UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍA
DIVISION DE ELECTRONICA Y COMPUTACION

DEPARTAMENTO DE CIENCIAS COMPUTACIONALES

Programación de Bajo Nivel (IL358)

Reporte de Actividad Práctica

Práctica 4: "Conversiones entre tipos de datos ASCII y Binario"

Alumno: Torres Rivera Rodrigo
Código: 397423431

Sección: D03

Profesor: Jose Manuel Espinoza

Fecha 19 de octubre de 2025
firma de revisado

Requerimientos de la práctica:

- crea un programa que pida 2 números, cada número debe ser de 3 digitos
- que aplique una suma de dos números
- que sea capaz de imprimir un numero de 4 cifras

Desarrollo:

Sumar dos números en ensamblador es sencillo, basta con colocarlos de la memoria a registros y utilizar la instrucción de suma:

```
135     ; Realizar la suma
136     MOV AX, [num1]
137     ADD AX, [num2]
138     JC Overflow             ; Si hay acarreo, hay overflow
139     MOV [resultado], AX
```

El detalle es pedir los datos al usuario, digito a digito, lo cual se logra con el procedimiento siguiente:

```
26      ; Procedimiento para leer un número de hasta 3 dígitos (ASCII a Binario)
27      ; Retorna el número en AX
28      LeerNumero PROC
29      PUSH BX
30      PUSH CX
31      PUSH DX
32
33      XOR BX, BX          ; BX = 0 (acumulador del número)
34      MOV CX, 3           ; Máximo 3 dígitos
35
36      LeerDigito:
37      MOV AH, 01H         ; Leer carácter del teclado
38      INT 21H
39
40      CMP AL, 0DH         ; ¿Es Enter?
41      JE FinLectura       ; Si es Enter, terminar
42
43      CMP AL, '0'         ; Verificar si es dígito válido
44      JB LeerDigito       ; Si es menor que '0', ignorar
45      CMP AL, '9'
46      JA LeerDigito       ; Si es mayor que '9', ignorar
47
48      ; Convertir ASCII a número
49      SUB AL, 30H         ; AL = dígito numérico
50
51      ; BX = BX * 10 + AL
52      PUSH AX             ; Guardar AL
53      MOV AX, BX          ; AX = BX
54      MOV DX, 10
55      MUL DX              ; AX = AX * 10
56      MOV BX, AX          ; BX = AX
57      POP AX              ; Recuperar AL
58      XOR AH, AH          ; AH = 0
59      ADD BX, AX          ; BX = BX + AL
60
61      LOOP LeerDigito     ; Repetir hasta 3 dígitos
62
63      FinLectura:
64      MOV AX, BX          ; Retornar número en AX
65
66      POP DX
67      POP CX
68      POP BX
69      RET
70      LeerNumero ENDP
```

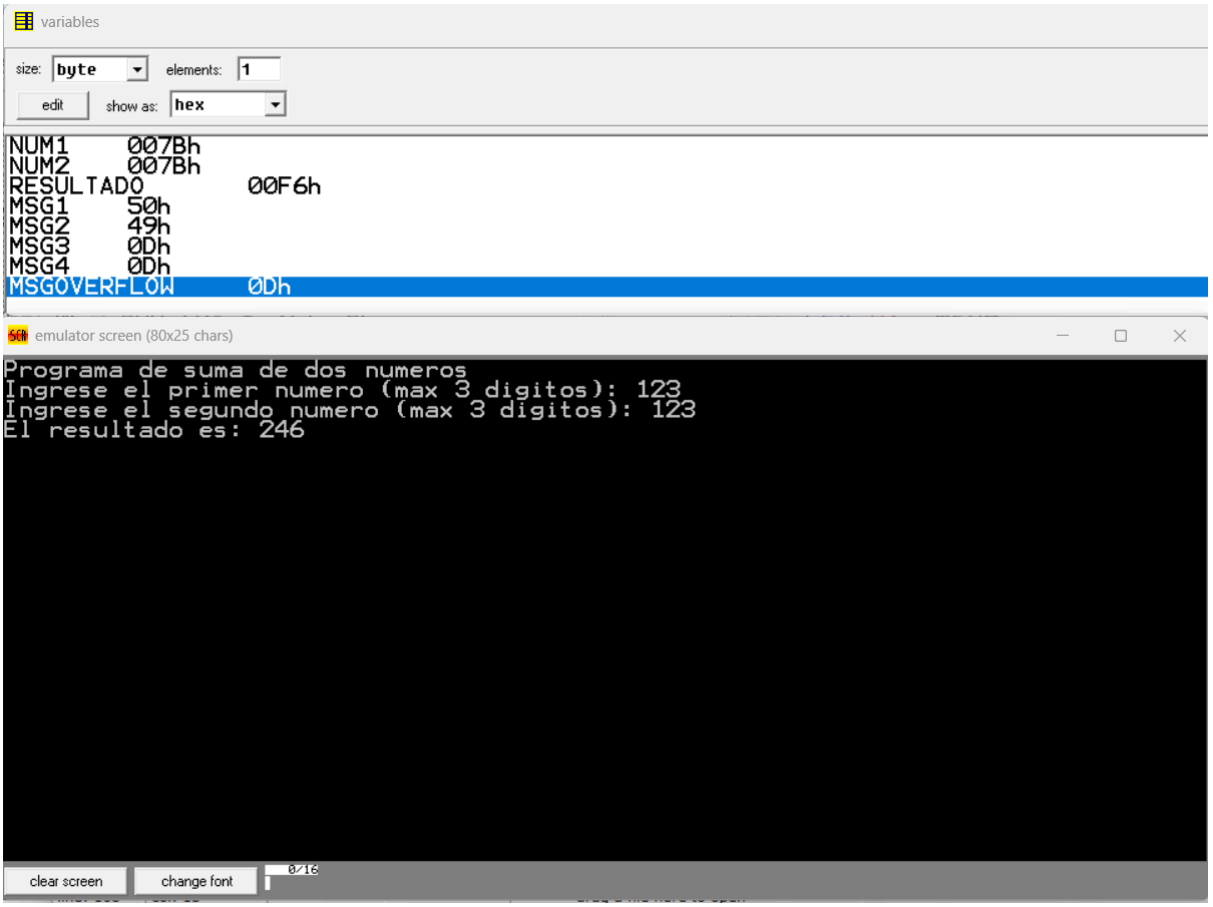
Donde hay que tener las consideraciones verificar si el dígito leído corresponde a un número en ASCII (líneas 43 al 36), después convertir el número de ASCII a Binario restando 30H (línea 49). Esto debe hacerse tres veces en un loop (línea 36 a 61), en este lazo se debe de multiplicar cada dígito ingresado por su correspondiente potencia de 10, para sumarlo al contenido del registro BX el cual contendrá el resultado parcial (líneas 52 a la 59). regresamos del procedimiento el resultado en AX para guardarlo en la variable num1(línea 126). el proceso se repite para el segundo número.

Tras realizar la suma entra el procedimiento

```
71
72      ; Procedimiento para mostrar un número (Binario a ASCII)
73      ; Recibe el número en AX
74      MostrarNumero PROC
75      PUSH AX
76      PUSH BX
77      PUSH CX
78      PUSH DX
79
80      XOR CX, CX          ; Contador de dígitos
81      MOV BX, 10          ; Divisor
82
83      ; Caso especial: si el número es 0
84      CMP AX, 0
85      JNE Dividir
86      PUSH AX
87      INC CX
88      JMP MostrarDigitos
89
90      Dividir:
91      XOR DX, DX          ; DX = 0
92      DIV BX              ; AX = AX / 10, DX = resto
93      PUSH DX             ; Guardar dígito en la pila
94      INC CX              ; Incrementar contador
95      CMP AX, 0           ; ¿Quedan más dígitos?
96      JNE Dividir         ; Si quedan, continuar
97
98      MostrarDigitos:
99      POP DX              ; Recuperar dígito
00      ADD DL, 30H         ; Convertir a ASCII
01      MOV AH, 02H         ; Mostrar carácter
02      INT 21H
03      LOOP MostrarDigitos ; Repetir para todos los dígitos
04
05      POP DX
06      POP CX
07      POP BX
08      POP AX
09      RET
10      MostrarNumero ENDP
```

Tomamos el número Binario y comenzamos a extraer cada dígito dividiendo entre 10, inverso a la conversión de ASCII a Binario y sumando 30H para convertirlo a su valor ASCII (lineas 90 al 96) , repetimos el proceso en un loop (línea 90 a 100). Debemos tener cuidado para el caso de que un número sea 0 (líneas 83 a 88), ya que la división por cero causa error y no será necesaria para la conversión del número.

Resultados Obtenidos:



El programa se comporta como lo esperamos. ingresamos dos números de no más de tres dígitos y nos regresa el resultado.

Conclusiones:

El ensamblador es un lenguaje de programación que requiere mucha atención al detalle y las operaciones que tiene a disposición son limitadas y muy centradas a tratar con dígitos binarios, para hacer algo útil hay que crear muchos procedimientos con antelación, que no son parte de la tarea a realizar, en este caso sumar dos números.