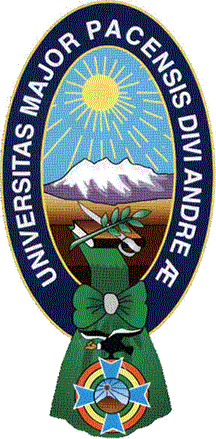
UNIVERSIDAD MAYOR DE SAN ANDRÉS

FACULTAD DE INGENIERÍA

INGENIERÍA ELECTRÓNICA



SUMA POR TECLADO EN PPI

ETN 903 II/2019

**DOCENTE:** Ing. Javier SANABRÍA GARCÍA

**ESTUDIANTE:** Oscar Humberto CORNEJO GUILLEN

**MATERIA:** Sistemas de computación

**SIGLA:** ETN - 903

2019

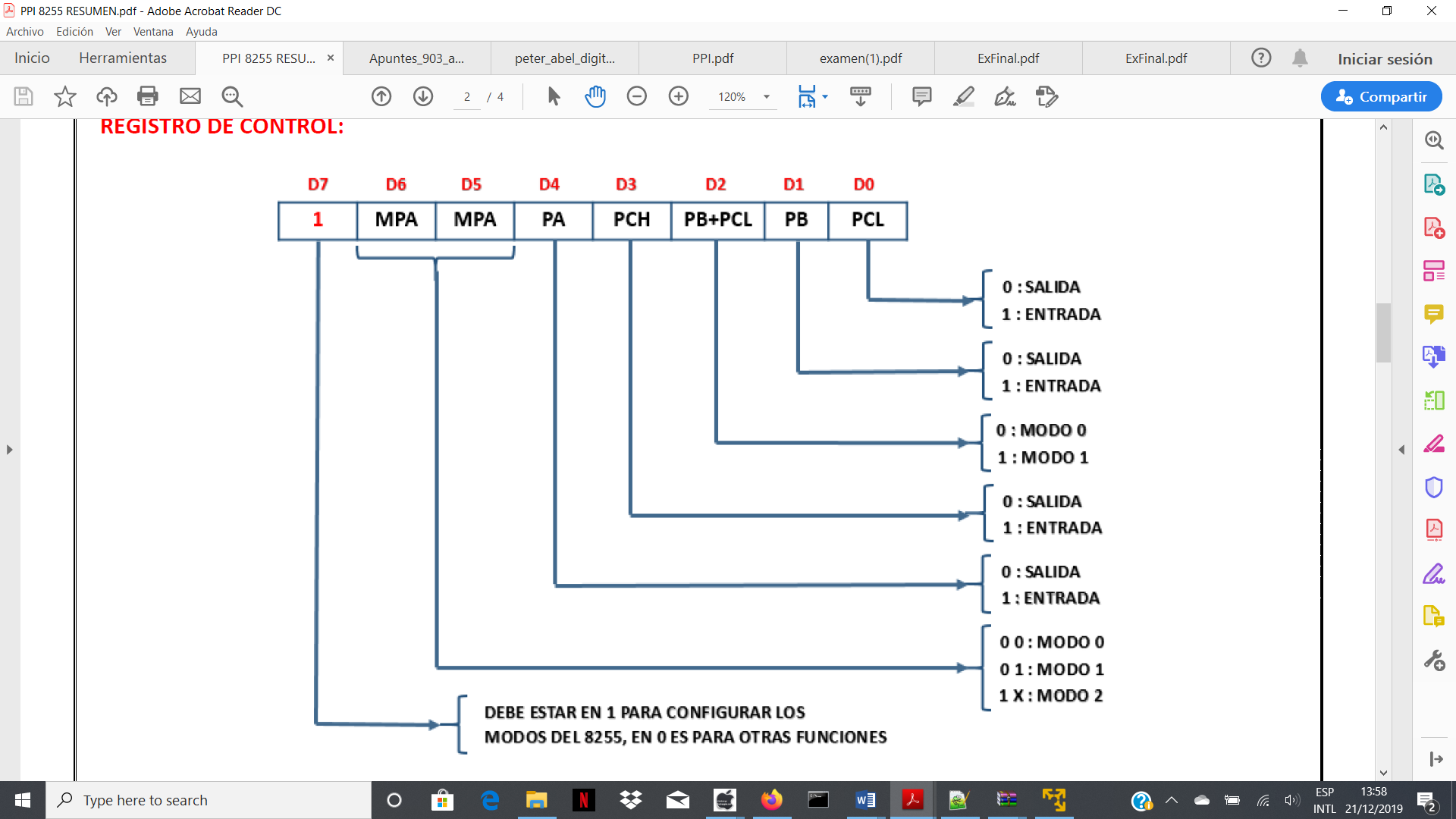
**LA PAZ – BOLIVIA**

**SUMA POR TECLADO EN PPI ETN-903**

1. Diseñar un sistema de computación que permita leer desde un teclado números enteros de máximo dos dígitos. Los números serán sumados y el resultado se presentara en un display compuesto por displays de 7 segmentos ánodo común. Para el desarrollo utilizar el PPI, lenguaje C/C++, la librería NCURSES y plataforma Linux.

**ANÁLISIS**

Para tal propósito se utilizará el siguiente diseño para el desarrollo del enunciado:

* Considerando el uso de los puertos PA, PB, PC y Control, se usará el modo 0, donde el puerto A se usará como entrada para introducir los datos desde el teclado, y los puertos PB y PC como salida, donde de los 8 bits del puerto PB, PBH se usará para las introducir el código BCD de las unidades a un integrado 7447, que es un convertir de BCD a 7 segmentos ánodo común, y PBL se usará para introducir el código BCD de las decenas a otro integrado 7447, finalmente se usará 4 bits (PCL) del puerto PC que será el código BCD de las centenas que entrará a otro integrado 7447. Teniendo así el registro de control para el PPI:
* 
* PA se usará como entrada
* PB se usará como salida
* PC se usará como salida

Así la palabra de control será:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| D7 | MPA | MPA | PA | PCH | PB+PCL | PB | PCL |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

=90h

* Considerando que al sumar dos números de dos dígitos se usará 3 displays dado que la respuesta oscila entre:

**Valor máximo**

|  |  |  |
| --- | --- | --- |
|  | **9** | **9** |
| **+** | **9** | **9** |
| **1** | **9** | **8** |

**Valor mínimo**

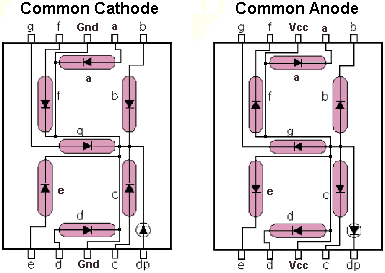
|  |  |  |
| --- | --- | --- |
|  | **0** | **0** |
| **+** | **0** | **0** |
| **0** | **0** | **0** |

**Conversor BCD a 7 segmentos ánodo común**

Las salidas en BCD serán las siguientes, las cuales entrarán al integrado 7447, así se tendrán los siguientes casos:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Decimal** | **D** | **B** | **C** | **A** | **BI/RBO** | **a** | **b** | **c** | **d** | **e** | **f** | **g** |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 4 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 6 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

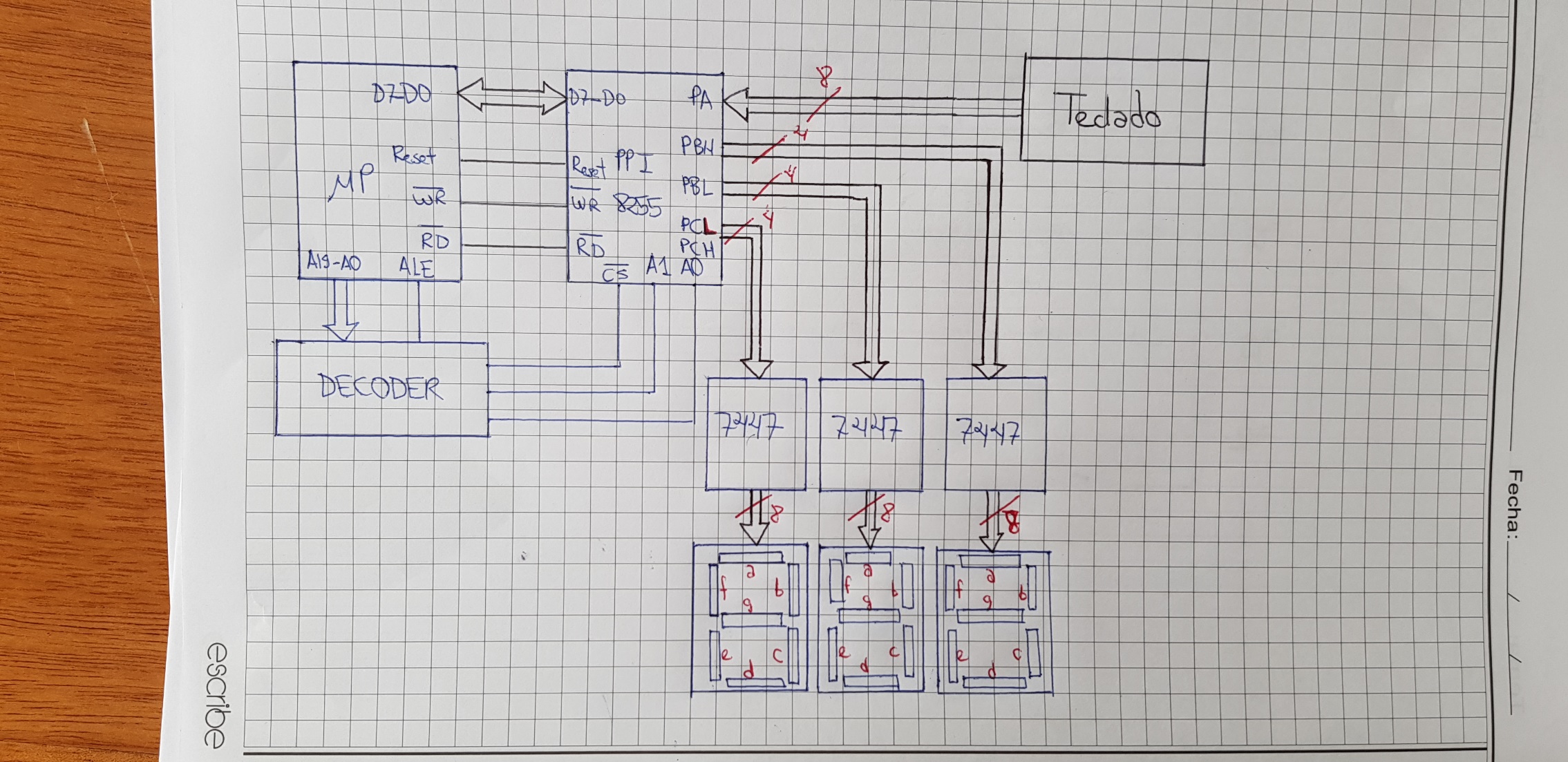
**Display ánodo común de 7 segmentos**

****

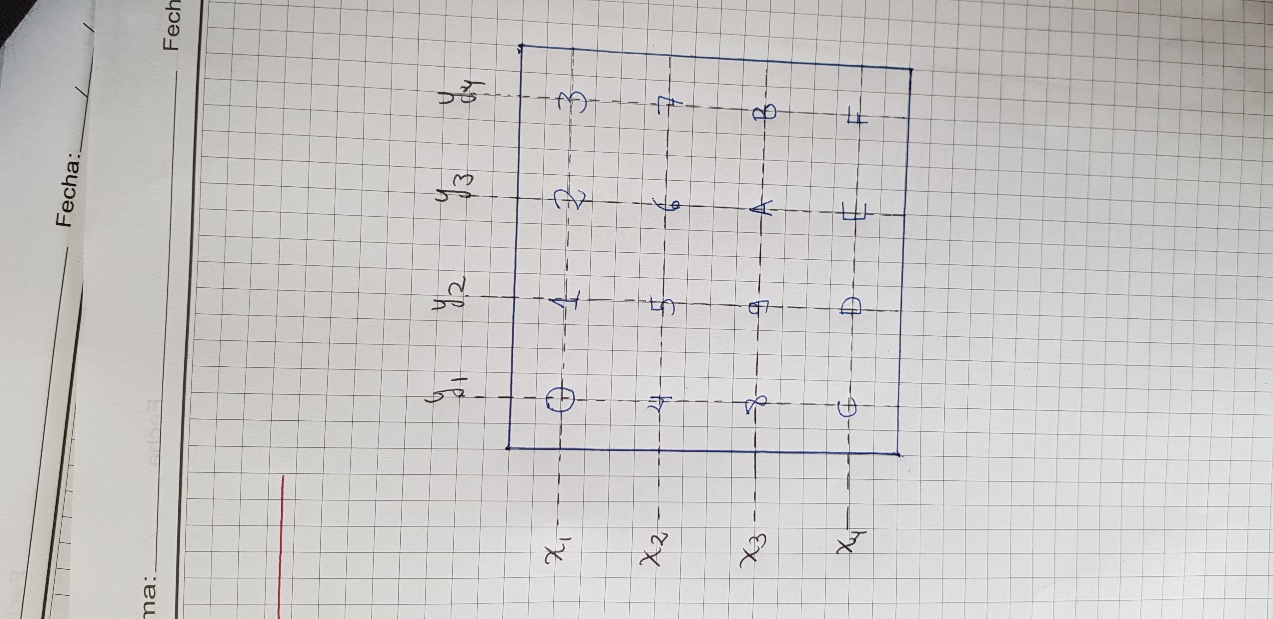
El display ánodo común funciona de la siguiente manera:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Número** | **BI/RBO** | **a** | **b** | **c** | **d** | **e** | **f** | **g** |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 2 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 3 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 4 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 5 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 6 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 7 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| - | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| + | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

* Así, en base a lo mencionado, y al Hardware mencionado se implementará el sistema a partir del siguiente Layout del circuito:



* Donde para el teclado, que será del tipo matricial, se tiene la siguiente topología:



Donde para cada tecla presionada se tienen las siguientes equivalencias en hexadecimal:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| x4 | x3 | x2 | x1 | y4 | y3 | y2 | y1 |

|  |  |  |
| --- | --- | --- |
| **Hexadecimal equivalente** | **Teclado binario** | **Teclado**  **hexadecimal** |
| 0 | 1110 1110 | 0xEE |
| 1 | 1110 1101 | 0xED |
| 2 | 1110 1011 | 0xEB |
| 3 | 1110 0111 | 0xE7 |
| 4 | 1101 1110 | 0xDE |
| 5 | 1101 1101 | 0xDD |
| 6 | 1101 1011 | 0xDB |
| 7 | 1101 0111 | 0xD7 |
| 8 | 1011 1110 | 0xBE |
| 9 | 1011 1101 | 0xBD |
| A | 1011 1011 | 0xBB |
| B | 1011 0111 | 0xB7 |
| C | 0111 1110 | 0x7E |
| D | 0111 1101 | 0x7D |
| E | 0111 1011 | 0x7B |
| F | 0111 0111 | 0x77 |

* Donde este código en hexadecimal se debe transformar a BCD para introducirlo en el proceso donde se efectuará la suma, donde el resultado se operará para obtener las decenas centenas y unidades a partir de los siguiente parámetros tomando en cuenta el caso más crítico donde se obtendría una respuesta de tres dígitos:

centenas = resultado**/**100 🡪 división entera

unidades = resultado **módulo** 10 🡪 obtenemos el residuo

decenas = (resultado **módulo** 100 - unidades)**/**10 🡪 le restamos las unidades al número quitando las unidades y lo dividimos en forma entera entre 10

* Obteniendo estos números entre 0 y 9 que inicialmente estarán en código ASCII, se transformará en BCD natural:

|  |  |
| --- | --- |
| **ASCII** | **BCD** |
| 0011 0000 | 0000 0000 |
| 0011 0001 | 0000 0001 |
| 0011 0010 | 0000 0010 |
| 0011 0011 | 0000 0011 |
| 0011 0100 | 0000 0100 |
| 0011 0101 | 0000 0101 |
| 0011 0110 | 0000 0110 |
| 0011 0111 | 0000 0111 |
| 0011 1000 | 0000 1000 |
| 0011 1001 | 0000 1001 |

* Los cuales se pasará a través de los puertos B y C, que actuarán como salidas a los integrados 7447 que se transformarán en 7 segmentos ánodo común finalmente mostrándose la respuesta en éstos.

**ALGORITMO**

**-** Poner en modo 0 el sistema donde el puerto A actuará como entrada y los puertos B y C como salidas.

**-** Obtener los números A y B a través del puerto A que estará programado como entrada.

- Transformar los datos A y B de código de teclado a decimal y guardarlos en variables.

- Sumar los números A y B, que estaban en las variables del anterior paso.

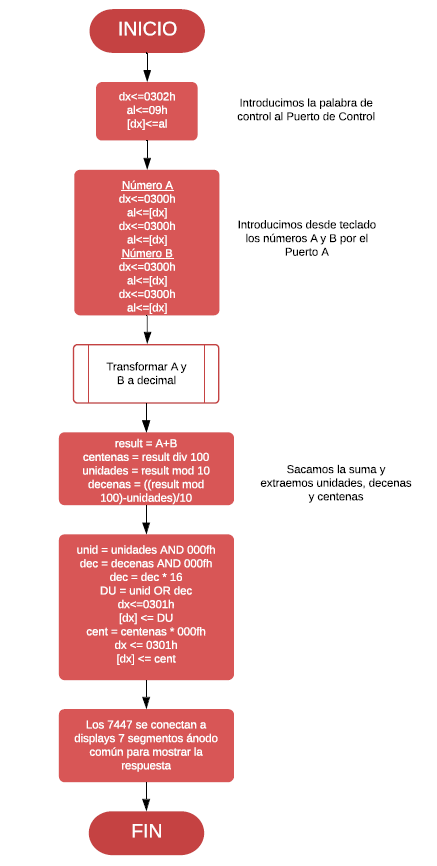
- En el resultado se obtendrá los dígitos correspondientes de las decenas, centenas y unidades inicialmente en código ASCII, a partir de la utilización de las operaciones de división entera y la operación módulo.

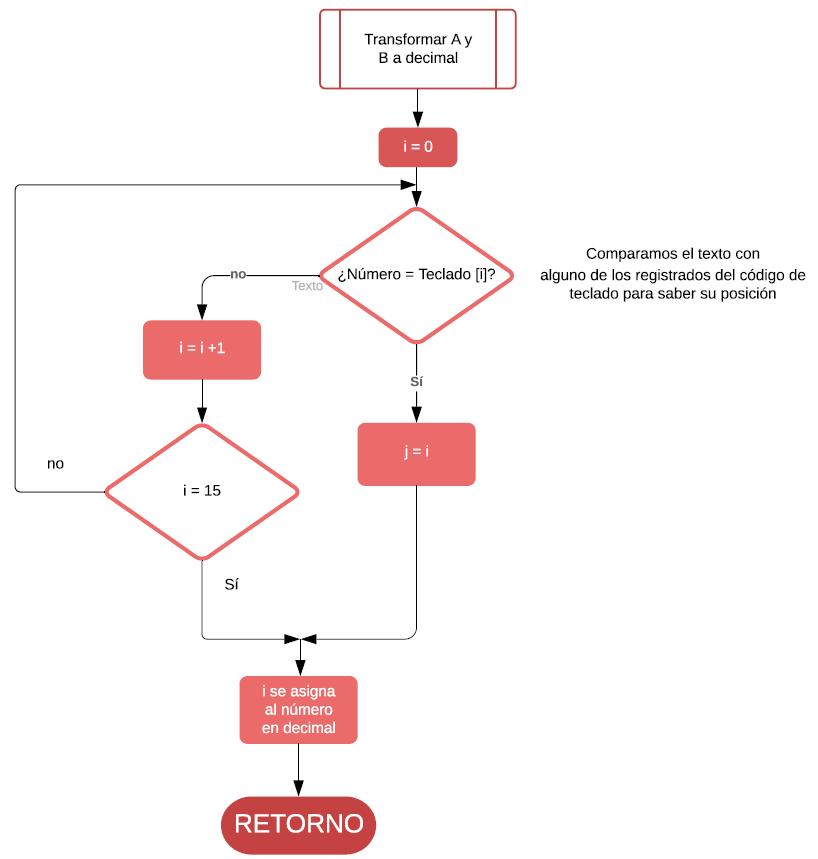
- Transformar las decenas, centenas y unidades a BCD a partir del AND con la máscara 000fh de cada dígito, y la generación de la unión de decenas y unidades DU (que entrarán al puerto PB) de la siguiente manera trabajando en forma hexadecimal:

DU= (Dígito de las decenas \* 16) OR (Dígito de las unidades)

Donde el resultado se sacará por el puerto PB, donde PBL se conectará al 7447 que está conectado al display ánodo común para las unidades, y PBH se conectará al 7447 que está conectado al display ánodo común para las decenas.

* Las centenas simplemente se generan haciendo un AND de la variable con la máscara 000fh e introduciendo el dato al puerto PC donde el dato válido se guardará en PCL y este será el que se conecte al 7447 que se conectará al display ánodo común para las centenas.
* El resultado final se mostrará en los 3 displays de 7 segmentos ánodo común.

**DIAGRAMA DE FLUJO**



**CÓDIGO EN C/C++ PLATAFORMA LINUX UTILIZANDO LA LIBRERÍA NCURSES**

Revisar el documento adjunto “sum.cpp” donde para compilar utilice la siguiente línea:

g++ sum.cpp -o sum -lncurses