

2016

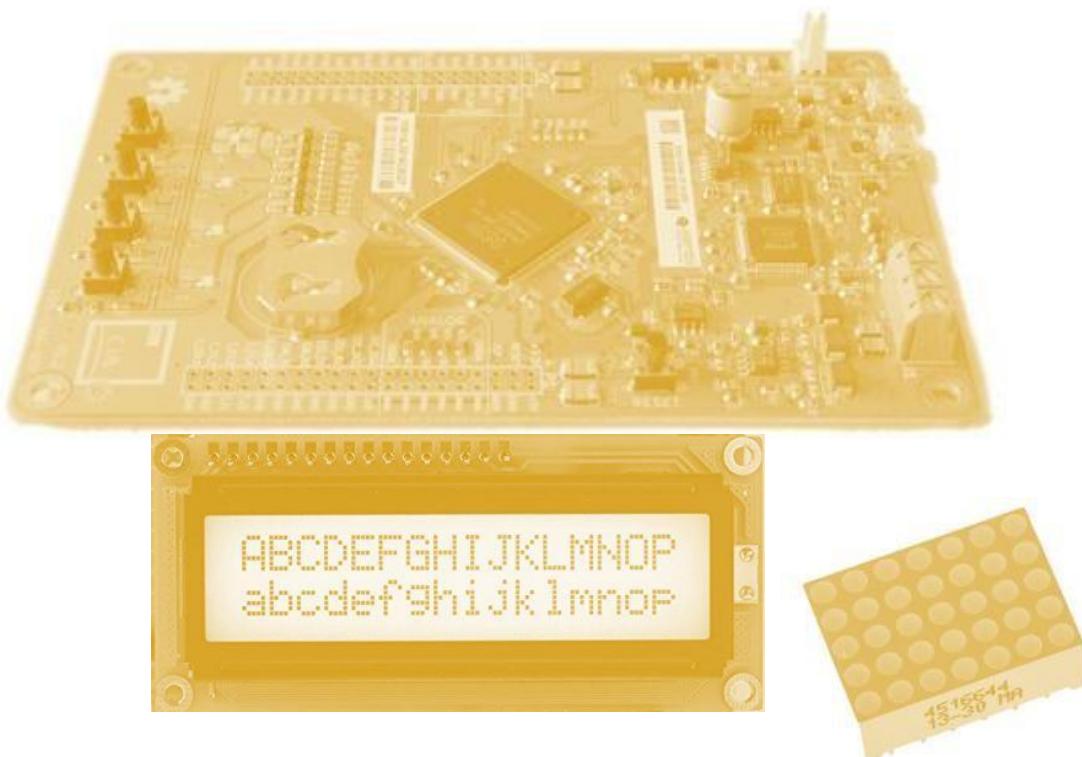
Taller de Proyecto I

Ingeniería en Computación

Facultad de Ingeniería | UNLP

Grupo 8

Jourdón, Julián (424/7)
Maicá, Juan Manuel (436/2)
Lopez Acuña, Axel (248/9)
Scorza, Facundo Ricardo (615/3)



[INFORME FINAL]

CONTROLADOR DE CARTEL

LED

Proyecto Github: www.github.com/csraxl/taller1-g8

Índice

1. Introducción.....	1
1.1. Objetivos del proyecto	1
1.2. Requerimientos del sistema.....	1
1.2.1. Requerimientos funcionales.....	1
1.2.2. Requerimientos no funcionales	2
2. Presupuesto	3
3. Diseño de Hardware	4
3.1. Consideraciones iniciales	4
3.2. Puertos de E/S.....	4
3.3. Alimentación de los dispositivos	5
3.4. Circuito integrado 74HC595	5
3.5. Display LCD de 16x2	6
3.6. Circuito integrado ULN2803	8
3.7. Matriz de LEDs TC07-11EWA.....	9
4. Diseño del esquemático y el PCB	11
4.1. Consideraciones iniciales	11
4.2. Diseño del esquemático	12
4.2.1. Raíz	12
4.2.2. Conector del poncho.....	12
4.2.3. Circuito principal	12
4.2.4. Cartel LED	12
4.3. Diseño del PCB.....	13
5. Diseño de Software	15
5.1. Consideraciones iniciales	15
5.2. Interfaz de usuario y máquina de estados	16
5.3. Sistema operativo y tareas.....	25
5.4. Modularización	30
6. Observaciones finales	32
7. Apéndice	33

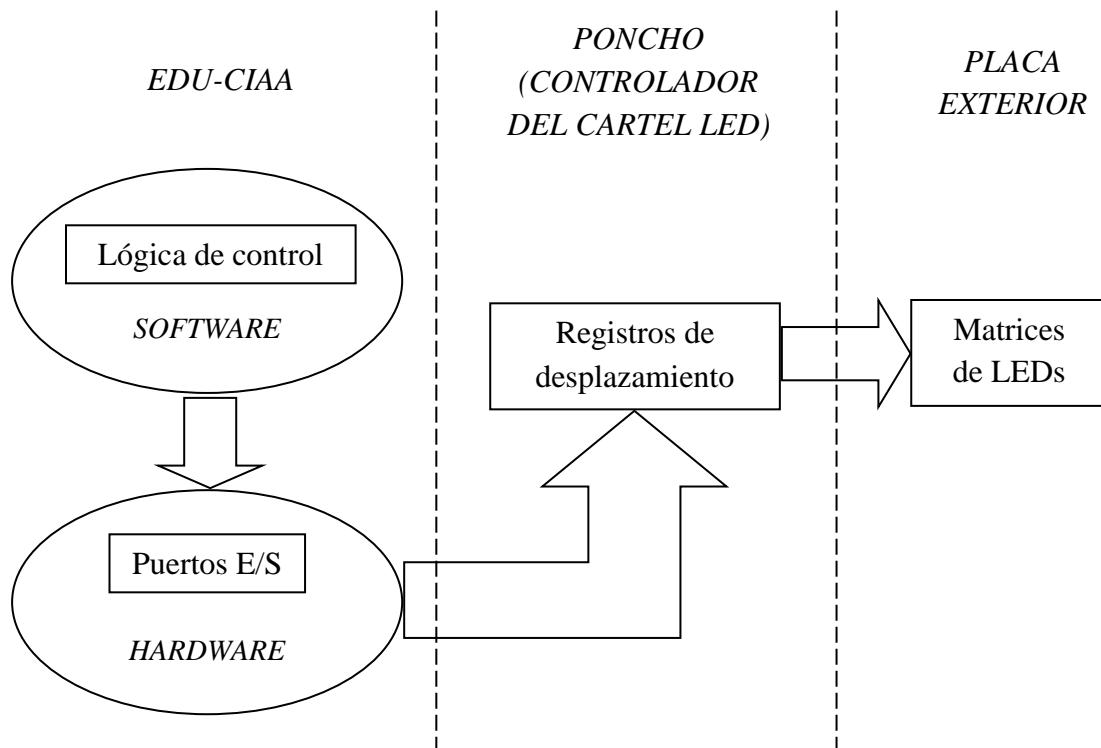
1. Introducción

1.1. Objetivos del proyecto

Nuestro proyecto consiste en desarrollar un controlador para un cartel LED que podría estar conformado por una o varias matrices de LEDs acopladas e interconectadas entre sí. El controlador contará con una lógica tal que podrá mostrar en el cartel cualquier tipo de mensaje y lo desplazará a través de las matrices para que el usuario pueda verlo en su totalidad.

Como podrá intuirse, tenemos que diseñar dos placas para implementar nuestra idea: por un lado estaría el poncho propiamente dicho, el cual sería el “corazón” de nuestro proyecto, en el cual se incluirían todas las conexiones y componentes que necesitaría el controlador para funcionar correctamente y, por otro, una placa adicional exterior que alojaría el cartel LED que utilizaríamos para ser controlado por el poncho y mostrar el sistema en funcionamiento.

A continuación se incluye un esquema inicial de la disposición de todos los componentes principales, tanto físicos (hardware) como lógicos (software) que constituyen nuestro proyecto:



1.2. Requerimientos del sistema

1.2.1. Requerimientos funcionales

A continuación se describen los requerimientos funcionales del sistema, los cuales indican cómo debe comportarse el mismo ante determinados estímulos y son independientes

de la implementación de la solución:

- Funciones principales del sistema:
 - ❖ Ingresar directamente un mensaje para mostrar en el cartel LED.
 - ❖ Realizar una operación aritmética (suma, resta, multiplicación o división) y mostrar el resultado en el cartel LED.
 - ❖ Llevar a cabo una conversión entre sistemas de numeración (decimal, binario o hexadecimal) y mostrar el resultado en el cartel LED.
- Interacción del usuario con el cartel LED:
 - ❖ Cambiar el tamaño de los caracteres del mensaje.
 - ❖ Cambiar la velocidad de desplazamiento del mensaje.
 - ❖ Detener el desplazamiento del mensaje:
 - ❖ Encontrándose el mensaje detenido, retrocederlo, es decir, desplazarlo de izquierda a derecha.

1.2.2. Requerimientos no funcionales

A continuación se describen los requerimientos no funcionales del sistema, los cuales indican restricciones sobre el sistema, impuestas tanto por la cátedra como por nosotros mismos, que limitan nuestras elecciones en la construcción de una solución al problema:

- Interacción del usuario con el sistema:
 - ❖ El sistema deberá interactuar con el usuario permitiéndole ingresar datos e instrucciones mediante la botonera integrada en la EDU-CIAA y mostrándole sus salidas a través de un display LCD de 16x2 (el cual deberá integrarse en el poncho) y, si corresponde, también mediante el cartel LED que está siendo controlado por nuestro poncho.
 - ❖ Interacción del usuario con el display LCD:
 - ✓ Cambiar el carácter a ingresar en el sistema, hacia adelante o hacia atrás.
 - ✓ Avanzar o retroceder el cursor a través del display.
- Dimensiones del cartel LED:
 - ❖ El sistema debe ser capaz de controlar un cartel LED con unas dimensiones de hasta 8 filas por 16 columnas. No obstante, incorporaremos 8 salidas adicionales al controlador que podrán ser utilizadas como filas o columnas, según prefiera el usuario, resultando un cartel de hasta 16 filas por 16 columnas u 8 filas por 24 columnas respectivamente.
- Implementación del poncho:
 - ❖ Su diseño debe ser flexible y escalable de forma tal que el usuario pueda incorporarle hardware y conexiones adicionales que permitan expandirlo para ser compatible con un cartel LED cuyas dimensiones sean superiores a las que soporta nativamente.
- Dimensiones de las matrices de LEDs:
 - ❖ Para poder implementar un cartel LED con las máximas dimensiones toleradas por el sistema, deberán utilizarse matrices de LEDs de 8 filas por 8 columnas. Si no se consiguieran, se las reemplazarán por matrices de 7 filas por 5 columnas, aunque se perderá una parte del potencial ofrecido por el controlador.
- Sistema operativo y lenguaje de programación:

- ❖ El programa deberá ser implementado mediante el lenguaje de programación C, ampliamente utilizado para la programación de sistemas embebidos.
- ❖ El sistema operativo a utilizar tendrá que ser el OSEK-OS, un sistema estático de tiempo real en el que deben definirse de antemano todas las tareas a realizar con su período de activación (si son periódicas).

2. Presupuesto

En este presupuesto simplemente se enunciarán todos los componentes utilizados para la realización del proyecto para que cualquier interesado sepa el costo que le conllevaría llevarlo a cabo. La descripción de cada uno de ellos se realizará en las próximas secciones:

Ítem	Descripción	Cantidad	Costo Unitario (\$)	Costo Total (\$)
1	Placa de desarrollo EDU-CIAA-NXP	1	550	550
2	Tira de 40 pines macho	3	20	60
3	Tira de 40 pines hembra	2	20	40
4	Zócalo para circuito integrado de 2x8	4	4	16
5	Circuito integrado 74HC595	4	15	60
6	Resistencia 1 Ohm	1	0,30	0,30
7	Resistencia 150 Ohm	1	0,30	0,30
8	Resistencia 1000 Ohm	1	0,30	0,30
9	Display LCD 16x2	1	150	150
10	Resistencia 180 Ohm	7	0,30	2,10
11	Zócalo para circuito integrado de 2x9	3	4	12
12	Circuito integrado ULN2803	3	20	60
13	Zócalo para circuito integrado de 2x7	4	4	16
14	Matrid de LEDs T707-11EWA	4	50	200
15	Placa de cobre de 10x15	2	70	140
16	Percloruro Férrico 250cc	1	30	30
Total				1.337

(precios promedio en La Plata a Octubre de 2015)

3. Diseño de Hardware

3.1. Consideraciones iniciales

Una vez planteadas las funcionalidades con las que contaría nuestro sistema, el siguiente paso fue definir qué circuitos electrónicos sería necesario implementar y qué componentes electrónicos utilizaríamos para ello. Basándonos en consultas realizadas tanto con los docentes como en investigaciones realizadas a través de internet, decidimos que la mejor forma de llevar a cabo nuestro proyecto era utilizando un controlador basado en registros de desplazamiento de 8 bits para habilitar y deshabilitar el encendido de los distintos leds de las matrices a partir de la fila y la columna en la que cada uno esté ubicado. Asimismo, para limitar la corriente suministrada a estas últimas e impedir que los integrados utilizados se quemen y se vuelvan inservibles, fue necesario diseñar conexiones adicionales que contemplaran la existencia de resistencias y transistores varios, que se describirán próximamente, aunque se puede anticipar que estos no formarían parte del controlador, sino que estarían directamente conectados a las matrices.

Por otra parte, si bien en el informe inicial se había sugerido la idea de utilizar matrices LED de 8x8 se terminaron usando otras de 7x5 debido a que nos resultó mucho más económico conseguir este tipo de matrices, y en última instancia esto no sería una gran diferencia en el producto final, ya que el principal trabajo de diseño sería el que se realizó sobre la placa controladora mientras que la placa que tuviera estas matrices sería el medio para mostrar que todo el sistema funcionaba correctamente.

3.2. Puertos de E/S

Como se sabrá, la CIAA cuenta con un gran número de puertos de entrada/salida, muchos de ellos susceptibles de ser utilizados para múltiples propósitos. Aunque hay varios cuyos nombres de identificación sugerirían que ya están programados para cumplir una función específica (por ejemplo, LCD1, LCD2, LCD3 y LCD4 para los datos del display LCD) hemos podido comprobar que todos requieren algún tipo de configuración previa por parte del programador para habilitarse.

Sin más preámbulos, procederemos a indicar qué puertos de la CIAA hemos utilizado para nuestro proyecto y qué uso particular les hemos conferido:

Puerto E/S	Funcionalidad
GPIO2	Reloj de los registros de desplazamiento encargados de controlar el encendido de las filas del cartel led
GPIO3	Latch de los registros de desplazamiento encargados de controlar el encendido tanto de las filas como de las columnas del cartel led
GPIO5	Datos de entrada del registro de desplazamiento encargado de controlar el encendido de las primeras ocho filas del cartel led
GPIO7	Datos de entrada del registro de desplazamiento encargado de controlar el encendido de las primeras ocho columnas del cartel led
GPIO8	Reloj de los registros de desplazamiento encargados de controlar el encendido de las columnas del cartel led

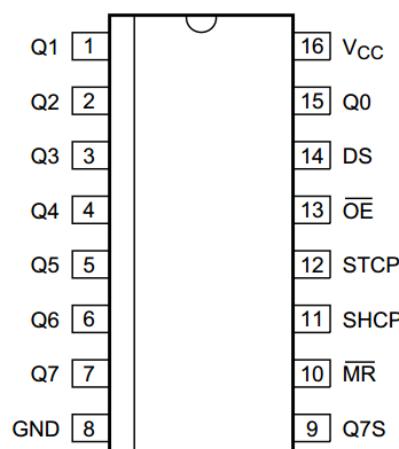
LCD1	Datos e instrucciones para el display LCD (pines 8, 9, 10 y 11 del display respectivamente)
LCD2	
LCD3	
LCD4	
LCD_EN	Habilitar el display LCD (pin 6 del display)
LCD_RS	Indicar al display LCD si se está transmitiendo un dato o una instrucción (pin 4 del display)

3.3. Alimentación de los dispositivos

En nuestro diseño, la idea es alimentar la placa controladora a partir de la alimentación provista por la CIAA y permitir que la placa externa que contenga las matrices puede ser alimentada también por esta tensión o externamente añadiendo transistores al diseño de la misma. Para nuestra implementación nuestra idea es no utilizar ninguna fuente de alimentación externa, es decir, hacer uso exclusivamente de las tensiones de alimentación proporcionadas por la CIAA. Afortunadamente, nuestro proyecto es compatible con dicha idea, ya que todos los dispositivos que utilizamos funcionan correctamente con una alimentación de 5 Voltios la cual, combinada con cualquiera de las conexiones a tierra que ofrece la CIAA, nos permite mantener un diseño sencillo que no requiera un lugar adicional para incorporar fuentes externas.

3.4. Circuito integrado 74HC595

El circuito integrado 74HC595 es, en otras palabras, el registro de desplazamiento de 8 bits de entrada serie y salida paralela o serie al que ya hemos estado haciendo referencia anteriormente, pero que ahora describiremos con mayor detenimiento. A continuación se incluye un esquema de todos sus pines, cuyas funcionalidades describiremos de uno en uno:

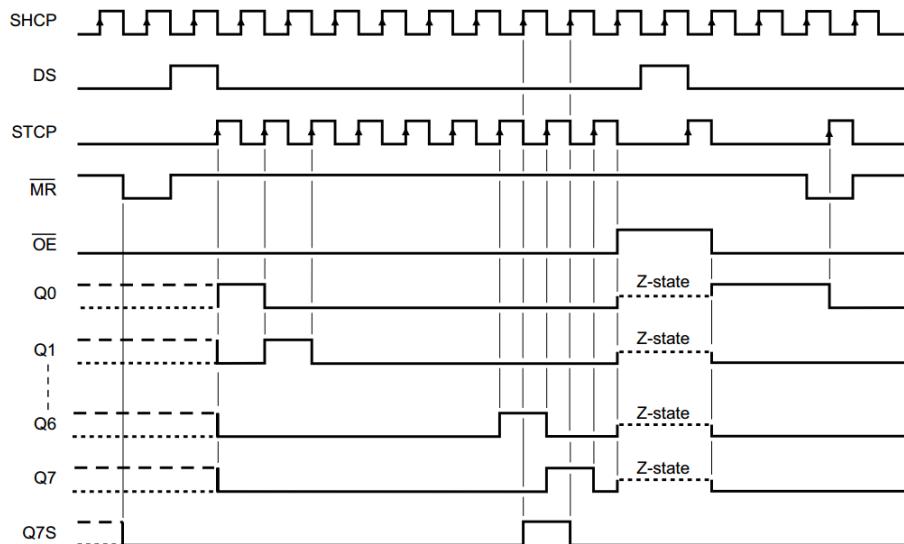


Pin	Funcionalidad
Q0, Q1, Q2, Q3, Q4, Q5, Q6 y Q7	8 bits de datos de salida paralela, del 0 al 7 respectivamente
Q7S	Datos de salida serie
V _{CC}	Conexión a la fuente de alimentación (5V)

GND	Conexión a tierra
DS	Datos de entrada serie
SHCP	Reloj del registro de desplazamiento (flanco ascendente)
STCP	Reloj del registro de almacenamiento (latch, flanco ascendente)
MR y OE	Reinicio y habilitación de las salidas respectivamente (activos en bajo)

Observando la tabla, es sencillo deducir que los pines MR y OE estarán permanentemente conectados a 5V y tierra respectivamente.

Este integrado está conformado por dos registros: uno que es el de desplazamiento propiamente dicho y otro de almacenamiento que guarda las salidas del registro de desplazamiento hasta que el usuario actualice sus salidas para que coincidan con las que tiene guardadas. Cada flanco ascendente que se produce en el reloj SHCP provoca un desplazamiento en cada uno de los bits del registro de desplazamiento, cuyas salidas se almacenan en el registro de almacenamiento. Sin embargo, las salidas del circuito integrado aún no reflejan dicho desplazamiento, sólo lo harán cuando se produzca un flanco ascendente en el reloj STCP. Para una mayor comprensión, se incluye una imagen con las variaciones en cada una de las señales a través del tiempo, obtenida de la hoja de datos del circuito:



3.5. Display LCD de 16x2

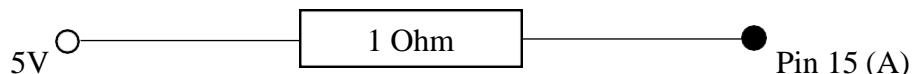
Como este dispositivo no tiene una participación directa en el control del cartel led, la descripción ofrecida del mismo será más breve. A continuación se incluye un esquema de todos sus pines:

15	16	1	2	3	4	5	6	7	8	9	10	11	12	13	14
----	----	---	---	---	---	---	---	---	---	---	----	----	----	----	----

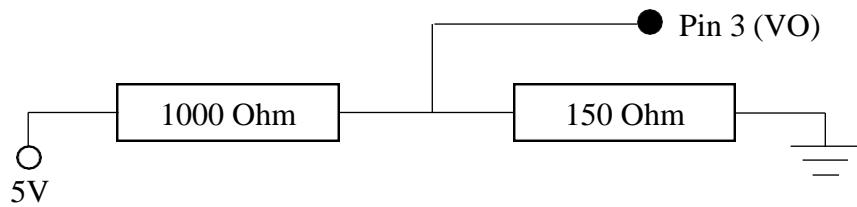
Pin	Símbolo	Descripción
1	V _{SS}	Conexión a tierra
2	V _{DD}	Conexión a la fuente de alimentación (5V)
3	VO	Contraste de la pantalla del display
4	RS	5V → Dato 0V → Instrucción
5	R/W	5V → Leer desde el display 0V → Escribir en el display
6	E	Habilitar el display
7	DB0	Bit 0 de datos
8	DB1	Bit 1 de datos
9	DB2	Bit 2 de datos
10	DB3	Bit 3 de datos
11	DB4	Bit 4 de datos
12	DB5	Bit 5 de datos
13	DB6	Bit 6 de datos
14	DB7	Bit 7 de datos
15	A	Iluminación de la pantalla del display (LED +)
16	K	Iluminación de la pantalla del display (LED -)

Como el display siempre se va a usar para escribir en él, el pin 5 permanecerá conectado a tierra. Para que el LCD funcione correctamente, deben realizarse conexiones especiales en algunos pines:

- Pin 15 (A) → Para limitar la corriente que ingresa al display e ilumina su pantalla, se debe incluir una resistencia en la conexión entre la fuente de alimentación de 5V y el pin 15. Con el objetivo de que la pantalla permanezca con una iluminación máxima elegimos la resistencia más baja posible, es decir, de 1 Ohm:

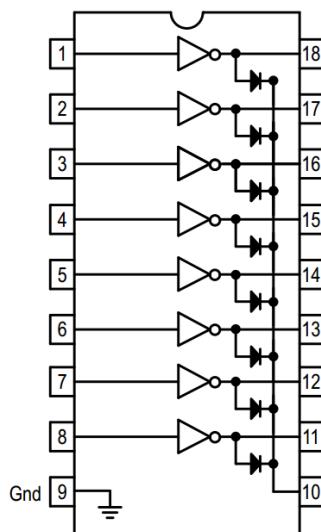


- Pin 3 (VO) → Para ajustar el contraste de la pantalla del display existen dos alternativas: incluir un potenciómetro en una conexión entre la fuente de 5V y tierra, conectado su salida al pin 3 y modificando el contraste manualmente mientras el display está funcionando; o conectar el pin a un divisor de tensión cuyo voltaje de salida sea un valor fijo, pero correcto para mantener un contraste adecuado de manera tal que no necesite ser modificado. Nosotros elegimos esta última alternativa, para la cual probamos distintas combinaciones de resistencias hasta encontrar una que encontramos satisfactoria:



3.6. Circuito integrado ULN2803

El circuito integrado ULN2803 es un conjunto de ocho transistores Darlington, cada uno de los cuales se utiliza como interruptor para activar y desactivar la carga a la que su salida está conectada. A continuación se incluye un esquema de todos sus pines:

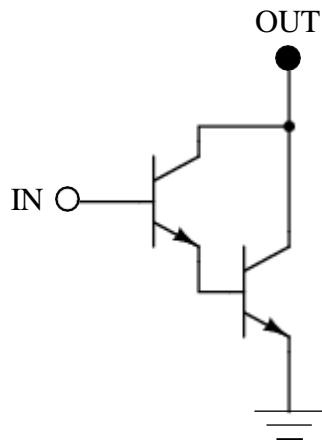


Pin	Funcionalidad
1 a 8	Entradas para cada uno de los ocho transistores: 5V → Activa la carga a la que su respectiva salida está conectada 0V → Desactiva la carga a la que su respectiva salida está conectada
9	Conexión a tierra
10	Conexión común de protección frente a cargas inductivas
11 a 18	Salidas para cada uno de los ocho transistores. A ellas se conectan las cargas.

Sólo se le debe prestar atención al pin 10 si alguna de las cargas conectadas al circuito presenta inductores, en cuyo caso se debe conectar al pin 10 a la fuente de tensión más alta que esté alimentando a las cargas. Como en nuestro proyecto las cargas conectadas no son inductivas (las matrices led en ningún momento involucran inductores), lo podríamos dejar desconectado.

Para comprender mejor el funcionamiento eléctrico del ULN2803 se debe comenzar por explicar qué es un transistor Darlington: se trata de dos transistores NPN que actúan como un único transistor que provee una alta ganancia de corriente. Dentro del par de transistores la corriente amplificada por el primer transistor vuelve a amplificarse en el

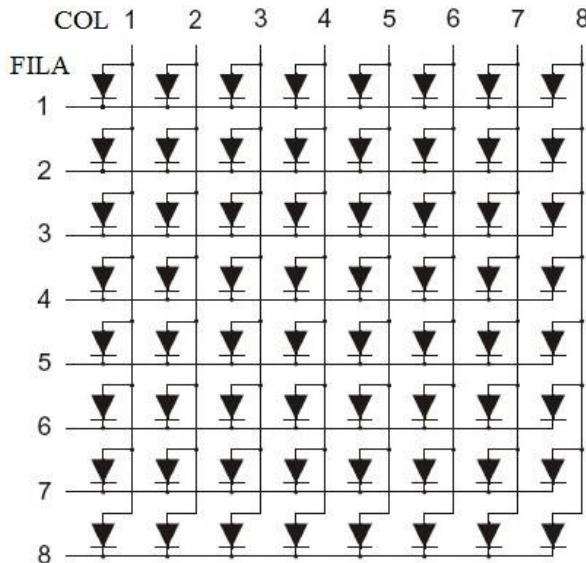
segundo, proveyendo una muy alta corriente a la salida. Por eso el ULN2803 suele utilizarse para controlar cargas de alta potencia: la mayoría de los chips operan con señales de bajo nivel como TTL y CMOS, que operan en un rango de entre 0 y 5V y son capaces de controlar cargas inductivas de alta potencia. Sin embargo, este chip toma las mismas señales TTL de bajo nivel y las usa para activar o desactivar las cargas de alta tensión que están conectadas a las salidas. Los transistores Darlington contenidos en el circuito proveen la amplificación de corriente requerida por las cargas. Éste es el esquema de un transistor Darlington:



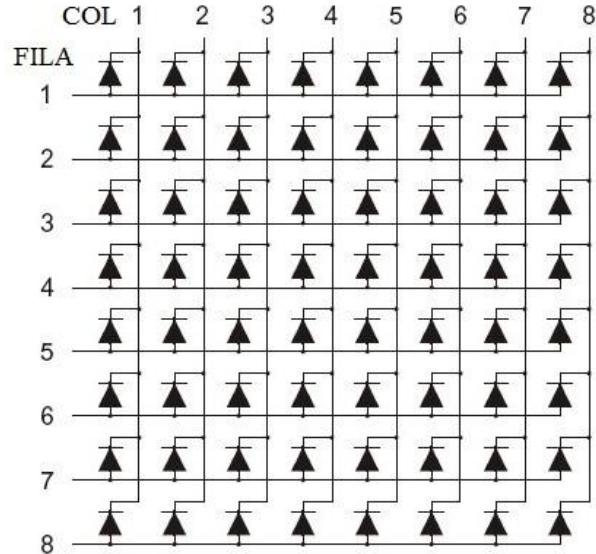
Cuando la entrada es de 0V, no habrá corriente de base por lo que el transistor quedará inactivo y la carga que está conectada a la salida, al no contar con ninguna conexión a tierra, permanecerá desconectada. En cambio, cuando la entrada es de 5V los dos transistores empiezan a conducir corriente, otorgándole a la carga un camino a tierra y activándola. Por lo tanto, cuando se aplica una entrada no nula su correspondiente pin de salida cae a 0V, permitiéndole a la carga conectada que complete su conexión a tierra.

3.7. Matriz de LEDs TC07-11EWA

Éste es el componente central de nuestro trabajo ya que conforma el cartel que queremos controlar. Existen distintas disposiciones (8x8, 7x5, etc.) pero sólo repercuten en el número de conexiones que deben realizarse y no en sus características eléctricas (corriente máxima que pueden soportar, etc.). Sin embargo, hay otro criterio de clasificación que sí debe tenerse en cuenta que ataña a la disposición de los diodos LED dentro de la matriz. De este modo, se distinguen dos tipos de matrices:



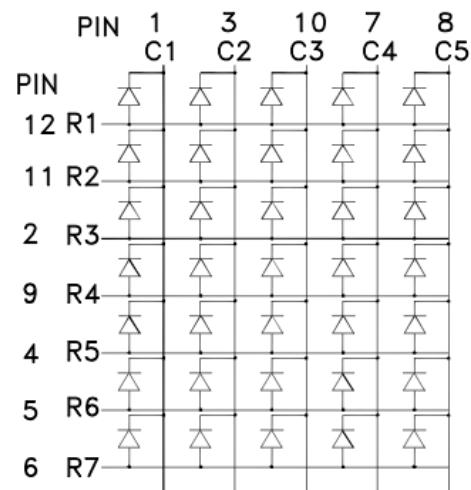
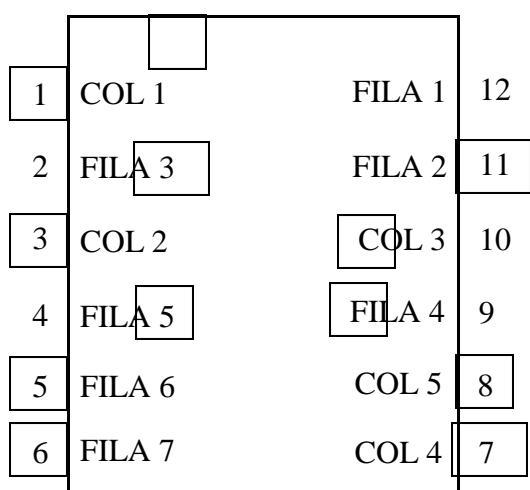
Cátodo Común



Ánodo Común

Este cambio en la disposición de los diodos tiene profundas repercusiones sobre las conexiones eléctricas que deben hacerse para alimentar a las matrices ya que, como se puede ver, para encender un diodo en una matriz cátodo común se debe conectar su fila a tierra y su columna a una tensión positiva. En cambio, en una matriz ánodo común la conexión es inversa: la columna debe conectarse a tierra y la fila, a una tensión positiva. Estas variaciones afectan el posicionamiento de las resistencias necesarias para limitar la corriente de los diodos y de los transistores para activarlos y desactivarlos: una matriz cátodo común tendrá los transistores conectados a sus filas y las resistencias, a sus columnas; mientras que en una matriz ánodo común será al revés. Obsérvese en el presupuesto que los circuitos integrados ULN2803 son bastante más costosos que las resistencias, por lo que conviene tener en cuenta cuáles serán las dimensiones del cartel para analizar la disposición de diodos más conveniente a fin de usar la mayor cantidad posible de resistencia y la menor de chips ULN2803.

Una vez explicado esto, puede pasarse a mostrar un esquema de la matriz que vamos a utilizar para nuestro proyecto (modelo TC07-11EWA, disposición 7x5) para obtener algunas conclusiones a partir de él:



Se puede observar fácilmente que se trata de una matriz ánodo común, por lo que las resistencias irán conectadas a sus filas y los transistores, a sus columnas.

Por otra parte, aún queda por determinar la magnitud de dichas resistencias. Para ello, buscamos en la hoja de datos de la matriz cuál era su máxima corriente tolerada, y encontramos que era de 30 mA. Por lo tanto, planteando un pequeño esquema simplificado de la conexión eléctrica de uno de los diodos y haciendo uso de la Ley de Ohm:



$$5V = I_{MÁX} * R_{MÍN} \rightarrow R_{MÍN} = 5V / I_{MÁX} = 5V / 0,03 A = 166,67 \text{ Ohm} \rightarrow R \geq 166,67 \text{ Ohm}$$

Descubrimos que las resistencias comerciales más cercanas a este valor eran de 180 Ohm, por lo que éstas fueron las que adquirimos y resultaron ser adecuadas para nuestro proyecto, ya que ningún diodo se quemó y todos presentaron una fuerte intensidad cuando los encendimos.

4. Diseño del esquemático y el PCB

4.1. Consideraciones iniciales

Dado que desde un principio la idea del proyecto fue que el controlador permitiera escalabilidad tanto desde el punto de vista del diseño, esto es, que si se quisiera agrandar el sistema se lo pueda realizar con facilidad y siguiendo unos lineamientos fáciles de entender e implementar; como desde el punto de vista del producto final, es decir que incluso con el sistema ya en marcha este permitiera extender su funcionalidad fácilmente, el diseño esquemático del sistema se realizó acorde con esta meta. Para ello se planteó inicialmente un sistema mínimo que permitiera controlar un arreglo de matrices LED de 8x8 puntos dispuestas tanto en forma rectangular como cuadrada, es decir que este sistema mínimo permite controlar matrices con una disposición de 1x3 (8x24 puntos LED) y una de 2x2 (16x16 puntos LED).

Una vez decidido este sistema mínimo se pensó en la separación en capas de hardware que se debería realizar para que esta escalabilidad sea posible. Por ello se decidió implementar una capa que tuviera todos los elementos de hardware necesarios para el control de las matrices (los registros de desplazamiento y el display LCD), y una segunda capa que tuviera el cartel LED propiamente dicho, incluyendo tanto las matrices de leds como las resistencias para limitar la corriente que circulaba a través de ellas y los transistores para que actuaran como interruptores, habilitando y deshabilitando cada uno de los leds según correspondiera.

De esta manera, siendo la capa de control independiente de la capa de las matrices se hace posible que la primera se pueda utilizar sin cambios con varias disposiciones de arreglos de matrices, e incluso con arreglos de matrices de distintas dimensiones o con una diferente disposición interna de sus diodos realizando mínimos ajustes en el software de la CIAA.

También es necesario mencionar las limitaciones del sistema. Dado que para lograr que

el sistema mínimo permita controlar matrices en disposiciones 1x3 y 2x2 se debe realizar un cambio en el hardware de manera externa (a través de jumpers), la expansión del controlador sólo se puede realizar apropiadamente utilizando la disposición inicial de 1x3.

NOTA: Todos los esquemáticos y los diseños utilizados en la fabricación de las placas impresas pueden encontrarse en la sección “Apéndice”, al final del informe.

4.2. Diseño del esquemático

Teniendo en cuenta los aspectos antes mencionados y el método de control elegido el diseño esquemático de la placa controladora se realizó utilizando las plantillas disponibles en un repositorio de Github llamado “Ponchos”, accesible a través de la página oficial del proyecto CIAA. Para ella se usó una plantilla de dimensiones similares a las de la EDU-CIAA.

El esquema jerárquico del esquemático que se utilizó para el proyecto fue el siguiente:

- Raíz
 - ❖ Conector del poncho
 - ❖ Circuito principal

4.2.1. Raíz

Esta hoja está incluida por defecto con la plantilla utilizada del repositorio Github y permite especificar que señales de salida de la EDU-CIAA se utilizaran en el circuito principal del poncho.

4.2.2. Conector del poncho

Esta hoja también está incluida por defecto en la plantilla y contiene la especificación de las salidas con las que cuenta la EDU-CIAA, a través de ella se tienen disponibles las señales que van a hacer utilizadas en las otras hojas del proyecto.

4.2.3. Circuito principal

Esta hoja contiene el diseño de las conexiones realizadas en la placa controladora, tanto las relacionadas con el control de los registros de desplazamiento como del control de la pantalla LCD.

4.2.4. Cartel LED

Para demostrar el correcto funcionamiento de la placa de control, fue imperiosa la necesidad de realizar un diseño adicional para un cartel LED que recibiera las señales de control de los registros de desplazamiento y se comportara de la manera esperada. A tal efecto, basándonos en las matrices de 7x5 con las que contábamos, decidimos utilizar tantas salidas de la placa de control como nos fuera posible para aprovechar todo el potencial ofrecido por el controlador y mejorar la legibilidad del mensaje que se mostrara en el cartel. Además, al hacer uso de un elevado número de registros de desplazamiento, la lógica de control sería más compleja, por lo que quedaría más demostrado el correcto funcionamiento

de nuestro diseño e implementación. Concluimos que la mejor alternativa era utilizar el registro REGX1 para controlar ocho columnas adicionales, por lo que nos quedaron en total un registro (REGF1) para controlar las siete filas de las matrices (la última salida paralela del registro no se utilizaría) y otros tres (REGC1, REGC2, REGX1) para controlar hasta veinticuatro columnas; como cada matriz es de cinco columnas, podrán controlarse hasta veinte columnas, permaneciendo cuatro salidas del registro REGX1 desconectadas. Por lo tanto, el cartel LED resultante fue de 7x20, quedando conformado por cuatro matrices de 7x5 ubicadas una al lado de la otra.

4.3. Diseño del PCB

Respecto de la maquetación general de la placa de control hay que mencionar que los lineamientos generales fueron:

- Mantenerse dentro de las dimensiones de la propia EDU-CIAA.
- Distribuir los integrados de manera tal que el ruteo de las pistas sea circular.
- Ubicar el conector para la pantalla LCD de manera tal que pueda ser leída mientras se utilizan los botones de la EDU-CIAA.

Debido a la densidad de líneas a las que se debía rutear en un primer momento se realizó un diseño a doble faz con el que la distribución de los integrados estaba centrada en la placa y se dejaba a los lados las salidas de estos para facilitar la conexión hacia una placa de salida ubicada externamente a la EDU-CIAA y la placa de control.

Sin embargo, este diagrama inicial se descartó debido a las complicaciones que podría llegar a traer su realización mediante el método casero de transferencia térmica que utilizaríamos. Por ejemplo, la impresión a un lado y al otro de la placa podrían no quedar bien alineadas.

Por ello se volvió a realizar el diseño, esta vez a simple faz, pero utilizando un plano de masa para facilitar las conexiones a tierra de los integrados y añadiendo puentes. De esta manera se solventó el problema de la realización de la placa a doble faz, pero sacrificando la disposición de los integrados y sus salidas que quedaron invertidas, es decir, los integrados hacia los lados de las placas y las salidas de estos en el centro:

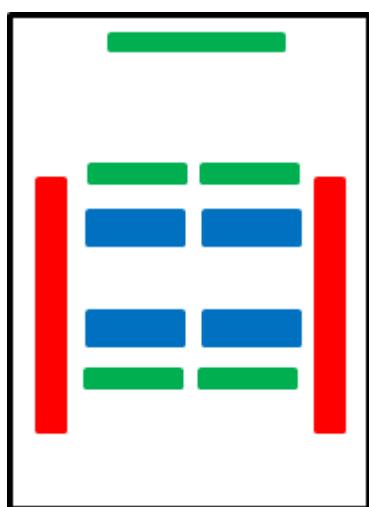


Diagrama inicial

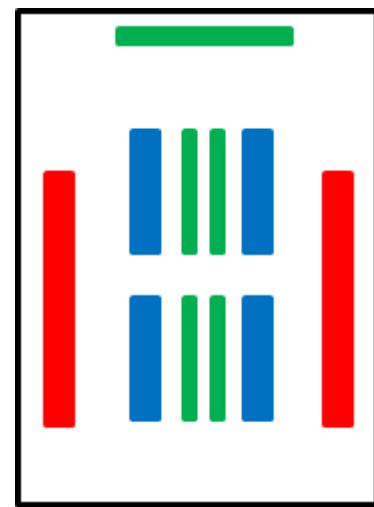


Diagrama final

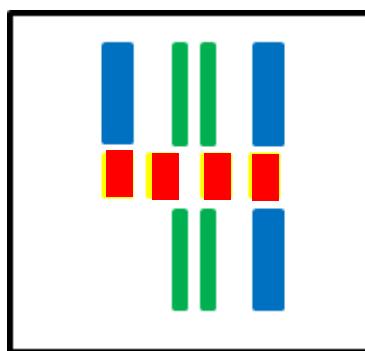
- Referencias:

- ❖ **Conectores hacia la EDU-CIAA.**
- ❖ **Circuitos integrados.**
- ❖ **Tiras de pines de salida.**

En cuanto al diseño de la placa que contiene a las matrices, originalmente la idea fue que esta se conecte por medio de cables a la placa controladora ya que precisamente tiene sentido que quien utilice el sistema pueda estar en un lugar diferente del que se encuentra la placa con el mensaje. Sin embargo, teniendo en cuenta que nuestro proyecto es un prototipo de un sistema real, y por una cuestión de comodidad en el transporte y manipulación del hardware decidimos diseñar la placa de matrices para que se acople a las placas restantes por medio de tiras de pines.

Esta decisión, sumada a la cantidad de líneas que se necesitaban rutear y la disposición de los pines correspondientes a las filas y columnas en las matrices hicieron que sea necesario que el diseño se realice en dos capas. Con lo que no quedó alternativa que lidiar con los problemas de su fabricación casera.

En cuanto a la disposición de los componentes en esta placa se siguieron las mismas consideraciones que para la placa controladora: el tamaño se ajustó para que no molestara a la hora de presionar los botones de la EDU-CIAA y que permitiera leer la pantalla LCD. La ubicación de los integrados se realizó de la manera más cómoda teniendo en cuenta la ubicación de las salidas en la placa controladora, mientras que las matrices se mantuvieron centradas en la placa y con la mínima distancia posible entre ellas para facilitar la lectura:



- Referencias:

- ❖ **Circuitos integrados.**
- ❖ **Tiras de pines de salida.**
- ❖ **Matrices de LEDs.**

5. Diseño de Software

5.1. Consideraciones iniciales

Una vez definidos todos los componentes electrónicos necesarios para desarrollar el proyecto y diseñadas las conexiones entre los mismos, debemos implementar la lógica de control para tomar las decisiones que posibiliten que el sistema posea el funcionamiento esperado.

A tal efecto, podemos comenzar mencionando que para habilitar y deshabilitar las distintas luces del cartel LED utilizaremos una técnica conocida como multiplexación, la cual funciona de la siguiente manera:

- A partir del mensaje que desea mostrar en el cartel, el programa obtiene una matriz “lógica” con la disposición que deberían tener los leds del cartel para que el mismo pueda visualizarse correctamente, asignándole un cero a las posiciones que estarían apagadas y un uno a las que estarían encendidas. Se dice que esta matriz es lógica porque sólo es una abstracción y una simplificación asequible gracias al software, es decir, física y electrónicamente no es ésta la configuración que posee el cartel, aunque el usuario así lo crea cuando la observe en funcionamiento. Este fenómeno se explicará a continuación.
- El programa debe enviar un uno lógico, lo cual se traduce en una tensión positiva, a la entrada serie del registro de desplazamiento encargado de controlar las primeras ocho columnas del cartel led. Como se recordará, el pin de la CIAA elegido para esta función fue el GPIO7, y el registro es el que hemos bautizado como C1_OUT1. Luego se debe enviar un flanco ascendente por el reloj SHCP de dicho registro (asociado al pin GPIO8 de la CIAA) para desplazar el uno lógico enviado hacia el primer bit del registro de desplazamiento. No obstante, recordemos que las salidas del registro no se actualizarán efectivamente hasta enviar un flanco ascendente por el latch STCP, lo cual aún no haremos porque resta por realizarse una última acción: revisar la primera columna de la matriz lógica para determinar a qué filas de la misma se le deben enviar unos lógicos para encender sus leds; estas filas serán todas aquellas cuyo valor en esta columna en particular sea un uno y no un cero. Notemos que, al estar utilizando matrices de ánodo común con resistencias conectadas a sus filas y transistores a sus columnas, se le deberá enviar un uno tanto a la fila como a la columna del led que se quiera encender: la corriente se transmitirá a través de la fila y el transistor cerrará la conexión para permitir el encendido del led. Así, se tiene un control absoluto tanto de las filas como de las columnas y se puede determinar con total precisión qué leds se quieren encender.
- Una vez determinadas las filas de la columna cuyos leds se desean encender, se configura el registro de desplazamiento que controla las filas del cartel (registro F1_OUT1) para que sus salidas envíen simultánea y paralelamente unos lógicos a las filas que correspondan. Para esta configuración, como cabe esperarse, debe hacerse uso de la entrada serie del registro y de su reloj SHCP (pines GPIO5 y GPIO2 de la CIAA respectivamente) para desplazar los unos y los ceros a través de las distintas filas.
- Ahora que tanto la columna como las filas de dicha columna están correctamente configuradas y listas para que sus señales sean transmitidas al cartel, se envía un flanco ascendente al latch STCP (pin GPIO3 de la CIAA), común a todos los registros de

desplazamiento, para que sus salidas se actualicen y finalmente se logre el encendido de los leds que correspondan.

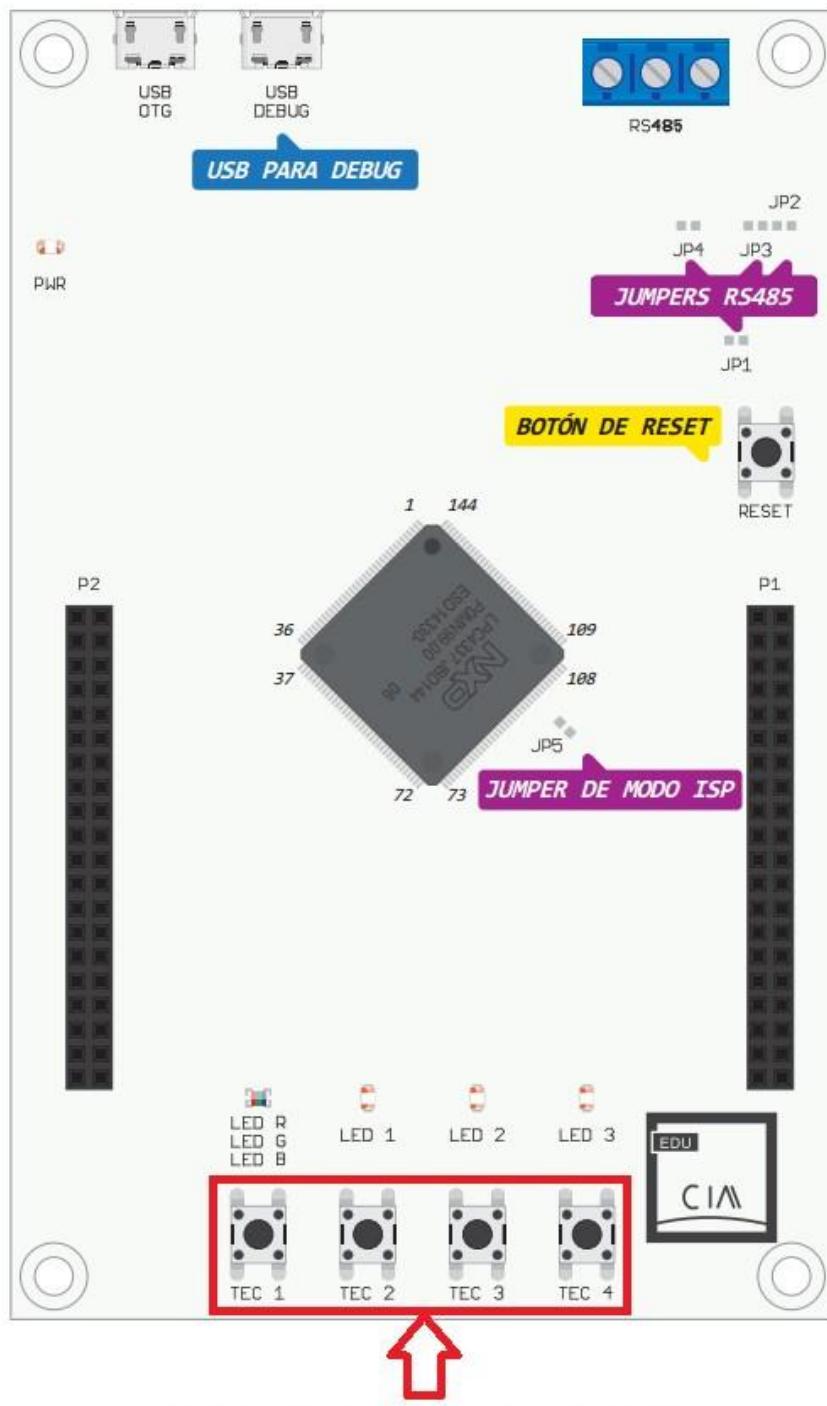
- Nótese que todo este proceso es periódico: una vez terminado, se debe transmitir un flanco ascendente hacia el reloj SHCP de los registros de desplazamiento de las columnas para que el uno lógico transmitido inicialmente hacia el primer bit se desplace hacia el segundo, y todas las etapas se repetirán, con la salvedad de que ahora deberá revisarse la segunda columna de la matriz lógica en vez de la primera para determinar en qué filas se ubican los leds que se deben encender; y así se deberá desplazar este uno a través de todas las columnas y revisarlas una por una, encendiéndo los leds que correspondan (una vez alcanzada la última columna, se regresará a la primera y se volverá a recorrer todo el cartel). Huelga decir que la periodicidad de esta tarea deberá ser lo suficientemente corta como para que el usuario que observa la matriz tenga la ilusión de que todos los leds del cartel se encienden simultáneamente cuando en realidad esto es así: los mismos se encienden de a una columna por vez, aunque a una velocidad imperceptible para él. Al realizarse el recorrido del cartel a través de sus columnas, decimos que esta multiplexación es por columnas. Intentamos realizarla por filas, pero tuvimos el siguiente inconveniente: si se observan las conexiones internas entre los diodos de las matrices, se notará que, si se activan simultáneamente una fila y varias columnas, la corriente que ingresa por la fila se distribuirá a través de las columnas activas, por lo que la corriente que circulará a través de cada diodo será mucho menor. Este motivo justificaba el fenómeno de que, si en algunas filas se activaban más columnas que en otras, sus diodos brillaban con una menor intensidad, y el resultado no era agradable a la vista. En cambio, esto no ocurre con la multiplexación por columnas, porque en todo momento nunca hay más de un led por fila encendido simultáneamente, a lo sumo todos los leds de una misma columna estarán encendidos, pero al tener no influir esto sobre la corriente suministrada a cada diodo, todos los leds siempre brillan con la misma intensidad. El único cuidado que se debe tener es con las dimensiones del cartel: si posee un número demasiado alto de filas, la revisión que el programa hace de todas ellas mientras habilita una columna puede hacerse demasiado larga, y al tener ser el sistema operativo utilizado de tiempo real, tiene una política de scheduling muy rigurosa, lo cual significa que, si una tarea es demasiado larga, el programa se detendrá y devolverá un error de ejecución. Hablaremos de este tema con mayor detalle en el apartado “Sistema operativo y tareas”.

5.2. Interfaz de usuario y máquina de estados

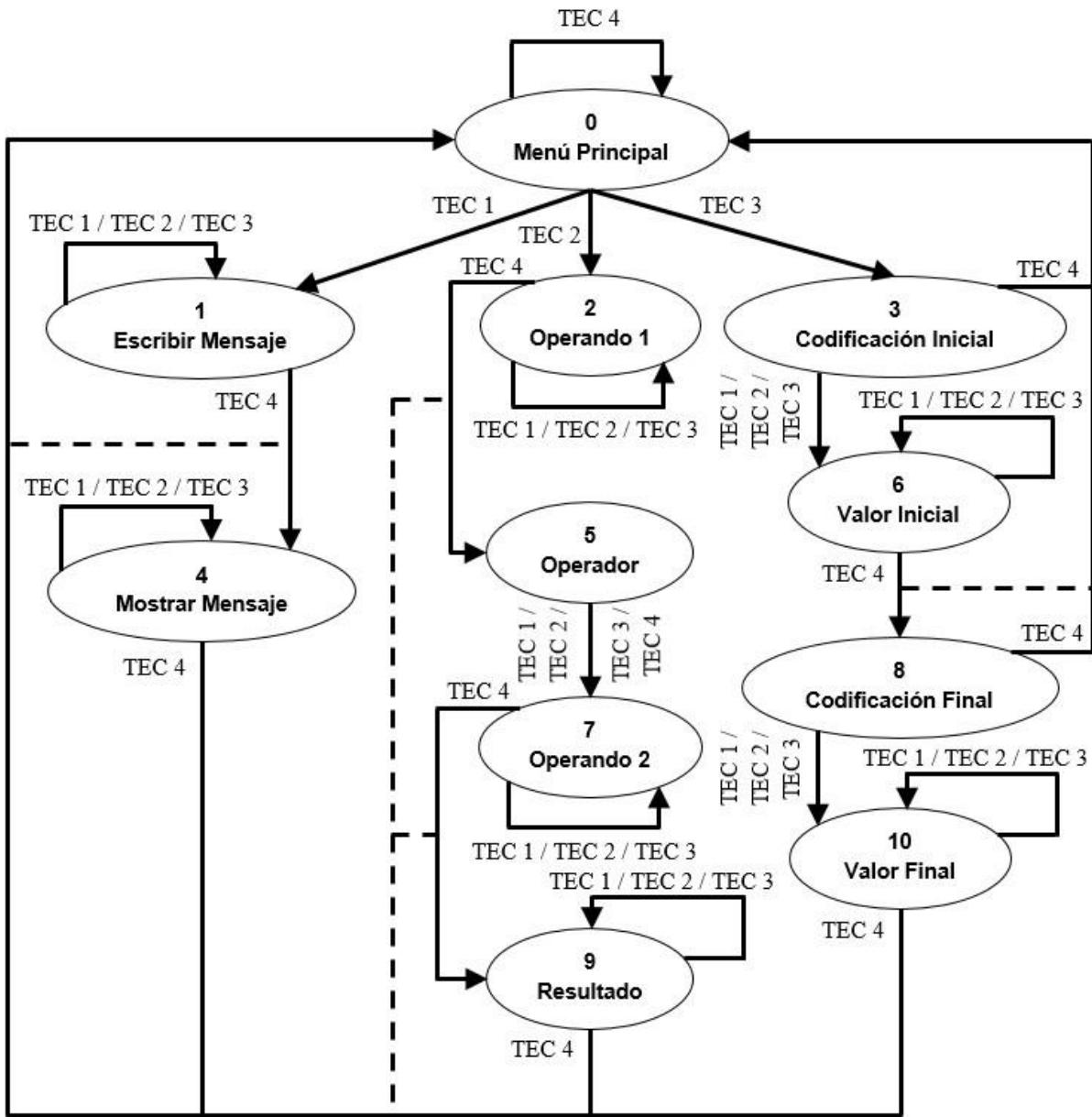
Los medios que hemos elegido para interactuar con el usuario son sencillos y austeros, aunque no por ello menos efectivos: las entradas están constituidas por los interruptores que ya se encuentran integrados en la CIAA (TEC 1, TEC 2, TEC 3 y TEC 4) y las salidas, por el display LCD de 16x2 y el cartel LED de 7x20. Aunque consideramos al cartel como un medio de interacción con el usuario, cabe mencionar que dicha interacción es muy limitada y reducida, ya que el usuario solamente puede realizar con el cartel acciones muy sencillas como cambiar el tamaño de los caracteres y su velocidad de desplazamiento, pero no puede interactuar con él de ninguna manera que provoque cambios “reales” y profundos en el sistema.

Por su parte, luego de analizar exhaustivamente todas las alternativas posibles para resolver el problema de la implementación del software para el sistema, concluimos que la más conveniente era una máquina de estados cuyas entradas fueran las que ingresara el usuario a través de los interruptores y sus salidas dependerían exclusivamente del estado actual de la máquina (máquina de Moore) y serían visibles para el usuario a través del display LCD y también, si es que correspondiera, del cartel LED.

A continuación explicaremos exacta y detalladamente cómo interactuará el usuario con el sistema y cómo dichas interacciones permitirán las transiciones entre estados de la máquina a fin de que el sistema cumpla con las funcionalidades especificadas en la introducción del informe:



Entradas de la interfaz de usuario



Referencias:

- Estado 0 (Menú Principal):
 - ❖ Se le presentará al usuario un menú en el display LCD conformado por dos submenús que se alternarán entre sí con un período de 2 segundos. A partir de ellos podrá elegir cuál de todas las funcionalidades del sistema desea probar. Los submenús tendrán los siguientes formatos:

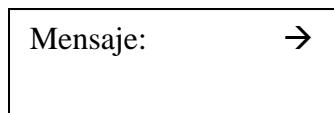
Menú Principal
1. Mensaje

2. Calculadora
3. Conversor

Submenú 1

Submenú 2

- ❖ Si se presiona el interruptor TEC 1, se habrá seleccionado la opción de ingresar un mensaje para mostrar en el cartel LED. Téngase en cuenta que, al ser el display utilizado de 16x2, el mensaje podrá contener a lo sumo 16 caracteres.
- ❖ Si se presiona el interruptor TEC 2, se habrá seleccionado la opción de realizar una operación aritmética y mostrar el resultado en el cartel LED. Téngase en cuenta que la calculadora sólo admite operandos decimales, de 8 bits y sin signo, por lo que su rango es de 0 a 255.
- ❖ Si se presiona el interruptor TEC 3, se habrá seleccionado la opción de realizar una conversión entre dos sistemas de numeración y mostrar el resultado en el cartel LED. Téngase en cuenta que el conversor sólo admite valores de 8 bits y sin signo por lo que su rango será de 0 a 255 (decimales), de 0 a 11111111 (binarios) y de 0 a FF (hexadecimales).
- ❖ Como el interruptor TEC 4 no posee ninguna funcionalidad en este estado, si se lo pulsa no se provocará ningún cambio sobre el sistema.
- Estado 1 (Escribir Mensaje):
 - ❖ El usuario podrá hacer uso de los interruptores para ingresar el mensaje que desea mostrar en el cartel led. El mismo se irá mostrando en el display LCD a medida que el usuario lo ingrese para que compruebe si lo está introduciendo correctamente. Éste sería el formato del display LCD mientras el usuario ingresa su mensaje:

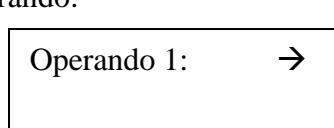


- ❖ El mensaje se iría mostrando en la línea inferior del display a medida que el usuario lo introduzca.
- ❖ Si se presiona el interruptor TEC 1, cambiará el carácter sobre el que esté actualmente posicionado el cursor. Prestar atención a la flecha que se incluye en el display ya que, si apunta hacia la derecha, significa que el cambio se producirá hacia adelante, es decir, si hay una B, se pasará a una C, pero si la flecha apunta hacia la izquierda, el carácter cambiará hacia atrás, es decir, pasará a ser una A.
- ❖ A continuación se muestran los caracteres disponibles para incluir en el mensaje, ubicados según su orden de aparición al utilizar el interruptor TEC 1:

Espacio	A	B	C	D	E	F	G	H	I	J	K	L	M	N
O	P	Q	R	S	T	U	V	W	X	Y	Z	+	-	*
/	=	,	0	1	2	3	4	5	6	7	8	9		

- ❖ Si se presiona el interruptor TEC 2, se desplazará el cursor a través de la línea inferior del display, en la que se está escribiendo el mensaje. Prestar atención a la flecha que se incluye en el display ya que, si apunta hacia la derecha, significa que el desplazamiento se producirá hacia adelante (hacia la derecha), pero si la flecha apunta hacia la izquierda, el mismo será hacia atrás (hacia la izquierda).

- ❖ Si se presiona el interruptor TEC 3, cambiará la orientación de la flecha del display, lo cual invertirá el sentido tanto del cambio de los caracteres como del desplazamiento del cursor.
- ❖ Al presionar el interruptor TEC 4, si el primer y/o el último carácter del mensaje es un espacio, el sistema lo rechazará y le volverá a mostrar al usuario el menú principal (estado 0).
- Estado 2 (Operando 1):
 - ❖ El usuario podrá hacer uso de los interruptores para ingresar el primer operando de la operación aritmética que desea realizar. El operando se irá mostrando en el display LCD a medida que el usuario lo ingrese para que compruebe si lo está introduciendo correctamente. Éste sería el formato del display LCD mientras el usuario ingresa el operando:



- ❖ El operando se iría mostrando en la línea inferior del display a medida que el usuario lo introduzca.
- ❖ Si se presiona el interruptor TEC 1, cambiará el carácter sobre el que esté actualmente posicionado el cursor. Prestar atención a la flecha que se incluye en el display ya que, si apunta hacia la derecha, significa que el cambio se producirá hacia adelante, es decir, si hay un 2, se pasará a un 3, pero si la flecha apunta hacia la izquierda, el carácter cambiará hacia atrás, es decir, pasará a ser un 1.
- ❖ A continuación se muestran los caracteres disponibles para ingresar el operando, ubicados según su orden de aparición al utilizar el interruptor TEC 1:

Espacio	0	1	2	3	4	5	6	7	8	9
---------	---	---	---	---	---	---	---	---	---	---

- ❖ Si se presiona el interruptor TEC 2, se desplazará el cursor a través de la línea inferior del display, en la que se está escribiendo el mensaje. Prestar atención a la flecha que se incluye en el display ya que, si apunta hacia la derecha, significa que el desplazamiento se producirá hacia adelante (hacia la derecha), pero si la flecha apunta hacia la izquierda, el mismo será hacia atrás (hacia la izquierda).
- ❖ Si se presiona el interruptor TEC 3, cambiará la orientación de la flecha del display, lo cual invertirá el sentido tanto del cambio de los caracteres como del desplazamiento del cursor.
- ❖ Al presionar el interruptor TEC 4, si alguno de los caracteres del operando es un espacio o el operando no está incluido dentro del rango de la calculadora, el sistema lo rechazará y le volverá a mostrar al usuario el menú principal (estado 0).

- Estado 3 (Codificación Inicial):
 - ❖ Se le presentará al usuario un menú en el display LCD conformado por dos submenús que se alternarán entre sí con un período de 2 segundos. A partir de ellos podrá elegir el sistema de numeración a partir del cual desea realizar la conversión. Los submenús tendrán los siguientes formatos:

Codif. inicial:
1. Binaria

2. Decimal
3. Hexadecimal

Submenú 1

Submenú 2

- ❖ El pulsado de los interruptores TEC 1, TEC 2 y TEC 3 le indicará al sistema que el usuario desea hacer una conversión a partir de un valor binario, decimal y hexadecimal respectivamente.
- ❖ Si se presiona el TEC 4, se cancelará la conversión y se le volverá a mostrar al usuario el menú principal (estado 0).
- Estado 4 (Mostrar Mensaje):
 - ❖ Se procederá a mostrarle al usuario en el cartel LED el mensaje ingresado. El mismo se desplazará a través del cartel de derecha a izquierda para que la persona pueda visualizarlo en su totalidad.
 - ❖ El formato del display LCD será idéntico al del estado 1, con la salvedad de que el mensaje permanecerá “congelado” en él, imposibilitándole al usuario cualquier posibilidad de modificarlo o interactuar con él.
 - ❖ Ahora el usuario podrá hacer uso de los interruptores TEC para interactuar con el cartel en vez del display. Sus interacciones podrán ser las siguientes:
 - TEC 1: Cambiar el tamaño de los caracteres mostrados en el cartel. Si son grandes, se achicarán, y si son pequeños, se agrandarán.
 - TEC 2: Modificar la velocidad de desplazamiento del mensaje a través del cartel. Hay tres velocidades: lenta, normal y rápida. Inicialmente la velocidad es la normal, presionando el interruptor se pasará a la rápida y, si se lo vuelve a pulsar, a la lenta, para finalmente regresar a la normal y así sucesivamente.
 - TEC 3: Detener el desplazamiento del mensaje a través del cartel. Ahora el mismo quedará “congelado” en él, pero no se borrará. Estando el mensaje detenido, si se presiona el interruptor TEC 2 se lo podrá retroceder una columna por cada vez que se lo presione.
 - TEC 4: Se borra el mensaje del cartel y se le vuelve a mostrar al usuario el menú principal en el display LCD (estado 0).
- Estado 5 (Operador):
 - ❖ Se le presentará al usuario un menú en el display LCD mediante el cual podrá elegir cuál es el operador que desea aplicar entre los operandos de su operación aritmética. Su formato será el siguiente:

Operador:
1. + 2. - 3. * 4. /

- ❖ El pulsado de los interruptores TEC 1, TEC 2, TEC 3 y TEC 4 le indicará al sistema que el usuario desea hacer una suma, una resta, una multiplicación y una división respectivamente.

- ❖ Éste es el único estado en el cual el usuario no cuenta con ningún medio para regresar al menú principal (estado 0).
- Estado 6 (Valor Inicial):
 - ❖ El usuario podrá hacer uso de los interruptores para ingresar el valor a partir del cual desea realizar la conversión, en el sistema de numeración especificado previamente en el estado 3. El valor se irá mostrando en el display LCD a medida que el usuario lo ingrese para que compruebe si lo está introduciendo correctamente. Éste sería el formato del display LCD mientras el usuario ingresa el valor de conversión:

Valor inicial:	→
----------------	---

 - ❖ El valor se iría mostrando en la línea inferior del display a medida que el usuario lo introduzca.
 - ❖ Si se presiona el interruptor TEC 1, cambiará el carácter sobre el que esté actualmente posicionado el cursor. Prestar atención a la flecha que se incluye en el display ya que, si apunta hacia la derecha, significa que el cambio se producirá hacia adelante, es decir, si hay un 0, se pasará a un 1, pero si la flecha apunta hacia la izquierda, el carácter cambiará hacia atrás, es decir, pasará a ser un espacio.
 - ❖ A continuación se muestran los caracteres disponibles para ingresar el valor de conversión, ubicados según su orden de aparición al utilizar el interruptor TEC 1. Téngase en cuenta que los mismos variarán en función del sistema de numeración elegido para el valor:
 - Valor binario:

Espacio	0	1
---------	---	---

 - Valor decimal:

Espacio	0	1	2	3	4	5	6	7	8	9
---------	---	---	---	---	---	---	---	---	---	---

 - Valor hexadecimal:

Espacio	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
---------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---
 - ❖ Si se presiona el interruptor TEC 2, se desplazará el cursor a través de la línea inferior del display, en la que se está escribiendo el mensaje. Prestar atención a la flecha que se incluye en el display ya que, si apunta hacia la derecha, significa que el desplazamiento se producirá hacia adelante (hacia la derecha), pero si la flecha apunta hacia la izquierda, el mismo será hacia atrás (hacia la izquierda).
 - ❖ Si se presiona el interruptor TEC 3, cambiará la orientación de la flecha del display, lo cual invertirá el sentido tanto del cambio de los caracteres como del desplazamiento del cursor.
 - ❖ Al presionar el interruptor TEC 4, si alguno de los caracteres del valor de conversión es un espacio o el valor no se encuentra incluido dentro del rango del conversor, el sistema lo rechazará y le volverá a mostrar al usuario el menú principal (estado 0).
- Estado 7 (Operando 2):
 - ❖ El usuario podrá hacer uso de los interruptores para ingresar el segundo operando de la operación aritmética que desea realizar. El operando se irá mostrando en el

display LCD a medida que el usuario lo ingrese para que compruebe si lo está introduciendo correctamente. Éste sería el formato del display LCD mientras el usuario ingresa el operando:

Operando 2: →

- ❖ El operando se iría mostrando en la línea inferior del display a medida que el usuario lo introduzca.
- ❖ Si se presiona el interruptor TEC 1, cambiará el carácter sobre el que esté actualmente posicionado el cursor. Prestar atención a la flecha que se incluye en el display ya que, si apunta hacia la derecha, significa que el cambio se producirá hacia adelante, es decir, si hay un 2, se pasará a un 3, pero si la flecha apunta hacia la izquierda, el carácter cambiará hacia atrás, es decir, pasará a ser un 1.
- ❖ A continuación se muestran los caracteres disponibles para ingresar el operando, ubicados según su orden de aparición al utilizar el interruptor TEC 1:

Espacio	0	1	2	3	4	5	6	7	8	9
---------	---	---	---	---	---	---	---	---	---	---

- ❖ Si se presiona el interruptor TEC 2, se desplazará el cursor a través de la línea inferior del display, en la que se está escribiendo el mensaje. Prestar atención a la flecha que se incluye en el display ya que, si apunta hacia la derecha, significa que el desplazamiento se producirá hacia adelante (hacia la derecha), pero si la flecha apunta hacia la izquierda, el mismo será hacia atrás (hacia la izquierda).
- ❖ Si se presiona el interruptor TEC 3, cambiará la orientación de la flecha del display, lo cual invertirá el sentido tanto del cambio de los caracteres como del desplazamiento del cursor.
- ❖ Al presionar el interruptor TEC 4, si se produce alguna de las siguientes situaciones, el sistema rechazará la operación y le volverá a mostrar al usuario el menú principal (estado 0):
 - El segundo operando no se encuentra incluido dentro del rango de la calculadora.
 - La operación es una resta y el sustraendo (segundo operando) es mayor al minuendo (primer operando).
 - La operación es una división y el divisor (segundo operando) es un cero.

- Estado 8 (Codificación Final):

- ❖ Se le presentará al usuario un menú en el display LCD conformado por dos submenús que se alternarán entre sí con un período de 2 segundos. A partir de ellos podrá elegir el sistema de numeración hacia el cual desea realizar la conversión. Los submenús tendrán los siguientes formatos:

Codif. final:
1. Binaria

2. Decimal
3. Hexadecimal

Submenú 1

Submenú 2

- ❖ El pulsado de los interruptores TEC 1, TEC 2 y TEC 3 le indicará al sistema que el usuario desea hacer la conversión hacia el sistema binario, decimal y hexadecimal respectivamente.
- ❖ Si se presiona el TEC 4, se cancelará la conversión y se le volverá a mostrar al usuario el menú principal (estado 0).
- Estado 9 (Resultado):
 - ❖ Se procederá a mostrarle al usuario en el cartel LED el resultado de la operación aritmética realizada. Dicho resultado se desplazará a través del cartel de derecha a izquierda para que la persona pueda visualizarlo en su totalidad, y presentará el siguiente formato:

OP1 OPER OP2 = RES

- OP1: Primer operando de la operación.
- OPER: Operador de la operación.
- OP2: Segundo operando de la operación.
- RES: Resultado de la operación.
- ❖ Por su parte, el formato del display LCD será el siguiente (el mensaje permanecerá “congelado” en la pantalla):

Resultado:
OP1 OPER OP2 = RES

- ❖ Ahora el usuario podrá hacer uso de los interruptores TEC para interactuar con el cartel en vez del display. Sus interacciones podrán ser las siguientes:
 - TEC 1: Cambiar el tamaño de los caracteres mostrados en el cartel. Si son grandes, se achicarán, y si son pequeños, se agrandarán.
 - TEC 2: Modificar la velocidad de desplazamiento del mensaje a través del cartel. Hay tres velocidades: lenta, normal y rápida. Inicialmente la velocidad es la normal, presionando el interruptor se pasará a la rápida y, si se lo vuelve a pulsar, a la lenta, para finalmente regresar a la normal y así sucesivamente.
 - TEC 3: Detener el desplazamiento del mensaje a través del cartel. Ahora el mismo quedará “congelado” en él, pero no se borrará. Estando el mensaje detenido, si se presiona el interruptor TEC 2 se lo podrá retroceder una columna por cada vez que se lo presione.
 - TEC 4: Se borra el mensaje del cartel y se le vuelve a mostrar al usuario el menú principal en el display LCD (estado 0).

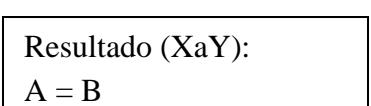
- Estado 10 (Valor Final):

- ❖ Se procederá a mostrarle al usuario en el cartel LED el resultado de la operación aritmética realizada. Dicho resultado se desplazará a través del cartel de derecha a izquierda para que la persona pueda visualizarlo en su totalidad, y presentará el siguiente formato:



A = B

- A: Valor inicial de la conversión.
- B: Valor final de la conversión.
- ❖ Por su parte, el formato del display LCD será el siguiente (el mensaje permanecerá “congelado” en la pantalla):



Resultado (XaY):
A = B

- X: Sistema de numeración inicial de la conversión (B → Binario; D → Decimal; H → Hexadecimal).
- Y: Sistema de numeración final de la conversión (B → Binario; D → Decimal; H → Hexadecimal).
- ❖ Ahora el usuario podrá hacer uso de los interruptores TEC para interactuar con el cartel en vez del display. Sus interacciones podrán ser las siguientes:
 - TEC 1: Cambiar el tamaño de los caracteres mostrados en el cartel. Si son grandes, se achicarán, y si son pequeños, se agrandarán.
 - TEC 2: Modificar la velocidad de desplazamiento del mensaje a través del cartel. Hay tres velocidades: lenta, normal y rápida. Inicialmente la velocidad es la normal, presionando el interruptor se pasará a la rápida y, si se lo vuelve a pulsar, a la lenta, para finalmente regresar a la normal y así sucesivamente.
 - TEC 3: Detener el desplazamiento del mensaje a través del cartel. Ahora el mismo quedará “congelado” en él, pero no se borrará. Estando el mensaje detenido, si se presiona el interruptor TEC 2 se lo podrá retroceder una columna por cada vez que se lo presione.
 - TEC 4: Se borra el mensaje del cartel y se le vuelve a mostrar al usuario el menú principal en el display LCD (estado 0).

5.3. Sistema operativo y tareas

A diferencia de otros sistemas operativos como Linux y Windows, el que nosotros utilizamos para ejecutar nuestro programa, OSEK-OS, es estático (además de ser un sistema de tiempo real). Esto significa que las tareas, sus prioridades, cantidad de memoria que utilizan, etc., es definido antes de compilar el código, en un proceso que se llama generación.

En OSEK no es posible crear una tarea de forma dinámica, no se puede cambiar la prioridad a una tarea como estamos acostumbrados en Windows y Linux. Por ende OSEK

sería un sistema impensable para una computadora o un celular donde constantemente estamos cargando nuevos programas, corriendolos, cerrándolos etc. OSEK-OS está pensado para un sistema embebido que debe realizar una tarea específica en tiempo real y donde no se necesita cargar nuevas tareas de forma dinámica.

El ser un sistema operativo estático trae grandes ventajas a su comportamiento en tiempo real. El sistema se comporta de forma totalmente determinística. No existe la posibilidad de que una tarea no pueda ser cargada porque no hay más memoria disponible como podría pasar en Linux/Windows. Las tareas tienen una prioridad asignada de ante mano, por ende una tarea de gran prioridad porque realiza un control crítico de seguridad tendrá siempre esa misma prioridad.

Esto es sobre todo importante en sistemas de control críticos con requerimientos SIL. Pero además para la funcionalidad en sistemas en donde los fallos no son aceptables o tienen un costo demasiado alto.

Al ser OSEK-OS un sistema estático es necesario configurarlo: indicar cuántas tareas hay definidas, qué prioridad tienen estas tareas, etc. Para ello OSEK definió otro estándar llamado OSEK Implementation Language, comúnmente llamado OIL. Es un lenguaje textual muy fácil de interpretar, similar al C, donde uno indica las características del OS, tareas, etc. Luego de configurar el OS debemos generar el sistema operativo. Una vez generado del OS y antes de compilar debemos crear el código C con nuestra tarea.

A diferencia de otros sistemas operativos donde las tareas corren por un tiempo indeterminado hasta ser terminadas, en OSEK y por lo general en sistemas de tiempo real las tareas realizan su cometido y terminan. No se utilizan estructuras como while(1) para mantener la tarea corriendo, ni sleeps para dormir entre activaciones, sino que se inicia la tarea, la misma realiza su cometido y luego termina. Ya sea leer la temperatura de un sensor, recibir un paquete de comunicación, procesar datos de audio, sin importar lo que sea: se comienza, se procesa y se termina.

En OSEK-OS las prioridades de las tareas se definen de forma estática en el OIL. La prioridad es un número entero entre 1 y 255. Mayor sea el número mayor es la prioridad. Si dos tareas tienen la misma prioridad son ejecutadas según su orden de llegada FIFO. Una tarea que se encuentra corriendo nunca va a ser interrumpida por una de menor prioridad ni por una de la misma prioridad.

Las alarmas son utilizadas para realizar una acción luego de determinado tiempo. Las alarmas de OSEK-OS pueden realizar tres tipos de acciones:

- Activar una tarea.
- Setear un evento de una tarea.
- Llamar una callback en C.

Con respecto a su aplicación en nuestro proyecto, hemos podido comprobar que la rigurosidad y la intolerancia ante fallos por parte de la política de scheduling del OSEK-OS es tal que, si el tiempo de ejecución de una tarea superaba alguno de los períodos de ejecución de cualquiera de las demás, o del suyo propio, se detenía la ejecución del programa y se devolvía un error de ejecución. Para evitar este problema, hemos tenido que distribuir la carga de ejecución lo más equitativamente posible entre todas las tareas creadas para que ninguna se mantuviera en ejecución durante demasiado tiempo.

A continuación se muestran las tareas y alarmas que hemos creado haciendo uso del lenguaje OIL, y las explicaremos detalladamente:

```
TASK InitTask {
    PRIORITY = 1;
    ACTIVATION = 1;
    AUTOSTART = TRUE {
        APPMODE = AppMode1;
    }
    STACK = 512;
    TYPE = EXTENDED;
    SCHEDULE = NON;
    RESOURCE = POSIXR;
    EVENT = POSIXE;
}

TASK LcdTask {
    PRIORITY = 4;
    ACTIVATION = 1;
    STACK = 512;
    TYPE = EXTENDED;
    SCHEDULE = NON;
    RESOURCE = POSIXR;
    EVENT = POSIXE;
}

ALARM ActivateLcdTask {
    COUNTER = HardwareCounter;
    ACTION = ACTIVATETASK {
        TASK = LcdTask;
    }
}

TASK MatrizTask {
    PRIORITY = 3;
    ACTIVATION = 1;
    STACK = 512;
    TYPE = EXTENDED;
    SCHEDULE = NON;
    RESOURCE = POSIXR;
    EVENT = POSIXE;
}

ALARM ActivateMatrizTask {
    COUNTER = HardwareCounter;
    ACTION = ACTIVATETASK {
        TASK = MatrizTask;
    }
}

TASK RegistrosTask {
    PRIORITY = 2;
    ACTIVATION = 1;
    STACK = 512;
```

```

    TYPE = EXTENDED;
    SCHEDULE = NON;
    RESOURCE = POSIXR;
    EVENT = POSIXE;
}

ALARM ActivateRegistrosTask {
    COUNTER = HardwareCounter;
    ACTION = ACTIVATETASK {
        TASK = RegistrosTask;
    }
}

TASK SwitchesTask {
    PRIORITY = 5;
    ACTIVATION = 1;
    STACK = 512;
    TYPE = EXTENDED;
    SCHEDULE = NON;
    RESOURCE = POSIXR;
    EVENT = POSIXE;
}

ALARM ActivateSwitchesTask {
    COUNTER = HardwareCounter;
    ACTION = ACTIVATETASK {
        TASK = SwitchesTask;
    }
}

```

- InitTask:
 - ❖ Esta tarea es creada simplemente para que se inicie automáticamente al comienzo de la ejecución del programa y realice algunas tareas de inicialización de los puertos e interruptores de la CIAA, el cartel LED y el display LCD. Además, activa las demás tareas mediante sus alarmas.
 - ❖ La tarea se ejecuta una sola vez al principio y nunca más.
- LcdTask:
 - ❖ Controla todas las actividades directamente relacionadas con el funcionamiento del display LCD:
 - Alternar submenús entre sí si es que corresponde según el estado de la máquina.
 - Actualizar el contenido de la pantalla, ya sea un menú y/o caracteres ingresados por el usuario.
 - ❖ Actúa directamente sobre los pines de la CIAA conectados al display LCD.
- ActivateLcdTask:
 - ❖ Es la alarma asociada con la tarea LcdTask. Al escribir el código en C, se la configuró para activar la tarea con un retardo inicial de 50 milisegundos y un período de 10 milisegundos.

- MatrizTask:
 - ❖ Realiza la configuración de la matriz “lógica” ya explicada en las consideraciones iniciales, determinando los leds del cartel que se deben encender y desplazando el mensaje que se muestra en el mismo un lugar hacia la izquierda cada vez que se ejecuta. No actúa directamente sobre ninguno de los componentes electrónicos.
- ActivateMatrizTask:
 - ❖ Es la alarma asociada con la tarea MatrizTask. Al escribir el código en C, se la configuró para activar la tarea con un retardo inicial de 50 milisegundos y un período de 25 milisegundos.
- RegistrosTask:
 - ❖ Envía todas las señales necesarias a los registros de desplazamiento de la placa de control a través de los pines de la CIAA que estén conectados a ellos para encender los leds del cartel determinados previamente en la tarea MatrizTask.
 - ❖ Aplica la multiplexación por columnas: cada vez que se ejecuta analiza una columna distinta.
- ActivateRegistrosTask:
 - ❖ Es la alarma asociada con la tarea RegistrosTask. Al escribir el código en C, se la configuró para activar la tarea con un retardo inicial de 50 milisegundos y un período de 1 milisegundo para hacer creer al usuario que todos los leds del cartel se están encendiendo simultáneamente.
- SwitchesTask:
 - ❖ Averigua si el usuario ha presionado alguno de los interruptores de la CIAA desde la última vez que se ejecutó la tarea y actúa en consecuencia sobre la máquina de estados.
 - ❖ Fue necesario aplicar un antirrebote por software para evitar que un pulsado de un interruptor se interpretara como si hubiera sido presionado en múltiples oportunidades. Sin embargo, como el OSEK no nos permitía implementar el antirrebote mediante un retardo ya que la tarea duraba demasiado, implementamos un contador que se incrementara cada vez que se ejecutaba la tarea hasta alcanzar un valor que consideramos adecuado para que la tarea pudiera continuar verificando si se había vuelto a presionar algún pulsador.
- ActivateSwitchesTask:
 - ❖ Es la alarma asociada con la tarea SwitchesTask. Al escribir el código en C, se la configuró para activar la tarea con un retardo inicial de 50 milisegundos y un período de 10 milisegundos.
- Referencias adicionales:
 - ❖ PRIORITY: Prioridad de la tarea.
 - ❖ AUTOSTART = TRUE: La tarea se iniciará automáticamente al inicio de la ejecución del programa.
 - ❖ STACK: Tamaño de la pila de la tarea (en bytes).
 - ❖ TYPE: Las tareas básicas (BASIC) no tienen eventos (EVENT) y por ende no pueden permanecer en el estado waiting. En cambio, las extendidas (EXTENDED) pueden tener uno o más eventos y esperar.

- ❖ **SCHEDULE = NON:** Las tareas con una política de scheduling no preemptiva no pueden ser interrumpidas nunca por otra tarea, sin importar qué prioridad tengan. Se ejecutan sin interrupción hasta que terminan, pasan al estado waiting esperando un evento o llaman a la función Schedule.
- ❖ **RESOURCE:** Los recursos en todo sistema sea o no de tiempo real son por lo general escasos. Por ello para acceder a ellos se utilizan en los sistemas convencionales semáforos o mutex para poder acceder a ellos desde varias tareas sin generar interferencias. La utilización de semáforos y mutex genera dos problemas conocidos: inversión de prioridades y deadlocks. OSEK-OS ofrece una solución para el acceso a recursos que evita estos dos problemas.
- ❖ **COUNTER:** Para la implementación de las alarmas, ya sea 1 o n el sistema operativo necesita un contador de HW. Con un contador es posible configurar la cantidad de alarmas que sea necesario.
- ❖ **ACTION:** Determina qué acción debe realizar la alarma cuando expira. Ésta puede ser activar la tarea a la que está asociada (ACTIVATETASK) o un evento por el que la misma está esperando (SETEVENT).

5.4. Modularización

Al ser conscientes de que aplicar la estrategia “divide y vencerás” nos simplificaría en gran medida la implementación del sistema inicial, decidimos dividirla en pequeñas entidades funcionalmente independientes que cumplan tareas bien específicas. Estas entidades, conocidas como módulos, contarían con un alto nivel de abstracción del problema original que nos permita aislarnos del sistema que estamos desarrollando y concentrarnos en crear soluciones a problemáticas concretas que, una vez se combinen, nos permitirán dar una respuesta al problema inicial con mayor facilidad y velocidad. Además, podríamos ofrecerle a cualquier programador interesado en nuestro proyecto un código más elegante, legible, intuitivo y fácil de interpretar y mantener. Asimismo, algunos de los módulos tendrían una abstracción tal que sus funcionalidades podrían ser adaptables a cualquier otro trabajo que no guarde ninguna relación el nuestro y su código sería fácilmente portable y reutilizable.

Con estas ideas en mente, hemos desarrollado un gran número de módulos para ofrecer las funcionalidades del sistema prometidas, cada uno de los cuales atiende una tarea específica y abstraída de dichas funcionalidades. Cada módulo por sí mismo no resuelve el problema inicial pero, combinados, logran realizar todas las tareas necesarias para solucionarlo. También, como se había anticipado, existen varios que pueden ser fácilmente incorporados a otros proyectos completamente independientes del nuestro. A continuación se describen cada uno de los módulos implementados junto con la tarea que realizan para ayudar a que el programa ofrezca las funcionalidades que debe tener el sistema:

- **Calculadora:** recibe dos operandos y un operador y aplica la correspondiente operación aritmética entre ellos. Devuelve el resultado en un buffer listo para ser presentado tanto en los dispositivos de salida que se tengan disponibles, desde un display LCD hasta una matriz de leds. Al tener una abstracción tan alta con respecto a la problemática original de nuestro proyecto, este módulo es altamente reutilizable.

- Conversor: recibe un valor de conversión inicial junto con su sistema de numeración y el sistema al que se desea convertir y realiza la conversión. Devuelve en un buffer el valor final de la conversión, pero aún necesita para seguir siendo procesado para ser presentado en un dispositivo de salida con un formato adecuado. Al tener una abstracción tan alta con respecto a la problemática original de nuestro proyecto, este módulo es altamente reutilizable.
- Interfaz: maneja las interacciones entre el usuario y el display LCD, escribiendo en el buffer del display el contenido que debe mostrar en su pantalla, ya sea un menú, un resultado o los caracteres que el usuario está ingresando o ya ha ingresado al sistema mediante los interruptores de la CIAA. Este módulo no está tan abstraído de la problemática original de nuestro proyecto como los demás, así que es improbable que pueda reutilizarse otro trabajo cuyo objetivo no sea similar al nuestro.
- Lcd: controla el funcionamiento del display LCD a un nivel más bajo que el módulo anterior, ya que accede y escribe directamente sobre los pines de la CIAA conectados a él y actualiza su pantalla con el contenido del buffer que aún no haya sido leído. Este módulo fue tomado íntegramente de un trabajo realizado para la cátedra “Circuitos digitales y microcontroladores” y se le hicieron unas modificaciones mínimas para adaptarlo a la CIAA. Al tener una abstracción tan alta con respecto a la problemática original de nuestro proyecto, este módulo es altamente reutilizable.
- Matriz: aquí se determinan cuáles son los leds del cartel que se deben encender para mostrar el mensaje o el resultado que corresponda, y a partir de esta información se crea una matriz que deberá ser leída por otro módulo que opere sobre los registros a un más bajo nivel. También aquí se realiza el desplazamiento en forma lógica de los caracteres del cartel de derecha a izquierda. Este módulo no está tan abstraído de la problemática original de nuestro proyecto como los demás, así que es improbable que pueda reutilizarse otro trabajo cuyo objetivo no sea similar al nuestro.
- Proyecto: es el archivo principal en el cual se inicializan todos los dispositivos (display LCD, matriz de leds y puertos de la CIAA) y se definen, activan y ejecutan todas las tareas que harán uso de ellos a partir de los demás módulos. Este módulo no está tan abstraído de la problemática original de nuestro proyecto como los demás, así que es improbable que pueda reutilizarse otro trabajo cuyo objetivo no sea similar al nuestro.
- Puertos: este módulo opera directamente sobre los pines de la CIAA utilizados en nuestro proyecto, escribiendo o leyendo valores de ellos, sin importarle el fin que se esconda detrás de estas lecturas y escrituras. Probablemente es el que presenta el mayor grado de reusabilidad de todos, ya que cualquier programador que cuente con una CIAA y quiera hacer uso de estos mismos puertos debería poder hacerlo sin inconvenientes.
- Registros: al igual que el módulo Lcd, es un módulo que opera a un nivel más cercano al hardware ya que trabaja directamente sobre los pines de la CIAA conectados a los registros de desplazamiento de la placa de control con el expreso fin de enviarle las señales que sean necesarias para encender los leds indicados en la matriz lógica definida en el módulo Matriz. Implementa la multiplexación por columnas. Este módulo no está tan abstraído de la problemática original de nuestro proyecto como los demás, así que es improbable que pueda reutilizarse otro trabajo cuyo objetivo no sea similar al nuestro.

- Switches: averigua si el usuario ha presionado alguno de los interruptores de la CIAA y actúa en consecuencia sobre la máquina de estados. Aunque la lectura de los interruptores presenta un alto nivel de abstracción con respecto a la problemática original de nuestro proyecto, no ocurre lo mismo con la máquina de estados, por lo que en el mejor de los casos tal vez podría reutilizarse parte del código de este módulo para otros trabajos.
- Tiempo: permite retardar la ejecución de una tarea durante un tiempo determinado. Es especialmente útil para el display LCD, el cual suele realizar tareas que requieren un cierto tiempo de espera para completarse. Sin embargo, como se explicó anteriormente, se debe tener cuidado con que la tarea no se ejecute durante demasiado tiempo ya que en caso contrario el sistema operativo OSEK detendrá la ejecución del programa y devolverá un error. Al tener una abstracción tan alta con respecto a la problemática original de nuestro proyecto, este módulo es altamente reutilizable.

6. Observaciones finales

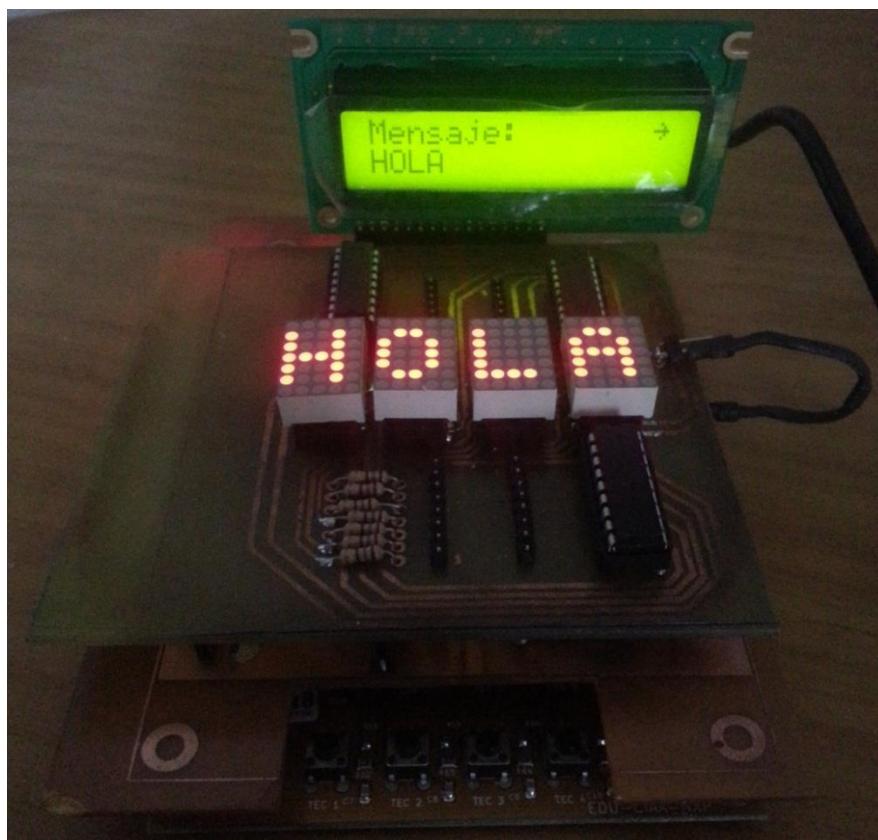
- Puede encontrarse el código en C del programa desarrollado en nuestro repositorio en Github, alojado en la carpeta Firmware, dentro del directorio Software. Téngase en cuenta que, aunque podría creerse que la totalidad del código está en la carpeta “proyecto”, esto no es así, ya que hubo algunas implementaciones e inicializaciones de puertos de la CIAA que utilizamos para nuestro proyecto que no se encontraban incluidas ni configuradas por defecto en la IDE que descargamos de la página oficial de la CIAA por lo que tuvimos que agregarlas manualmente por nuestra cuenta. Estas implementaciones se encuentran en otros directorios, por lo que toda la carpeta Firmware debe ser incorporada y compilada en el Eclipse para que la ejecución del código sea correcta.
- Para obtener más información sobre algunos esquemáticos que hemos diseñado tanto para la placa de control como para el cartel LED y no incluimos en el informe, dirigirse al repositorio en Github, directorio Hardware/PCB (carpeta ControladorMatrizLED para la placa de control y MatrizLED para la del cartel LED). Hemos puesto a disposición de cualquier interesado tanto los esquemáticos desarrollados en Kicad para que puedan ser utilizados con cualquier fin deseado como también los circuitos impresos en formato PDF para que pueda apreciarse cómo adaptamos las conexiones entre las placas y los componentes para que encajaran en el espacio disponible. No obstante, advertimos que, sin una lectura previa del informe y de los esquemáticos, pueden tornarse difíciles de interpretar.
- Entre algunos accidentes que hemos tenido durante el desarrollo del proyecto, han ocurrido algunos que son dignos de mención, a saber:
 - ❖ Hemos vuelto inservible una matriz de leds al quemar algunos de sus diodos por conectarlos a una fuente de alimentación sin utilizar resistencias para protegerlos.
 - ❖ Malgastamos una placa de cobre por perforarla apresurada e impacientemente, tornándose imposible la tarea de conectarle cualquiera de los componentes electrónicos para los que la habíamos diseñado.
 - ❖ A modo de curiosidad, descubrimos que el puerto GPIO1 de la CIAA que nos fue proporcionada no funciona correctamente si se lo somete a sucesivos cambios

de valores a altas velocidades: al utilizarse como reloj SHCP para los registros de desplazamiento, presentaba inconvenientes. Sin embargo, si la usábamos como entrada serie, no tenía problemas. De todas formas, por si acaso decidimos descartar por completo la posibilidad de utilizarlo para nuestro proyecto y lo reemplazamos por el GPIO2 que respondió a la perfección en todas las pruebas ante las que lo sometimos.

- A continuación se indica la distribución final del proyecto y de sus responsabilidades y tareas entre los integrantes del grupo:

Tarea	Responsable
Conexión y puesta a prueba de un prototipo del sistema en una protoboard previo a contar con la CIAA	Jourdón, Julián
Diseño de los esquemáticos y circuitos PCB en Kicad	Maicá, Juan Manuel
Impresión de los circuitos PCB, perforado de las placas y soldadura de los componentes electrónicos	López Acuña, Axel
Diseño e implementación del software (interfaz de usuario, máquina de estados, tareas y modularización)	Scorza, Facundo Ricardo
Integración entre hardware y software y prueba final del sistema	Tarea Conjunta

- A modo de cierre, incluimos una imagen del sistema final en funcionamiento, al cual le ingresamos previamente el mensaje “HOLA” para que lo mostrara en el cartel LED:



7. Apéndice

A continuación se incluyen los diagramas esquemáticos y los diseños impresos utilizados para la realización de las PCB:

<http://www.proyecto-ciaa.com.ar/>

Conector del Poncho

RESET

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

X

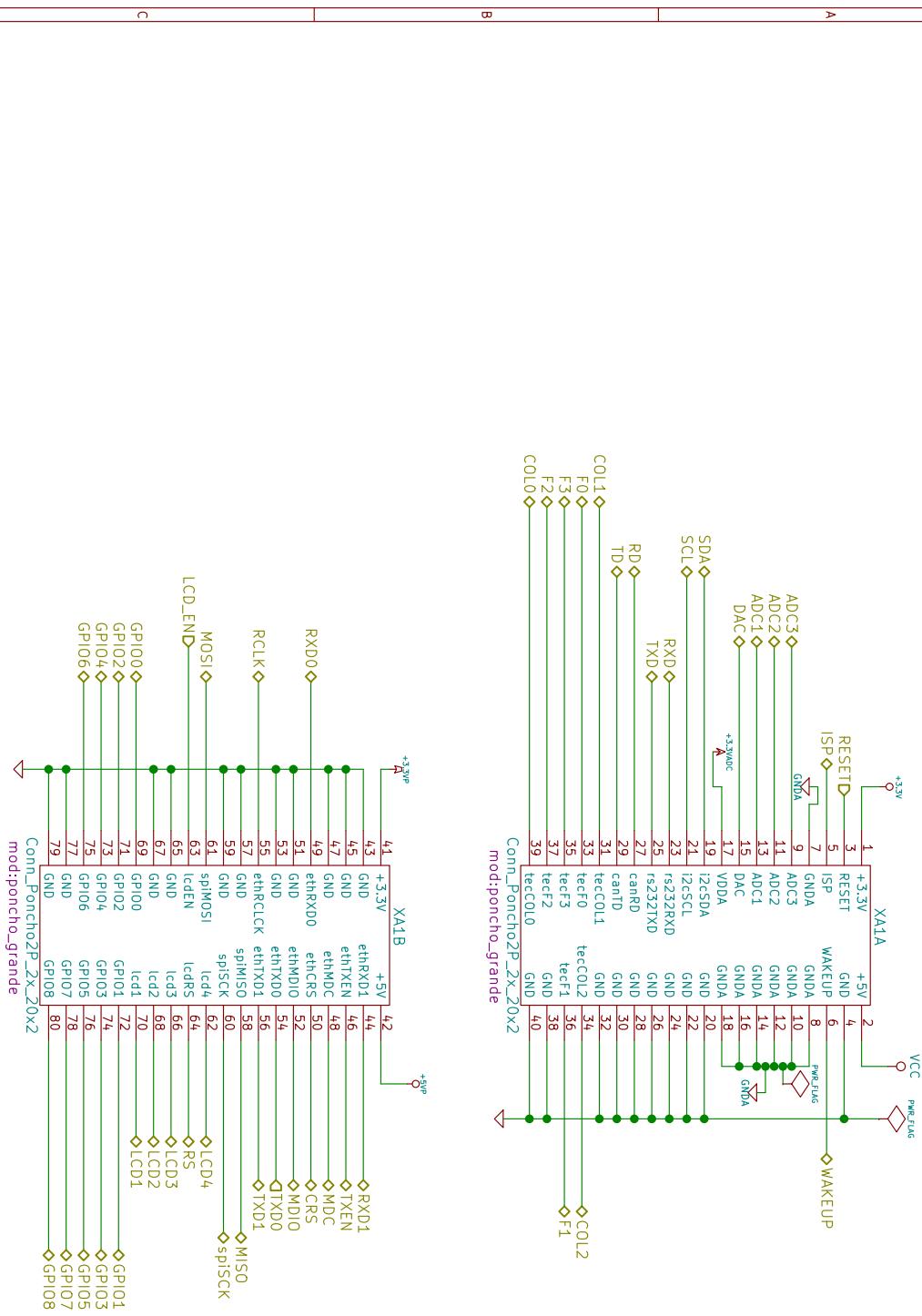
X

X

X

X

</



CÓDIGO PONCHO:

סימן מס' 10 – מילויים נומריים וארצניים

Proyecto CIAA – COMPUTADOR

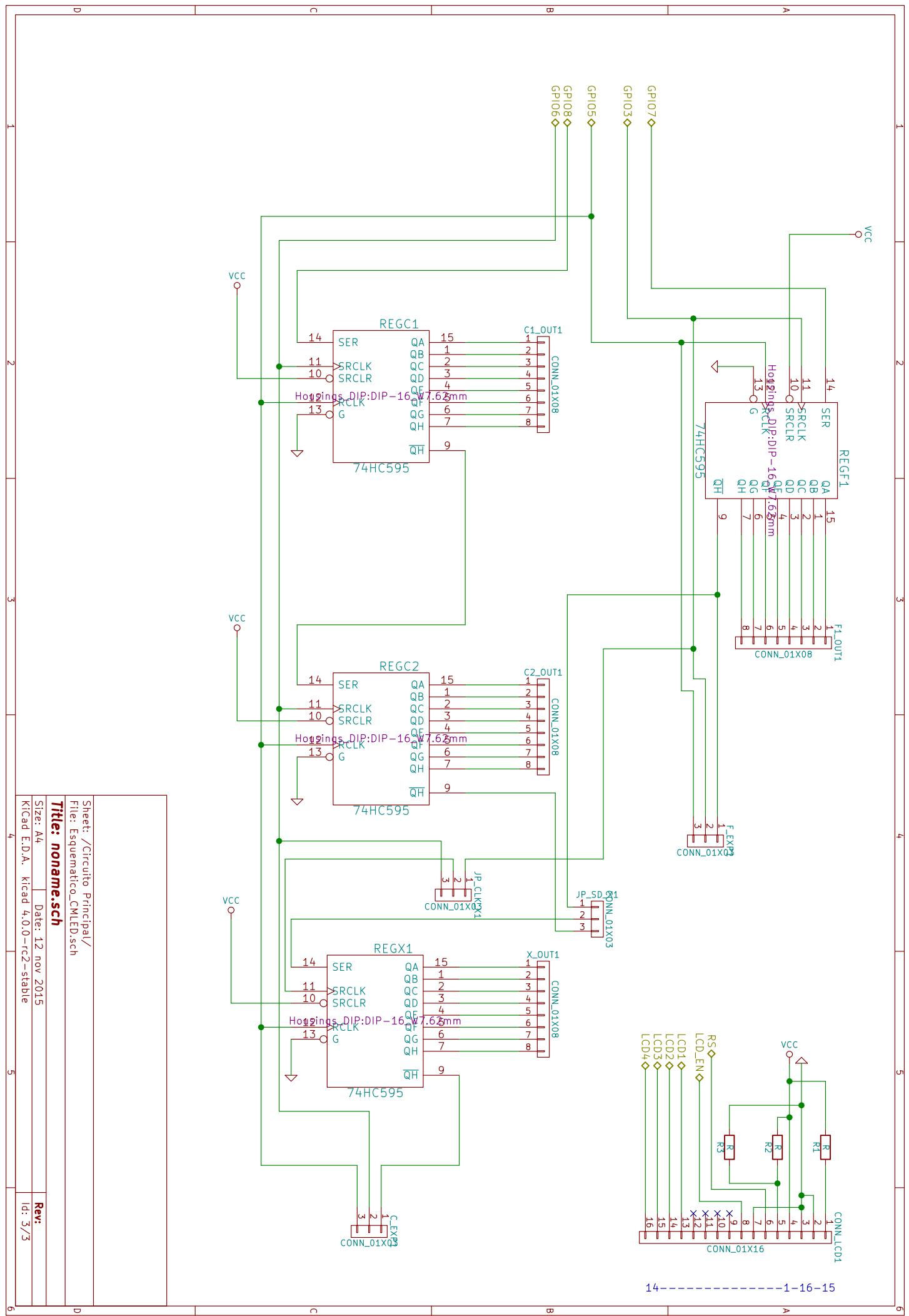
הנִזְקָנָה בְּבֵית־יְהוָה

Proyecto CIAA – COMPUTADORA INDUSTRIAL ABIERTA ARGENTINA
Sheet: /Conector del Poncho/

VA

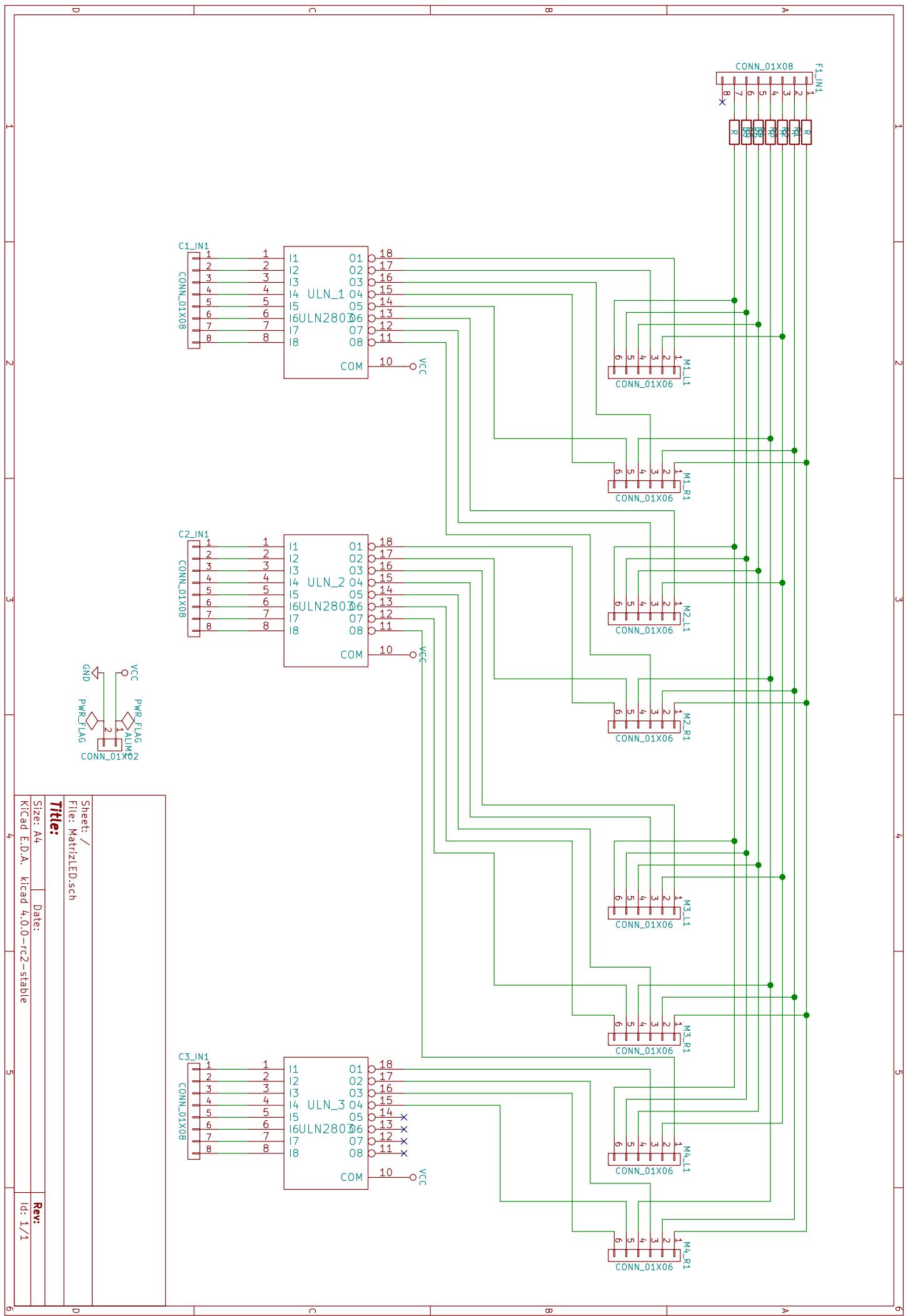
11

הנִזְקָנָה בְּבֵית־יְהוָה



- Referencias:

- ❖ 74HC595 → Registro de desplazamiento de 8 bits (utilizar la numeración de los pines como referencia para entender la funcionalidad de cada uno):
 - REGF1 → Controla las filas 1 a 8 del cartel LED.
 - REGC1 → Controla las columnas 1 a 8 del cartel LED.
 - REGC2 → Controla las columnas 8 a 16 del cartel LED.
 - REGX1 → Puede controlar las columnas 17 a 24 o las filas 9 a 16 del cartel LED en función de las necesidades del usuario.
- ❖ F1_OUT1 → Salidas de la placa de control que deberán conectarse a las filas 1 a 8 del cartel LED.
- ❖ C1_OUT1 → Salidas de la placa de control que deberán conectarse a las columnas 1 a 8 del cartel LED.
- ❖ C2_OUT1 → Salidas de la placa de control que deberán conectarse a las columnas 9 a 16 del cartel LED.
- ❖ X_OUT1 → Salidas de la placa de control que pueden conectarse a las columnas 17 a 24 o las filas 9 a 16 del cartel LED en función de las necesidades del usuario.
- ❖ JP_CLK_X1 y JP_SD_X1 → Aquí es donde el usuario podrá determinar, mediante unos componentes de hardware conocidos como jumpers, qué uso le desea dar al registro REGC3, dependiendo de cómo los conecte: controlar ocho filas u ocho columnas adicionales del cartel LED.
- ❖ F_EXP1 → Salidas de la placa de control que pueden ser utilizadas para incorporar registros de desplazamiento adicionales de manera tal de poder controlar un mayor número de filas del cartel LED que las toleradas en forma nativa por la placa.
- ❖ C_EXP1 → Salidas de la placa de control que pueden ser utilizadas para incorporar registros de desplazamiento adicionales de manera tal de poder controlar un mayor número de columnas del cartel LED que las toleradas en forma nativa por la placa.
- ❖ CONN_LCD1 → Salidas de la placa de control que deberán conectarse directamente a un display LCD de 16x2 para permitir la interacción del sistema con el usuario.
- ❖ R1, R2 y R3 → Resistencias de 1 Ohm, 1000 Ohm y 150 Ohm respectivamente. La primera limita la corriente que ingresa al display LCD y las otras dos regulan su contraste.



- Referencias:
 - ❖ F1_IN → Salidas de la placa de control destinadas a controlar las filas del cartel LED.
 - ❖ C1_IN1 – C3_IN1 → Salidas de la placa de control destinadas a controlar las columnas del cartel LED.
 - ❖ M1_L1 – M4_L1 → Conexiones hacia los pines del lado izquierdo de las matrices LED.
 - ❖ M1_R1 – M4_R1 → Conexiones hacia los pines del lado derecho de las matrices LED.
 - ❖ ULN_2803 → Circuito que integra ocho transistores Darlington utilizados como interruptores:
 - ULN_1 → Habilita las columnas 1 a 8 del cartel LED.
 - ULN_2 → Habilita las columnas 9 a 16 del cartel LED.
 - ULN_3 → Habilita las columnas 17 a 24 del cartel LED.
 - ❖ R → Resistencia de 180 Ohm para limitar la corriente que ingresa a los diodos del cartel LED.

