INSTITUTO UNIVERSITARIO AERONÁUTICO.

INGENIERÍA INFORMÁTICA

Arquitectura I

Proyecto Integrador

Profesor:

- Toledo, Luis Eduardo

Integrantes:

- Suppia, Sofía Milagros
- Teyo, Máximo

Ciclo lectivo 2023

Fecha: 17/11/23

Introducción

Este proyecto representa una extensión y aplicación práctica de los conceptos fundamentales abordados en la materia de Arquitectura de Computadoras 1. Enfocándonos en la manipulación de una placa DE0-Nano, la finalidad principal es crear un sistema interactivo que permita al usuario controlar y experimentar con diversas secuencias de luces.

Es relevante destacar que la totalidad de la codificación del proyecto se realiza en lenguaje de programación Assembly. Este enfoque, ofrece una perspectiva única para comprender a fondo la arquitectura subyacente de la placa DE0-Nano y proporciona una experiencia inmersiva en la programación de sistemas embebidos.

A través de la implementación de estructuras switch-case y funciones en código nemotécnico, se busca no solo abordar los aspectos teóricos de la arquitectura, sino también consolidar la habilidad para traducir conceptos de más alto nivel, como la estructura de control de un menú, en instrucciones de bajo nivel comprensibles por el hardware.

Adicionalmente, la introducción de un sistema de control de acceso mediante password y la manipulación de secuencias de luces, ya sea mediante algoritmo o tablas de datos, amplían la complejidad del proyecto, promoviendo un enfoque integral en la implementación de soluciones robustas.

Este trabajo, por lo tanto, no solo se presenta como un desafío técnico, sino como una oportunidad para fusionar los conocimientos teóricos con la habilidad práctica, brindando a los estudiantes una experiencia integral en el desarrollo de sistemas embebidos a nivel de hardware. La combinación de los conceptos de Arquitectura de Computadoras 1 con la implementación práctica en Assembly sitúa a los participantes en la vanguardia de la programación de sistemas embebidos.

Objetivo

Este proyecto tiene como propósito la integración coherente y funcional de los ejercicios prácticos desarrollados en la materia. Los objetivos específicos son los siguientes:

1. Desarrollar un Menú Interactivo:
    - Implementar un menú que permita al usuario elegir entre cuatro secuencias de luces distintas.
    - Incluir dos secuencias predefinidas, "Auto fantástico" y "Choque", desarrolladas respectivamente mediante algoritmo y tabla de datos.
    - Facilitar la creación de las otras dos secuencias, siguiendo el formato establecido por las secuencias predefinidas.
2. Simulación en IDE Quartus 1.19:
    - Integrar el menú y las cuatro secuencias en el simulador del IDE Quartus 1.19 para verificar su funcionalidad y corrección.
3. Programación en Placa DE0-Nano:
    - Cargar el código en hexadecimal generado por el proyecto en la placa "DE0-Nano".
    - Ejecutar el programa en la placa para asegurar su desempeño en un entorno de hardware real.
4. Control de Velocidad de Secuencias:
    - Implementar una opción que permita al usuario ajustar la velocidad de ejecución de cada secuencia.
    - Habilitar la posibilidad de incrementar o reducir la velocidad presionando las teclas correspondientes.
5. Implementación de Control de Acceso:
    - Integrar un sistema de control de acceso mediante password al menú principal del sistema.
    - Desarrollar la funcionalidad para mostrar asteriscos mientras se ingresa el password y comparar con una clave predefinida.
    - Abortar el programa después de tres intentos fallidos de ingreso.
6. Interfaz Gráfica Dinámica:
    - Cambiar dinámicamente la pantalla cada vez que el usuario selecciona una secuencia, indicando claramente cuál secuencia está en ejecución.
    - Implementar la capacidad de abandonar una secuencia en curso y regresar al menú principal, apagando todas las luces asociadas a la secuencia.
7. Mantenimiento de Velocidades entre Secuencias:
    - Conservar la velocidad ajustada por el usuario cuando cambia entre diferentes secuencias, garantizando una transición fluida y coherente.

Estos objetivos se alinean con la finalidad del proyecto de aplicar los conocimientos teóricos en un contexto práctico, proporcionando a los estudiantes una experiencia integral en el desarrollo de sistemas embebidos mediante la manipulación de la placa DE0-Nano y el uso del lenguaje de programación Assembly.

Desarrollo

A continuación, vamos a explicar que contiene el trabajo y como fue desarrollado. A vez se va a mostrar su funcionamiento.

Primero vamos a mostrar como se han indicado los Pines:

- *INport*: A continuación, se detalla que pines corresponden a cada señal de entrada *INport* del módulo top:

1	1	1	E1	M15	B9	T8	M1
[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]

  Como se puede observar los tres bits más significativos no han sido asignados a ningún pin, lo que implica que, por defecto, su valor será 1.

- *OUTport*: A continuación, se detalla que pines corresponden a cada señal de entrada *OUTport* del módulo top:

L3	B1	F3	D1	A11	B13	A13	A15

- *resetLow*: Para esta señal utilizamos el botón key_0 asignado al pin J15.

- *clk*: Para esta señal utilizamos el pin R8 asignado al clock de 50 MHz propio de la placa.

Ahora veamos cómo funciona la placa:

Dado que los pines E1 y J15, correspondientes a los botones key_0 y key_1 tienen funcionamiento por bajo (es decir, mientras no están presionados, la señal que emiten corresponde a un uno lógico, mientras que cuando son presionados, emiten un cero lógico) realizamos la siguiente corrección al código del ARM reducido implementado para llevar a cabo el proyecto.

```
6   module top(input  logic       clk, reset,
7              input  logic [7:0] INport,
8              output logic [7:0] OUTport);
9
10    logic [31:0] WriteData, DataAdr;
11    logic        MemWrite, MemtoReg, PortSel;
12    logic [31:0] PC, Instr, ReadData, MemData;
13    logic [7:0] INData;
14
```

```
6   module top(input  logic       clk, resetLow,
7              input  logic [7:0] INport,
8              output logic [7:0] OUTport);
9
10    logic [31:0] WriteData, DataAdr;
11    logic        MemWrite, MemtoReg, PortSel;
12    logic [31:0] PC, Instr, ReadData, MemData;
13    logic [7:0] INData;
14
15    logic reset;
16    assign reset = ~resetLow;
```

La corrección consiste en la negación de la señal externa del *clock* (*resetLow* en la imagen de la derecha) proveniente de la placa, valor que luego asignamos a una señal interna (*reset*) para sustituir la señal original.

Por último, vemos la lectura de datos por tabla y la ampliación de memoria:

Para poder implementar secuencias leídas desde tabla, tuvimos que insertar las siguientes líneas de código al ARM reducido. A su vez, debido al aumento de datos tuvimos que ampliar la RAM de RAM[63:0] paso a RAM[127:0]

```
30  module dmem(input  logic        clk, we,
31              input  logic [31:0] a, wd,
32              output logic [31:0] rd);
33
34    logic [31:0] RAM[127:0];
35
36  initial
37        $readmemh("C:\\Users\\Maxi\\Desktop\\Arqui1\\dmem_io.dat",RAM);
38
39
40    assign rd = RAM[a[31:2]]; // word aligned
41
42    always_ff @(posedge clk)
43      if (we) RAM[a[31:2]] <= wd;
44  endmodule
```

Algo similar sucede con el módulo imem, debido a la cantidad de instrucciones que necesitamos debimos aumentar la RAM de [63:0] paso a [511:0] de RAM

```
46  module imem(input  logic [31:0] a,
47              output logic [31:0] rd);
48
49    logic [31:0] RAM[511:0];
50
51    initial
52        $readmemh("C:\\Users\\Maxi\\Desktop\\Arqui1\\mem_io.dat",RAM);
53
54    assign rd = RAM[a[31:2]]; // word aligned
55  endmodule
```

Luego de indicar y mostrar cómo están inicializados los pines y los cambios que se realizaron al módulo top, ahora veremos cómo es el funcionamiento. Primero se mostrará como se realizó el password, luego el menú y finalmente las secuencias.

Password: para la realización de este se utilizaron los 4 dip - switches ([3][2][1][0]) prendidos (1111). En la siguiente imagen se puede observar la porción de código de este:

```
/-------------------------------------------------------------------------/
/----------------------------- Password ----------------------------------/
/-------------------------------------------------------------------------/
Inicio:
    SUB     R0, R15, R15    //00
    ADD     R2, R0, #0x01   // luces a prender
    ADD     R8, R0, #3   // contador de intentos

Intento:
    STR     R2, [R0, #0x800]   //3
    ADD     R2, R2, R2   // sumador para encender luces

    Vuelta:
    ADD     R4, R0, #2      // Movemos un 2 al registro R4, contador para el bucle de lectura  //5

Lectura:
    SUBS    R4, R4, #0x01   // Restamos uno al contador y actualizamos banderas    //6
    LDR     R1, [R0, #0x800]   // Leemos el puerto, guardando en R1 los valores de los switches y del key_1
    BNE     Lectura         // lectura #0x18

    ANDS    R6, R1, #0x10   // Verificamos pin E1 apagado (botón presionado)
    BEQ     Verificacion    // verificacion #0x30
    B       Vuelta          // Vuelta #0x14


Verificacion:
    ANDSEQ  R5, R1, #0x0F    // Primer máscara, verificamos switches encendidos    //12
    BEQ     MENU            // menu  #0x50
    SUBS    R8, R8, #1   // resto contador de intentos
    BNE     Intento         // Intento si el contador no esta en cero #0xC
    B       Fallo         // Si esta en cero fallo  #0x44

Fallo:
    ADD     R7, R0, #0xFF   // prende todas las luces  //17
    STR     R7, [R0, #0x800]
    B       Fallo           // fallo  #0x44

```

Menú: para este también se utilizan los 4 dip - switches ([3][2][1][0]) para seleccionar la secuencia que se quiere observar. Si seleccionamos el dip – switches [0] observaremos la secuencia 1, si se selecciona el dip – switches [1] podremos ver la secuencia 2, si se selecciona el dip – switches [2] se puede ver la secuencia 3 y si se selecciona el dip – switches [3] se puede observar la secuencia 4.

A continuación, se muestra una imagen del menú y como se seleccionan las secuencias:

```
38  /-------------------------------------------------------------------------/
39  /--------------------------------- Menu ----------------------------------/
40  /-------------------------------------------------------------------------/
41
42              SUB R1,R1,R1  //20
43              SUB R2,R2,R2
44              SUB R3,R3,R3
45
46  inicio:     ADD     R2, R0, #0x80       Movemos a R2 el valor #0x80, sera nuestra luz del menu   //23
47              STR     R2, [R0, #0x800]     Sacamos al puerto 0x800, correspondiente a la placa, el registro R2
48
49  vuelta:     ADD     R3, R0, #2           Movemos un 2 al registro R3, contador para el bucle de lectura  //25
50
51  lectura:    SUBS    R3, R3, #1           Restamos uno al contador y actualizamos banderas //26
52              LDR     R1, [R0, #0x800]     Leemos el puerto, guardando en R1 los valores de los switches y del key_1
53              BNE     lectura               #0x68
54
55              ANDS    R2, R1, #0x0F        Primer mascara, verificamos la presencia de switches encendidos
56              BEQ     vuelta              #0x64
57
58              ANDSNE  R2, R1, #0x01       Segunda mascara, verificamos pin M1 encendido
59              STRNE   R0, [R0, #0x800]     Apagamos luces del menu
60              BNE     sel1                 #0xB0
61
62              ANDSEQ  R2, R1, #0x02        Tercer mascara, verificamos pin T8 encendido
63              STRNE   R0, [R0, #0x800]     Apagamos luces del menu
64              BNE     sel2                 #0xE8
65
66              ANDSEQ  R2, R1, #0x04       Cuarta mascara, verificamos pin B9 encendido
67              STRNE   R0, [R0, #0x800]     Apagamos luces del menu
68              BNE     sel3                 #0x120
69
70              ANDSEQ  R2, R1, #0x08        Quinta mascara, verificamos pin M15 encendido
71              STRNE   R0, [R0, #0x800]     Apagamos luces del menu
72              BNE     sel4                 #0x158
73              B       menu                #0x50   Si ninguna condicion se cumplio, volvemos al inicio
74
```

```
75  /-------------------------------------------------------------------------/
76  /------------------------------ Seleccion 1 ------------------------------/
77  /-------------------------------------------------------------------------/
78
79              SUB R1,R1,R1  //44
80              SUB R2,R2,R2
81              SUB R3,R3,R3
82
83  sel1:       ADD     R2, R1, #0x01        Movemos un 1 a R2 //47
84              STR     R2, [R0, #0x800]     Sacamos al puerto la luz 1, correspondiente a la opcion elegida
85
86  vuelta:     ADD     R3, R0, #2           Movemos un 2 al registro R3, contador para el bucle de lectura  //49
87
88  lectura:    SUBS    R3, R3, #1           Restamos uno al contador y actualizamos banderas   //50
89              LDR     R1, [R0, #0x800]     Leemos el puerto, guardando en R1 los valores de los switches y del key_1
90              BNE     lectura              #0xC8
91
92              ANDS    R2,R1, #0x10         Primer mascara, verificamos pin E1 apagado (boton presionado)
93              BEQ     sec1                #0x190
94              ANDSNE  R2, R1, #0x01        Segunda mascara, verificamos pin M1 apagado
95              BEQ     menu                 #0x50
96              BNE     vuelta              #0XC4      Si el pin M1 sigue encendido, esperamos a E1 apagado o M1 apagado
97
```

```
98  /------------------------------------------------------------------------/
99  /------------------------------ Seleccion 2 -----------------------------/
100 /------------------------------------------------------------------------/
101
102             SUB R1,R1,R1  //58
103             SUB R2,R2,R2
104             SUB R3,R3,R3
105 sel2:       ADD     R2, R1, #0x02      Movemos un 2 a R2   //61
106             STR     R2, [R0, #0x800]    Sacamos al puerto la luz 2, correspondiente a la opcion elegida
107
108 vuelta:     ADD     R3, R0, #2          Movemos un 2 al registro R3, contador para el bucle de lectura  //63
109
110 lectura:    SUBS    R3, R3, #1          Restamos uno al contador y actualizamos banderas   //64
111             LDR     R1, [R0, #0x800]    Leemos el puerto, guardando en R1 los valores de los switches y del key_1
112             BNE     lectura             #0x100
113
114             ANDS    R2, R1, #0x10       Primer mascara, verificamos pin E1 apagado (boton presionado)
115             BEQ     sec2                #0x1F4
116             ANDSNE  R2, R1, #0x02       Segunda mascara, verificamos pin T8 apagado
117             BEQ     menu                #0x50
118             BNE     vuelta              #0xFC Si el pin M1 sigue encendido, esperamos a E1 apagado o M1 apagado
119
```

```
120 /------------------------------------------------------------------------/
121 /------------------------------ Seleccion 3 -----------------------------/
122 /------------------------------------------------------------------------/
123
124             SUB R1,R1,R1   //72
125             SUB R2,R2,R2
126             SUB R3,R3,R3
127 sel3:       ADD     R2, R1, #0x04       Movemos un 4 a R2   //75
128             STR     R2, [R0, #0x800]    Sacamos al puerto la luz 3, correspondiente a la opcion elegida
129
130 vuelta:     ADD     R3, R0, #2         Movemos un 2 al registro R3, contador para el bucle de lectura  //77
131
132 lectura:    SUBS    R3, R3, #1         Restamos uno al contador y actualizamos banderas  //78
133             LDR     R1, [R0, #0x800]   Leemos el puerto, guardando en R1 los valores de los switches y del key_1
134             BNE     lectura             #0x138
135
136             ANDS    R2, R1, #0x10      Primer mascara, verificamos pin E1 apagado (boton presionado)
137             BEQ     sec3                #0x2B0
138
139             ANDSNE  R2, R1, #0x04      Segunda mascara, verificamos pin B9 apagado
140             BEQ     menu                #0x50
141             BNE     vuelta              #0x134  Si el pin M1 sigue encendido, esperamos a E1 apagado o M1 apagado
142
```

```
143   /------------------------------------------------------------------------/
144   /------------------------------ Seleccion 4 -----------------------------/
145   /------------------------------------------------------------------------/
146
147             SUB R1,R1,R1  //86
148             SUB R2,R2,R2
149             SUB R3,R3,R3
150   sel4:     ADD     R2, R1, #0x08      Movemos un 8 a R2   //89
151             STR     R2, [R0, #0x800]    Sacamos al puerto la luz 4, correspondiente a la opcion elegida
152
153   vuelta:   ADD     R3, R0, #2          Movemos un 2 al registro R3, contador para el bucle de lectura   //91
154
155   lectura:  SUBS    R3, R3, #1         Restamos uno al contador y actualizamos banderas   //92
156             LDR     R1, [R0, #0x800]   Leemos el puerto, guardando en R1 los valores de los switches y del key_1
157             BNE     lectura             #0x170
158
159             ANDS    R2, R1, #0x10       Primer mascara, verificamos pin E1 apagado (boton presionado)
160             BEQ     sec4                #0x310
161
162             ANDSNE  R2, R1, #0x08      Segunda mascara, verificamos pin M15 apagado
163             BEQ     menu                #0x50
164             BNE     vuelta              #0x16C Si el pin M1 sigue encendido, esperamos a E1 apagado o M1 apagado
165
```

Las líneas de código que aparecen antes del rotulo que indica la selección de la opción muestran los registros que se utilizaron en el password inicializados en cero, esto nos permite que en caso de que el registro tenga almacenado algún otro dato no altere el funcionamiento del resto del programa.

Secuencio 1: LA CARRERA

A continuación, se muestra el código con la ilustración de la secuencia

```
1   /*  Secuencia "El Choque"   */
2   /* Realizada por tabla*/
3
4   00000001  -> - - - - - - - *
5   00000001  -> - - - - - - - *
6   00000002  -> - - - - - - * -
7   00000002  -> - - - - - - * -
8   00000004  -> - - - - - * - -
9   00000004  -> - - - - - * - -
10  00000008  -> - - - - * - - -
11  00000008  -> - - - - * - - -
12  00000011  -> - - - * - - - *
13  00000012  -> - - - * - - * -
14  00000024  -> - - * - - * - -
15  00000028  -> - - * - * - - -
16  00000050  -> - * - * - - - -
17  00000060  -> - * * - - - - -
18  000000C0  -> * * - - - - - -
19  00000080  -> * - - - - - - -
```

```
161  /-------------------------------------------------------------------------/
162  /------------------------------ Secuencia 1 ------------------------------/
163  /-------------------------------------------------------------------------/
164
165              SUB R1,R1.R1
166              SUB R5,R5,R5
167              SUB R7,R7,r7
168              SUB R8,R8,R8
169              SUB R6,R6,R6
170  sec1:       LDR r5, [r0, #4]           ; E5905004 ; Cargamos el valor de tabla del delay
171              ADD r7, r0, #12            ; E2807008 ; Guardamos el valor de la primer palabra de la secuencia
172              ADD r8,r0,#16               ; E2808008 ; Contador de palabras de la tabla
173
174  loop1:      LDR r9, [r7]               ; E5979000 ; Lee el dato guardado en R7 (led a prender)
175              STR r9, [r0, #0x800]       ; E5809800 ; Saca la luz al puerto
176              ADD r6,r0,r5               ; E0806005 ; Guardamos en r6 el valor del delay en r5
177
178  delay:      SUBS r6, r6, #1            ; E2566001 ; Restamos hasta llegar a cero
179              BNE #0x120                 ; 1AFFFFFD ;
180
181              ADD r7, r7, #4             ; E2877004 ; Aumentamos en 4 para pasar a la proxima palabra de tabla
182              SUBS r8,r8,#1              ; E2588001 ; Restamos 1 al contador de instrucciones
183              ADDEQ r7,r0,#12            ; 02807008 ; Si llega a cero, suma la cantidad de lineas de la tabla
184              ADDEQ r8,r8,#16             ; 02888008 ; Y vuelve a la primer palabra de la secuencia
185              ADD R3,R0,#0x2             ; E2803002 ; Contador de doble lectura
186
187  lectura:    SUBS R3,R3,#1              ; E2533001 ; Lectura del puerto
188              LDR R1,[R0,#0x800]         ; E5901800 ;
189              BNE lectura                ; 1AFFFFFC ;
190
191              ANDS R1,R1,#0x01           ; E2111001 ; Mascara: verifica que se haya apagado el pin_M1
192              BEQ inicio                 ; 0AFFFFAC ;
193              BNE loop1                  ; 1AFFFFEF ; Si no se apaga el PIN_M1 vuelve a la secuencia
194
```

Secuencio 2: Auto Fantástico

A continuación, se muestra el código con la ilustración de la secuencia

```
1    // Secuencia "Auto Fantastico" desarrollada por algoritmo
2
3    00000080 --> * - - - - - - -    00000002 --> - - - - - - * -
4    00000040 --> - * - - - - - -    00000004 --> - - - - - * - -
5    00000020 --> - - * - - - - -    00000008 --> - - - - * - - -
6    00000010 --> - - - * - - - -    00000010 --> - - - * - - - -
7    00000008 --> - - - - * - - -    00000020 --> - - * - - - - -
8    00000004 --> - - - - - * - -    00000040 --> - * - - - - - -
9    00000002 --> - - - - - - * -    00000080 --> * - - - - - - -
10   00000001 --> - - - - - - - *
```

```
/-------------------------------------------------------------------------/
/------------------------------ Secuencia 2 ------------------------------/
/-------------------------------------------------------------------------/

            SUB R1,R1.R1  //125
            SUB R5,R5,R5
            SUB R3,R3,R3
            SUB R4,R4,R4
            SUB R8,R8,R8
            SUB R6,R6,R6
            SUB R7,R7,R7
sec2:       ADD     R4, R0, #7          Movemos un 7 al registro R4, contador del loop1 //132
            ADD     R8, R0, #0           Movemos un 0 al registro R8, contador de luces encendidas
            ADD     R6, R0, #0x80        Movemos el valor 0x80 al registro R6, este sera nuestro registro de luces

loop1:      STR     R6, [R0, #0x800]     Sacamos la luz de la secuencia al puerto   //135
            ADD     R5, R0, #6           Movemos un 6 al registro R5, contador de la division
            ADD     R7, R0, #1          Movemos un 1 al registro R7, acumulador para division
            SUBS    R5, R5, R8           Corregimos R5 segun la cantidad de luces que hayamos encendido y actualiza banderas
            BEQ     salto  #0x23C        En caso de ser 0, debemos evitar el loop division

division:   SUBS    R5, R5, #1          Restamos uno al contador y actualizamos banderas  //140
            ADD     R7, R7, R7           Acumulamos R7
            BNE     division  #0x230

salto:      ADD     R8, R8, #1           Sumamos 1 al registro R8, correspondiente a la luz que sacamos   //143
            SUB     R6, R6, R7          Division por 2 de R6
            LDR     R9, [R0, #0x4]      Lectura del delay desde tabla, guardamos en el registro R9

delay       SUBS    R9, R9, #1          Restamos uno al contador y actualizamos banderas  //146
            BNE     delay #0x248

            SUBS    R4, R4, #1         Restamos uno al contador del loop1
            BNE     loop1   #0x21C

            STR     R6, [R0, 0x800]      Sacamos la ultima luz de la primera mitad de la secuencia (0x01)
            ADD     R4, R0, #7          Reseteamos el contador del loop

            ADD     R3,R0,#0x2          Movemos un 2 al registro R2, contador para el bucle de lectura

 lectura:    SUBS    R3,R3,#1              Restamos uno al contador y actualizamos banderas   //153
             LDR     R1,[R0,#0x800]       Leemos el puerto, guardando en R1 los valores de los switches y del key_1
             BNE     lectura  #0x264

             ANDS    R1,R1,#0x02         Primer mascara, verificamos que el pin T8 se haya apagado
             BEQ     menu #0x50

 loop2:      ADD     R6, R6, R6           Multiplicamos R6 por 2  //158
             LDR     R9, [R0, #0x4]       Lectura del delay desde tabla, guardamos en el registro R9

 delay       SUBS    R9, R9, #1          Restamos uno al contador y actualizamos banderas   //160
             BNE     delay #0x280

             STR     R6, [R0, 0x800]      Sacamos la luz al puerto
             SUBS    R4, R4, #1           Restamos uno al contador del loop2
             BNE     loop2  #0x278

             ADD     R3,R0,#0x2          Movemos un 2 al registro R2, contador para el bucle de lectura

 lectura:    SUBS    R3,R3,#1             Restamos uno al contador y actualizamos banderas  //166
             LDR     R1,[R0,#0x800]       Leemos el puerto, guardando en R1 los valores de los switches y del key_1
             BNE     lectura  #0x298

             ANDS    R1,R1,#0x02          Segunda mascara, verificamos que el pin T8 se haya apagado
             BEQ     menu   #0x50
             BNE     sec2   #0x1F4             De no haberse apagado el pin T8, se reinicia la secuencia
```

Secuencio 3: Arbolito de Navidad

A continuación, se muestra el código con la ilustración de la secuencia

```
/*  Secuencia "Arbol de navidad"   */
/* Realizada por tabla*/

10101010  -> * - * - * - * -
10101010  -> * - * - * - * -
01010101  -> - * - * - * - *
01010101  -> - * - * - * - *
00000000  -> - - - - - - - -
10000000  -> * - - - - - - -
11000000  -> * * - - - - - -
11100000  -> * * * - - - - -
11110000  -> * * * * - - - -
11111000  -> * * * * * - - -
11111100  -> * * * * * * - -
11111110  -> * * * * * * * -
11111111  -> * * * * * * * *
01111111  -> - * * * * * * *
00111111  -> - - * * * * * *
00011111  -> - - - * * * * *
00001111  -> - - - - * * * *
00000111  -> - - - - - * * *
00000011  -> - - - - - - * *
00000001  -> - - - - - - - *
00000000  -> - - - - - - - -
```

```
268     /-------------------------------------------------------------------------/
269     /------------------------------ Secuencia 3 ------------------------------/
270     /-------------------------------------------------------------------------/
271
272                 SUB R1,R1.R1  //172
273                 SUB R5,R5,R5
274                 SUB R3,R3,R3
275                 SUB R7,R7,r7
276                 SUB R4,R4,R4
277                 SUB R9,R9,R9
278     sec3:       ADD r4, r0, #58          Contador de palabras de la tabla //178
279                 ADD r5, r0, #72            Guardamos el valor de la primer palabra de la secuencia
280
281     loop1:      LDR r2, [r5]               Lee el dato de la tabla guardado en r2  //180
282                 STR r2, [r0, #0x800]       Sacamos la luz al puerto del LED
283                 LDR r9, [r0, #4]           Cargamos en el registro el valor de tabla del delay
284
285     delay:      SUBS r9, r9, #1            Resta 1 hasta llegar a cero  //183
286                 BNE delay  #0x2DC
287
288                 ADD r5, r5, #4              Aumentamos en 4 para pasar a la proxima palabra de tabla
289                 SUBS r4, r4, #1             Restamos 1 al contador de instrucciones
290                 ADDEQ r4,r0,#58           Si llega a cero, suma la cantidad de lineas de la tabla
291                 ADDEQ r5,r0,#72             Y vuelve a la primer palabra de la secuencia
292                 ADD R3,R0,#0x2              Contador de doble lectura
293
294     lectura:    SUBS R3,R3,#1               Lectura del puerto  //190
295                 LDR  R1,[R0,#0x800]
296                 BNE  lectura   #0x2F8
297
298                 ANDS R1,R1,#0x04            Mascara: verifica que se haya apagado el pin_B9
299                 BEQ menu   #0x50
300                 BNE loop1   #0x2D0              Si no se apaga el PIN_B9 vuelve a la secuencia
301
```

Secuencio 4: Meteorito

A continuación, se muestra el código con la ilustración de la secuencia

```
1   // Secuencia "Meteorito" desarrollada por algoritmo
2
3   10000000 --> * - - - - - - -
4   01000000 --> - * - - - - - -
5   00100000 --> - - * - - - - -
6   00010000 --> - - - * - - - -
7   00001000 --> - - - - * - - -
8   00010100 --> - - - * - * - -
9   00100010 --> - - * - - - * -
10  01000001 --> - * - - - - - *
11  00000000 --> - - - - - - - -
```

```
/------------------------------------------------------------------------/
/------------------------------ Secuencia 4 -----------------------------/
/------------------------------------------------------------------------/

            SUB R1,R1,R1  //196
            SUB R3,R3,R3
            SUB R4,R4,R4
            SUB R5,R5,R5
            SUB R6,R6,R6
            SUB R7,R7,R7
            SUB R8,R8,R8
            SUB R0,R0,R0
            SUB R0,R0,R0

sec4:       ADD     R4, R0, #5          ; E2804005 ; Movemos un 5 al registro R5, contador del loop1   //205
            ADD     R8, R0, #0          ; E2808000 ; Movemos un 0 al registro R8, contador de luces encendidas
            ADD     R6, R0, #0x80       ; E2806080 ; Movemos el valor 0x80 al registro R6, sera nuestro registro de luces

loop1:      STR     R6, [R0, #0x800]    ; E5806800 ; Sacamos luz al puerto  //208
            ADD     R5, R0, #6          ; E2805006 ; Movemos un 6 al registro R5, contador de la division
            ADD     R7, R0, #1          ; E2807001 ; Movemos un 1 al registro R7, acumulador de la division
            SUB     R5, R5, R8          ; E0455008 ; Corregimos R5 segun la cantidad de luces que hemos encendido

division:   SUBS    R5, R5, #1          ; E2555001 ; Restamos uno al contador y actualizamos banderas   //212
            ADD     R7, R7, R7          ; E0877007 ; Acumulamos R7
            BNE     division     #0x350       ; 1AFFFFFC ;

            ADD     R8, R8, #1          ; E2888001 ; Sumamos uno a R8, correspondiente a la luz encendida
            SUB     R6, R6, R7          ; E0466007 ; Division de R6 por 2
            LDR     R9, [R0, #0x4]      ; E5909004 ; Lectura del delay desde tabla, guardamos en el registro R9

delay       SUBS    R9, R9, #1          ; E2599001 ; Restamos uno al contador y actualizamos banderas    //218
            BNE     delay     #0x368          ; 1AFFFFFD ;

            SUBS    R4, R4, #1          ; E2544001 ; Restamos uno al contador del loop y actualizamos banderas
            BNE     loop1               #0x340; 1AFFFFF1 ;

            ADD     R6, R0, #0x08       ; E2806008 ; Movemos a R6 el valor 0x08, correspondiente al cuarto led menos significativo
            ADD     R4, R0, #3          ; E2804003 ; Movemos un 3 al registro R4, contador del loop2
            ADD     R10, R6, R6         ; E086A006 ; Multiplicamos R6 por 2 y lo guardamos en R10
            ADD     R8, R0, #0          ; E2808000 ; Volvemos R8 a 0, cumplira el mismo proposito
            ADD     R3,R0,#0x2          ; E2803002 ; Movemos un 2 al registro R3, contador para el bucle de lectura

lectura:    SUBS    R3,R3,#1            ; E2533001 ; Restamos uno al contador y actualizamos banderas    //227
            LDR     R1,[R0,#0x800]      ; E5901800 ; Leemos el puerto, guardando en R1 los valores de los switches y del key_1
            BNE     lectura      #0x38c       ; 1AFFFFFC ;

            ANDS    R1,R1,#0x08         ; E2111008 ; Primer mascara, verificamos que el pin_M15 se haya apagado
            BEQ     menu      #0x50        ; 0AFFFF56 ;

loop2:      ADD     R5, R0, #2          ; E2805002 ; Movemos un 2 al registro R5, contador de la division    //232
            ADD     R7, R0, #1          ; E2807001 ; Movemos un 1 al registro R7, acumulador de la division
            SUBS    R5, R5, R8          ; E0555008 ; Corregimos R5 segun la cantidad de luces que hayamos encendido
            BEQ     salto   #0x3BC            ; 0A000002 ; En caso de ser 0, debemos evitar el loop division

division:   SUBS    R5, R5, #1          ; E2555001 ; Restamos uno al contador y actualizamos banderas     //236
            ADD     R7, R7, R7          ; E0877007 ; Acumulamos R7
            BNE     division    #0x3B0        ; 1AFFFFFC ;

            ADD     R8, R8, #1          ; E2888001 ; Sumamos uno a R8, correspondiente a la luz encendida
            SUB     R6, R6, R7          ; E0466007 ; Division de R6 por 2
            ORR     R11, R10, R6        ; E18AB006 ; Mediante una OR logica, unimos las 2 señales para formar una sola
            STR     R11, [R0, 0x800]    ; E580B800 ; Sacamos luz al puerto
            ADD     R10, R10, R10       ; E08AA00A ; Multiplicamos R10 por 2
            LDR     R9, [R0, #0x8]      ; E5909004 ; Lectura del delay desde tabla, guardamos en el registro R9

delay       SUBS    R9, R9, #1          ; E2599001 ; Restamos uno al contador y actualizamos banderas    //245
            BNE     delay  #0x3D4             ; 1AFFFFFD ;

            SUBS    R4, R4, #1          ; E2544001 ; Restamos uno al contador del loop
            BNE     loop2    #0x3A0           ; 1AFFFFEE ;

            ADD     R6, R0, #0x00       ; E2806000 ; Movemos un 0 al registro R6, correspondiente a la ultima luz de la secuencia
            STR     R6, [R0, 0x800]     ; E5806800 ; Sacamos luz al puerto

            LDR     R9, [R0, #0x8]      ; E5909004 ; Lectura del delay desde tabla, guardamos en el registro R9

delay       SUBS    R9, R9, #1          ; E2599001 ; Restamos uno al contador y actualizamos banderas     //252
            BNE     delay   #0x3F0            ; 1AFFFFFD ;

            ADD     R3,R0,#0x2          ; E2803002 ; Movemos un 2 al registro R3, contador para el bucle de lectura

lectura:    SUBS    R3,R3,#1            ; E2533001 ; Restamos uno al contador y actualizamos banderas    //255
            LDR     R1,[R0,#0x800]      ; E5901800 ; Leemos el puerto, guardando en R1 los valores de los switches y del key_1
            BNE     lectura   #0x3FC          ; 1AFFFFFC ;

            ANDS    R1,R1,#0x08         ; E2111008 ; Segunda mascara, verificamos que el pin_M15 se haya apagado
            BEQ     menu    #0x50          ; 0AFFFF3D ;
            BNE     sec4 #0x310               ; 1AFFFFCA ; De no haberse apagado, repetimos la secuencia
```

Conclusión

Con este trabajo podemos concluir que hemos tenido una muy cercana percepción de lo que es programar en bajo nivel, lo cual requiere suma atención y responsabilidad con todo lo que se hace ya que cualquier mínimo error puede provocar que nada funcione. Además, se nos agregó la dificultad de solo poder utilizar únicamente 7 operaciones de una ARM reducida, pero finalmente con muchas horas de intentos tras intentos pudimos lograr cumplir con casi todos los objetivos planteados al comienzo de este trabajo.