura 12. Algunas actividades son las siguientes.

- Descubriendo la computadora: teclado, ratón, diferentes movimientos del ratón, etc.
- Aritmética: tabla de memoria, enumeración, tabla de doble entrada (balance), imagen espejo.
- Ciencia: El canal, El ciclo del agua, El submarino.
- Geografía: Coloca los países en el mapa.
- Juegos: ajedrez, memoria.
- Lectura: práctica de lectura
- Etc.

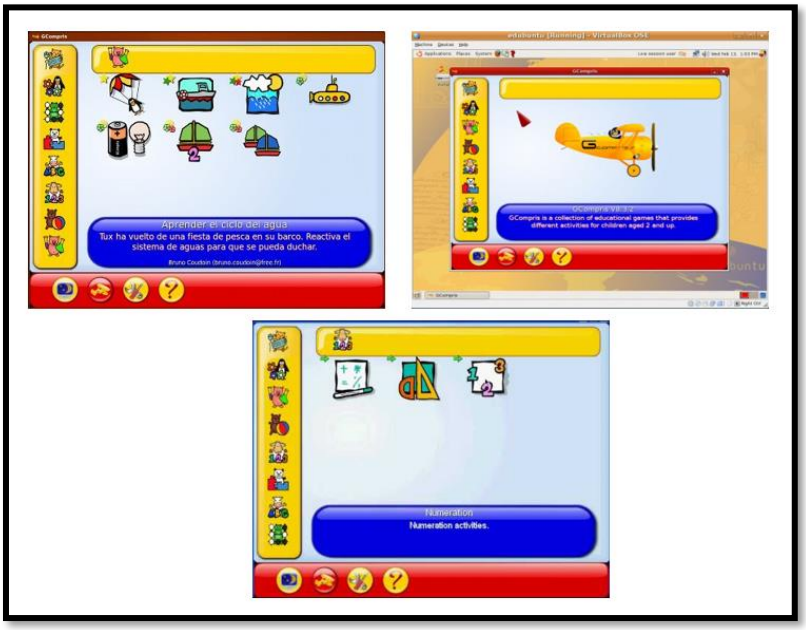

Figura 1.12. Plataforma del juego GCompris

- **Childsplay**

Es un juego con un conjunto de actividades para que los niños a aprendan el uso de la computadora y al mismo tiempo les enseña un poco de matemáticas, letras del abecedario, ortografía, coordinación de ojos y manos, etc., figura 1.13.

Este juego contiene varias actividades algunas de ellas son:

- Actividades de memoria que son divertidos para jugar y al mismo tiempo aprenden sonidos, imágenes, letras y números.
- Actividades que capacitan al niño a utilizar el ratón y el teclado.
- Actividades de juego puros como rompecabezas, pong, Pacman y billar.
- Soporte multilingüe, incluso lenguajes de derecha a izquierda (a través de Pango).

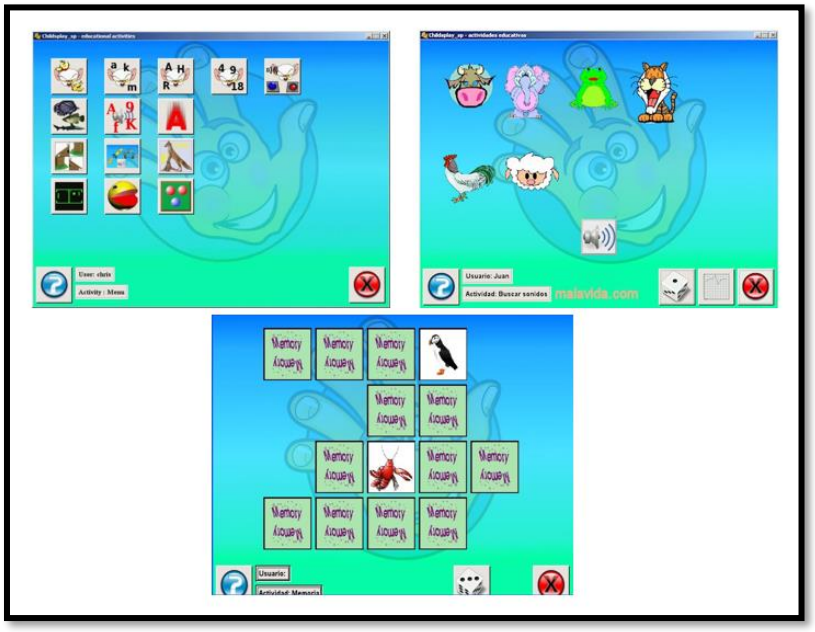

Figura 1.13. Plataforma del juego Childsplay

- **Kitsune**

Es un programa utilizado para resolver problemas de aritmética, encontrar un número determinado a partir de varios números y las cuatro operaciones aritméticas elementales. El programa muestra todas las soluciones (o las mejores aproximaciones) y permite configurar el tamaño del número final y cuántos números se utilizan para calcular el número final, figura 1.14.

El programa no necesita instalación: hay que ejecutar directamente el programa kitsune.exe.

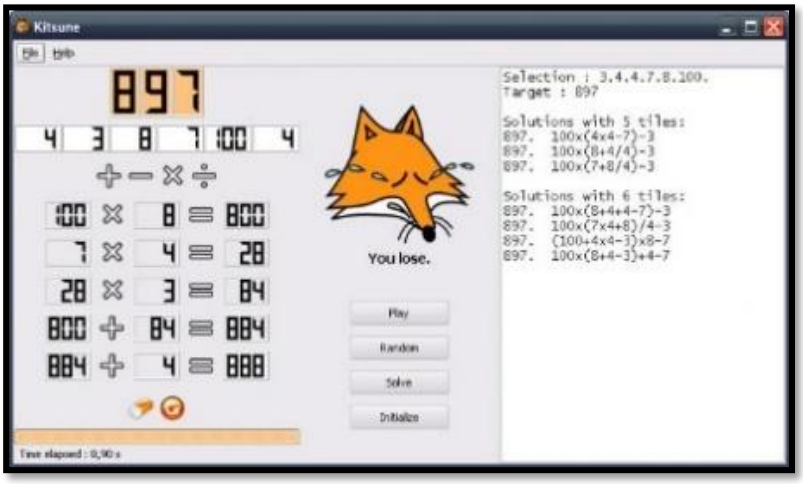

Figura 1.14. Plataforma del juego Kitsune

1.5. Materiales educativos físicos.

Los materiales educativos físicos también llamados “recurso educativo” son productos diseñados con intención didáctica, para apoyar el desarrollo de los procesos de aprendizaje y enseñanza. Estos materiales son empleados por el docente para apoyar, complementar, acompañar o evaluar el proceso educativo que orienta.

FUNCIÓN	DESCRIPCIÓN	EJEMPLOS DE SOFTWARE
Lúdica	Permite realizar actividades educativas mediante el entretenimiento.	Juegos educativos. Juego de bloques para armar
Innovadora	Utilizan tecnología recientemente incorporada a los centros educativos y, en general, suelen permitir muy diversas formas de uso. Esta versatilidad abre amplias posibilidades de	Varios.

	experimentación didáctica e innovación educativa en el aula.	

Tabla 1.2. Clasificación de material educativos físicos

1.5.1 Ejemplos de materiales educativos físicos.

A continuación, mostraremos algunos ejemplos de juegos que son utilizados como material didáctico para las diferentes asignaturas, pero de igual manera sirven para estimular y motivar al alumno a tener un mejor rendimiento en su aprendizaje.

- Juego de madera formas de colores.

Juego de madera formado por un tablero de 29 Cm, 15 anillas de colores y 5 piezas con los números del 1 al 5; se trata de colocar el número de anillas correspondiente al número indicado en la ficha, figura 1.15.

Figura 1.15. Juego de formas de colores

- Bloques de madera ilustrados.

Conjunto de 11 bloques con superficies coloreadas y diferentes tamaños, que emiten sonidos y ofrecen además la función de ábaco. Más que meros bloques, este es el juguete ideal para crear escenarios imaginativos que estimulan la creatividad y los sentidos del niño, figura 1.16.

Figura 1.16. Juego de Bloques

- Reloj madera puzzle

Reloj de madera de 30x30 Cm que es a la vez un puzzle de encaje con 12 piezas de colores y con distintas formas que indican las horas. Ayuda al desarrollo del razonamiento y de la lógica en los más pequeños, figura 1.17.

Figura 1.17. Juego de reloj de madera

- Juego de bloques de colores

Juego educativo de madera para la estimulación temprana e intelectual de bebés y niños, así como personas con capacidades diferentes.

Material didáctico Montessori, construido a partir de la madera de caucho, Los colores brillantes y formas estimulan el desarrollo capacidad operativa y práctica. Uso para niños a partir de 2 años de edad, figura 1.18.

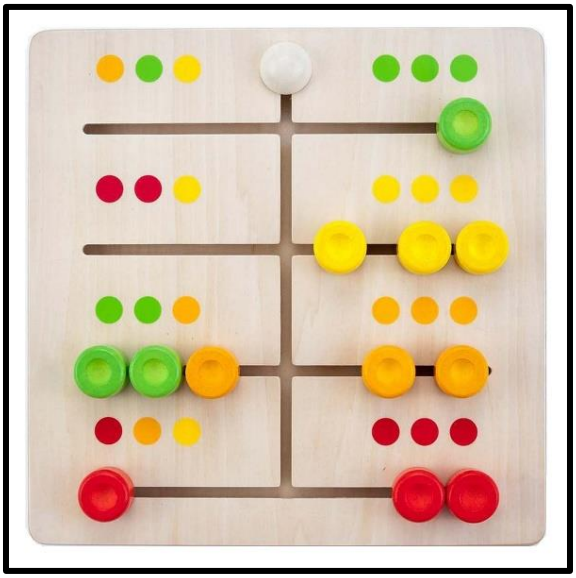

Figura 1.18. Bloques de colores

Capítulo 2

DESCRIPCIÓN FUNCIONAL DEL LABORATORIO PARA EL APRENDIZAJE DE FRACCIONES

INTRODUCCIÓN. De manera general y con la idea de que el alumno participe de manera interactiva con el aprendizaje y uso de fracciones se desarrolló un prototipo didáctico que puede utilizarse de manera simultánea por dos personas que compiten por la misma pregunta relacionada con el uso de fracciones, aquel cuya respuesta sea la correcta hace que un móvil se desplace de manera horizontal hasta llegar a la meta, el competidor en llegar primero a la meta será el ganador, en la figura 2.1 se muestran los móviles.

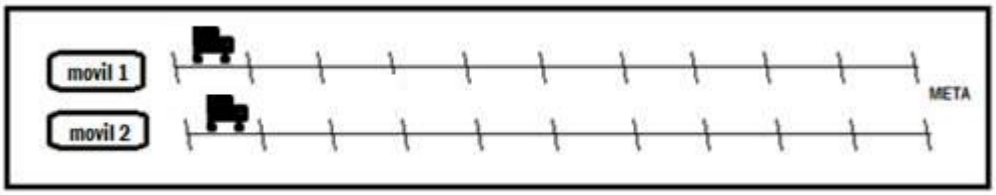

Figura 2.1. Carriles de autos.

2.1 Presentación del laboratorio

El proyecto global se divide en dos partes denominadas:

- **Visualización**
- **Estructura interna**

Visualización: En la figura 2.2 se muestra el diagrama esquemático del laboratorio como puede apreciarse el laboratorio tiene dos zonas de competencia una para cada jugador. Los elementos que lo conforman se enumeran a continuación:

1- Pantalla de cristal líquido LCD que da la bienvenida al jugador y muestra la fracción ingresada.
2- 5 matrices de leds 8x8 MAX7219 utilizadas para representar la fracción solicitada.
3- Bloque de 50 diodos emisores de luz “Leds” 5mm usados para representar el numerador de la fracción.

4- Bloque de 50 diodos emisores de luz “Leds” 5mm usados para representar el denominador de la fracción.

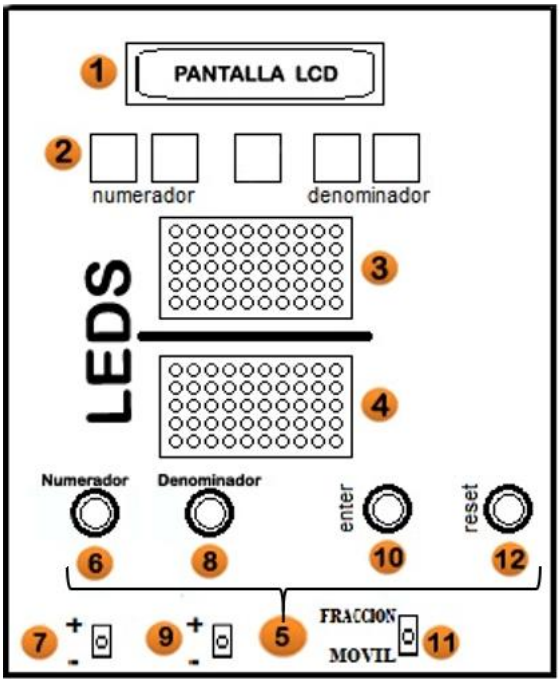

Figura 2.2. Componentes para visualizar la fracción.

5- La “sección 5” muestra cuatro botones tipo Arcade usados comúnmente en las consolas de videojuegos, estos son utilizados para monitorear y controlar las acciones del juego.

6- Botón utilizado para ingresar de uno en uno el numerador.

7- El interruptor tiene la función de incrementar (+) o decrementar (-) la fracción del numerador.

8- Botón utilizado para ingresar de uno en uno el denominador.

9- El interruptor tiene la función de incrementar (+) o decrementar (-) la fracción del denominador.

10- El botón “Enter” tiene dos funciones que dependen de su interruptor identificado como el número 11, si elegimos la función de **“Fracción”**, habilita las matrices, si elegimos la función de **“Móvil”** avanza el móvil.

11- El tercer interruptor ayuda al “botón enter” para activar los paneles de las matrices (función **“Fracción”**) o validar el avance del móvil (función **“Móvil”**); así como también ayuda al “botón reset” a retroceder el móvil (función **“Móvil”**).

12- Por último, tenemos el botón “Reset” que se utiliza para borrar las fracciones ya ingresadas en el prototipo didáctico y retroceder el móvil.

El diseño final se muestra en la Figura 2.3, como puede notarse existen dos móviles, uno para cada competidor y un tablero de visualización para cada participante.

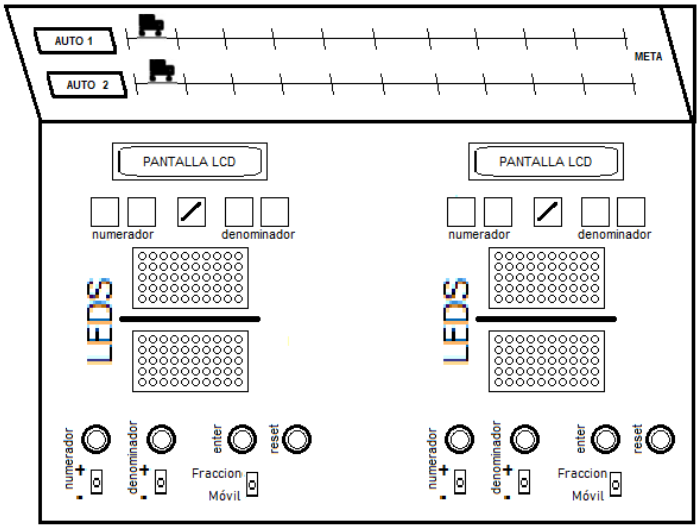

Figura 2.3. Diseño eléctrico completo del prototipo de material didáctico.

Estructura interna. En la parte interna la estructura es la siguiente:

- Existe un dispositivo lógico programable CPLD que controla cada uno de los bloques de 50 leds, estos, gracias a su gran capacidad de terminales de entrada/salida permiten encender o apagar cada uno de los leds utilizados en el numerador o denominador, en la figura 2.4 se muestra el esquema de conexión.

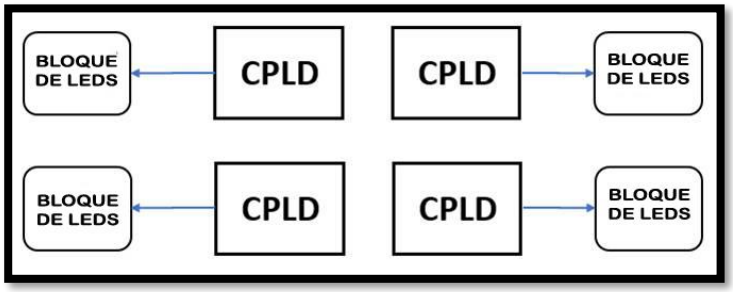

Figura 2.4. Conexión de CPLD.

- La parte encargada de monitorear el funcionamiento de las matrices de leds de 8x8 MAX7219 y el display de cristal líquido "LCD" se realizó mediante el microcontrolador ATMEL ATMEGA2560 de la tarjeta Arduino MEGA mostrada en la figura 2.5.

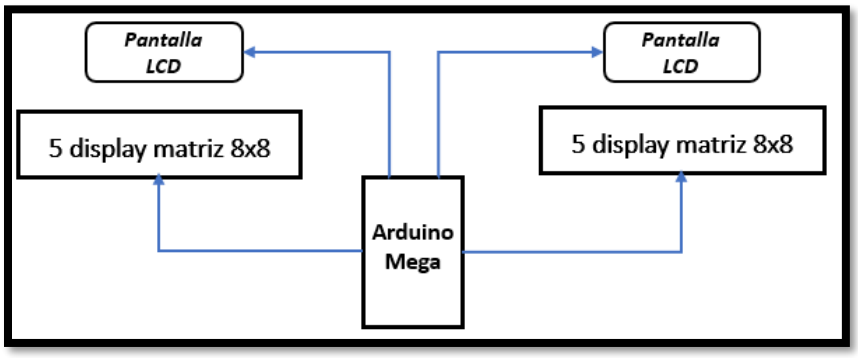

Figura 2.5. Conexión de LCD y matriz 8x8 al Arduino.

Igualmente, el Arduino Mega activa los botones e interruptores con los cuales ingresamos las fracciones y manipulamos los móviles en los carriles. Como se muestra en la figura 2.6

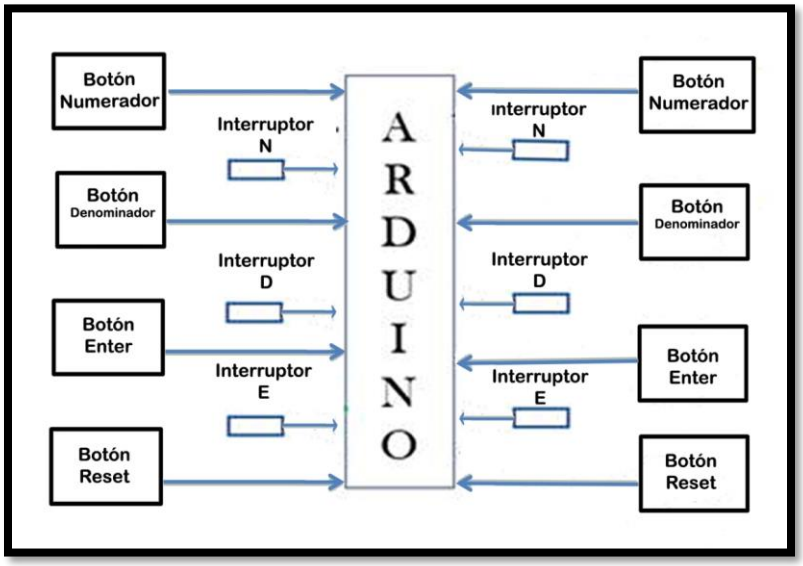

Figura 2.6. Conexión de las entradas de los botones e interruptores al Arduino.

Finalmente el microcontrolador ATMEGA2560 envía el pulso reloj a los CPLD, para que este a su vez controlen los bloques de Leds del numerador y del denominado de cada jugador, figura 2.7.

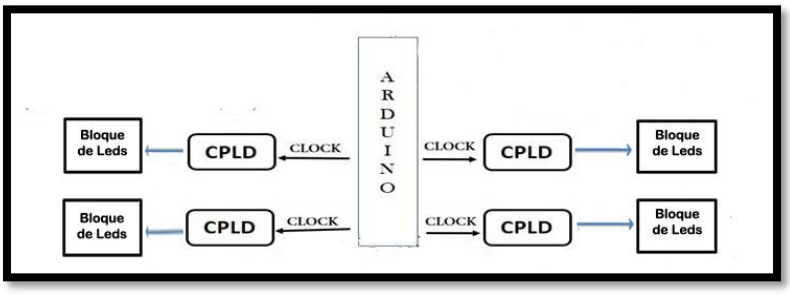

Figura 2.7. Pulso del reloj del Arduino al CPLD.

De una manera más global y generalizada, el funcionamiento de nuestro prototipo quedaría esquematizado tal y como se muestra en la figura 2.8.

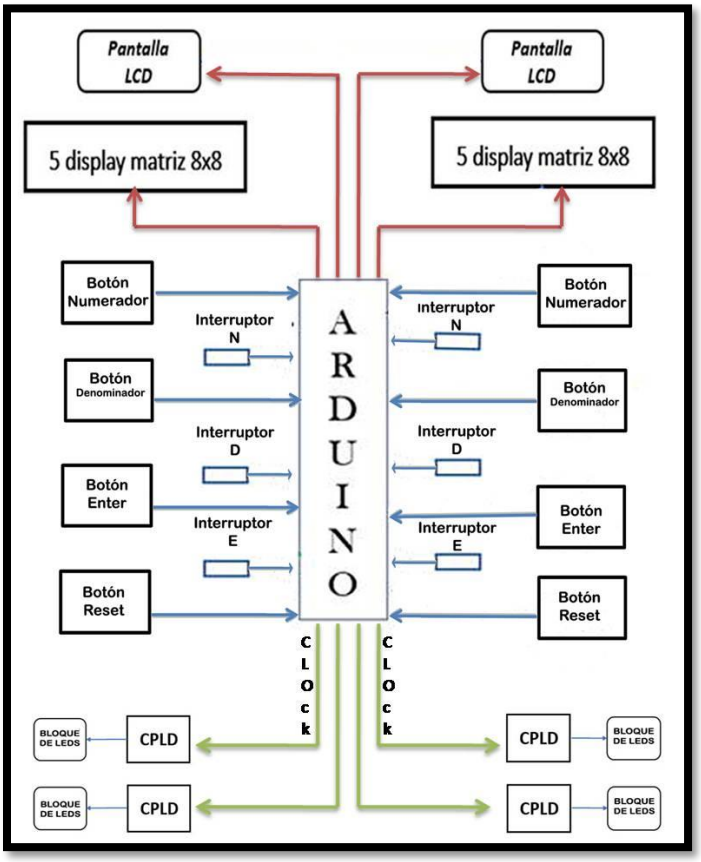

Figura 2.8. Arquitectura general del proyecto.

2.2. Descripción funcional del material educativo

Para explicar el uso y manejo del laboratorio de fracciones consideremos lo siguiente:

El profesor elige a dos alumnos para participar en una competencia de solución de fracciones, cada respuesta correcta a cada pregunta, permite avanzar el móvil, hasta llegar a la meta figura 2.9, estableciendo así al niño ganador.

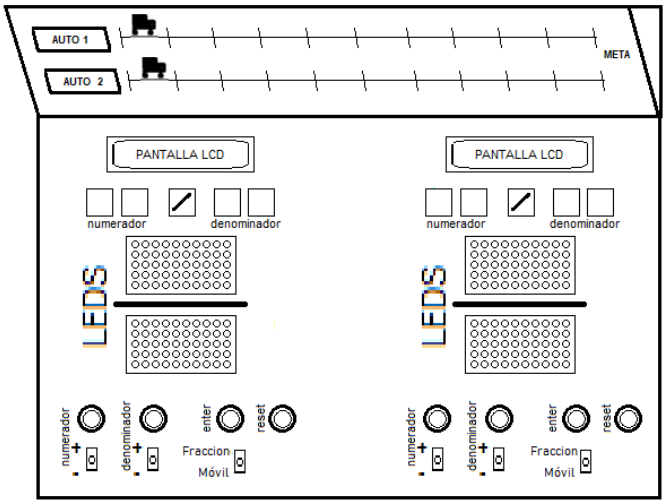

Figura 2.9. Vista superior del prototipo.

Procedimiento:

1.- Para comenzar a utilizar el prototipo es necesario realizar las siguientes instrucciones:

a) El participante deberá conectar el prototipo a la corriente eléctrica.
b) Verificar que las pantallas LCD estén encendidas y muestren mensaje de bienvenida al participante.
c) Todos los bloques de leds y matrices 8x8 MAX7219 deberán estar apagados.

2.- Inicialmente el profesor realiza la pregunta:

¿Qué fracción representa la parte iluminada de la figura 2.10?

Figura 2.10. Imagen del problema.

3.- La solución a este ejercicio es 1/8. Para ingresar la respuesta a esta pregunta el participante debe seguir la siguiente secuencia:

a) Inicialmente todos los interruptores deben estar hacia arriba figura 2.11.

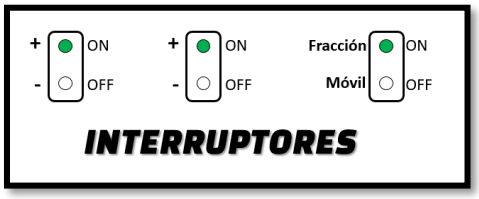

Figura 2.11. Posición de interruptores.

b) Presionar “Enter” para activar y encender las dos primeras matrices de 8x8 MAX7219 que corresponden al numerador de la fracción en el tablero del participante, como se muestra en la figura 2.12.

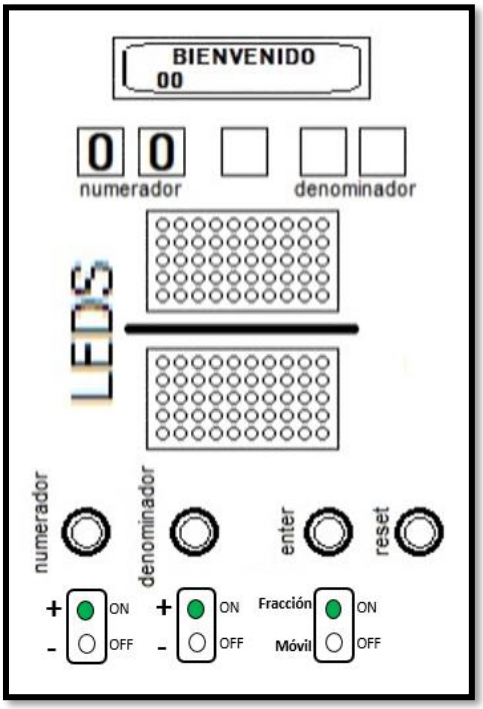

Figura 2.12. Módulos de matrices y leds activados para la parte del numerador.

c) Observemos que el botón, tanto del numerador como del denominador, cuentan con un interruptor que sirven para incrementar (+) y/o decrementar (-) la cifra que se esté ingresando, figura 2.13.

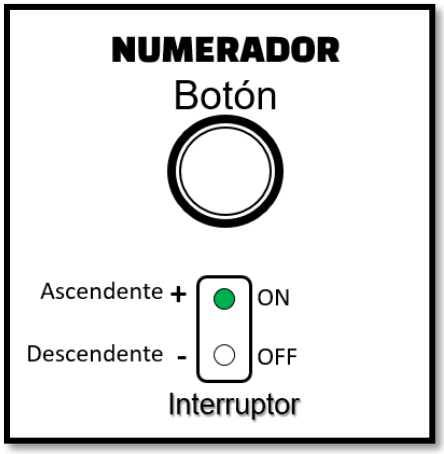

Figura 2.13. Botón numerador.

d) Pulsamos el botón “Numerador las veces necesarias hasta obtener el número deseado, en este caso se presionará solo una vez para ingresar el número 1.
e) Observemos en el tablero, el número 01 visualizado en las dos matrices 8x8 MAX7219 y en la pantalla LCD y encendido un led en el bloque de leds del numerador, figura 2.14.

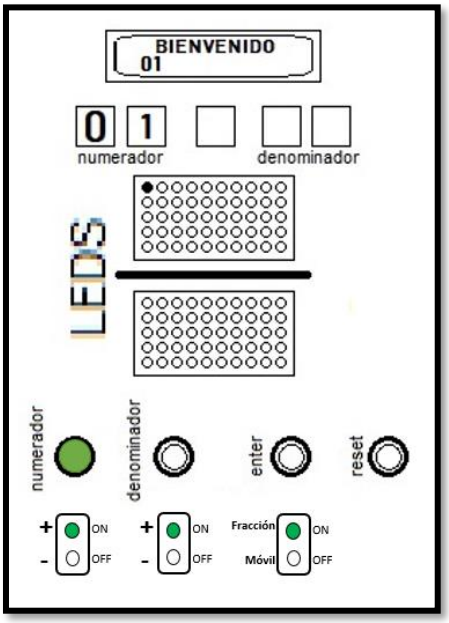

Figura 2.14. Parte del numerador ingresada.

f) Presionamos nuevamente el botón “Enter” para mostrar el símbolo de la diagonal en la tercera matriz 8x8 MAX7219.
g) Recordemos que los interruptores del numerador y denominador se encuentran en posición ascendente (+), por lo que no debe haber mayor problema al momento de ingresar las cifras.
h) Nuevamente volvemos a presionar “Enter” para activar y encender las últimas dos matrices 8x8 MAX7219 del tablero que corresponden al denominador de la fracción, tal y como se muestra en la figura 2.15.

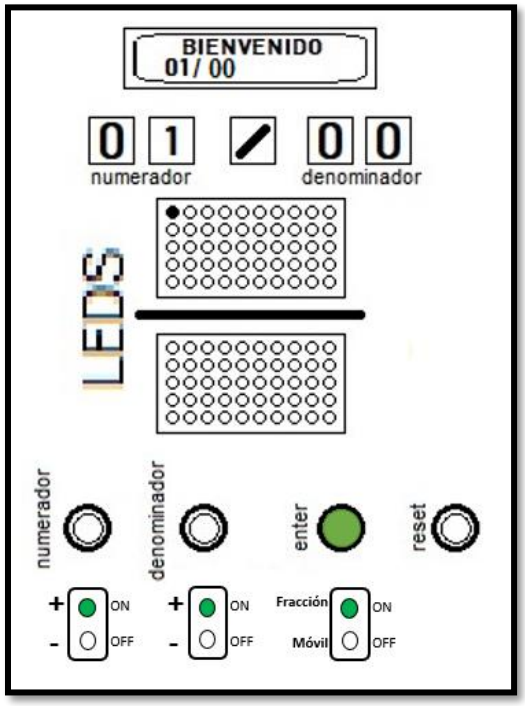

Figura 2.15. Módulos de matrices y leds activados para la parte del denominador.

i) Pulsamos el botón “Denominador” las veces que sean necesarias hasta llegar al número deseado; en este caso, el número 8.
j) Observemos el tablero, notamos que se visualiza la fracción “01/08” en la pantalla LCD y las matrices 8x8 MAX7219, por otra parte, en los bloques de leds notamos que en el primer bloque este 1 led encendido y en el segundo tenemos 8 leds encendidos, figura 2.16.

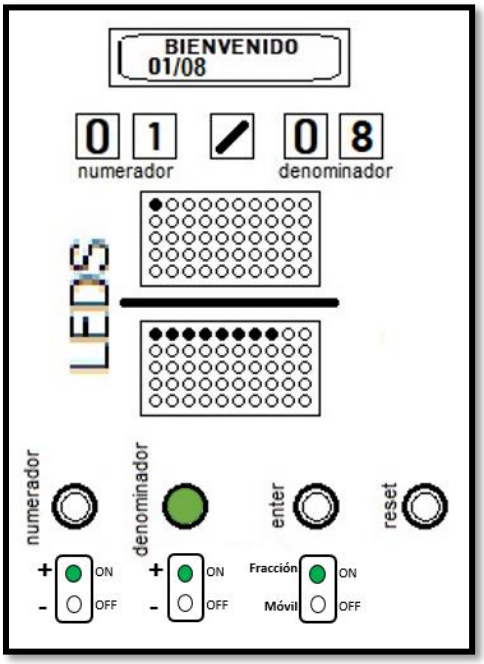

Figura 2.16. Parte del denominador ingresada.

k) Por último, cuando tengamos la fracción visualizada, el participante presionara una última vez el botón “Enter” para validar la respuesta, como se muestra en la figura 2.17; en este caso, el profesor verificara el resultado ingresado.

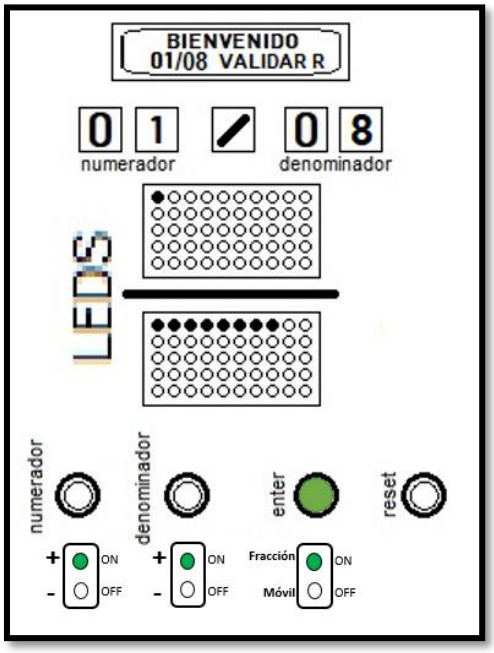

Figura 2.17. Visualización final de fracciones.

4.- Cuando el profesor valida la respuesta, indicará a los participantes quien ha contestado correctamente y deberá avanzar su carro (móvil) una posición hacia delante, de lo contrario, se quedará en el mismo lugar.

5.- Para este ejercicio de ejemplo vamos a suponer que ambos participantes ingresaron la respuesta correcta; es decir, 1/8. El profesor les autoriza a los dos participantes avanzar cada uno su auto (móvil).

6.- Para mover los móviles es necesario seguir las siguientes indicaciones:

a) Primero cambiaremos la posición del interruptor del botón “Enter”, pasaremos a la posición “Móvil”, figura 2.18.

Figura 2.18. Cambio de posición del interruptor del botón Enter.

b) Con el interruptor en esta posición manipularemos el móvil con los botones “Enter” y “Reset”.
c) Para que el móvil avance presionaremos el botón “Enter” las veces que sean necesarias hasta que llegue a la posición deseada, como se muestra en la figura 2.19.

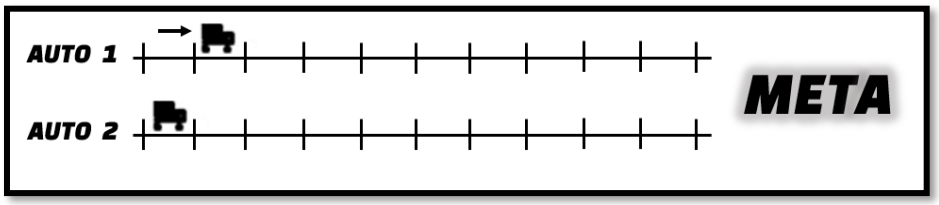

Figura 2.19. Avanza el móvil una posición.

d) Para retroceder el móvil, presionaremos el botón “Reset” las veces que sean necesarias hasta llegar a la posición deseada, como se muestra en la figura 2.20.

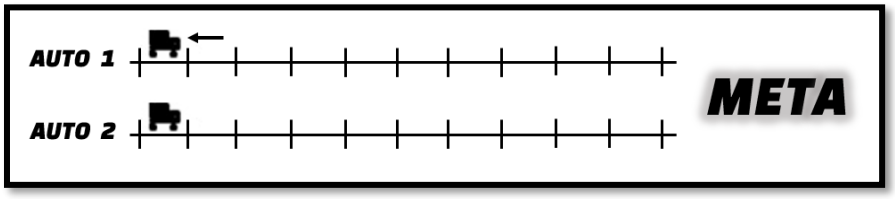

Figura 2.20. Retrocede el móvil una posición.

7- Al final, tal y como podemos ver en la figura 2.21, las fracciones de los participantes quedan mostradas en todos los componentes de visualización y los móviles se han movido de posición.

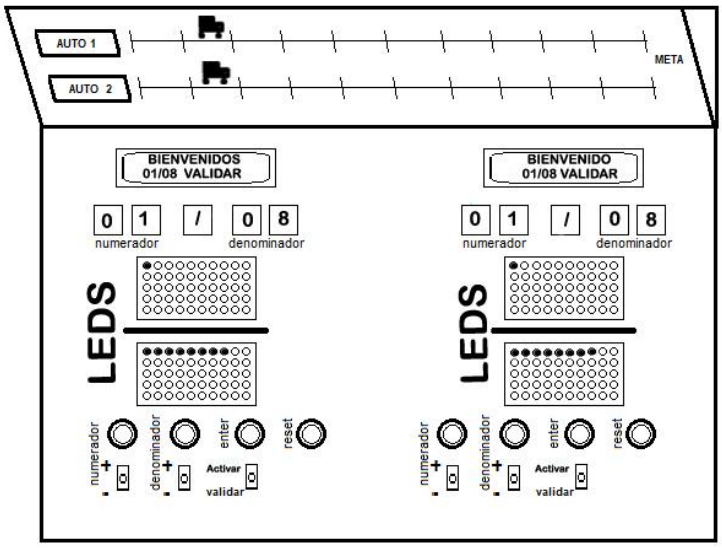

Figura 2.21. Se validaron respuestas y se dieron indicaciones de movilidad para los autos.

¿Qué pasaría si en algún momento el participante quisiera borrar la fracción ingresada?

8.- Para estos casos; sin importar el motivo por el cual se desee borrar la fracción, se ha agregado un botón “Reset”, Figura 2.22.

Este botón tiene la función; en cualquier momento, borrar los datos ingresados en los componentes de visualización y apagarlos, cuando sea presionando.

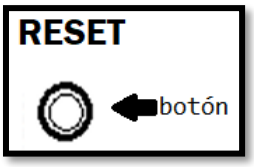

Figura 2.22. Botón Reset

Para reforzar la información consideremos el siguiente ejemplo. La figura 2.23 pide que se anote con número la fracción correspondiente a las partes iluminadas con azul, es decir: 2/8.

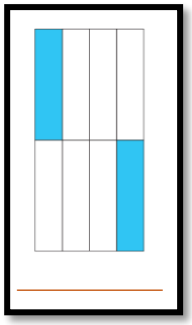

Figura 2.23. ¿A qué fracción corresponde las partes iluminadas de azul?.

1. Antes de ingresar otra fracción a nuestro prototipo, verifiquemos que todos los componentes de visualización estén apagados y sin ninguna fracción mostrada.
2. Si tenemos una fracción visualizada en el tablero, usamos el botón Reset para borrar la información y apagar los componentes.
3. Para esto, si el interruptor del botón “Enter” está en posición “móvil”, es necesario cambiarlo a la posición “fracción”, figura 2.24.

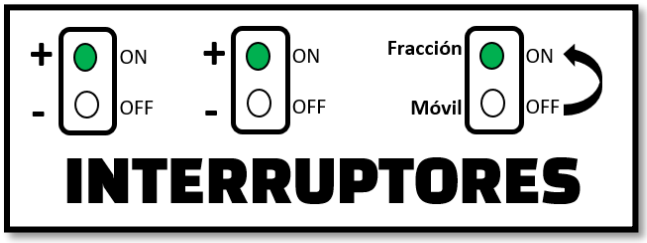

Figura 2.24. Cambiar de posición el interruptor del botón Enter, móvil a fracción.

4. Después de comprobar que los 3 interruptores están en nivel alto (HIGH), presionamos el botón “Reset” para borrar los datos de los componentes de visualización y apagarlos, tal como se muestra en la figura 2.25.

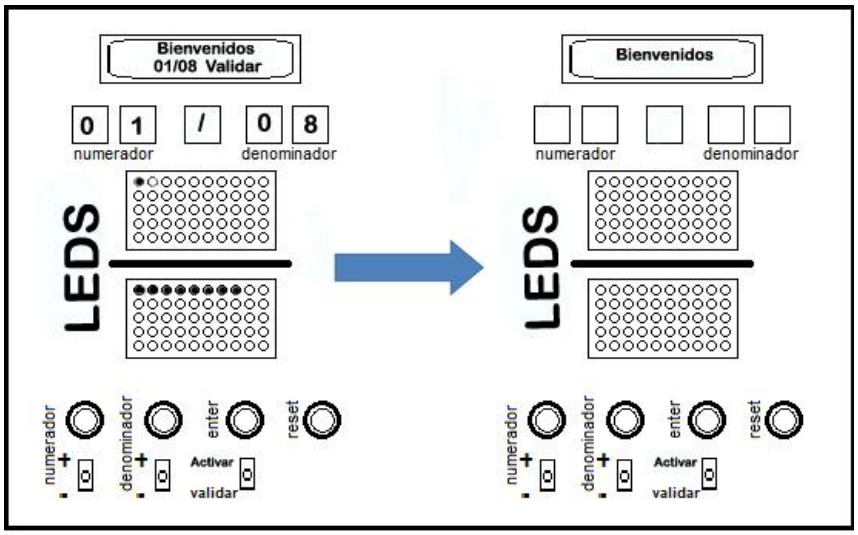

Figura 2.25. Presionar “Reset” para borrar los datos ingresados.

5. Mostrando ambos tableros en la figura 2.26, activamos las primeras dos matrices 8x8 MAX7219 correspondientes al numerador, presionando el botón “Enter”.

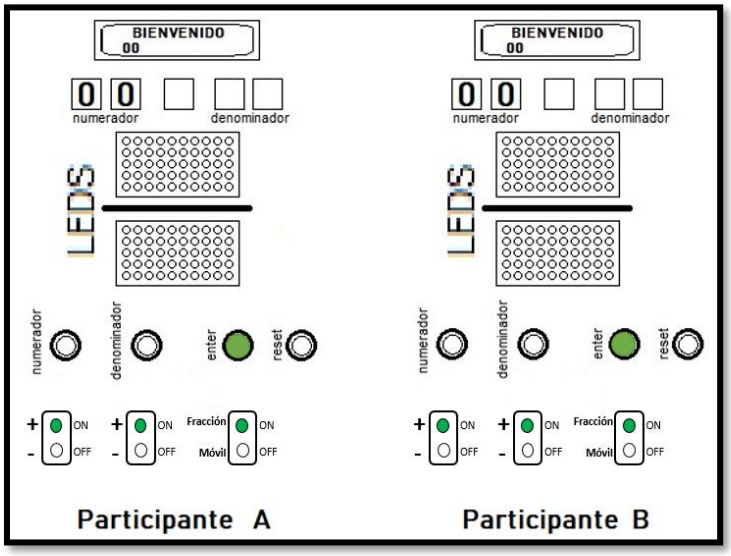

Figura 2.26. Módulos de matrices y leds activados para la parte del numerador de ambos participantes.

6. Presionamos el botón “Numerador” las veces necesarias hasta mostrar el número deseado; figura 2.27, en este caso el número 2 (dos).

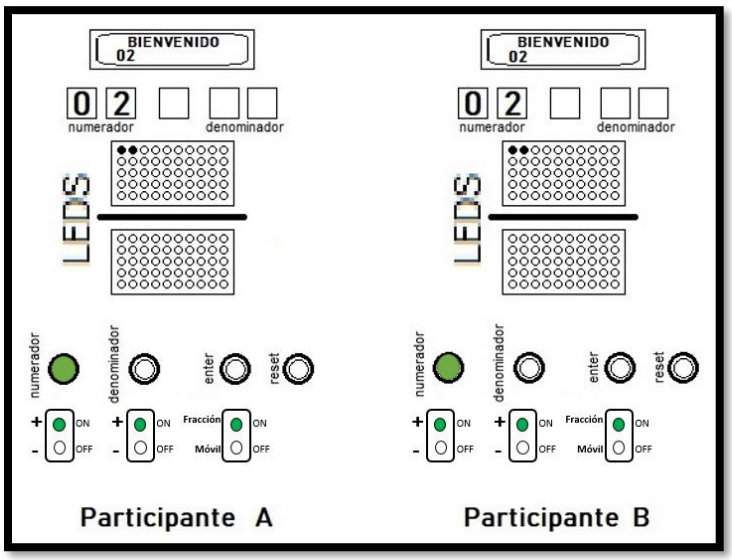

Figura 2.27. Los participantes ingresan el valor del numerador.

7. Presionamos nuevamente el botón “Enter” para mostrar el signo “diagonal” en la tercera matriz 8x8 MAX7219.
8. Activamos las dos últimas matrices 8x8 MAX7219 que corresponde al denominador pulsando “Enter” una vez más, tal como se muestra en la figura 2.28.

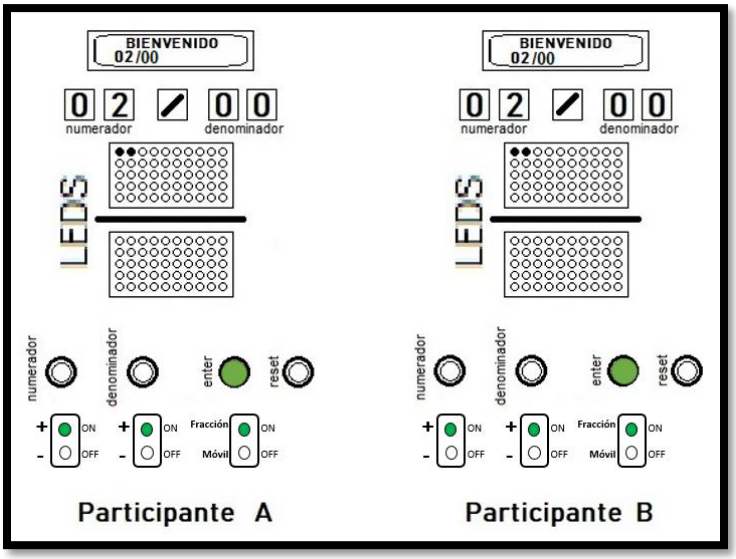

Figura 2.28. activamos la sección del denominador.

9. Presionamos el botón “denominador” las veces necesarias hasta llegar al número deseado, en este ejemplo, un participante tendrá la respuesta distinta al otro, figura 2.29.

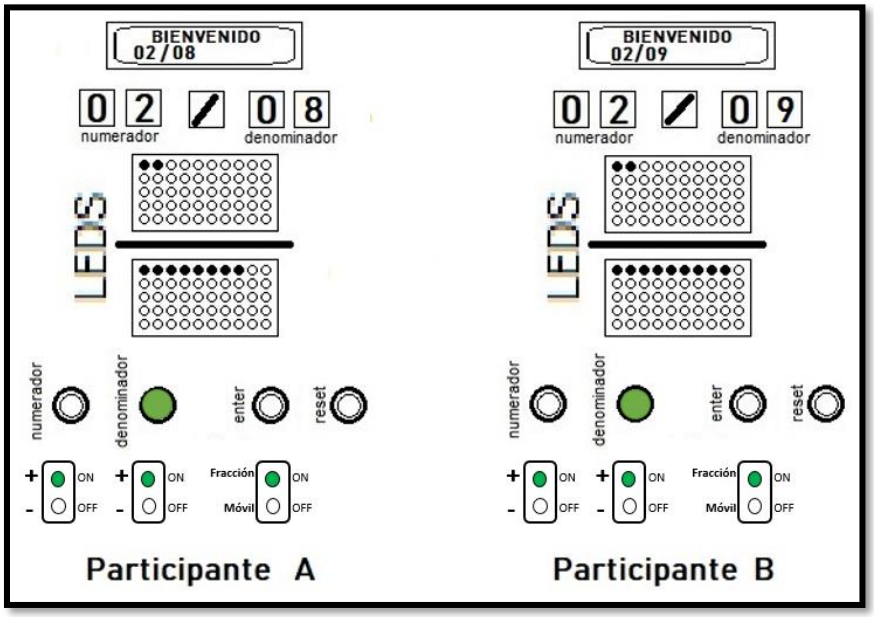

Figura 2.29. Los participantes ingresan el valor del denominador.

10. Presionamos el botón “Enter” para validar la respuesta. El profesor verifica si la respuesta de cada participante es correcta, figura 2.30.

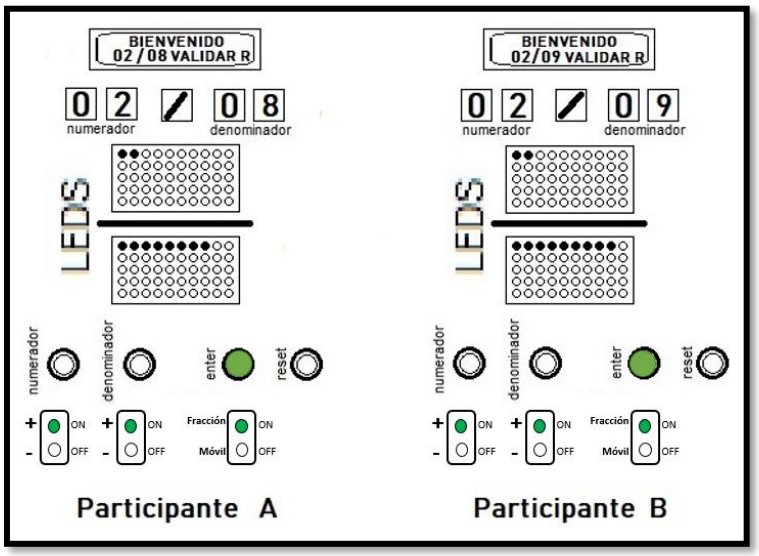

Figura 2.30. Validar la respuesta ingresada.

11. Después de validar las respuestas, el profesor menciona si los dos participantes tienen la respuesta correcta o alguno se equivocó.
12. En este ejemplo, el participante B ingreso la respuesta incorrecta, por lo que solo el participante A avanzara su móvil una posición adelante mientras que el participante B queda en la misma posición.
13. Para mover el móvil, el participante A, cambiara de posición el interruptor del botón “Enter”. De la posición “Fracción” a la posición “móvil”.
14. Presionar el botón “Enter” las veces necesarias para que el móvil avance y llegue a la posición deseada, figura 2.31.

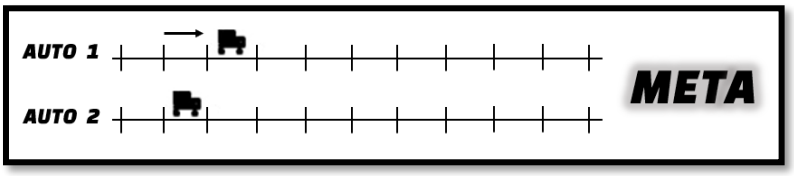

Figura 2.31. El Auto 1 avanza una posición y el auto 2 queda en el mismo lugar.

15. Tal como se muestra en la figura 2.32, vemos el resultado que cada participante ingreso en su tablero correspondiente.

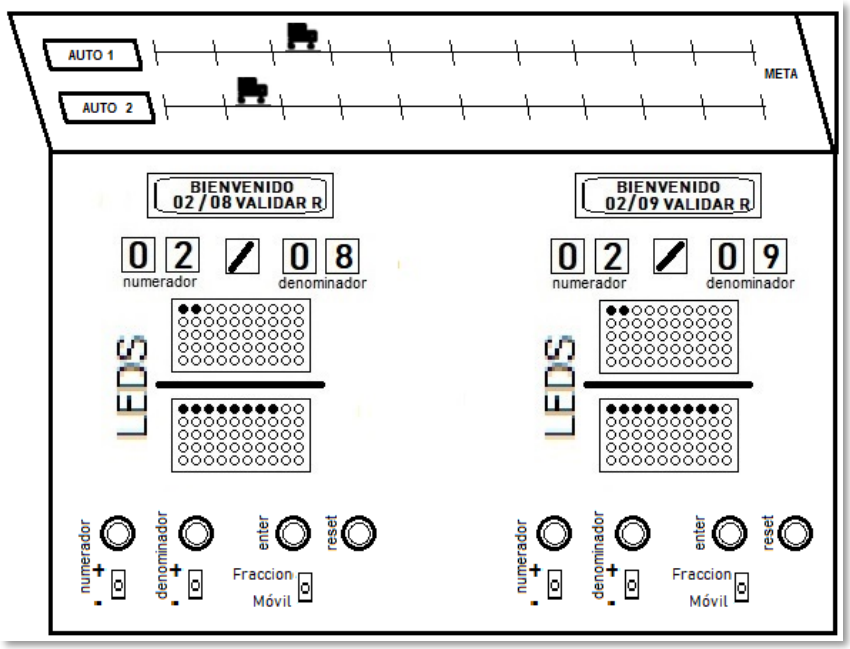

Figura 2.32. Resultado final.

2.3. Conexión interna del prototipo.

Hasta el momento hemos mencionado el diseño exterior y descrito a detalle con un par de ejemplos la función de cada elemento, pero aún hay algo que falta por explicar, básicamente lo más importante, y esto es la parte interna del prototipo, las conexiones.

En el interior de la caja se encuentra todas las conexiones de las tarjetas y de más dispositivos, por lo que ahora toca describir cómo trabajan las tarjetas Arduino y CPLD’s en conjunto con los leds, matrices y botones.

Comenzaremos con las placas de 50 leds, para este prototipo ocupamos 4 placas con 50 leds cada una, dos para cada participante y así formar la fracción. Cada placa está conectada a una tarjeta CPLD MAX II EPM240, elegimos este tipo de tarjeta debido a que cuenta con 100 pines de entrada/salida/clock, suficientes para conectar 50 leds, un interruptor, un botón de reset y una señal de reloj que vendrá del Arduino, esto con el fin de que las fracciones se muestren de manera simultánea con forme se vayan presionando los botones, figura 2.33.

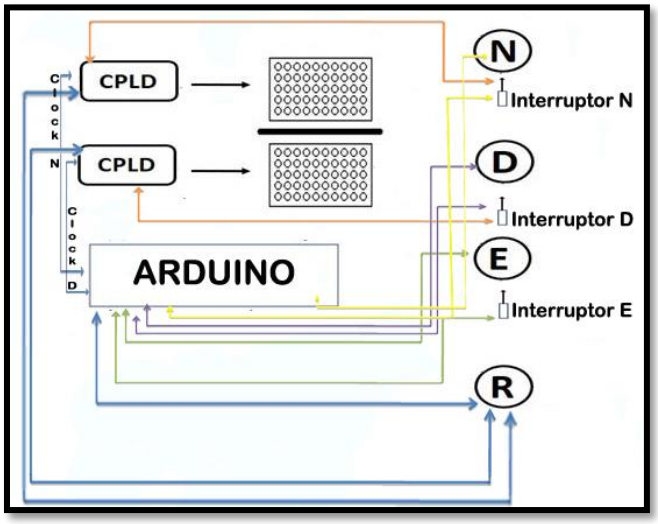

Figura 2.33. Conexión de matrices de leds y CPLD.

Después tenemos en total 10 display's matriz 8x8 para todo el prototipo, 5 por participante para formar la fracción y 2 pantallas LCD. Las matrices 8x8y las pantallas LCD están conectadas directamente al Arduino Mega como se observa en la figura 2.34, los cuales serán manipulados por los botones; básicamente el Arduino es quien controla todo el prototipo.

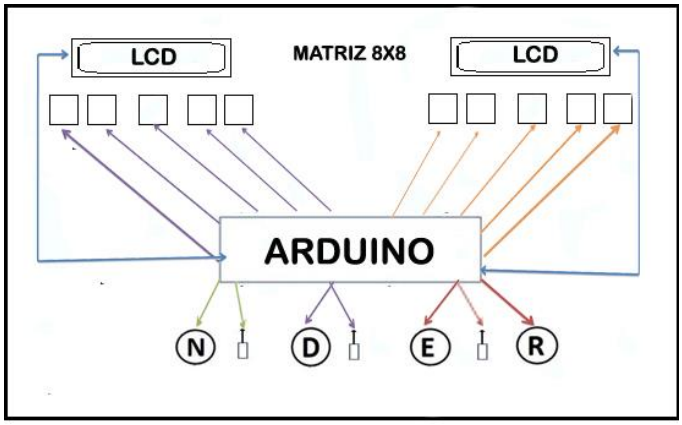

Figura 2.34. Conexión de pantalla LCD y display's matriz 8x8 con Arduino Mega.

Por último, tenemos los dos carriles para los móviles, cada uno está controlado por un motor reductor. Estos móviles avanzarán o retrocederán con ayuda del botón Enter, el interruptor de este botón tendrá que estar en la posición de “Avanzar” para poder mover el móvil un lugar hacia adelante conforme el participante conteste correctamente, figura 2.35.

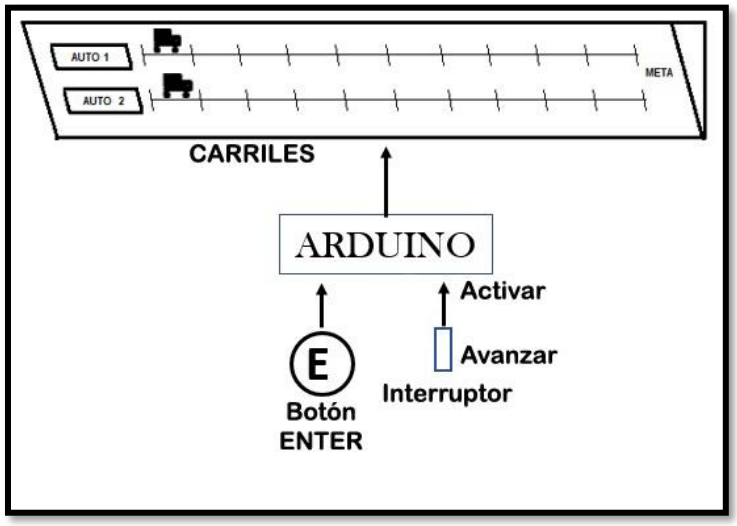

Figura 2.35. Conexión de motores con Arduino Mega.

Habiendo mostrado de manera individual cada conexión entre componentes, tarjetas, motores; finalizaremos con un diagrama eléctrico general de todo el prototipo para tener un enfoque global de la estructura electrónica de nuestro proyecto como se muestra en la figura 2.3

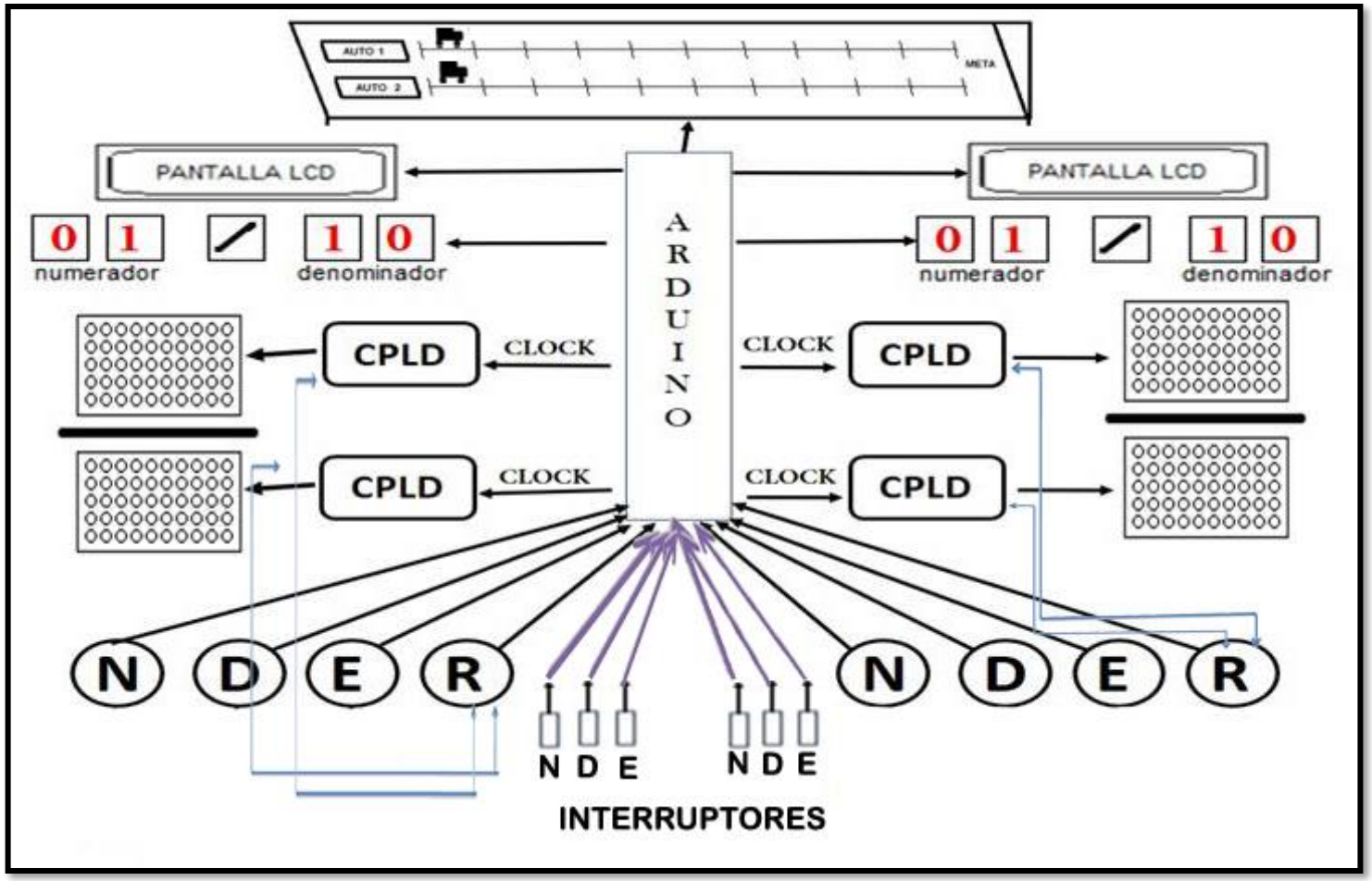

Figura 2.36. Diagrama eléctrico completo del prototipo,

2.4 Materiales.

A continuación, se muestra una lista con todos los materiales que utilizaremos para construir nuestro prototipo:

- 2 Tarjetas Arduino MEGA
- 4 Tarjetas CPLD
- 10 Display 8x8 MAX7219
- 200 Leds
- 8 Botones Arcade
- 6 Interruptores de palanca ON/OFF
- 2 Pantalla LCD
- 2 Interfaz I2C
- 4 Placa PCB perforada para circuitos y prototipos 7x9 cm
- Cable
- 1 Caja de madera pre diseñada
- 2 Motores
- 2 Carros

Capítulo 3

LABORATORIO DE FRACCIONES: MONTAJE

En este capítulo se explica paso a paso como se diseñó y armo el prototipo. Para ello, se diseña en una hoja de papel un dibujo en donde se colocarán todos los elementos electrónicos de control y visualizadores de resultados.

3.1. DISEÑO DEL PROTOTIPO

En la Figura 3.1, se muestra la idea original con las medidas requeridas para montar todos los sistemas electrónicos.

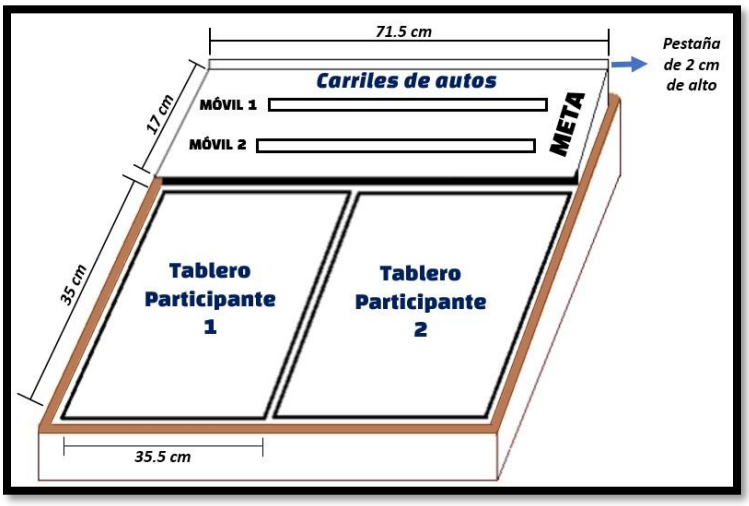

Figura 3.1. Borrador del diseño de estructura del prototipo.

En la figura 3.2 se muestra las medidas en centímetros y los subsistemas que conforman el tablero del participante.

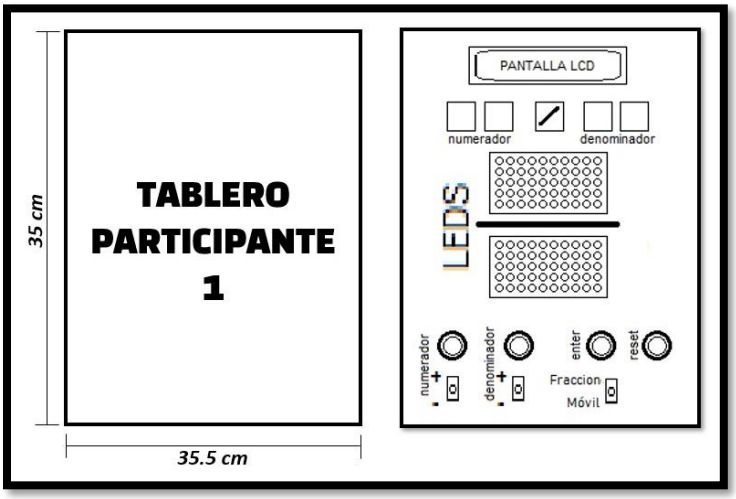

Figura 3.2. Tablero del participante 1.

En base al diseño anterior, se mandó a construir una estructura de madera con las medidas indicadas. Teniendo físicamente la base de madera, podemos comenzar con la construcción del laboratorio.

3.2. Diseño y armado del tablero para el participante.

En la estructura de madera, se colocan todos los componentes de control y de visualización sobre la parte inferior de la base, para distribuirlos dentro de los tableros de los participantes de tal manera que la visualización de la fracción sea más clara y atractiva figura 3.3.

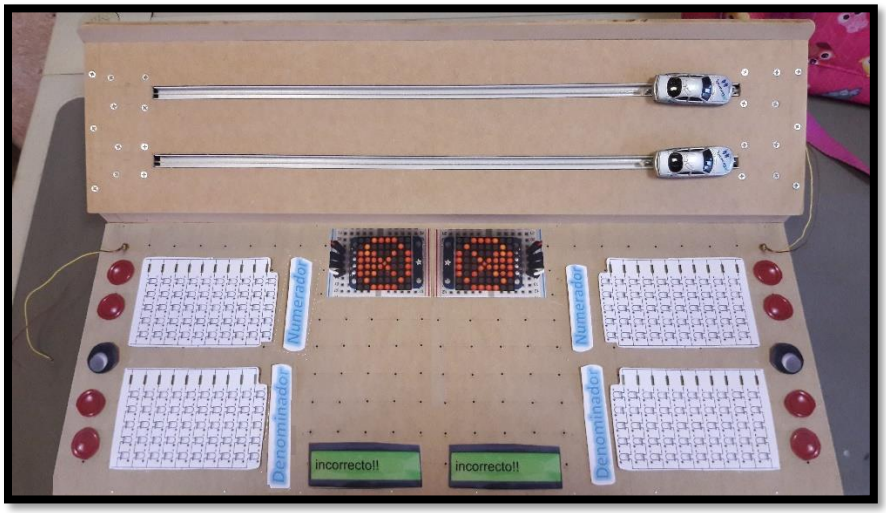

Figura 3.3. Ejemplo de distribución de los componentes en el tablero.

Sobre el tablero, de manera visible, se han colocado, los siguientes dispositivos:

- Dos pantallas display LCD 16x2
- Matrices 8x8 MAX7219
- Cuatro placas de 50 leds 5mm cada una
- Ocho botones arcade
- 2 autos
- 2 carriles para autos

Después de varias combinaciones se deja el panel formado con los componentes como se muestra en la figura 3.4. Se verifica que los espacios entre cada componente fuera el necesario y al momento de hacer las conexiones, la distancia entre cada uno de ellos no interfiriera con el trabajo a realizar.

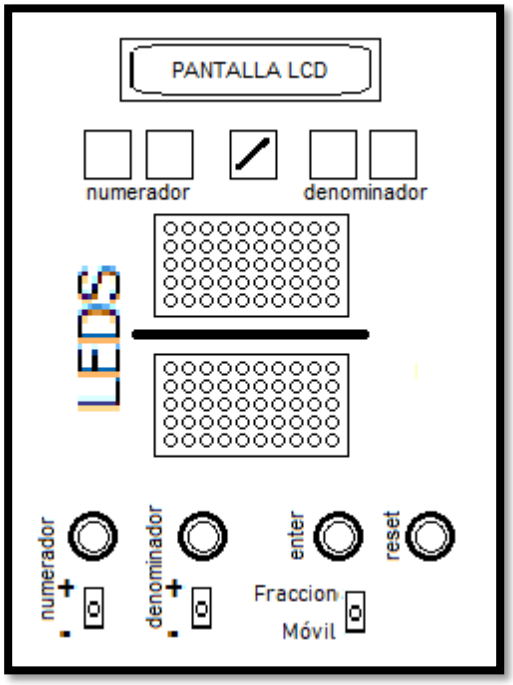

Figura 3.4. Distribución de los componentes para la visualización de fracciones.

Ambos paneles trabajaran de forma separada cada uno, por lo que fue necesario construir primero uno para verificar que cada conexión se hiciera de forma correcta y corregir cualquier falla que se presentara. Después armar el primer panel, haciendo las conexiones de los componentes, programando las tarjetas CPLD y Arduino Mega, y realizando pruebas para verificar que todo funcionara correctamente, procedimos a construir el segundo panel siendo este una copia del primero, puesto que ambos realizarían las mismas actividades, pero de manera independiente.

3.3. Conexión de bloque de leds.

Este procedimiento que describiremos a continuación será el mismo para los cuatro bloques de leds que se ocupa para todo el proyecto.

Para esto se toma una placa PCB perforada y se coloca los leds uno a uno conectado a una resistencia de 330 Ohms. La conexión entre ambos se realizó como se muestra en la Figura 3.5, un lado de la resistencia se

conecta a 5V y el otro a la parte positiva del led dejando la parte negativa del led directo a un pin de la tarjeta CPLD.

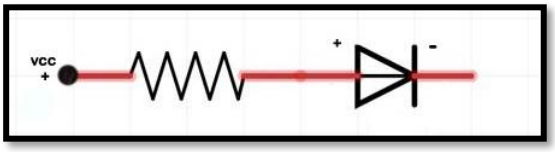

Figura 3.5. Circuito de conexión de led y resistencia.

Se colocan uno por uno formando 5 hileras de 10 leds. Se conectan en serie todos los hilos de las resistencias que van directamente al voltaje dejando como resultado solo una salida la cual estará alimentada por medio del CPLD, figura 3.6.

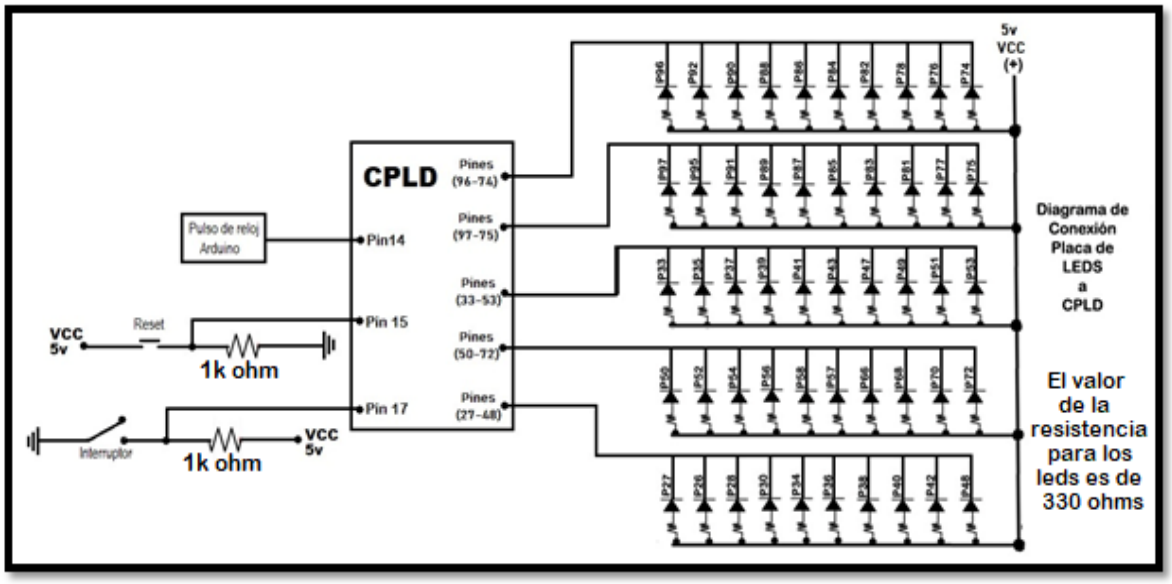

Figura 3.6. Conexión de placa de leds al CPLD con pines asignados.

Mediante un cable, se solda la parte negativa del led a un pin del CPLD y se cubre con un pedazo de termofit que sirve para que las conexiones no hagan contacto entre sí y no causar algún tipo de corto circuito.

Esta acción se repite con cada uno de los leds hasta terminar de conectar los 50 leds a los pines de salida asignados en el CPLD, es importante tener muy bien identificados estos números ya que serán necesarios al momento de programar y configurar el CPLD.

Se trata de cubrir todas las conexiones que se hagan en la tarjeta con el termofit, puesto que los pines de la tarjeta están muy juntos uno del otro, cualquier roce entre ellos podrían causar interferencia con el envío de datos; en este caso, el encendido/apagado del led, Figura 3.7.

Figura 3.7. Conexiones cubiertas con termofit.

Recordando que cada bloque de leds representa una parte de la fracción; es decir, numerador o denominador, por lo que un bloque recibirá el pulso del botón del numerador y otra con el botón del denominador, mediante la tarjeta Arduino Mega. Una vez que estén conectados los 50 leds a la tarjeta CPLD, se sigue con la conexión de los botones y el interruptor.

Para cada bloque, se asignan 3 pines del CPLD que servirán para conectar el botón del reset, el pulso del botón numerador/denominador que recibirá del Arduino Mega y el interruptor correspondiente al numerador/denominador. Estos pines están asignados de la siguiente manera:

- ✓ Pin14: Clock (pulso que recibe del Arduino Mega mediante el botón numerador/denominador).
- ✓ Pin15: Botón Reset.
- ✓ Pin17: Interruptor del numerador/denominador.

Se toma el espacio libre de una de las placas PCB para realizar las conexiones de los botones. Se coloca en una orilla de la placa, dos líneas para la alimentación de los componentes que a su vez estarán conectadas a los pines VCC y GND del CPLD. Después se realizan las conexiones de todos los 4 botones y 3 interruptores que se ocuparan en el tablero del participante, basándose en el circuito de la figura 3.8.

El botón numerador estará conectado al Pin 4 (entrada) del Arduino Mega para que este mismo envíe un pulso al CPLD por medio del Pin 3 (salida), este pulso lo recibe el CPLD en el Pin 14. El botón reset va conectado directamente al pin 15 del CPLD y el interruptor del numerador se conecta al pin 17 del CPLD.

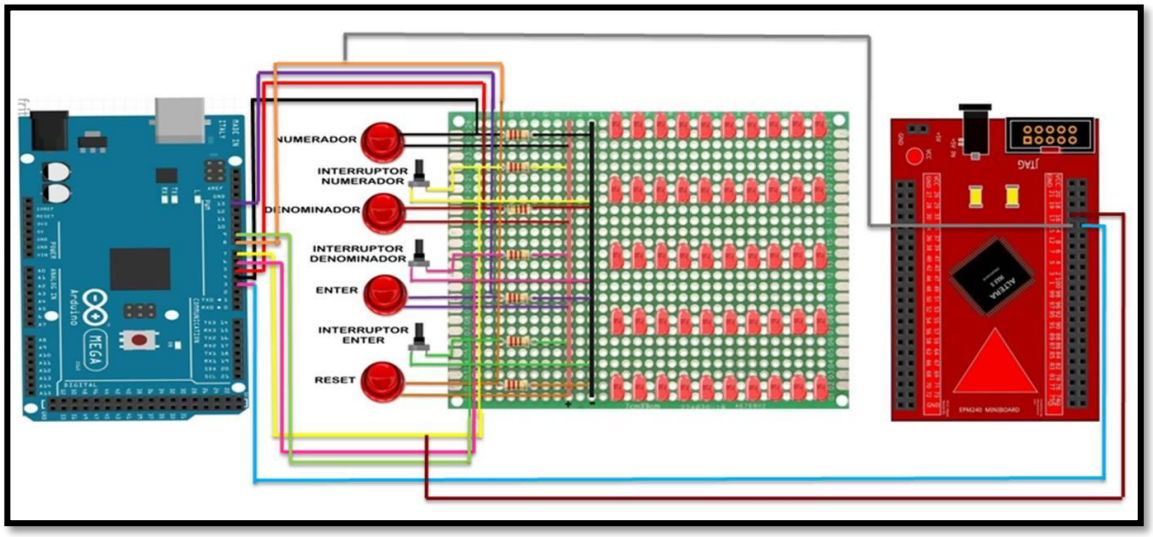

Figura 3.8. Diagrama de conexiones del bloque de leds(numerador) al CPLD.

Hasta este punto solo se conecta al CPLD, el botón de reset, el interruptor y el pulso del botón numerador que viene de la tarjeta Arduino Mega, ya que se aborde el tema de la programación del Arduino se detallará más a fondo la manera en que se envía la señal al CPLD, figura 3.9.

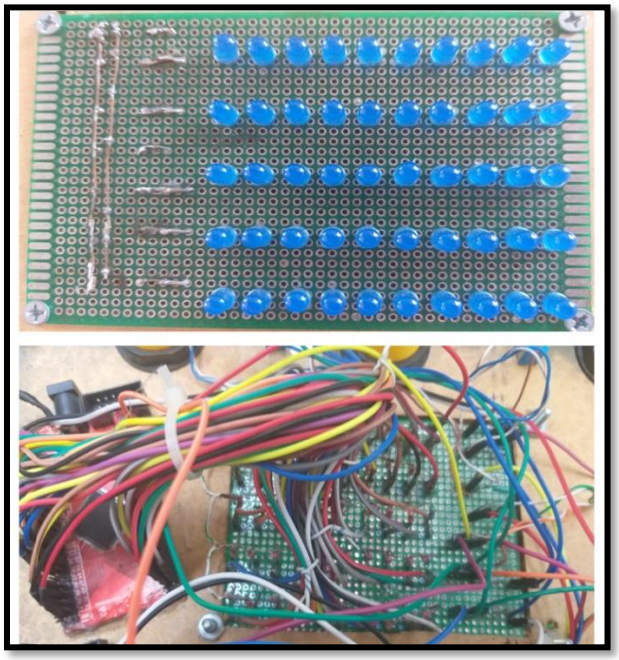

Figura 3.9. CPLD y Placa de leds.

3.4. Conexión de matrices 8x8.

Una matriz LED de 8x8 consiste en un arreglo de diodos. La matriz que se utiliza trabaja en conjunto con un circuito integrado de interface en serie MAX7219, con el cual se controla las 16 terminales de la matriz y al final solo se tienen 3 terminales que controlen todos los leds. Estas terminales son Vcc, GND, DIN, CS y CLK, figura 3.10.

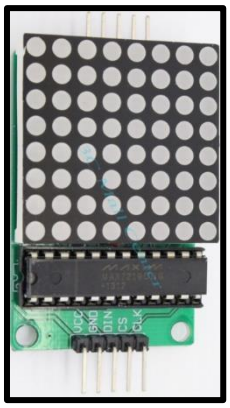

Figura 3.10. Matriz leds 8x8 MAX7219.

Conexión. Los displays MAX7219 tienen rotulado en la parte inferior la función de cada pin esa parte es la conexión inicial de entrada y en el otro extremo se tienen las salidas pin a pin para poder realizar una conexión en cascada, figura 3.11.

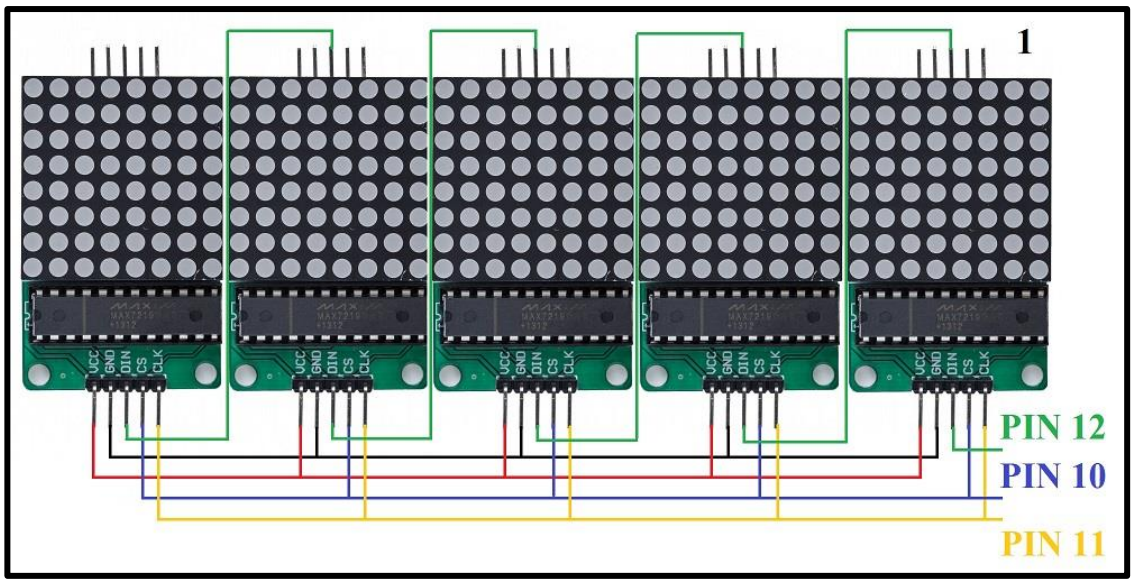

Figura 3.11. Conexión tipo cascada de los displays 8x8 MAX7219.

Después se conectan las terminales DIN, CLOCK y DS a los pines 12, 11 y 10 del Arduino Mega respectivamente, así como también las terminales VCC y GND. De esta manera todos los displays trabajaran de manera simultánea, figura 3.12.

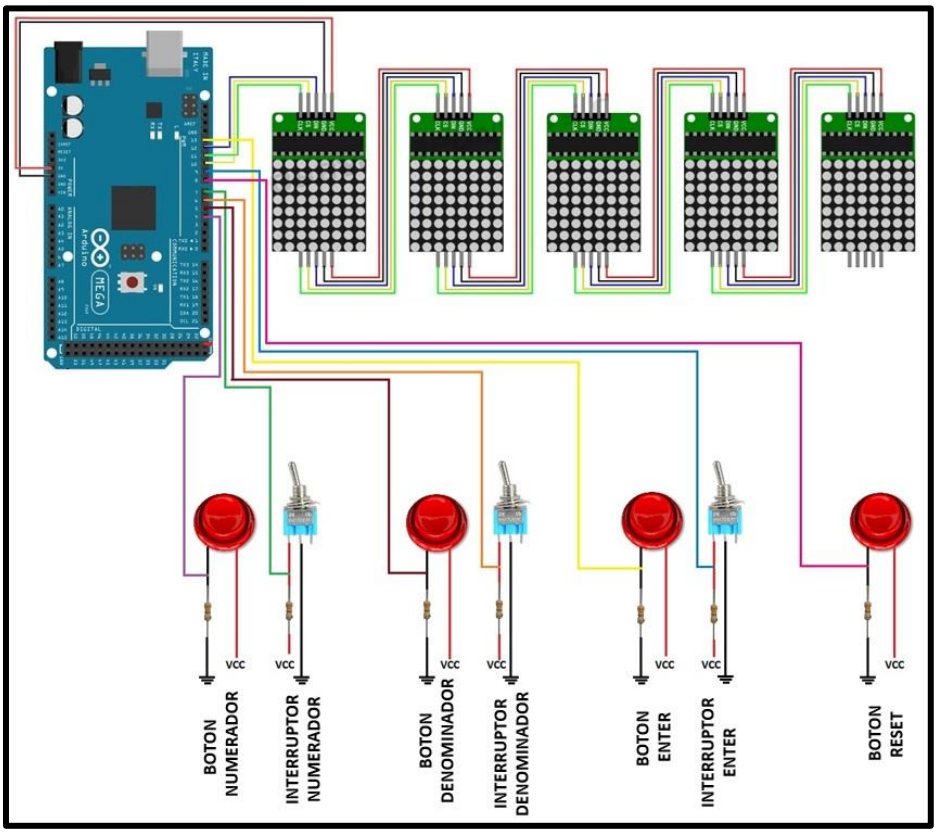

Figura 3.12. Matrices de leds 8x8 MAX7219 conectadas a Arduino Mega.

3.5. Conexión de Pantalla LCD.

Para el tablero del participante se utiliza una pantalla LCD 16x2 para mostrar mensaje de texto de bienvenida, así como también la fracción ingresada. Trabajar con este tipo de pantalla es un poco complicado por el número de líneas de conexión que se necesitan por lo que se añade a esta pantalla una interfaz I2C que se soldará en la parte trasera de la pantalla, figura 3.13.

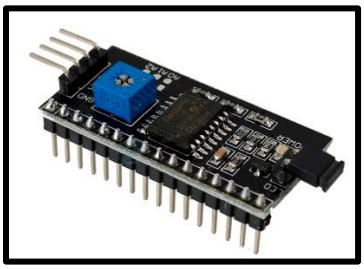

Figura 3.13. Interfaz I2C

Conexión. En seguida se muestra cómo se conectan estos dos componentes entre sí. Esto permitirá conectar la pantalla a la tarjeta Arduino Mega usando solamente dos líneas digitales.

Para usar la pantalla LCD 16×2 por I2C con Arduino el primer paso es soldar el adaptador I2C en la parte de trasera de la pantalla. Al finalizar la soldadura, el adaptador debe verse como en la figura 3.14

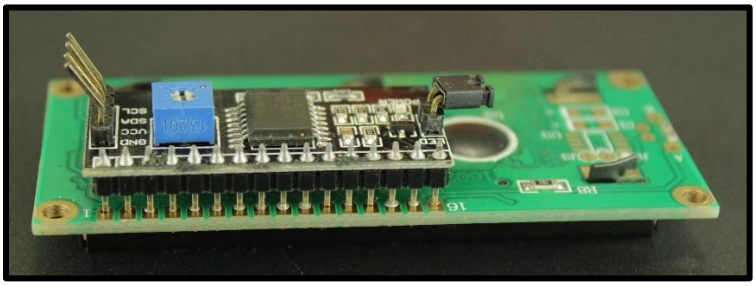

Figura 3.14. Líneas de conexión soldadas de la pantalla LCD con I2C.

Como se puede ver en la imagen, este adaptador ayuda a reducir el tiempo en las conexiones puesto que ya incluye el potenciómetro para regular el contraste de la pantalla, así como también todo lo necesario para el funcionamiento del backlight (retroiluminación), pudiendo incluso controlar esta función a través de software.

Una vez soldada la pantalla y el adaptador, se identifican los pines del I2C en la tarjeta Arduino Mega. Estos dos pines son SDA y SCL que en la tarjeta Arduino Mega se encuentra con los números 20 y 21. Los dos pines restantes del I2C son el VCC y GND respectivamente, al final la conexión queda como se muestra en la figura 3.15.

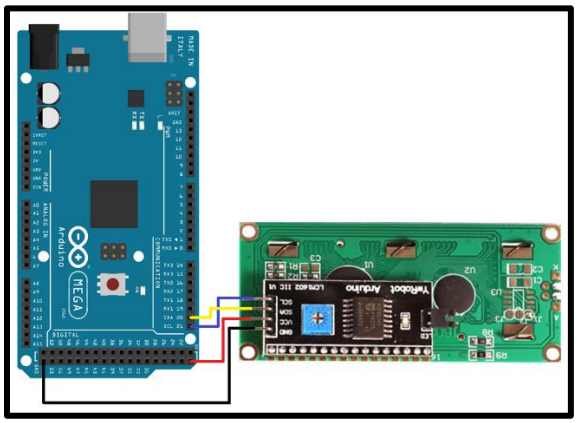

Figura 3.15. Diagrama de conexión Arduino mega y pantalla LCD con I2C.

3.6. Construcción de tablero para participante.

Teniendo las conexiones y la programación de la pantalla LCD, las matrices 8x8 MAX7219 y los bloques de leds con tarjeta CPLD se procede a continuar con el montaje de todos estos componentes en una tabla de MDF de 6mm. Se corta un cuadro de 35 x 35.5 cm, se coloca la pantalla LCD y las matrices 8x8 encima de la tabla para marcar la posición en la que se colocaran cada una de ellas. Se distribuyen los displays de manera que se visualicen como una fracción y que la distancia entre ellos no sea un conflicto al momento de hacer las conexiones. Teniendo marcado los cuadros de los displays, se corta y coloca uno a uno cada componente, checando que queden alineados y bien sujetos con tornillos, tal como se muestra en la figura 3.16.

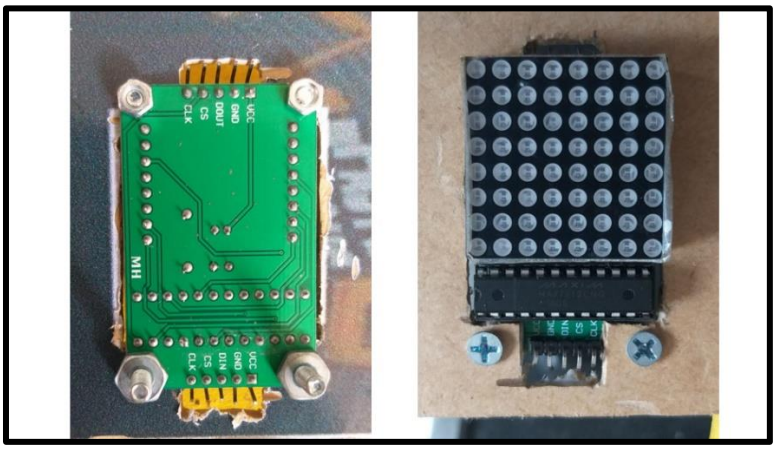

Figura 3.16. Montaje de matrices 8x8 MAX7219.

De esta manera se continua con la colocación de la pantalla LCD que estará situada en parte superior del tablero, como se muestra en la figura 3.17.

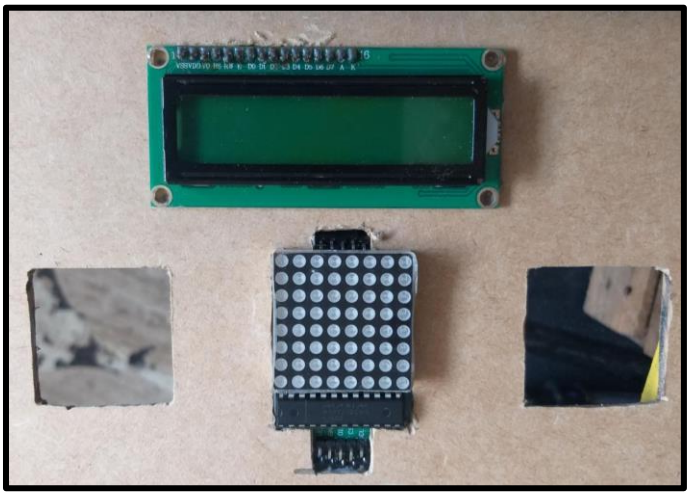

Figura 3.17. Montaje de pantalla LCD.

Se sigue con los bloques de leds, se mide la placa PCB en la tabla centrado a la mitad el cuadro de leds, de tal manera que queden a la altura de la pantalla LCD y la tercera matriz 8x8, esto puede verse en la figura 3.18.

Figura 3.18. Montaje de bloque de leds.

Así como se coloca el primer bloque de leds, se coloca el segundo bloque abajo del primero, con estos dos bloques se pretende visualizar la fracción; numerador y denominador, respectivamente. En la figura 3.19, se puede ver que ambos bloques quedan alineados al mismo nivel que la pantalla LCD y la matriz 8x8. De esta manera se tiene 3 formas distintas de visualizar la fracción que el participante ingrese.

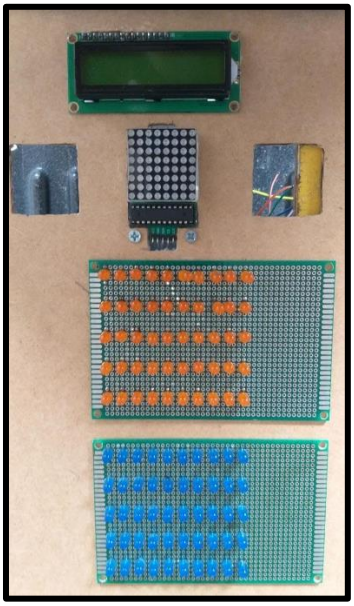

Figura 3.19. Visualización de fracción formada con bloques de leds.

Habiendo montado todos los dispositivos en la tabla, se coloca en la parte inferior los botones e interruptores que se van a utilizar para la manipulación de los componentes. Finalmente, el tablero se vería como se muestra en la figura 3.20, por lo que se procede a realizar las conexiones finales.

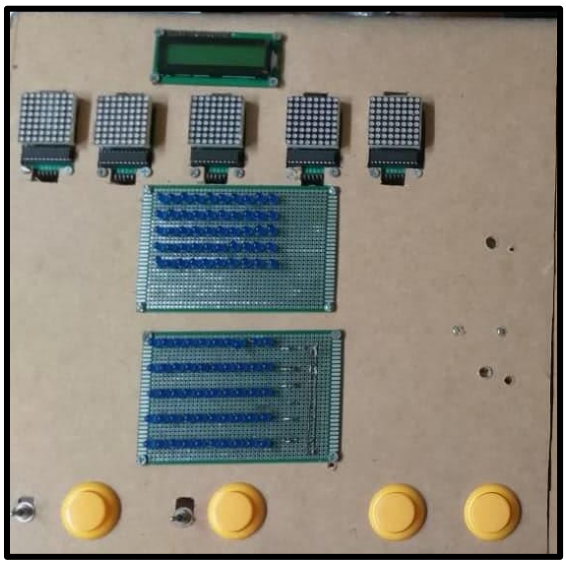

Figura 3.20. Vista externa del tablero.

Ahora se observa en la parte interna del prototipo, en la cual se realizan las conexiones entre los dispositivos y la tarjeta Arduino Mega. Puesto que cada tablero será manejado por una tarjeta independiente, el siguiente paso es colocar esta tarjeta en la parte trasera de la tabla sujeta con tornillos, cerciorándose de que no se mueva y no cree conflicto a la hora de conectar los demás componentes, figura 3.21.

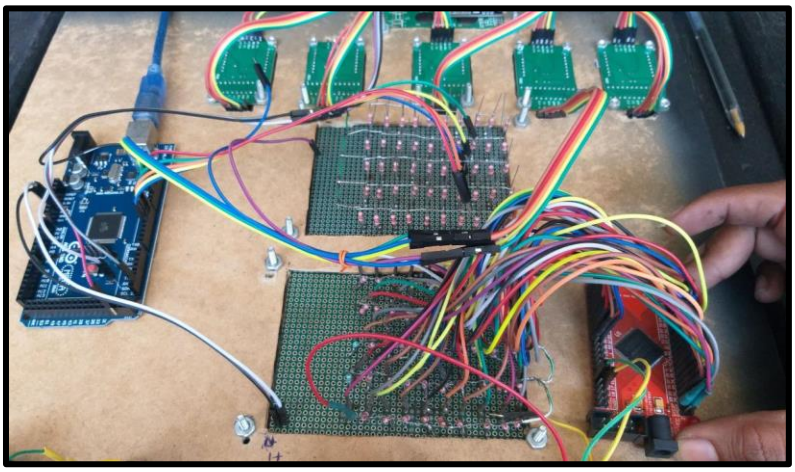

Figura 3.21. Montaje de tarjeta Arduino.

Se sujetan las tarjetas CPLD a la tabla con ayuda del silicón frio, se coloca una pequeña cantidad en cada orilla de la tarjeta y se sujetan hasta que queden fijas y no se muevan.

Se realizan las conexiones que se han detallado anteriormente de cada grupo de componente en la tarjeta Arduino Mega. Se verifica que cada pin de entrada y salida de la Tarjeta Arduino esté conectado de manera correcta tal y como están declarados en el programa.

Se procura que la mayoría de las conexiones que se realicen estén soldadas y cubiertas por un protector de plástico (termofit), para que no cause conflicto algún movimiento que se realice en el manejo del prototipo, un ejemplo de esto se muestra en la figura 3.22.

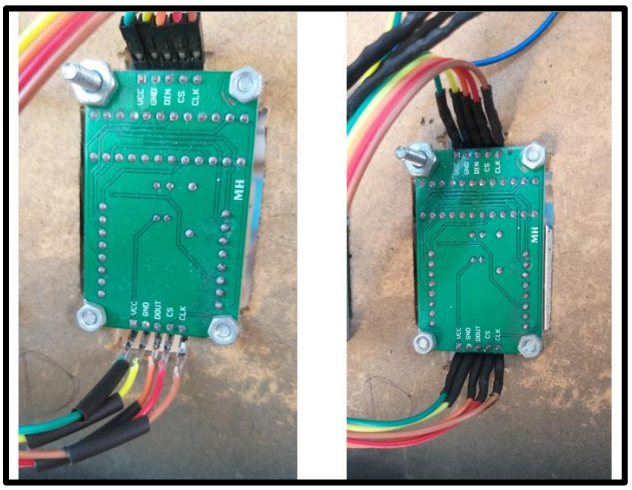

Figura 3.22. Conexiones soldadas y cubiertas con termofit.

Después de conectar la pantalla LCD, matrices 8x8 MAX7219, bloques de leds con CPLD y botones e interruptores en la tarjeta Arduino Mega, checamos que los cables estén bien sujetos y se agrupa sujetándose con cinchos. Se revisa cuidadosamente que todo esté conectado como esta descrito en los diagramas, figura 3.23.

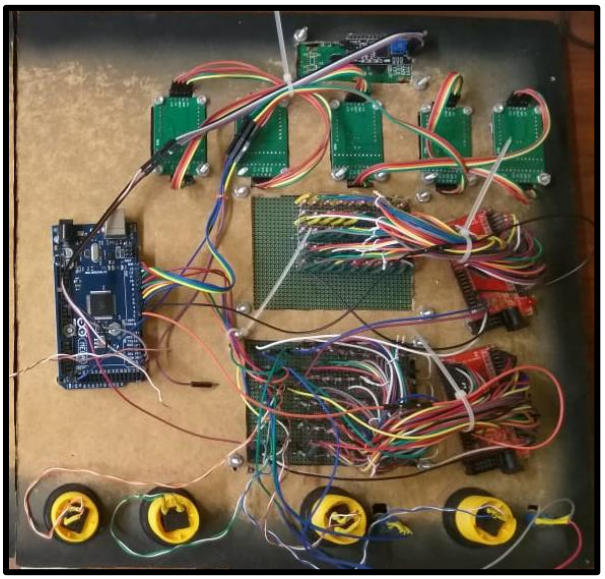

Figura 3.23. Conexiones del tablero.

Procedemos a realizar pruebas con el tablero, se ingresa fracciones y se corrobora que el ciclo se cumpla conforme lo programado. Se comienza pruebas presionando el botón enter para activar las matrices del

numerador y se ingresa la primera cifra con el botón numerador, figura 3.24.

Figura 3.24. Ingresar el numerador.

Se presiona de nuevo el botón enter y activamos la matriz del signo diagonal, presionamos una vez más el botón enter para activar las matrices del denominador, figura 3.25.

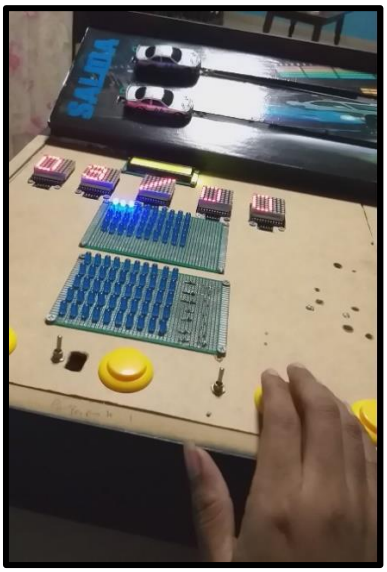

Figura 3.25. Activar la matriz del signo diagonal y las matrices del denominador

Se ingresa la cifra que corresponda al denominador y se verifica que la fracción que ingresa se muestra en todos los dispositivos y de las 3 distintas formas que se emplea, obteniendo un resultado como se muestra en la figura 3.26.

Figura 3.26. Visualización de las fracciones.

Después de realizar varias pruebas y verificado que el tablero funciona correctamente. Se procede a realizar el segundo tablero para el participante 2. Tomando en cuenta que ambos tableros son básicamente lo mismo en todo sentido; materiales y programación, se realiza los mismos procedimientos que ya se han descrito al comienzo de este capítulo. Ya que se tienen los dos tableros listos, solo nos un último paso y no por ello menos importante, que es el movimiento de los móviles.

3.7. Conexión de los motores.

Teniendo listo los dos tableros, se colocan encima de la base de madera y se sujetan con pequeños clavos para que no se muevan. El siguiente paso es conectar los motores a las tarjetas Arduino Mega. Cada motor estará conectado al tablero que le corresponda según el número de móvil, sea 1 o 2.

Conexión. Para la conexión de estos motorreductores se utiliza un controlador de motores L298N. El L298N es un controlador de motores, que permite encender y controlar dos motores de corriente continua desde Arduino, variando tanto la dirección como la velocidad de giro. La corriente máxima que el L298N puede suministrar a los motores es, en teoría, 2A por salida y una tensión de alimentación de 3V a 35V.

En la figura 3.27, se ve como está compuesto este controlador:

- **Motor A** y **Motor B** son las salidas para los motores, en estas estarán conectados los dos motorreductores respectivamente.

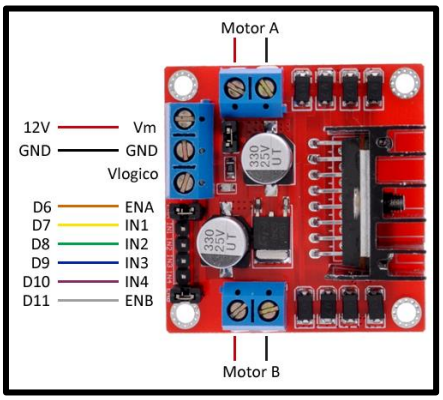

Figura 3.27. Controlador de motor L298D.

- 4 entradas (**IN1**, **IN2**, **IN3**, **IN4**) que regulan la velocidad y el sentido del giro, 2 para cada motor, es decir, **IN1** e **IN2** corresponden al **Motor A** y para el **Motor B** tenemos **IN3** e **IN4**.
- **ENA** y **ENB** permite habilitar/deshabilitar los motores.
- 3 terminales de alimentación (12v, GND y 5v)

Básicamente esto es lo que interesa manejar para el proyecto, por lo que no se entrará a detalle en cómo está formado el controlador.

En la figura 3.27, se muestra el diagrama de conexión del controlador L298N con los motorreductores a las tarjetas Arduino Mega.

Los pines de entrada del controlador estarán distribuidos y conectados de la siguiente manera:

- In1 pin 24 (tarjeta Arduino Mega Tablero 1)
- In2 pin 22 (tarjeta Arduino Mega Tablero 1)
- In3 pin 22 (tarjeta Arduino Mega Tablero 2)
- In4 pin 24 (tarjeta Arduino Mega Tablero 2)

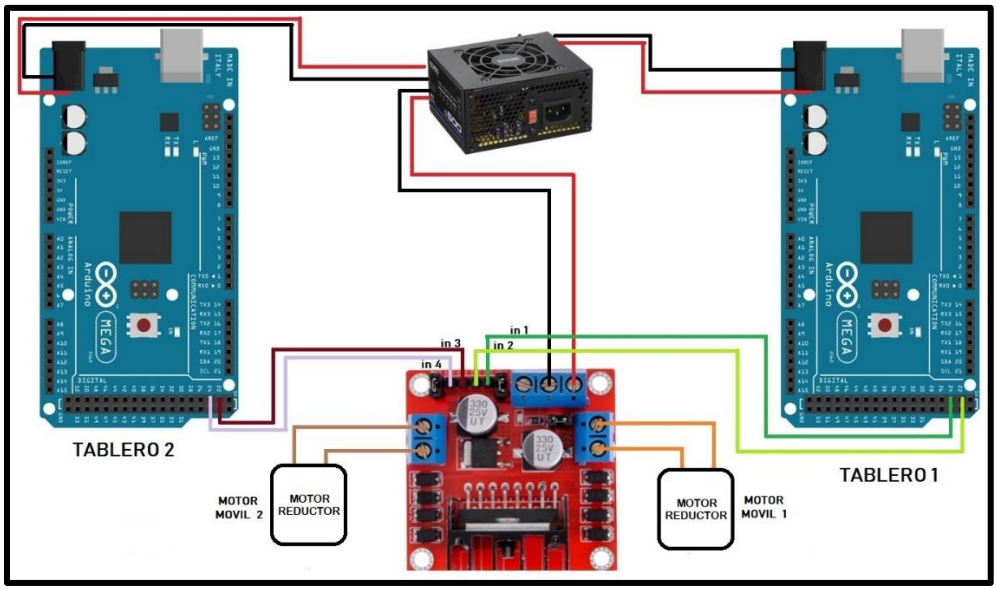

Figura 3.28. Diagrama de conexión de L298N.

Como se ha mencionado anteriormente, ambos tableros trabajan de manera independiente, es por eso que en el diagrama anterior se muestran ambas tarjetas Arduino, ya que el controlador L298N sirve para conectar 2 motores.

Para alimentar todo el prototipo se usa una fuente de poder para PC, la cual cuenta con varias salidas de diversos valores de voltaje. Para la alimentación de las tarjetas Arduino Mega y el controlador de motor, se

utilizan salidas de 12v y para las tarjetas CPLD se utilizan salidas de 5v. En la figura 3.28, se tiene el controlador L298N conectado a una terminal de cable amarillo y negro, el cual; según la configuración de la fuente, el cable amarillo da un voltaje de 12v y el cable negro viene siendo la tierra (gnd). Ambos se conectan a las terminales de alimentación del controlador, en el lugar que les corresponda, puesto que este controlador se puede alimentar con 5 o 12v respectivamente.

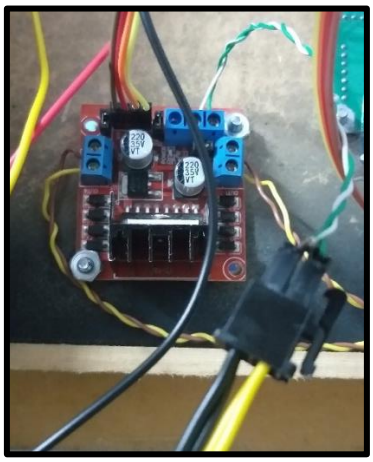

Figura 3.29. Controlador L298N.

Capítulo 4

LABORATORIO DE FRACCIONES: PROGRAMACIÓN

4.1 Programación de bloque de leds.

A continuación, se explicará la estructura del programa en VHDL “Lenguaje de descripción en hardware” que se realizó para el encendido/apagado de leds y configuración de pines en el CPLD “Dispositivo lógico programable complejo”. Para lo cual se ocupa trabajar con contadores y decodificadores, que ayudará a enviar la señal de encendido y apagado de cada led.

Las primeras 4 líneas muestran las librerías que se utilizan para este programa, de la línea 6 a la 11 se declara la entidad “PlacaLeds” dentro de la cual se declaran los pins de entrada reset, clkl, interruptor y la salida F, respectivamente, figura 4.1.

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3   use ieee.std_logic_arith.all;
4   use ieee.std_logic_unsigned.all;
5
6   entity PlacaLeds is
7   port (reset:in std_logic;
8         clkl:in std_logic;
9   interruptor:in std_logic;
10            F:out std_logic_vector (0 to 49));
11  end PlacaLeds;
12
```

Figura 4.1. Declaración de pins de entrada y salida.

En la línea 13 se comienza la Arquitectura denominada “Cuenta”, donde se declara la señal Q que se manejara como el enlace entre el contador y el decodificador, todo este proceso se realiza de manera interna. Para realizar el conteo de 0 a 50, lo hicimos de manera binaria, esto significa que se ocupa que el vector sea de 6 bits, mostrado en la figura 4.2.

```
13  architecture cuenta of PlacaLeds is
14    signal Q: std_logic_vector (5 downto 0);
15  begin
16
```

Figura 4.2. Inicio de la estructura del programa.

Se continua con el proceso del contador línea 17 y como elementos de la lista sensitiva se encuentra; (reset, clkl, interruptor), figura 4.3.

En la línea 19 se genera el pulso de reloj activo en la transición de alto a bajo, en la línea 20 se evalúa la condición del interruptor si está en valor de ‘1’ entonces el contador es incrementado una unidad, “Q<=Q+1”. En caso de que esta condición no se cumpla se utiliza la sentencia ELSIF que significa cuando “**sino sí**”; es decir, en la línea 23 se lee como “**sino si,** si el interruptor es igual a uno y Q igual a 50” expresado en binario como “110010” entonces el valor de Q adopta el valor de 0 “000000”.

Como puede observarse esta descripción permite hacer que dependiendo del valor del interruptor el conteo pueda ser ascendente o descendente línea 33. De lo contrario, si ninguna de las condiciones anteriores no se cumple el valor de “Q” disminuirá uno a la vez, mostrado en la línea 36. Se finaliza el proceso indicando una sentencia para el botón reset, en caso de que este botón se encuentre en estado ALTO el valor de “Q” será cero.

```
17  process (reset, clkl, interruptor)
18  begin
19     if (clkl'event and clkl='1') then
20        if (interruptor='1') then
21           Q<=Q+1;
22
23           elsif (interruptor='1' and Q="110010") then
24           Q<="000000";
25
26           elsif (interruptor='1' and Q="000000") then
27           Q<=Q+1;
28
29           elsif (interruptor='0' and Q="000000") then
30           Q<="110010";
31
32           elsif (interruptor='0' and Q="110010") then
33           Q<=Q-1;
34
35           else
36           Q<=Q-1;
37        end if;
38     end if;
39
40     if (reset='1') then
41     Q<="000000";
42     end if;
43  end process;
```

Figura 4.3. Proceso del contador.

En la figura 4.4, se tiene la estructura del decodificador, donde se muestran todas las combinaciones de “Q” que corresponde al conteo del 0 al 50 y el valor de salida de “F” para cada una de ellas, el cual se verá reflejado en el bloque de leds cada vez que el botón numerador/denominador sea presionado, como puede observarse el nuevo proceso inicia en la línea 45 y su variable sensitiva es Q.

```
Decodificador: process (Q)
begin
case Q is
when "000000"=>F<="11111111111111111111111111111111111111111111111111111111";--0
when "000001"=>F<="01111111111111111111111111111111111111111111111111111111";--1
when "000010"=>F<="00111111111111111111111111111111111111111111111111111111";--2
when "000011"=>F<="00011111111111111111111111111111111111111111111111111111";--3
when "000100"=>F<="00001111111111111111111111111111111111111111111111111111";--4
when "000101"=>F<="00000111111111111111111111111111111111111111111111111111";--5
when "000110"=>F<="00000011111111111111111111111111111111111111111111111111";--6
when "000111"=>F<="00000001111111111111111111111111111111111111111111111111";--7
when "001000"=>F<="00000000111111111111111111111111111111111111111111111111";--8
when "001001"=>F<="00000000011111111111111111111111111111111111111111111111";--9
when "001010"=>F<="00000000001111111111111111111111111111111111111111111111";--10
when "001011"=>F<="00000000000111111111111111111111111111111111111111111111";--11
when "001100"=>F<="00000000000011111111111111111111111111111111111111111111";--12
when "001101"=>F<="00000000000001111111111111111111111111111111111111111111";--13
when "001110"=>F<="00000000000000111111111111111111111111111111111111111111";--14
when "001111"=>F<="00000000000000011111111111111111111111111111111111111111";--15
when "010000"=>F<="00000000000000001111111111111111111111111111111111111111";--16
when "010001"=>F<="00000000000000000111111111111111111111111111111111111111";--17
when "010010"=>F<="00000000000000000011111111111111111111111111111111111111";--18
when "010011"=>F<="00000000000000000001111111111111111111111111111111111111";--19
when "010100"=>F<="00000000000000000000111111111111111111111111111111111111";--20
when "010101"=>F<="00000000000000000000011111111111111111111111111111111111";--21
when "010110"=>F<="00000000000000000000001111111111111111111111111111111111";--22
when "010111"=>F<="00000000000000000000000111111111111111111111111111111111";--23
when "011000"=>F<="00000000000000000000000011111111111111111111111111111111";--24
when "011001"=>F<="00000000000000000000000001111111111111111111111111111111";--25
when "011010"=>F<="00000000000000000000000000111111111111111111111111111111";--26
when "011011"=>F<="00000000000000000000000000011111111111111111111111111111";--27
when "011100"=>F<="00000000000000000000000000001111111111111111111111111111";--28
when "011101"=>F<="00000000000000000000000000000111111111111111111111111111";--29
when "011110"=>F<="00000000000000000000000000000011111111111111111111111111";--30
when "011111"=>F<="00000000000000000000000000000001111111111111111111111111";--31
when "100000"=>F<="00000000000000000000000000000000111111111111111111111111";--32
when "100001"=>F<="00000000000000000000000000000000011111111111111111111111";--33
when "100010"=>F<="00000000000000000000000000000000001111111111111111111111";--34
when "100011"=>F<="00000000000000000000000000000000000111111111111111111111";--35
when "100100"=>F<="00000000000000000000000000000000000011111111111111111111";--36
when "100101"=>F<="00000000000000000000000000000000000001111111111111111111";--37
when "100110"=>F<="00000000000000000000000000000000000000111111111111111111";--38
when "100111"=>F<="00000000000000000000000000000000000000011111111111111111";--39
when "101000"=>F<="00000000000000000000000000000000000000001111111111111111";--40
when "101001"=>F<="00000000000000000000000000000000000000000111111111111111";--41
when "101010"=>F<="00000000000000000000000000000000000000000011111111111111";--42
when "101011"=>F<="00000000000000000000000000000000000000000001111111111111";--43
when "101100"=>F<="00000000000000000000000000000000000000000000111111111111";--44
when "101101"=>F<="00000000000000000000000000000000000000000000011111111111";--45
when "101110"=>F<="00000000000000000000000000000000000000000000001111111111";--46
when "101111"=>F<="00000000000000000000000000000000000000000000000111111111";--47
when "110000"=>F<="00000000000000000000000000000000000000000000000011111111";--48
when "110001"=>F<="00000000000000000000000000000000000000000000000001111111";--49
when "110010"=>F<="00000000000000000000000000000000000000000000000000111111";--50

when others=>F<="11111111111111111111111111111111111111111111111111111111";--0

end case;
end process Decodificador;
end cuenta;
```

Figura 4.4. Decodificador.

Después de diseñar el programa, se guardan los cambios y se compila el programa mediante el software Quartus II para verificar que no se tenga ningún error, en caso de haberlo se tiene que corregir y volver a compilar hasta que muestre el mensaje de 0 errores como en la figura 4.5.

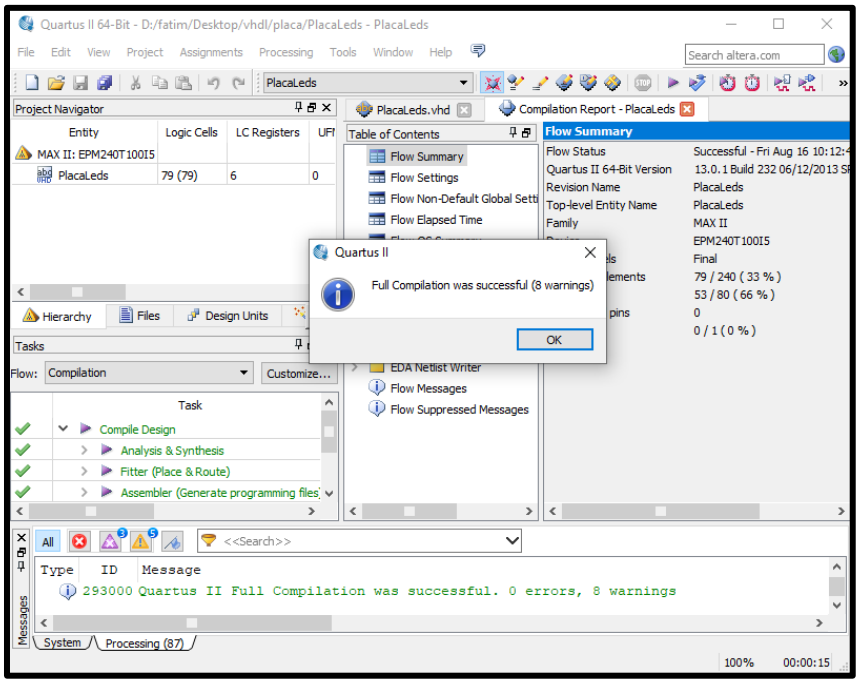

Figura 4.5. Compilación del programa.

Para realizar la asignación de pins debemos seguir con el siguiente procedimiento mostrado en la figura 4.6:

1. Inicialmente seleccionamos la opción de Assignments.
2. Posteriormente la selección PinPlanner.

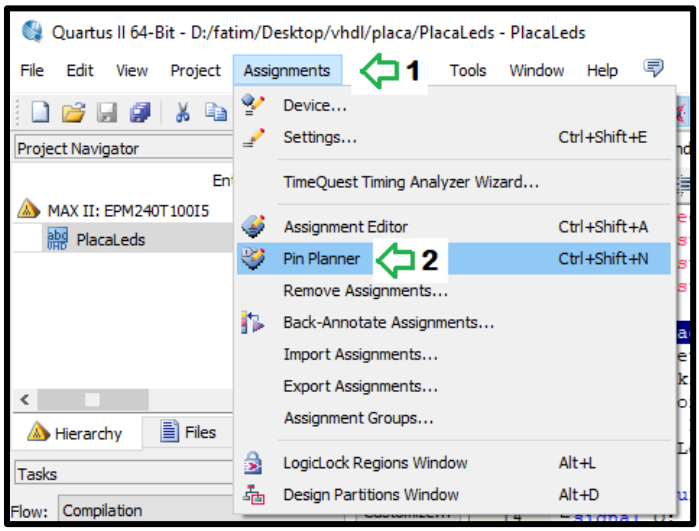

Figura 4.6. Procedimiento para la asignación de pins.

Al seleccionar la opción de Pin Planner se abre una ventana emergente, figura 4.7, en ella se observan los siguientes elementos:

- El esquema pin a pin del dispositivo programable en nuestro caso MAX II circuito EPM240T100C5 (1).
- Node Name se ubican las variables de entrada y salida (2).
- En la sección Filter Location se muestran los pines que recomienda el software para la implementación del circuito (3).
- Se sugiere que esta sea la disposición que usted elija, para ello es necesario reescribir estos pins en la sección de Location (4).

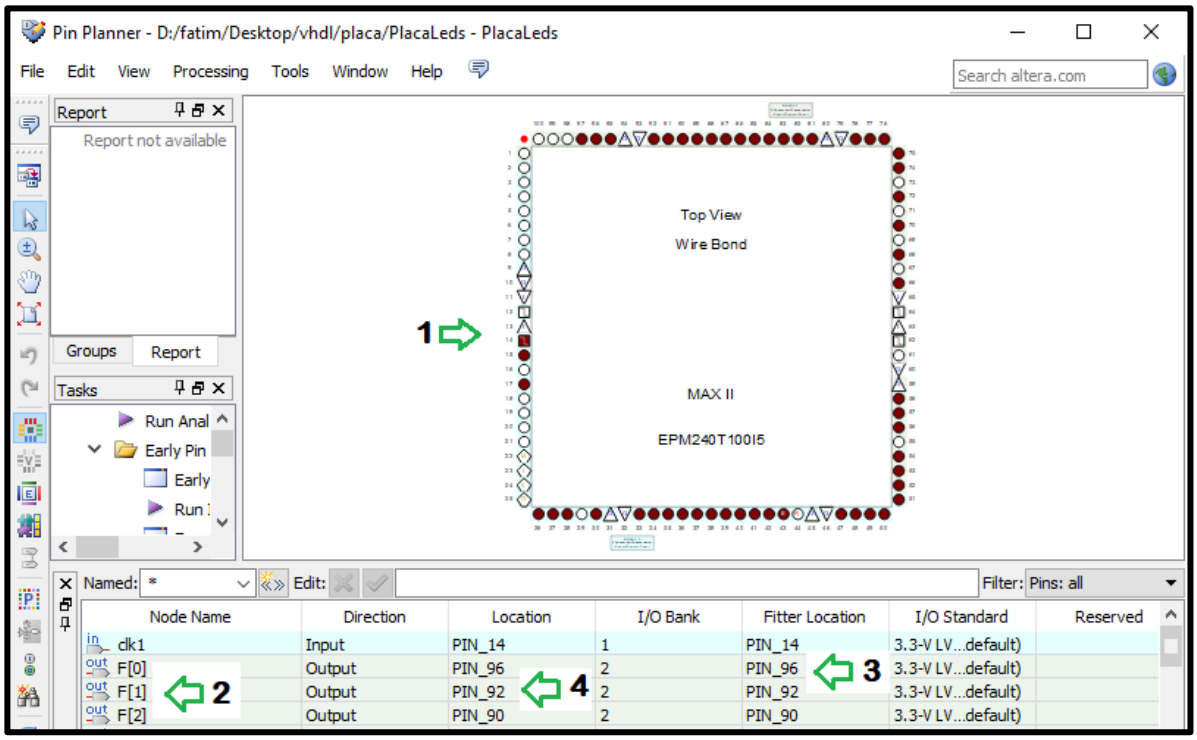

Figura 4.7. PINPLANER para configurar pines.

Se asigna a cada uno de los 50 leds un número de pin, que se ha explicado en el capítulo 3 en la figura 3.6, así como también los 3 pines que corresponden a los botones y el interruptor.

Al terminar la asignación de pins es necesario volver a compilar para que al hacerlo el programa considere la asignación y pueda activar las entradas salidas del circuito. Se cierra la ventana “PinPlaner”, y se procede a carga el programa en la tarjeta CPLD para verificar que funcione correctamente.

Para poder descargar el código de la aplicación al circuito programable debemos tener instalado el USB-Blaster, para determinar si el Blaster está activado entonces debemos, activar la opción de Tools (1), posteriormente seleccionar la opción Programmer (2), tal y como se muestra en la figura 4.8.

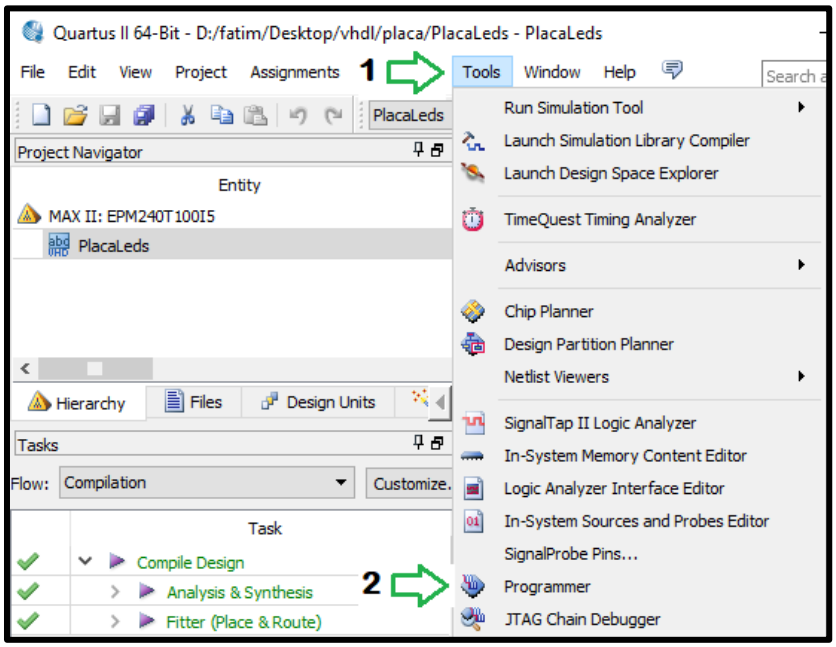

Figura 4.8. Pantalla de inicio para la programación del circuito.

Después de seleccionar la opción de Programmer aparece la pantalla mostrada en la figura 4.9, en ella debe observarse si se encuentra activado (1) el USB-Blaster (USB-O), en caso de no aparecer es necesario realizar la activación correspondiente

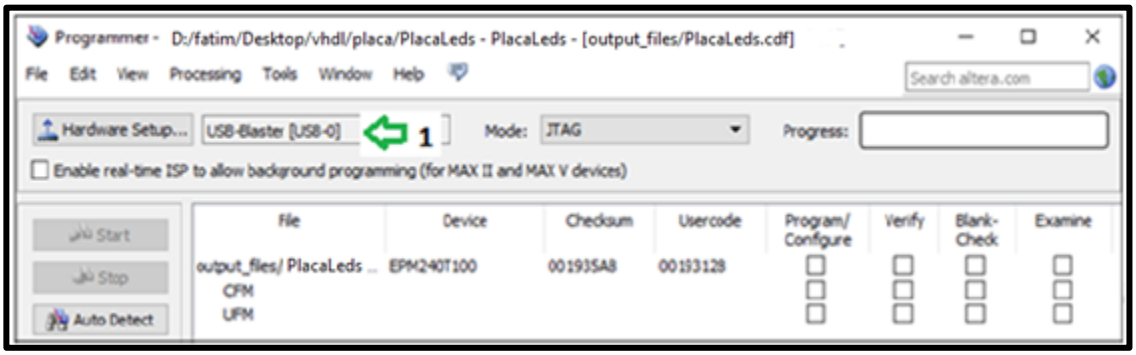

Figura 4.9. Verificación de conexión del USB-Blaster.

Para realizar la programación del dispositivo es necesario activar la opción de tools y posteriormente programmer para verificar que el USB Blaster se encuentre instalado, tal como se muestra en la figura 4.10 se siguen los siguientes pasos:

1. Inicialmente verificamos que el USB Blaster este activado.
2. Seleccionamos Add File.
3. Seleccionamos output_files.
4. Seleccionar el archivo que se requiere descargar, para nuestro proyecto es PlacaLeds.pof.

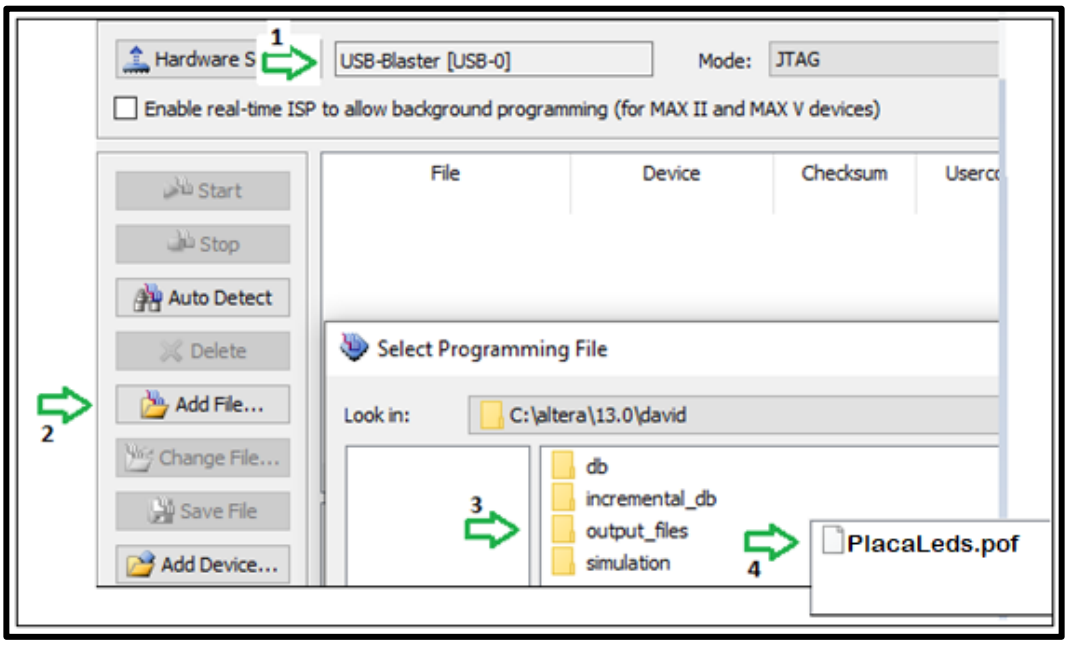

Figura 4.10. Secuencia para descargar el circuito para su grabación

La extensión (.pof) es generada cuando el programa a grabar es un dispositivo de la familia MAX ll CPLD EPM240T100C5. Para la grabación se requiere conectar el USB Blaster a la tarjeta en el adaptador JTAG, figura 4.11.

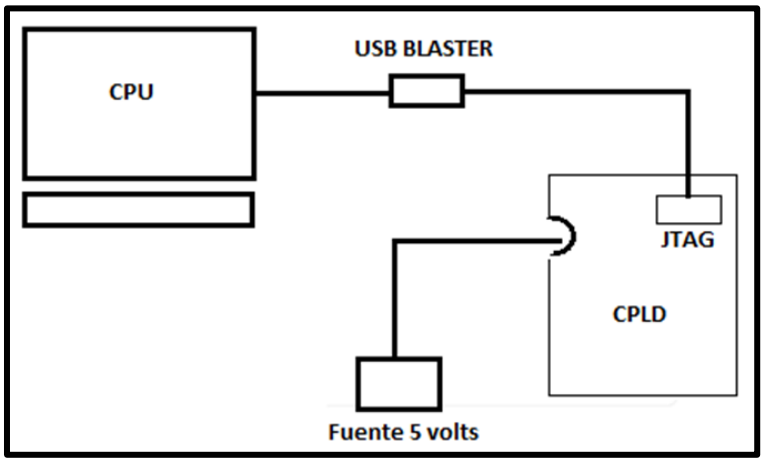

Figura 4.11. Conexión del USB Blaster a la Tarjeta CPLD.

El dispositivo programable requiere de una alimentación de 5 volts y debe tenerse cuidado de no exceder el voltaje requerido. Con el dispositivo conectado al USB Blaster debe seguirse la siguiente secuencia para la descarga:

1) El archivo con extensión.pof deberá aparecer en pantalla.
2) Revisar que el dispositivo a programar aparezca en la pantalla.
3) Deberá activar los tres bloques marcados en la figura

Figura 4.12. Solo seleccione los cuadros marcados.

4) Aplicar Start para iniciar con la grabación
5) En Progress deberá aparecer el porcentaje de avance hasta llegar al 100%.
6) Después de la grabación podrá conectar los pines de la tarjeta a la aplicación correspondiente.

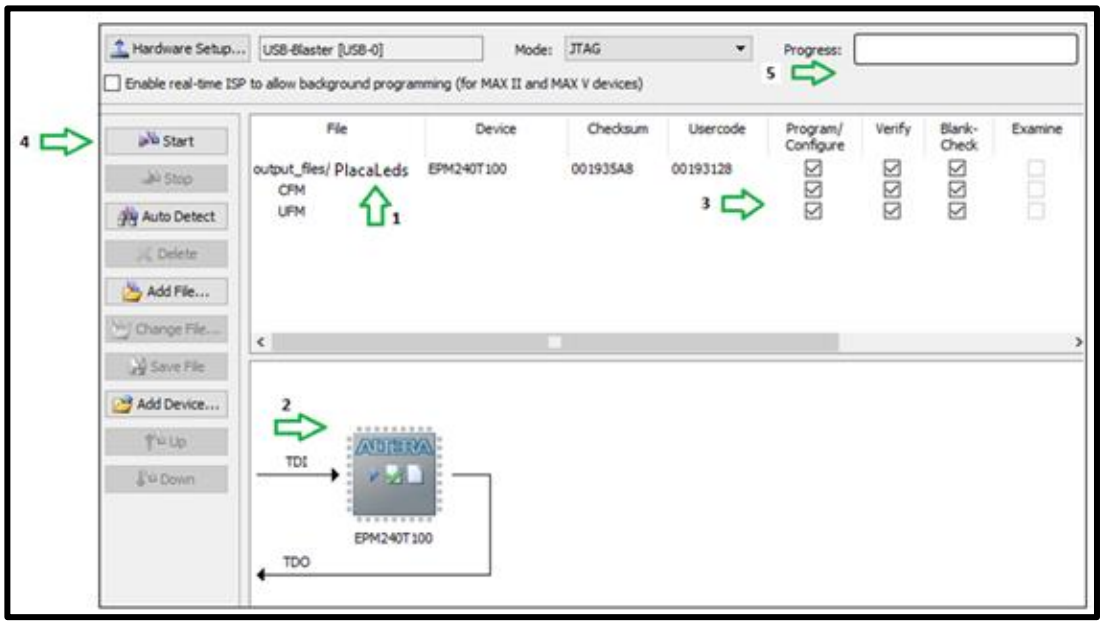

Figura 4.13. Secuencia de pasos para la grabación del dispositivo programable

4.2 Programación de matrices 8x8 con Arduino Mega.

A partir de este tema, se comienza a programar utilizando la plataforma de programación Arduino. En este proyecto la tarjeta de programación que se utiliza es Arduino MEGA, en la cual se agregara el programa final que manipulara todos los dispositivos de visualización del prototipo.

Para comenzar, es necesario abrir una ventana nueva en la plataforma Arduino, previo a esto, deberá estar instalado el software en la computadora. Se da click en el icono de Arduino y a continuación, como se muestra en la figura 4.14, se abre la ventana principal de Arduino para comenzar a ingresar el código.

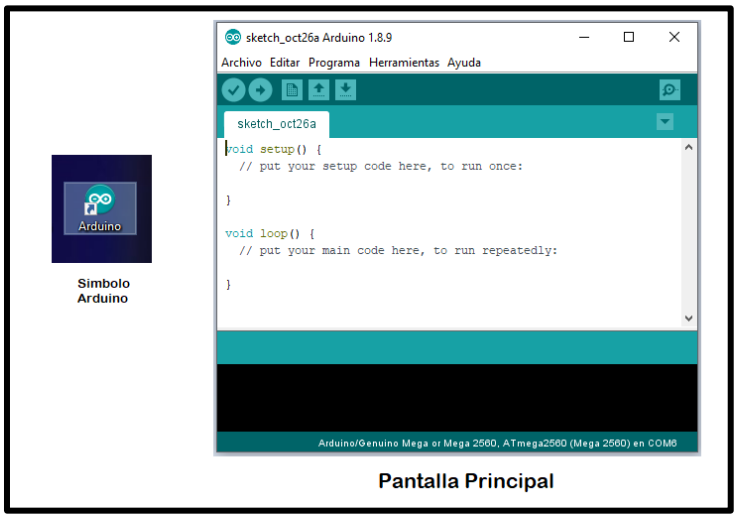

Figura 4.14. Icono y pantalla principal de Arduino.

Para programar las matrices de 8x8 es necesario agregar la librería "**LedControlMS.h**".

Teniendo instalada la librería en Arduino, se inicia el programa agregando la librería en la línea 3 con un “include” y en seguida en la línea 4 se define la variable FRACCION con la cantidad de matrices con la que se va a trabajar, en este caso, 5. En la línea 6, se declara los pines que controlan las matrices 8x8; es decir, 12, 11 y 10, así como la variable que contiene el número de displays conectados en serie, como se muestra en la figura 4.15.

Figura 4.15. Librería LedControlIMS.

A partir de la línea 17 hasta la 23, se declaran las variables que se ocuparan para los botones e interruptores, así como de la línea 27 a la 32, declaramos las variables necesarias para crear los contadores que mostraran las cifras en el numerador y el denominador, figura 4.16.

```
10 String digits= "0123456789";
11 String digits1= "012345";
12 int tiempo=100;

14 //+++++++++++++++ VARIABLES CORRESPONDIENTES AL PARTICIPANTE 1 ++++++++++++++++++++++++
17 const int boton=4;     //boton numerador
18 const int boton2=5;    //boton denominador
19 const int boton3=13;   //boton enter
20 const int boton4=8;    //boton reset
21 const int IntNum=7;    //interruptor para numerador
22 const int IntDen=6;    //interruptor para denominador
23 const int IntEnter=9;  //interruptor para Enter
25 const int motor1a=22;   //motor
26 const int motor1b=24;   //motor
28 int Orden=0;  //CONTADOR PARA ENTER
29 int conta=0;    //CONTADOR PARA NUMERADOR
30 int conta1=0;
31 int cont=0; //CONTADOR PARA DENOMINADOR
32 int cont1=0;
```

Figura 4.16. Definición de variables.

En esta parte, se realiza la configuración inicial de las variables. Se asigna el tipo que serán las variables que se declararon anteriormente; en este caso, de la línea 44 a la 50, todos los botones e interruptores son variables de entrada ya que el Arduino recibirá la señal que emitan, cada vez que los botones sean presionados o en su defecto la posición que tengan los interruptores. En la línea 54 iniciamos la instrucción FOR que ayuda a activar las matrices 8x8, figura 4.17.

```
34 void setup() {
35   //+++++++++++++++DISPOSITIVOS DEL PANEL DEL PARTICIPANTE 1 ++++++++++++++++++++++++
44   pinMode (boton,  INPUT);
45   pinMode (boton2, INPUT);
46   pinMode (boton3, INPUT);
47   pinMode (boton4, INPUT);
48   pinMode (IntNum, INPUT);
49   pinMode (IntDen, INPUT);
50   pinMode (IntEnter, INPUT);
54 for (int i=0; i<FRACCION; i++){
55     lc.shutdown(i,false);     // Activar las matrices
56     lc.setIntensity(i,8);     // Se ajusta el brillo de los LED´s como se desee (0-15)
57     lc.clearDisplay(i);}      // Se borra todo
58 }
```

Figura 4.17. Función Void Setup.

Es necesario tener presente que la distribución de las matrices consiste en que las dos primeras corresponden a un valor del 0 al 50 para el numerador, la tercera matriz muestra la diagonal que define la forma de

fracción y al final, las dos últimas muestran el valor del 0 al 50 para el denominador.

Para visualizar la diagonal, se crea una función llamada SIGNO que compete de la línea 60 a la 77, como esta función no devuelve ningún valor será del tipo “void”, que significa “función vacía”. Dentro de esta función se desarrolla la figura de la diagonal mediante patrones en binario que definen el estado de cada led de la matriz, este código binario se muestra entre las líneas 68 a la 75 y es llamado por medio de la instrucción SetRow y una variable llamada SG del tipo byte, tal como se muestra en la figura 4.18.

```
60 //++++++++++++++++++++ SIGNO DE LA MATRIZ DEL PANEL DEL PARTICIPANTE 1 ++++++++++++++++++++++++++
61 void SIGNO(){
65     /*estos son los caracteres para mostrar el signo de diagonal */
66     byte sg[8]={B11000000,B01100000,B00110000,B00011000,B00001100,B00000110,B00000011,B00000001};
67     /*manda a imprimir cada uno de los caracteres */
68   lc.setRow(2,0,sg[0]);
69   lc.setRow(2,1,sg[1]);
70   lc.setRow(2,2,sg[2]);
71   lc.setRow(2,3,sg[3]);
72   lc.setRow(2,4,sg[4]);
73   lc.setRow(2,5,sg[5]);
74   lc.setRow(2,6,sg[6]);
75   lc.setRow(2,7,sg[7]);
76     delay (tiempo);
77   }
```

Figura 4.18. Función SIGNO.

Para visualizar el valor del numerador y del denominador se crean dos funciones en las cuales, ambas tendrán un contador que realiza el conteo cada vez que el botón correspondiente sea presionado y a su vez mostrara en las matrices 8x8 el valor obtenido. Solo se mostrará la estructura de una de las dos funciones, puesto que ambas realizan el mismo trabajo, la única diferencia es que trabajan con botones e interruptores diferentes. En la línea 81 se comienza con la función “numerador”, que como su nombre lo indica, mostrara el valor en las dos primeras matrices 8x8 que se asigne. Esta función será del tipo entero (int) puesto que al final devolverá un valor de número entero.

A partir de la línea 82 a la 88 se crean las variables que necesarias para realizar el contador y el manejo de los botones e interruptor, figura 4.19.

```
81 int NUMERADOR1(){
82 int mas=digitalRead(boton);      //Conteo ASCENDENTE NUMERADOR
83 int menos=digitalRead(boton);    //conteo DESCENDENTE NUMERADOR
85 char uni=digits[conta];         //digitos de unidades
86 char des=digits1[conta1];       //digitos de decenas
87 int Numerador=conta;            //Devuelve el valor final del numerador
88 int interruptor1=digitalRead(IntNum);   //interruptor para cambio ascendente-descendente
```

Figura 4.19. Función NUMERADOR.

En la figura 4.20 se observa; de la línea 97 a la 112, el desarrollo del contador ascendente indicando; en la línea 99, que este será activo cuando el interruptor se encuentre en estado alto (HIGH).

```
 97 if (interruptor1==HIGH){
 98
 99   if (mas==HIGH){ //CONTROLADOR ASCENDENTE
100     conta++;
101     if (conta==10)
102         {
103           conta1++;
104           conta=0;
105             }
106     if (conta1==5)
107         {
108             if (conta==1)
109                 {
110                    conta1=0;
111                    conta=0;
112                   }}}}
```

Figura 4.20. Contador ascendente del numerador.

O en caso contrario; figura 4.21, si el interruptor correspondiente al botón numerador se encuentra en estado bajo (LOW), el contador trabajará de manera descendente. Esta sentencia se muestra a partir de la línea 114 hasta la línea 128.

```
114 else {
115 if(menos==HIGH)     //CONTROLADOR DESCENDENTE
116 {
117  if(conta==0)
118  {
119   conta=10;
120     if(contal==0)
121          {
122          contal=5;
123          conta=1;
124           }}
125  if (conta==10)
126  {contal--;}
127 conta--;
128 }}
```

Figura 4.21. Contador descendente del numerador.

De la mano del contador, en la línea 128, se utiliza la instrucción “switch-case” que permite hacer una serie de comparaciones sin necesidad de “if’s” anidados que ayudara a mostrar en los displays, los dígitos (unidad y decena), uno a uno las veces que el botón sea pulsado. Dentro de cada sentencia se tienen las instrucciones para mostrar los números tanto en las matrices 8x8 MAX7219 (lc) como en la pantalla LCD (lcd) que más adelante se describe como es su conexión y programación, figura 4.22.

```
128 switch (conta)
129 {
130 case 0:
131 lc.displayChar(0, lc.getCharArrayPosition(uni));  // unidades
132 lc.displayChar(1, lc.getCharArrayPosition(des));  // decenas
133 lcd.setCursor(0, 1);
134 lcd.print(des);
135 lcd.print(uni);
136 delay(tiempo);
137 break;
139 case 1:
140 lc.displayChar(0, lc.getCharArrayPosition(uni));
141 lc.displayChar(1, lc.getCharArrayPosition(des));
142 lcd.setCursor(0, 1);
143 lcd.print(des);
144 lcd.print(uni);
145 delay(tiempo);
146 break;
148 case 2:
149 lc.displayChar(0, lc.getCharArrayPosition(uni));
150 lc.displayChar(1, lc.getCharArrayPosition(des));
151 lcd.setCursor(0, 1);
152 lcd.print(des);
153 lcd.print(uni);
154 delay(tiempo);
155 break;
156
157 case 3:
158 lc.displayChar(0, lc.getCharArrayPosition(uni));
159 lc.displayChar(1, lc.getCharArrayPosition(des));
160 lcd.setCursor(0, 1);
161 lcd.print(des);
162 lcd.print(uni);
163 delay(tiempo);
164 break;
166 case 4:
167 lc.displayChar(0, lc.getCharArrayPosition(uni));
168 lc.displayChar(1, lc.getCharArrayPosition(des));
169 lcd.setCursor(0, 1);
170 lcd.print(des);
171 lcd.print(uni);
172 delay(tiempo);
173 break;
175 case 5:
176 lc.displayChar(0, lc.getCharArrayPosition(uni));
177 lc.displayChar(1, lc.getCharArrayPosition(des));
178 lcd.setCursor(0, 1);
179 lcd.print(des);
180 lcd.print(uni);
181 delay(tiempo);
182 break;
184 case 6:
185 lc.displayChar(0, lc.getCharArrayPosition(uni));
186 lc.displayChar(1, lc.getCharArrayPosition(des));
187 lcd.setCursor(0, 1);
188 lcd.print(des);
189 lcd.print(uni);
190 delay(tiempo);
191 break;
193 case 7:
194 lc.displayChar(0, lc.getCharArrayPosition(uni));
195 lc.displayChar(1, lc.getCharArrayPosition(des));
196 lcd.setCursor(0, 1);
197 lcd.print(des);
198 lcd.print(uni);
199 delay(tiempo);
200 break;
201
202 case 8:
203 lc.displayChar(0, lc.getCharArrayPosition(uni));
204 lc.displayChar(1, lc.getCharArrayPosition(des));
205 lcd.setCursor(0, 1);
206 lcd.print(des);
207 lcd.print(uni);
208 delay(tiempo);
209 break;
211 case 9:
212 lc.displayChar(0, lc.getCharArrayPosition(uni));
213 lc.displayChar(1, lc.getCharArrayPosition(des));
214 lcd.setCursor(0, 1);
215 lcd.print(des);
216 lcd.print(uni);
217 delay(tiempo);
218 break;
219 }
220 return Numerador;  }
```

Figura 4.22. Estructura del switch – case para el contador.

Como se mencionó anteriormente, el desarrollo de la función “numerador” es básicamente la misma que la función “denominador”, lo único que cambia es el nombre de las variables que se utilizan puesto que ambos son contadores independientes manipulados por distintos componentes.

Se finaliza el programa con la función **void loop()**, una de las dos partes básicas de la estructura de un programa en Arduino. En la línea 371 comienza esta función en la cual se ingresa el código que se repetirá de forma indefinida. Esta función solo se ejecutará una vez, después de cada encendido o reinicio de la placa Arduino.

La función comienza con la declaración de las variables en las líneas 375, 376 y 377, que leerán las señales de entrada de los botones enter y reset, así como también el interruptor del enter. Figura 4.23.

Figura 4.23. Función loop().

Se continua en las líneas 380, con las indicaciones para el interruptor del “Enter”, este interruptor tendrá dos funciones, la de activar las matrices 8x8 para introducir las fracciones y la de avanzar los móviles cuando sus respuestas hayan sido validadas. En la primera; como se muestra en la línea 380, cuando el interruptor este en alto (HIGH), activará las matrices 8x8 por secciones, es decir, en las líneas 389 a la 391, el primer pulso del botón “Enter” activa las dos primeras matrices que corresponden al numerador. De la línea 393 a la 395, correspondiente al segundo pulso, activa la tercera matriz que muestra la diagonal. De la línea 397 la 399, corresponde al tercer pulso que activa la sección del denominador. De la línea 401 a la 405 correspondiente al último pulso del “Enter” validara la fracción ingresada, en este caso el profesor es quien, valida esta respuesta, figura 4.24.

fraccion_interruptor

```
379 //+++++++++++++++++++++++++ FUNCIONES DEL BOTON ENTER ++++++++++++++++++++++++++++++++
380 if (InterruptorEnter==HIGH)
381 {
382 if (enter==HIGH)
383         {Orden++;
384       if (Orden==5)
385           {Orden=0;}
386         }
387
388 switch (Orden){
389 case 1: NUMERADOR1();
390 delay(tiempo);
391 break;
392
393 case 2: SIGNO();
394 delay(tiempo);
395 break;
396
397 case 3: DENOMINADOR1();
398 delay(tiempo);
399 break;
400
401 case 4:
402 lcd.setCursor(6,1);
403 lcd.print("VALIDAR R");
404 delay (tiempo);
405 break;
406 }}
```

Figura 4.24. Instrucciones para el interruptor en modo ACTIVAR.

La segunda función del interruptor es para mover los móviles, esta función se activa cuando el interruptor está en bajo (LOW); línea 408, en esta posición, cada pulso del enter servirá para que el móvil avance la distancia que tenga señalada el carril, esto se muestra en la figura 4.25.

fraccion_interruptor

```
408 if (InterruptorEnter==LOW)
409 {
410   if (enter==HIGH){
411   digitalWrite(motorla, LOW);
412   digitalWrite(motorlb, HIGH);
413   delay(100);
414   }
415   else
416   {
417     digitalWrite(motorla, LOW);
418    digitalWrite(motorlb, LOW);}
419   }
```

Figura 4.25. Instrucciones para el interruptor en modo AVANZAR.

Finalmente se ingresa el bloque de instrucciones que corresponden al botón reset; figura 4.26. Con este botón se borra toda información o datos que hayan sido ingresados al prototipo, es decir, el valor de los contadores cambia a cero, las matrices 8x8 MAX7219 y los bloques de leds se apagaran, así como en la pantalla LCD se borrará la fracción ingresada. Todo volverá a su estado inicial, pero también este mismo botón tendrá una función más, así como regresa los valores a su estado inicial, así también servirá para retroceder el móvil la distancia que se requiera. Con esto se cierra la función loop(), y se termina de escribir todo el programa.

fraccion_interruptor

```
//++++++++++++++++++++++++++++ FUNCIONES DEL BOTON RESET ++++++++++++++++++++++++++++++++
if (reset==HIGH)
  {
    if (InterruptorEnter==LOW)
  {
     digitalWrite(motorla, HIGH);
     digitalWrite(motorlb, LOW);
     delay(100);
  }
    else
  {
    digitalWrite(motorla, LOW);
    digitalWrite(motorlb, LOW);
    delay(100);
  }

//TODOS LOS CONTADORES REGRESAN AL VALOR CERO
conta=0;
conta1=0;
cont=0;
cont1=0;
Orden=0;
lc.clearDisplay(0);
lc.clearDisplay(1);
lc.clearDisplay(2);
lc.clearDisplay(3);
lc.clearDisplay(4);
lcd.clear();
}}
```

Figura 4.26. Instrucciones para el botón reset.

Compilamos el programa para encontrar algunos errores que se hayan producido al momento de la redacción, en caso de haberlos corregimos y compilamos por propia cuenta hasta que ya no se tenga ningún error. Se carga el programa a la tarjeta Arduino Mega y se realizan pruebas.

4.3 Programación de Pantalla LCD con Arduino Mega.

Después de realizar las conexiones, se procede a ingresar el código al programa que ya se realizó anteriormente con las matrices 8x8 MAX7219. Se complementará el código para que los componentes trabajen de manera simultánea.

Se comienza con descargar e instalar la librería que se ocupará para el adaptador I2C, esta librería se llama **LiquidCrystal_I2C.h**.; en la línea 2 se añade la librería al programa, figura 4.27.

```
1 #include <Wire.h>
2 #include <LiquidCrystal_I2C.h>
```

Figura 4.27. Librerías para adaptador I2C.

En este punto se hará un pequeño paréntesis, antes de continuar con el código es importante mencionar que cada componente que se conecte al I2C tiene una dirección única, cada mensaje u orden que se transmita a este adaptador, lleva anexa esta dirección para indicar quien es el receptor de este mensaje. Para conocer esta dirección de una manera más fácil y rápida es necesario descargar un pequeño programa Arduino (I2C scanner) que ayudará a escanear el adaptador y conocer la dirección del componente con el que está trabajando, esto se puede hacer con alguna tarjeta Arduino que se tenga extra. La pantalla LCD maneja la dirección 0x3E hexadecimal.

Con esta información ya se puede continuar con el código. Se crea una instancia del objeto LiquidCrystal_I2C llamada lcd en la línea 7, donde se anexa la dirección y tamaño de la pantalla, figura 4.28.

```
7 LiquidCrystal_I2C lcd (0x3E,16,2);
```

Figura 4.28. Instancia LCD.

Dentro de la función Setup() se agrega las instrucciones de la línea 34 a la 37; figura 4.29, que ayudarán a activar la pantalla LCD.

```
33 void setup() {
34   Wire.begin();
35   lcd.begin(16,2);      // Inicializar el LCD participante 1
36   lcd.clear();
37   lcd.backlight();       //Encender la luz de fondo.
```

Figura 4.29. Activación de pantalla LCD.

Siguiendo con las funciones, dentro de la función **SIGNO()** se anexan las siguientes líneas que se muestran en la figura 4.30, donde se indica la posición del cursor dentro de la pantalla (línea 60) y el mensaje que se mostrará en ese lugar (línea 61), en este caso es la diagonal. Así cada vez que se mande a llamar esta función, aparecerá en la pantalla LCD la diagonal.

```
59 void SIGNO(){
60 lcd.setCursor(2, 1); // Ubicamos el cursor en la tercera posición(columna:2) de la segunda línea(fila:1)
61 lcd.print("/");      // Escribimos el simbolo de la diagonal
```

Figura 4.30. Mostrar diagonal en la pantalla LCD.

Para lo que sigue, se debe recordar que anteriormente se habló de las matrices 8x8 MAX7219 y dentro de la programación explicábamos la estructura de la función **NUMERADOR()**, en la cual se usa un **switch–case**. A esta sentencia le agregan tres líneas en cada **case**, que nos servirán para que la pantalla LCD y las matrices 8x8 MAX7219 trabajen de forma simultánea al momento de mostrar las fracciones ingresadas, figura 4.31.

```
133 lcd.setCursor(0, 1);
134 lcd.print(des);
135 lcd.print(uni);
```

Figura 4.31. Escribir la fracción ingresada.

Finalmente, en la función **loop()**, en las líneas 372 y 373 se indica que se muestre un mensaje de bienvenida en la primera línea de la pantalla, puesto que la fracción que se introduzca se mostrara en la segunda línea, figura 4.32. Este mensaje estará visible en la pantalla siempre que la función l**oop()** inicie su ciclo.

```
371 void loop() {
372 lcd.setCursor(0,0);
373 lcd.print("BIENVENIDO");
```

Figura 4.32. Muestra mensaje de bienvenida en pantalla LCD.

También se agrega al código unas líneas de instrucción en la parte del switch-case que maneja el botón enter, tal como se muestra en la figura 4.33, cuando el participante a ingresado la fracción al prototipo, presionará de nuevo enter y en la pantalla aparecerá el siguiente mensaje "VALIDAR R", en este punto el profesor verifica que la fracción que se ingreso sea correcta.

```
379 //+++++++++++++++++++++++++ FUNCIONES DEL BOTON ENTER ++++++++++++++++++++++++++++++++
380 if (InterruptorEnter==HIGH)
381 {
382 if (enter==HIGH)
383         {Orden++;
384       if (Orden==5)
385           {Orden=0;}
386         }
388 switch (Orden){
389 case 1: NUMERADOR1();
390 delay(tiempo);
391 break;
393 case 2: SIGNO();
394 delay(tiempo);
395 break;
397 case 3: DENOMINADOR1();
398 delay(tiempo);
399 break;
401 case 4:
402 lcd.setCursor(6,1);
403 lcd.print("VALIDAR R");
404 delay (tiempo);
405 break;
406 }}
```

Líneas de código para mostrar mensaje.

Figura 4.33. Escribe VALIDAR R en LCD.

Cada vez que se presiona el botón "Reset" se borrará todo lo que este escrito en la segunda línea de la pantalla. Para esto solo se agrega la siguiente línea de código, **lcd.clear()**, dentro de la instrucción que esté trabajando con el botón reset. En la figura 4.34, se ve la pantalla después de ingresar la fracción y la respuesta cuando se presiona enter.

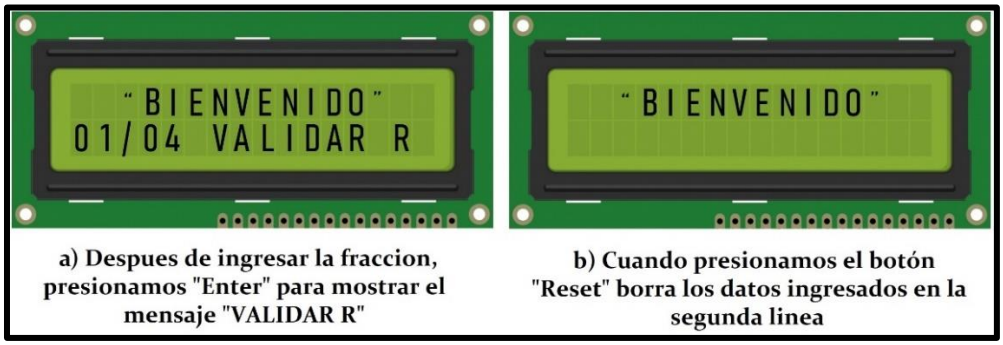

Figura 4.34. Función del botón reset.

Después de agregar el código al programa Arduino, se compila para verificar que no tenga errores, en caso de tenerlos, se corrige y compila de nueva cuenta. Si la compilación no presenta errores, se carga el programa a la tarjeta Arduino Mega y se realizan pruebas con los componentes.

4.4. Programación de motores con Arduino Mega.

La parte de la programación es muy sencilla, solo se agregan un par de líneas al programa principal. Lo que interesa es que los móviles se muevan hacia adelante y hacia atrás, según sea el caso. Para que esto suceda se utilizaran los botones de enter y reset, respectivamente.

Lo primero que se tiene que hacer es declarar las dos entradas que se conectan de la tarjeta Arduino al controlador, esto servirá para enviar las instrucciones al controlador que moverán los móviles hacia adelante o hacia atrás. En la figura 4.35, en las líneas 24 y 25 tenemos “motor1a” y “motor1b” para las entradas IN1 e IN2 ó IN3 e IN4 según sea el caso.

```
24 const int motorla=22;   //motor
25 const int motorlb=24;   //motor
```

Figura 4.35. Declaración de variables de entrada del controlador.

Dentro de la función **setup()**, en las líneas 49 y 50 se ingresa la configuración de estas variables, es decir, de que tipo serán los pines, en este caso, de salida, figura 4.36.

```
49  pinMode(motorla, OUTPUT);   //variables para el motor
50  pinMode(motorlb, OUTPUT);   //variables para el motor
```

Figura 4.36. Configuración de las variables.

Puesto que solo se tiene que los móviles avancen o retrocedan, como se mencionó anteriormente, estas acciones las realizaremos con los botones Enter y Reset. En la figura 4.37, de la línea 408 a la 419 se tiene el código que se agrega al programa principal en la que se indica por medio de una sentencia “if” que cuando el interruptor del botón “enter” este en posición “bajo” (low), se cumplirá las instrucciones que se agregan dentro de las llaves, dentro de la cual, se ha agregado un “if-else” para indicar que cada vez que el botón “enter” sea presionado el móvil avanzará hacia adelante. Esta sentencia solo se cumplirá cuando el interruptor del botón enter este en posición de avanzar (low), la cual ya se ha mencionado en el capítulo anterior donde se habla del funcionamiento del prototipo.

```
408 if (InterruptorEnter==LOW)
409 {
410   if (enter==HIGH){
411   digitalWrite(motorla, LOW);
412   digitalWrite(motorlb, HIGH);
413   delay(100);
414   }
415   else
416   {
417     digitalWrite(motorla, LOW);
418    digitalWrite(motorlb, LOW);}
419   }
```

Figura 4.37. Instrucción para avanzar el móvil.

Ahora bien, si lo que se quiere es que el carro(móvil) retroceda cierta distancia, ya sea unos centímetros o volver a la posición de salida, entonces se usará el botón “reset” sin mover de posición el interruptor del botón “enter” que se usa para avanzar el carro(móvil). En la figura 4.38,

dentro de la sentencia “if” en la que se tienen las instrucciones que se deben cumplir cada vez que el botón “reset” sea presionado, se agrega una sentencia “if-else” en el cual se indica que mientras el “InterruptorEnter” este en estado bajo (low), se cumplirá las indicaciones dentro de las llaves, de lo contrario se cumplirán las instrucciones dentro de la sentencia “else”.

En otras palabras, mientras el interruptor del botón “enter” se encuentra en estado bajo (low) o posición de “avanzar”, cada vez que se presiona el botón “reset”, el carro(móvil) ira retrocediendo, si se mantiene el botón presionado el carro retrocederá hasta que se deje de presionar.

```
421 //++++++++++++++++++++++++++++ FUNCIONES DEL BOTON RESET ++++++++++++
422 if (reset==HIGH)
423   {
424     if (InterruptorEnter==LOW)
425   {
426      digitalWrite(motorla, HIGH);
427      digitalWrite(motorlb, LOW);
428      delay(100);
429   }
430     else
431   {
432     digitalWrite(motorla, LOW);
433     digitalWrite(motorlb, LOW);
434     delay(100);
435   }
```

Figura 4.38. Instrucción para retroceder el móvil.

Después de agregar las líneas de código al programa principal de Arduino Mega para el funcionamiento de los motores, se compila y verifica que no se tenga ningún error. Carga el programa a la tarjeta Arduino Mega y se realizan pruebas de funcionamiento.

APÉNDICE 1

Programa General en Arduino para controlar los dispositivos de visualización y manipulación del tablero del participante.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "LedControlMS.h"
#define FRACCION 5

LedControl lc=LedControl(12,11,10, FRACCION);
LiquidCrystal_I2C lcd (0x3E,16,2);   //PARTICIPANTE 1
                                     //participante 2 es (0x3E,16,2);

String digits= "0123456789";
String digits1= "012345";
int tiempo=100;

//++++++ VARIABLES CORRESPONDIENTES AL PARTICIPANTE 1 ++++++
int numclk=3;            //señal del botón para el CPLD NUMERADOR
int denclk=2;            //señal del botón para el CPLD DENOMINADOR
const int boton=4;     //botón numerador
const int boton2=5;    //botón denominador
const int boton3=13;   //boton enter
const int boton4=8;    //boton reset
const int IntNum=7;    //interruptor para numerador
const int IntDen=6;    //interruptor para denominador
const int IntEnter=9;  //interruptor para Enter
const int motor1a=22;   //motor
const int motor1b=24;   //motor

int Orden=0;  //CONTADOR PARA ENTER
int conta=0;    //CONTADOR PARA NUMERADOR
int conta1=0;
int cont=0; //CONTADOR PARA DENOMINADOR
int cont1=0;

void setup() {
  Wire.begin();
  lcd.begin(16,2);      // Inicializar el LCD participante 1
  lcd.clear();
  lcd.backlight();      //Encender la luz de fondo.

//+++++++++ DISPOSITIVOS DEL PANEL DEL PARTICIPANTE 1 +++++++
```

```
  pinMode (numclk, OUTPUT);
  pinMode (denclk, OUTPUT);
  pinMode (boton,  INPUT);
  pinMode (boton2, INPUT);
  pinMode (boton3, INPUT);
  pinMode (boton4, INPUT);
  pinMode (IntNum, INPUT);
  pinMode (IntDen, INPUT);
  pinMode (IntEnter, INPUT);
  pinMode(motor1a, OUTPUT);   //variables para el motor
  pinMode(motor1b, OUTPUT);   //variables para el motor

for (int i=0; i<FRACCION; i++){
    lc.shutdown(i,false);     // Activar las matrices
    lc.setIntensity(i,8);     // Se ajusta el brillo de los LED´s como se desee (0-15)
    lc.clearDisplay(i);}      // Se borra todo
}

//++++++ SIGNO DE LA MATRIZ DEL PANEL DEL PARTICIPANTE 1 ++++++
void SIGNO(){
lcd.setCursor(2, 1); // Ubicamos el cursor en la tercera posición(columna:2) de
la segunda línea(fila:1)
lcd.print("/");      // Escribimos el simbolo de la diagonal

    /*estos son los caracteres para mostrar el signo de diagonal */
    byte
sg[8]={B11000000,B01100000,B00110000,B00011000,B00001100,B0000011
0,B00000011,B00000001};
    /*manda a imprimir cada uno de los caracteres */
  lc.setRow(2,0,sg[0]);
  lc.setRow(2,1,sg[1]);
  lc.setRow(2,2,sg[2]);
  lc.setRow(2,3,sg[3]);
  lc.setRow(2,4,sg[4]);
  lc.setRow(2,5,sg[5]);
  lc.setRow(2,6,sg[6]);
  lc.setRow(2,7,sg[7]);
    delay (tiempo);
  }

//+++++++++++++++++ PANEL DEL PARTICIPANTE  1 +++++++++++++++

int NUMERADOR1(){
int mas=digitalRead(boton);      //Conteo ASCENDENTE NUMERADOR
```

```
int menos=digitalRead(boton);    //conteo DESCENDENTE NUMERADOR

char uni=digits[conta];        //digitos de unidades
char des=digits1[conta1];      //digitos de decenas
int Numerador=conta;           //Devuelve el valor final del numerador
int interruptor1=digitalRead(IntNum);   //interruptor para cambio ascendente-
descendente

 if (mas==HIGH)      //manda señal de reloj al CPLD del Numerador
  {
    digitalWrite(numclk, HIGH);
    delay (250);
    digitalWrite(numclk, LOW);
    }

if (interruptor1==HIGH){

  if (mas==HIGH){ //CONTROLADOR ASCENDENTE
    conta++;
    if (conta==10)
       {
         conta1++;
         conta=0;
          }
    if (conta1==5)
       {
          if (conta==1)
             {
               conta1=0;
               conta=0;
              }}}}

else {
if(menos==HIGH)    //CONTROLADOR DESCENDENTE
{
 if(conta==0)
 {
  conta=10;
    if(conta1==0)
       {
       conta1=5;
       conta=1;
        }}
 if (conta==10)
```

```
 {conta1--;}
conta--;
}}

switch (conta)
{
case 0:
lc.displayChar(0, lc.getCharArrayPosition(uni));  // unidades
lc.displayChar(1, lc.getCharArrayPosition(des));  // decenas
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 1:
lc.displayChar(0, lc.getCharArrayPosition(uni));
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 2:
lc.displayChar(0, lc.getCharArrayPosition(uni));
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 3:
lc.displayChar(0, lc.getCharArrayPosition(uni));
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 4:
lc.displayChar(0, lc.getCharArrayPosition(uni));
```

```
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 5:
lc.displayChar(0, lc.getCharArrayPosition(uni));
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 6:
lc.displayChar(0, lc.getCharArrayPosition(uni));
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 7:
lc.displayChar(0, lc.getCharArrayPosition(uni));
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 8:
lc.displayChar(0, lc.getCharArrayPosition(uni));
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;

case 9:
```

```
lc.displayChar(0, lc.getCharArrayPosition(uni));
lc.displayChar(1, lc.getCharArrayPosition(des));
lcd.setCursor(0, 1);
lcd.print(des);
lcd.print(uni);
delay(tiempo);
break;
}
return Numerador;  }

int DENOMINADOR1(){
int asc=digitalRead(boton2);     //Conteo ASCENDENTE
int des=digitalRead(boton2);    //Conteo DESCENDENTE

char unidad=digits[cont];
char decena=digits1[cont1];
int Interruptor2=digitalRead(IntDen);  //interruptor para cambio ascendente-
descendente
int Denominador=cont;        //devuelve el valor final ingresado al denominador

if (asc==HIGH)    //Manda la señal de reloj al CPLD del Denominador
 {
   digitalWrite(denclk, HIGH);
   delay (250);
   digitalWrite(denclk, LOW);
 }

if (Interruptor2==HIGH){  //CONTROLADOR DESCENDENTE

  if (asc==HIGH)
  {
    cont++;
    if (cont==10)
       {
         cont1++;
         cont=0;
          }
    if (cont1==5)
       {
          if (cont==1)
             {
               cont1=0;
               cont=0;
              }}}}
```

```
 else      //CONTROLADOR DESCENDENTE
{
if (des==HIGH)
{
 if(cont==0)
 {
  cont=10;
    if(cont1==0)
        {
        cont1=5;
        cont=1;
         }}

 if (cont==10)
 {
  cont1--;
 }
cont--;
}}

switch (cont)
{
case 0:
lc.displayChar(3, lc.getCharArrayPosition(unidad));   // unidades
lc.displayChar(4, lc.getCharArrayPosition(decena));  // decenas
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 1:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
lc.displayChar(4, lc.getCharArrayPosition(decena));
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 2:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
lc.displayChar(4, lc.getCharArrayPosition(decena));
```

```
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 3:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
lc.displayChar(4, lc.getCharArrayPosition(decena));
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 4:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
lc.displayChar(4, lc.getCharArrayPosition(decena));
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 5:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
lc.displayChar(4, lc.getCharArrayPosition(decena));
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 6:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
lc.displayChar(4, lc.getCharArrayPosition(decena));
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 7:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
```

```
lc.displayChar(4, lc.getCharArrayPosition(decena));
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 8:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
lc.displayChar(4, lc.getCharArrayPosition(decena));
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;

case 9:
lc.displayChar(3, lc.getCharArrayPosition(unidad));
lc.displayChar(4, lc.getCharArrayPosition(decena));
lcd.setCursor(3, 1);
lcd.print(decena);
lcd.print(unidad);
delay(tiempo);
break;
}
return Denominador;}

//+++++++++++++++++ INTRUCCIONES DEL PROGRAMA ++++++++++++++
void loop() {
lcd.setCursor(0,0);
lcd.print("BIENVENIDO");

int InterruptorEnter= digitalRead(IntEnter);  //Interruptor del enter para
activar/avanzar
int enter=digitalRead(boton3);   //boton enter
int reset=digitalRead(boton4);   //boton reset

//+++++++++++++++++ FUNCIONES DEL BOTON ENTER +++++++++++++++++
if (InterruptorEnter==HIGH)
{
if (enter==HIGH)
       {Orden++;
     if (Orden==5)
        {Orden=0;}
```

```
      }

switch (Orden){
case 1: NUMERADOR1();
delay(tiempo);
break;

case 2: SIGNO();
delay(tiempo);
break;

case 3: DENOMINADOR1();
delay(tiempo);
break;

case 4:
lcd.setCursor(6,1);
lcd.print("VALIDAR R");
delay (tiempo);
break;
}}

if (InterruptorEnter==LOW)
{
  if (enter==HIGH){
  digitalWrite(motor1a, LOW);
  digitalWrite(motor1b, HIGH);
  delay(100);
  }
  else
  {
    digitalWrite(motor1a, LOW);
   digitalWrite(motor1b, LOW);}
  }

//+++++++++++++++ FUNCIONES DEL BOTON RESET +++++++++++++++
if (reset==HIGH)
  {
    if (InterruptorEnter==LOW)
  {
     digitalWrite(motor1a, HIGH);
     digitalWrite(motor1b, LOW);
     delay(100);
  }
```

```
   else
  {
    digitalWrite(motor1a, LOW);
    digitalWrite(motor1b, LOW);
    delay(100);
  }

//TODOS LOS CONTADORES REGRESAN AL VALOR CERO
conta=0;
conta1=0;
cont=0;
cont1=0;
Orden=0;
lc.clearDisplay(0);
lc.clearDisplay(1);
lc.clearDisplay(2);
lc.clearDisplay(3);
lc.clearDisplay(4);
lcd.clear();
}}
```

CONCLUSION

El propósito general de este proyecto es diseñar una herramienta para los alumnos de 3° a 6° grado de nivel primaria que coadyuve al aprendizaje, conozca y comprenda las fracciones de una forma más fácil y lúdica.

Así pues, para probar su funcionalidad se llevó a cabo una serie de pruebas del prototipo de material didáctico “Aprendizaje de fracciones” consultando en la escuela primaria “Hermenegildo Galeana”, ubicada en pie de la cuesta, con el grupo de quinto grado, donde se pudo observar que el alumno:

1. Manifestó interés en conocer el prototipo de laboratorio.
2. Les resulto interesante responder las preguntas del maestro y usar el prototipo para colocar las repuestas y a su vez observar que el móvil avanzará con las respuestas correctas.
3. Trabajaron en equipo.
4. Obtuvieron un aprendizaje significativo.
5. Despertó el interés tanto en alumnos y maestros.

El personal docente, tuvo una respuesta positiva al conocer el prototipo, ya que les pareció una herramienta novedosa para aplicar el tema de fracciones.

Para finalizar se plantea las posibles mejoras que puede tener este prototipo de material interactivo de fracciones, en base a las observaciones arriba mencionadas que se hicieron durante las pruebas con los alumnos, como continuación natural del trabajo desarrollado en esta tesis.

Una línea futura inmediata pudiese ser las siguientes:

1. Desarrollar de forma más compacta y ligera el prototipo interactivo, para una mejor movilidad.
2. Utilizarlo como herramienta didáctica para conocer y contar los números del 1 al 100 en 1ro y 2do grado de nivel primaria.
3. Otra funcionalidad futurista para los grados superiores de 3er a 6to nivel primaria es la resolución operaciones básicas suma, resta, multiplicación y división de números enteros.

Por tanto, este prototipo interactivo está diseñado para hacer más amigable el aprendizaje en alumnos de nivel primaria y que el docente posea una herramienta practica para desarrollar sus temas de matemáticas e impartir una clase de forma más interactiva.

REFERENCIAS

Maxinez David G., Alcalá Jessica: Diseño de Sistemas Embebidos a través del
Lenguaje de Descripción en Hardware VHDL. XIX Congreso Internacional
Académico de Ingeniería.

Floyd T. L.: Fundamentos de Sistemas Digitales. Prentice Hall, 1998.

Organización Mundial de VHDL: www.vhdl.org

Randolph H.: Applications of VHDL to Circuit Design. Kluwer Academic Publisher.

C. Kloos, E. Cerny: Hardware Description Language and their Aplications.
Specification, modelling, verification and synthesis of microelectronic
systems. Chapman &Hall, 1997.

E J. Ashenden: The Designer's guide to VHDL. Morgan Kauffman
Publishers, Inc., 1995.

Minibloq + Arduino. Utilización del entorno de programación Minibloq
para programar la Tarjeta Arduino
https://forum.arduino.cc/index.php?topic=233873.0

Mis Proyectos con Arduino, Duemilanove Atmega 328P-PU.
2ª Versión. Actualizada. 20/08/2017, José Miguel Castillo Castillo.

Proserquisa de c.v., Equipo de laboratorio didáctico,
“Excelencia en la experimentación científica”, Curso de Arduino
Lección 1, http://cursoarduino.proserquisa.com/

Arduino y el internet de las cosas, Jonny Novillo-Vicuña, Dixys Hernández -Rojas,

Bertha Mazón-Olivo, Jimmy Molina Ríos, Oscar Cárdenas Villavicencio.
https://www.3ciencias.com/wp-content/uploads/2018/10/ARDUINO-Y-EL-INTERNET-DE-LAS-COSAS.pdf

Las matemáticas. Obtenido de
https://es.wikipedia.org/wiki/Matem%C3%A1ticas

Fracciones. Obtenido de
https://www.portaleducativo.net/sexto-basico/764/que-es-una-fraccion

Tipos de fracciones. Obtenido de
https://www.portaleducativo.net/sexto-basico/765/tipos-de-fracciones
https://www.portaleducativo.net/quinto-basico/532/Tipos-fracciones-fraccion-propia-fraccion-impropia-numero-mixto

Materiales educativos. Obtenido de
https://www.educacioninicial.com/c/001/078-materiales-educativos/
https://educacion.gob.ec/tips-de-uso/
https://issuu.com/edisue/docs/gcompris_tutorial
https://www.ecured.cu/Childsplay
https://pr.wellindal.com/bebes-ninos/juinsa/juego-madera-formas-colores-
numeros
https://alcodistribuciones.com/bloques-madera--actividades-oops-21x15/601343
https://www.amazon.es/Engelhart-Juegos-did%C3%A1cticos-madera-reproducci%C3%B3n/dp/B076JHD7SX
https://www.amazon.es/JUINSA-madera-puzzle-encajes-12910/dp/B074WHLL9R

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Capítulo 1

Capítulo 2

Capítulo 3

Capítulo 4