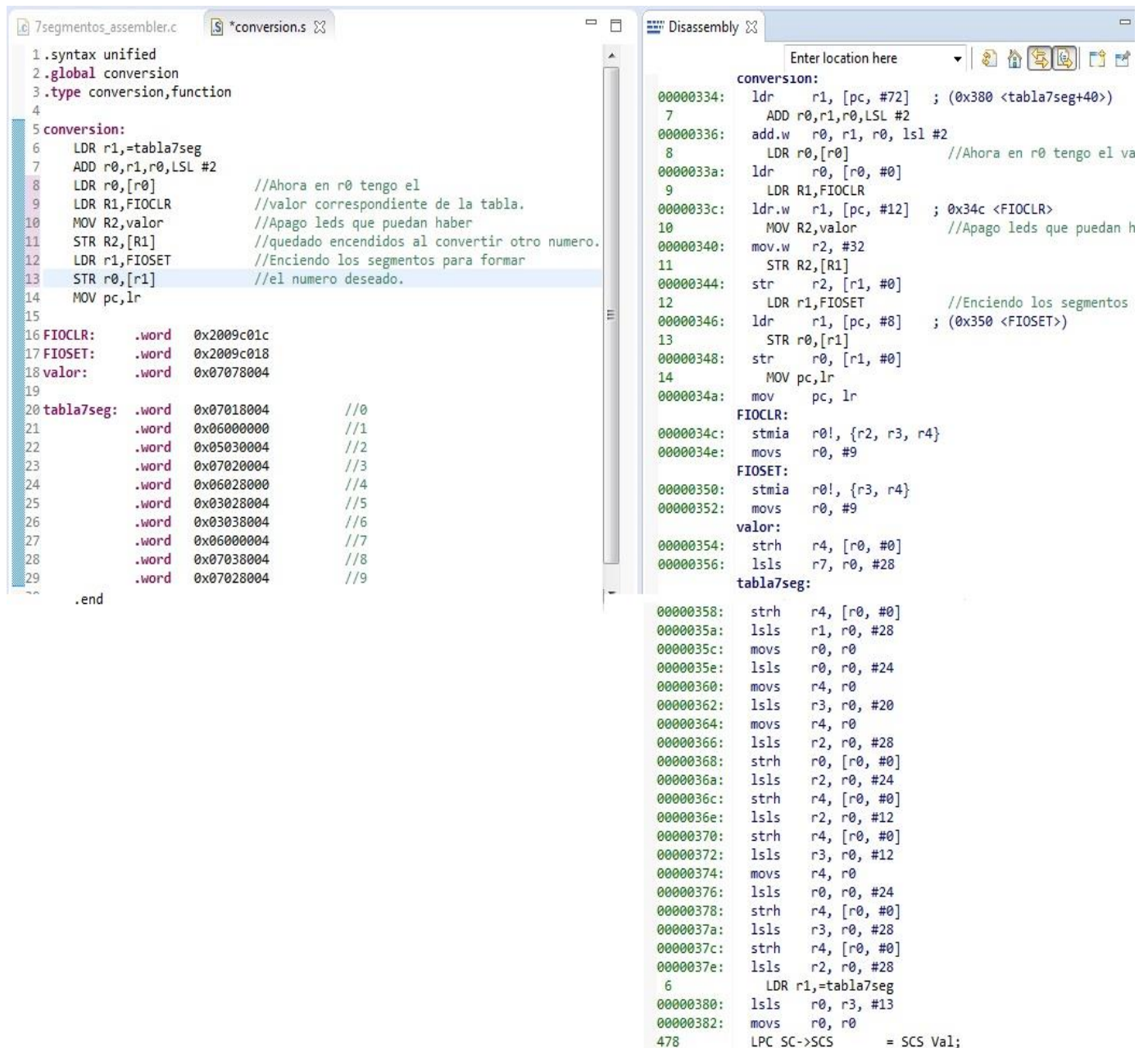


## Tarea clase 12/10/16

Alumno:

DNI:

**Consigna:** Redactar un programa que convierta un numero decimal del 0 al 9 de manera tal que dicho numero sea representado mediante un display de 7 segmentos. Dicha conversión se debe realizar mediante una rutina en assembler.



```
1.syntax unified
2.global conversion
3.type conversion,function
4
5.conversion:
6    LDR r1,=tabla7seg
7    ADD r0,r1,r0,LSL #2
8    LDR r0,[r0]           //Ahora en r0 tengo el
9    LDR R1,FIOCLR         //valor correspondiente de la tabla.
10   MOV R2,valor          //Apago leds que puedan haber
11   STR R2,[R1]           //quedado encendidos al convertir otro numero.
12   LDR r1,FIOSET         //Enciendo los segmentos para formar
13   STR r0,[r1]          //el numero deseado.
14   MOV pc,lr
15
16.FIOCLR:    .word 0x2009c01c
17.FIOSET:    .word 0x2009c018
18.valor:     .word 0x07078004
19
20.tabla7seg:  .word 0x07018004    //0
21             .word 0x06000000    //1
22             .word 0x05030004    //2
23             .word 0x07020004    //3
24             .word 0x06028000    //4
25             .word 0x03028004    //5
26             .word 0x03038004    //6
27             .word 0x06000004    //7
28             .word 0x07038004    //8
29             .word 0x07028004    //9
30
31.end
```

```
conversion:
00000334: ldr    r1, [pc, #72] ; (0x380 <tabla7seg+40>)
7      ADD r0,r1,r0,LSL #2
00000336: add.w  r0, r1, r0, lsl #2
8      LDR r0,[r0]           //Ahora en r0 tengo el va
9      LDR R1,FIOCLR
0000033a: ldr    r0, [r0, #0]
10     LDR.w r1, [pc, #12] ; 0x34c <FIOCLR>
11     MOV R2,valor          //Apago leds que puedan h
00000340: mov.w  r2, #32
12     STR R2,[R1]
00000344: str    r2, [r1, #0]
13     LDR r1,FIOSET         //Enciendo los segmentos
00000346: ldr    r1, [pc, #8] ; (0x350 <FIOSET>)
14     STR r0,[r1]
00000348: str    r0, [r1, #0]
15     MOV pc,lr
0000034a: mov    pc, lr
FIOCLR:
0000034c: stmia  r0!, {r2, r3, r4}
0000034e: movs   r0, #9
FIOSET:
00000350: stmia  r0!, {r3, r4}
00000352: movs   r0, #9
valor:
00000354: strh   r4, [r0, #0]
00000356: lsls   r7, r0, #28
tabla7seg:
00000358: strh   r4, [r0, #0]
0000035a: lsls   r1, r0, #28
0000035c: movs   r0, r0
0000035e: lsls   r0, r0, #24
00000360: movs   r4, r0
00000362: lsls   r3, r0, #20
00000364: movs   r4, r0
00000366: lsls   r2, r0, #28
00000368: strh   r0, [r0, #0]
0000036a: lsls   r2, r0, #24
0000036c: strh   r4, [r0, #0]
0000036e: lsls   r2, r0, #12
00000370: strh   r4, [r0, #0]
00000372: lsls   r3, r0, #12
00000374: movs   r4, r0
00000376: lsls   r0, r0, #24
00000378: strh   r4, [r0, #0]
0000037a: lsls   r3, r0, #28
0000037c: strh   r4, [r0, #0]
0000037e: lsls   r2, r0, #28
6      LDR r1,=tabla7seg
00000380: lsls   r0, r3, #13
00000382: movs   r0, r0
478    LPC_SC->SCS = SCS_Val;
```

En la imagen anterior se presentó la rutina de assembler desarrollada para la tarea solicitada. A continuación se adjunta otra imagen con la rutina en C correspondiente:

7segmentos\_assembler.c
conversion.s

```

30 Name      : 7segmentos_assembler.c
10
11 #ifndef __USE_CMSIS
12 #include "LPC17xx.h"
13 #endif
14 #include <cr_section_macros.h>
15
16 extern void conversion(int);
17 int main(void)
18 {
19     int numero=3; /* ACunidad a b c */
20     LPC_GPIO0->FIODIR |= (1<<21)|(1<<2)|(1<<26)|(1<<25);
21     /* d e f g dp */
22     LPC_GPIO0->FIODIR |= (1<<24)|(1<<16)|(1<<15)|(1<<17)|(1<<18); //Salidas Display
23     LPC_GPIO0->FIOCLR |= (1<<21); //Activo el enable del 7segmentos
24     while(1)
25     {
26         conversion(numero); //llamo a la funcion de assembler
27     } //que se encargara de la conversion a 7segmentos.
28     return 0 ;
29 }
30

```

Disassembly

```

Enter location here
main:
00000301: push    {r7, lr}
00000303: sub     sp, #8
00000305: add     r7, sp, #0
19      int numero=3; /* ACunidad a b
00000306: movs    r3, #3
00000308: str     r3, [r7, #4]
20      LPC_GPIO0->FIODIR |= (1<<21)|(1<<2)|(1<<2
0000030a: ldr     r2, [pc, #52] ; (0x340 <main+64>
0000030c: ldr     r3, [pc, #48] ; (0x340 <main+64>
0000030e: ldr     r3, [r3, #0]
00000310: orr.w   r3, r3, #102760448 ; 0x620000
00000314: orr.w   r3, r3, #4
00000318: str     r3, [r2, #0]
22      LPC_GPIO0->FIODIR |= (1<<24)|(1<<16)|(1<<
0000031a: ldr     r2, [pc, #36] ; (0x340 <main+64>
0000031c: ldr     r3, [pc, #32] ; (0x340 <main+64>
0000031e: ldr     r3, [r3, #0]
00000320: orr.w   r3, r3, #17170432 ; 0x106000
00000324: orr.w   r3, r3, #98304 ; 0x18000
00000328: str     r3, [r2, #0]
23      LPC_GPIO0->FIOCLR |= (1<<21); //Activo el
0000032a: ldr     r2, [pc, #20] ; (0x340 <main+64>
0000032c: ldr     r3, [pc, #16] ; (0x340 <main+64>
0000032e: ldr     r3, [r3, #28]
00000330: orr.w   r3, r3, #2097152 ; 0x200000
00000334: str     r3, [r2, #28]
26      conversion(numero); //Llamo a la fun
00000336: ldr     r0, [r7, #4]
00000338: bl      0x344 <conversion>
27      } //que se encarga
0000033c: b.n     0x336 <main+54>
0000033e: nop
00000340: stmia   r0!, {}
00000342: movs    r0, #9

```

Y por último se adjunta una fotografía del programa presentado siendo corrido en una LPC 1769/CD, el número a transformar en este caso fue el “3”:

